DIMENSION DECOUPLED REGION PROPOSAL NETWORK FOR OBJECT DETECTION

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

MEHMET CAN BAYTEKİN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

JANUARY 2023

DIMENSION DECOUPLED REGION PROPOSAL NETWORK FOR OBJECT DETECTION

submitted by **MEHMET CAN BAYTEKİN** in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems Department, Middle East Technical University** by,

Prof. Dr. Banu Günel Kılıç _____
Dean, **Graduate School of Informatics**

Prof. Dr. Altan Koçyiğit _____
Head of Department, **Information Systems**

Prof. Dr. Sevgi Özkan Yıldırım _____
Supervisor, **Information Systems, METU**

Assist. Prof. Dr. Yücel Çimtay _____
Co-supervisor, **Computer Engineering, TEDU**

Examining Committee Members:

Prof. Dr. Altan Koçyiğit
Information Systems, METU _____

Prof. Dr. Sevgi Özkan Yıldırım
Information Systems, METU _____

Assist. Prof. Dr. Venera Adanova
Computer Engineering, TEDU _____

Date: 19.01.2023

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: MEHMET CAN BAYTEKİN

Signature :

ABSTRACT

DIMENSION DECOUPLED REGION PROPOSAL NETWORK FOR OBJECT DETECTION

BAYTEKİN, MEHMET CAN
M.S., Department of Information Systems
Supervisor: Prof. Dr. Sevgi Özkan Yıldırım
Co-Supervisor: Assist. Prof. Dr. Yücel Çimtay

Jan 2023, 55 pages

In this work, we propose a dynamic anchor, dynamic assignment region proposal network named DA2RPN to improve the traditional Region Proposal Network (RPN). Classical Region Proposal Network places a set of pre-determined anchor boxes to generate object proposals which are later consumed by detection heads to produce final results. Problems of the Region Proposal Network are for each feature map point same anchor boxes are placed and static threshold values based on intersection-over-union scores are used to label anchor boxes. However, anchors mostly residing out of the image plane are not that useful to detect objects as objects only reside in the image plane. On the other hand, anchor box-ground truth box pairs having the same intersection-over-union values may not be equally useful for the detection of different objects. One anchor can be the best candidate for that object but another anchor can be a poor choice for another object even if they have the same iou values. To mitigate these problems, we generate a different number of anchors per feature map point and use a dynamic thresholding mechanism adaptive to the quality of selected anchors per ground-truth box. Additionally, to ease the training and to prevent sparsity caused by dynamic anchor generation we decouple anchor dimensions. Our results experimented on the COCO dataset show improvements over the Region Proposal Network.

Keywords: object detection, region proposal network, dynamic anchor box, dynamic assignment

ÖZ

NESNE TESPİTİ İÇİN BOYUTLARI AYRILMIŞ BÖLGE ÖNERİ AĞI

BAYTEKİN, MEHMET CAN
Yüksek Lisans, Bilişim Sistemleri Bölümü
Tez Yöneticisi: Prof. Dr. Sevgi Özkan Yıldırım
Ortak Tez Yöneticisi: Dr. Öğr. Üyesi. Yücel Çimtay

2023, 55 sayfa

Bu tez çalışmasında, derin öğrenme tabanlı nesne testip algoritmalarında sıklıkla kullanılan geleneksel Bölge Öneri Ağı'nı (RPN) geliştirmek için DA2RPN adlı dinamik anchoları ve dinamik atama bölgeleri üretebilen Bölge Öneri Ağı önermekteyiz. Klasik Bölge Öneri Ağı, nihai sonuçları üretmek için tespit katmanı tarafından girdi olarak alınan nesne önerilerini önceden belirlenmiş bir dizi bağlantı kutuları yerleştirerek üretir. Bölge Öneri Ağı bazı sorunlara sahiptir, bunlar her özellik haritası noktası için anchor kutularının yerleştirilmesi ve bağlantı kutularını etiketlemek için birleşim üzerinden kesişme puanlarına dayalı statik eşik değerlerinin kullanılmasıdır. Bununla birlikte, nesneler yalnızca görüntü düzleminde bulunduğundan, çoğunlukla görüntü düzleminin dışında bulunan çapalar, nesneleri algılamak için kullanışlı değildir. Öte yandan, aynı kesişme-birleşim değerlerine sahip bağlantı kutusu-zemin doğruluk kutusu çiftleri, farklı nesnelerin algılanması için eşit derecede yararlı olmayabilir. Bir çapa, o nesne için en iyi aday olabilir, ancak başka bir çapa, aynı iou değerlerine sahip olsalar bile, başka bir nesne için kötü bir seçim olabilir. Bu sorunları azaltmak için, özellik haritası noktası başına farklı sayıda çapa üretiyoruz ve yer gerçeği kutusu başına seçilen çapaların kalitesine uyarlanan dinamik bir eşikleme mekanizması kullanıyoruz. Ek olarak, eğitimi kolaylaştırmak ve dinamik bağlantı oluşturmanın neden olduğu seyrekliği önlemek için bağlantı boyutlarını ayırıyoruz. COCO veri seti üzerinde denediğimiz sonuçlarımız, Bölge Öneri Ağı üzerinde iyileştirmeler göstermiştir.

Anahtar Kelimeler: nesne tespiti, bölge öneri ağı, dinamik anchor kutusu, dinamik atama

To Mom, Dad and Sister

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to all those who have supported me throughout the journey of my thesis.

Firstly, I would like to express my appreciation to my advisor Prof. Dr. Sevgi Özkan Yıldırım, for taking the time to review my work and providing insightful feedback that has greatly enhanced the final outcome.

I would like to extend my sincere thanks to my thesis co-advisor Assist. Prof. Dr. Yücel Çimtay, for his invaluable guidance, encouragement, and support throughout the course of my research. Their expertise and unwavering support have been crucial in shaping my work and helping me to stay focused on my goals.

Finally, I would like to extend my special thanks to my family Şengül, Mustafa and Esra, for their unwavering support, love, and encouragement. Their belief in me has given me the strength to persevere through the toughest moments of my journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
GPU	Graphics Processing Unit
MS-COCO	Microsoft Common Objects in Context
VOC	Visual Object Classes
SVM	Support Vector Machine
KNN	K-Nearest Neighbor
VGG	Visual Geometry Group
SSD	Single Shot Detector
YOLO	You Look Only Once
IOU	Intersection Over Union
R-CNN	Regions with Convolutional Neural Network
RoI	Region of Interest
RPN	Region Proposal Network
FPN	Feature Pyramid Network
FCOS	Fully Convolutional One-Stage Object Detection
TP	True Positive
FP	True Positive
TN	True Negative
FN	False Negative
AP	Average Precision
AR	Average Recall
DA2PRN	Dimension Decoupled Region Proposal Network

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

Object detection is extremely popular and highly demanded topic in the defense and retail industry to reduce human workforce and human based errors. Due to advances of imaging technology, the images have high quality resolutions. However, traditional image processing techniques are not sufficient to interpret these images.

As a result of the high resolution of the images, the images have more features. The learning based algorithms can learn the patterns from these features and can produce meaningful results to meet the industry's needs. Deep Convolutional Neural Networks are designed to extract features from frames and to analyze the images in every aspect.

These extracted features are very rich in terms of object information in the image. However, using only Deep CNNs without proposed regions has reduced the accuracy of object detection. Region Proposal methods try to determine where objects can locate to produce better quality proposals. If the generated proposal has a higher chance to contain objects, the learning based method performs well in the training and test process.

Although with great improvements in the Deep Learning area, the models can still have problems with objects that have different aspect ratios. Due to region proposal methods using pre-determined fixed size anchor boxes, the generated object proposals cannot include all objects that are very long and wide.

In the real life images or videos, often objects are located in the center of the frame and the scene has a lot of different kinds of objects with complex backgrounds. Another problem with the region proposal methods is using a fixed number of anchor boxes for images' all pixels without separating cases if the pixel is located in corners or center. The produced anchor boxes from the corners of the image, mostly don't have the objects and unnecessarily consume the memory and power of the GPU.

To solve these problems that we mentioned above, we proposed a new method called Dimension Decoupled Region Proposal Network for Object Detection. The introduced model has dynamic anchor generation and dynamic label assignment modules. In literature, the former proposed region based methods use MS-COCO and Pascal

VOC datasets to evaluate their model's performance. To make a fair comparison, we evaluate our method with these datasets.

1.2 Scope of thesis

This thesis proposes a developed new method and will benefit the studies on this subject in 3 aspects. the first of these is dynamic anchor generation. In this way, different numbers of anchor boxes are proposed in different locations. the other one is label assignment, the model will eliminate negative examples and learn from positive examples. the third method aimed to obtain more accurate results by learning the widths and lengths of the anchors separately. As a result of this, also the model complexity is decreased.

1.3 The Outline of the Thesis

In Chapter 2, object detection subject is investigated and the state-of-the-art models are explained.

In Chapter 3 the method that we proposed Dimension Decoupled Region Proposal Network for Object Detection is explained in detail.

In Chapter 4 The results of proposed method are investigated and compared to other state-of-the-art models.

In Chapter 5, the conclusion is shared about proposed method.

CHAPTER 2

OBJECT DETECTION

2.1 Related Works on Object Detection

Identifying objects from images or frames of videos is a very common and highly attractive attention problem for many industries like defense, surveillance, retail and automated systems. To detect and classify objects, formerly traditional image processing techniques [25] [26][27][28] are applied. Apart from classical image processing methods [29][30][31][32] also neural networks [33][34][35][36] use for image classification task [37][38][39][40] and object detection task.

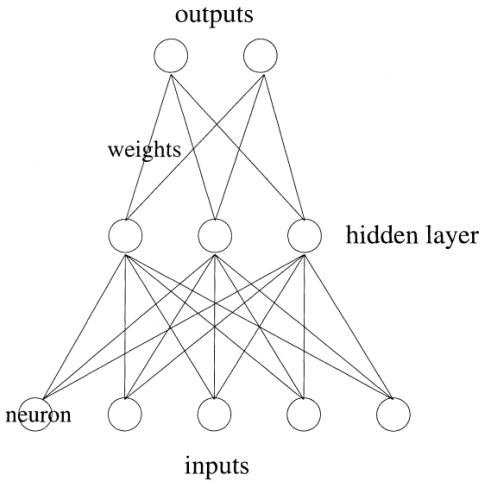

Figure 1: Architecture of basic neural network. Taken from [1].

In 1940s, the studies started on artificial neurons [41][42][43] for computational models [44] and in 1960s first many layers artificial neural network [45][46][47] proposed and first time used functionally for data handling [48][49][50].

Artificial neural networks, shown in Figure 1, could learn patterns [51] [52] and give information about images. Image processing methods were improving but still not

sufficient for classifying all objects on target frames. The importance of knowing which objects are in a certain image was increasing with a growing economy.

Independently for image processing methods, probabilistic [53][54] and learning based methods [55] are used for image classification. The problem can be solved in two general ways offered as supervised [56][57][58] and unsupervised learning [59][60][61] classification. In an unsupervised way, the data doesn't have labels, the algorithms calculate distances between vectors [62] and try to cluster the data.

In 1997, Paul Scheunders applied a clustering algorithm to an image in the article A genetic c-means clustering algorithm applied to color image quantization [63]. The most commonly used unsupervised algorithm is K-Means [64][65][66].

K-means is a tremendously simple and fast way to cluster the points in the space. All distances are calculated between elements and the as a result of this, the elements are assigned to the center points. Therefore, images which have very similar image primitives, belong to the same cluster and classification is done. An example result of the K-Means algorithm is shown in Figure 2.

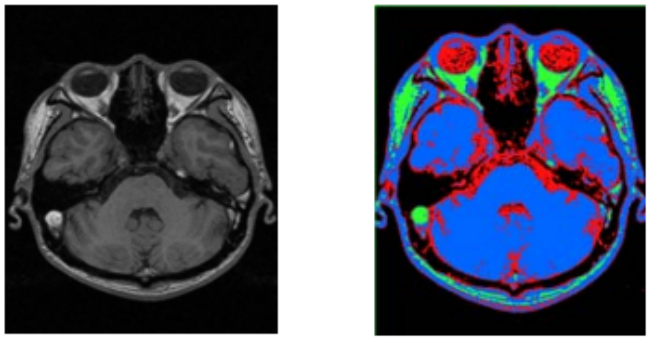

Figure 2: Basic K-Means algorithm example on medical image (a), after applied K-Means clustering (b). Taken from [2].

On the other hand, if the data is labeled, the labels and images feed the algorithm during the learning process in the supervised classification. Support Vector Machine [67][68] concept is a strong mathematical way to classify both linear and non-linear data. SVM turns the data into multi-dimensional space and calculates marginal distances between vectors. Therefore, the algorithm divides the data into classes with a hyper-plane, see in Figure 3. If the data is non-linear Radial Basis Function is applied to the algorithm and the solution is performed with SVM.

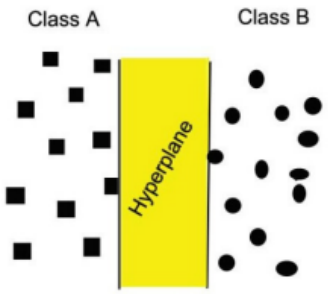

Figure 3: Binary classification example with SVM algorithm. Taken from [3].

In 1995, Cortes C. clarified the learning of Support Vector Machines and Survey on SVM and their application in image classification is detailed in this article. SVMs can work on very large and very small datasets and are also highly preferred algorithms in different domains as a result of their pace.

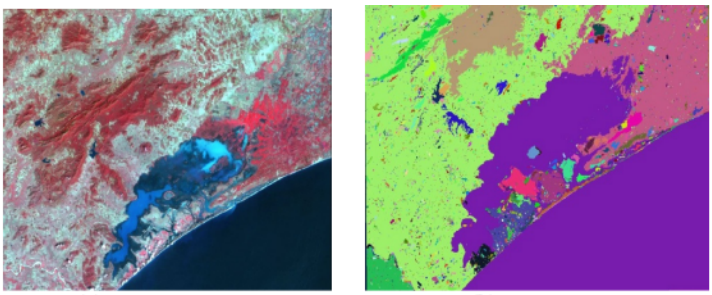

Figure 4: Multi-class classification with SVM (b) on multi-spectral image (a), Taken from [3].

Another efficient way to image classification performing K-Nearest Neighbor algorithm [69][70] that is a non-parametric statistical way to apply for classification and regression [71] problems. Since K-NN is distance based method, first of all, feature scaling is applied to the data and distance based processes are applied later. The main logic of the algorithm to calculate the distance between the neighbor's features and try to find the most similar classes. KNN and SVM can achieve success on small size datasets and are easy to apply with programming languages. KNNs are showing more accuracy compared to SVM on multi-class label problems. KNN classification example is shown in Figure 6.

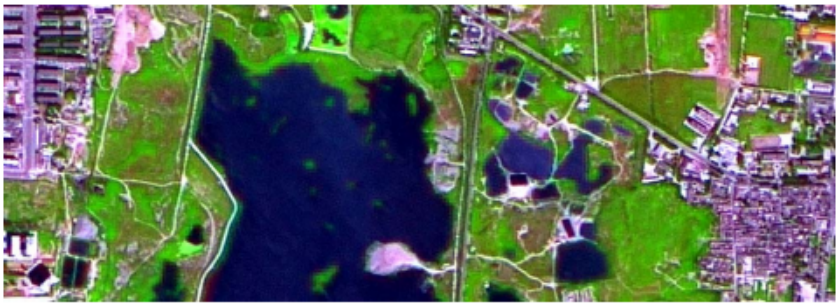

Figure 5: Original remote sensing image before K-NN algorithm applied, Taken from [4].

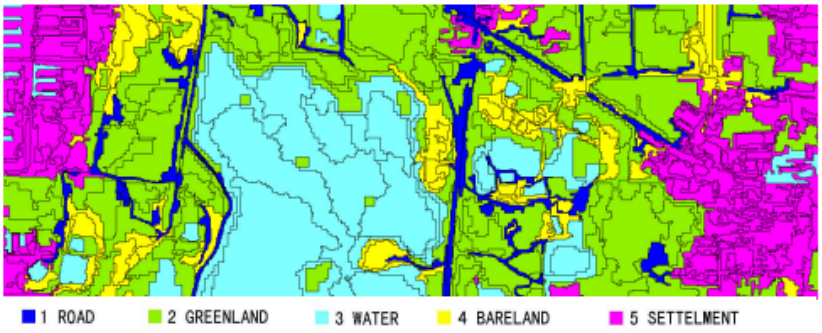

Figure 6: KNN classification results on Figure 2.6, Taken from [4].

In the 1980s, convolutional neural networks were proposed and in 1989 Yann Lecun used that method for handwritten digit recognition [72]. After this date, artificial neural networks gained momentum in the usage of classifying images.

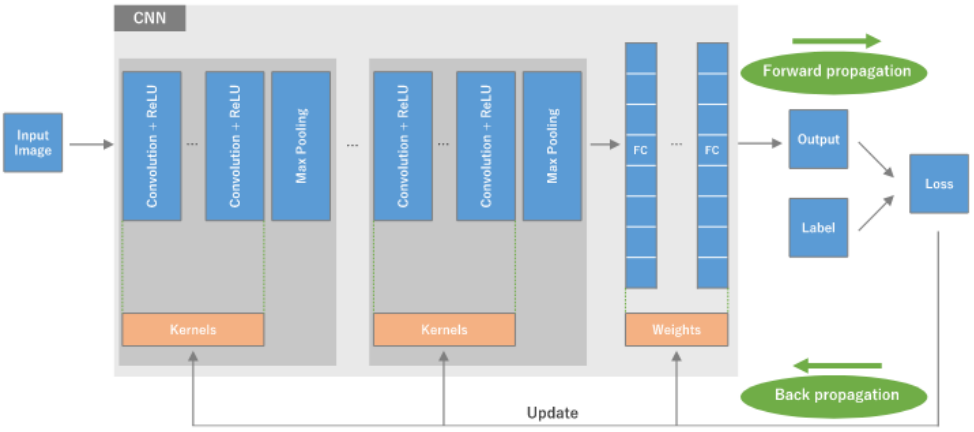

Figure 7: CNN Architecture, Taken from [5].

In CNN-based classification problems, features are extracted [73][74] from image applying layer operations to aim to increase the pixel's nonlinearity. After understanding of CNN is a powerful method to classify images, LeNet [75] is proposed by Yann Lecun for classifying handwritten letters. LeNet consists of 7 layers and accepts 32x32 pixel image, given in Figure 8. Layers have operations called average pooling [76] and activation function [77][78] for extracting features. However, this method is not much efficient.

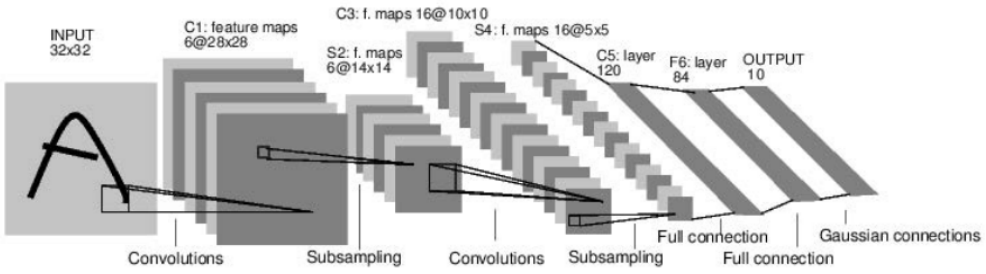

Figure 8: LeNet Architecture, Taken from [6].

With improvements on technology, codes started to work faster on Graphics Processing Units(GPUs). Thanks to GPUs, CNNs architectures are designed deeper and more efficiently.

In 2010s, AlexNet [79], ResNet [80] and VGG [81] methods are published and image classification problem performs huge improvements. AlexNet is shown in Figure 9.

ResNet and VGG networks have huge importance for convolutional neural networks, these algorithms are widely used as the backbone for the other neural networks. Two

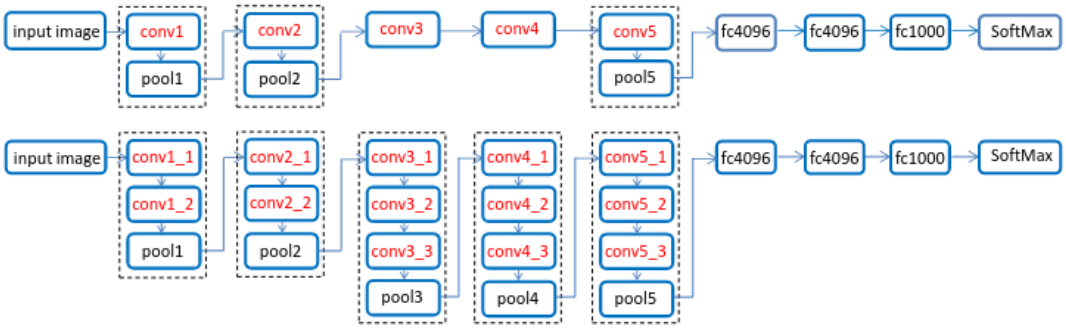

Figure 9: AlexNet Architecture, Taken from [7].

of them have many hidden layers to extract features. The architecture of these algorithms is given in Figure 10.

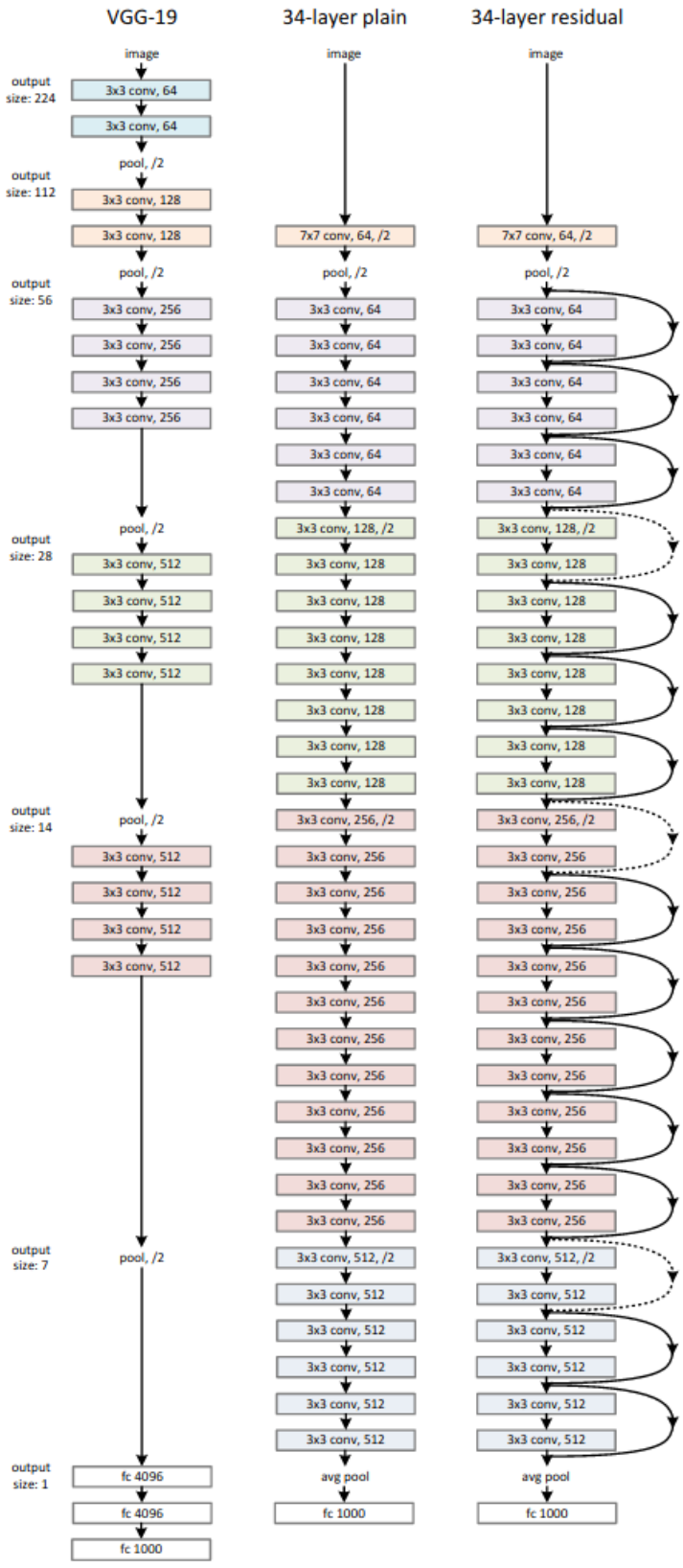

Figure 10: VGG-19 Architecture on the left, 34-layer CNN on the middle, ResNet on the right. Taken from [8].

After huge improvements in image classification, as another problem object detection started to solved easier. The main focus on object detection with CNNs [82] is first to classify the image and apply localization to find object's exact location. Then, classified features take as a input for other layers and localization solutions apply to them. The learning based object detection methods are grouped under two titles, One stage object detection and two stage object detection methods.

2.2 One Stage Object Detection Methods

One stage object detection models are aiming to find object's class and coordinates with just one passing on the neural network. There is no need to feature extraction or proposed regions formerly.

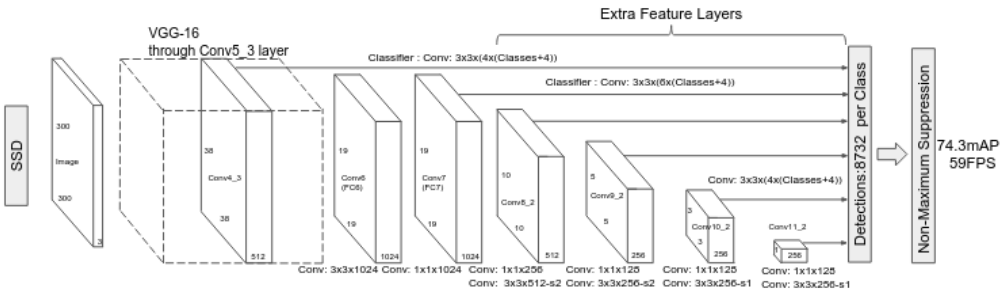

Figure 11: SSD Architecture. Taken from [9].

2.2.1 Single Shot Detector

In 2016, Single Shot MultiBox Detector(SSD) [9], a model is designed by Erhan which is able to detect objects in real time from video captures. SSD uses convolutional neural networks to only extract features. Different from the former applied models, SSD doesn't use the CNNs for generating anchor boxes [12][14]. Firstly, the image or frame is given to the backbone, which is mainly a CNN to use for feature extraction, and the backbone reduces the image's size and creates meaningful features for the other convolutional layers, the architecture is shown in Figure 11.

The SDD model consists of one backbone and 6 convolutional layers. After the first feature extraction, the convolutional layers are fed by feature maps [83][84]. Every convolutional layer takes the feature maps, which is the output of former convolutional layers, as input. The feature maps' sizes are getting smaller by layer to layer. To generate anchor boxes, default 4 boxes are assigned to every object on the image during training. The anchor boxes are shown in Figure 12.

At the end of the model, 7 feature maps are created by layers and backbone and the algorithm generates boxes on the different size feature maps. When the ground truth

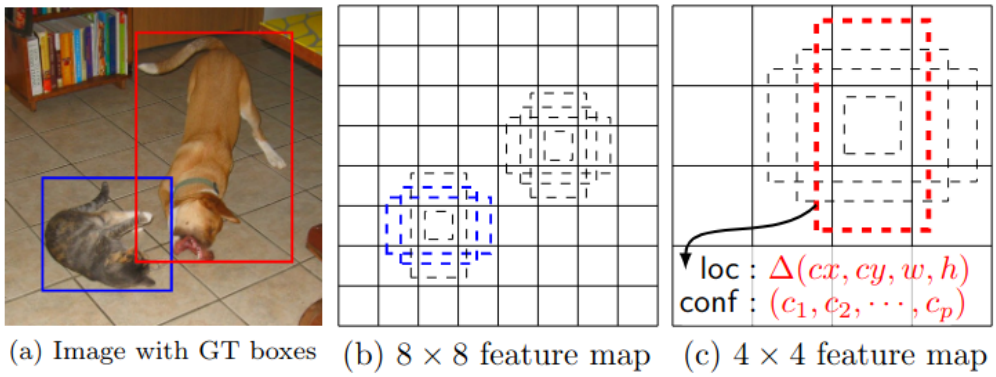

Figure 12: (a) Ground truth of images, (b) 8x8 feature map with anchor boxes, 4x4 feature map with anchor boxes. Taken from [9].

objects are read, the model try to minimize loss between the ground truth object's coordinates and the generated anchor boxes' location. Because of having many different size feature maps, the model has a higher chance to find large scale objects. The convolution filters are applied to the feature maps, after that, the confidence score of boxes is calculated in a way of classification and regression. At the end of the model, the non-maximum suppression is applied the all feature maps to eliminate and detect the same object over and over. SSD performs better on low resolution images and large scale objects against small scale objects. To improve small object detection results, the default boxes' size is had to be smaller.

2.2.2 You Look Only Once(YOLO) Family

The first proposed one stage object detector is YOLO(You Look Only Once) [10]. As a real time object detector, YOLO tries to solve localization problems like simple regression problem and doesn't have a special network or method to generate anchor boxes. Only one CNN is used for classification and localization, this is the main reason why YOLO is capable of working real time as a detector. For this reason, the algorithm uses two different loss functions [85][86], one for the localization measures the difference between the object's ground truth's location coordinates. The summary of YOLO is shown in Figure 13.

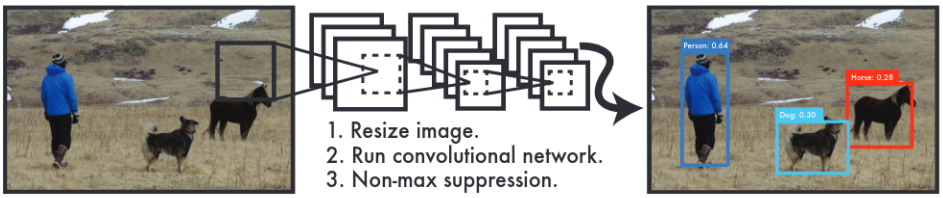

Figure 13: Input and output of YOLO algorithm. Taken from [10].

The other one is for measuring the difference between the ground truth class and the predicted class of the object. During the training, the algorithm tries to minimize the loss function. The meaning of the loss function's decreasing, the code is learning the difference between real values and predicted values. When the image is given to CNN, the image is divided by SxS grid by the algorithm and if the grid cell contains the center point of the object, that cell is responsible for finding the object's location and probability of the classes, the process is shown in Figure 14.

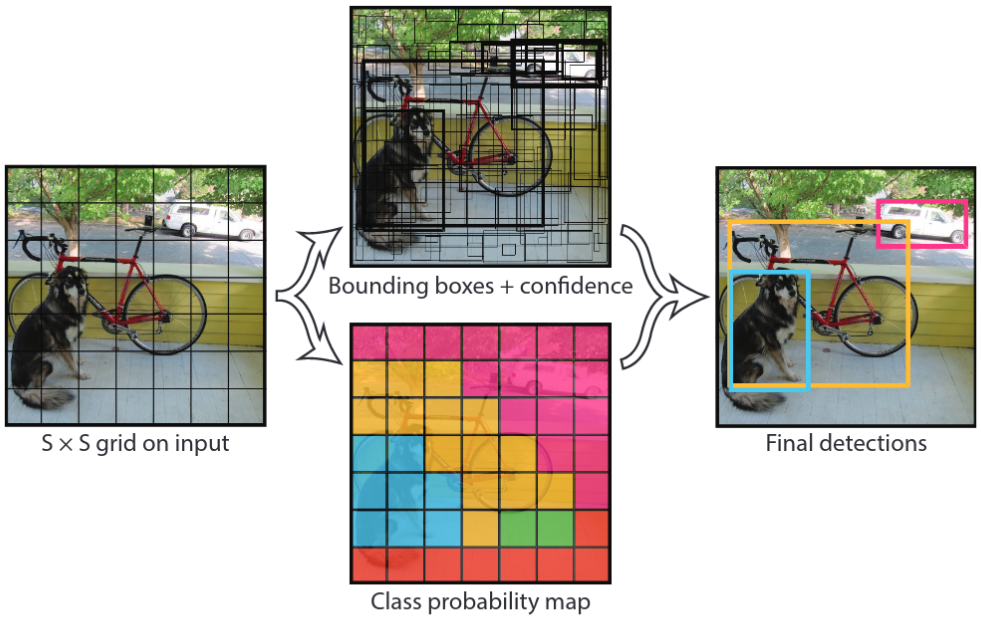

Figure 14: Grids and bounding boxes of YOLO algorithm. Taken from [10].

When the confidence is calculated, the probability of objectness score and IOU between gt and the predicted box is multiplied. Each box prediction has five values x, y,

w, h and the probability of the value. The YOLO consists of 24 convolutional layers and 2 dense layers. In YOLO, immediately after dense layers, the sigmoid activation function [87] is applied to the outputs to obtain values between 0 and 1. This allows the algorithm to make probabilistic predictions, where the predicted values represent the probabilities of different classes or events, the architecture is shown in Figure 15.

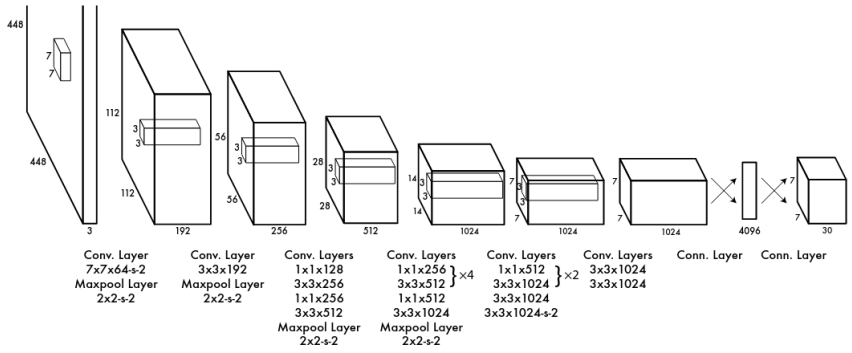

Figure 15: Architecture of YOLO algorithm. Taken from [10].

In 2016, Joseph Redmon published the Yolov2(YOLO9000) [11], which is an improved version of Yolov1 algorithm. YoloV2 takes the title of the state-of-the-art real time object detection subject. The different techniques are tried on Yolov2 and significant improvements are observed. The structure of YOLOv1 is changed and Darknet-19 is used for feature extraction, the architecture is shown in Figure 16.

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Figure 16: Input and output of YOLO algorithm. Taken from [11].

The Darknet-19 consists of 19 convolutional layers and five maximum pooling operations. After every convolutional layer, batch-normalization is applied to the output and this operation is make more successful the algorithm 2 percent. Batch normalization is a technique that is simple and very effective. Mainly focused on normalizing the inputs by subtracting the mean and dividing by the standard deviation. This makes us to ensure, the inputs have a mean of zero and a standard deviation of one, which can help the network to converge faster and make the network more stable. The other improvement has occurred using higher resolution images while training and accuracy increased by 4 percent. The input images are divided into grid cells and assigned 5 anchor boxes for each grid cell. The model is trained with COCO [23] and ImageNet [88] datasets with different size images and a total of 9418 classes. Thus, the variety of images provides more accuracy for the model.

YOLOv3 [89] is proposed by Redmon in 2018. There are two small changes from Yolov2. Darknet-53 is used for the backbone instead of Darknet-19. The other one is adding IoU to loss function,

2.3 Two Stage Object Detection Methods

The feature extraction is the basis of the deeper convolutional neural networks. On image classification problems, feature extraction is applied to the whole image. First, features are extracted and then these features are given into CNNs.

2.3.1 R-CNN Family

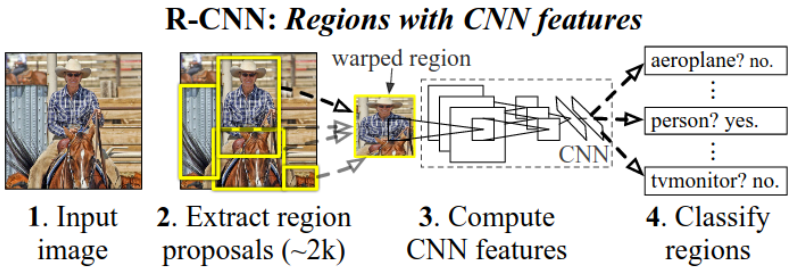

Figure 17: 1.Input Image, 2.Extracted Features, 3.CNN, 4.Classify Head. Taken from [12].

In 2014, Ross Girshick et al. [12], developed a method called Rich feature hierarchies for accurate object detection and semantic segmentation(R-CNN). R-CNN has two parts. The main parts of the algorithm are shown in Figure 17. First part tries to find where objects are located on an image by applying selective search [13]. The selective Search algorithm generates 2000 object candidate boxes for every image, object candidate boxes are called region proposals. These region proposals are rescaled and fit in the same size as each other. After that, CNN is fed by these regions to classify images and object detections. The features, extracted by CNN, are given to SVM classifiers and bounding box regressors. Thus, the class of regions and boxes is identified. The R-CNN method is successful with its new suggestions. However, it doesn't work in real time due to regions are classified per image and it takes 47 seconds per image. For this reason, Ross Girshick evolves his own method and published in 2015 Fast R-CNN model [14].

Selective Search is a method for finding objects on the image. First introduced in 2012 Uijlings et Al. The algorithm generates boxes that are candidates to contain objects. Selective search based on two strong methods exhaustive search [90] and segmentation [91]. Exhaustive search strives to locate every viable place by methodically listing all potential candidates. Segmentation is a way to determine candidate pixels to represent objects. Firstly, segmentation is applied to the input image and the similarity of each pixel is calculated with every adjacent pixel. During this calculation process, color, texture, size similarity and shape compatibility are involved in the process. Then, the final similarity is calculated. When the algorithm repeats these

steps after by after, region proposals are created by method. This operation is called the hierarchical segmentation process. A selective search example is shown in Figure 18.

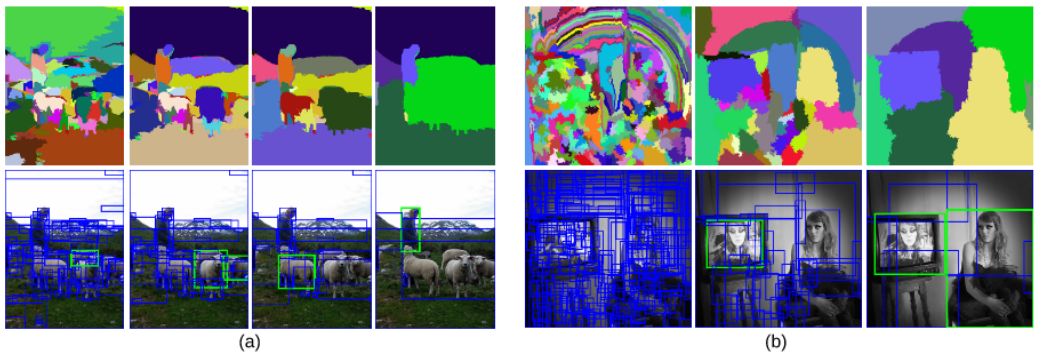

Figure 18: Selective search applied (a) and (b) images and the final results are displayed. Taken from [13].

Fast R-CNN have improvements on R-CNN's problems like slowness and high cost of training and inference process. The Fast R-CNN model takes images as CNN input for feature extraction instead of using proposed regions by selective search. The model introduces a new pooling method named as RoI Pooling. Also, candidate object proposals are generated from image and it is called regions of interest(RoI). The RoI pooling takes the extracted features from CNN and generated region of interest from an image as an input and outputs them as a fixed sized features map. Fully Connected Layers is fed by these feature maps to produce RoI feature vectors. On the last step of Fast R-CNN, respectively fully connected layer, softmax activation function and box regressor are applied on RoI feature vectors and the model gets the predictions. In this way, the accuracy and speed of the new model pass the R-CNN model. The inference time is reduced from 47 seconds to 2.3 seconds per image with Fast R-CNN improvements. The enhanced architecture is shown in Figure 19.

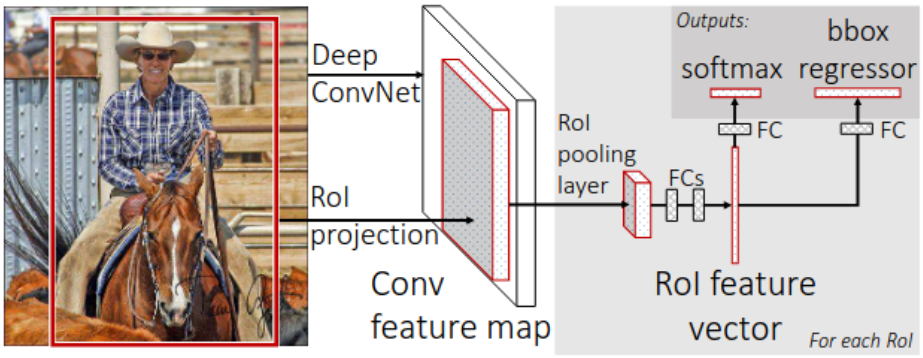

Figure 19: CNN takes the image as input. From one image both features and RoIs extracted. Taken from [14].

Faster R-CNN. Although there is huge progress in inference time, still the Artificial Intelligence community doesn't have models which are working in real time. In 2015 Shaoqing Ren developed a new model based on Fast R-CNN [15] and named Faster R-CNN. It is a nearly real time working object detector and gives output in 0.2 seconds per image. The model offers a new method for extracting region proposals. Instead of using the traditional selective search method, which actually is significantly slow, offers a using CNN for proposing objects. Similarly, on Fast R-CNN, A CNN takes an image as an input and extracts features, see in Figure 20. After that, these features are fed to the new CNN, called Region Proposal Network(RPN). The high quality region candidates are proposed by RPN and the pooling operation is applied with other features that come from former CNN. After the ROI pooling operation classification layers are fed by these feature maps and the prediction scores and bounding boxes are obtained.

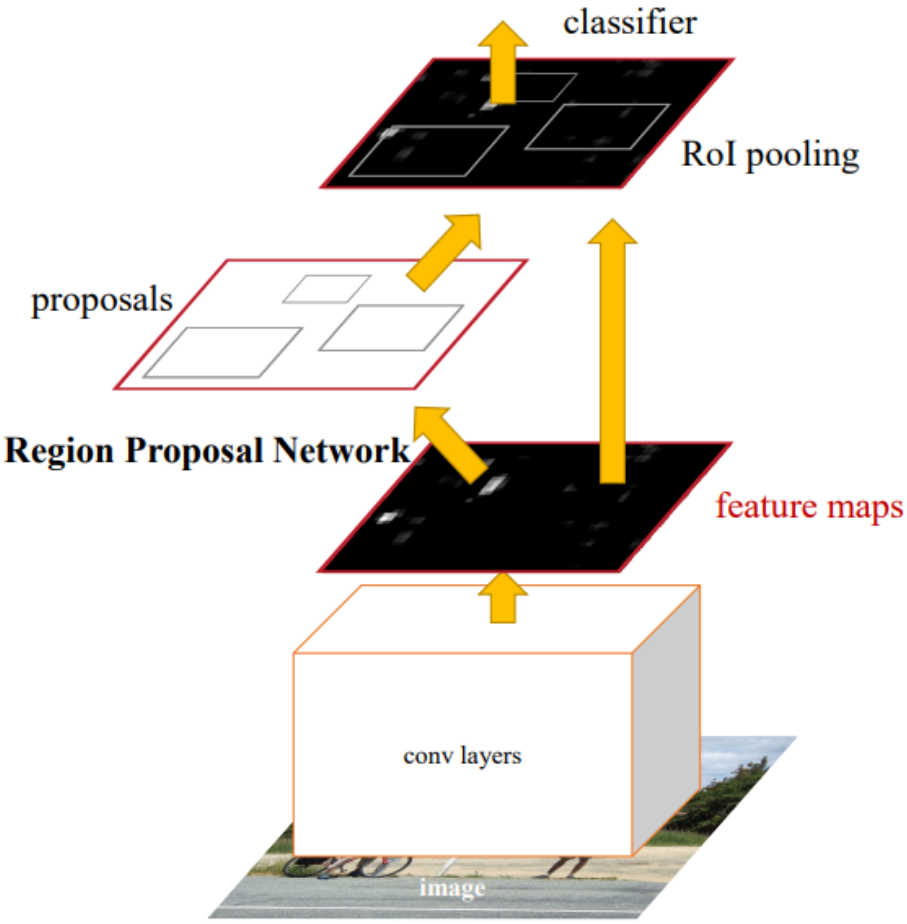

Figure 20: Faster R-CNN architecture. Taken from [15].

2.4 Anchor-Based Methods

In the object detection problem, the goal of the model is to identify the type and find the locations of objects in a frame. Generally, the successful algorithms proposed to generate random or specified size candidate squares or rectangles on the image. The model learns the patterns from combined generated anchor boxes and the extracted features from the image. Therefore, the score of the predictions is increased.

The algorithms are gained advantages by using anchor based methods. Giving the anchor's standards to the model provides huge simplicity in a way of memory usage, thanks to fixes sized anchors, the algorithm doesn't use to memory for generating anchor boxes from scratch each time. The anchor's standards can be fixed to every individual dataset if the dataset contains small, medium or large size objects. When

the size of anchor boxes is similar to the object's size, the algorithm can learn more accurately.

2.4.1 RetinaNet

In 2017, Tsung-Yi Lin et al proposed an anchor-based method RetinaNet [16] which is very similar to the working style of the human visual system and organ called the retina. RetinaNet is an algorithm that consists of one backbone and two CNN. The backbone, which is responsible for extracting the image features has two parts. The first part of the backbone is called ResNet, the main mission of the ResNet is extracting multi-scale feature maps and feeding the next part of the backbone which is called Feature Pyramid Network(FPN). FPN's aim is to generate different sizes of anchor boxes. The FPN has five layers and upsamples the extracted features between these 5 layers. FPN architechture is shown in Figure 21. Each layer is also responsible for generating different scale anchor boxes.

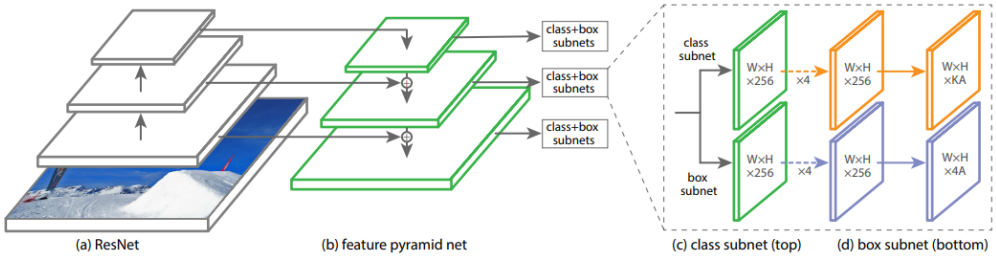

Figure 21: RetinaNet use as a backbone (a) ResNet and (b) feature pyramid network. Extracted features from these backbones followed by two separate subnets: (c) class subnet and (d) box subnet. Taken from [16].

The anchors are fixed in the size of given [32,54,128,256,512] respectively on the pyramid's layers and also different aspect ratios are applied for each size anchor [1:1,1:2,2:1]. Thus 15 anchors are generated for each location. As a result of this, RetinaNet has the capability of detecting objects on different scales. Some of the anchor boxes' size exceeds the size of the image, for this reason, these anchor boxes are ignored. The outputs from every layer of FPN are taken as the input of the classification and regression subnet. Both the subnets are attached to FPN's every layer and these subnets consist of four 4x4 fully connected layers. For the first three layers' output, the ReLU activation function [92] is applied, in contrast to the fourth layer. The sigmoid activation function is applied for the last layer.

The other proposed method in this work is focal loss [16]. Focal Loss is used on the classification subnet for providing more accurate learning activity. It is proposed to solve the class imbalance problem which is based on object detection datasets on

classification tasks. The cause of the imbalance classes problem is the image could contain several objects. However, the model has to focus on the main object that is desired to be learned. The other objects learn as a background of the image. If there are many background objects, the model cannot converge the foreground object and the algorithm shows poor performance for some foreground objects.

RetinaNet proposed to use the Focal Loss Function used in the issue of class imbalance. The model has gained a huge advantage when the dataset has more negative samples compared the positive samples. Simply, the focal loss reduces the weights of easy to detect objects and focuses on harder cases.

The model evaluates every anchor box to predict the probability of if the anchor box has obtained an object or not. Then the algorithm assigns a label to each anchor box. If the anchor box has an object, the label of this image is assigned as 1. Otherwise 0. Then, the calculated probabilities of anchor boxes multiply by the weights. If the probability is close to 0, the weight is large. The probability is high, the weight is small. The model calculates the final focal loss by summing the weighted losses for all anchor boxes and averaging them.

Basically, focal loss improves the accuracy of object detection algorithms by up-weighting the loss for challenging examples and down-weighting the loss for simple examples (those with a low predicted probability). This enables the model to better handle the class imbalance between positive and negative instances and to concentrate more on the difficult examples, which are frequently the most crucial for the task.

2.5 Anchor-Free Methods

Anchor free methods are as known as cost-friendly methods and their algorithms are designed without using anchor boxes as can be understood from the name of the method. The algorithms try to find an object's location, height and width using one of the pixels, grid or key-point based calculations. Not designing anchor boxes before or during the training is a beneficial way to implement more lightweight models. As a result of design-free anchor methods, the computational complexity and memory requirements are reduced.

2.5.1 CenterNet

In 2019, CenterNet [17], a very innovative model to detect objects, is introduced. In contrast to former traditional methods, the proposed algorithm focuses on the center of the objects, the objects represent a point in the whole image. The model consists of one backbone and two similar structures to estimate the location center and corners of the objects, shown in Figure 22. One of the structures focus on center pooling and a center heatmap to find the center of the object, the other one is responsible for finding corner by applying corner pooling.

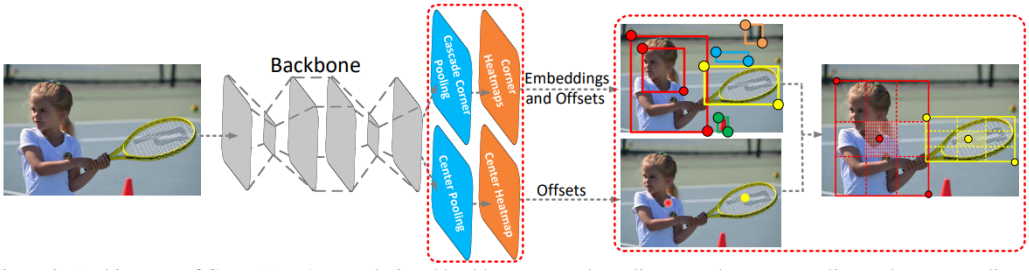

Figure 22: CenterNet's backbone is an hourglass network and produces embeddings and offsets. Taken from [17].

2.5.2 CornerNet

CornerNet [18] is proposed by researchers from Carnegie Mellon University as an anchor-free algorithm to detect and classify objects from images or videos. The algorithm tries to find the location of the object using paired key-points, the pairs consist of the top-left corner and bottom-right corner. Thus, there is no need for using anchor boxes. The CornerNet has a simple architecture with one backbone and two different prediction modules which contain CNNs. The Hourglass Network is used as a backbone, formerly it is used on human pose estimators. It can be one or more hourglass modules and it is actually a fully convolutional neural network. Two hourglass modules are used in this algorithm. The modules are used respectively, the images are given as input to the first hourglass.

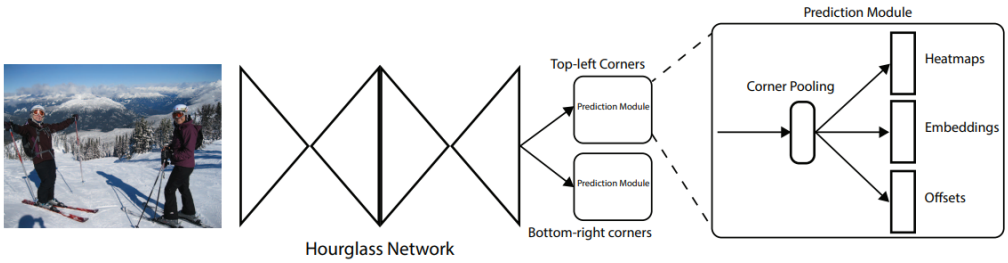

Figure 23: CornerNet's design consists of one hourglass network and two prediction modules for top-left corners and bottom-right corners. Taken from [18].

The hourglass network, (See in Figure 23) first downsamples the inputs by applying the CNNs and max pooling layers. Then, the upsampling operations are performed to the reduced features. As a result of this, the model learns the global and local features of the image. After two hourglass network, two different modules tries to predict the

top-left and bottom-right corners' coordinates and extracts heatmaps, embeddings and offsets. To make sure to find the corners, the Corner Pooling operation is performed on heatmaps, embeddings and offsets. Helping of these elements and loss function, the algorithm learns where is the image's corners.

2.5.3 Fully Convolutional One-Stage Object Detection(FCOS)

One-stage object detection technique FCOS [19] (Fully Convolutional One-Stage Object Detection) seeks to achieve better accuracy and can work on real-time. The idea was initially put out by Facebook AI researchers in their work "FCOS: Fully Convolutional One-Stage Object Detection," and it has subsequently grown in acceptance in the computer vision community.

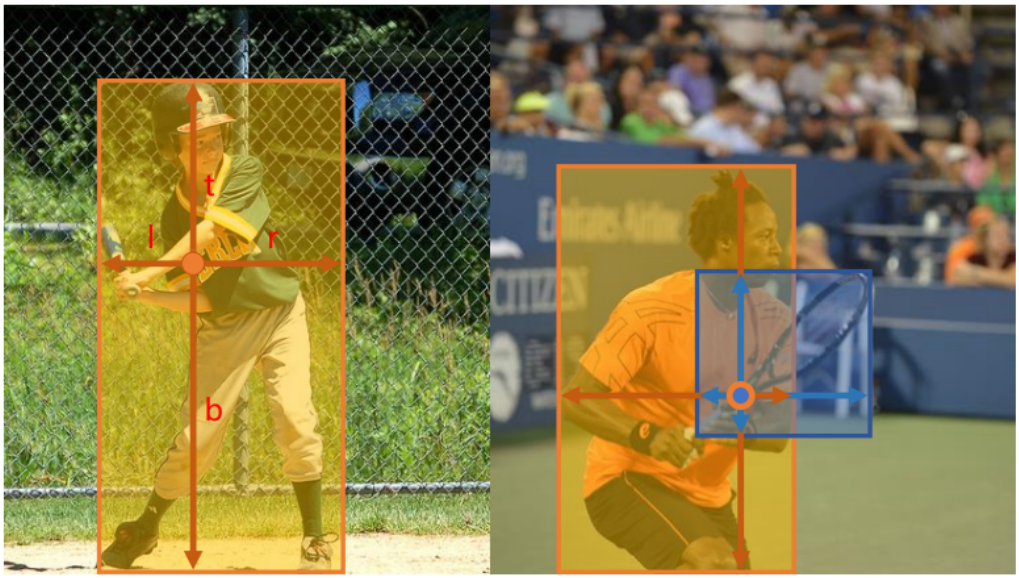

Figure 24: FCOS learns the object's location pixel based distances without anchor-boxes. Taken from [19].

FCOS employs a fully convolutional network for object detection, which implies that there are no fully connected (fc) layers in the network; this is one of the main characteristics of FCOS. In jobs requiring object detection where the size of the input picture might change, this enables the network to accept input of any size and provide the output of the same size. The detected objects by FCOS is shown in Figure 24.

In FCOS, a regular grid of anchor boxes—pre-defined boxes with varying sizes and aspect ratios—is used to anticipate item positions and class probabilities at each point. The network predicts if each anchor box contains an item and, if yes, to which class

it belongs by placing the anchor boxes over the input picture at regular intervals. The generated bounding boxes for the items in the image are then surrounded by the expected positions and confidence score of the class.

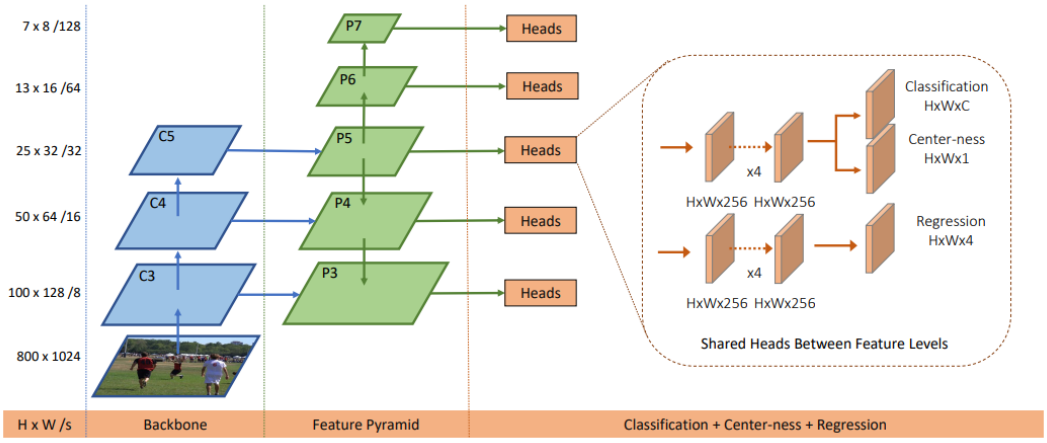

Figure 25: Architecture of FCOS. Taken from [19].

FCOS's ability to handle objects of diverse sizes and aspect ratios better than other single-stage object detection techniques like YOLO is one of its benefits (You Only Look Once). The reason for that is the FCOS makes use of a collection of anchor boxes with various scales and aspect ratios, enabling it to more accurately match the size and form of the objects in the image.

The fact that FCOS does not need many stages of computation to obtain the final detections makes it more efficient than two-stage object detection techniques like R-CNN (Regional Convolutional Neural Network) and its variations. As a result of this, it works well in applications that require real-time object identification and where speed is an issue.

In conclusion, FCOS is an anchor-free object detection approach that estimates item positions and class probabilities at each place in a regular grid of anchor boxes using a fully convolutional network, shown in Figure 25. It is more efficient than two-stage object identification techniques and good at handling objects of diverse sizes and aspect ratios, making it appropriate for real-time applications.

2.6 One Stage Object Detectors and Two Stage Object Detectors

There are two distinct methods for object recognition in computer vision: one stage object detectors and two stage object detectors.

One-stage object detectors are made to quickly identify the type and position of items in a picture in a single step. To analyze the full image and make a direct prediction about the type and position of objects, they employ a convolutional neural network (CNN). Due to the lack of a separate proposal generation process, one stage detectors are often quicker and more effective than two stage detectors. As a result of that, they are unable to alter their initial predictions of the objects,they could not be as precise as two stage detectors. YOLO and SSD are widely known and most used one stage object detectors.

Objects in a frame are found using a two-step method by two stage object detectors. They create a group of candidate object proposals in the first stage, which are areas of the image that are like to contain an object. Using the produced proposals, they categorize and fine-tune the position of the objects in the second stage. Because they have the chance to improve the object suggestions before making the final prediction, two stage detectors are usually more accurate than one stage detectors. The detections of SSD and Faster R-CNN are compared in Figure 26. Generating region proposals have huge importance with the role of highly accurate object detection. Generally, two stage object detectors are implemented based on R-CNN model.

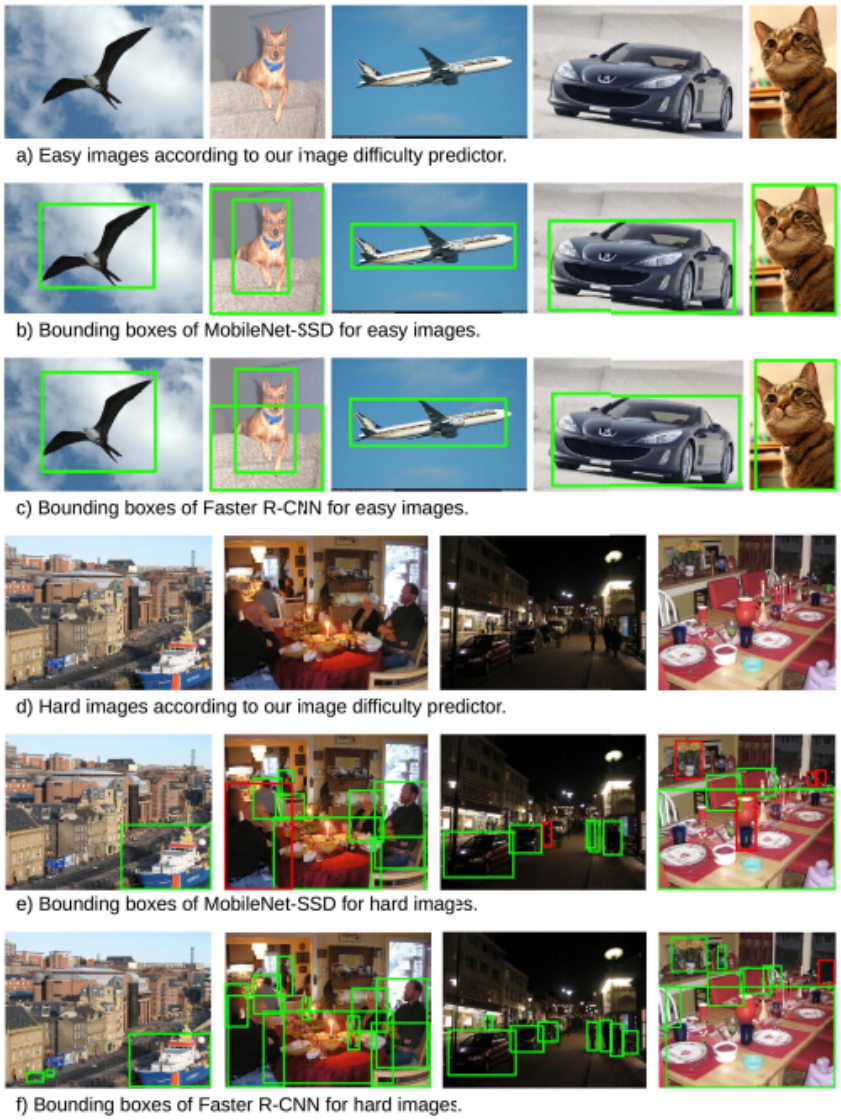

Figure 26: Comparing SSD and Faster R-CNN detection results on images. As it can be understood, Faster R-CNN performs better on smaller objects. Taken from [20].

In Conclusion, the decision of which method to employ is based on the specific details of the application, such as the needed speed and accuracy. A number of object detection benchmarks have shown that both one stage and two stage object detectors perform efficiently. However, by using optimal parameters, two-stage object detector's accuracy is better.

2.7 Anchor Based Methods vs Anchor-Free Methods

Except for one stage and two stage detectors, there are other methods in order to recognize and localize objects in photos or videos. Also, anchor-based and anchor free based methods have great results for object detection.

The spatial extent of objects in the picture is represented by predetermined anchor boxes using anchor-based approaches, also referred to as two-stage object detectors. The dimensions, aspect ratio, and location of these anchor boxes are determined by a grid of cells that divides the image into several parts. To better match the size and form of the objects in the training data, the model modifies the parameters of the anchor boxes during training. The algorithms employ the anchor boxes to provide object suggestions at the time of inference, which is subsequently improved using further convolutional layers and scored to produce the final object detections.

Anchor-based systems have trouble reliably detecting objects with a wide range of sizes and forms, although reasonably quick and effective. The model must be trained for a predefined set of object sizes when using anchor boxes, which might restrict the model's flexibility and ability to adjust to new tasks or data.

On the other hand, anchor-free approaches directly forecast the bounding box coordinates of the image's objects rather than using predefined anchor boxes. These techniques are frequently one-stage, which means they complete the duty of object detection in a single pass without the need for a subsequent proposal generating stage. Since they don't rely on predetermined anchor boxes and can find objects of any size and form, anchor-free approaches have the potential to be more versatile and flexible than anchor-based methods. However, because they demand additional calculations from the model at inference time, they could be a little slower and less effective than anchor-based techniques.

The selection between anchor-based and anchor-free approaches will ultimately come down to the particular requirements of the work at hand, including the complexity of the objects being identified, the available processing resources and the required speed and accuracy of the object detector.

2.8 Region Proposal Methods

In order to construct a list of possible bounding boxes or regions, that are most likely to contain objects of interest, region proposal methods are performed for object detection. These techniques help object detection algorithms operate more effectively by lowering the frequency of false positives. Using Region Proposal methods increases the accuracy of the model. Generally, the CNNs are fed by region proposals and the algorithm learns from more meaningful extracted features.

2.8.1 Selective Search

Selective search is a frequently used technique to generate region proposals on the area of object detection. The algorithm proposes the possible region by classifying the pixel according to its attributes like color, texture, size similarity and shape compatibility.

2.8.2 RPN

Region Proposal Network is a specially designed network for producing object proposal boxes. The network consists of two parts. The first part is the feature extractor, the image is taken as an input to the convolutional layer and extracted features are created. Then, the region proposal layer generates the proposed region using these extracted features. There are several specific methods for generating proposal boxes. Region Proposal Boxes are shown in Figure 27.

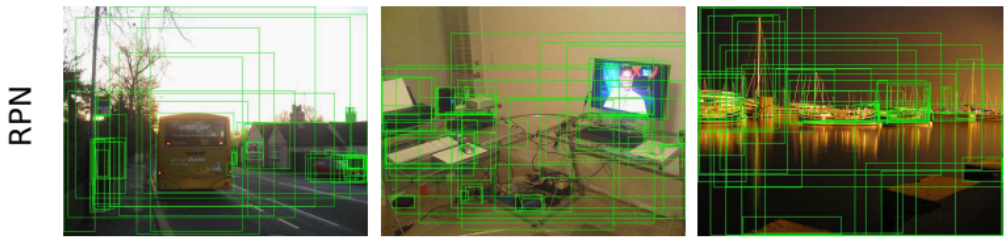

Figure 27: Example of proposed boxes from RPN. Taken from [21].

2.8.2.1 Iterative RPN

The Iterative Region Proposal Network method [22] is proposed for the increasing quality of the generated object proposals. The algorithm uses an iterative process for proposal generation, the architecture is given in Figure 28. The iterative term comes from the process of repeatedly creating proposals and improving them until a complete set of object detections is achieved.

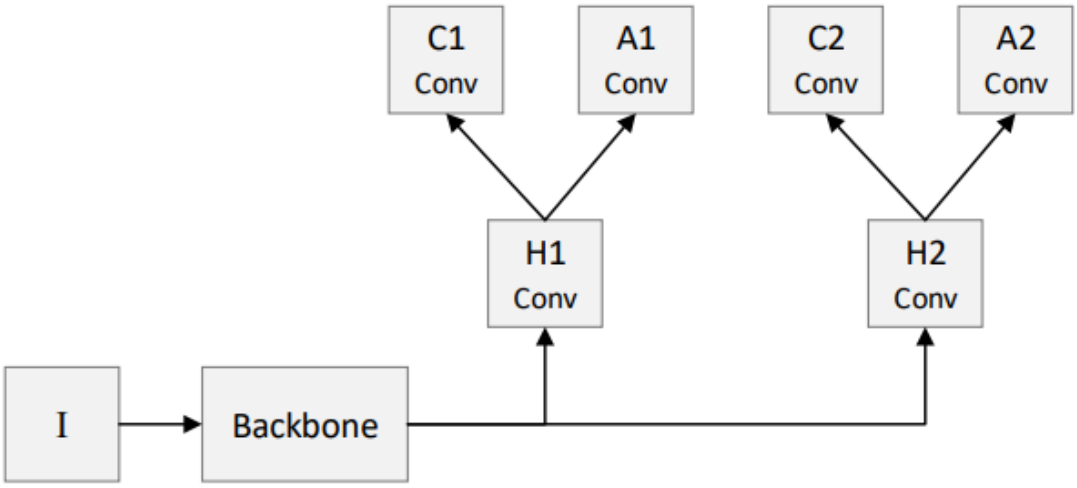

Figure 28: Example of Iterative RPN. Taken from [22].

By swiping a small window across feature maps taken from the input data and ranking each window based on how likely it is that it includes an object, the RPN creates proposals. This procedure is usually done several times, with the RPN changing the settings of the scoring function and the window's size and position to produce a larger number of recommendations.

2.8.2.2 Cascade RPN

The Cascade Region Proposal Network [22] is an architecture of CNN that aims to generate better quality region proposals and improve the accuracy of object detection (See in Figure 29).

The introduced model consists of two parts. The first part is a backbone. Backbone is responsible for extracting features of the input image and ResNet50 is used as a backbone. This CNN is responsible for generating region proposals. The cascade RPN uses the predefined anchor boxes, the size of the anchor boxes and aspect ratios is determined before the training.

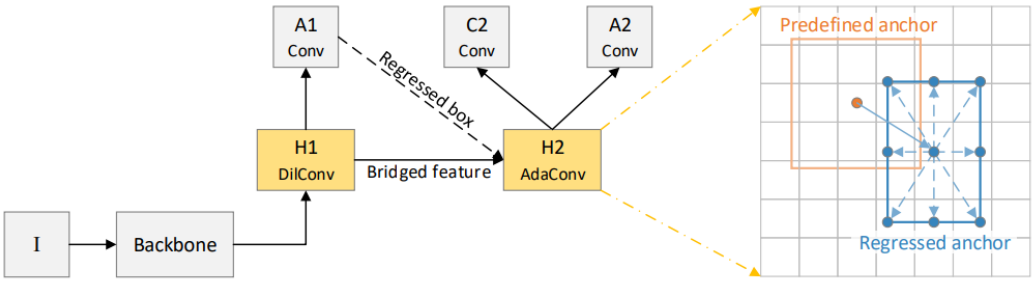

Figure 29: Architechture of Cascade RPN. Taken from [22].

The second stage is the refinement stage and takes the region proposals generated by the first stage as input. This stage tries to increase the quality of the proposals and find the location and class probability of the object. The layers calculate the difference between the proposed regions and the object's ground truth. Thus, the algorithm learns how it can refine the predicted boxes. After the refinement process, the regressor and the classifier heads predict the location and the class score of the object.

To sum up, Because it incorporates the benefits of both anchor-based and anchor-free techniques, the cascade RPN design enables an improved object detection procedure. Many object detection systems, including the well-known Faster R-CNN method, have utilized it.

CHAPTER 3

DIMENSION DECOUPLED REGION PROPOSAL NETWORK FOR OBJECT DETECTION

3.1 Method

In this section, we first summarize Region Proposal Network and mentioned the anchor mechanism that we propose to generate dynamic number of anchors. After that, we show the decoupled network architecture that enables us to predict anchor classes and regression values. Later, we describe the algorithm assigning positive/negative labels to the anchors for training supervision. Finally, how proposals are generated by merging results is explained.

3.1.1 Region Proposal Network

Region Proposal Network is a CNN aiming to find object proposals with higher quality compared to the other methods. RPN takes feature maps and their objectness scores for each as an input of CNN and after passing layer to layer. Sliding operation is performed on feature maps, sliding sizes must be NxN dimension, the CNN layers are fed by these sliding windows. After one layer is performed, the features are going through two layers. One of the layers is the classification layer and the other is the regression layer. Every point on the sliding window has features from the real image. These points are called anchor points and every anchor point has to have anchor boxes. When generating introduced anchor boxes scale and aspect ratios are used. The aspect ratio is calculated as with of image divided by the height of the image and the anchor box's scale is the size of the image. As a result of representing parts of the image, every point's anchor boxes have to represent image dimensions. the method uses strides to ensure representation the image's sizes and generates k different anchor boxes to obtain objects. The number of k consists of multiplying the aspect ratio's number and size of the scale points. The first assigned anchor boxes are dummy boxes, which have very few accuracies to contain objects. Additionally to this, some of the anchor boxes might not contain any objects, to fix this, the learning operation is applied by the next layers. The class of the object and coordinates are learned on labeled data. The intersection over union with the labeled data is calculated and the regressor layer learned the task on the rate of IOU. The coordinates of the object are specified as offsets x,y,w,h by the regression layer. The center point of

boxes are shown as x,y and the width and height of boxes are shown as w,h. data The feature's meaningfulness is protected while dimension reduction. At the end of the classifier and regressor, there is an elimination process of results to avoid finding the same object over and over. Non-maximum suppression is applied to the results and reducing the number of region proposal network is boosting the speed of the main model.

3.1.2 Dynamic Anchor Generation

In this thesis, we developed decoupled anchor dimensions to produce anchor sets for feature points. After passing an image through a backbone network, we obtain a feature map tensor. For anchor widths, we generate lengths ranging from 0 to w, and for anchor heights, lengths ranging from 0 to h are generated. These generated boxes with a combination of different sizes of w and h, labeled as positive and negative on the step. After, we take Cartesian products of the dimensions to produce a set of anchor boxes that intersect with objects having very different sizes and aspect ratios. In order to control the number of anchors, parameter k is introduced. The k parameter emerged experimentally and aimed to measure the effect of the number of anchor boxes produced on the performance. With this parameter, w/k lengths for width and h/k lengths for height are generated. The formula of total rectangles an mxn grid has is given by $(m+1)n(n+1)/4$. With the inclusion of the k parameter, we generate $\left[(w+k)h(h+k)\right]/(4k^4)$ anchors in total.

To make clear to the method, the following example is given, consider a 768x768 image where a 48x48 feature map (w=48, h=48) is obtained from an image with a stride value of 16. Setting k=1 produces 1,382,976 (100/100) anchors in total. Increasing the value of k to 2 gives 90,000 (6.5/100) total anchors. Further increasing the k value to 4 gives 6084 (0.04/100) anchors in sum. Producing too many anchors requires a lot of computing power and complicates learning, but objects are covered better. On the other hand, with less anchor, faster computations are made and the learning task simplifies, but intersection rates with objects reduce especially for the small objects. In Figure 30, a torch-like code of the anchor generation step is depicted.

```python
def gen_anchors(w, h, k):
    """create 1-d tensors with values from
    the interval [0, w|h] with a step k"""
    x = torch.arange(0, w+1, k)
    y = torch.arange(0, h+1, k)
    "create all x1, x2 and y1, y2 pair"
    x1x2 = torch.cartesian_prod(x, x)
    y1y2 = torch.cartesian_prod(y, y)
    "filter invalid pairs"
    x1x2 = x1x2[x1x2[:, 1] > x1x2[:, 0]]
    y1y2 = y1y2[y1y2[:, 1] > y1y2[:, 0]]
    "repeat tensors to create x, y pairs"
    x1x2 = x1x2.unsqueeze(0).repeat(y1y2.shape[0], 1, 1)
    y1y2 = y1y2.unsqueeze(1).repeat(1, x1x2.shape[1], 1)
    "reshape tensors from (N, N, 2) to (N*N, 2)"
    x1x2 = x1x2.view(-1, 2)
    y1y2 = y1y2.view(-1, 2)
    "concatenate tensors to form anchors"
    anchors = torch.cat((x1x2[:, 0:1], y1y2[:, 0:1],
    x1x2[:, 1:2], y1y2[:, 1:2]), dim=-1)
    return anchors
```

Figure 30: Dynamic Anchor Generation Code

3.1.3 Network Architecture

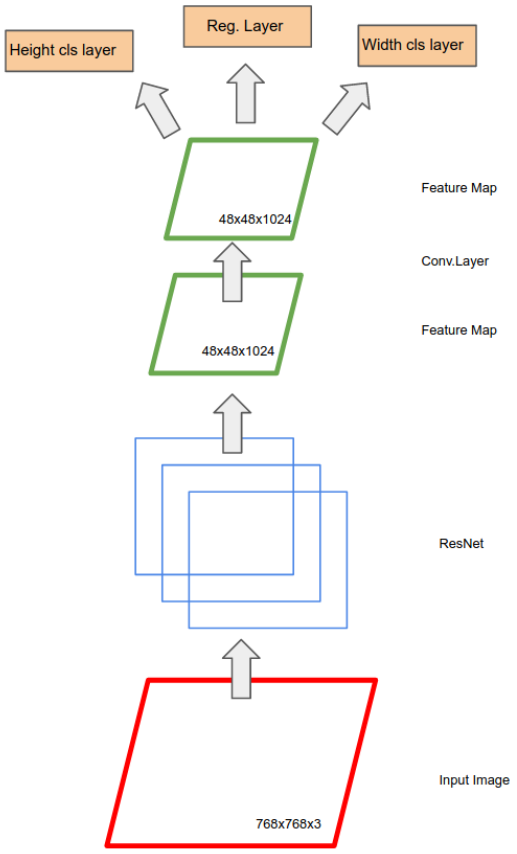

Figure 31: Network architecture.

Our network architecture is depicted in Figure 31. The input image is fed into the ResNet-101 backbone network and a feature tensor is obtained. As an example in Figure 31, an image having 768 widths and 768 heights is fed into the backbone network and a feature tensor with dimensions 48x48x1024 is obtained. The backbone network is pre-trained with the Image-Net [88] dataset for the image classification task and the classification head is discarded. For the training of the object detection task, the weights of the backbone network are frozen.

It means the backbone network is used as a feature extractor. The backbone network learns generic features and weights are frozen to not perturb learned weights during object detection training. In order to learn proposal features specific to images a hidden representation layer with 3x3 convolution filters is used. This layer outputs a tensor with the same dimensions as the input tensor. After this layer, two classifications and one regression layer are used to classify anchors and learn offset parameters. One classification layer for heights and one classification layer for width is used. Classification layers have N output channels. $W/(s*k)$ and $H/(s*k)$ classes

can be created for width and height lengths. For an average size 768x768 image, 48, 24, and 12 length classes can be constructed with k = 1, 2, 4 in order. Therefore, we change the N value according to the k value in my experiments. The regression head has 4 output channels. we force each feature point to make one prediction where one-to-one matching with anchors and ground-truth objects is made which is explained in the next section. The rest of the network is the same as Faster R-CNN where object proposals are pooled with an ROI-Pooling operation and a feature tensor having the same shape is obtained. Later, classification and regression heads are used to make multi-class classification and regression fine-tuning.

3.1.4 Dynamic Label Assignment

The prediction boxes produced during the inference have a certain intersection with the ground truth of the object but every anchor has different intersection-over-union (IoU) ratios with ground-truth objects. Some anchors have high-overlapping with objects, some of them barely intersect with objects and others have no intersection with them. Since it will make it difficult for us to get healthy results, we need to eliminate some of the boxes. The elimination method is done by labeling the box as positive or negative by looking at the intersection of the produced boxes with the ground truth and the area they have.

High-overlapping anchors should be labeled as positive, anchors having no or low intersection should be labeled negative and in-between anchors should be ignored as they can send noisy signals that can harm training.

Our labeling algorithm works as follows: All-pair IoU rates between anchors and ground-truth objects in the related image are calculated. For each anchor box which ground-truth box they have the maximum iou is found. For each ground-truth box, anchor boxes that have the maximum iou with it are taken into account. Center points of these anchor boxes are found and unique centers are stored. Then maximum iou of each unique center is found. Finally, anchor boxes having maximum iou values are saved. If any ground-truth box has a higher iou value than any other anchor with the same center, the saved anchor box is replaced with the higher overlapping one. In this way, elimination is made among the boxes with high intersections. After these steps, some ground-truth boxes may have no assigned anchor boxes in the end. In order to assign anchor boxes to these ground-truth boxes another loop is started. Anchors not claimed by other anchors are sorted according to the iou values with the ground truth in question and top anchors assigned to it.

With this procedure, although some ground-truth boxes have plenty of good overlapping anchor boxes, some anchors with low iou values are still assigned to these ground-truth boxes (getting a positive label). To prevent the assignment of low iou anchor boxes to the ground-truth boxes, we propose dynamic thresholding to eliminate poor anchor boxes. This procedure works as follows: A set of threshold values and a set of target counts are determined as hyper-parameters. For each ground-truth object iou values of anchor boxes threshold-ed with predetermined threshold values

and the number of anchors counted. The maximum threshold value of the threshold values meeting the "anchor count is bigger than the target count" criteria is selected as the threshold for the ground-truth box. Anchor boxes that can not meet this criterion are labeled as ignored. Thus, we prevent ambiguous anchors from directing training. Two mentioned procedures are given in Figure 32.

```
def assign_labels_dyn(temp_iou, temp_hwg):
    # selected threshold list
  s_thrs_ls = []
    # for each gt box label anchors with dynamic thresholding
  for gt_we in range(len(gt_boxes)):
    gt_box_we = gt_boxes.tensor[gt_i]
    gt_m = (gt_we == temp_hwg[:, :, 2])
    # select thr
    ious = temp_iou[gt_m].unsqueeze(-1).repeat(1, len(thrs))
    counts = (ious > thrs.unsqueeze(0)).sum(dim=0)
    s_thrs = thrs[counts >= target_counts]
    s_thr = s_thrs.max() if len(s_thrs) else thrs[0]
    s_thrs_ls.append(s_thr.item())
    # set h/w labels
    iou_m = temp_iou[gt_m] > s_thr
    s_cycx = gt_m.nonzero()[iou_m]
    s_hwg = temp_hwg[gt_m][iou_m]
    s_cy, s_cx, s_h, s_w = s_cycx[:, 0], s_cycx[:, 1], s_hwg[:, 0], s_hwg[:, 1]
    h_labels[s_cy, s_cx, s_h] = 1.
    w_labels[s_cy, s_cx, s_w] = 1.
    # set reg targets
    half_h, half_w = (s_h + 1), (s_w + 1)
    s_rs = torch.stack((s_cx - half_w, s_cy - half_h, s_cx + half_w, s_cy + half_h)).T
    s_anchors = (s_rs * stride) + stride // 2
    r_labels[s_cy, s_cx] = (gt_box_we - s_anchors) / stride
    r_mask[s_cy, s_cx] = True
    # return w, h, regression labels and regression targets
    return h_labels, w_labels, r_labels, r_mask, rs, s_thrs_ls
```

Figure 32: Assinging Dynamic Labels Code.

3.1.5 Producing Proposal Boxes

In this proposed method, as we made dimension decoupling and predictions are made separately for each dimension, we need to merge results to form proposal boxes. my bounding box proposal construction method works as follows: height and width scores are calculated with the sigmoid function from logits. Unique center-width and center-height triplets are found. Scores of remaining indices are ignored. This is because not all anchor centers consist of all dimensions like in traditional RPN. For example, the topmost and leftmost feature map point contains only a single (smallest) anchor box. The reason for this is any other anchor would have excess image dimensions.

On the other hand, the feature map point that resides in the center of the feature map has the most number of anchors. Figure 37 shows my anchor enumeration compared to the RPN in which the same anchors for all feature map points are used. After ignoring unrelated dimensions, we find the dimensions having maximum scores and construct proposal boxes from these. As the last step, non-maximum suppression is applied to remove highly overlapping proposals. My proposal generation algorithm is given below.

3.1.6 Non-Maximum Suppression

Non-Maximum Suppression is an algorithm that is generally used in the last part of the object detection models. At the end of the detection process, the proposed object detection model predicts a lot of boxes. Some of them of these boxes can have intersections with each other. The meaning of this, the algorithm produces predictions for the same object but with different confidence scores. A successful deep learning model could make only one estimation for each object and it must have high class probability score. To achieve this goal, the NMS algorithm takes highest probability bounding boxes and then, looks for intersection of over union with the other predicted boxes. If the IoU is bigger than 0.5, omits the other predicted box. The algorithm repeats this loop until eliminates the low scored bounding boxes and predicts only one box with the highest confidence score.

CHAPTER 4

EXPERIMENTS

4.1 Object Detection Datasets

The most commonly used and large scale dataset in object detection is Microsoft's Common Objects in Context. Generally, the new proposed algorithms' success is measured by their accuracy on the MS-COCO dataset.

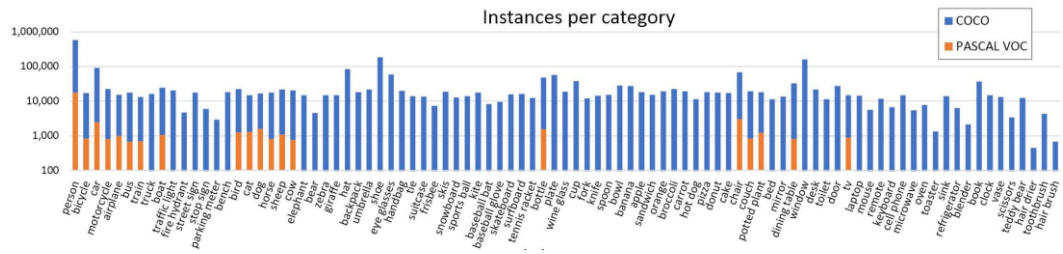

Figure 33: Class instances comparison between MS-COCO and PASCAL VOC datasets. Taken from [23].

MS-COCO is prepared to comprise commonly encountered objects in daily life and consists of 328.000 images. The images are taken from complex scenes and have 2,500,000 labeled instances. Many computer vision studies are completed with COCO dataset in areas of image classification, object detection, object recognition, human pose-estimation and image segmentation. These images contain information like co-ordinates of the object's location, segmentation details and 17 key-points of human objects. The dataset has 80 classes and is shown in Figure 33.

There are several different data label formats for an individual tasks like object detection, keypoint detection, stuff segmentation, panoptic segmentation, densepose, and image captioning. The object detection label format is stored on JSON. The file has information about a specific image's name, id, object's category, object coordinates and segmentation information.

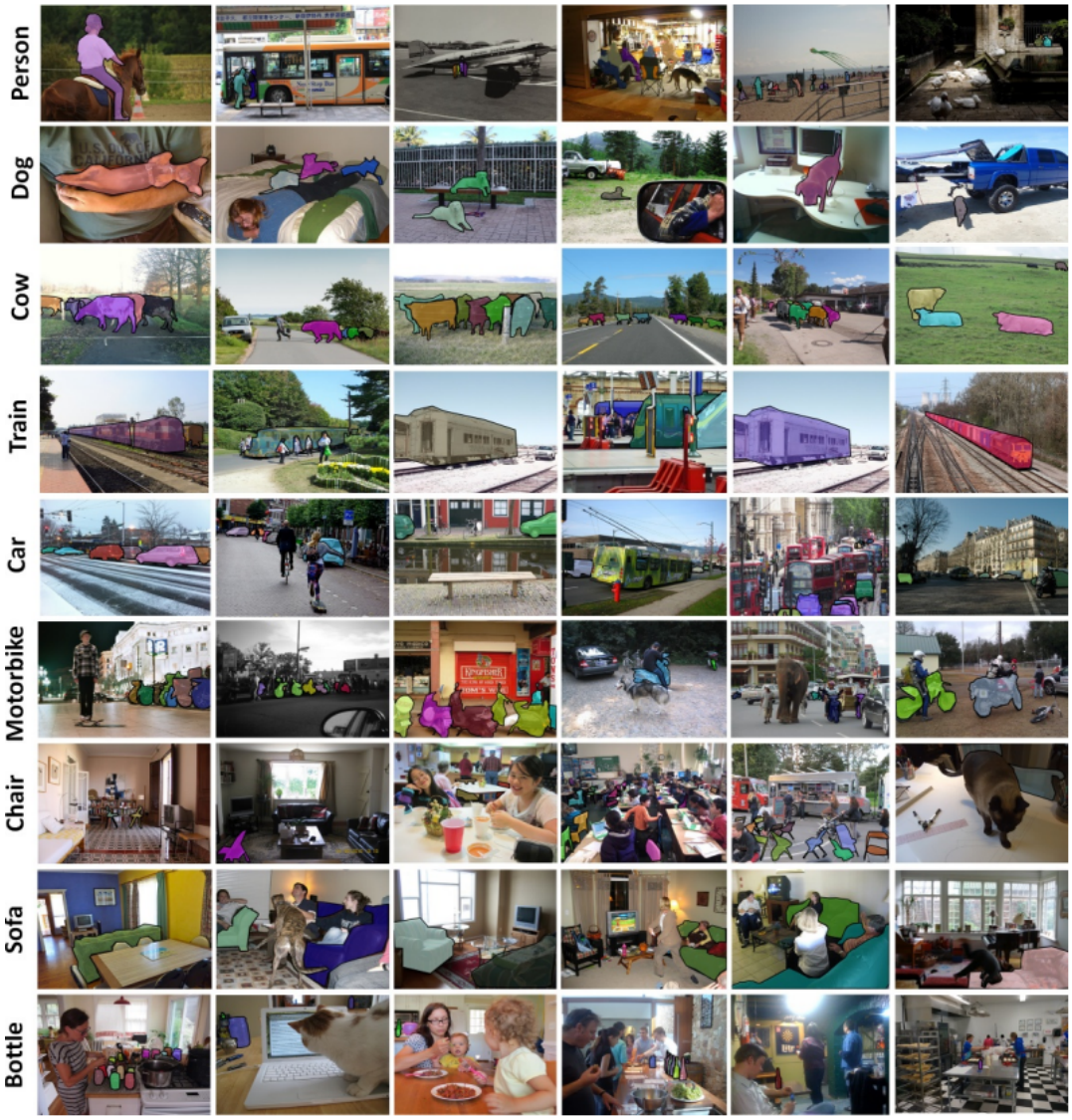

Figure 34: Example images from MS-COCO dataset. Taken from [23]

Thanks to having lots of objects in one image, the algorithms which are trained by COCO have gained advantages to distinguish foreground objects from background objects. The photos are given in Figure 34.

The other popular dataset in object detection is PASCAL VOC first introduced in 2005 with only 4 classes and 1578 images. From 2005 to 2012 the number of images reaches 11,530 and the number of classes increased 20. Like MS-COCO dataset, Pascal VOC, Figure 35 dataset also has images from real life with many different objects in one image. The dataset is accepted as a benchmark dataset for image classification and object detection.

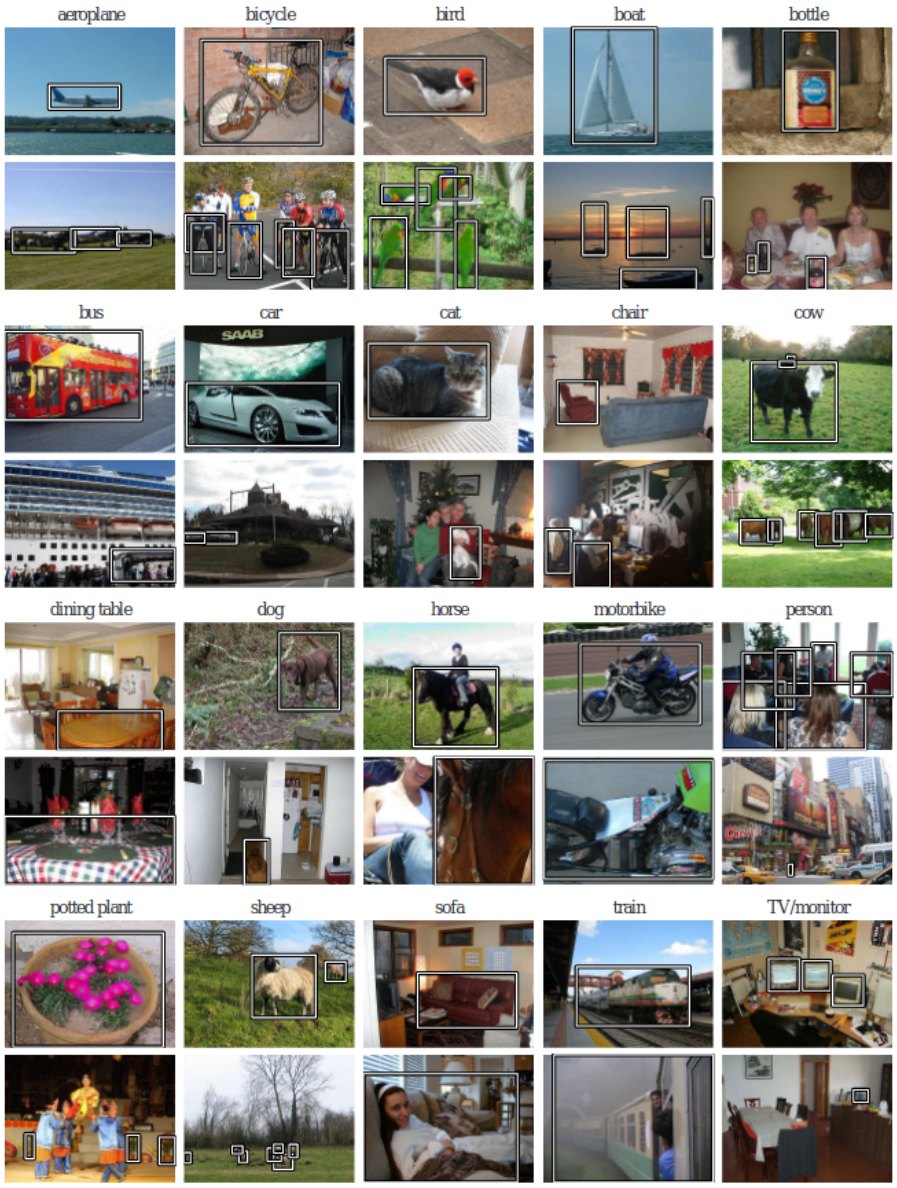

Figure 35: Example images from PASCAL VOC dataset. Taken from [24]

Annotations are kept in XML files and have location information of objects and this information is easily read by the developers.

4.2 Evaluation Metrics

Evaluation metrics are globally accepted methods used to measure the performance and accuracy of the developed algorithm. During training, the dataset is divided into

two training and test datasets. The algorithm trained with the training dataset is tested on the test dataset that it has never seen. During the test, the predictions of the algorithm are compared with the ground lines of the test dataset and the prediction results are recorded. According to the recorded results, true positive, true negative, false positive and false negative numbers are analyzed by following various procedures such as Accuracy, Precision, Recall, Specificity and F1 score [93].

The fact that the algorithm contains correct results sheds light on whether the model has learned the task. The accuracy of the algorithm is the ratio of correct results to all correct or incorrect results.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{1}$$

Precision is a metric that measures how accurate are our predictions.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

The precision metric has different types. There are two main types of precision metrics, based on the dimensions (small, medium and large) of the objects and the percentage of intersection of the ground line with the detection.

The average precision metric has a variation called AP_S (AP Small) that exclusively assesses how well the model performs with regard to small items. It is crucial to assess the model's performance on small things individually because they are typically harder to identify than larger objects. AP Small is calculated in the same way as the standard average precision measure, but only takes into account ground truth boxes and detections that are narrower or taller. Depending on the task and dataset, the precise meaning of small may change, but often, a threshold is set at a height or width of less than 32 pixels. The threshold for medium-sized objects is set to a height or width between 32 and 96 pixels. Objects larger than the specified dimensions are called large objects. Precision abbreviations are shown as AP_M for medium objects and AP_L for large objects.

With an emphasis on the high-recall portion of the curve, the AP_{50} [93] metric in object detection assesses the precision-recall trade-off for a model's object detections at a specific intersection over union (IoU) threshold. A detection is only considered a genuine positive by AP_{50} if its IoU with the ground-truth bounding box is at least 50/100. The precision-recall curve for the model's detections is generated first, and the average precision overall recall levels, up to a recall of 50/100, are then computed to produce the AP_{50} measure. AP_{50} is a helpful indicator for evaluating the performance of various models and is frequently used to assess the effectiveness of object detection models. By percentage of intersection, the number next to the AP may change (AP_{75} etc).

Recall measures the rate of true positives over the predicted true samples. Average Recall [93] is computed for a certain number of detections. For example, if we calculate average recall for 100 detections, it is shown as AR_{100}.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

To find the F1 score, the harmonic mean is calculated using precision and recall values. The harmonic mean approaches 1, when precision and recall achieve high values at the same time. If the F1 score is close to 1, it shows that the model is working very successfully, and if it is close to 0, it shows that the model is far from making predictions.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4}$$

Sensitivity calculates the proportion of correctly detected positive classes. This metric gives how well the model does at recognizing a positive class.

$$Specificity = \frac{TN}{FP + TN} \tag{5}$$

The graph between True Positive Rate and False Positive Rate is plotted using ROC curves. Plots like this are produced at various classification levels. Therefore, if our classification threshold is low, we can categorize more things as positive, which will increase the number of both False Positives and True Positives. A typical ROC curve is shown in Figure 36.

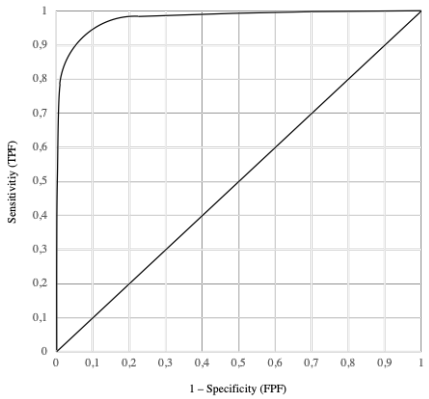

Figure 36: An example of ROC curve.

4.3 Results

We did experiments on the COCO dataset [23] to evaluate our algorithm and compare it with the region proposal network. Using various backbones, traditional RPN and the DA2RPN method that we recommend were compared by looking at the average recall and average precision values. Comparison results are shown in Table 1. Initially, 100, 300 and 1000 proposals were produced and results were obtained for the ResNet-50-FPN backbone. Then, the observations are given for the deeper convolutional neural networks ResNet-100-FPN and ResNext-100-FPN, keeping the RPN method constant. Average recall results increase when the model is trained with a deeper backbone and more proposals with the RPN method. The performance of the proposed DA2RPN method was examined by keeping constant the backbones used to produce the result for the RPN method.

Table 1: Comparison of the recall values of RPN and proposed method with respect to the number of anchors generated.

Method	Backbone	AR_{100}	AR_{300}	AR_{1000}
RPN [15]	ResNet-50-FPN	42.5	51.2	57.1
RPN [15]	ResNet-101-FPN	45.4	53.2	58.7
RPN [15]	ResNext-101-FPN	47.8	55.0	59.8
DA2RPN	ResNet-50-FPN	44.0	52.9	58.4
DA2RPN	ResNet-101-FPN	47.2	54.8	59.3
DA2RPN	ResNext-101-FPN	49.5	56.2	60.1

Our method involves the k parameter which controls the anchor number in the anchor generation step. We investigated the effect of this parameter in our experiments. k parameter directly affects the number of generated anchor boxes. When k equals 1, the most numbers of anchor boxes are proposed and when k equals 4, the least number of anchor boxes are generated. The optimal values are obtained when k equals 2. The reason that getting the lowest values when k equals 4 is the method proposed the least number of boxes. Therefore, the algorithm cannot find the object's true location. The results are shown in Table 2.

Table 2: The effects of the k parameter over the proposed method.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Baseline	35.9	58.0	38.4	21.2	39.5	46.4
DA2-RPN (k=1)	37.1	58.7	39.2	22.3	40.4	47.2
DA2-RPN (k=2)	38.4	59.9	40.3	23.6	41.5	48.2
DA2-RPN (k=4)	33.4	53.2	34.5	20.1	36.9	43.8

Table 3 depicts the effect of the proposed modules. When Dynamic Anchor and Dynamic Thresholding modules are applied separately, both of the modules increase the accuracy of the baseline method. When we applied two of them at the same time. We obtained the best results and AP metric increased from 35.9 to 38.4.

Table 3: Average recall and precision scores when the proposed method are applied to baseline.

Module	AR_{100}	AR_{1000}	AP_{50}	AP
Baseline	42.5	57.1	58	35.9
Dynamic Anchor	43.1	57.8	58.9	37.1
Dynamic Thresholding	43.3	57.9	59.0	37.3
Both Modules	44.0	58.4	59.9	38.4

Our region proposal method's example proposed regions, over an image belongs to MS-COCO dataset, are shown in Figure 37. Traditional RPN method's example proposed regions, from an image belongs to MS-COCO dataset, are shown in Figure 38. The results of the popular and most used object detection method's results on MS-COCO dataset are shown in Table 4.

Table 4: Detection Algorithms' scores on MS-COCO dataset.

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOv2 [11]	DarkNet-19	21.6	44	19.2	5	22.4	35.5
SSD [9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
Faster R-CNN [15]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
RetinaNet [16]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [16]	ResNext-101-FPN	40.8	61.1	44.1	24.1	44.2	51.2
YOLOv3 [89]	Darknet-53	33	57.9	34.4	18.3	35.4	41.9
FCOS [19]	ResNet-50	37.1	55.9	39.8	21.3	41.0	47.8
CornerNet [18]	ResNet-50	40.5	56.5	43.1	19.4	42.7	53.9

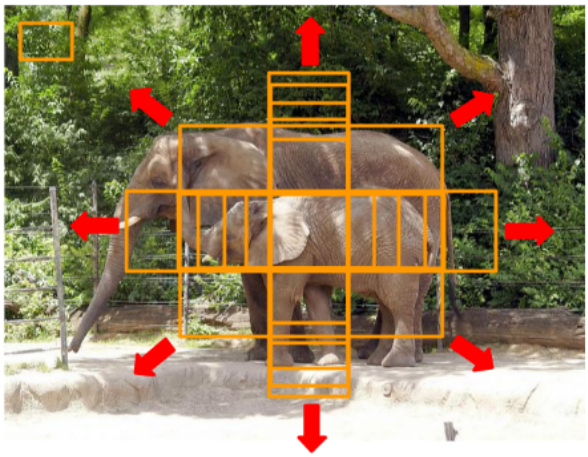

Figure 37: Our method's generated anchor boxes

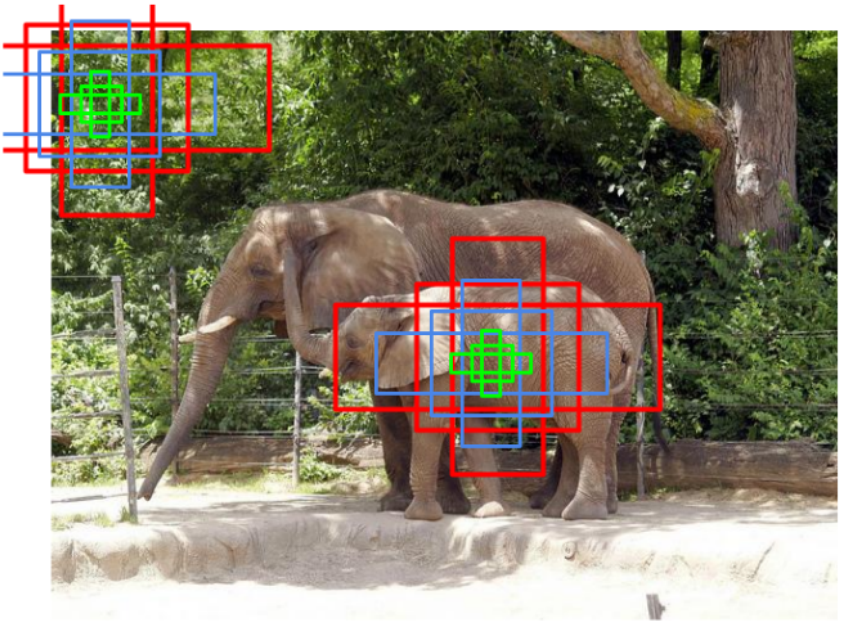

Figure 38: Traditional RPN method generated anchor boxes

CHAPTER 5

CONCLUSION

Although the traditional region suggestion method is critical to the performance of 2-stage object detection methods, because it uses a fixed size and number of anchors and fixed thresholds, it doesn't take full advantage of its anchor usage potential. To address these shortcomings, we proposed a method that improves both generating anchors and assigning anchors to ground truth objects.

The method we propose assigns different sizes and numbers of anchors to the extracted feature points and does not produce unnecessary anchors in places where it is unlikely that an object of that size will exist. On the other hand, it produces many anchors of different sizes in the midpoints of the image where objects of all sizes can be located. In addition, even if the IoU value of the anchors produced for each object is the same, they are not of the same quality, so object-specific threshold values are determined, thus providing a higher quality anchor tag assignment.

Thanks to these two modules we recommend, the region proposal network has been able to use anchors more effectively and it has been observed that the performance in object detection datasets has increased. In future studies, it is aimed to reduce the cost of calculation speed by eliminating unnecessary anchors by using the attention mechanism and to increase the performance by providing smoother training.

REFERENCES

[1] S.-C. Wang, "Artificial neural network," in *Interdisciplinary computing in java programming*, pp. 81–100, Springer, 2003.

[2] H. Ng, S. Ong, K. Foong, P.-S. Goh, and W. Nowinski, "Medical image segmentation using k-means clustering and improved watershed algorithm," in *2006 IEEE southwest symposium on image analysis and interpretation*, pp. 61–65, IEEE, 2006.

[3] A. Betancourt, P. Morerio, L. Marcenaro, M. Rauterberg, and C. Regazzoni, "Filtering svm frame-by-frame binary classification in a detection framework," in *2015 IEEE International Conference on Image Processing (ICIP)*, pp. 2552–2556, IEEE, 2015.

[4] Y. Li and B. Cheng, "An improved k-nearest neighbor algorithm and its application to high resolution remote sensing image classification," in *2009 17th International Conference on Geoinformatics*, pp. 1–4, 2009.

[5] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[6] C.-C. J. Kuo, "Understanding convolutional neural networks with a mathematical model," *Journal of Visual Communication and Image Representation*, vol. 41, pp. 406–413, 2016.

[7] Z.-W. Yuan and J. Zhang, "Feature extraction and image retrieval based on alexnet," in *Eighth International Conference on Digital Image Processing (ICDIP 2016)*, vol. 10033, pp. 65–69, SPIE, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[11] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[13] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[14] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[16] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[17] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6569–6578, 2019.

[18] 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.

[19] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9627–9636, 2019.

[20] P. Soviany and R. T. Ionescu, "Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction," in *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pp. 209–214, IEEE, 2018.

[21] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[22] T. Vu, H. Jang, T. X. Pham, and C. D. Yoo, "Cascade rpn: Delving into high-quality region proposal network with adaptive convolution," in *Neural Information Processing Systems*, 2019.

[23] 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," in *European conference on computer vision*, pp. 740–755, Springer, 2014.

[24] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–308, 2009.

[25] L. S. Kovásznay and H. M. Joseph, "Image processing," *Proceedings of the IRE*, vol. 43, no. 5, pp. 560–570, 1955.

[26] M. P. Ekstrom, *Digital image processing techniques*, vol. 2. Academic Press, 2012.

[27] G. Marbach, M. Loepfe, and T. Brupbacher, "An image processing technique for fire detection in video images," *Fire safety journal*, vol. 41, no. 4, pp. 285–289, 2006.

[28] J. G. Berryman, "Measurement of spatial correlation functions using image processing techniques," *Journal of Applied Physics*, vol. 57, no. 7, pp. 2374–2384, 1985.

[29] L. S. Davis, "A survey of edge detection techniques," *Computer graphics and image processing*, vol. 4, no. 3, pp. 248–270, 1975.

[30] H. Hou and H. Andrews, "Cubic splines for image interpolation and digital filtering," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 26, no. 6, pp. 508–517, 1978.

[31] S. Anastasiadis, J.-K. Chen, J. Koberstein, A. Siegel, J. Sohn, and J. Emerson, "The determination of interfacial tension by video image processing of pendant fluid drops," *Journal of colloid and interface science*, vol. 119, no. 1, pp. 55–66, 1987.

[32] A. Beghdadi and A. Le Negrate, "Contrast enhancement technique based on local detection of edges," *Computer vision, graphics, and image processing*, vol. 46, no. 2, pp. 162–174, 1989.

[33] R. Lippmann, "An introduction to computing with neural nets," *IEEE Assp magazine*, vol. 4, no. 2, pp. 4–22, 1987.

[34] T. Khanna, *Foundations of neural networks*. Addison-Wesley Longman Publishing Co., Inc., 1990.

[35] J. J. Hopfield, "Artificial neural networks," *IEEE Circuits and Devices Magazine*, vol. 4, no. 5, pp. 3–10, 1988.

[36] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE transactions on pattern analysis and machine intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[37] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.

[38] M. J. Lyons, J. Budynek, and S. Akamatsu, "Automatic classification of single facial images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 21, no. 12, pp. 1357–1362, 1999.

[39] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[40] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen, "Medical image classification with convolutional neural network," in *2014 13th international conference on control automation robotics & vision (ICARCV)*, pp. 844–848, IEEE, 2014.

[41] C. Ilie and M. Ilie, "Artificial neural networks in management and business administration," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

[42] A. Krenker, J. Bešter, and A. Kos, "Introduction to the artificial neural networks," *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech*, pp. 1–18, 2011.

[43] S. Agatonovic-Kustrin and R. Beresford, "Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research," *Journal of pharmaceutical and biomedical analysis*, vol. 22, no. 5, pp. 717–727, 2000.

[44] H. Deans and L. Lapidus, "A computational model for predicting and correlating the behavior of fixed-bed reactors: I. derivation of model for nonreactive systems," *AIChE Journal*, vol. 6, no. 4, pp. 656–663, 1960.

[45] M. Marvin and A. P. Seymour, "Perceptrons," *Cambridge, MA: MIT Press*, vol. 6, pp. 318–362, 1969.

[46] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[47] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, p. e00938, 2018.

[48] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[49] A. G. Ivakhnenko and V. G. Lapa, *Cybernetic predicting devices*. CCM Information Corporation, 1973.

[50] A. Ivakhnenko, *Cybernetics and forecasting techniques*.

[51] F. Attneave and M. D. Arnoult, "The quantitative study of shape and pattern perception.," *Psychological bulletin*, vol. 53, no. 6, p. 452, 1956.

[52] S. K. Reed, "Pattern recognition and categorization," *Cognitive psychology*, vol. 3, no. 3, pp. 382–407, 1972.

[53] W. Chong, D. Blei, and F.-F. Li, "Simultaneous image classification and annotation," in *2009 IEEE Conference on computer vision and pattern recognition*, pp. 1903–1910, IEEE, 2009.

[54] C. Samson, L. Blanc-Féraud, G. Aubert, and J. Zerubia, "A variational model for image classification and restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 5, pp. 460–472, 2000.

[55] N. K. Sarkar, M. M. Singh, and U. Nandi, "Learning based image classification techniques," in *International Conference on Computational Intelligence in Communications and Business Analytics*, pp. 28–44, Springer, 2022.

[56] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning*, pp. 161–168, 2006.

[57] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd international conference on Machine learning*, pp. 625–632, 2005.

[58] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," *Advances in neural information processing systems*, vol. 29, 2016.

[59] H. B. Barlow, "Unsupervised learning," *Neural computation*, vol. 1, no. 3, pp. 295–311, 1989.

[60] Z. Ghahramani, "Unsupervised learning," in *Summer school on machine learning*, pp. 72–112, Springer, 2003.

[61] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 8595–8598, IEEE, 2013.

[62] R. Gray, "Vector quantization," *IEEE Assp Magazine*, vol. 1, no. 2, pp. 4–29, 1984.

[63] P. Scheunders, "A genetic c-means clustering algorithm applied to color image quantization," *Pattern recognition*, vol. 30, no. 6, pp. 859–866, 1997.

[64] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society. series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[65] K. Krishna and M. N. Murty, "Genetic k-means algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 3, pp. 433–439, 1999.

[66] R. Rollet, G. Benie, W. Li, S. Wang, and J. Boucher, "Image classification algorithm based on the rbf neural network and k-means," *International Journal of Remote Sensing*, vol. 19, no. 15, pp. 3003–3009, 1998.

[67] W. S. Noble, "What is a support vector machine?," *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.

[68] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[69] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE transactions on systems, man, and cybernetics*, no. 4, pp. 580–585, 1985.

[70] S. A. Dudani, "The distance-weighted k-nearest-neighbor rule," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 4, pp. 325–327, 1976.

[71] N. R. Draper and H. Smith, *Applied regression analysis*, vol. 326. John Wiley & Sons, 1998.

[72] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," *Advances in neural information processing systems*, vol. 2, 1989.

[73] M. Nixon and A. Aguado, *Feature extraction and image processing for computer vision*. Academic press, 2019.

[74] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 6232–6251, 2016.

[75] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. D. Jackel, Y. LeCun, U. A. Muller, E. Sackinger, P. Simard, *et al.*, "Comparison of classifier methods: a case study in handwritten digit recognition," in *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3-Conference C: Signal Processing (Cat. No. 94CH3440-5)*, vol. 2, pp. 77–82, IEEE, 1994.

[76] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[77] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," *arXiv preprint arXiv:1412.6830*, 2014.

[78] B. Karlik and A. V. Olgac, "Performance analysis of various activation functions in generalized mlp architectures of neural networks," *International Journal of Artificial Intelligence and Expert Systems*, vol. 1, no. 4, pp. 111–122, 2011.

[79] 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, 2017.

[80] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[81] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[82] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[83] A. Aimar, H. Mostafa, E. Calabrese, A. Rios-Navarro, R. Tapiador-Morales, I.-A. Lungu, M. B. Milde, F. Corradi, A. Linares-Barranco, S.-C. Liu, *et al.*, "Nullhop: A flexible convolutional neural network accelerator based on sparse representations of feature maps," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 3, pp. 644–656, 2018.

[84] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, "Object detection networks on convolutional feature maps," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1476–1481, 2016.

[85] L. Bottou *et al.*, "Stochastic gradient learning in neural networks," *Proceedings of Neuro-Nımes*, vol. 91, no. 8, p. 12, 1991.

[86] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*, pp. 240–248, Springer, 2017.

[87] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[88] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[89] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[90] K. I. Forster and E. S. Bednall, "Terminating and exhaustive search in lexical access," *Memory & Cognition*, vol. 4, no. 1, pp. 53–61, 1976.

[91] G. B. Coleman and H. C. Andrews, "Image segmentation by clustering," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 773–785, 1979.

[92] J. Schmidt-Hieber, "Nonparametric regression using deep neural networks with relu activation function," *The Annals of Statistics*, vol. 48, no. 4, pp. 1875–1897, 2020.

[93] R. Padilla, S. L. Netto, and E. A. Da Silva, "A survey on performance metrics for object-detection algorithms," in *2020 international conference on systems, signals and image processing (IWSSIP)*, pp. 237–242, IEEE, 2020.