REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Information Technologies

ARABIC SIGN LANGUAGE RECOGNITION BASED ON ARTIFICIAL INTELLIGENCE TECHNIQUES

Mustafa Raad Sabri SABRI

Master's Thesis

Supervisor

Asst. Prof. Dr. Oğuz KARAN

İstanbul, 2023

ARABIC SIGN LANGUAGE RECOGNITION BASED ON ARTIFICIAL INTELLIGENCE TECHNIQUES

Mustafa Raad Sabri SABRI

Information Technologies

Master’s Thesis

ALTINBAŞ UNIVERSITY

2023

The thesis titled ARABIC SIGN LANGUAGE RECOGNITION BASED ON ARTIFICIAL INTELLIGENCE TECHNIQUES prepared by MUSTAFA RAAD SABRI SABRI and submitted on 29/12/2023 has been **accepted unanimously** for the degree of Master of Science in Information Technologies.

Asst. Prof. Dr. Oğuz KARAN

Supervisor

Thesis Defense Committee Members:

Asst. Prof. Dr. Oğuz KARAN	Department of Software Engineering, Altınbaş University	_____
Assoc. Prof. Dr. Sefer KURNAZ	Department of Computer Engineering, Altınbaş University	_____
Asst. Prof. Dr. Serdar KARGIN	Department of Biomedical Engineering, İstanbul Arel University	_____

I hereby declare that this thesis meets all format and submission requirements for a Master's thesis.

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Mustafa Raad Sabri SABRI

Signature

ABSTRACT

ARABIC SIGN LANGUAGE RECOGNITION BASED ON ARTIFICIAL INTELLIGENCE TECHNIQUES

SABRI, Mustafa Raad Sabri

M.Sc., Information Technologies, Altınbaş University,

Supervisor: Asst. Prof. Dr. Oğuz KARAN

Date: December / 2023

Pages: 75

Arabic Sign Language is the means of communication utilised between deaf individuals. It's a collection of body, hand movements, and facial expressions. SL is not understood by hearing people, which creates a gap in communication at the same time, causing Significant difficulties for deaf individuals' daily lives and affecting their careers and the ability to have a good life. Artificial intelligence and deep learning technologies have developed computer vision in recent years. SL is one of the Significant research projects in computer vision. We propose a real-time Recognition system for Arabic SL alphabets, numbers, and words to fill the gap between hearing and deaf individuals by creating three datasets containing thousands of images taken by five volunteers in various places, lighting conditions, hands, and backgrounds. The training and testing of the model are done by utilising different weights of the deep learning model Yolov8, the last model in the Yolo models, and it achieves an accuracy of approximately 98%, and the experiment shows efficiency and speed in detection.

Keywords: Arabic Sign Language Recognition, YOLO, Deep Learning, Convolutional Neural Network, Computer Vision.

TABEL OF CONTENTS

Pages

ABSTRACT ..... **v**

LIST OF TABLES ..... **viii**

LIST OF FIGURES ..... **ix**

ABBREVIATIONS ..... **xii**

LIST OF SYMBOLS ..... **xiii**

1. INTRODUCTION **1**

1.1 OVERVIEW 1

1.2 PROBLEM STATEMENT 2

2. RELATED RESEARCH REVIEW **3**

2.1 ARABIC SIGN LANGUAGE (ARSL) 3

2.2 ARABIC SIGN LANGUAGE RECOGNITION SYSTEM 4

2.2.1 Vision-Based Recognition (VBR) and Sensor-Based Recognition (SBR) 5

2.2.2 Static Gestures and Dynamic Gestures 5

2.3 COMPUTER VISION TECHNIQUES 7

2.3.1 Deep Learning (DL) 7

2.3.2 Convolutional Neural Network (CNN) 8

2.3.3 Object Detection 16

2.3.4 YOLO Method 19

3. METHODOLOGY **34**

3.1 DATASET ACQUISITION 34
3.1.1 Collecting Data 34
3.1.2 Augmentation and Annotation 37
3.1.3 Spilt Dataset 39
3.2 PROPOSED MODEL 39
3.2.1 YOLOv8 Architecture 39
3.2.2 Training Based YOLOv8 Model 42
4. RESULT AND DISCUSSION 44
4.1 ALPHABETS MODEL RESULTS 44
4.2 NUMBERS MODEL RESULTS 47
4.3 WORDS MODEL RESULTS 50
5. CONCLUSION AND FUTURE WORK 54
REFERENCES 56

LIST OF TABLES

Pages

Table 3.1: Empirical Hyperparameter Values for Alphabet, Words, and Numbers of Model, ... 43

Table 4.1: The Performance of Yolov8 Models on ASL Alphabets. 44

Table 4.2: Results of Performance Evaluation For 32 Alphabets Models. 45

Table 4.3: Evaluation Results of YOLOv8m for 32 Alphabets Signs. 45

Table 4.4: Performance of The Yolov8 Model on ASL Numbers. 47

Table 4.5: Results of Performance Evaluation Numbers Models. 48

Table 4.6: Evaluation Results of YOLOv8m for 10 Numbers Signs. 48

Table 4.7: Performance of Yolov8 Model on ASL Words..... 51

Table 4.8: Evaluation Results of Yolov8m For 19 Words Signs. 51

Table 4.9: Results of Performance Evaluation Words Models. 52

LIST OF FIGURES

Pages

Figure 2.1: The Most Spoken Language in The World. 3

Figure 2.2: The Categorization of Approaches for Recognizing ArSL [15]. 5

Figure 2.3: Example of Static and Dynamic Gesture [15]. 6

Figure 2.4: CNN Basic Models and Derived Models [23]. 8

Figure 2.5: CNN Architecture for Image Classification [23]. 9

Figure 2.6: Receptive Neuron Field in The Next Layer [24]. 10

Figure 2.7: The Pooling Procedure Is Performed by Picking A 2 X 2 Window [24]. 11

Figure 2.8: The Architecture of Lenet-5, Is Characterized by Distinct Feature Maps Represented by Each Box [31]. 13

Figure 2.9: AlexNet Architecture [38]. 14

Figure 2.10: Object Detection Can Be Classified into Two Main Categories: General Object Detection and Specialized Object Detection [46]..... 17

Figure 2.11: The Actual Applying of Different Image Techniques [46]..... 17

Figure 2.12: Classical, Traditional Object Detection and Object Detection Based on DCNNs [46]. 19

Figure 2.13: A Timeline of YOLO Versions [52]. 19

Figure 2.14: The Computation of IoU Entails the Division of The Intersection of Two Boxes By Their Union. The Provided Illustrations Showcase Three Distinct IoU Values, Each Corresponding to Various Bounding Box Positions [52]. 22

Figure 2.15: The Provided Diagram Depicts a Rudimentary YOLO Model That Showcases a Grid Structure of Dimensions Three-By-Three. This Model Is Designed to Classify Objects into Three Distinct Classes. Each Grid Element in The Diagram Is Associated with A Single Class Prediction, Resulting in A Vector Including Eight Values [52]. 23

Figure 2.16: Anchor Boxes: For Every Grid Cell, Yolov2 Creates Numerous Anchor Boxes [52]. 25

Figure 2.17: Bounding Box Prediction: To forecast bounding boxes, use the predicted T_x and T_y values. These values run through a sigmoid function and are offset by the grid cell's position (C_x, C_y). This yields the box's centre coordinates. The previous width P_w is used to calculate the height and breadth of the final box [52]. 25

Figure 2.18: YOLOv3 Darknet-53 Backbone [52]..... 27

Figure 2.19: YOLOv3 Multi-Scale Detection Architecture [52] 28

Figure 2.20: YOLOv4 Architecture [52]. 29

Figure 2.21: YOLOv5 Architecture [52]. 30

Figure 2.22: YOLOv6 Architecture [80]. 32

Figure 2.23: YOLOv7 Architecture[83]. 33

Figure 3.1: The Alphabets ArSL Dataset. 35

Figure 3.2: The Numbers 0 To 9 In the ArSL Numbers Dataset..... 36

Figure 3.3: The Selected Words in The ArSL Word Dataset. 37

Figure 3.4: Labelimg [87]..... 38

Figure 3.5: Two Images from The Number's Dataset. 38

Figure 3.6: Detailed Illustration of Yolov8 Architecture. The Backbone, Neck, and Head Are the Three Parts of Our Model, And C2f, ConvModule, DarknetBottleNeck, And SPPF Are Modules [89]. 41

Figure 4.1: Depicts the Metrics Achieved After Completing the Training and Validation Processes Over 200 Epochs. 46

Figure 4.2: ArSL Alphabets of Recognition Results on Test Images. 47

Figure 4.3: Depicts the Metrics Achieved After Completing the Training and Validation Processes Over 200 Epochs. 49

Figure 4.4: ArSL Numbers of Recognition Results on Test Images. 50

Figure 4.5: Depicts the Metrics Achieved After Completing the Training and Validation Processes Over 200 Epochs. 52

Figure 4.6: ArSL Words of Recognition Results on Test Images. 53

ABBREVIATIONS

AI	:	Artificial Intelligence
ArSLR	:	Arabic Sign Language Recognition
DL	:	Deep Learning
ML	:	Machine Learning
CV	:	Computer Vision
CNN	:	Convolutional Neural Network
YOLO	:	You Only Looks Once
LAS	:	League of Arab States
SLR	:	Sign Language Recognition
NPL	:	Natural Language Processing
HCI	:	Human Interaction with Computers
ArSL	:	Arabic Sign Language
AP	:	Average Precision
mAP	:	Mean Average Precision
SL	:	Sign Language
IoU	:	Intersection Over Union
ANN	:	Artificial Neural Network
ReLU	:	Rectified Linear Unit

LIST OF SYMBOLS

σ : The Sum of Multiple Items

1. INTRODUCTION

1.1 OVERVIEW

Sign Language (SL) serves as a means of communication between individuals. It is critical in the lives of those with speech and hearing impairments, often viewed as their primary mode of communication; this language relies on manual gestures to convey nonverbal messages [1]. According to World Health Organization (WHO) data, more than 5% of the global population, equivalent to 430 million individuals (comprising 432 million adults and 34 million children), require rehabilitation to address their significant hearing impairments, Projections indicate that by the year 2050, this number will surge to over 700 million people, representing 1 in 10 individuals worldwide, experiencing disabling hearing loss, 'disabling' hearing loss denotes a hearing impairment exceeding 35 decibels (dB) in the better-hearing ear, nearly 80% of those affected by such hearing loss reside in low- and middle-income countries; additionally, the prevalence of hearing loss escalates with age, with over 25% of individuals above the age of 60 grappling with disabling hearing loss [2].

ArSL encounters a scarcity of essential resources, such as standardised dictionaries and linguistic databases. This shortage of resources hinders the advancement and comprehension of ArSL as a unique and independent language. Unified dictionaries present a formidable obstacle for deaf individuals and researchers, impeding their access to a comprehensive and universally accepted resource for ArSL signs [3] Modern technological progress assists individuals with deaf disabilities in surmounting numerous communication challenges. Contemporary breakthroughs in deep learning and computer vision models are revolutionising human interaction by enabling unprecedented capabilities, and the Arabic language remains a fertile field for research [4]. ArSL hasn't enough research like American sign language (ASL), and it's still in an early stage of development, especially in Deep Learning (DL) and Machine Learning (ML) techniques. This research proposes a system that employs deep learning-based object detection models to recognise ArSL in real-time.

1.2 PROBLEM STATEMENT

As reported by the Centre for Strategic and International Studies in Washington DC, there are approximately 11 million individuals in the Middle East with disabling hearing loss; however, the absence of a standardised ArSL has led to a shortage of proficient SL interpreters for example in Saudi, this ratio is one for every 93,000 individuals who can bridge the communication gap between the deaf community and the broader world [5].

Effective communication is a vital skill for humans to engage with their surrounding communities; without it, sharing experiences, expressing thoughts, and conveying emotions become formidable challenges; communication serves as the conduit for discussing various matters that affect humans, enabling them to address these issues and devise suitable solutions to enhance their daily lives; furthermore, communication plays a pivotal role in an individual's mental well-being [6]. Individuals with hearing impairments face challenges in communicating with others because they are required to acquire SL skills, which serve as their primary means of interacting with the community; the effectiveness of image-based solutions for this issue relies heavily on the quality of segmentation and the selection of features that accurately represent the critical visual characteristics of SL gestures [7].

SL is the predominant communication mode for individuals with hearing impairments. To facilitate communication with these individuals, it is essential for those who are not hearing impaired to have the ability to understand and recognise SL. Consequently, there is a need to implement a SLR system to aid individuals with hearing impairments by creating a deep learning real-time recognition model to convert the SL to text to fill up the gap between deaf people and hearing people, that is what we will discuss in the proposal.

2. RELATED RESEARCH REVIEW

2.1 ARABIC SIGN LANGUAGE (ARSL)

Arabic is the sixth spoken language with more than 274 million [8], as shown in Figure 2.1. The Arabic language is renowned for its extensive vocabulary and abundance of synonyms for individual words. Conversely, ArSL possesses a more limited vocabulary. As a result, SL interpreters frequently omit specific words, prefixes, and suffixes from Arabic sentences. This selective translation approach focuses solely on conveying the essential elements of a sentence, ensuring that the intended meaning is effectively communicated to the target audience [9]. Arabic is a prevalent language throughout the Middle East, and ArSL exhibits distinct variations, each corresponding to a specific country in the region.

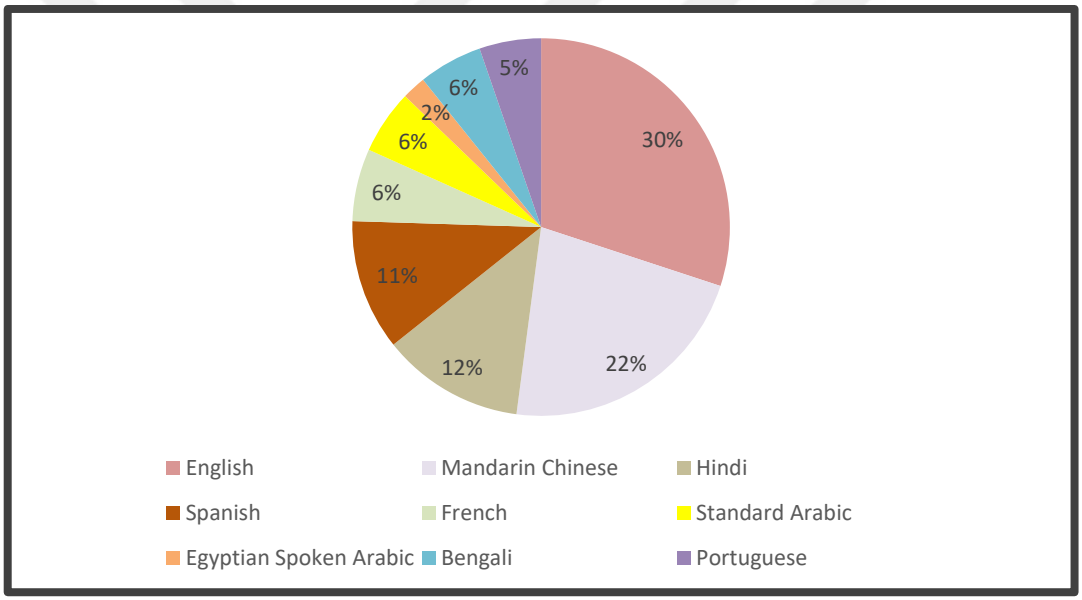


Figure 2.1: The Most Spoken Language in The World.

SL comprises diverse visual cues extending beyond hand and finger gestures, incorporating eye movements, facial expressions, lip movements, eyebrows, and other non-manual markers. Facial expressions are crucial in effectively communicating subtle messages [10]. Despite being commonly perceived as more formal, SL could convey animated and expressive communication, integrating life with artistic essence using hand movements, body language, and facial expressions. In contrast to conventional spoken language, SL showcases its distinct creative and refined attributes [10]. The publication of the first

segment of the unified dictionary of ArSL was achieved in 1999 through the collaboration of The League of Arab States (LAS) and the Arab League Educational, Cultural and Scientific Organization (ALECSO). The project aimed to establish a standardised version of ArSL. In 2007 the second dictionary segment was published [11]. Deaf individuals living in Arabic countries experience plenty of challenges due to the limited availability of services tailored to their needs. These challenges manifest in several forms, including limited access to essential information and educational opportunities, difficulties communicating with the hearing community, and limited participation in community events [3].

2.2 ARABIC SIGN LANGUAGE RECOGNITION SYSTEM

Sign Language Recognition (SLR) models can be broadly classified into two main categories: vision-based techniques and sensor glove-based approaches [12]. Sensor glove-based approaches refer to a technique that requires direct physical interaction between sensor devices and users. This approach often involves using instrumented gloves with various sensors such as electromyography, inertial measurements, or electromagnetic sensors [12]. These sensors are utilised to capture data related to finger positions, flexion, direction, or angles during the execution of SL gestures; in contrast, the vision-based approach is dependent on data obtained by streams life or photos acquired by a camera as the input for the system [12]. The preference towards vision-based methodologies often stems from the obvious advantage of not requiring specialised gloves or supplementary equipment beyond the camera for the recognition procedure [12]. Coloured gloves were utilised in specific vision-based methodologies to streamline the process of hand segmentation. However, vision-based approaches face notable problems, including the impact of elements such as lighting conditions, noise, differences in perspective, and complex backgrounds on the accuracy of the results[12]. The field of hand gesture recognition has garnered significant interest from many scholars, mostly because of its extensive array of potential uses, encompassing fields such as robotics, gaming, virtual reality, SL, and human-computer interaction [13]. SL, a systematised that mainly relies on manual gestures, is considered the most efficient mode of communication for those with hearing difficulties. The effective recognition of dynamic, isolated SL motions requires dealing with three primary challenges: hand segmentation, encoding of hand shape information, and recognising gesture sequences [13]. Traditional approaches to SL identification often utilise colour-based algorithms to

segment the hand, manually engineered techniques to extract hand shape data and the application of Hidden Markov Models (HMM) for sequence recognition [13].

2.2.1 Vision-Based Recognition (VBR) and Sensor-Based Recognition (SBR)

The VBR strategy is more commonly utilised in ArSLR than the SBR approach. Furthermore, numerous study endeavours have attempted to explore how to recognise the Arabic alphabet and isolated words in SL, compared to comparing continuous SLR in Arabic [14]. The illustration is shown in Figure 2.1.

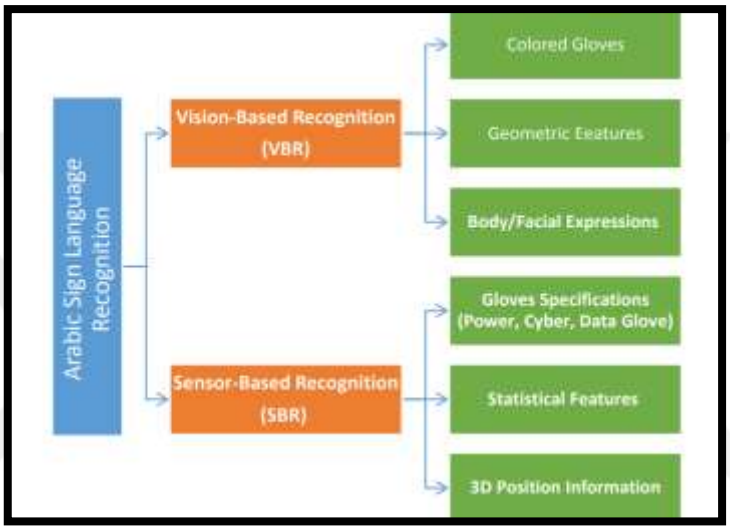


Figure 2.2: The Categorization of Approaches for Recognizing ArSL [15].

2.2.2 Static Gestures and Dynamic Gestures

The exact positioning and alignment of the hands are crucial factors in the refinement of static gestures, which refer to the many hands poses and postures, and dynamic gestures, which involve hand movements such as waving within a defined space and time, all achieved without any physical movement. Moreover, static gestures can involve a solitary hand position without associated bodily movement [16]. Static gestures involve the utilisation of one or more frames of signals as input, while dynamic gestures operate by utilising a continuous sequence of frames extracted from a video. The interpretation of static gestures primarily relies on the angular orientation and finger form, with the hand remaining immobile throughout the whole duration of the motion, as shown in Figure 2.3; on the other hand, dynamic hand gestures exhibit continuous movement of the hand's position, with their significance conveyed through a series of distinct phases such as the stroke phase, retraction,

and preparation; each of these phases is characterised by unique movements that occur over some time [17].

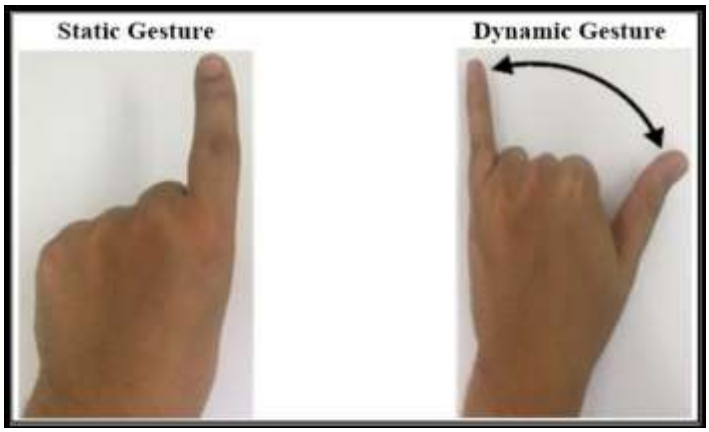


Figure 2.3: Example of Static and Dynamic Gesture [15].

Various properties, including orientations, finger shape, flex angles, relative location to the body, and contextual surrounds, can characterise static gestures. On the other hand, dynamic gestures exhibit distinguishing attributes such as orientations, finger configuration, flexion angles, hand paths, hand orientations, motion velocity and direction, and scale [18].

Accurately recognising gestures in ArSL involves four main phases. The four processes involved in this procedure are 1) data collecting, 2) pre-processing and segmentation, 3) feature extraction, and 4) classification. The phases are essential components of both the conventional SLR and Automatic Sign Language Recognition (ASLR) models [19]. In the first step, an appropriate input device is employed to capture the image of the hand; following this, the image is subjected to a segmentation process to ascertain the precise position of the hand with the adjacent body components while being distinguished from the background; the location image is after that subjected to several processing steps aimed at noise reduction, contour identification, and model generation [19]. classification: re-processing of the photos and movements, the subsequent phase involves feature extraction; this step entails the extraction of data of the shape, position, orientation, movements, and hand placements for the classification; after doing analysis and employing modelling techniques, it has been determined that the acquired images can be classified as appropriate gestures; SLR can be classified into three distinct levels within a broader framework; these levels comprise the recognition of SL alphabets, isolated words, and continuous sentences [19].

There are two primary classifications for SLR systems: isolated and continuous. In the context of isolated SLR, it is standard procedure to interpret each image as a representation of a unique letter of the alphabet, a numerical value, or a distinct motion. In contrast, continuous SLR is specifically developed to enable uninterrupted and continuous signing, and it provides the 999capability to identify and translate entire sentences accurately [20]. SLR classification employs a range of methodologies, including Hidden Markov Models (HMM), CNN, and Artificial Neural Networks (ANN) [20].

2.3 COMPUTER VISION TECHNIQUES

2.3.1 Deep Learning (DL)

Deep learning is a branch of machine learning distinguished by implementing neural networks consisting of three or more layers. The objective of these neural networks is to emulate the cognitive processes and activities observed in the human brain, hence facilitating their ability to acquire and enhance their capabilities through exposure to vast amounts of data [21], [22]. Although a single-layer neural network can reach a basic level of predictive capability, the potential of deep learning networks is greatly enhanced by their increased depth [22]. Deep learning encompasses a wide array of applications in artificial intelligence, facilitating the enhancement of automation and the accomplishment of intricate analytical tasks without requiring direct human involvement. In addition to implementing sophisticated technologies such as autonomous vehicles, deep learning methodologies are also widely employed in diverse services and applications, including mobile device digital assistants, voice-controlled devices, and the identification of credit card fraud, among other pragmatic applications [22].

The comparative analysis of the efficacy of CNN, Restricted Boltzmann Machines (RBMs), Autoencoders, and Sparse Coding in computer vision (CV) tasks, it was finally determined that CNNs exhibited the highest level of suitability as an architectural choice [23]. Nevertheless, within that specific timeframe, significant obstacles arose because of constraints in accuracy and the dimensions of the models and the hurdles encountered encompassed several aspects: firstly, the absence of comprehensive understanding of the superior performance of specific architectures; secondly, the limitations imposed by training with a restricted dataset; thirdly, the complexities associated with attaining real-time

applications; and finally, the imperative need for more robust models [23]. The study [23] investigates the latest deep learning (DL) developments and examines eight emerging techniques established as fundamental models in different computer vision (CV) application fields. The broad range of deep learning applications can be categorised into four main use cases: recognition, visual tracking, semantic segmentation, and image restoration, as shown in Figure 2.4.

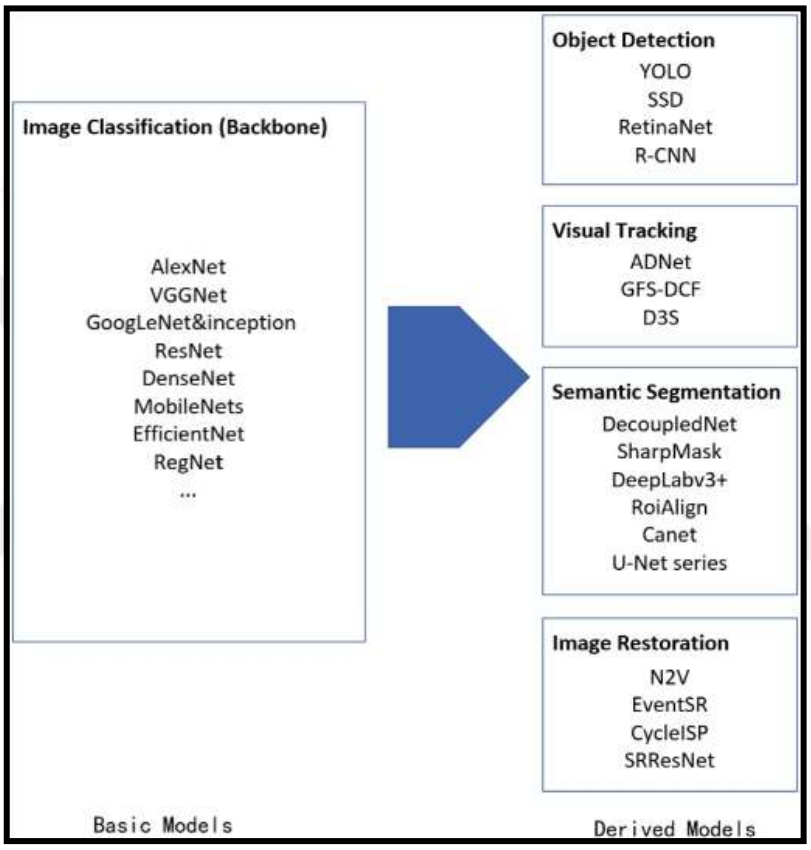


Figure 2.4: CNN Basic Models and Derived Models [23].

2.3.2 Convolutional Neural Network (CNN)

The CNN deep learning methodology extensively employed to tackle complex problems, demonstrating superior performance compared to traditional machine learning approaches [24]. CNNs present a viable alternative methodology for the autonomous acquisition of domain-specific features, so as a result, this unique approach has triggered a reassessment of all aspects within the wider domain of computer vision [24]. CNNs have gained prominence as major deep learning methodologies following their remarkable performance

in the ImageNet competition, and Image classification is considered one of the crucial domains in the realm of computer vision. The CNN, also known as ConvNet, is characterised by its deep feed-forward architecture and demonstrates exceptional generalisation skills, surpassing networks that utilise fully connected layers [26]. CNNs are designed using the underlying concept of hierarchical feature identification, drawing inspiration from biological principles [27]. They demonstrate exceptional proficiency in acquiring complex conceptual attributes and identifying various entities [24]. There are several compelling justifications for adopting CNN over alternative conventional methods. The primary benefit of CNN is in their utilisation of weight sharing, a technique that effectively decreases the parameters that need to be trained, and this reduction in parameters leads to improved generalisation capabilities of the network [28].

decreases the parameters enables CNN to undergo effective training while mitigating the risk of overfitting [29]. The architecture employed by CNN for image classification encompasses convolutional layers, pooling layers, and fully connected layers. The convolutional layers of a neural network employ several kernels to conduct convolutions on the entire picture, resulting in the creation of intermediate feature maps that possess a wide range of distinct features. Pooling layers are utilized in CNNs to decrease the dimensions of feature maps, hence simplifying the data representation. On the other hand, fully connected layers, positioned at the end of the CNN architecture, function as a classifier. The interwoven layers that are formed as a result facilitate efficient classification of images [23]. The architecture of the CNN is illustrated in Figures 2.5.

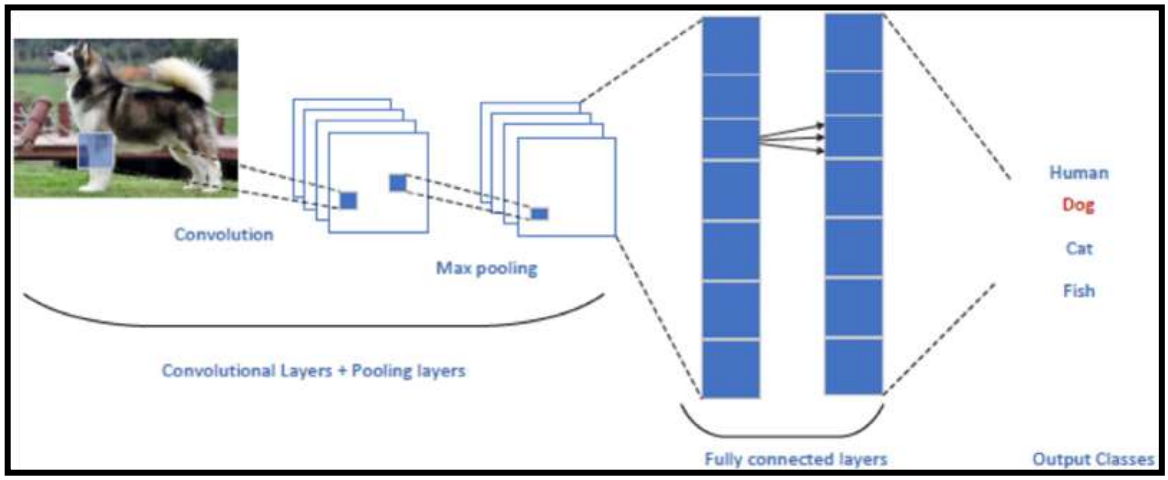


Figure 2.5: CNN Architecture for Image Classification [23].

a. Convolutional Layers: Structure Explanation

i. Convolutional Layer

The procedure begins by feeding an image to the initial layer and generating an output with a predicted class. This prediction is made based on the analysis of features retrieved from the image [30]. A singular neuron forms connections within the subsequent layer with a specific selection of neurons in the preceding layer; this localised connecter is commonly called the "receptive field" [26]. Utilising the receptive field allows for extracting local details from the input image [31]. Each neuron's receptive field, inked to a distinct area in the preceding layer, constitutes a weight vector that remains constant across all sites inside the plane [32]. In this context, the term "plane" pertains to the neurons in the subsequent layer. Sharing identical weights among neurons within the same plane enables them to recognize comparable characteristics at distinct positions within the supplied data [33]. The visual representation of this concept is illustrated in Figure 2.6.

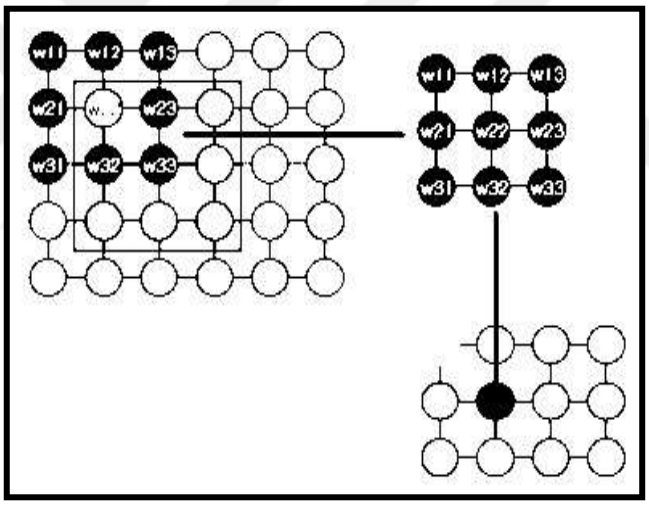


Figure 2.6: Receptive Neuron Field in The Next Layer [24].

The weight vector, a filter or kernel, traverses the input vector to generate the feature map [34]. The function of moving the filter in horizontal and vertical directions is called the convolution process [24]. Various characteristics are Based on the provided picture inside a singular layer, creating filters and N feature maps [33]. Each of these feature maps represents unique and different features. Utilising the local receptive field results in a notable reduction in trainable parameters [35]. The computation of the output value a_{ij} in the succeeding layer at a certain place (i, j) is determined by executing the convolution.

operation, utilising the formula specified in reference [36], as depicted in equation 2.1.

$$a_{ij=}\,\sigma\left((W*X)_{ij}+b\right) \tag{2.1}$$

In the provided context, X denotes the input supplied to the layer, W stands for the filter or kernel traversing the input, b represents the bias term, signifies the convolution operation, and σ indicates the non-linear element incorporated within the network [24].

ii. Pooling Layer

Once a particular characteristic has been determined, the exact positioning of this characteristic becomes less crucial [31]. Consequently, following the convolutional layer, it is customary to include a pooling or sub-sampling layer [32]. One of the main advantages of using the pooling technique is the significant decrease in the number of trainable parameters and the incorporation of translation invariance [35]. The pooling procedure is performed by selecting a window and applying a pooling function to the input items within that window [24], as depicted in Figure 2.7.

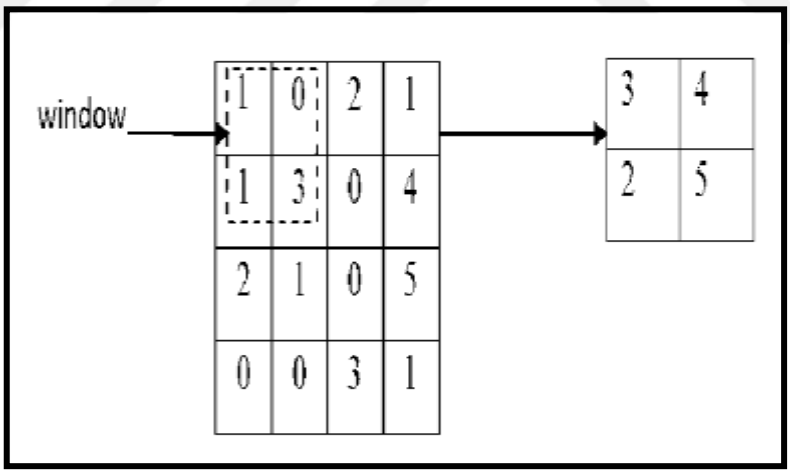


Figure 2.7: The Pooling Procedure Is Performed by Picking A 2 X 2 Window [24].

The pooling operation creates an additional output vector [24]. Other pooling strategies can be utilised, such as average pooling and max pooling, and among these methods, max pooling is commonly preferred due to its significant reduction in map size [34]. When

calculating mistakes, it is crucial to acknowledge that the error is not communicated back to the successful group in max-pooling, as it does not engage in the forward flow [24].

iii. Fully Connected Layer

The fully connected layer, like conventional models, accepts the output generated by the first convolution and pooling procedures. The computation entails performing a dot product operation between the weight vector and the input vector [35]. Gradient descent, sometimes referred to as batch mode learning or an offline technique, aims to minimize the cost function by estimating costs throughout the complete training dataset. The aforementioned methodology involves the adjustment of parameters after the completion of dataset processing within a single period. Although this approach is successful in achieving global minima, it results in extended training periods when dealing with huge datasets. To address this issue, the introduction of stochastic gradient descent emerged as an alternate method for minimizing the cost function.

iv. Activation Function

Using the Rectified Linear Unit (ReLU) has demonstrated more benefits than its predecessor, primarily attributed to two crucial features. To begin with, computing the partial derivative of the ReLU function is quite simple [35]. Furthermore, when considering the importance of training time, the non-saturating non-linearities denoted by $fx = max\ (0, X)$, such as ReLU, exhibit superior performance in terms of speed compared to saturating non-linearities like the sigmoid function, denoted by $f(x) = (1 + e - x - 1)$ [29]. Thirdly, the ReLU serves to mitigate the issue of vanishing gradients; however, the efficacy of the ReLU reduces in situations when a significant gradient propagates across the neural network, resulting in the phenomenon known as Dying ReLU, wherein neurons fail to activate. One potential approach to address this concern is the utilisation of the Leaky ReLU activation function [24]. The punctured ReLU operates in the following manner: for values of x > 0, the activation function is defined as $f(x) = x$; however, for values of x < 0, the activation function is defined as αx, where α denotes a tiny constant [24].

b. Common CNN Architectures

i. LeNet Architecture

Multi-layer networks demonstrate high suitability to the picture identification tasks according to the ability learn from intricate and high-dimensional input [24]. The LeNet architecture, introduced in 1998 by [31], is an architectural design that incorporates a dataset [37] and is briefly described in the subsequent paragraph. The architecture of LeNet5, as depicted in Figure 2.8, comprises eight layers, encompassing five convolutional layers and three fully linked layers, and each component within a two-dimensional space possesses a total of 25 distinct inputs [24]. The units within the initial hidden layer get input from a 5x5 region corresponding to a subset of the complete image; hence, a limited segment of the initial picture is conveyed to the initial hidden layer, and specific input region denoted as the unit's receptive field. All units within a plane possess an identical weight vector, and the result of a unit is saved at an identical index within the feature map [24].

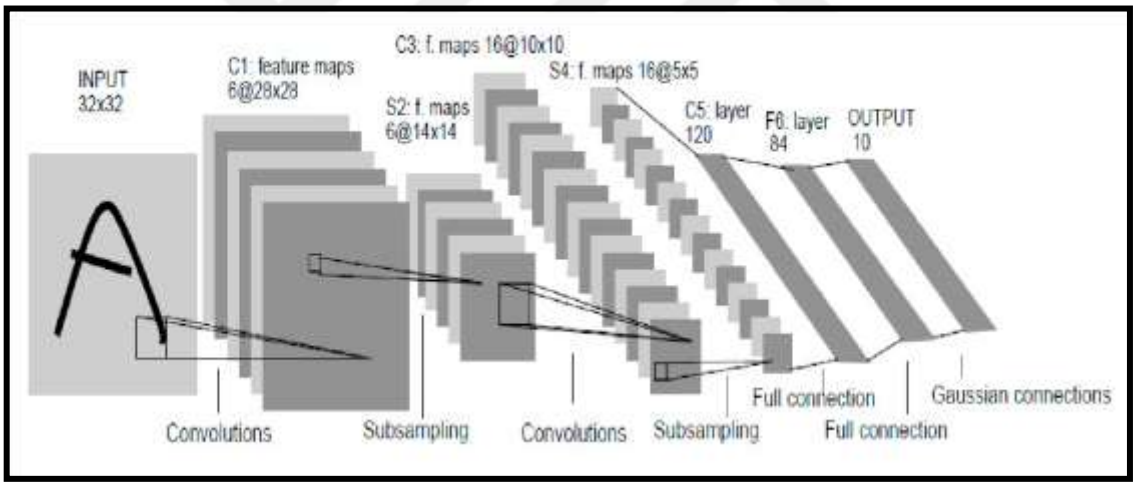


Figure 2.8: The Architecture of Lenet-5, Is Characterized by Distinct Feature Maps Represented by Each Box [31].

Adjacent units in the previous layer have an impact on neighbouring units in the feature map, resulting in the formation of overlapping receptive fields. The first convolutional layer, as seen in Figure 2.8, utilizes a 5x5 area to produce a variety of feature maps using horizontal shifting, hence facilitating various feature extraction [24]. CNNs have the capacity to preserve the stability of feature maps even in the face of tiny alterations in the input. This characteristic allows CNNs to prioritize the overall existence of features rather than their

precise location, which can result in a reduction in accuracy during the process of subsampling. The second layer, seen in Figure 2.8, illustrates the process of subsampling, which leads to the generation of feature maps equal in number to those produced by convolution. The subsampling layer employs a 2x2 area to compute the mean of four input values. This mean is then multiplied by a coefficient that can be adjusted by training, followed by the addition of a trainable bias. The resulting value is subjected to a sigmoid function and thereafter propagated through hierarchical layers. The procedure involves the augmentation of feature maps while simultaneously reducing spatial resolution, a process that is enabled by the utilization of the backpropagation algorithm for the acquisition of knowledge [24].

ii. AlexNet Architecture

Figure 2.9 briefly summarises the modified iteration of the LeNet architecture, commonly referred to as AlexNet, as originally presented by reference [29].The structure of AlexNet comprises five convolutional layers and three fully connected layers [24]. The resulting outputs are further fed into a SoftMax layer with 1000 nodes, aiming to classify 1.2 million high-resolution photo datasets into 1000 unique categories.

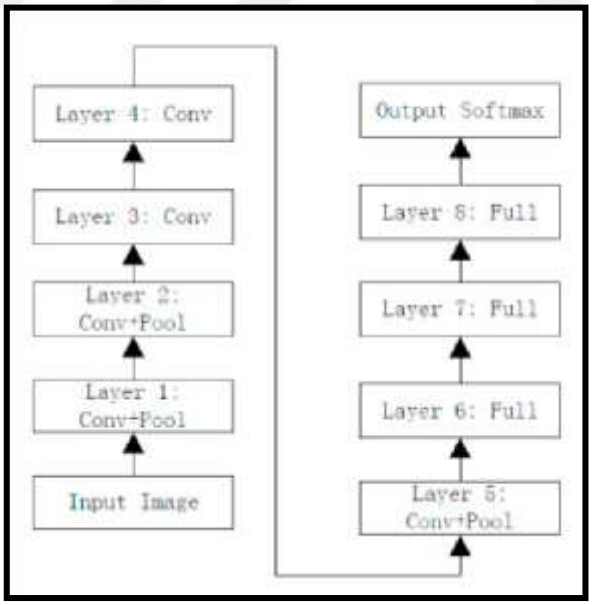


Figure 2.9: AlexNet Architecture [38].

To improve the network's speed training, the utilisation of non-saturating neurons is combined with the utilisation of efficient GPUs. To facilitate the classification of objects

within an extensive dataset comprising millions of photos, utilising a substantial network becomes imperative [24]. Nevertheless, this phenomenon might give rise to a significant need for weight training, perhaps resulting in overfitting concerns. To address this difficulty, the dropout approach was implemented; this strategy entails the temporary inactivation of neurons with a probability of 0.5 throughout both the forward and backward propagation stages, effectively attenuating their activity [24]. The neurons that rely on these attenuated neurons are motivated to acquire more resilient characteristics autonomously, thereby substantially reducing overfitting. It is important to acknowledge that utilising the dropout approach effectively results in a twofold increase in the number of iterations necessary for achieving convergence. In network training, a pair of GTX 580 3GB GPUs is employed, with the normal duration of the operation ranging from five to six days. An important characteristic of this architectural design is the integration of the ReLU non-linearity into the convolutional neural network, significantly improving the convergence rate [38].

iii. GoogleNet Architecture

GoogleNet model and originally proposed by [39], developed to design a more economical model that could effectively minimise power consumption, the number of trainable parameters, and memory utilisation. This model led to a significant reduction in the overall count of trainable parameters in the neural network. The overarching structure can be briefly described as follows: it efficiently utilises 12 million fewer parameters compared to the model put forward by [29]. The primary goal of this structure was to construct a network that possesses enhanced precision in object recognition within photographs; one potential method for achieving this objective is augmenting the network's dimensions and incorporating additional layers [24]. However, a significant limitation of this strategy is the possibility of an escalated parameter count, which may result in the issue of overfitting.

A further obvious limitation emerges when the quantity of filters is augmented, as this leads to elevated computational demands and, consequently, heightened overhead; to tackle this matter, a potential resolution has been put up which entails the utilisation of sparse matrices [24]. In this approach, units that exhibit a high degree of correlation are grouped in the preceding layer, thereby furnishing input to the subsequent layer. As the reference outlines, this technique aims to build an ideal network topology [39]. It is important to mention that

non-uniform sparse matrices can also be utilised. However, despite their potential to reduce calculations by a significant factor of up to 100, the issue of cache misses remains an overhead [24]. Despite their ability to enhance computational speed, the utilisation of extensively optimised numerical libraries fails to alleviate this concern. Hence, the prevailing approach in the field mostly depends on using uniform sparse matrices [24].

2.3.3 Object Detection

The human beings possess limited capabilities when it comes to effectively processing an extensive volume of visual input; as a result, there is an increasing demand for automated handling of this data through the utilisation of computers to tackle visual obstacles on a wide scale [40]. As the level of comprehension about image processing technology advances, it becomes increasingly important to possess an in-depth knowledge of images and the accurate detection of objects inside them [41]. There has been a shift in people's interests from basic picture categorisation to a more exact determination of the semantic category of items inside an image and the identification of their distinctive locations [42]. As a result, there has been considerable focus on object-detecting technology [43]. Object detection technology uses concepts and techniques derived from image processing and pattern recognition to discover and identify items within an image [40]. This involves determining the semantic categories of these objects and accurately determining their precise locations [44]. Object detection is a significant computer vision task that involves the identification of instances of Object categories inside digital images, such as persons, animals, or vehicles [45]. Object detection is an important field within computer vision important in scientific research and practical industrial applications. Particularly notable cases can be found in Figure 2.10.

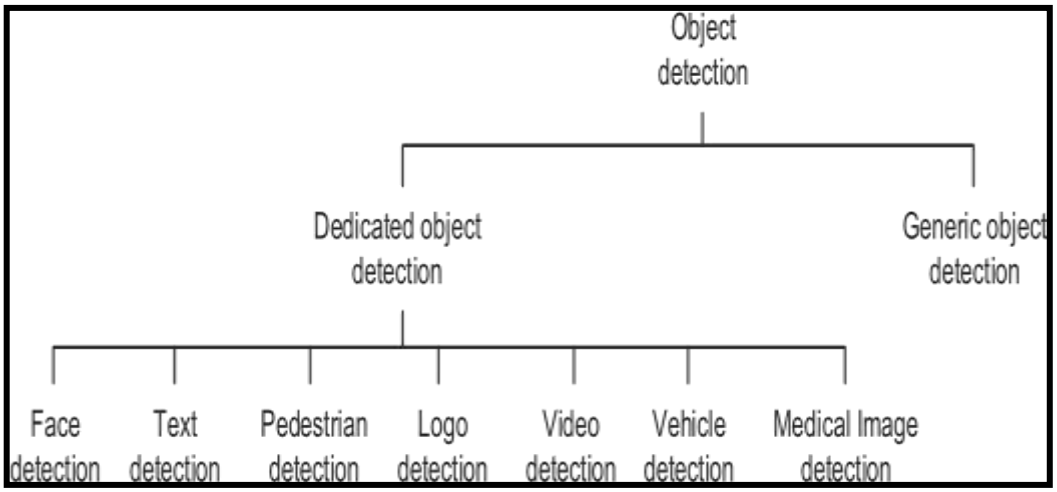


Figure 2.10: Object Detection Can Be Classified into Two Main Categories: General Object Detection and Specialized Object Detection [46].

The primary objective of object detection is to create computational models and methodologies that deliver fundamental information necessary for computer vision applications, specifically the localization of objects. Object detection is closely related to object classification, semantic segmentation, and instance segmentation. as depicted in Figure 2.11.

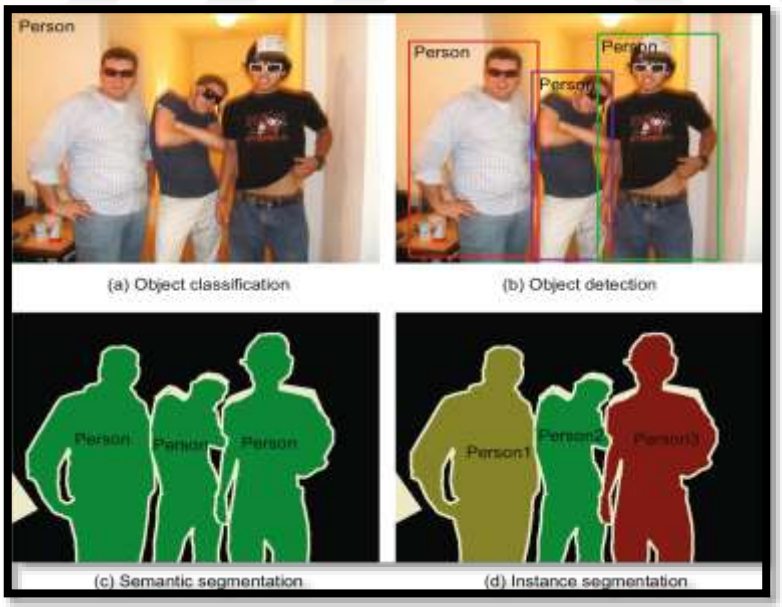


Figure 2.11: The Actual Applying of Different Image Techniques [46].

The primary criteria in object detection are cantered on accuracy, which encompasses both classification and localization accuracy, as well as speed [45]. This resulted in astonishing

discoveries and has made object detection a study area of considerable interest, attracting unprecedented attention. Object detection has become prevalent in various practical domains, including autonomous driving, robot vision, and video surveillance. Employing computer technology to identify objects in real-world scenarios automatically presents significant challenges [40]. Object detection can be considerably impacted by multiple factors, including but not limited to complicated backgrounds, noise interference, occlusion, low-resolution images, and changes in scale and orientation [40]. Conventional approaches for object detection heavily depend on manually designed characteristics, rendering them vulnerable to challenges such as variations in lighting conditions and limited ability to generalise effectively. It is essential to mention a very slow advancement in object detection between 2010 and 2012, as evidenced by the observations made in the PASCAL VOC challenge [47]. Only minimal enhancements were accomplished during this period using ensemble systems and minor modifications to conventional methodologies [48]. In consideration of these challenges. The advent of Convolutional Neural Networks (CNN) [49], a very effective deep learning architecture [50], has brought about a significant transformation in the field of object detection through its ability to facilitate hierarchical feature learning. The organisers of the ImageNet competition introduced a task in 2013 that focused on object detection, and this job required participants to recognise and classify 200 different items within a dataset consisting of 40,000 photos [40].

Nevertheless, the prevailing methodology depended on manually engineered characteristics, leading to a mean Average Precision (mAP) of 22.581%. Following this, the integration of deep learning and proposal techniques, such as R-CNN, resulted in a notable enhancement, attaining a considerable 43.933% mean Average Precision (mAP) in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [48]. Object detection has experienced significant advancements, marked by the widespread adoption of deep learning methodologies. Figure 2.12 presents a visual contrast between conventional and object detection techniques that rely on Deep Convolutional Neural Networks (DCNNs). The constant improvement of detection accuracy is essential requirements of diverse and intricate scenarios. Furthermore, enhancements have been made to the detection speed to meet the requirements of real-time system applications, maintaining high accuracy [46]. Hence, it is imperative to consider the balance between precision and efficiency in forthcoming scholarly

investigations [51]. The consideration of the trade-off between accuracy and speed is crucial for achieving state-of-the-art results [46].

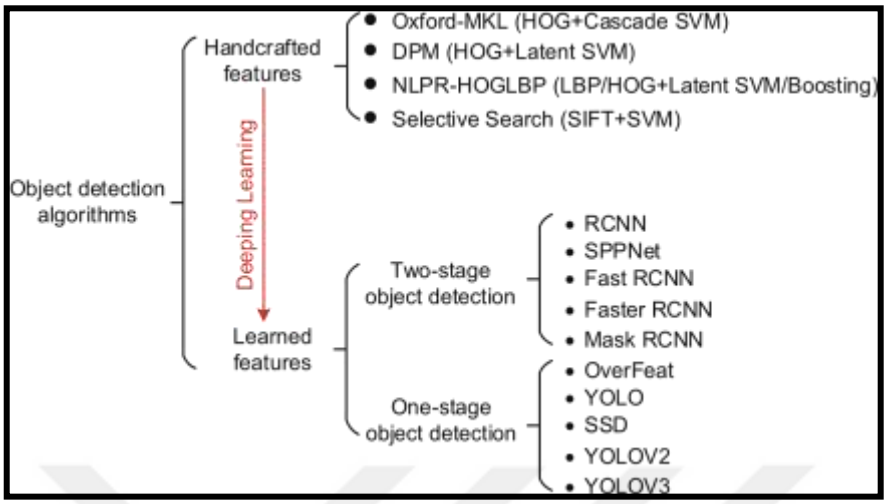


Figure 2.12: Classical, Traditional Object Detection and Object Detection Based on DCNNs [46].

2.3.4 YOLO Method

The application of real-time object detection has become increasingly crucial in various domains, such as autonomous cars, robots, video surveillance, and augmented reality [52]. The YOLO framework has garnered recognition in object detection due to its notable blend of speed and accuracy. Ts framework enables efficient and reliable identification of objects within images [52]. The YOLO family of algorithms has experienced multiple revisions since its inception, with each subsequent version making advancements over previous iterations to address constraints and boost overall performance (see Figure 2.13) [52].

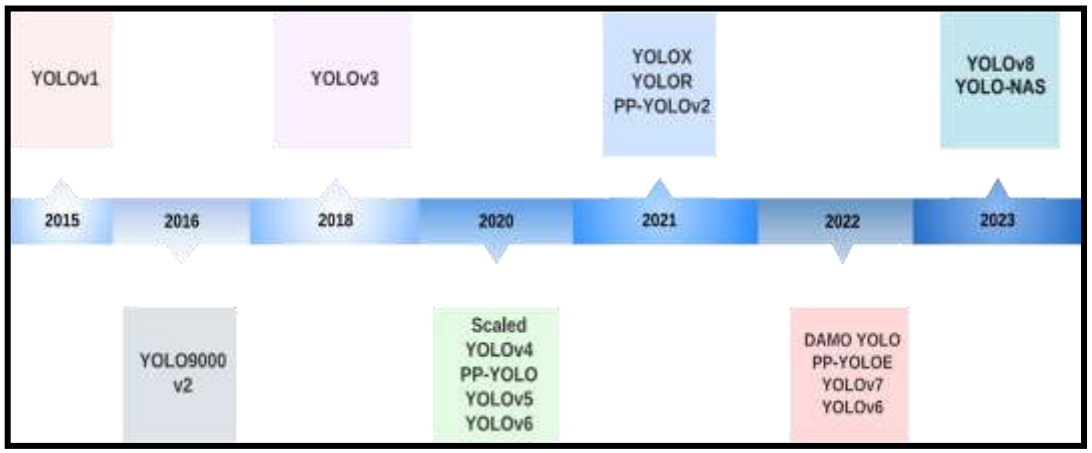


Figure 2.13: A Timeline of YOLO Versions [52].

The YOLO method and its subsequent architectural iterations have demonstrated notable advancements in detection accuracy, occasionally surpassing that of two-stage detectors. The broad acceptance of YOLOs in numerous applications can be attributed mostly to their higher inference speeds, as opposed to their exclusive emphasis on detection accuracy [52]. For example, whereas YOLO achieves a detection accuracy of 63.4% and Fast-RCNN achieves 70%, the inference time of YOLO is nearly 300 times faster; moreover, the use of YOLOs has resulted in their effective incorporation into diverse applications for object detection and recognition in varied scenarios, surpassing the performance of the two-stage detectors [52].

a. YOLO Applications

The YOLO algorithm has proven instrumental in real-time object detection, particularly in autonomous driving systems, where it efficiently identifies and tracks various objects such as automobiles, pedestrians, bicycles, and obstacles. Its applications extend to surveillance, sports analysis, and human-computer interaction. In agriculture, YOLO models contribute to crop detection, pest identification, and disease diagnosis, advancing precision agricultural techniques. The algorithm finds utility in biometric applications for face detection, enhancing security measures and facial recognition systems. In the medical field, YOLO is applied to cancer diagnosis, skin segmentation, and pill identification, leading to improved diagnostic accuracy, and streamlined treatment procedures. Additionally, YOLO plays a role in remote sensing for tasks like land use mapping and environmental monitoring. It has been integrated into security systems for real-time monitoring, compliance with social distance guidelines, and recognizing individuals wearing face masks. Other applications include surface inspection for quality control in manufacturing, traffic management for license plate detection and traffic sign recognition, wildlife detection for biodiversity preservation, and utilization in robotics and object detection using drone imagery. All can be evidence in [53]–[68].

b. YOLO Model Evaluation

Object identification models are often evaluated using Average Precision (AP), commonly known as Mean Average Precision (mAP). The mean accuracy, computed across all categories, provides a single outcome for model comparison. The AP metric, calculated

through precision-recall metrics and the IoU method, handles multiple object categories. The mAP can be computed using the accompanying mathematical expression:

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{2.1}$$

i. Precision and recall are vital metrics for evaluating a model's performance. Precision gauges the accuracy of positive predictions, while recall quantifies the proportion of actual positives identified by the model. Often, there's a trade-off between precision and recall, where an increase in identified items (higher recall) may lead to more false positives (lower precision). The AP metric captures this trade-off through the precision-recall curve, illustrating the relationship at various confidence thresholds. The precision equation can be expressed as:

$$precision = \frac{true\ positive}{true\ positive + false\ positive} \tag{2.3}$$

The recall value may be determined using the following equation:

$$Recall = \frac{true\ positive}{true\ positive + false\ negative} \tag{2.4}$$

ii. Multiple Object Categories The challenge of object detection models involves recognising and localising multiple categories of objects in an image. The difficulty is addressed by the AP measure, which calculates the average accuracy (AP) for each category and then computes the mean average precision by averaging these AP scores across all categories [52]. This methodology guarantees that the performance of the model is assessed individually for each area, this analysis aims to offer a thorough evaluation of the model's overall performance. efficacy [52].

iii. Intersection over Union (IoU) Object detection, aiming for accurate object localization through predicted bounding boxes, is commonly evaluated using the IoU metric. The AP

metric employs IoU to assess the accuracy of predicted bounding boxes, quantifying the overlap between predicted and ground truth bounding boxes in relation to their combined area. The COCO benchmark assesses model performance using various IoU criteria, evaluating object localization accuracy at different levels [52]. (see Figure 2.14)

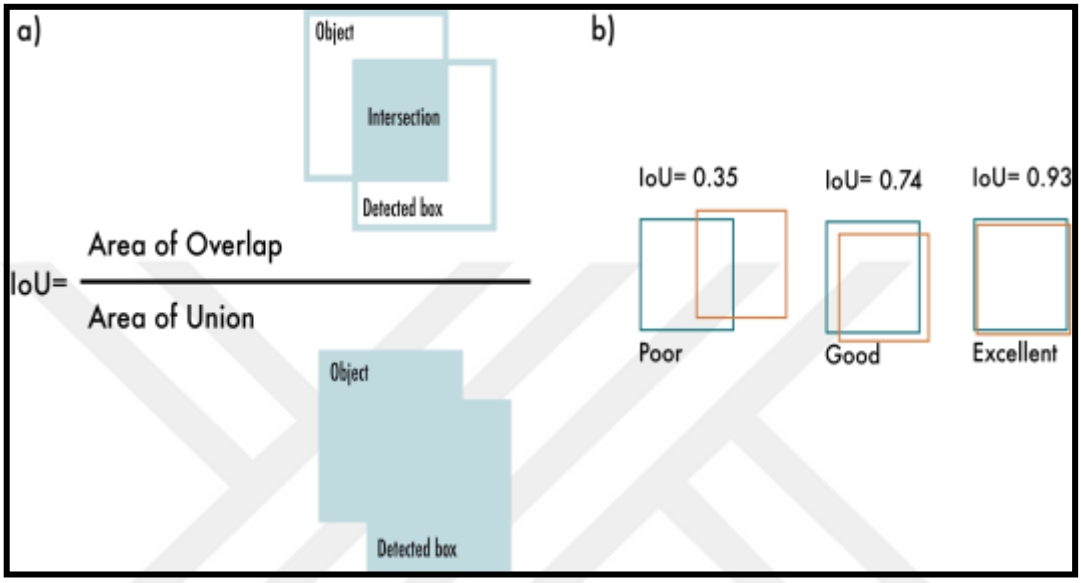


Figure 2.14: The Computation of IoU Entails the Division of The Intersection of Two Boxes By Their Union. The Provided Illustrations Showcase Three Distinct IoU Values, Each Corresponding to Various Bounding Box Positions [52].

c. YOLO Architectures

i. YOLO: You Only Look Once

The YOLO (You Only Look Once) algorithm was introduced by Joseph Redmon et al. in their article at the Conference on Computer Vision and Pattern Recognition (CVPR) in 2016 [69]. The YOLO algorithm is a notable advance in real-time object detection since it has fundamentally transformed the traditional methodology [52]. The tagline "You Only Look Once" appropriately signifies its pioneering capability to execute object detection within only one iteration of the network. This approach deviates from prior methodologies that either utilised sliding windows in conjunction with a classifier, requiring multiple iterations for each image, or employed advanced algorithms that divided the process into two separate stages [52]. The previous methodologies entailed an initial stage of finding prospective regions containing objects or region proposals, followed by a subsequent stage of applying

a classifier. In contrast, the YOLO model employed a streamlined methodology by exclusively employing regression techniques to predict detection outcomes in contrast to Fast R-CNN [70], The approach included two separate outputs, with one output used for estimating classification probabilities and the other for performing box coordinate regression. The YOLO model significantly advanced computer vision and object detection by offering a novel methodology that enables the simultaneous detection of all bounding boxes. The objective was accomplished by partitioning the input image into a grid with dimensions S × S [52]. Subsequently, forecasts were generated for B bounding boxes belonging to the same class within each grid element.

Furthermore, confidence ratings are assigned to C distinct classes, and each bounding box prediction includes Pc (confidence score), bx, by (centre coordinates), bh, and bw (height and width). The YOLO algorithm produces output dimensions of S × S × (B × 5 + C), which can be processed with non-maximum suppression (NMS) to eliminate redundant detections [52]. Figure 2.15 illustrates a simplified output vector. YOLOv1 achieved an average accuracy (AP) of 63.4 on the PASCAL VOC2007 dataset [52].

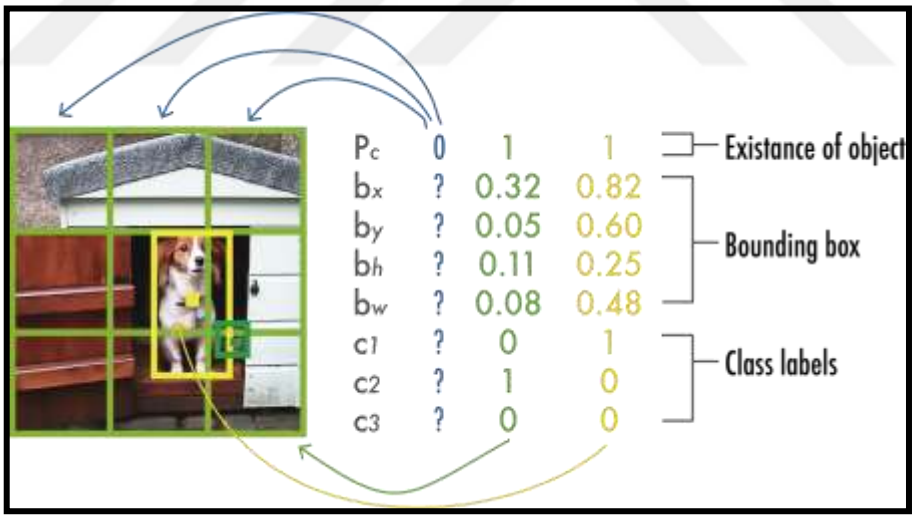


Figure 2.15: The Provided Diagram Depicts a Rudimentary YOLO Model That Showcases a Grid Structure of Dimensions Three-By-Three. This Model Is Designed to Classify Objects into Three Distinct Classes. Each Grid Element in The Diagram Is Associated with A Single Class Prediction, Resulting in A Vector Including Eight Values [52].

The YOLOv1 architecture consists of 24 convolutional layers followed by two fully connected layers, responsible for predicting bounding box coordinates and probabilities.

Leaky ReLU activations are used in all layers except the final one, which employs a linear activation function. Influenced by GoogLeNet and Network models, the YOLO algorithm integrates 1×1 convolutional layers to reduce feature map quantity while maintaining a reasonably low parameter count. A streamlined version, Fast YOLO, features nine convolutional layers.

ii. YOLOv2

The YOLOv2 model, presented by [71], introduced several notable improvements compared to the initial YOLO model. The enhancements were implemented to preserve its velocity while substantially enhancing its functionalities, enabling it to identify a wider spectrum of 9000 classifications [52]. The YOLOv2 model has introduced some significant enhancements in architecture and training methodologies. An important improvement was the integration of batch normalization on every convolutional layer, which improved the convergence rate and served as a regularization method to address the overfitting issue[52]. To address the requirement of managing high-quality images, the model performed fine-tuning with a resolution of 448 × 448, and this represents a significant deviation from the previous resolution of 224 × 224 that was employed during the pre-training phase on ImageNet [52]. The alteration resulted in a notable enhancement in the network's efficiency when handling inputs of greater resolution. Moreover, YOLOv2 moved to a convolutional architecture, eliminating dense layers, and adopting a more streamlined and efficient design [52]. Incorporating anchor boxes played a crucial role in accurately estimating bounding boxes by introducing predetermined shapes for object alignment, as shown in Figure 2.16. The authors chose the anchor box priors through the utilisation of dimension clustering. This involved applying k-means clustering to the training bounding boxes to identify the most suitable priors. The YOLOv2 model brought about a significant advancement in location prediction by directly calculating the coordinates of bounding boxes with grid cells; this was achieved by training the network to predict five boxes for every cell, with each box being characterized by five values as shown in Figure 2.17 [52].

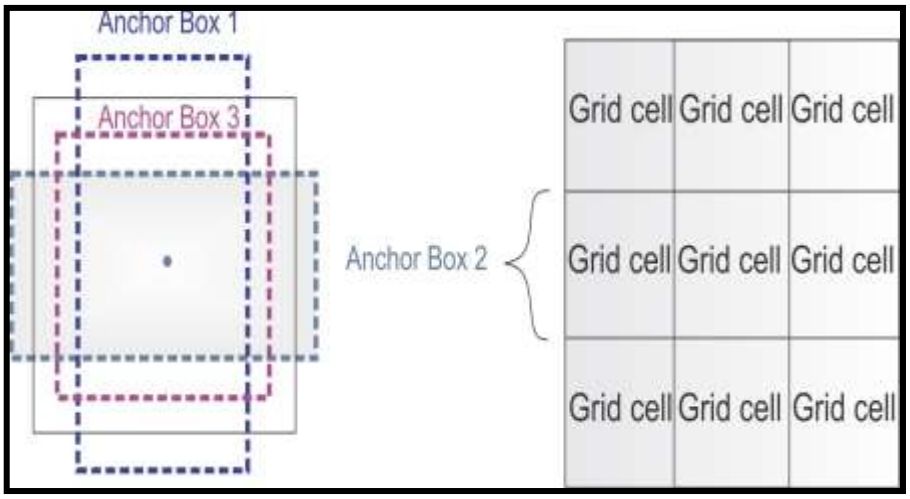


Figure 2.16: Anchor Boxes: For Every Grid Cell, Yolov2 Creates Numerous Anchor Boxes [52].

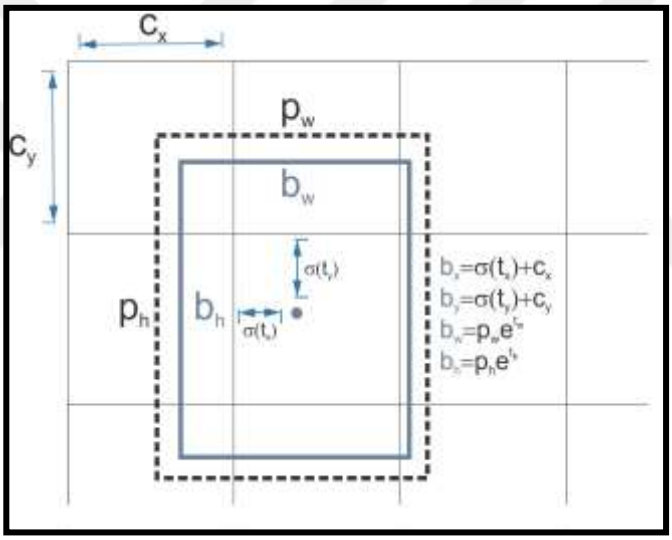


Figure 2.17: Bounding Box Prediction: To forecast bounding boxes, use the predicted T_x and T_y values. These values run through a sigmoid function and are offset by the grid cell's position (C_x, C_y). This yields the box's centre coordinates. The previous width P_w is used to calculate the height and breadth of the final box [52].

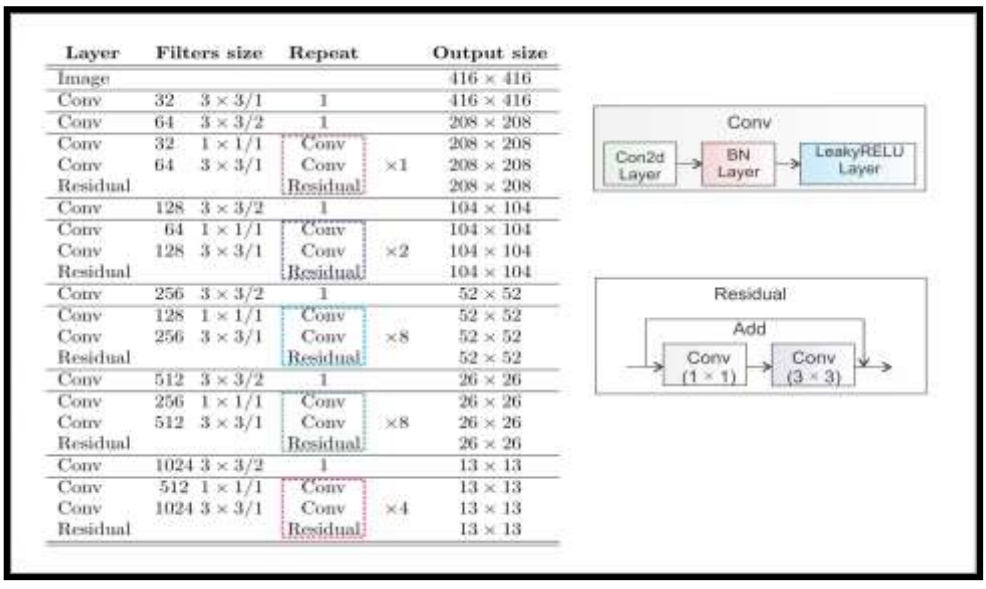
Furthermore, the model preserved more detailed features and incorporated a passthrough layer to concatenate feature maps efficiently, ensuring the preservation of valuable information across the network's structure [52]. Incorporating multi-scale training was critical in creating YOLOv2, as it facilitated the model's ability to cope with a wide range of input sizes, from 320 × 320 to 608 × 608. This method has significantly enhanced the robustness and versatility of YOLOv2 in various input conditions, and the integration of

architectural enhancements and training methodologies in YOLOv2 has resulted in a notable advancement in object detection within the domain of computer vision [52]. This progress holds great potential for achieving enhanced accuracy and efficiency in the detection and localization of objects. The architecture of YOLOv2 is structured around the Darknet-19 backbone, comprising 19 convolutional layers and five max-pooling layers. like the YOLOv1 model, this design is influenced by the Network in Network concept. This technique, which utilises 1 × 1 convolutions within the 3 × 3 convolutions to decrease the parameters. Moreover, as previously stated, batch normalization is employed to manage and facilitate the convergence process. The predictions generated by YOLOv2 on the PASCAL VOC dataset consist of five bounding boxes, each characterized by five numerical values, and contain 20 distinct classes [52]. In the object classification component, the final four convolutional layers are substituted by a solitary convolutional layer with 1000 filters. Subsequently, a global average pooling layer is applied, followed by a SoftMax operation. As a result of these improvements, YOLOv2 achieved an average precision (mean average precision) of 78.6% on the PASCAL VOC 2007 dataset, representing a substantial enhancement compared to the 63.4% achieved by YOLOv1 [52].

iii. YOLOv3

The YOLOv3 model, developed by Joseph Redmon and Ali Farhadi in 2018, incorporated significant revisions and improved architecture to attain a cutting-edge performance while preserving its ability to operate in real-time; the modifications were implemented in contrast to YOLOv2 [52]. YOLOv3 has various substantial improvements and modifications. Initially, the network preserves the forecast of four coordinates (t_x, t_y, $t_{w,}$ $t_{h)}$ for every bounding box, like the approach employed in YOLOv2 [52]. However, the current model also utilises logistic regression to predict a score for each bounding box without considering the presence of an object [52]. The score assigned to the anchor box with the maximum overlap with the ground truth is 1, differentiating it from the remaining anchor boxes assigned a score of 0 [52]. YOLOv3 employs a distinctive approach by assigning a solitary anchor box exclusively to each ground truth item. When an object is not assigned any anchor box, YOLOv3 experiences a classification loss without any localization or confidence loss [52]. This contrasts the methodology employed by Faster R-CNN [72]. In addition, YOLOv3

undergoes a movement in its approach to classification, moving from the utilisation of SoftMax to the adoption of binary cross-entropy; this shift allows for the training of separate logistic classifiers, thereby framing the problem as one of multilabel classification [52]. This modification enables greater adaptability and accuracy in class projections. The architecture of YOLOv3 also includes a feature extractor that is more comprehensive, including 53 convolutional layers with residual connections [52]. In addition, the researchers have incorporated a modified spatial pyramid pooling (SPP) block into the underlying architecture, incorporating multiple kernel sizes; this integration has resulted in an improved receptive field and enhanced performance, particularly in the YOLOv3-spp variant, which demonstrated a notable 2.7% gain in AP50 [52]. The YOLOv3 model incorporates multi-scale predictions, like the approach used in Feature Pyramid Networks. In contrast to YOLOv2, YOLOv3 utilises k-means clustering to establish bounding box priors for anchor boxes [52]. This approach involves the utilisation of three prior boxes, each corresponding to a separate scale, as opposed to YOLOv2's employment of five prior boxes per cell. The YOLOv3 architecture includes a backbone known as Darknet-53, which has undergone substantial revisions. In recent advancements, stride convolutions have placed max-pooling layers while integrating residual connections have been introduced. Darknet-53 comprises 53 convolutional layers; detailed information regarding this architecture can be found in Figure 2.18. The Darknet-53 backbone exhibits comparable Top-1 and Top-5 accuracies to ResNet-152 but operates at nearly double the speed [52].



Layer	Filters	size	Repeat		Output size
Image					416 × 416
Conv	32	3 × 3/1	1		416 × 416
Conv	64	3 × 3/2	1		208 × 208
Conv	32	1 × 1/1	Conv		208 × 208
Conv	64	3 × 3/1	Conv	×1	208 × 208
Residual			Residual		208 × 208
Conv	128	3 × 3/2	1		104 × 104
Conv	64	1 × 1/1	Conv		104 × 104
Conv	128	3 × 3/1	Conv	×2	104 × 104
Residual			Residual		104 × 104
Conv	256	3 × 3/2	1		52 × 52
Conv	128	1 × 1/1	Conv		52 × 52
Conv	256	3 × 3/1	Conv	×8	52 × 52
Residual			Residual		52 × 52
Conv	512	3 × 3/2	1		26 × 26
Conv	256	1 × 1/1	Conv		26 × 26
Conv	512	3 × 3/1	Conv	×8	26 × 26
Residual			Residual		26 × 26
Conv	1024	3 × 3/2	1		13 × 13
Conv	512	1 × 1/1	Conv		13 × 13
Conv	1024	3 × 3/1	Conv	×4	13 × 13
Residual			Residual		13 × 13

Figure 2.18: YOLOv3 Darknet-53 Backbone [52].

iv. YOLOv3 Multi-Scale

The overarching architectural framework, YOLOv3, incorporates a notable progression in multi-scale predictions, wherein predictions are generated at many grid sizes. This invention has demonstrated significant value in improving bounding box precision and enhancing all object recognition, addressing a notable limitation observed in prior iterations of YOLO [52]. The detection architecture, as depicted in Figure 2.19, functions at several scales in the following manner: The initial output, referred to as **y1**, is the output of the YOLOv2 algorithm, which is generated based on a grid of size 13×13 [52]. The formation of the second output, denoted as **y2,** involves concatenating the output obtained after multiplying the output of Darknet-53 by the (*Res* × 4) with the output obtained after multiplying it by the (*Res* × 8). An up-sampling process is conducted before concatenation due to the varying sizes of the feature maps, specifically 13×13 and 26×26. In conclusion, the third output, **y3**, merges the 26×26 feature maps with the 52×52 feature maps through an additional up-sampling procedure. The output tensor dimensions for each scale in the COCO dataset, which consists of 80 categories, are N×N× [3× (4+1+80)]. Here, N×N specifies the size of the feature map or grid cell. The numerical value "3" represents the number of boxes assigned to each cell, while the notation "4+1" comprises the four coordinates and the objectness score [52].

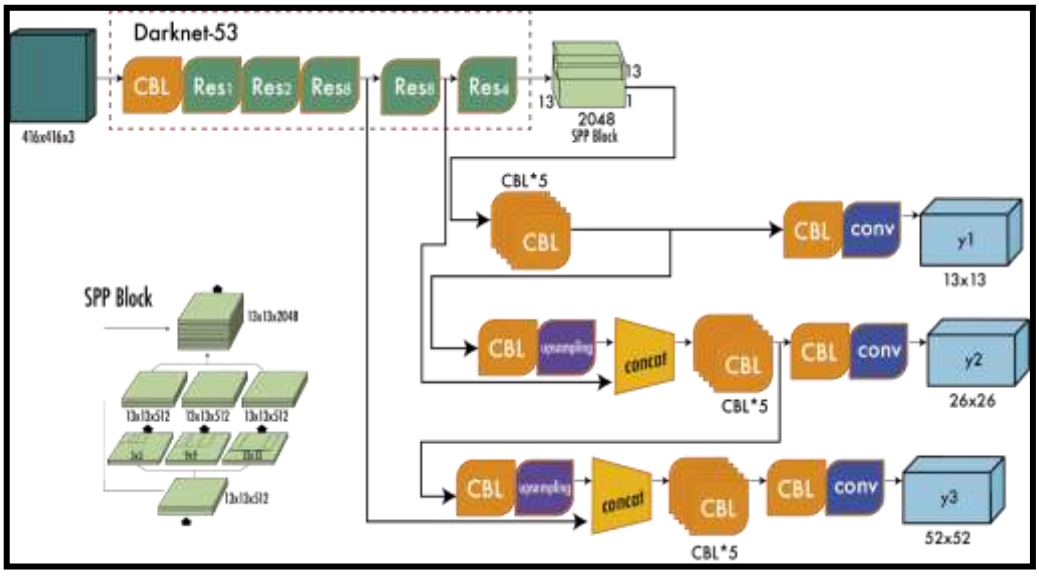


Figure 2.19: YOLOv3 Multi-Scale Detection Architecture [52]

v. YOLOv4

In April 2020, a noteworthy advancement occurred in object detection with the publication of a research article on YOLOv4 by [73]. This development came after two years without any updates to the YOLO framework [52]. At first glance, the introduction of multiple authors proposing a novel "official" iteration of YOLO may have appeared unconventional [52]. Nevertheless, YOLOv4 adhered to the fundamental principles of YOLO, which prioritised real-time processing, as depicted in Figure 2.20, a single-shot methodology, and dependency on the darknet framework. The enhancements in YOLOv4 were of such magnitude that the community expeditiously embraced it as the authoritative iteration of YOLOv4 [52]. The key modifications architecture introduced in YOLOv4 can be summarized as follows, CSPDarknet53-PANet-SPP, optimizing object detection by incorporating Bag-of-Specials (BoS). The modified Darknet-53 with cross-stage partial connections (CSPNet) and the Mish activation function outperformed other backbone topologies. The model utilizes a customized path aggregation network (PANet) with a modified spatial attention module (SAM) and anchors for detection, deviating from YOLOv3's FPN. The Bag-of-Freebies (BoF) technique enhances training methodologies through mosaic augmentation, Drop Block for regularization, class label smoothing, CIoU loss function, and Cross Mini-Batch Normalization (CmBN). Self-Adversarial Training (SAT) is employed to enhance the model's resilience to disturbances by manipulating input images. Genetic algorithms optimize hyperparameters, with a cosine annealing scheduler dynamically adjusting the learning rate during training, ensuring optimal performance.

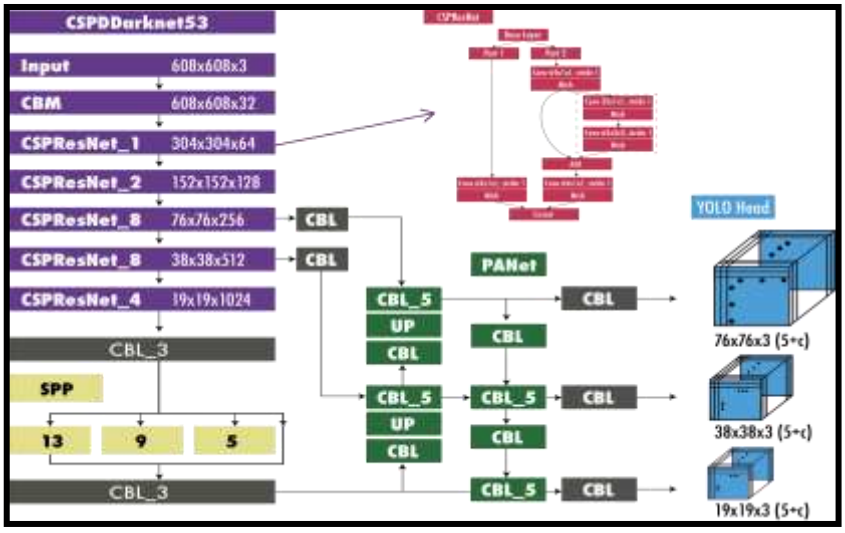


Figure 2.20: YOLOv4 Architecture [52].

vi. YOLOv5

In 2020, YOLOv5, introduced by Glen Jocher [74], builds upon the advancements of YOLOv4 but utilizes the PyTorch framework instead of Darknet. The pre-training methodology involves analysing and adapting anchor boxes, leveraging a k-means function for initial conditions and a Genetic Evolution (GE) algorithm for refinement. YOLOv5 adopts a modified CSPDarknet53 framework with a distinctive Stem design to minimize memory consumption. The architecture integrates Spatial Pyramid Pooling Fast (SPPF) for multi-scale feature addressing, a modified Cross Stage Partial Aggregation Network (CSP-PAN) in the neck segment, and YOLOv3-inspired head module. The implementation includes augmentations like Mosaic and Mix-up, enhancing grid sensitivity for stability. YOLOv5 offers scaled variations (nano to extra-large) catering to diverse applications. YOLOv5x, optimized for superior performance, achieves 50.7% Average Precision (AP) at 640-pixel image size and 55.8% with a 1536-pixel input, demonstrating flexibility for different hardware and application needs. Figure 2.21 depicts the structure of YOLOv5; all can be evidence by [75]–[79].

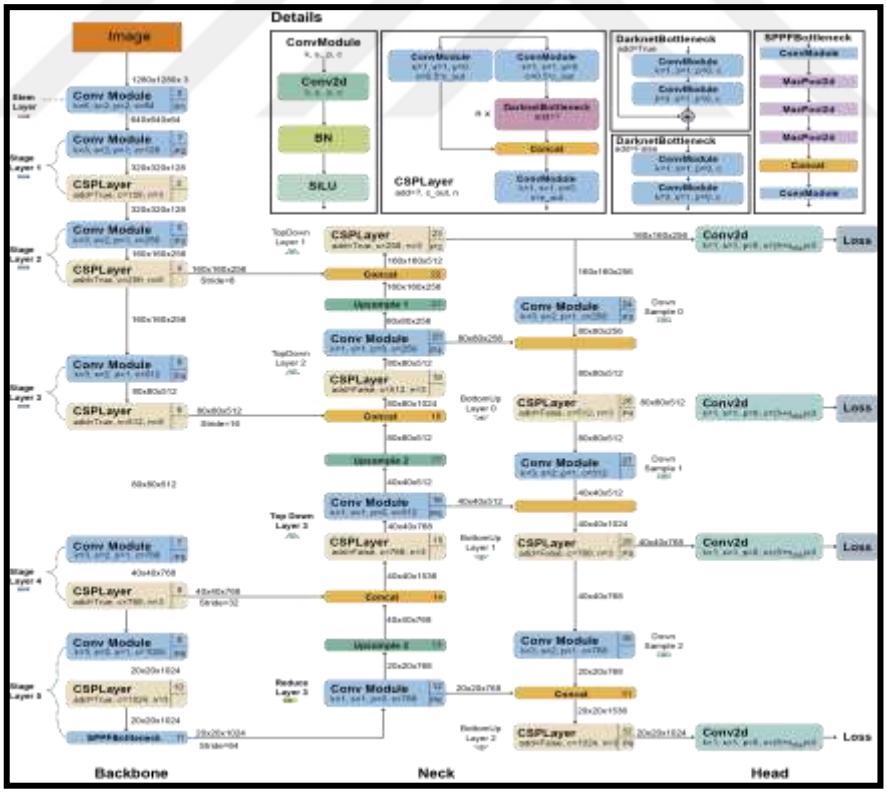


Figure 2.21: YOLOv5 Architecture [52].

vii. YOLOv6

The YOLOv6 model, presented by [80], consists of a network architecture that includes an efficient backbone with RepVGG or CSPStackRep blocks, a neck structure based on the PAN topology, and an efficient decoupled head that incorporates a hybrid-channel strategy. In addition, the research study presents sophisticated quantization methods that utilise post-training quantization and channel-wise distillation. These techniques result in detectors that exhibit improved speed and accuracy. YOLOv6 demonstrates superior performance in accuracy and speed measures compared to earlier state-of-the-art models; Figure 2.22 refers to YOLOv6 architecture [52]. This model presents several significant advancements: the introduction of a unique EfficientRep backbone, which is based on RepVGG and specifically designed to enhance parallel processing capabilities; the use of RepBlocks or CSPStackRep Blocks in an improved PAN neck for larger models; and the implementation of an efficient decoupled head that aligns [52]. The proposed methodology utilises a Task alignment learning approach to assign labels, incorporates novel classification and regression losses such as VariFocal and SIoU/GIoU, and integrates self-distillation techniques for both regression and classification tasks. In addition, the system has a detection quantization approach that utilises RepOptimizer and channel-wise distillation techniques, substantially enhancing the detector's computational efficiency [52]. The researchers introduce eight scaled models, denoted as YOLOv6-N to YOLOv6-L6. Notably, the largest model achieves a remarkable average precision of 57.2% while operating at a speed of about 29 FPS on an NVIDIA Tesla T4. This evaluation uses the MS COCO dataset's test-dev 2017 subset [52].

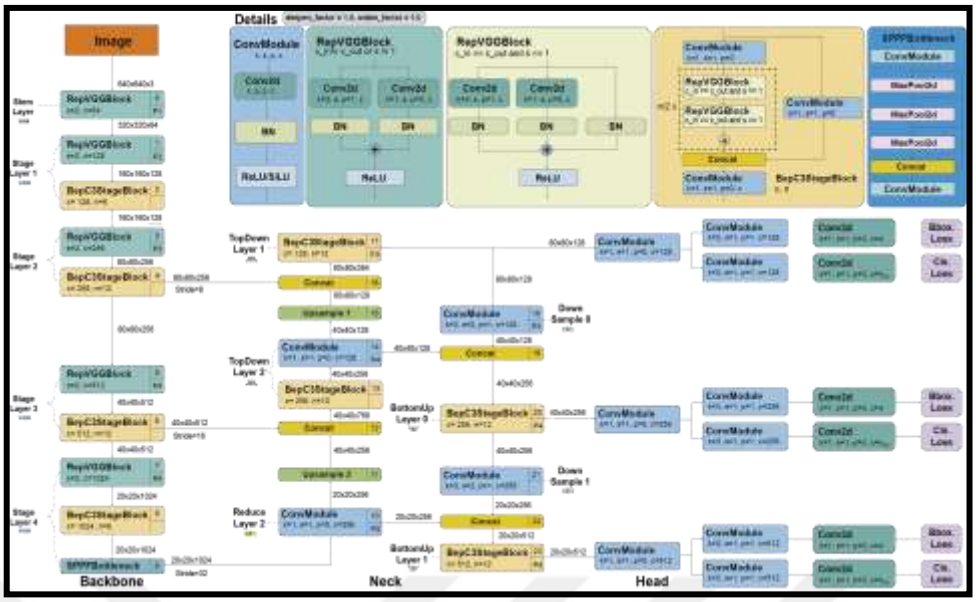


Figure 2.22: YOLOv6 Architecture [80].

viii. YOLOv7

The creators of YOLOv4 and YOLOR introduced YOLOv7 in July 2022, which demonstrated a noteworthy accomplishment by surpassing all current object detection models in speed and accuracy [81]. This achievement encompasses a wide spectrum, ranging from 5 to 160 frames per second; like its previous iteration, YOLOv4, this version was trained exclusively using the MS COCO dataset and without dependence on pre-learned backbones. The YOLOv7 model featured a variety of architectural enhancements and integrated many "bag-of-freebies" techniques to improve its accuracy while still preserving its efficiency in terms of inference speed; however, it should be noted that these upgrades resulted in a minor increase in the duration of the training process [52]. To understand its architectural structure comprehensively, please consult Figure 2.23. The architectural modifications included in YOLOv7 consist of two fundamental components; first and foremost, YOLOv7 presents the Extended Efficient Layer Aggregation Network (E-ELAN), which expands the ELAN approach [82]. The ELAN framework is designed to enhance the effectiveness of learning and convergence in deep models through the regulation of the shortest longest gradient path; YOLOv7 further advances its capabilities by introducing E-ELAN, a specialised framework designed to accommodate models using a wide range of stacked computational blocks [52]. This is accomplished by reorganising and consolidating cardinality from separate feature groups, enhancing network learning capabilities while

maintaining the integrity of the original gradient route. Moreover, YOLOv7 effectively tackles the requirement for model scaling, particularly in the context of concatenation-based architectures; the utilisation of traditional scaling methods, such as depth scaling, may lead to an imbalance in the ratios of input and output channels inside transition layers, hence affecting hardware efficiency. However, YOLOv7 presents a novel scaling methodology to address this issue. This approach guarantees that the dimensions of model blocks, in terms of depth and width, are evenly scaled by a consistent factor, thus maintaining the model's ideal structural integrity [52].

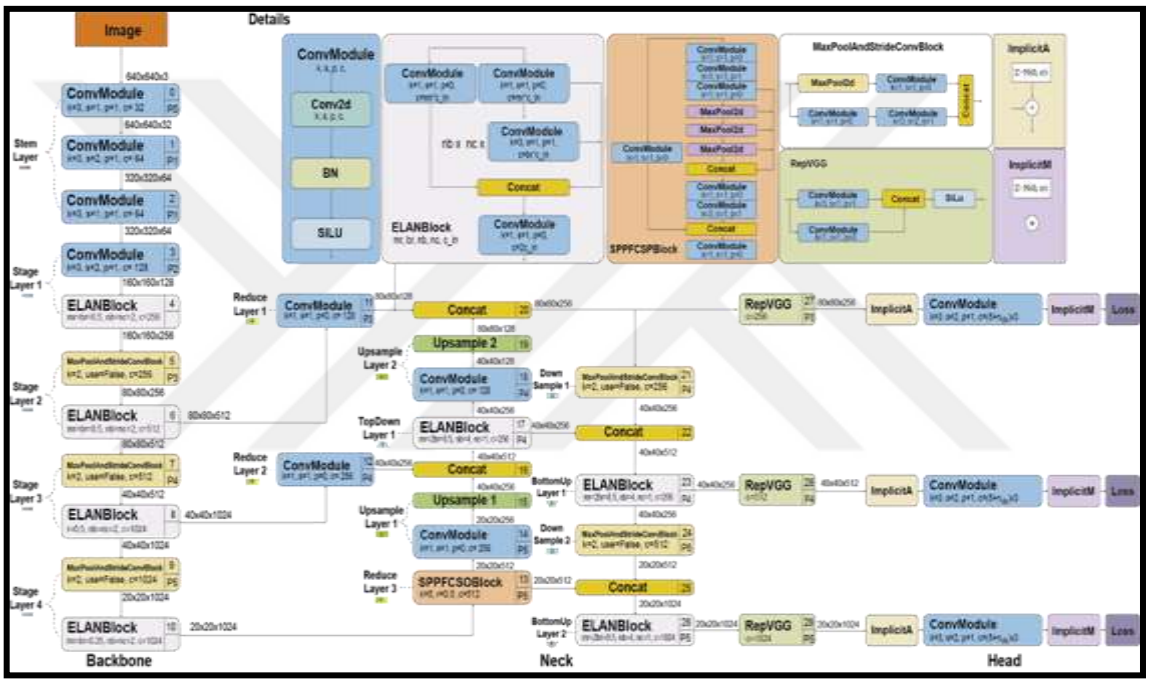


Figure 2.23: YOLOv7 Architecture[83].

3. METHODOLOGY

3.1 DATASET ACQUISITION

3.1.1 Collecting Data

The dataset collected photos and consisted of three distinct subsets, each representing the alphabet, numbers, and words. ArSL lacks comprehensive and up-to-date dictionaries, resulting in a shortage of standardized resources. This can be attributed to the fact that each Arab country has its distinct SL, owing to variations in Arabic dialects employed across these regions. Using the Classical or standard Arabic language is not prevalent in everyday communication or as a spoken form. Instead, it is mostly employed on television channels, particularly in news broadcasts, official written documents, and literary works. The use of ArSL is not universally consistent but exhibits variation across different countries and even within the same country. In the Arab region, various SL are utilized, including those employed in Saudi Arabia, Iraq, Morocco, and Egypt [3]. The initial release of the unified dictionary of ArSL in 1999, a collaborative effort by The League of Arab States (LAS) and the Arab League Educational, Cultural and Scientific Organization (ALECSO), aimed to standardize ArSL. The second dictionary segment followed in 2007. Deaf individuals in Arabic-speaking nations face challenges due to limited services for their community, requiring assistance in accessing information, education, communication with the hearing community, and active participation in events. The dataset used in this study aligns with the 2007 Unified ArSL vocabulary [3]. ArSL involves two sign types, static and dynamic gestures; however, this dataset predominantly comprises static gestures, enabling the implementation of an object detection model based on single-frame photographs. The initial dataset was sourced from [84] and it's available online in Kaggle [85] and includes a curated and annotated collection of 14,202 photos representing 32 letter signs in ArSL, as shown in Figure 3.1. These images exhibit diverse backgrounds and were obtained from 50 individuals.

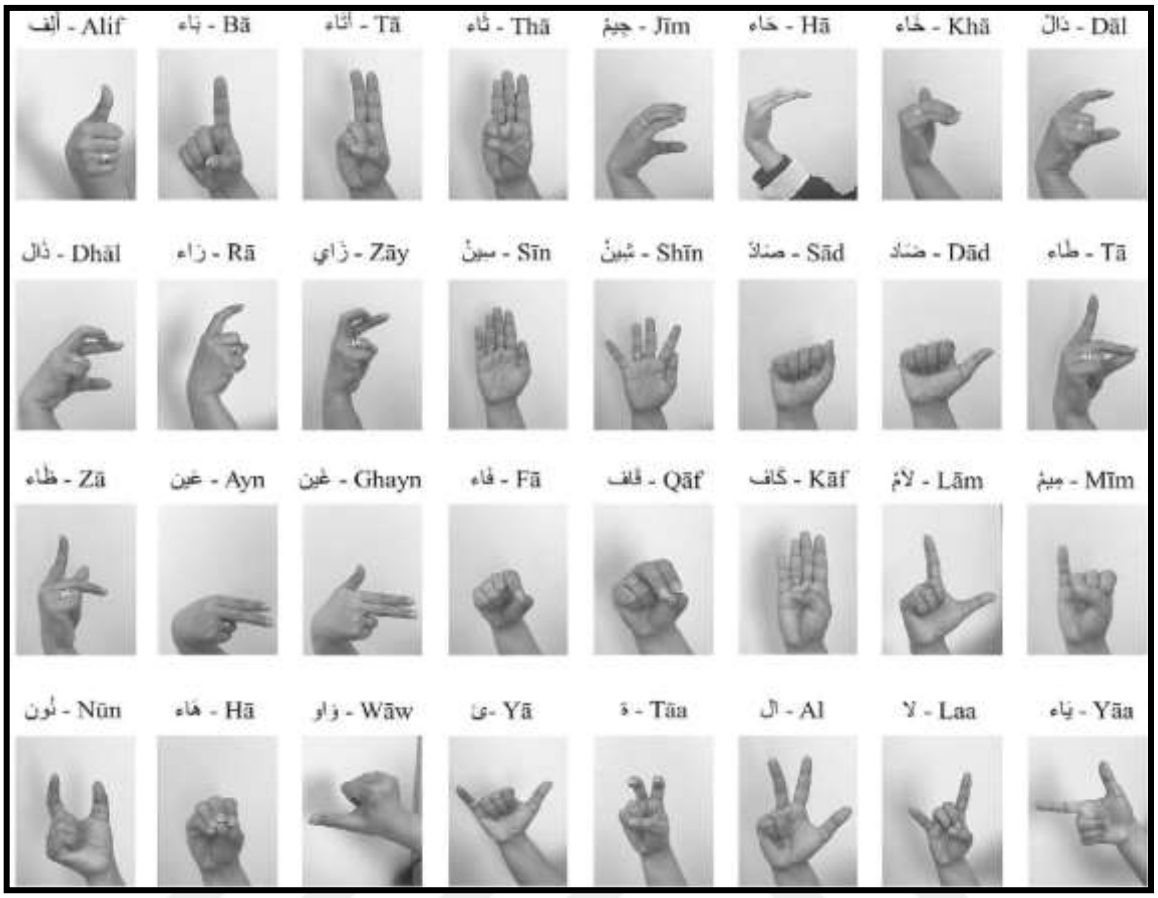


Figure 3.1: The Alphabets ArSL Dataset.

The second dataset consists of numerical values and their corresponding static gestures, specifically focusing on numbers 0 to 9, as shown in Figure 3.2. This dataset was meticulously constructed, involving the collection of 1160 images. Four volunteers actively performed the gestures, following the guidelines described in the unified ArSL dictionary. The data collection occurred in various locations, backgrounds, and lighting conditions.

Figure 3.2: The Numbers 0 To 9 In the ArSL Numbers Dataset.

The third dataset consists of 19 important words: accident, aim at, bandage, certificate, emergency, fixed, government, identity conformity, injury, necessary, negative outcome, no, plate, positive outcome, ready, slow, student, support, and true as shown in Figure 3.3. This dataset specifically focuses on the emergency medical station, approval and rejection answers, and other related words. A single hand gesture is also required for this dataset. In total, 1345 images were captured by four participating volunteers in different locations, backgrounds, and lighting conditions. The photographs were captured using the camera of a Realme 6.0 smartphone running on the Android 11 operating system, including a camera resolution of 64 megapixels.

Figure 3.3: The Selected Words in The ArSL Word Dataset.

3.1.2 Augmentation and Annotation

The process of data annotation holds significant importance in the context of object detection activities. Each image within the dataset was assigned a label indicating its equivalent representation of the Arabic alphabet, numbers, or words. The bounding box annotation process was conducted to determine the target object's precise location accurately. The bounding box surrounding the hand gesture can be determined using coordinates (x, y, h, w), subsequently employed to specify its width and height. The labels were stored in Txt files. The annotation process was conducted using Anaconda Navigator and Labelimg, as shown in Figure 3.4, and the augmentation was conducted using Roboflow [86]; its platform serves as a tool for developers and data scientists to effectively handle, preprocess, and enhance their image collections to train machine learning models. The utilization of image data in diverse machine-learning frameworks is facilitated by streamlining the preparation process.

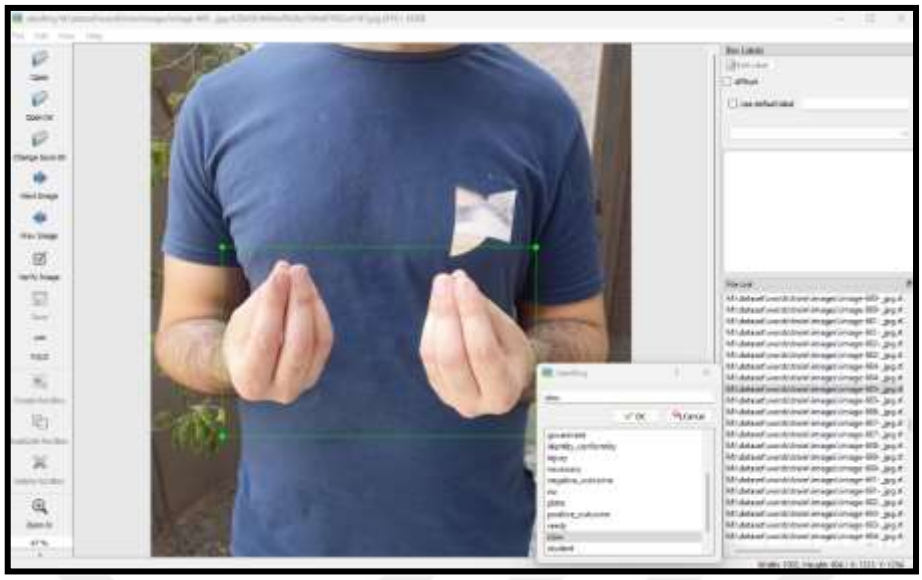


Figure 3.4: Labelimg [87].

The datasets underwent a sample augmentation process, which included rotation, as shown in Figure 4.5 and increased photos. As a result, the numbers dataset expanded to include 2348 images, while the words dataset now has 2116 images. The augmentation process is expected to enhance the detection accuracy by training the model with larger images, improving its precision.

Figure 3.5: Two Images from The Number's Dataset.

3.1.3 Spilt Dataset

The training and testing procedure of YOLOv8 involves the utilization of image Splitting. The collection of images is partitioned into training, validation, and test sets randomly. These sets comprise approximately 90%, 5%, and 5% of the sign data. The determination of these separation values was based on tests that were done. The size of the training set is increased to enhance the model's training process using annotated images, whilst the validation and test sets are exclusively used to assess the model's detection accuracy. The process of image splitting is conducted utilizing the Roboflow platform. As a result, the dataset containing numerical values has been partitioned into 2121 images for training, 116 images for validation, and 111 for testing. In the same way, the word dataset has been partitioned into 1850 photos for training, 133 images for validation, and an additional 133 images for testing.

3.2 PROPOSED MODEL

3.2.1 YOLOv8 Architecture

In January 2023, Ultralytics, the company responsible for the advancement of YOLOv5, introduced YOLOv8 [88]. The YOLOv8 framework provides five variants that have been scaled accordingly: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), and YOLOv8x (extra-large). The current version of YOLO has been enhanced to handle a range of vision tasks, such as object identification, segmentation, pose estimation, tracking, and classification. As previously mentioned, YOLOv8 is an object detection methodology based on deep learning techniques. The architectural design of YOLOv8 encompasses a diverse range of components and features. Now, a shortage of scholarly literature on the architectural aspects of YOLOv8 exists. Consequently, our analysis necessitates drawing upon discerning observations to establish a comparative framework vis-a-vis preceding iterations of YOLO.

a. Backbone

The Cross Stage Partial (CSP) design, which splits the feature map into two halves, is incorporated into the suggested model's CSPDarknet53 base architecture for feature extraction. While the second is added to the output of the previous segment, the first is

subjected to convolution operations. By reducing computational weight, this convolutional neural network (CNN) design improves the CNN's learning ability. To enhance gradient flow information, the C2f module is integrated and combined with the C3 module in the YOLOv8 model. As shown in Figure 3.6, the C2f module is made up of two ConvModule and "n" DarknetBottleNeck components connected by Split and Concat, whereas the C3 module has three ConvModule and DarknetBottleNeck components. Conv-BN-SiLU is the component of each ConvModule, where "n" is the number of bottlenecks. In contrast to YOLOv5, our model uses the C2f module in favor of the C3 module, and for computational efficiency, we have fewer blocks in each stage than in YOLOv5. Furthermore, the Spatial Pyramid Pooling-Fast (SPPF) module to improve the current Spatial Pyramid Pooling (SPP) module and speed up the model's inference.

b. Neck

Typically, networks with greater depth tend to record a wider range of feature information, leading to enhanced capabilities in making dense predictions. Nevertheless, it is worth noting that deep networks that are excessively complex can result in a decrease in the accuracy of object localization. Additionally, many convolution processes can potentially result in losing valuable information, especially when dealing with smaller objects. Hence, integrating Path Aggregation Network (PAN) and Feature Pyramid Network (FPN) designs are required for the effective integration of multi-scale data. The Neck component in our model architecture uses multi-scale feature fusion to combine features from different network levels, as shown in Figure 3.6. Because fewer convolution layers are used in the lower levels, more layers in the network help get information in the top layers while maintaining location details. Additional network layers facilitate information acquisition in the upper layers whilst preserving location information, attributed to utilizing fewer convolution layers in the lower layers. Primarily inspired by the YOLOv5 model, it is observed that the Feature Pyramid Network (FPN) employs an up-sampling technique from the top to the bottom to augment the feature information contained within the bottom feature map. Conversely, the Path Aggregation Network (PAN) utilizes a down-sampling approach from the bottom to the top to acquire additional information from the top feature map. These two feature outputs are merged to enhance the accuracy of predictions for images with diverse dimensions.

c. Head

This module predicts the class and places objects inside the image. The utilization of anchor boxes is employed to estimate the location of an object's location and its corresponding class scores. The primary neural network is trained to optimize the IoU metric, which measures the overlapping between the predicted and ground-truth bounding boxes. Furthermore, the method also integrates the non-maximum suppression (NMS) technique, eliminating redundant bounding boxes that overlap and guaranteeing that only the most reliable predictions are preserved. In contrast to the YOLOv5 model, our methodology incorporates a decoupled head architecture, wherein the classification and detection heads are distinct and independent entities. To summarize, the mask indices are used to determine the specific anchor scales and anchor boxes inside the resulting tensor to make predictions regarding the bounding boxes and confidence ratings for each object in the image.

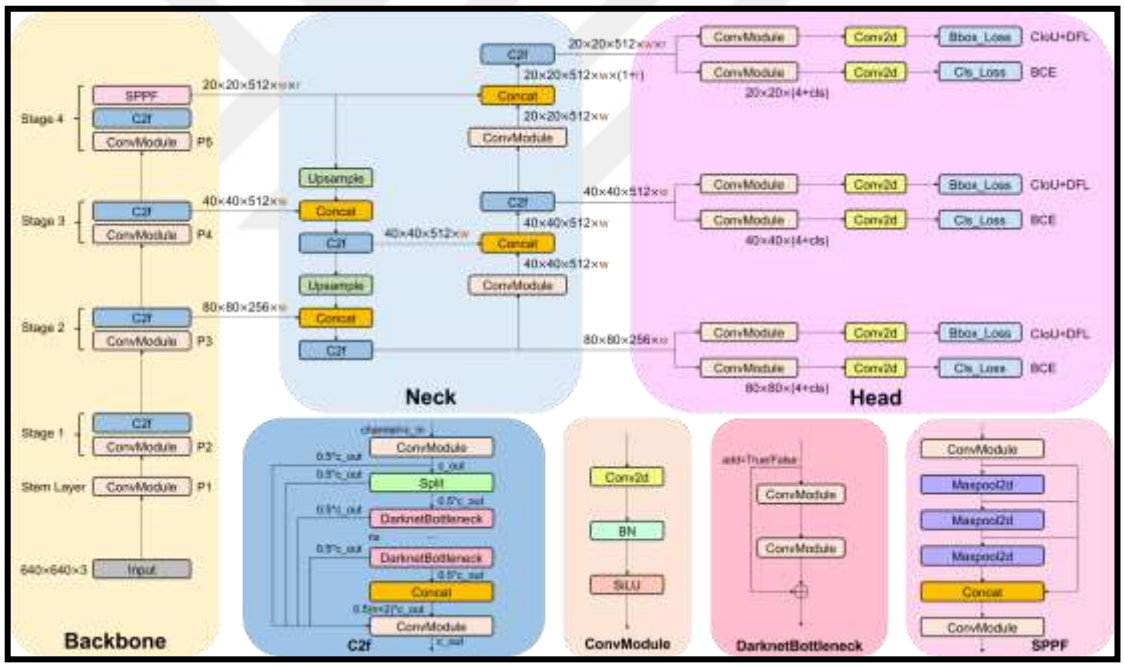


Figure 3.6: Detailed Illustration of Yolov8 Architecture. The Backbone, Neck, and Head Are the Three Parts of Our Model, And C2f, ConvModule, DarknetBottleNeck, And SPPF Are Modules [89].

3.2.2 Training Based YOLOv8 Model

a. Platform

In the training model, Google Colab, or Google Collaboratory, was used to train the YOLOv8 model on the three datasets. Google Collaboratory is a freely available Integrated Development Environment (IDE) provided by Google. Its primary purpose is to support and enhance research and educational activities in Artificial Intelligence (AI). Colab functions as a coding environment based on Jupyter Notebook, offering the advantage of free access to both the Graphics Processing Unit (GPU) and Tensor Processing Unit (TPU). The platform has pre-installed libraries commonly employed in deep learning research, such as PyTorch, TensorFlow, Keras, and OpenCV. The recognition of the computational demands of machine learning and deep learning algorithms, which often rely on GPU support for efficient processing, is acknowledged by Colab. To meet these requirements, Colab offers access to a cloud-based GPU, specifically the Tesla V100 and TPU (TPUv2), renowned for their exceptional performance, to cater to the needs of AI researchers. Our approach used NVIDIA A100 Tensor Core GPU; the A100 model exhibits a significant performance improvement, boasting up to 20 times higher capabilities than its predecessor. Additionally, it has the flexibility to be divided into seven separate GPU instances, allowing for dynamic adjustments to accommodate changing computational requirements. The A100 80GB introduces the highest memory bandwidth in the world, exceeding two terabytes per second (TB/s), enabling the efficient execution of extensive models and datasets to enhance the training model process to ensure the best performance and accuracy.

b. Training

The selected YOLOv8 model will be trained using the pre-processed ArSL datasets. The training will involve optimizing the model's hyperparameters and refining the architecture. The YOLOv8 models, specifically YOLOv8n, YOLOv8s, and YOLOv8m, will undergo training using hyperparameters that have been derived through the utilization of a genetic algorithm All the datasets will train in nano, small and medium weight of the YOLOv8. To obtain a summary of the training hyperparameters, refer to Table 3.1.

Table 3.1: Empirical Hyperparameter Values for Alphabet, Words, and Numbers of Model,

Model	**YOLOv8m**	**YOLOv8s**	**YOLOv8n**
Image size	640	640	640
Batch size	32	32	32
epoch	200	200	200
Patience	100	100	100
Workers	8	8	8
Parameters	25.9 million	11.2 million	3.2 million
Optimizer	SGD	SGD	SGD
Decay	0.0005	0.0005	0.0005

4. RESULT AND DISCUSSION

ArSLR system will be evaluated comprehensively, utilizing well-established metrics outlined in Section 2.3.4.2. The model's accuracy in recognizing ArSL alphabets, numbers, and words will be assessed using commonly used metrics such as precision, recall, mean Average Precision (mAP) and accuracy, F1 and FPS. Furthermore, the evaluation will consider real-world circumstances to gauge the actual usability and resilience of the system.

4.1 ALPHABETS MODEL RESULTS

The 32 classes and signs of the alphabet model achieve high accuracy in the detection, as shown in Table 4.1. All three models reached convergent results, but the highest result was for the YOLOv8m (medium) model with 98.0% Precision, 98.1% Recall and 98.9% mAP 0.5.

Table 4.1: The Performance of Yolov8 Models on ASL Alphabets.

Models	Precision	Recall	mAP 0.5
YOLOv8n	98.00%	97.40%	98.80%
YOLOv8s	98.10%	97.70%	98.80%
YOLO8vm	98.00%	98.10%	98.90%

Finally, the F1 score is calculated as the harmonic mean of precision and recall. It can be computed during the training phase using the following formula:

$$F1 = \frac{2 * (precision * recall)}{precision + recall} \quad (4.1)$$

The accuracy of our model during the detection step is calculated using the following equation:

$$accuracy = \frac{number\ of\ correct\ predication}{tota\ number\ of\ samples} \quad (4.2)$$

Based on the previous equations, the model’s accuracy, F1 and FPS (Frame Per Second) values after training for 200 epochs are shown in Tabel 4.2

Table 4.2: Results of Performance Evaluation For 32 Alphabets Models.

Model	F1	Accuracy	FPS
YOLOv8n	80%	91.00%	65.36
YOLOv8s	81%	90.00%	55.25
YOLOv8m	85%	94.00%	28.49

The evaluation metrics for each 32 classes in the ArSL using the YOLOv8m model are presented in Table 4.3.

Table 4.3: Evaluation Results of YOLOv8m for 32 Alphabets Signs.

Class	P	R	mAP50	Class	P	R	mAP50
ain	0.993	0.997	0.995	laam	0.99	0.947	0.993
al	0.986	0.993	0.993	meem	1	0.987	0.995
aleff	0.998	0.993	0.995	meem	0.965	0.959	0.984
bb	1	0.985	0.995	ra	1	0.988	0.995
dal	0.954	0.931	0.943	saad	0.985	0.973	0.994
dha	0.983	0.985	0.993	seen	0.995	0.97	0.988
dhad	0.977	0.971	0.987	sheen	0.995	1	0.995
fa	0.963	0.961	0.99	ta	0.962	1	0.989
gaaf	0.985	0.988	0.994	taa	0.983	0.978	0.994
ghain	0.989	1	0.995	thaa	1	0.983	0.995
ha	0.992	0.993	0.995	thal	0.981	0.963	0.99
haa	0.957	1	0.981	toot	0.999	0.993	0.995
jeem	0.978	0.971	0.993	waw	0.947	0.967	0.97
kaaf	0.967	1	0.986	ya	0.941	0.978	0.983
khaa	0.976	0.993	0.995	yaa	0.986	1	0.994
la	0.998	1	0.995	zay	0.956	0.955	0.977

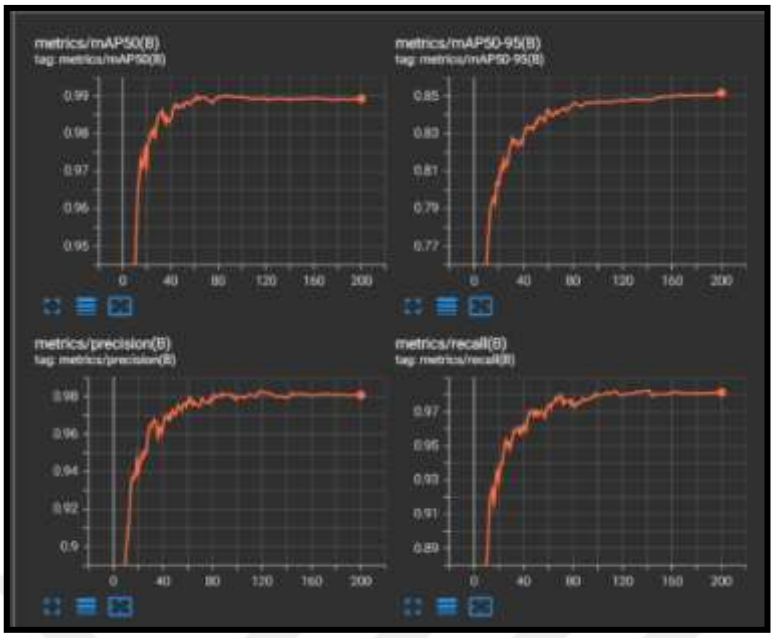


Figure 4.1: Depicts the Metrics Achieved After Completing the Training and Validation Processes Over 200 Epochs.

The analysis Figure 4.1 shows a positive correlation between the number of epochs and the precision of our model. This means an increased number of epochs leads to improved precision. Simultaneously. Accuracy is utilized as a metric for evaluating our model, providing a measure of its overall correctness by indicating the frequency with which the model made valid predictions. Precision evaluates the model's capacity to make correct predictions for a single category. In contrast, recall quantifies the model's efficacy in identifying instances of a particular category. The following figures depict the process of alphabet detection. Figure 4.2 refers to the implementation of the model in test set images, and it’s worth noting that all the sign of the alphabet needs one hand to express.

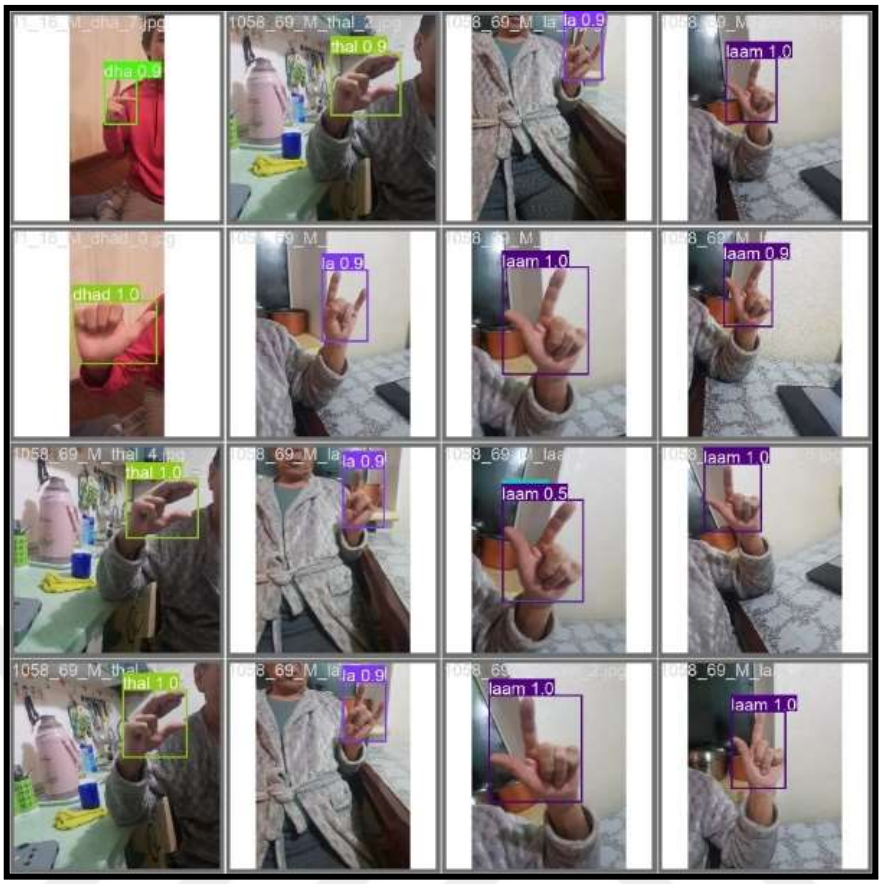


Figure 4.2: ArSL Alphabets of Recognition Results on Test Images.

4.2 NUMBERS MODEL RESULTS

The Table 4.6 illustrates that the ten classes and signs in the number model from 0 to 9 demonstrate high accuracy. All three models exhibit convergent results, with the YOLOv8m (medium) model performing the best, achieving a 97.6% mean Average Precision (mAP) compared to the other two models, as shown in Table 4.4. It also attains 95.5% precision and 96.5% recall.

Table 4.4: Performance of The Yolov8 Model on ASL Numbers.

Models	Precision	Recall	mAP 0.5
YOLOV8n	95.50%	94.20%	97.30%
YOLOV8s	96.10%	96.60%	96.00%
YOLOV8m	95.90%	96.50%	97.60%

All the models exhibit precise detection capabilities in real-world webcam scenarios and photos. To assess the models' performance, we utilised equations (4.2) and (4.1) to obtain the accuracy performance, the F1 score, and the speed, all shown in Table 4.5.

Table 4.5: Results of Performance Evaluation Numbers Models.

Model	F1	Accuracy	FPS
YOLOv8n	56%	75.00%	27.1
YOLOv8s	78%	73.00%	23.26
YOLOv8m	77%	80.00%	28.49

Table 4.6 presents comprehensive data about each of the ten classes in this model.

Table 4.6: Evaluation Results of YOLOv8m for 10 Numbers Signs.

Class	P	R	mAP50
0	0.969	1	0.995
1	1	0.865	0.99
2	1	0.992	0.995
3	0.987	1	0.995
4	1	0.828	0.995
5	0.861	1	0.995
6	0.986	1	0.995
7	0.956	1	0.995
8	1	0.874	0.995
9	0.964	1	0.994

Significant improvements in precision, recall, and mean average precision (mAP) were observed during the validation stage, as illustrated in Figure 4.3. Finally, Figure 4.4 shows the image recognition in the test set, and it turns out that all numerical signs require a single-hand movement to express them.

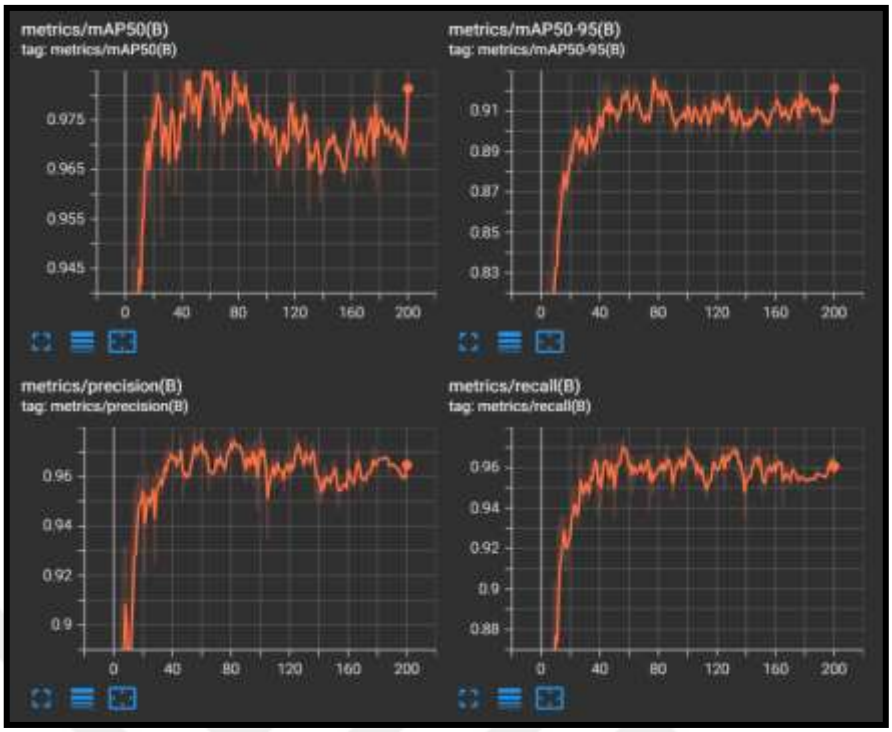


Figure 4.3: Depicts the Metrics Achieved After Completing the Training and Validation Processes Over 200 Epochs.

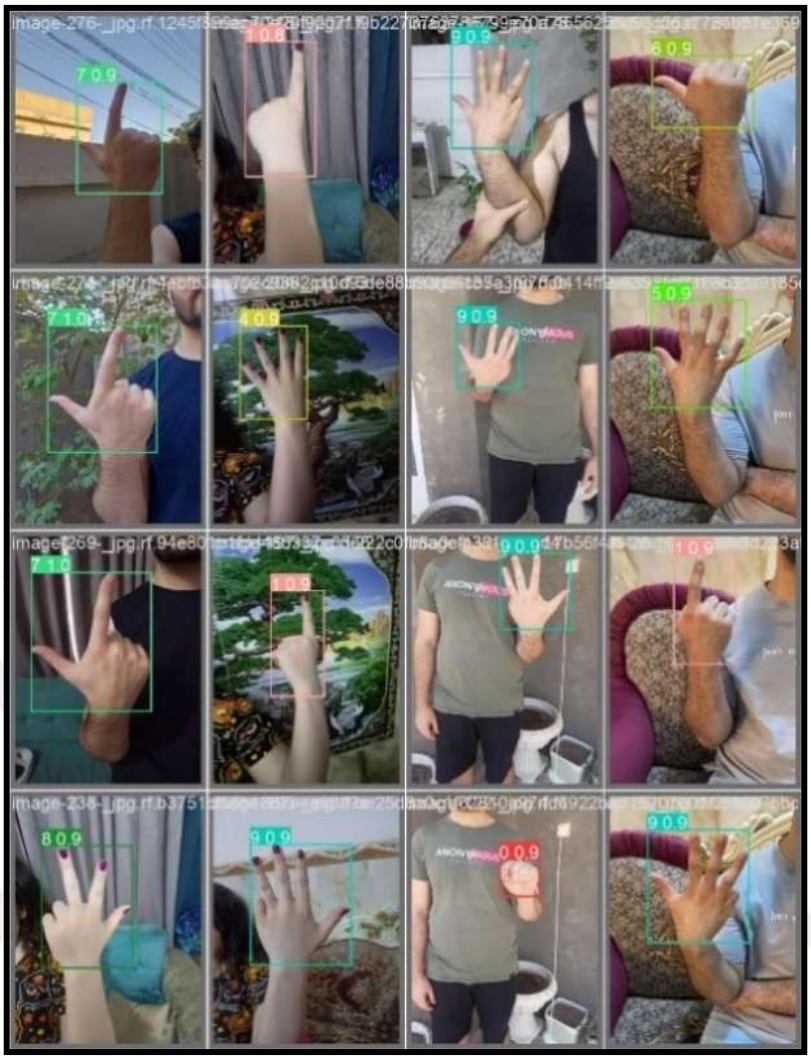


Figure 4.4: ArSL Numbers of Recognition Results on Test Images.

4.3 WORDS MODEL RESULTS

Similarly, the previous models, the YOLOv8, performed so efficiently with words dataset in all wights, the models so convergent in results, especially YOLOv8s and YOLOv8m. In the end, we chose to use the medium weight. The precision is 99.0%, Recall 99.9% and mAP 99.5%; all the details are shown in Table 4.7. The common thing in all the models is all the results were so convergent with high accuracy. The proposal model is efficient and fits the solution for the core problem and could be developed as an official translation tool based on AI computer vision because YOLOv8 could train a big dataset size, providing opportunities to extend and develop the model. More details about the performance in this model for the Precision, Recall and mAP 0.5 for each class are in Table 4.8.

Table 4.7: Performance of Yolov8 Model on ASL Words.

Models	Precision	Recall	mAP 0.5
YOLOV8n	98.50%	99.30%	99.50%
YOLOV8s	99.00%	99.90%	99.50%
YOLOV8m	99.00%	99.90%	99.50%

Table 4.8: Evaluation Results of Yolov8m For 19 Words Signs.

Class	P	R	mAP50
accident	1	1	0.995
aim_at	1	1	0.995
bandage	0.988	1	0.995
certificate	0.982	1	0.995
emergency	0.992	1	0.995
fixed	1	1	0.995
government	0.981	1	0.995
conformity	1	1	0.995
injury	1	1	0.995
necessary	1	1	0.995
negative outcome	0.978	1	0.995
no	1	1	0.995
plate	1	1	0.995
positive outcome	1	1	0.995
ready	1	1	0.995
slow	0.988	1	0.995
student	0.991	1	0.995
support	1	1	0.995
True	0.982	1	0.995

Convergence in real-world webcam detection and image processing leads to the simultaneous improvement of performance and accuracy. After using equations 4.2 and 4.1 on the testing set, the accuracy, F1 score, and speed achieved are all shown in Table 4.9.

As depicted in Figure 4.6, some signs need one hand to express, and others need both hands to achieve optimal outcomes.

Table 4.9: Results of Performance Evaluation Words Models.

Model	F1	Accuracy	FPS
YOLOv8n	95%	95.00%	27.1
YOLOv8s	96%	84.00%	25.51
YOLOv8m	95%	85.00%	23.7

During the traning and vaildtion stage all the values shown in Figure 4.5 show fluctuations in the initial epochs, but after epoch 160 they begin to stabilize with similar results.

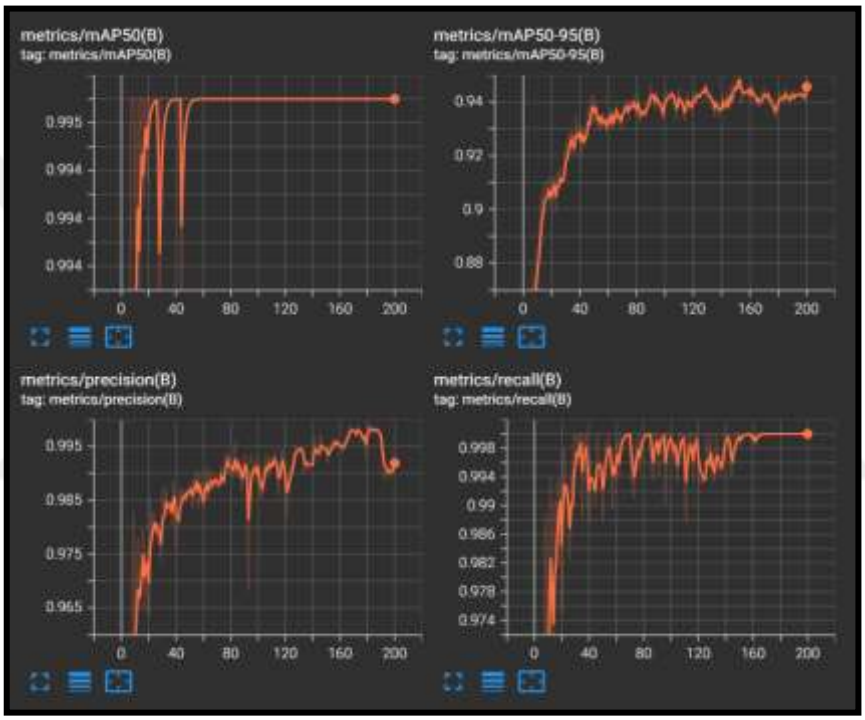


Figure 4.5: Depicts the Metrics Achieved After Completing the Training and Validation Processes Over 200 Epochs.

Figure 4.6: ArSL Words of Recognition Results on Test Images.

5. CONCLUSION AND FUTURE WORK

ArSL is the main communication form within the deaf community, enabling effective communication and engagement with society. Nevertheless, a notable obstacle arises from the limited comprehension of SL among people who can hear, resulting in a substantial disparity across multiple facets of the lives of individuals with hearing impairments. This disparity encompasses career prospects, engagement in communal activities, and broader assimilation within society. The deaf community in numerous Arab countries, which are confronted with economic and political instability, encounters heightened difficulties that adversely affect their access to treatment and support. The field of computer vision and deep learning detection technologies has witnessed significant progress in the past few years, offering academics promising avenues to tackle the problems mentioned above. Our suggested system aims to create an operational recognition system using the YOLOv8 deep learning model. The architectural foundation of YOLOv8 incorporates CSPDarknet53, an effective technique for extracting features, by utilizing Cross Stage Partial (CSP) to enhance the learning process and minimize computational costs. Substituting the C3 module with the C2f module improves the transmission of gradient flow information.

A deliberate block decrease within each stage is also implemented to optimize computing performance. The Neck component, drawing inspiration from FPN and PAN architectures, integrates multi-scale feature fusion techniques to enhance the accuracy of predictions for images with diverse sizes. The prediction module, which includes anchor boxes and a decoupled head, is designed to maximize the IoU metric and utilizes Non-Maximum Suppression (NMS) to enhance the accuracy of predictions. Although there is a scarcity of information on the architecture of YOLOv8, valuable insights form the basis for doing a comparison analysis with earlier versions of YOLO. Furthermore, we have generated two datasets from inception, encompassing 2,348 images for numerical representations, 2,116 for textual representations, and 14,202 for alphabetical representations. Upon completing the model's training and subsequent testing, we obtained a mean Average Precision (mAP) of 98.9% at an Intersection over Union (IoU) threshold of 0.5 for alphabets.

Similarly, for numbers, the mAP at IoU 0.5 was found to be 97.60%. Lastly, for words, the mAP at IoU 0.5 was determined to be 99.0%. The presented model exhibits efficacy and precision in detecting webcams in practical scenarios, facilitating the advancement of a commercially viable application for converting ArSL into textual form. We suggest augmenting the dataset with a larger corpus of words, sentences, and numerical data to enhance future research. Additionally, it would be beneficial to construct a separate dataset specifically tailored to capture dynamic gesture movements. While multiple techniques are available for detecting and recognizing signs, our primary focus is cost-effective solutions promoting inclusivity and accessibility for all individuals.

REFERENCES

[1] R. Nima and T. Han, “Interpreting Arabic Sign Alphabet by using the Deep Learning,” In Press, 2022.

[2] W. H.Organization, “Deafness and hearing loss.” Accessed: Oct. 05, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss

[3] A. H. Aliwy and A. A. Alethary, “Development of arabic sign language dictionary using 3D avatar technologies,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 21, no. 1, pp. 609–616, 2021, doi: 10.11591/ijeecs.v21.i1.

[4] R. M. Duwairi and Z. A. Halloush, “Automatic recognition of Arabic alphabets sign language using deep learning,” *International Journal of Electrical and Computer Engineering*, vol. 12, no. 3, pp. 2996–3004, 2022, doi: 10.11591/ijece.v12i3.pp2996-3004.

[5] The Center for Strategic and International Studies, “Reading the Signs: Diverse Arabic Sign Languages.” Accessed: Oct. 05, 2023. [Online]. Available: https://www.csis.org/analysis/reading-signs-diverse-arabic-sign-languages

[6] C. T. Martin and N. Chanda, “Mental Health Clinical Simulation: Therapeutic Communication,” *Clin Simul Nurs*, vol. 12, no. 6, pp. 209–214, Jun. 2016, doi: 10.1016/j.ecns.2016.02.007.

[7] B. Yahya AlKhuraym, M. Maher Ben Ismail, and O. Bchir, “Arabic Sign Language Recognition using Lightweight CNN-based Architecture.” [Online]. Available: www.ijacsa.thesai.org

[8] “The most spoken languages worldwide in 2023.” Accessed: Nov. 17, 2023. [Online]. Available: https://www.statista.com/statistics/266808/the-most-spoken-languages-worldwide/

[9] T. Jamil, “Design and Implementation of an Intelligent System to translate Arabic Text into Arabic Sign Language,” in *Canadian Conference on Electrical and Computer Engineering*, Institute of Electrical and Electronics Engineers Inc., Aug. 2020. doi: 10.1109/CCECE47787.2020.9255774.

[10] D. R. Kothadiya, C. M. Bhatt, T. Saba, A. Rehman, and S. A. Bahaj, “SIGNFORMER: DeepVision Transformer for Sign Language Recognition,” *IEEE Access*, vol. 11, pp. 4730–4739, 2023, doi: 10.1109/ACCESS.2022.3231130.

[11] “Arabic sign language dictionary.” Accessed: Oct. 09, 2023. [Online]. Available: https://menasy.com/

[12] M. Mustafa, “A study on Arabic sign language recognition for differently abled using advanced machine learning classifiers,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 3. Springer Science and Business Media Deutschland GmbH, pp. 4101–4115, Mar. 01, 2021. doi: 10.1007/s12652-020-01790-w.

[13] S. Aly and W. Aly, “DeepArSLR: A novel signer-independent deep learning framework for isolated arabic sign language gestures recognition,” *IEEE Access*, vol. 8, pp. 83199–83212, 2020, doi: 10.1109/ACCESS.2020.2990699.

[14] A. Kuznetsova, L. Leal-Taixé, and B. Rosenhahn, “Real-time sign language recognition using a consumer depth camera,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013. doi: 10.1109/ICCVW.2013.18.

[15] A. S. Al-Shamayleh, R. Ahmad, N. Jomhari, and M. A. M. Abushariah, “Automatic Arabic Sign Language Recognition: a Review, Taxonomy, Open Challenges, Research Roadmap and Future directions,” *Malaysian Journal of Computer Science*, vol. 33, no. 4, pp. 306–343, 2020, doi: 10.22452/mjcs.vol33no4.5.

[16] A. S. Al-Shamayleh, R. Ahmad, M. A. M. Abushariah, K. A. Alam, and N. Jomhari, “A systematic literature review on vision based gesture recognition techniques,” *Multimed Tools Appl*, vol. 77, no. 21, 2018, doi: 10.1007/s11042-018-5971-z.

[17] P. K. Pisharady and M. Saerbeck, “Recent methods and databases in vision-based hand gesture recognition: A review,” *Computer Vision and Image Understanding*, vol. 141, 2015, doi: 10.1016/j.cviu.2015.08.004.

[18] S. Kausar and M. Y. Javed, “A survey on Sign Language recognition,” in *Proceedings - 2011 9th International Conference on Frontiers of Information Technology, FIT 2011*, 2011, pp. 95–98. doi: 10.1109/FIT.2011.25.

[19] A. M.Ahmed, R. Abo Alez, M. Taha, and G. Tharwat, “Automatic Translation of Arabic Sign to Arabic Text (ATASAT) System,” Academy and Industry Research Collaboration Center (AIRCC), Apr. 2016, pp. 109–122. doi: 10.5121/csit.2016.60511.

[20] M. Madhushree, P. Raja, S. Thuja, D. Raja, and S. Assistant Professor, “EasyChair Preprint Certain Investigations in Hand Gesture Recognition-a Survey Certain Investigations in Hand Gesture Recognition-A Survey,” 2023.

[21] A. Alshaheen, “AMERICAN SIGN LANGUAGE RECOGNITION USING YOLOV4 METHOD.”

[22] Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553. Nature Publishing Group, pp. 436–444, May 27, 2015. doi: 10.1038/nature14539.

[23] J. Chai, H. Zeng, A. Li, and E. W. T. Ngai, “Deep learning in computer vision: A critical review of emerging techniques and application scenarios,” *Machine Learning with Applications*, vol. 6, p. 100134, 2021, doi: 10.24433/CO.0411648.v1.

[24] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, “Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach,” in *Procedia Computer Science*, 2018. doi: 10.1016/j.procs.2018.05.069.

[25] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Deep learning for visual understanding: A review,” *Neurocomputing*, vol. 187, 2016, doi: 10.1016/j.neucom.2015.09.116.

[26] C. Neubauer, “Evaluation of convolutional neural networks for visual recognition,” *IEEE Transactions on Neural Networks*, vol. 9, no. 4. 1998. doi: 10.1109/72.701181.

[27] J. Fieres, J. Schemmel, and K. Meier, “Training convolutional networks of threshold neurons suited for low-power hardware implementation,” in *IEEE International Conference on Neural Networks - Conference Proceedings*, 2006. doi: 10.1109/ijcnn.2006.246654.

[28] I. Arel, D. Rose, and T. Karnowski, “Deep machine learning-A new frontier in artificial intelligence research,” *IEEE Comput Intell Mag*, vol. 5, no. 4, 2010, doi: 10.1109/MCI.2010.938364.

[29] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, “Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks,” *AASRI Procedia*, vol. 6, 2014, doi: 10.1016/j.aasri.2014.05.013.

[30] J. Fang, Y. Zhou, Y. Yu, and S. Du, “Fine-Grained Vehicle Model Recognition Using A Coarse-to-Fine Convolutional Neural Network Architecture,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, 2017, doi: 10.1109/TITS.2016.2620495.

[31] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, 1998, doi: 10.1109/5.726791.

[32] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, “Face recognition: A convolutional neural-network approach,” *IEEE Trans Neural Netw*, vol. 8, no. 1, 1997, doi: 10.1109/72.554195.

[33] F. Palsson, J. R. Sveinsson, and M. O. Ulfarsson, “Multispectral and Hyperspectral Image Fusion Using a 3-D-Convolutional Neural Network,” *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, 2017, doi: 10.1109/LGRS.2017.2668299.

[34] K. B. Lee, S. Cheon, and C. O. Kim, “A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 30, no. 2, 2017, doi: 10.1109/TSM.2017.2676245.

[35] Y. Zhou, H. Wang, F. Xu, and Y. Q. Jin, “Polarimetric SAR Image Classification Using Deep Convolutional Neural Networks,” *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 12, 2016, doi: 10.1109/ LGRS.2016.2618840.

[36] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, “Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, 2017, doi: 10.1109/TGRS.2016.2612821.

[37] Y. LeCun, C. Cortes, and C. J. C. Burges, “The MNIST database of handwritten digits, 1998,” *URL http://yann. lecun. com/exdb/mnist*, vol. 10, no. 34, 1998.

[38] H. Liu, B. Li, X. Lv, and Y. Huang, “Image Retrieval Using Fused Deep Convolutional Features,” in *Procedia Computer Science*, 2017. doi: 10.1016/j.procs.2017.03.159.

[39] C. Szegedy *et al.*, “Going deeper with convolutions,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015. doi: 10.1109/CVPR.2015.7298594.

[40] W. Zhiqiang and L. Jun, “A review of object detection based on convolutional neural network,” in *Chinese Control Conference, CCC*, 2017. doi: 10.23919/ChiCC.2017.8029130.

[41] C. Szegedy, A. Toshev, and D. Erhan, “Deep Neural Networks for object detection,” in *Advances in Neural Information Processing Systems*, 2013. doi: 10.54097/hset.v17i.2576.

[42] K. Q. Huang, W. Q. Ren, and T. N. Tan, “A review on image object classification and detection,” *Jisuanji Xuebao/Chinese Journal of Computers*, vol. 37, no. 6, 2014, doi: 10.3724/SP.J.1016.2014.01225.

[43] X. Zhang, Y. H. Yang, Z. Han, H. Wang, and C. Gao, “Object class detection: A survey,” *ACM Computing Surveys*, vol. 46, no. 1. 2013. doi: 10.1145/2522968.2522978.

[44] T. Y. Lin *et al.*, “Microsoft COCO: Common objects in context,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014. doi: 10.1007/978-3-319-10602-1_48.

[45] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, “Object Detection in 20 Years: A Survey,” *Proceedings of the IEEE*, vol. 111, no. 3, 2023, doi: 10.1109/JPROC.2023.3238524.

[46] Y. Xiao *et al.*, “A review of object detection based on deep learning,” *Multimed Tools Appl*, vol. 79, no. 33–34, 2020, doi: 10.1007/s11042-020-08976-6.

[47] D. Hoiem, S. K. Divvala, and J. H. Hays, “Pascal VOC 2008 Challenge.”

[Online]. Available: http://www.cs.uiuc.edu/homes/dhoiem/projects/software.html

[48] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the*

IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2014. doi: 10.1109/CVPR.2014.81.

[49] N. Chumerin, "convolutional neural network," 2015.

[50] Y. LeCun, G. Hinton, and Y. Bengio, "Deep learning (2015), Y. LeCun, Y. Bengio and G. Hinton," *Nature*, vol. 521, 2015.

[51] C. Peng *et al.*, "MegDet: A Large Mini-Batch Object Detector," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018. doi: 10.1109/CVPR.2018.00647.

[52] J. Terven and D. Cordova-Esparza, "A Comprehensive Review of YOLO: From YOLOv1 and Beyond," Apr. 2023, [Online]. Available: http://arxiv.org/abs/2304.00501

[53] W. Lan, J. Dang, Y. Wang, and S. Wang, "Pedestrian detection based on yolo network model," in *Proceedings of 2018 IEEE International Conference on Mechatronics and Automation, ICMA 2018*, 2018. doi: 10.1109/ICMA.2018.8484698.

[54] Q. Li, X. Ding, X. Wang, L. Chen, J. Son, and J. Song, "Detection and Identification of Moving Objects at Busy Traffic Road based on YOLO v4," *The Journal of the Institute of Internet, Broadcasting and Communication*, vol. 21, no. 1, 2021.

[55] M. Lippi, N. Bonucci, R. F. Carpio, M. Contarini, S. Speranza, and A. Gasparri, "A YOLO-based pest detection system for precision agriculture," in *2021 29th Mediterranean Conference on Control and Automation, MED 2021*, 2021. doi: 10.1109/MED51440.2021.9480344.

[56] Y. Wang and J. Zheng, "Real-time face detection based on YOLO," in *1st IEEE International Conference on Knowledge Innovation and Invention, ICKII 2018*, 2018. doi: 10.1109/ICKII.2018.8569109.

[57] M. A. Al-masni *et al.*, “Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system,” *Comput Methods Programs Biomed*, vol. 157, 2018, doi: 10.1016/j.cmpb.2018.01.017.

[58] H. M. Ünver and E. Ayan, “Skin lesion segmentation in dermoscopic images with combination of yolo and grabcut algorithm,” *Diagnostics*, vol. 9, no. 3, 2019, doi: 10.3390/diagnostics9030072.

[59] L. Tan, T. Huangfu, L. Wu, and W. Chen, “Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification,” *BMC Med Inform Decis Mak*, vol. 21, no. 1, 2021, doi: 10.1186/s12911-021-01691-8.

[60] L. Cheng, J. Li, P. Duan, and M. Wang, “A small attentional YOLO model for landslide detection from satellite remote sensing images,” *Landslides*, vol. 18, no. 8. 2021. doi: 10.1007/s10346-021-01694-6.

[61] P. Kumar, S. Narasimha Swamy, P. Kumar, G. Purohit, and K. S. Raju, “Real-time, YOLO-based intelligent surveillance and monitoring system using jetson TX2,” in *Lecture Notes on Data Engineering and Communications Technologies*, vol. 54, 2021. doi: 10.1007/978-981-15-8335-3_35.

[62] K. Bhambani, T. Jain, and K. A. Sultanpure, “Real-Time Face Mask and Social Distancing Violation Detection System using YOLO,” in *Proceedings of B-HTC 2020 - 1st IEEE Bangalore Humanitarian Technology Conference*, 2020. doi: 10.1109/B-HTC50970.2020.9297902.

[63] Y. Du, N. Pan, Z. Xu, F. Deng, Y. Shen, and H. Kang, “Pavement distress detection and classification based on YOLO network,” *International Journal of Pavement Engineering*, vol. 22, no. 13, 2021, doi: 10.1080/10298436.2020.1714047.

[64] Hendry and R. C. Chen, “Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning,” *Image Vis Comput*, vol. 87, 2019, doi: 10.1016/j.imavis.2019.04.007.

[65] C. Dewi, R. C. Chen, X. Jiang, and H. Yu, “Deep convolutional neural network for enhancing traffic sign recognition developed on Yolo V4,” *Multimed Tools Appl*, vol. 81, no. 26, 2022, doi: 10.1007/s11042-022-12962-5.

[66] A. M. Roy, J. Bhaduri, T. Kumar, and K. Raj, “WilDect-YOLO: An efficient and robust computer vision-based accurate object localization model for automated endangered wildlife detection,” *Ecol Inform*, vol. 75, 2023, doi: 10.1016/j.ecoinf.2022.101919.

[67] D. H. Dos Reis, D. Welfer, M. A. De Souza Leite Cuadros, and D. F. T. Gamarra, “Mobile Robot Navigation Using an Object Recognition Software with RGBD Images and the YOLO Algorithm,” *Applied Artificial Intelligence*, vol. 33, no. 14, 2019, doi: 10.1080/08839514.2019.1684778.

[68] O. Sahin and S. Ozer, “YOLODrone: Improved YOLO Architecture for Object Detection in Drone Images,” in *2021 44th International Conference on Telecommunications and Signal Processing, TSP 2021*, 2021. doi: 10.1109/TSP52935.2021.9522653.

[69] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016. doi: 10.1109/CVPR.2016.91.

[70] R. Girshick, “Fast R-CNN,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Dec. 2015, pp. 1440–1448. doi: 10.1109/ICCV.2015.169.

[71] J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017. doi: 10.1109/CVPR.2017.690.

[72] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems*, 2015.

[73] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “Yolov4: Optimal speed and accuracy of object detection,” 2020.

[74] G. Jocher, “YOLOv5 by Ultralytics.” Accessed: Nov. 08, 2023. [Online]. Available: https://github.com/ultralytics/yolov5

[75] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, “Distance-IoU loss: Faster and better learning for bounding box regression,” in *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, 2020. doi: 10.1609/aaai.v34i07.6999.

[76] D. Hendrycks and K. Gimpel, “Gaussian error linear units (gelus)”.

[77] G. Ghiasi *et al.*, “Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021. doi: 10.1109/CVPR46437.2021.00294.

[78] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “MixUp: Beyond empirical risk minimization,” in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.

[79] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, “Albumentations: Fast and flexible image augmentations,” *Information (Switzerland)*, vol. 11, no. 2, 2020, doi: 10.3390/info11020125.

[80] C. Li *et al.*, “YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications,” 2022.

[81] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors,” 2023. doi: 10.1109/cvpr52729.2023.00721.

[82] C. Y. Wang, H. Y. M. Liao, and I. H. Yeh, “Designing Network Design Strategies Through Gradient Path Analysis,” *Journal of Information Science and Engineering*, vol. 39, no. 2, 2023, doi: 10.6688/JISE.202307_39(4).0016.

[83] M. Contributors, “YOLOv7 by MMYOLO.” Accessed: Nov. 08, 2023. [Online]. Available: https://github.com/open-mmlab/mmyolo/tree/main/configs/yolov7

[84] G. Batnasan, M. Gochoo, M. E. Otgonbold, F. Alnajjar, and T. K. Shih, “ArSL21L: Arabic Sign Language Letter Dataset Benchmarking and an Educational Avatar for Metaverse Applications,” in *IEEE Global Engineering Education Conference, EDUCON*, IEEE Computer Society, 2022, pp. 1814–1821. doi: 10.1109/EDUCON52537.2022.9766497.

[85] “Kaggle Arabic Sign Language Dataset 2022.” Accessed: Nov. 23, 2023. [Online]. Available: https://www.kaggle.com/datasets/ammarsayedtaha/arabic-sign-language-dataset-2022/data

[86] “Roboflow.” Accessed: Nov. 23, 2023. [Online]. Available: https://roboflow.com/

[87] Gaudenz Boesch, “LabelImg for Image Annotation.” Accessed: Nov. 10, 2023. [Online]. Available: https://viso.ai/computer-vision/labelimg-for-image-annotation/

[88] G. Jocher, A. Chaurasia, and J. Qiu, “YOLOv8 by Ultralytics.” Accessed: Nov. 10, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[89] Rui-Yang Ju and Weiming Cai, “Fracture Detection in Pediatric Wrist Trauma X-ray Images Using YOLOv8 Algorithm,” 2023.