**College of Computer and Information Sciences**
**Computer Science Department**

# "Multiple Bleeding Detection in Wireless Capsule Endoscopy"

*CSC 497– Final Report*

## Prepared by:


| | |
|---|---|
| Lena Alotaibi | 441200984 |
| Noura Almhizea | 441201055 |
| Shouq Alzeer | 441201267 |
| Sara Almuhanna | 441201325 |
| Ghaida Alkhudhair | 441926684 |


## Supervised by:

Prof. Ouiem Bchir

Research project for the degree of Bachelor in Computer Science
First/Second Quarter 1444

# English Abstract

Wireless Capsule Endoscopy (WCE) is a diagnostic technology for gastrointestinal tract pathology detection. It has emerged as an alternative to conventional endoscopy which could be distressing to the patient. However, the diagnosis process requires to view and analyze hundreds of frames extracted from WCE video. This makes the diagnosis tedious. For this purpose, researches related to the automatic detection of signs of gastrointestinal diseases have been boosted.

In this project, we design a pattern recognition system for detecting Multiple Bleeding Spots (MBS) using WCE video. The proposed system relies on the Deep Learning approach to accurately recognize multiple bleeding spots in the gastrointestinal tract. Specifically, the You Only Look Once (YOLO) Deep Learning models are explored in this research. Namely, YOLOv3, YOLOv4, YOLOv5 and YOLOv7

The results of experiments showed that YOLOv7 is the most appropriate model for designing the proposed MBS detection system. Specifically, the proposed system achieved a mAP of 0.86, and an IoU of 0.8. Moreover, the results of the detection were enhanced by augmenting the training data to reach a mAP of 0.883.

**Keywords**: Wireless Capsule Endoscopy (WCE), Multiple Bleeding Spots (MBS), Gastrointestinal (GI) Disease, Deep Learning, Pattern Recognition.

# Arabic Abstract


منظـار الكبسـولة اللاسـلكي (WCE) هـي تقنيـة تشـخيص للكشـف عـن أمـراض الجهـاز الهضـمي. ظهـرت هـذه التقنيـة كبـديل للمنظـار الـداخلي التقليـدي الـذي يمكـن أن يكـون مزعجًـا للمـريض. ولكـن تتطلـب عمليـة التشـخيص بمنظـار الكبسـولة اللاسـلكي عـرض وتحليـل مئـات الصـور المسـتخرجة مـن الفيـديو WCE . هـذا يجعـل عمليـة التشـخيص مضـنيةً. لهـذا الغـرض تـم تعزيـز الأبحـاث المتعلقـة بالكشـف التلقـائي عـن علامـات أمـراض الجهـاز الهضـمي.

فـي هـذا المشـروع، نعـزم علـى تصـميم نظـام يتعـرف علـى نمـط بقـع النزيـف المتعـدد المتواجـد فـي الجهـاز الهضـمي للإنسـان باسـتخدام فيـديو تقنيـة منظـار الكبسـولة اللاسـلكي (WCE). النظـام المقتـرح يعتمـد علـى الـتعلم العميـق للتعـرف بدقـة علـى بقـع النزيـف المتعـددة فـي الجهـاز الهضـمي. علـى وجـه التحديـد، سـيتم اسـتعمال نمـاذج الـتعلم العميـق You Only Look Once (YOLO) في هذا البحث.

أظهـرت نتـائج التجـارب أن الإصـدار السـابع مـن نمـوذج YOLO هـو النمـوذج الأنسـب لتصـميم النظـام المقتـرح. علـى وجـه التحديـد، حققـت نتـائج النظـام علـى mAP بقيمـة 0.86 وIoU بقيمـة 0.8. بالإضـافة علـى ذلـك، تـم تحسـين النتـائج عـن طريـق زيـادة البيانـات المستخدمة في تدريب النموذج حيث حصلت على mAP بقيمة 0.883.

# Table of Contents

# List of Tables

# List of Figures

# List of Abbreviations

| | |
|---|---|
| ANN | Artificial Neural Networks |
| AP | Average Precision |
| AUC | Area Under the Curve |
| BoF | Bag of Freebies |
| BoS | Bag of Specials |
| CBL | Convolution, Batch normalization, Leaky ReLU actviation function |
| CE | Capsule Endoscopy |
| CNN | Convolutional Neural Networks |
| CSP | Cross Stage Partial |
| DL | Deep Learning |
| DRF | Dense Region Fusion |
| DWT | Discrete Wavelet Transform |
| E-ELAN | Extended Efficient Layer Aggregation Network |
| EM | Expectation Maximization |
| FCM | Fuzzy C-means Clustering Algorithm |
| FLOP | Floating Point Operations per second |
| FN | False Negatives |
| FP | False Positives |
| FPN | Feature Pyramid Network |
| GAN | Generative Adversarial Networks |
| GI | Gastro-Intestinal |
| GLCM | Gray-Level Co-occurrence Matrix |
| GPU | Graphics Processing Units |
| ILSVRC | ImageNet Large Scale Vision Recognition Competition |
| IoU | Intersection over Union |
| KNN | K-Nearest Neighbors |
| LBP | Local Binary Pattern |
| LMT | Logistic Model Tree |
| mAP | mean Average Precision |
| MBS | Multiple Bleeding Spots |
| ML | Machine Learning |
| MPEG7 | Multimedia Content Description Interface |
| MRC | Multiregional Region Combination |
| NA | Not Available |
| PANet | Path Aggregation Network |
| ReLU | Rectified Linear Unit |

| | |
|---|---|
| ResNet | Residual Network |
| RF | Random Forest |
| RGB | Red-Green-Blue |
| RPR | Region Proposal Rejection |
| RT | Random Tree |
| SegNet | Semantic Segmentation |
| SPP | Spatial Pyramid Pooling |
| SRS | Salient Region Segmentation |
| SVM | Support Vector Machine |
| TL | Transfer Learning |
| TP | True Positives |
| U-Net | Convolutional Networks for Biomedical Image Segmentation |
| VCE | Video Capsule Endoscopy |
| VGG | Visual Geometry Group |
| WCE | Wireless Capsule Endoscopy |
| YOLO | You Only Look Once |

# Chapter 1: Introduction

The digestive system disorders have been a concern for physicians over years. In fact, millions of people around the world suffer from gastrointestinal (GI) diseases. Specifically, among more than 73 thousand participants in a worldwide study, 40% of them have functional gastrointestinal disorders [1]. In addition, disorders such as digestive system cancer are considered fatal and a major cause of mortality according to 2020 United States statistics [2].

Several pathogens can affect the gastrointestinal tract such as inflammation, infection, cancers, benign tumors, ulcers, and hemorrhoids. Some of these pathogens have similar symptoms. Specifically, cancer, benign tumors, ulcers, and hemorrhoids may yield Multiple Bleeding Spots (MBS) in the gastrointestinal tract [3]. The latter symptom consists of a loss of blood in the GI tract because of ruptured vessels indicating the presence of an abnormality [4]. These MBS appear as small dark red spots [Fig. 1, a] or as small light spots next to the red dark ones [Fig. 1, b].


(a) (b)

Fig. 1. WCE images with multiple bleeding spots which appear as (a) dark red spots, or (b) light spots [5].

Fortunately, with the emergence of new diagnostic techniques, it is possible for physicians to detect GI abnormalities. Endoscopy is the most common diagnostic technique for GI tract. Nerveless, it is inconvenient and painful for the patient. In order to alleviate this inconvenience, Wireless Capsule Endoscopy (WCE) developed in 2000, emerged as a new diagnostic technique. The diagnosing process consists of the patient swallowing a capsule as shown in [Fig. 2]. The latter contains a camera to record the journey of the

capsule internally to the GI tract. Then, the physician analyses the record to diagnose the patient by looking for abnormal spots.


Fig. 2. WCE Structure: (1) Optical dome (2) Lens holder (3) Lens (4) Illuminating Lens (5) CMOS (6) Battery (7) ASIC transmitter (8) Antenna [6].

## 1.1 Problem Statement

WCE generates an eight-hour video. In other words, 60,000 frames need to be visualized by the physician [7]. However, due to the small size of the lesion region and the visual fatigue, the disease diagnosis may be missed at an early stage. In light of this, a diagnostic technology related to image processing and pattern recognition would help in the rapid and accurate detection of the disease. Nevertheless, due to the likeness of the MBS and other intestinal characteristics such as, bubbles, holes, or small food debris, etc. as illustrated in [Fig. 1, Fig. 3], it is challenging to extract visual descriptors able to distinguish MBS pattern from the other ones. It is even more arduous due to the background clutter. In fact, MBS can occur in all parts of the GI tract exhibiting large variety of background in terms of color, and texture as shown in [Fig. 1]. One way to tackle this problem is through the use of Deep Learning (DL) models which learn automatically suitable features.


Fig. 3. Likeness of the MBS and other intestinal characteristics [5].

## 1.2 Goals and Objectives

The major goal of this project is to design a pattern recognition system to detect multiple bleeding spots from WCE videos using deep learning models designed for pattern recognition. In particular, recent You Only Look Once (YOLO) models are trained and tested. In this regard, the following research questions are addressed:

1. Is YOLO model able to detect multiple bleeding spots?
2. Which version of YOLO is better in terms of MBS recognition performance and time efficiency?
3. Dose data augmentation technique improve MBS recognition performance ?

Moreover, the project intend to:

- Investigate the background of the field of object recognition using deep learning.
- Review related works related to detecting MBS from WCE videos.
- Evaluate the performance of considered models and analyze their results.

## 1.3 Proposed Solution

In this research, we develop a multiple bleeding spot detection system for Wireless Capsule Endoscopy (WCE) videos. More specifically, we design a pattern recognition system based on deep learning models that are able to detect the bleeding spots through the GI tract. In particular, deep learning models adopted for pattern recognition were utilize. These models are designed to localize and categorize the object of interest. For this purpose, we employ the You Only Look Once (YOLO) deep learning approach [8]. In this regard, we propose to compare different versions of YOLO. These are YOLOv3 [9], YOLOv4 [10], YOLOv5 [11], and YOLOv7 [12].

## 1.4 Research Scope

This project takes into consideration the detection of multiple bleeding spots in the gastrointestinal tract. The detection is performed on frames (images) extracted from the WCE video. For this purpose, the performance of different YOLO models is compared. Namely, these are YOLOv3 [9], YOLOv4 [10], YOLOv5 [11], and YOLOv7 [12].

## 1.5 Research Significance

Over time, technology has revolutionized our way of life by providing many facilities in different fields. In particular, the medical field has benefited from new technologies, especially with time sensitive tasks, like diagnosing diseases. WCE is one of the diagnostic systems that help diagnosing MBS, and thus treating it at an early stage. Furthermore, the research advances in pattern recognition and image processing, especially with the deep learning boost, improved the automation of such diagnostic technologies. In particular, the deep learning recognition model, YOLO, would benefit the automation of MBS detection from WCE videos.

## 1.6 Ethical and Social Implications

MBS recognition system would benefit society by assisting physicians through the diagnosis procedure as well as making it more convenient. In addition, reducing the time required of resolving the diagnosis of MBS, allows the patient's treatment at early stage and thus increases the probability of the patient's recovery.

The dataset that is exploited in this research is collected with the approval and the consent of the hospital. Moreover, the collected data does not include any information that would identify the patient directly or indirectly, and thus does not expose any personal information of the patients.

## 1.7 Report Organization

The rest of the report is organized as follows. Chapter 2 highlights the background related to the research. Specifically, it explores Convolutional Neural Networks (CNN) and particularly YOLO model. As for chapter 3, it reviews related works to MBS detection from WCE videos. These are conventional pattern recognition approaches and deep learning-based ones. Alternatively, chapter 4 describes the proposed approach. Chapter 5 explains the experimental setting. Furthermore, chapter 6 addresses the environment including the programming language and tools used to conduct the experiments, whereas chapter 7 shows and discusses the experiment results. Finally, chapter 8 concludes this report and outlines the future works.

# Chapter 2: Background

In this project, we aim to design a pattern recognition system based on deep learning models that are able to detect the bleeding spots from the GI tract. In particular, deep learning models adopted for pattern recognition are utilized. These models are designed to localize and categorize the object of interest. For this purpose, we intend to employ YOLO [8] deep learning models. As such, in this chapter, Deep Learning [13] paradigm is first introduced. Particularly, Convolutional Neural Networks (CNN) are described. Then, YOLO [8] model architecture is described. Finally, Transfer Learning and Data Augmentation are presented.

## 2.1 Deep Learning

Deep Learning (DL) is sub-field of Machine Learning (ML) which has been proven to solve real world problems such as speech recognition, natural language processing, and image recognition [13]. DL uses multilayer neural networks, and it is designed to mimicry the human's behavior of solving problems and acquiring knowledge. In particular, the human's neural system has been an inspiration of one of the main models employed by DL. This model is Artificial Neural Networks (ANN). The latter contains neurons that are connected to process the outcome. The connections convey information from one neuron to another. The neuron processes then the received information and feed it to the forward neurons connected to it. The information processing at the neuron level consists of a weighted sum of the input information. The corresponding weights are represented by the connections referred to as edges. They constitute the model to be learned. Alternatively, the neurons are arranged into layers in such a way that the information is transferred from the first layer, referred to, input layer, to the last layer after going through one hidden layer or more [14]. A simple neural network composed of an input layer, one hidden layer, and an output layer are shown in [Fig. 4].

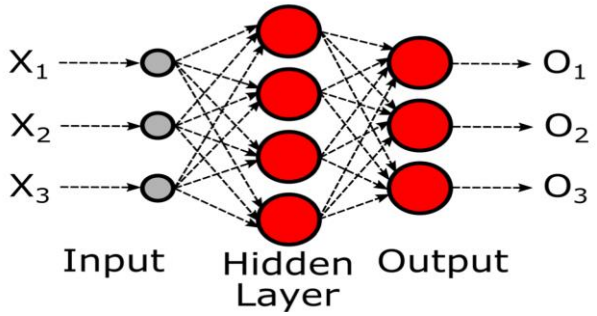



Fig. 4. The structure of a simple neural network [15].

## 2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of deep learning paradigm. It was initially designed to solve image classification problems [15]. CNN model architecture consists of multi-layer neural network. More specifically, CNN model is composed of three types of layers which are convolutional layers, pooling layers, and fully connected layers. These layers are stacked in particular order to define the CNN deep learning model.

In the context of image classification, the input of CNN is either a colored image or gray scaled one. They are presented as matrix of pixel values. In the case of colored image, it is 3D matrix, whereas it is a 2D matrix in the case of gray level image. [Fig. 5] depicts a colored image in the RGB domain. It is constituted of 3 channels, Red, Green, and Blue. Each channel is represented by a 2D matrix of values between 0 and 255.

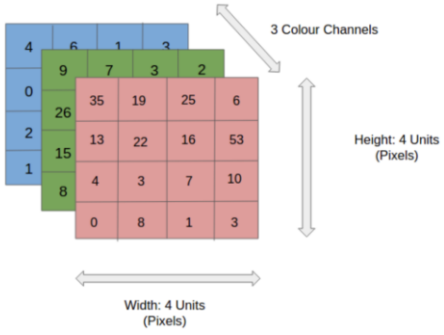



Fig. 5. Sample RGB image of size 4 × 4 × 3 [16].

As shown in [Fig. 6], the following layers form a sequence of convolution layers, and pooling layers. They are responsible of automatic feature extraction. The output of this

feature extraction module is flattened and fed to a fully connected network. The latter maps the extracted features into the final output. The fully connected network and the Softmax layer constitute the classification module.

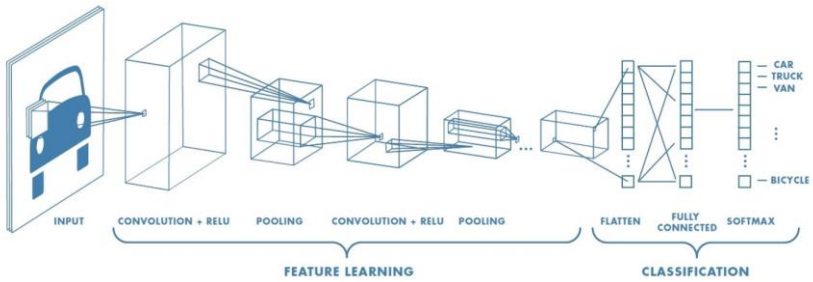



Fig. 6. Illustrative example of a CNN architecture [16].

Particularly, the convolution layer is constituted of a set of convolutional filters, which are convolved with the input feature map (the input image in the case of the first layer) to produce an output feature map [17]. Specifically, the convolution filter is a 2D matrix referred to as kernel. Its dimension is much smaller compared to the size of the image. When the convolution stride is set to 1, it is applied to each pixel of the image by convoluting the filter parameter with the pixel neighboring pixels. When the stride is greater than one, a number of pixels equal to the stride is skipped. It seeks to reduce the size of the feature map. [Fig. 7] shows an illustrative example of 3×3 convolution with a stride of 2. At the convolution layer, a set of convolution filters are applied. Then, on the obtained convolution output, an activation function such as the Rectified Linear Unit (ReLU) is performed to provide nonlinearity to the feature map [18].

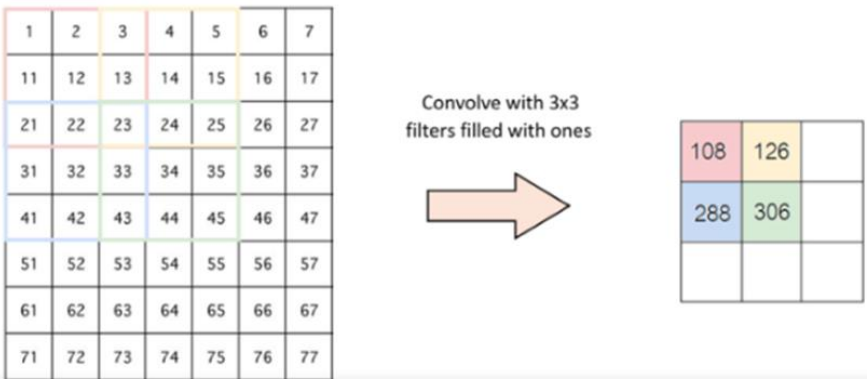



Fig. 7. Illustrative example of 3×3 convolution with a stride of 2 [19].

To reduce the dimensionality of the feature map, pooling layers are utilized [20]. The process involves choosing a representative for each feature map window. The window size is a parameter of the model. Likewise, to the convolution layer operation, the pooling layer operation may not be applied to every pixel. A stride parameter is defined to select pixels to be ignored. There are many types of pooling such as max, average, and sum [20]. [Fig. 8] depicts an illustrative example where the Max pooling function is adapted with a window size of 2×2 and a stride of 2. As shown in [Fig. 8], from the 2×2 green window, the maximum value is selected as representative and kept in the next layer. Similarly, from the other windows blue, yellow, and red which are defined with a stride of 2, the maximum pixel value is kept.

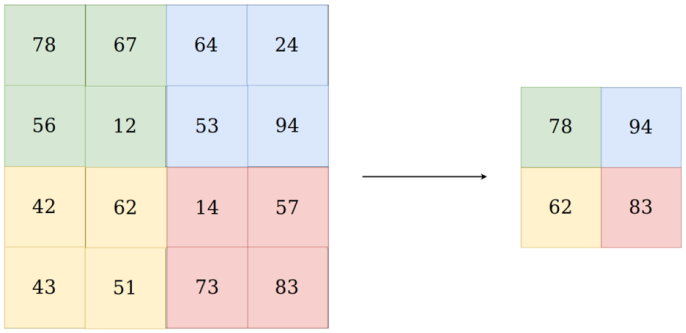



Fig. 8. Max pooling applied on 2×2 window size [19].

The number of convolution and pooling layers, their parameters, and their order of occurrence within the network are defined by the CNN model architecture.

The final feature map is flattened and conveyed to fully connected layers [21]. The latter are commonly used for classification. As displayed in [Fig. 9] the neurons of these layers are connected to all neurons of the previous layer [21]. Similarly, the number of fully connected layers and the number of their corresponding nodes are defined by the deep learning model.

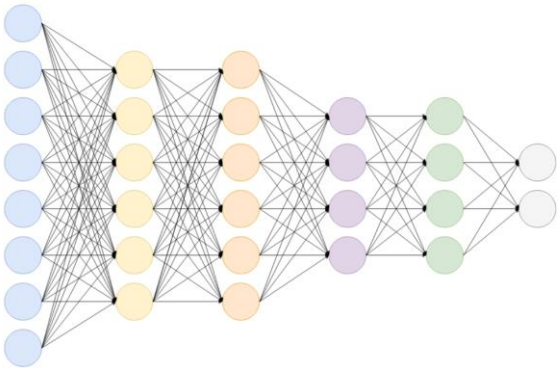


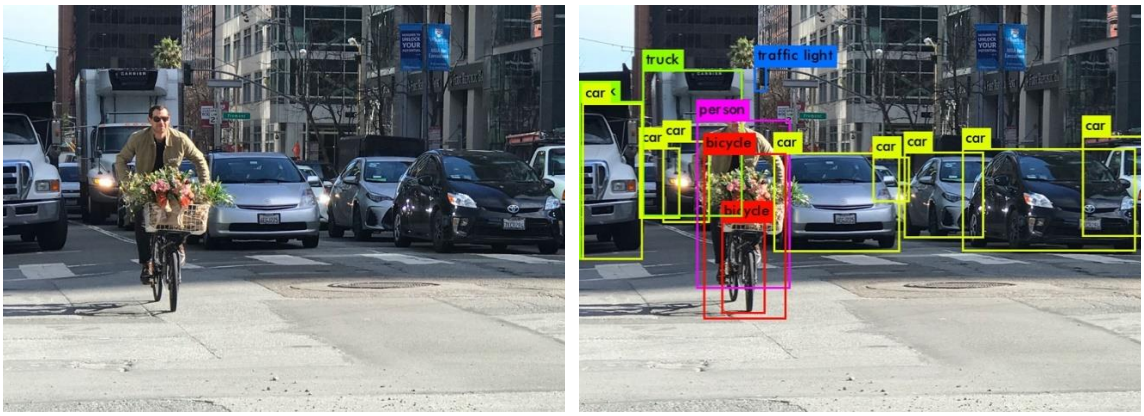
Fig. 9. Fully connected layers [19].

CNN models have been proven to be successful for many application [22]. In this regard, many architectures have been proposed such as AlexNet [23], VGG16 [24], GoogleNet [25], or ResNet [26]. Nevertheless, typical CNN models are not suitable for pattern recognition problems. For this purpose, DL models for pattern recognition have been proposed in the literature. In the following, we present the You Only Look Once (YOLO) [8] deep learning paradigm designed for pattern recognition.

## 2.3 You Only Look Once

You Only Look Once (YOLO) [8] is a deep learning paradigm which recognizes one or more object of interest within the image. In fact, it localizes the position of the object and categorize it. The localization is predicted through the learning of the bounding box that surrounds the object of interest. [Fig. 10, b] shows the bounding boxes and their corresponding labels learned by YOLO when the image shown in [Fig. 10, a] is conveyed to the input.

YOLO splits the image into S×S blocks. From each block, only one object which center belongs to the block is detected. This is referred to as the one object rule. Nonetheless, the object can spread over many blocks. Moreover, for each object, B bounding boxes can be predicted. [Fig. 11] illustrates an image split into 7×7 blocks. Two bounding boxes are predicted with respect to each object. As it can be seen, for the object "person riding the bicycle" which centroid lays in the yellow block, two blue rectangles are considered as tentative bounding boxes.




(a) (b)

Fig. 10. Illustrative example of YOLO object detection. (a) The input image, (b) the obtained results [8].

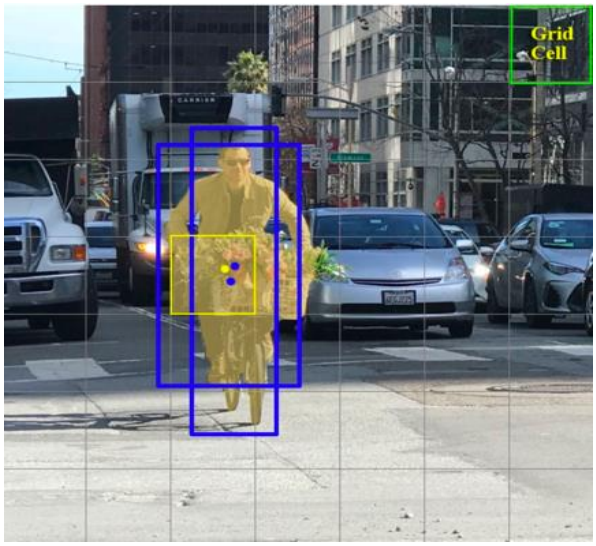



Fig. 11. Illustrative example of predicting two bounding boxes per block [8].

To predict a bounding box, YOLO learns 5 parameters. They define the upper left coordinates, the height and width, and the confidence score of the box. The box confidence score (objectness) is the likelihood an object resides inside the box. Furthermore, YOLO predicts conditional class probabilities with respect to all considered classes. As a result, YOLO learns $(5 * B + C) * S * S$ values where 5 represent the YOLO parameters, B is the bounding box, C is the number of classes, and S×S is the grid dimension. In order to learn these values, YOLO exploits the CNN model which output is a tensor of size $(S, S, 5 * B + C)$. [Fig. 12] displays an example of YOLO model which uses a 7×7 blocks, estimates two boundary boxes per block, and considers 15 classes. Therefore, the size of output the tensor is (7,7, 2×5, 15).

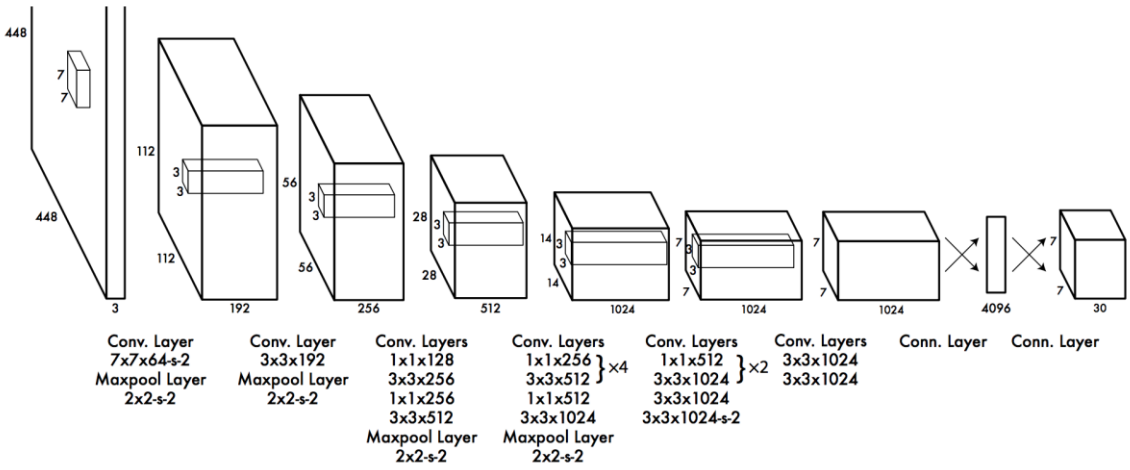



Fig. 12. YOLO architecture [8].

Using the predicted values at the output tensor, only one object is selected per block. It is the one having the highest box confidence score and the highest conditional probability. As such, a class confidence score is calculated as the product of the box confidence score and the conditional probability. It estimates the confidence on the localization and on the classification results. Finally, YOLO uses the Non-maximal suppression algorithm to deal with duplicated objects. In fact, if two detected objects overlap, the one which has the lowest class confidence is omitted.

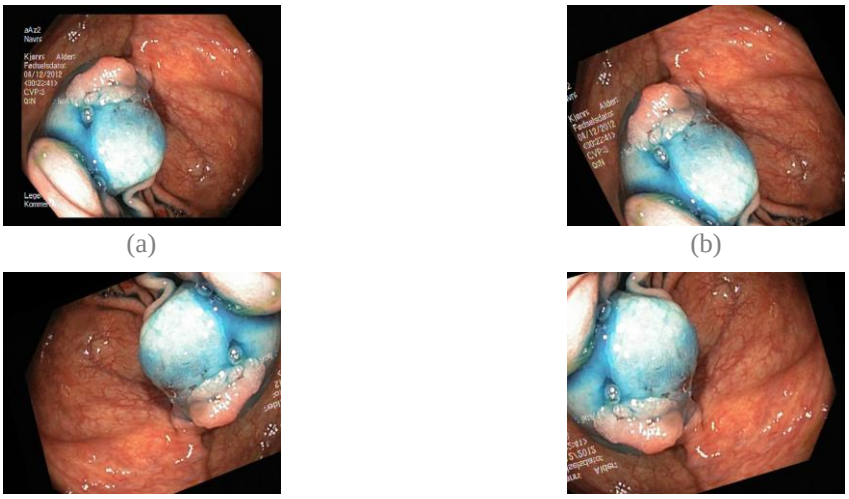
## 2.4 Transfer Learning and Data Augmentation

Training deep neural network requires the availability of large amount of data. Nevertheless, this is not always possible. In this case, techniques such as transfer learning and data augmentation are helpful.

Transfer Learning (TL) is a technique used to train deep learning model using a small amount of data [27]. It consists of using the knowledge acquired while addressing one problem, to solve a different problem. For instance, the trained model obtained to classify animals, can be adapted to classify furniture. In fact, the trained model parameter, solving the first problem, are used as starting parameters for the second model which aims at solving the second problem. In fact, the weights are transferred from the first model to the second one. As such, a smaller training dataset is sufficient to train the second model

and adapt it to solve the new problem. This significantly improves the efficiency of the second model in terms of training time and performance.

Alternatively, Data Augmentation is another technique which seeks increasing the data size [27]. To the original dataset, it adds new instances which are modified versions of the existing ones, or synthetically generated ones. Its objective is to overcome the overfitting problem when the size of the original data is not sufficient. In fact, the latter should be proportional the model number of parameters and to the complexity of the problem. The additional data is obtained by incorporating minor alterations to the original images. These alterations consist of small changes like translations, rotations, and scaling. It can also be generated by modifying synthetically the brightness, the orientation, the location, or adding noise to the image as [Fig. 13] shows.



(a) (b) (c) (d)

Fig. 13. Illustrative examples of augmentation operations (a) original image, (b) shearing and rotation, (c) rotation and shifting, (d) rotation and zooming [28].

These simple alterations may not be sufficient to describe all the situations of the real-world data. In this regard, a more powerful tool such as Generative Adversarial Networks (GANs) [29] is needed. It is a deep learning approach that generates new instances. Specifically, it trains two models. The first one generates new images, while the second one categorizes the obtained images as “real” or “fake”. These two networks are trained simultaneously until the second model classifies half of the generated images as “real”. This indicates that the first network is able to produce real images. In a way, it learns

the input data structure, and use it to output new realistic instances. For instance, GANs can change summer scene images to winter ones, or day images to night ones. [Fig. 14] shows an example of GANs transformation.


Fig. 14. Illustrative example of GAN generated images [27].

## 2.5 Summary

This chapter covers topics related to the proposed approach. Specifically, it presents Deep Learning, Convolutional Neural Networks (CNN), and YOLO basic architecture. Moreover, Transfer Learning and Data Augmentation are introduced.

# Chapter 3: Related Works

Recent researches have proposed aided-diagnosis systems for bleeding anomalies within the intestinal tract using WCE images. They can be categorized into (i) classification-based approaches, and (ii) detection-based approaches. The former approaches classify the whole WCE frame as including bleeding spots or not including bleeding spots. Whereas, the detection approaches not only classify the frame but also localize the bleeding spots within the frame. Moreover, each of these two categories bifurcates into conventional and deep learning approaches according to the machine learning paradigm that have been adopted. More specifically, conventional approaches use "engineered" features (also referred to as hand crafted features). Alternatively, deep learning approaches automatically extract the feature while training the deep learning model.

## 3.1 WCE Frame Classification System

### 3.1.1 Conventional Approaches

The work in [7] propose to classify WCE frames into "Bleeding" and "No Bleeding". For this purpose, it extracts a hand-crafted feature, namely, the color moment feature from WCE frames. Then, it is conveyed to a Support Vector Machine (SVM) [30] classifier. The choice of the visual feature to be adopted has been made through empirical experimentation. In fact, MPEG-7 visual descriptors, "color moment", "Discrete Wavelet Transform", "Edge Histogram Descriptor", "Gabor", and a combination of "Discrete Wavelet Transform" and "color moment".

Similarly, the proposed system in [31] extracts hand crafted features. More specifically, MPEG-7 features [33] are considered. These are the "color moments", the "color histogram", the "local color moments", the "Gabor filter", the "Discrete Wavelet Transform" (DWT) and the "Local Binary Pattern" (LBP) features [33]. The extracted features are then conveyed to a machine learning approach to categorize the frames as "Bleeding" or "No Bleeding". As shown by [Fig. 15], this is performed by clustering each of the training "Bleeding" frames, and the training "No Bleeding" frames into similar groups using Fuzzy C-Means (FCM) [34]. As such, in the testing phase, the unknown frame

is compared to the obtained cluster centroids from the training phase. It is then assigned to class of the closest centroid.

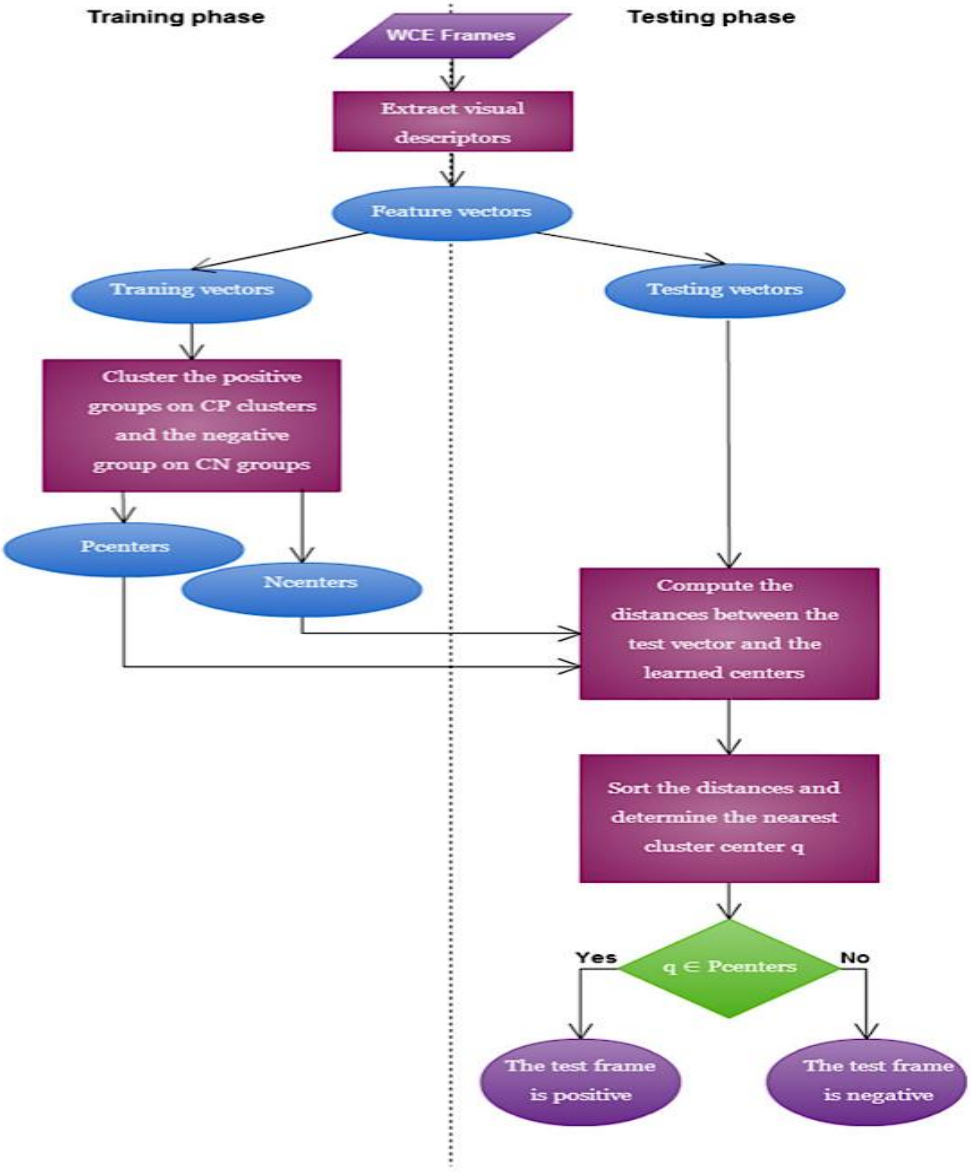



Fig. 15. Flowchart of the proposed approach in [31].

### 3.1.2 Deep Learning Approaches

The authors in [35] use a well-known CNN model that won of the ImageNet Large Scale Vision Recognition Competition (ILSVRC). Specifically, it exploits LeNet-5 [36] architecture as shown in [Fig. 16].

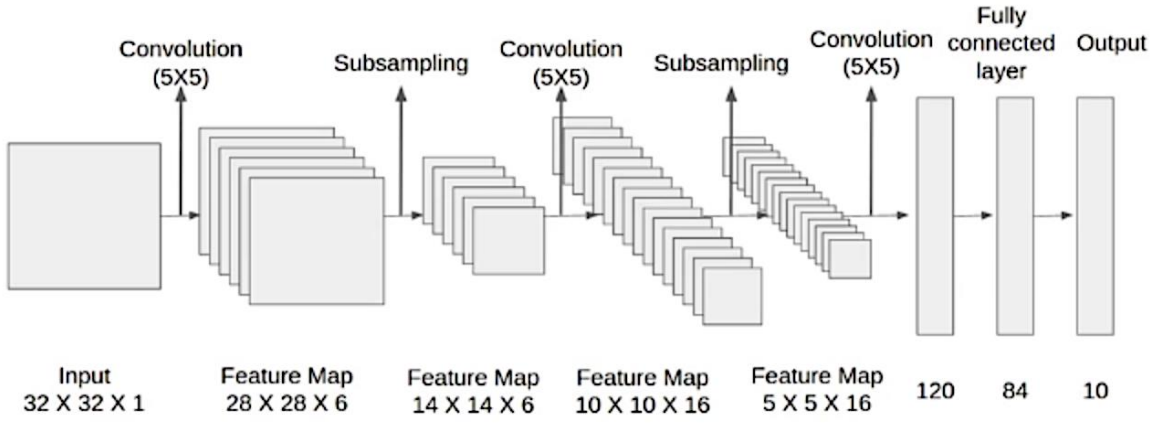



Fig. 16. Architecture of the proposed system in [35].

Alternatively, the work in [37] uses deep learning CNN models for feature extraction. In particular, VGG-19 [24], ResNet50 [39], and InceptionV3 [39] are adopted. Similarly, these are well known CNN models which won the ILSVRC competition. [Fig. 17] and [Fig. 18] show the model architectures of VGG-19 and ResNet50, respectively. Nevertheless, inceptionV3 is an evolved version of InceptionV1 used in GoogleNet. [Fig. 19] displays the architecture of inceptionV3.

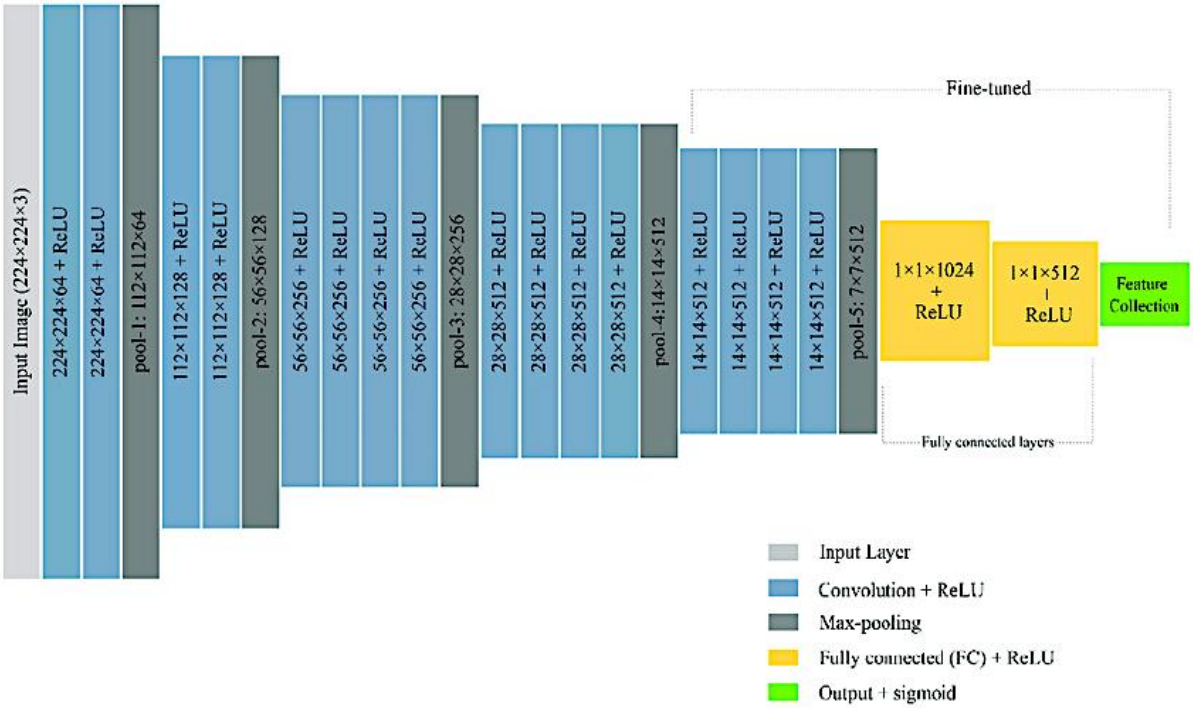



Fig. 17. VGG-19 model architecture [40].

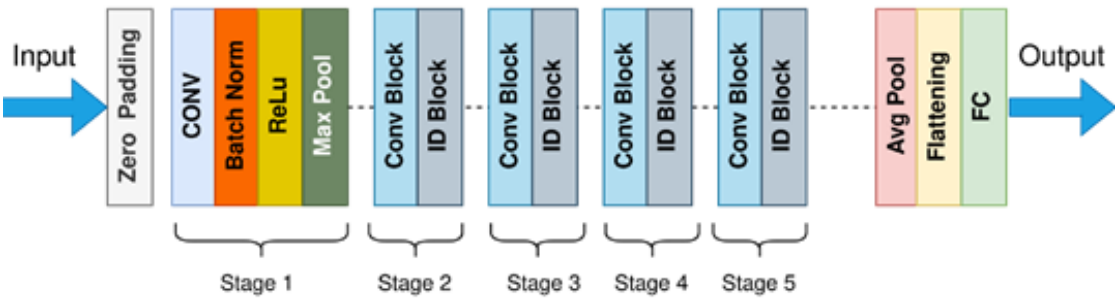



Fig. 18. ResNet50 model architecture [39].

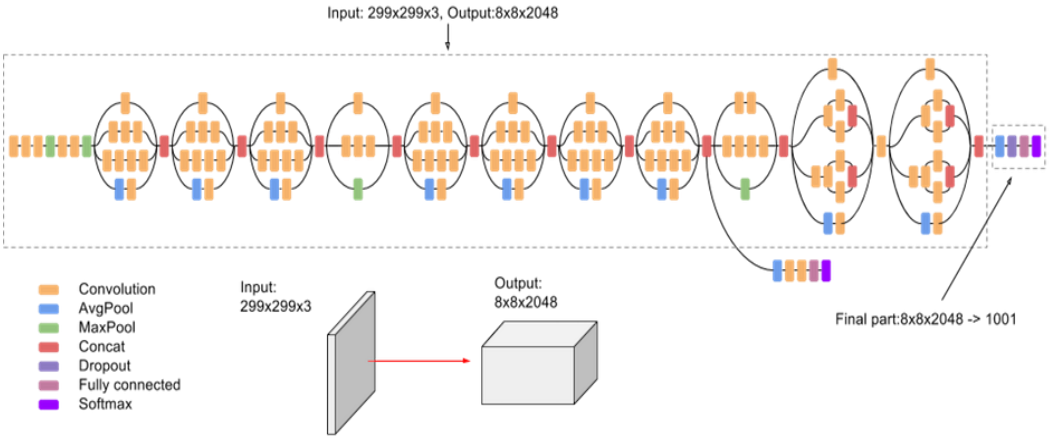



Fig. 19. InceptionV3 GoogleNet model [41].

The obtained features from the three considered models are concatenated. Then, a feature selection is performed to select the most distinctive features as illustrated in [Fig. 20]. The selected features are conveyed to SVM classifier [30] to categorize the frames as “Bleeding” or “No Bleeding”.

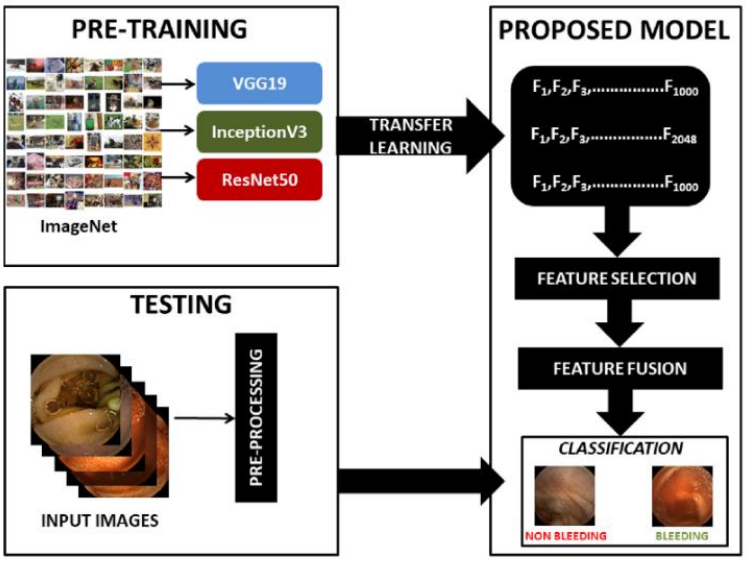



Fig. 20. Overview of the proposed system in [37].

The study in [42] proposed a system to diagnose the abnormalities in the GI. This study proposed a model which utilizes MobileNet [43]. The latter is a lightweight deep learning model. Specifically, as shown in [Fig. 21], it uses the independent convolutions for each depth dimension, then employs 1×1 pointwise convolution to recover the depth. As shown in [Fig. 22], the output of MobileNet [43] is fed to a custom built convolutional neural network model. It is constituted of 64 filters with a kernel size of 3×3. The resulting feature map is passed to a 3 fully connected layers for classification purpose.

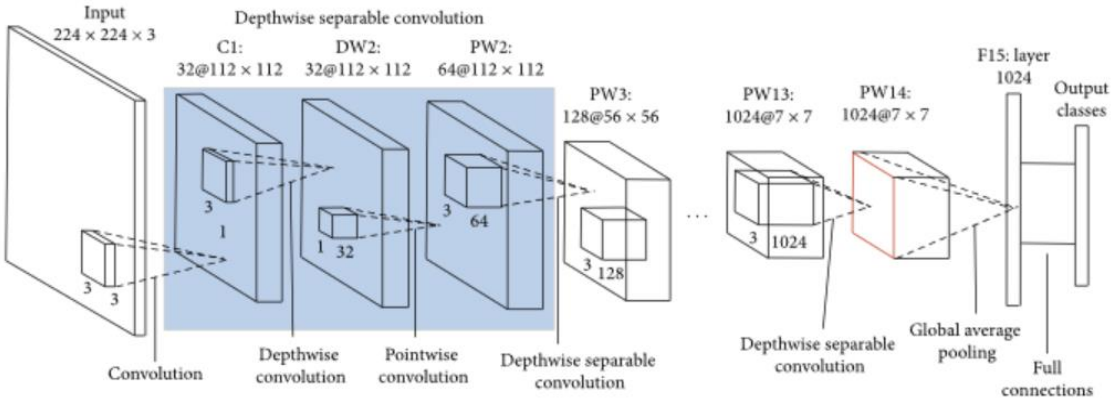



Fig. 21. MobileNet model architecture [43].

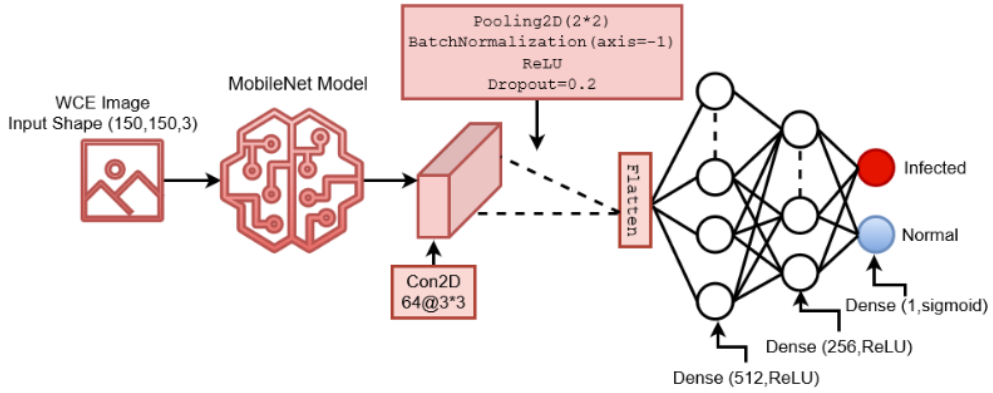



Fig. 22. Architecture of the proposed system in [42].

In [44] authors proposed to classify WCE frames as “Bleeding” and “No Bleeding”. They employ a customized CNN model architecture. As sown in [Fig. 23], it consists of an eight-layer convolutional neural network that is composed of three convolutional layers (C1-C3), three pooling layers (MP1-MP3) and two fully connected layers (FC1, FC2). Moreover, Support Vector Machine (SVM) [30] classifier is utilized instead of the Softmax layer.

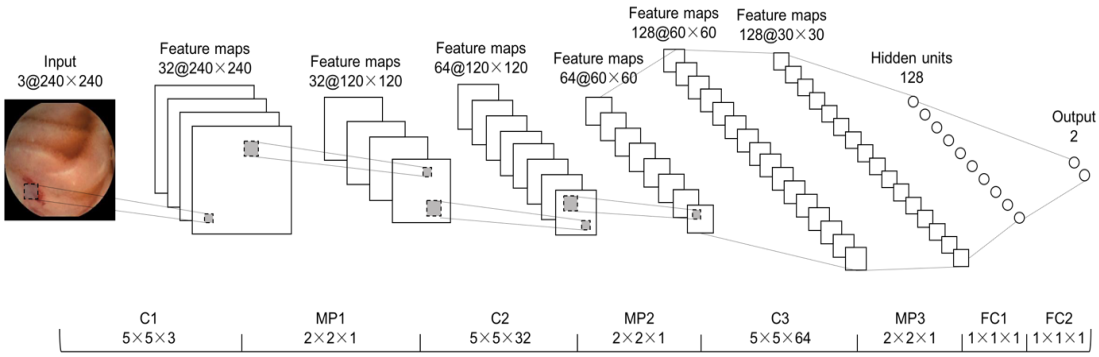



Fig. 23. An illustration of the proposed CNN architecture [44].

## 3.2 Bleeding Detection System

### 3.2.1 Conventional Approaches

The study in [45] extracted color and texture features. These features are used to generate bag of words using K-means clustering algorithm. Next, the Expectation Maximization (EM) is employed on the "Bag-of-Visual-Words" for super-pixel

segmentation. From the region of interest, geometric features like centroid, area, and eccentricity are extracted and fed to the SVM classifier [30].

The authors in [46] proposed an approach based on statistical color feature analysis. [Fig. 24] shows the main steps of the proposed detection technique. First, the frame is split into blocks. After that, dark or light blocks are excluded. Moreover, canny operator [47] is applied to discard the edges. Furthermore, Wavelet db2 with soft thresholding [48] is applied to reduce noise. The Red channel of the RGB color space is exploited to detect bleeding regions. More specifically, red ratio is computed for individual pixels. Finally, Support Vector Machine (SVM) is used to classify WCE frames into bleeding and non-bleeding classes.

Alternatively, the system described in [49] performs semantic segmentation by classifying the pixels as a “Bleeding” or “No Bleeding” pixel. This results in detecting the bleeding pixel within the frame. More specifically, the proposed system in [45] extracts the Red-Green-Blue (RGB) color feature [50] and the Gray-Level Co-occurrence Matrix (GLCM) texture feature [50]. These two features are combined and fed to Random Tree (RT) [51], Random Forest (RF) [52], and Logistic Model Tree (LMT) [53] classifiers.

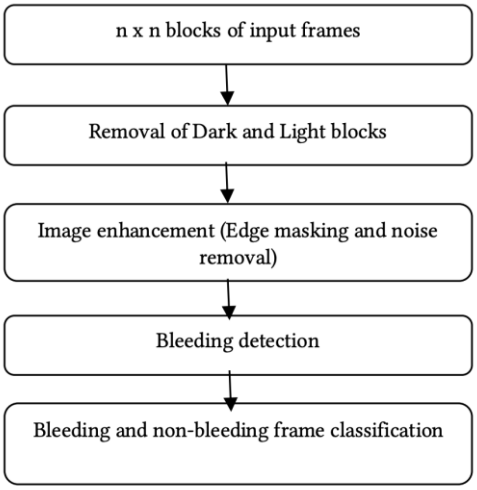



Fig. 24. Image processing steps of the proposed methodology [46].

### 3.2.2 Deep Learning Approaches

The authors in [54] use AlexNet [23] CNN model to classify the frames as "Bleeding", or "No Bleeding". This is a well-known CNN model, which is one of the earliest models that won the ILSVRC run by ImageNet. AlexNet model architecture is shown in [Fig. 25]. Once the bleeding frames are separated, they are segmented using SegNet [55] in order to detect the "Bleeding" areas. As shown in [Fig. 26], it is a deep learning model designed for image segmentation. It is constituted of convolutional stacked auto-encoder.

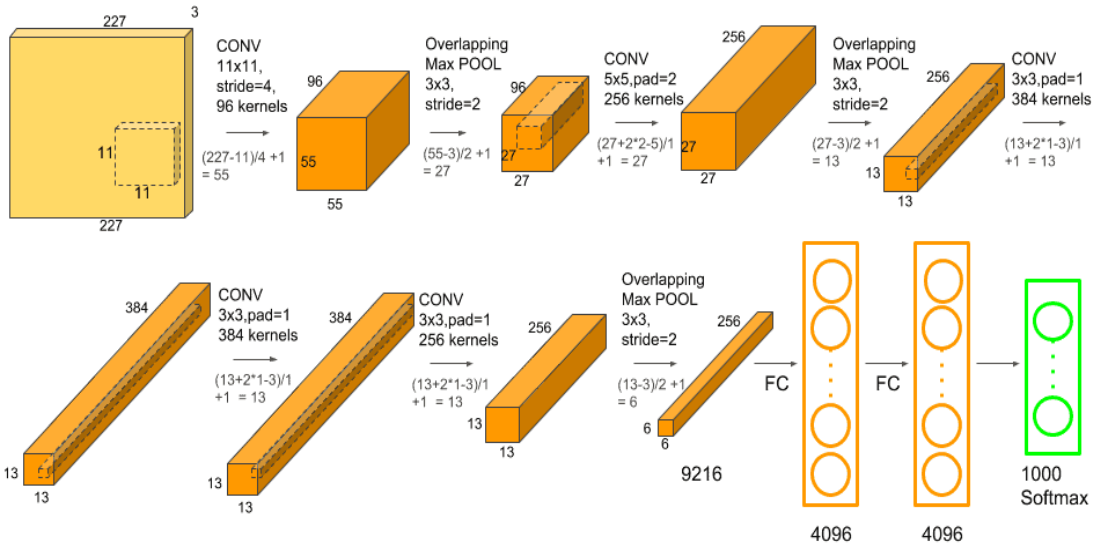



Fig. 25. AlexNet model architecture [56].

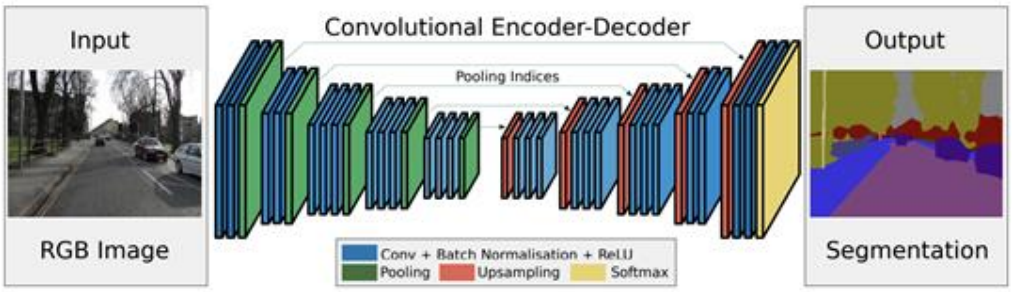



Fig. 26. SegNet model architecture [55].

Similarly, the authors in [57] use U-Net [57] deep learning segmentation approach to detect "Bleeding" regions in the small intestines. As shown in [Fig. 27], the model architecture has a "U" shape. The model down-samples the input image to a small feature map. Next, it up-samples it. The up-sampling process use skip connections to benefit from the down-sampling process. In fact, at each level, the down-sampled feature map is concatenated to the up-sampled one to generate the next up-sampled feature map.

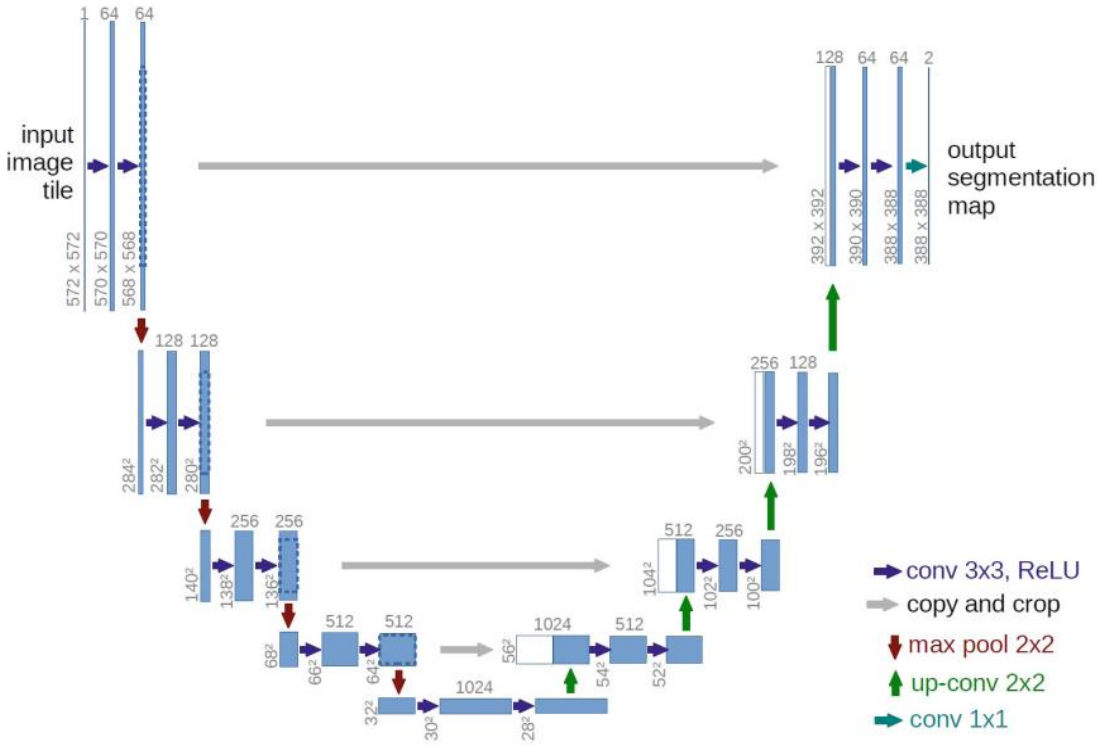



Fig. 27. U-Net model architecture [57].

The work in [58] employs a Cascade Proposal network to generate region of interest proposals as shown in [Fig. 28]. These are regions susceptible to include bleeding pattern. The proposed regions are then fed to the Region Proposal Rejection (RPR). The latter is a small network consisting of one convolutional layer, one fully connected layer, and two output layers. It is used to rank the regions based on a score. Its output is fed to a detection module which predicts the bounding box and the corresponding class. For the testing phase, the unseen image is provided to both a Salient Region Segmentation (SRS) and a Multiregional Region Combination (MRC). While SRC captures the exact location of the regions [58], and Multiregional Region Combination (MRC) that gains adequate coverage

of the concerned region and apply the SRS to locate region of interest's positions. Moreover, object boundaries are refined using the Dense Region Fusion (DRF) approach by checking the density of a specific area [58].

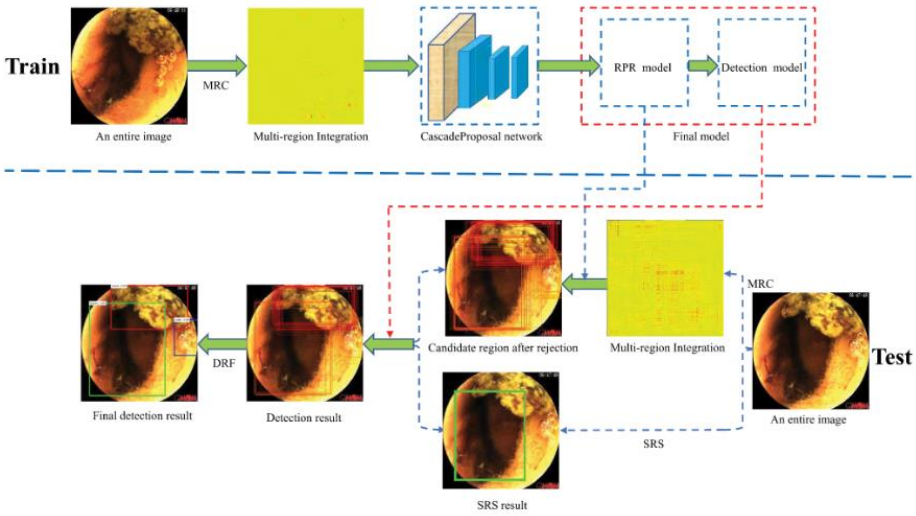



Fig. 28. A summary of the proposed system [58].

## 3.3 Discussion

The related works are summarized in Table 1. As it can be seen, the related works in [7], [31], [35], [37], [42], and [44] classify the frames into “Bleeding” and “No Bleeding”. While the earliest studies in [7] and [31] are based of extracting “hand crafted” features that are fed to a classifier, the works in [35], [37], [42], and [44] exploit deep learning models befitting therefore from the automatic learning of the features. In fact, using deep learning paradigm alleviates the problem of selecting the suitable features which is usually performed through empirical comparison of the features. Nevertheless, classification approaches don not localize the bleeding within the frame. Alternatively, the works in [45], [46], [49], [54], [57], and [58] perform bleeding detection. In particular, the studies in [45], [46], and [49] utilize “hand crafted” features. While the work in [45] and [46] splits the frame into blocks to transform the problem into a set of local problems and identifies in which block the bleeding occurs, the work in [49] perform semantic segmentation through pixelwise classification. The deep learning detection-based approaches in [54] and [57] are segmentation approaches. In fact, they exploit well known

deep learning segmentation approaches SegNet and U-Net. Nevertheless, these two approaches are known to be very slow and not suitable for real world applications [59]. On the other hand, the work in [58] is not employing segmentation. It learns a bounding box to localize the anomaly. Specifically, it is based on a customized CNN. Thus, the adopted model could be fit the considered datasets. Moreover, it includes several modules, namely, SRS, MRC, RPR, and detection modules. This is advantageous when compared to end-to-end model. In fact, the error inducted by one of these modules affects all other modules. Moreover, the error of the different modules gets accumulated.

## 3.4 Summary

We conclude that most of the state of the arts bleeding detection-based approaches rely on segmentation techniques. As such, for conventional approaches, “hand crafted” features for the segmentation task and for the classification task are required. On the other hand, the reported deep learning segmentation approaches are not suitable for real world application, since they suffer from high space and time complexity. The only approach that exploits bounding boxes is a customized one that can be overfitted to the considered data. Thus, none of the related works exploited deep learning approaches for object detection. In particular, YOLO family of object detection has not been investigated.

**Table 1. Related Works Summary**

| Classification/ Detection | Reference | Visual Features | Segmentation Technique | Classifier | Deep learning model | Dataset | Size | Performance |
|---|---|---|---|---|---|---|---|---|
| **Classification** | [7] | Color Moment | NA | SVM | NA | Imaging $PillCamR\ SB$ [60] | Nº. MBS= 230 Non-MBS=461 | AUC = 81.2% |
| **Classification** | [31] | Color Moment | NA | FCM and KNN | NA | Imaging PillCam [5] | Nº. MBS= 360 Non-MBS=915 | Accuracy= 90.92% |
| **Classification** | [35] | Automatically learned by LeNet-5 convolution layers | NA | Softmax | Lenet-5 | Dataset provided in [61] | Nº. training set= 9 Validation and test set=35 | Specificity = 87.3%, Sensitivity= 82.71%, Average= 89.07% |
| **Classification** | [37] | Fusion of automatically learned features by VGG19, ResNet50, and InceptionV3 | NA | SVM | VGG19 ResNet50, and InceptionV3 | 1. Kid dataset [62] 2. MICCAI2017 [57] | Nº. MBS= 1873 Non-MBS=4374 | Accuracy= 97.71% for KID dataset Accuracy= 95.08% for MICCAI 2017. |
| **Classification** | [42] | Automatically learned from MobileNet and the customized CNN | NA | Softmax | MobileNet and Customized CNN | Google collected $PillCam^{TM}SB3$ Dataset [63] | Nº. MBS= 25 Non-MBS=22 | Accuracy= 97.8% Precision= 100%, Recall=99.4%, F1 score=99.7%, and Cohen's kappa=99.5% |
| **Classification** | [44] | Automatically learned by a customized CNN | NA | SVM | Customized CNN | Dataset collected form WCE video [62], [64], [65] | Nº. MBS= 2850 Non-MBS=7150 | $F_1$ score =99.55% |

**Table 1. Related Works Summary**

| Classification/ Detection | Reference | Visual Features | Segmentation Technique | Classifier | Deep learning model | Dataset | Size | Performance |
|---|---|---|---|---|---|---|---|---|
| **Detection** | [45] | Bag of visual words based on color and texture features | Expectation Maximization (EM) | SVM | NA | NA | NA | NA |
| **Detection** | [46] | Statistical color feature using the red ratio | Split into blocks | SVM | NA | UMMC dataset [66] | Nº. MBS= 800 Non-MBS=400 | Accuracy= 97.67%, Sensitivity= 97.57% Specificity= 95.46% |

**Table 1. Related Works Summary**

| Classification/ Detection | Reference | Visual Features | Segmentation Technique | Classifier | Deep learning model | Dataset | Size | Performance |
|---|---|---|---|---|---|---|---|---|
| **Detection** | [49] | RGB color feature and GLCM texture feature | Pixel classification | Random Tree (RT), Random Forest (RF), and Logistic Model Tree (LMT) | NA | UMMC dataset [66] | Nº. MBS= 140 Non-MBS=157 | RT Accuracy= 96.2% RF Accuracy= 97.6% LMT Accuracy= 97.3% |
| **Detection** | [54] | Automatically learned by AlexNet | SegNet | SegNet | SegNet | KID dataset [62] Clinical dataset [67] | Nº. MBS= 450 Non-MBS=1900 | Accuracy = 94.42% F1 score= 98.49% and 88.39% |

**Table 1. Related Works Summary**

| Classification/ Detection | Reference | Visual Features | Segmentation Technique | Classifier | Deep learning model | Dataset | Size | Performance |
|---|---|---|---|---|---|---|---|---|
| **Detection** | [57] | NA | U-Net | U-Net | U-Net | Two datasets collected form WCE video [64] | Nº. lesion= 1570 Non-lesion=2325 | Accuracy= 93.56% Sensitivity= 99.56% Specificity= 93.93% |
| **Detection** | [58] | NA | SRS and MRC for the regions of interest | Customized CNN, RPR model, and Detection models to re-rank and classify the regions | Customized CNN | Imaging Incorporation and Chongqing Jinshan Science and Technology | Total Nº. images = 7200 | Final mean average precision (mAP) of 70.3% and improved (mAP)= 72.3% |

# Chapter 4: Proposed Approach

Most of the pathogens that affect the gastrointestinal system have similar symptoms. In particular, cancer, benign tumors, ulcers, and hemorrhoids can cause Multiple Bleeding Spots (MBS) in the gastrointestinal tract. Therefore, detecting MBS can help significantly in early diagnosis of such pathogens and prevent risks. WCE is one of the recent technologies that helps the physicians diagnosing anomalies in the Gastro-Intestinal (GI) tract. Nevertheless, although WCE records all the details of the GI tract, the physician is required to visualize an 8-hour long video. This is a burdensome task. Moreover, due to the small size of the lesion region, the disease diagnosis may be missed at an early stage.

## 4.1 Motivation

Computer aided-diagnosis can lessen the visualization task and help detecting automatically the MBS. As shown in the related works investigation, MBS aided diagnosis systems are based on image processing and machine learning techniques. In particular, most of the reported works related to detecting MBS employ segmentation techniques. As a result, “hand crafted” features for the segmentation task and for the classification task are required. This can be alleviated by the use of deep learning approaches designed for object detection. Nonetheless, to the best of our knowledge, deep learning models have not been explored for MBS detection. In particular, the end-to-end state of the art YOLO models were not investigated.

YOLO deep learning detection model outperformed the other object detection approaches in many pattern recognition applications [68], [69]. Moreover, the success of YOLO model and its applicability to real world applications, yield the evolution of the model and the publication of different versions. However, a throughout comparisons of these versions in terms of performance and efficiency needs to be performed. In this regard, YOLO model, specifically, its latest versions YOLOv3 [9], YOLOv4 [10], YOLOv5 [11], and YOLOv7 [12] are investigated for detecting MBS in the GI tract. In the following, we describe the four considered models.

### 4.1.1 YOLOv3 Architecture

YOLO version 3 (YOLOv3) [9] is an improved version of YOLO which seeks to enhance the performance through the use of residual blocks and different scale feature maps. Inspired by Residual Networks [70] YOLOv3 employs alternatively 3×3 and 1×1 convolutional layers to form a residual unit. [Fig. 29] shows the structure of the residual unit which involves the two convolution layers. This unit aims at avoiding the vanishing gradient problem faced by very deep network.

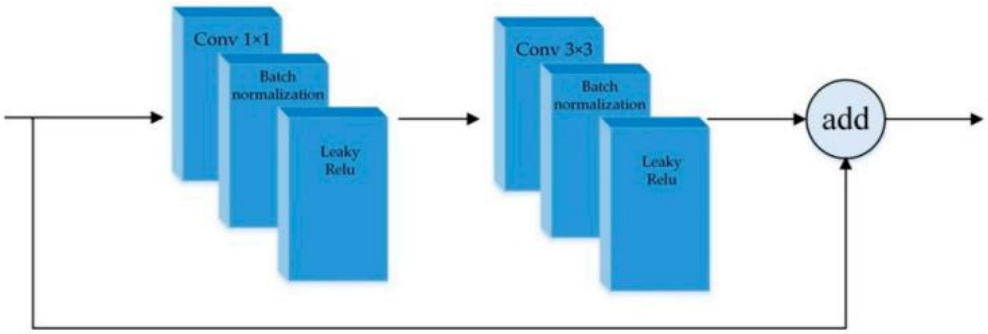



Fig. 29. Residual unit structure[70].

As shown in [Fig. 30], YOLOv3 is composed of five residual block which incorporate a number of residual units. Since a stride of 2 is used at each residual block, the input is down-sampled five times. In particular, the last three down-sampled feature maps are used for the prediction task. Specifically, after the third residual block, the feature map is down-sampled by factor 8. It is exploited for small object prediction. On the other hand, the output of the fourth residual block is down-sampled by a factor of 16, and it is utilized to generate scale 2 feature map. The latter is employed for medium object prediction. Alternatively, big objects, referred to as scale 1 objects, are predicted using the last residual block for which the feature is down-sampled by a factor of 32.

Furthermore, YOLOv3 performs feature fusion to benefit from the feature maps at the different scales. As such, it up-samples scale 1 feature map and concatenate it with scale 2 feature map. The obtained feature map is then up-sampled, and concatenate with scale 3 feature map [70].

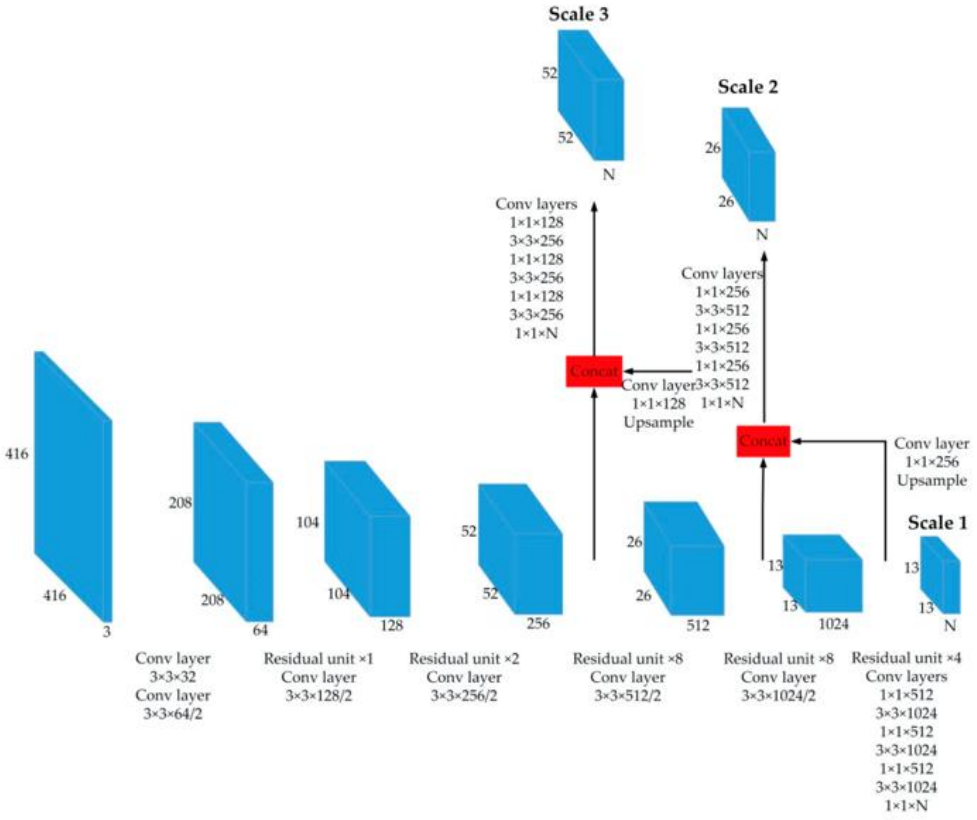



Fig. 30. YOLOv3 network [70].

### 4.1.2 YOLOv4 Architecture

YOLOv4 [71] is the fourth version of the YOLO model family. YOLOv4 model architecture is composed of multiple sections as shown in [Fig. 31]. Namely, they are the Input, the Backbone, the Neck, and the Head (dense prediction, and the sparse prediction). The backbone and the neck sections are responsible for feature extraction and aggregation, respectively. In particular, the CNN deep learning model, CSPDarkNet53 [10], is used as a feature extractor in the backbone section. Alternatively, Spatial Pyramid Pooling (SPP) and Path Aggregation Network (PANet) were utilized in the neck section to fuse the features using Bag of Specials (BoS). Finally, the head which is responsible for both localizing the object in the image and classifying it, amounts to YOLOv3 models. It consists of two stage detectors. The first one is the one stage object detector and the second one is the two-stage object detector [10].

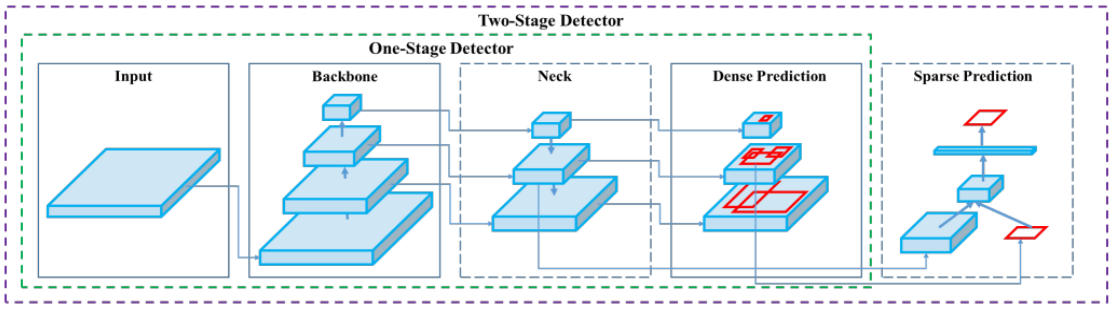



Fig. 31. YOLOv4 object detector parts [10].

Compared to the previous versions of YOLO, YOLOv4 mainly introduced 2 additional concepts. Bag of Freebies (BoF) and Bag of Specials (BoS). Bag of freebies are a set of techniques that alters the training framework or perform data augmentation. As shown in [Fig. 32], many techniques can be incorporated for the purpose of enhancing the model performance without affecting on the inference cost [10].

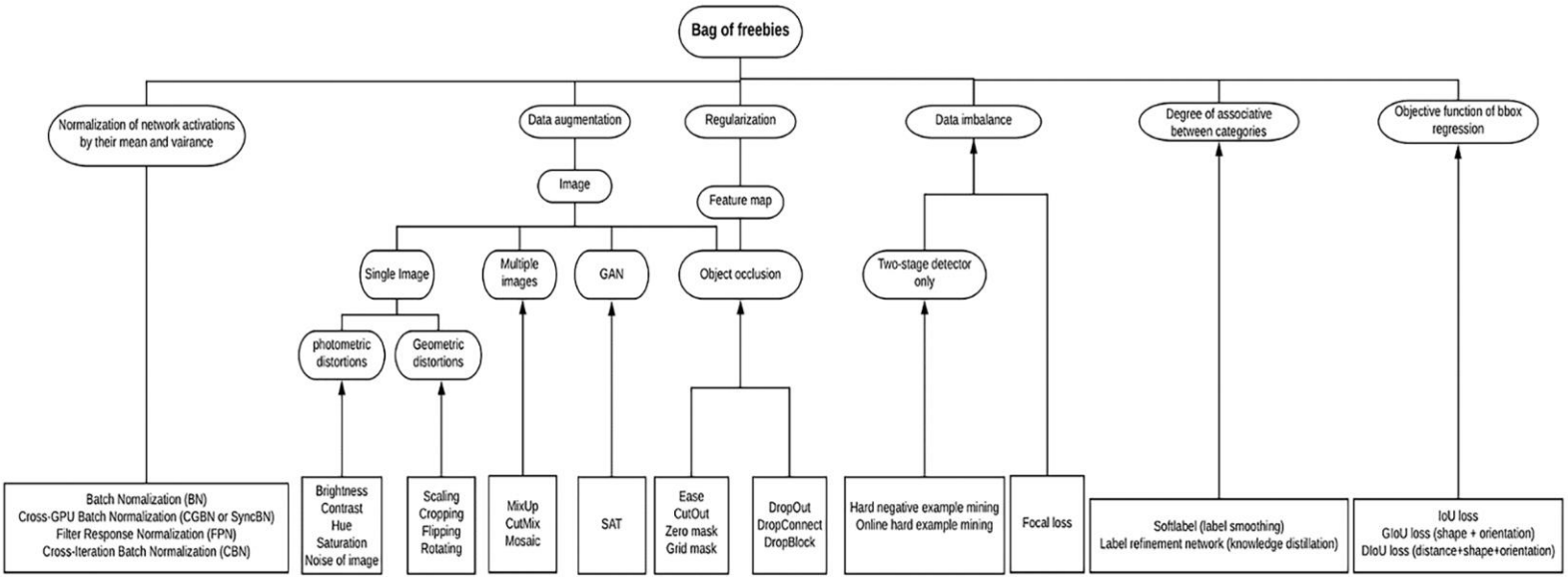



Fig. 32. Bag of Freebies techniques [72].

Alternatively, BoS are strategies such as enlarging the receptive field, integrating features, incorporating attention modules, or post-processing. These strategies aim at significantly enhancing the performance of accuracy at the expense of increasing the inference cost [10].

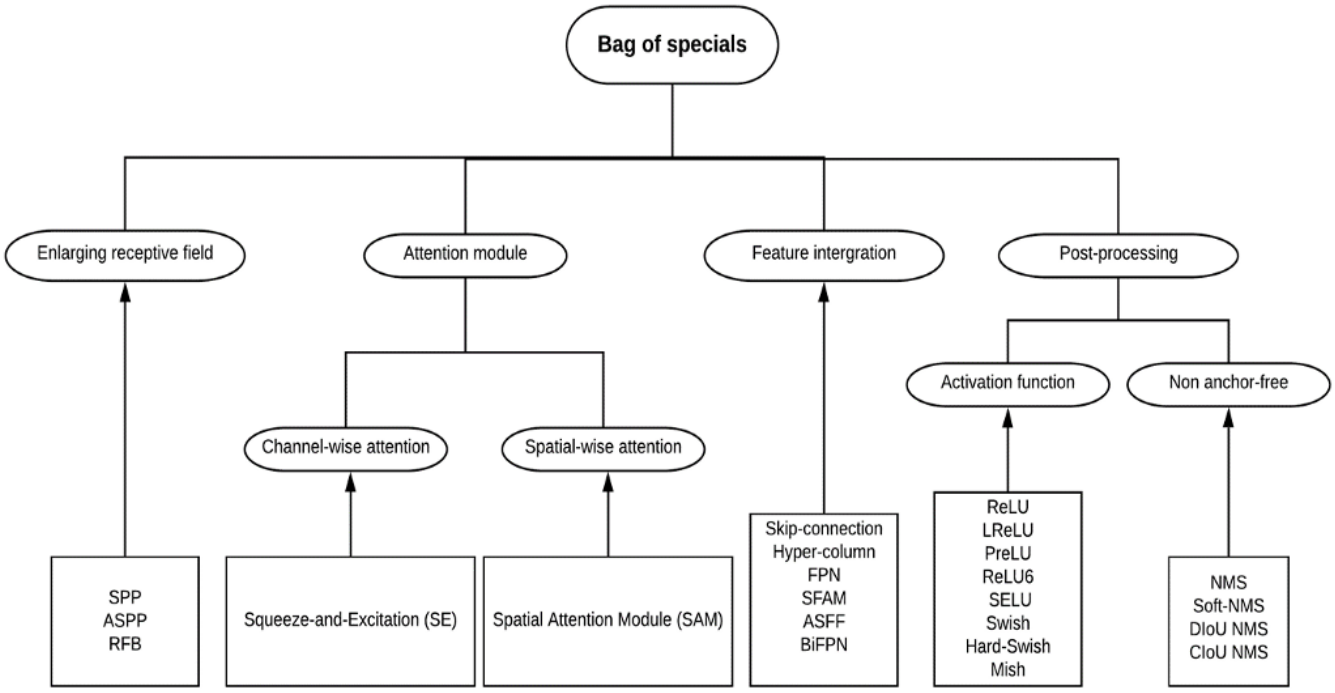



Fig. 33. Bag of Specials techniques [72].

### 4.1.3 YOLOv5 Architecture

YOLOv5 [11] is implemented using PyTorch which allows faster training [73]. As such, YOLOv5 allows rapid detection with the same accuracy as YOLOv4. Specifically, YOLOv5 has been proved to have higher performance than YOLOv4 under certain circumstances and partly gained confidence in the computer vision community besides YOLOv4. As shown in [Fig. 34], YOLOv5 model architecture is similar to YOLOv4 architecture. It employs CSPDarknet53 [73] for the backbone section as feature extractor. The latter aims at addressing the gradient in deep networks and decreases the inference time through the use of cross-layer connections between the network's front and back layers. Moreover, it seeks improving the accuracy and utilizing lightweight model. Furthermore, the SPP module referring to the Spatial Pyramid Pooling module, performs maximum pooling with several kernel sizes and then fuses the features by concatenating them together. Additionally, YOLOv5 exploits Path Aggregation Network (PANet) in the neck section as feature aggregator to increase the flow of information and to enhance the object localization. Besides, PANet incorporates a Feature Pyramid Network (FPN) [74].

On the other hand, the head is designed in the same way as YOLOv3 and YOLOv4. Specifically, it produces three different scale feature maps. As shown in [Fig. 34], The CSP network in the backbone is made up from one or more residual units, whereas the CSP network in the neck is made up of new module called CBL modules that replace the residual units. The CBL module consist of Convolution layers, Batch normalization layers, and Leaky ReLU activation function modules [75]. YOLOv5 introduces a new layer referred to as Focus layer [76] as shown in [Fig. 34]. It takes the place of the first three layers of YOLOv3. Therefore, it reduces the GPU requirement and decreases the number of layers.

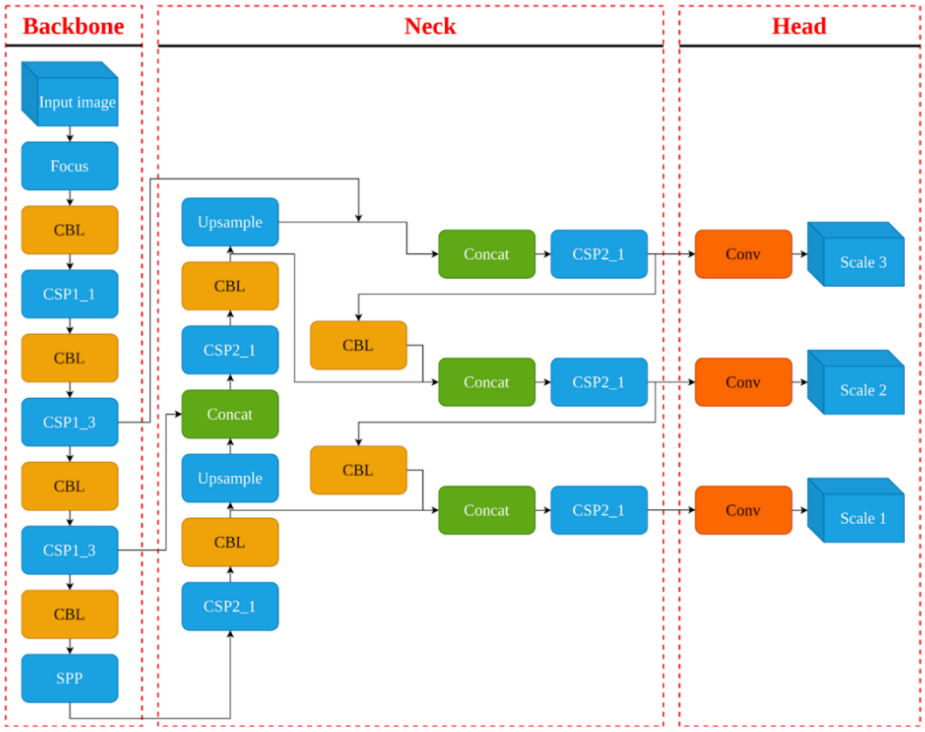



Fig. 34. YOLOv5 architecture [74].

### 4.1.4 YOLOv7 Architecture

The most recent YOLO architecture, YOLOv7 [77], is based on YOLOv4 version. The main modifications consist of (i) the introduction of the Extended Efficient Layer Aggregation Network (E-ELAN), (ii) the incorporation of model scaling component, (iii) the use of planned re-parameterized convolution, (iv) the employment of auxiliary head, and (v) the exploration of label assigner mechanism.

E-ELAN is a computational component in YOLOv7 backbone part. It enhances the prediction performance continuously by employing “expand, shuffle, merge cardinality”. Alternatively, the model scaling optimizes the number of layers, the number of channels, the number of stages in the feature pyramid, and the resolution of the input image in order to meet the requirements of various problems. Nevertheless, YOLOv7 introduces a new model scaling paradigm which optimizes the scaling factors jointly, not independently one from the other. Similarly, YOLOv7 modifies RepConv by discarding the identity connection. In fact, it uses RepConvN in order to prevent the presence of identity connection for re-parametrized convolution. Moreover, YOLOv7 exploits the Deep Supervision training technique. More specifically, YOLOv7 uses an auxiliary head in the intermediate layers to guide the training. The head responsible for the final prediction is referred to as lead head. Additionally, to further enhance the training, YOLOv7 outputs soft labels instead of hard one referring to the ground truth.

## 4.2 Proposed System

We propose to compare the performance between different YOLO approaches which are YOLOv3 [9], YOLOv4 [10], YOLOv5 [11], and YOLOv7 [12] in recognizing “Bleeding” spots. For this purpose, the considered models need to be trained. Therefore, each YOLO model is fed with images indicating the bleeding areas, if any, as seen in [Fig. 35]. Specifically, the coordinates of the bounding boxes surrounding the MBS patterns are provided as input along with the “Bleeding” images. They consist of the upper left corner coordinates (X, Y), the width, and the height of each box. Concerning the “Non-Bleeding” images, no boundary box is specified.

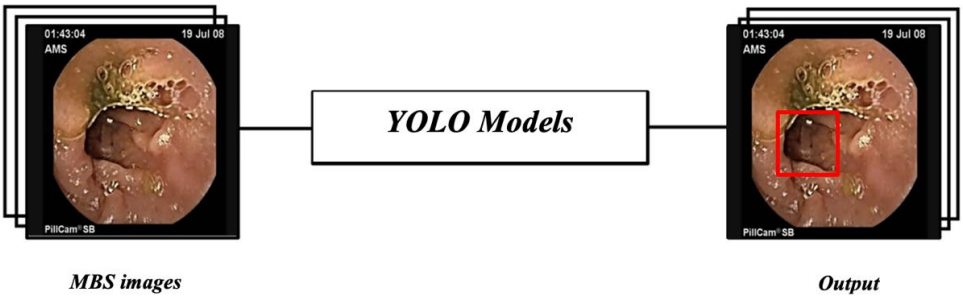



Fig. 35. YOLO models training framework.

To determine the best version of YOLO, the methodology shown in [Fig. 36] is adopted to select the best version. In particular, the considered YOLO models are evaluated using the test set. Specifically, the different models are tested in terms of the inference time, MBS localization and classification. The best performing model is adopted to build the required system as shown in [Fig. 37].

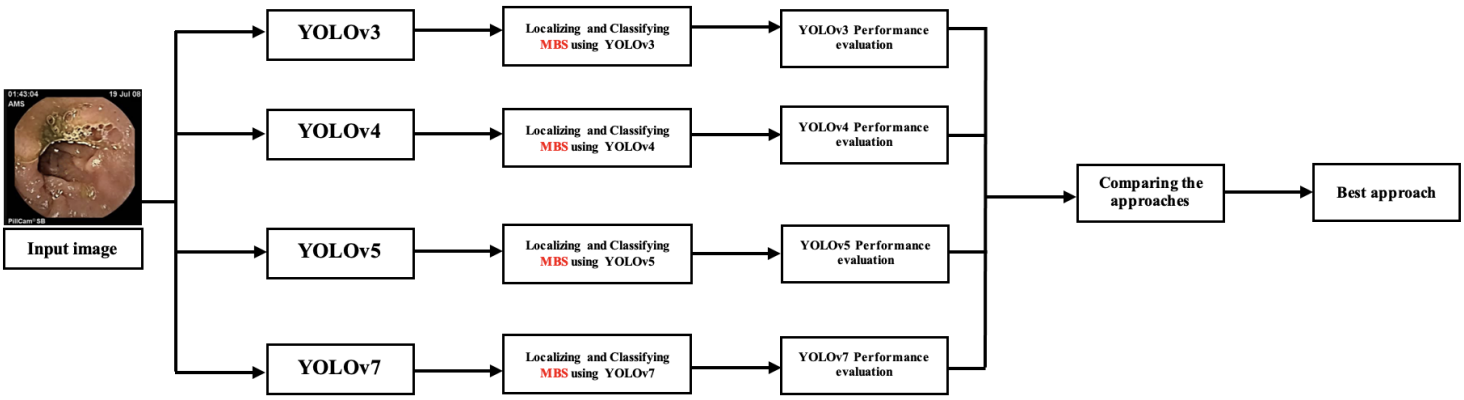



Fig. 36. Methodology to design YOLO based recognition system for MBS.

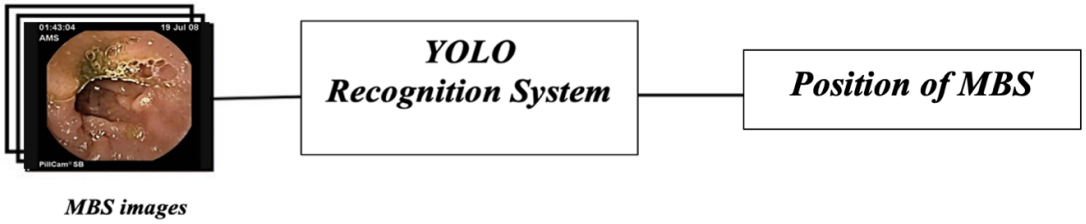



Fig. 37. Proposed system architecture.

## 4.3 Summary

The proposed approach is presented in this chapter. More specifically, the motivation for employing the YOLO deep learning recognition model is provided. Moreover, the four considered YOLO models are introduced. Furthermore, the methodology that is adopted of selecting the most suitable YOLO version is outlined.

# Chapter 5: Experimental Setting

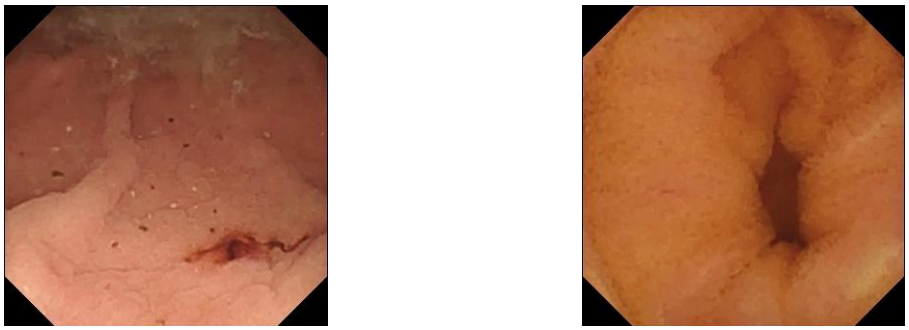
## 5.1 Dataset Description

Kvasir-Capsule dataset [78] is considered in this project. It is a dataset of WCE videos collected from clinical examinations performed at the Department of Medicine, Bærum Hospital, and Vestre Viken Hospital Trust in Norway. The tests were carried out using the Olympus Endocapsule 10 System45, which includes the Olympus EC-S10 endocapsule [Fig. 38, a] and the Olympus RE-10 endocapsule recorder [Fig. 38, b].


(a) (b)

Fig. 38. VCE equipment used for data collection (a) Olympus EC-S10 endocapsule, (b) Olympus RE-10 endocapsule recorder [78].

The considered dataset consists of 406 “Bleeding” images representing bleeding spots of different size, color, and texture. In addition, it includes 34338 “Non-Bleeding” images representing normal GI tract frames (without bleeding). [Fig. 39, a] shows a sample “Bleeding” image from the dataset; while [Fig. 39, b] shows a “Non-Bleeding” image.



(a) (b)

Fig. 39. Images of GI tract (a) affected by MBS (b) normal image [78] .

According to [79], it is not recommended to add images without region of interest ("non-bleeding" images) to the training set. More specifically, "non-bleeding" images should not exceed more than 10% of the total number of images in the training set. As such, only 328 non-bleeding images are first considered. This results in the distribution reported in Table 2 where the images are divided into 60% for training, 20% for validation, and 20% testing sets. Nevertheless, in order to get a glimpse of the models' performance on the real-world, 6000 non-bleeding images are used in the test set. More specifically, both test sets which are the test set after omitting most of non-bleeding images (Test 1) and the test set that containing 6000 background images (Test 2) are assessed.

**Table 2. Dataset Instance Distribution**

| | **Training set** | **Validation set** | **Testing set without additional non-bleeding (Test 1)** | **Testing set with additional non- bleeding (Test 2)** |
|---|---|---|---|---|
| **Bleeding** | 231 | 75 | 100 | 100 |
| **Non-bleeding** | 328 | 108 | 86 | 6000 |

The available Ground Truth consists of labeling the whole image as including bleeding or not. Nevertheless, in order to train YOLO, a different ground truth should be provided. In fact, the coordinates of the bounding boxes surrounding the bleeding spots should be fed to model to be trained. As such, the dataset is labeled using LabelImg labeling software tool [80]. As a result, 960 bleeding regions are considered.

## 5.2 Performance Measures

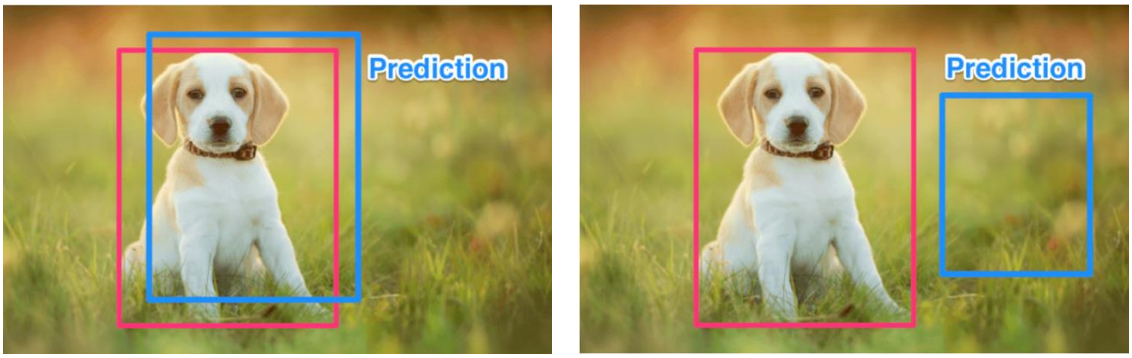
Two performance measures are considered to evaluate the performance of YOLOv3 [9], YOLOv4 [10], YOLOv5 [11], and YOLOv7 [12] in terms of recognizing MBS. Specifically, we considered Intersection over Union (IoU) [81] and mean Average Precision (mAP) [82], since the localization and the categorization of the object of interest are assessed using these performance measures. Moreover, Floating Point Operations per second (FLOP) [83] is also considered to compare the time efficiency of the considered YOLO models. In the following, detailed descriptions of these measures are provided.

### 5.2.1 Intersection over Union (IoU)

The Intersection over Union (IoU) [81] assesses the localization of the object of interest. Specifically, it compares the bounding box provided as ground truth and the one predicted by the model. It is defined as the area of intersection of two bounding boxes (prediction and ground truth) over the area of their union. IoU is defined as in (1).

$$IoU = \frac{Area\ of\ intersection}{Area\ of\ union} \tag{1}$$

The value of IoU is between zero and one indicating how much a prediction bounding box and the ground truth bounding box overlap. It is equal to one if the prediction is completely correct (the two bounding boxes are completely overlapping). The closer the value is to one the better the prediction result. Alternatively, the value will be equal to zero if the two bounding boxes do not overlap. [Fig. 40] shows sample recognition results for the object dog. Since both the ground truth box (red box) and the predicted one (blue box) are overlapping in [Fig. 40, a], the corresponding IoU is close to 1. Alternatively, since there is no overlapping between the two bounding boxes in [Fig. 40, b], the IoU is equal to zero.




(a) (b)

Fig. 40. Sample recognition results for the object dog. (a) Good localization of the dog, (b) Bad localization of the dog [81].

### 5.2.2 Average Precision (AP)

IoU and the confidence score are two parameters that must be considered in the context of object detection. In particular, IoU is used to compute the True Positives (TP), False Positives (FP) and False Negatives (FN). Expressly, the prediction is considered TP with respect to class *i* on condition that the IoU is greater than a specific threshold which is 0.5. Whereas if the result of the IoU is less than 0.5, the prediction is considered as FP with respect to class *i* [84]. As such, the precision [84] and the recall [84] metrics, with respect to class *i*, are evaluated using (2), and (3), respectively.

$$Precicion_i = \frac{TP_i}{TP_i + FP_i} \tag{2}$$

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \tag{3}$$

where $TP_i$ is the total number of True Positives with respect to class $i$, $FP_i$ is the total number of False Positives with respect to class $i$, and $FN_i$ is the total number of False Negatives with respect to class $i$ [84].

Average Precision (AP) is evaluated by taking in consideration 11 confidence score threshold values. Specifically, the confidence score threshold is tuned, and for each threshold, the resulting recall and precision are recorded. Then, the recall values: 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, and 1 and their corresponding precisions values are

retained. The AP of class $i$ is defined as the Area Under the Curve (AUC) of the obtained precisions versus the 11 recall values [85]. It can be expressed as in (4).

$$AP = \int_0^1 p(r)dr \tag{4}$$

where p denotes the precision value and r to the recall value.

### 5.2.3 Mean Average Precision

The mean Average Precision (mAP) is the average of the APs of all considered classes [82], as defined in (5).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{5}$$

where $N$ denotes total number of classes and $AP_i$ denotes the AP of class $i$.

### 5.2.4 Floating Point Operations per second (FLOP)

The time required to detect an object of interest is one of the demanding challenges in the field of real-time computer processing. In this regard, we consider measuring the time required of processing an image using Floating Point Operations per second (FLOPs) [83]. It is defined as the number of floating-point calculations needed for a forward run through a single neural network per second. This procedure is computed during the test phase.

## 5.3 Experiment Description

We assessed the performance of the four considered YOLO models in detecting MBS from WCE video. Specifically, we evaluated YOLOv3 [9], YOLOv4 [10], YOLOv5 [11], and YOLOv7 [12] for MBS recognition using the performance measures defined in section 5.2. We trained and evaluated these models using the WCE video images dataset from Kvasir-Capsule [78] described in section 5.1.

### 5.3.1 Experiment 1

In experiment 1, YOLOv3 [9] is trained on the 559 images from Kvasir-Capsule dataset [78]. The model's hyper-parameters are tuned using the validation data. Then, the trained model is tested using the test data using IoU, mAP, and FLOPs.

### 5.3.2 Experiment 2

Similarly, to experiment 1, YOLOv4 [10] is trained on the 559 images from Kvasir-Capsule [78] dataset and the corresponding ground truth. The hyperparameters are tuned using validation data. In the same way as in experiment 1, the trained YOLOv4 [10] model is tested and evaluated using the same performance measures.

### 5.3.3 Experiment 3

Similar to the experiment of YOLOv3 and the experiment of YOLOv4, 559 images from Kvasir-Capsule [78] dataset and the corresponding ground truth is used to train YOLOv5 [11]. After tuning the hyperparameters using the validation data, YOLOv5 [11] model is evaluated , and its performance is assessed using the same performance measures as for YOLOv3 [9] and YOLOv4 [10].

### 5.3.4 Experiment 4

Similarly, to the previous experiments, the 559 images from Kvasir-Capsule [78] dataset and the corresponding ground truth is used for training YOLOv7 [12] in experiment 4. The model's hyperparameters are set using validation data. Furthermore, the obtained model is tested using the test dataset (6100 images). The same performance measure, described in section 5.2, is adopted to assess the performance of YOLOv7.

After the four experiments have been conducted, their results are compared and analyzed. The final design of the proposed approach is determined based on these findings and hence determining its performance.

## 5.4 Summary

In this chapter, we presented the framework of the conducted. The dataset and the ground truth were first described. Then, the performance measures that were utilized in evaluating and analyzing the performance of YOLO models were outlined. Finally, the conducted experiments were described.

# Chapter 6: Implementation

In this chapter, we present the implementation environment. Specifically, we describe the tools used for data preparation, the framework, and the programming platform. Furthermore, details about the challenges and the issues faced during the project implementation are highlighted.

## 6.1. Implementation Environment

To implement the proposed system, Google Collaboratory is used. Google Collaboratory is a programming platform that facilitates the development and execution of python codes using a browser, and provides a useful experience for deep learning, and data analysis. Moreover, Roboflow [86] is used to split the dataset into training, validation and testing sets, it is also used to apply data augmentation to the training set.

In the conducted experiments, YOLO pretrained models are adopted. They are built on the PyTorch framework. The latter is an open-source machine learning library employed for training and evaluating deep learning models.

The following libraries are applied in order to implement the models.

- NumPy:

Large multidimensional arrays are now supported in the python programming language with NumPy which is a highly optimized and free open-source library. In addition to these arrays, NumPy also provides a collection of high-level mathematical functions. They include basic linear algebra, random simulation, Fourier analyses, trigonometric operations, and statistical operations [87].

- OpenCV:

OpenCV is an open-source computer vision and machine learning library. In order to facilitate the usage and speed up of machine perception, OpenCV was developed as a standard architecture for computer vision applications. The library contains more than 2500 efficient and optimized algorithms. These algorithms can be applied to detect and recognize objects [88].

- Pillow:

All of the fundamental image processing features are available in the Pillow library. You can modify, rotate, and resize images. With the help of the Pillow module, you may extract some statistics from an image using the histogram method, which you can then utilize for statistical analysis [89].

- PyYAML:

YAML is a format specified for data serialization, it is designed for interactions between human readability and scripting languages. PyYAML is considered a tool that parses and emits YAML format for Python codes. PyYAML can be used for a wide variety of tasks, including object serialization and persistence as well as sophisticated configuration files [90].

- Thop:

A tool used as a script to count the number of FLOPs [91].

## 6.2. Implementation Issues

In order to implement Deep Learning algorithms and train them in reasonable time, we needed a computationally powerful devices that can train the models effectively. Unfortunately, the personal computers did not have high specifications for training deep learning models. Consequently, to solve this problem, Google Collaboratory is used in order to provide additional services and computing resources including GPUs. Along with the hardware issues, we faced some software related issues, such as fixing the execution errors caused by bugs that appeared when running the codes that were taken from the official repository from GitHub. This obstructed the execution process and consumed time needed for training.

# Chapter 7: Results and Discussion

The aim of this chapter is to report and analyze the results of the four conducted experiments.

## 7.1. Experiment 1

In this experiment, we assessed the performance of YOLOv3 [9] to detect multiple bleeding spots. As such, YOLOv3 model was trained using different values of the learning rate, momentum, and the number of batches, starting with the initial model's values then tuning values based on the results. The learning rate is the most crucial hyper-parameter. In fact, a too small value may result in a long training process, whereas a too large value may result in overshooting the local minimum. Table 3 shows the four considered configurations.

**Table 3. YOLOv3 [9] Hyper-parameter Configuration**

| | Learning Rate | Momentum | Number of Batches |
|---|---|---|---|
| **Configuration 1** | **0.0001** | **0.99998** | **16** |
| **Configuration 2** | **0.0001** | **0.99998** | **8** |
| **Configuration 3** | **0.001** | **0.9998** | **8** |
| **Configuration 4** | **0.01** | **0.998** | **8** |

### 7.1.1. Results

In order to determine the best configuration, the results of the performance measures on the validation set are displayed in [Fig. 41]. These are the IoU and the mAP.

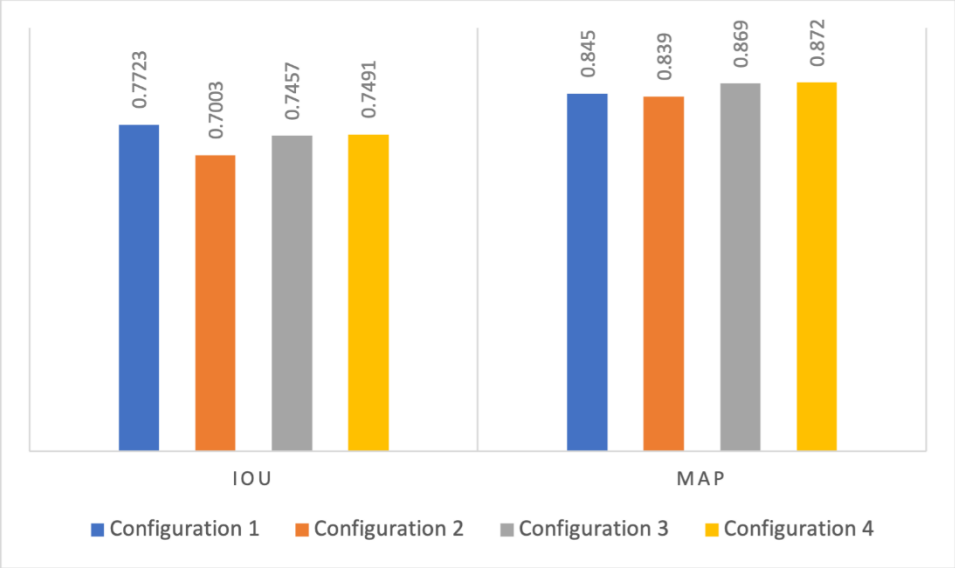



Fig. 41. YOLOv3 [9] performance results on the validation set.

### 7.1.2. Discussion

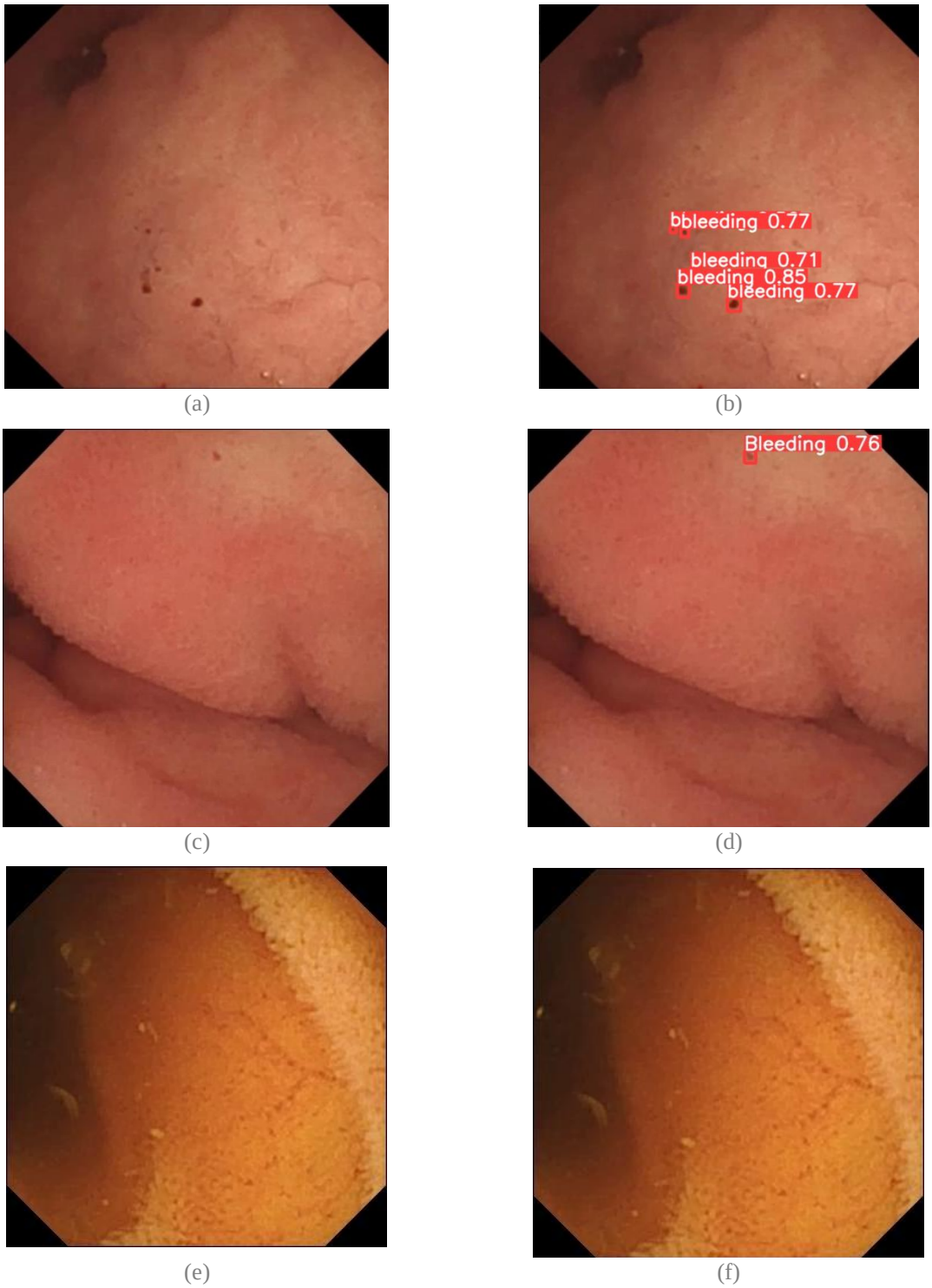
The best outcome is obtained with configuration 4, as shown in [Fig. 41]. Specifically, the learning rate is set to 0.01, the momentum to 0.998, and the number of batches to 8. Whereas the worst outcome is obtained with configuration 2, where the learning rate is set 0.0001, the momentum to 0.99998, and the number of batches to 8. We notice that the model gets better when incrementing the learning rate until a specific limit.

**Table 4. YOLOv3 [9] Performance Results**

| | mAP | IoU | FLOPs |
|---|---|---|---|
| **Validation Set** | 0.872 | 0.7491 | 154.5G |
| **Test Set 1** | 0.835 | 0.7395 | 154.5G |
| **Test Set 2** | 0.828 | 0.589 | 154.5G |

Table 4 reports the performance measures on both the validation set and the test sets. As it can be seen, there is no significant drop in the performance on the test sets with respect to the validation one. In addition, while training the model, a logical prevention response prevent overfitting. Therefore, we can conclude that the trained model is not over-

fitted. Moreover, the model acts almost the same on both test sets. Furthermore, the validation set and the test sets exhibit the same number of FLOPs.




(a) (b) (c) (d) (e) (f)

Fig. 42. Three sample results illustrating the results obtained when using YOLOv3 model. (a) Sample image 1 with multiple bleeding spots, (b) The output of sample image 1, (c) Sample image 2 with a bleeding spot, (d) The output of sample image 2, (e) Sample image 3 without any bleeding spots, (f) The output of sample image 3.

[Fig. 42] shows three examples of YOLOv3 [9] results. As it can be seen, YOLOv3 is able to recognize the bleeding spots, if any, with a confidence score larger than 0.7.

## 7.2. Experiment 2

In this experiment, we evaluate the performance of YOLOv4 [10] . Table 5 shows the 4 considered configurations. Initially the values of learning rate, momentum and number of batches is set to the default values then tuning the values based on the results.

**Table 5. YOLOv4 [10] Hyper-parameter Configuration**

| | Learning Rate | Momentum | Number of Batches |
|---|---|---|---|
| **Configuration 1** | 0.01 | 0.937 | 4 |
| **Configuration 2** | 0.01 | 0.95 | 2 |
| **Configuration 3** | 0.1 | 0.937 | 4 |
| **Configuration 4** | 0. 1 | 0.95 | 2 |

### 7.2.1. Results

[Fig. 43] shows the performance measures of YOLOv4 [10] on the validation set with respect to each considered configuration.

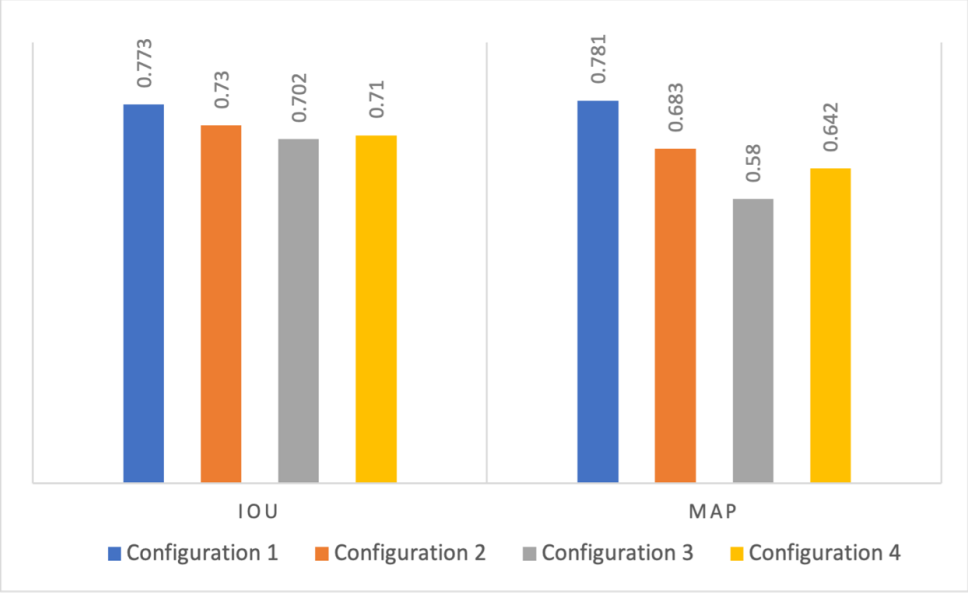



Fig. 43. YOLOv4 [10] performance results.

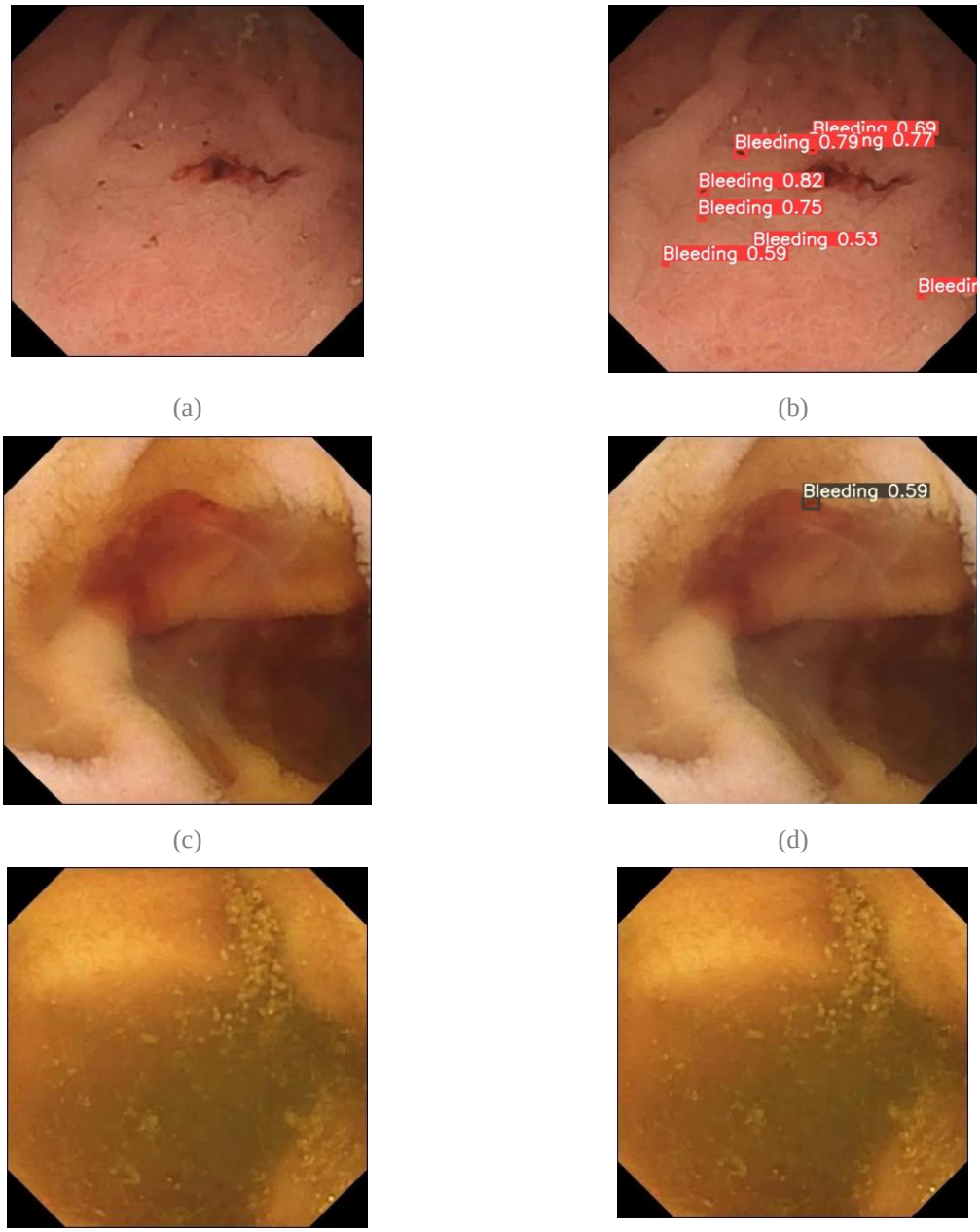
### 7.2.2. Discussion

As shown in [Fig. 43] the best results are obtained using configuration 1, while the worst results were obtained using configuration 3. Accordingly, the hyper-parameters of configuration 1 are set to 0.01 for the learning rate, 0.937 for the momentum, and 4 for the number of batches. The corresponding performance results are depicted in Table 6. As shown, the performance of the test set exhibited relatively small decline with respect to the performance of the validation set, and thus the overfitting assumption is discarded. Moreover, while training the model, a logical prevention response prevent overfitting. Furthermore, the model acts similar on both test sets. Furthermore, the validation data and the test sets exhibit the same number of FLOPs.

[Fig. 44] shows three examples of YOLOv4 [10] results where the bounding box surrounding the object of focus along with the confidence score are presented. We can notice that YOLOv4 is able to recognize the MBS. However, the confidence score is not considered high.

**Table 6. YOLOv4 [10] Performance Results**

| | **mAP** | **IoU** | **FLOPs** |
|---|---|---|---|
| **Validation Set** | 0.781 | 0.773 | 144G |
| **Test Set 1** | 0.741 | 0.729 | 144G |
| **Test Set 2** | 0.734 | 0.736 | 144G |




(a) (b)

(c) (d)

(e) (f)

Fig. 44. Three sample results illustrating the results obtained when using YOLOv4 model. (a) Sample image 1 with multiple bleeding spots, (b) The output of sample image 1, (c) Sample image 2 with a bleeding spot, (d) The output of sample image 2, (e) Sample image 3 without any bleeding spots, (f) The output of sample image 3.

## 7.3. Experiment 3

To detect MBS using YOLOv5 [11], we tuned the hyper-parameters based on the results after initializing each one with the model's default values, then we evaluated YOLOv5 model on the validation set with respect to the four considered configurations. The latter are illustrated in Table 7.

**Table 7. YOLOv5 [11] Hyper-parameter Configuration**

| | Learning Rate | Momentum | Number of Batches |
|---|---|---|---|
| **Configuration 1** | 0.1 | 0.99 | 8 |
| **Configuration 2** | 0.09 | 0.95 | 8 |
| **Configuration 3** | 0.01 | 0.937 | 16 |
| **Configuration 4** | 0.01 | 0.937 | 8 |

### 7.3.1. Results

[Fig. 45] displays the performance measures of YOLOv5 [11] on the validation set with respect to the considered configurations.

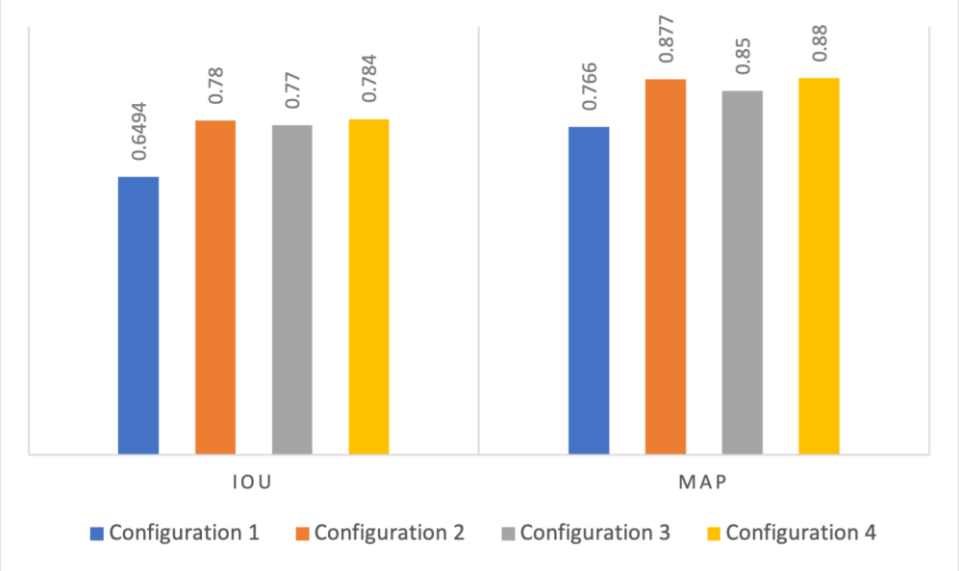



Fig. 45. YOLOv5 [11] performance results on the validation set.

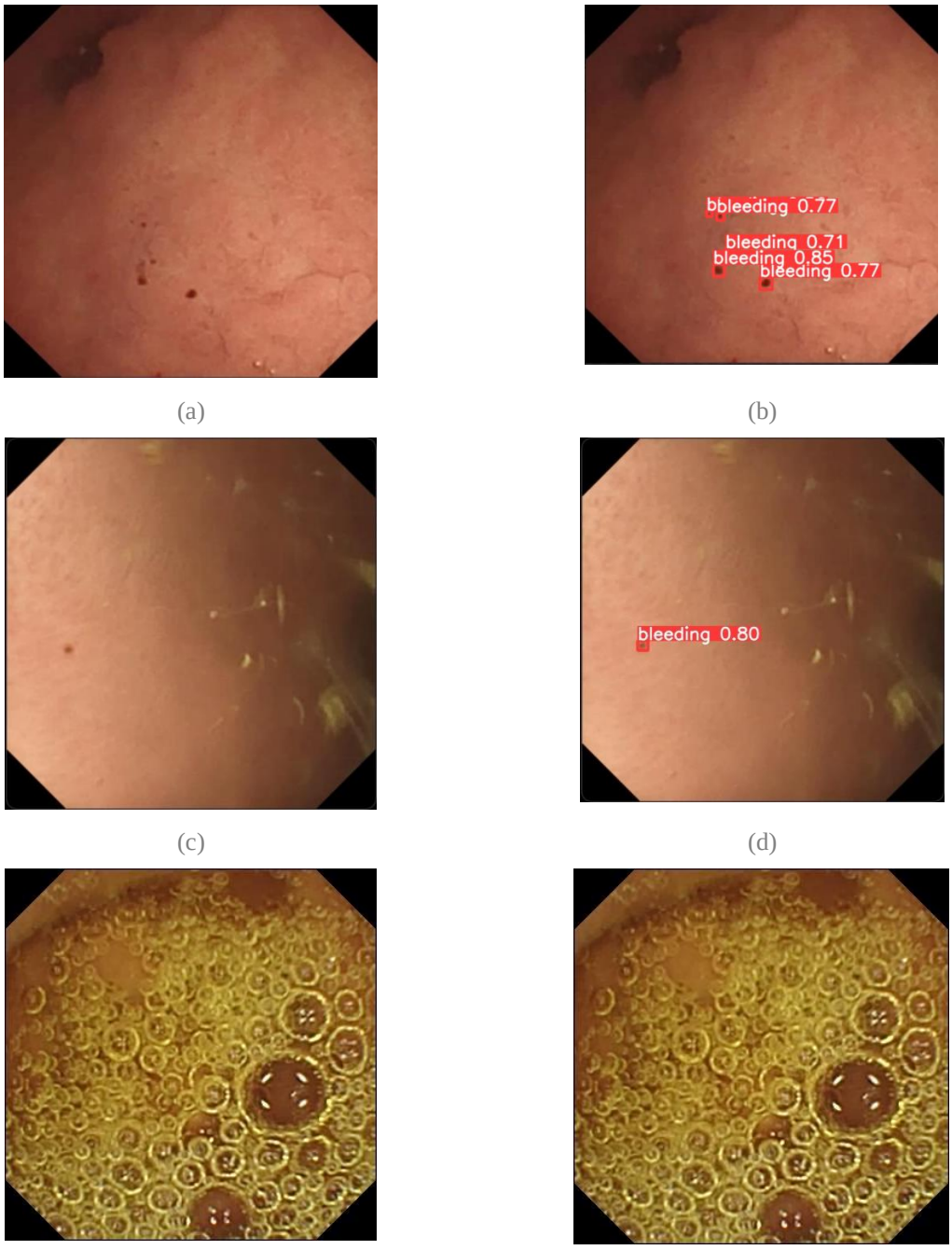
### 7.3.2. Discussion

As shown in [Fig. 45], the best performance is obtained when using configuration 4, and the worst performance is acquired when configuration 1 was used. In fact, the learning rate in configuration 4 was set to 0.01 while the learning rate in configuration 1 is set to 0.1. Thus, the learning rate of 0.1 yielded the overshoot of the model, yet we noticed that when decreasing the learning rate to a specific limit we got a better result. The obtained results on the validation set using configuration 1 are a mAP of 0.766 and an IoU of 0.6494. Table 8 reports the performance results on the validation set and the two test sets using configuration 4. As it can be seen, there is no significant performance drop when using the test sets, therefore, we can conclude that the trained model does not suffer from over-fitting. In fact, while training the model, a logical prevention response prevent overfitting. Furthermore, the validation data and the test sets exhibit the same number of FLOPs.

[Fig. 46] shows three examples of YOLOv5 [11] results. It displays the bounding boxes surrounding the objects of focus along with the confidence score. We can notice that YOLOv5 is able to recognize the MBS.

**Table 8. YOLOv5 [11] Performance Results**

| | mAP | IoU | FLOPs |
|---|---|---|---|
| **Validation Set** | 0.88 | 0.784 | 15.8G |
| **Test Set 1** | 0.843 | 0.745 | 15.8G |
| **Test Set 2** | 0.820 | 0.727 | 15.8G |




(a) (b)

(c) (d)

(e) (f)

Fig. 46. Three sample results illustrating the results obtained when using YOLOv5 model. (a) Sample image 1 with multiple bleeding spots, (b) The output of sample image 1, (c) Sample image 2 with a bleeding spot, (d) The output of sample image 2, (e) Sample image 3 without any bleeding spots, (f) The output of sample image 3.

## 7.4. Experiment 4

The purpose of this experiment is to evaluate YOLOv7 [12] performance. Table 9 shows the configurations that were taken into consideration. Each configuration has different values of the learning rate, momentum, and the number of batches, starting with the initial model's values then tuning values based on the results.

**Table 9. YOLOv7 [12] Hyper-parameter Configuration.**

| | Learning Rate | Momentum | Number of Batches |
|---|---|---|---|
| **Configuration 1** | 0.01 | 0.937 | 8 |
| **Configuration 2** | 0.01 | 0.937 | 2 |
| **Configuration 3** | 0.1 | 0.9 | 4 |
| **Configuration 4** | 0.001 | 0.94 | 16 |

### 7.4.1. Results

[Fig. 47] displays the performance measures of YOLOv7 [12] on the validation set with respect to the considered configurations.

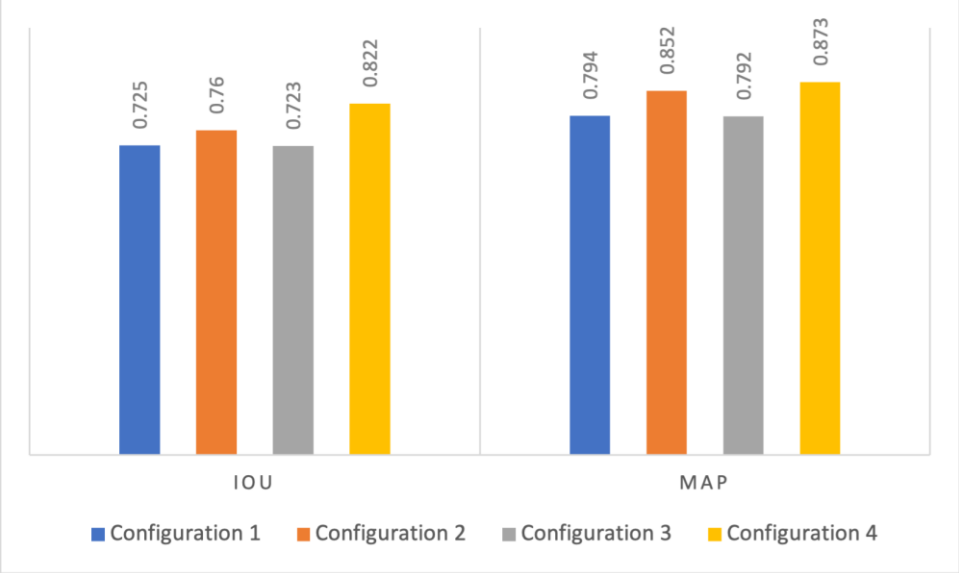



Fig. 47. YOLOv7 [12] performance results on the validation set.

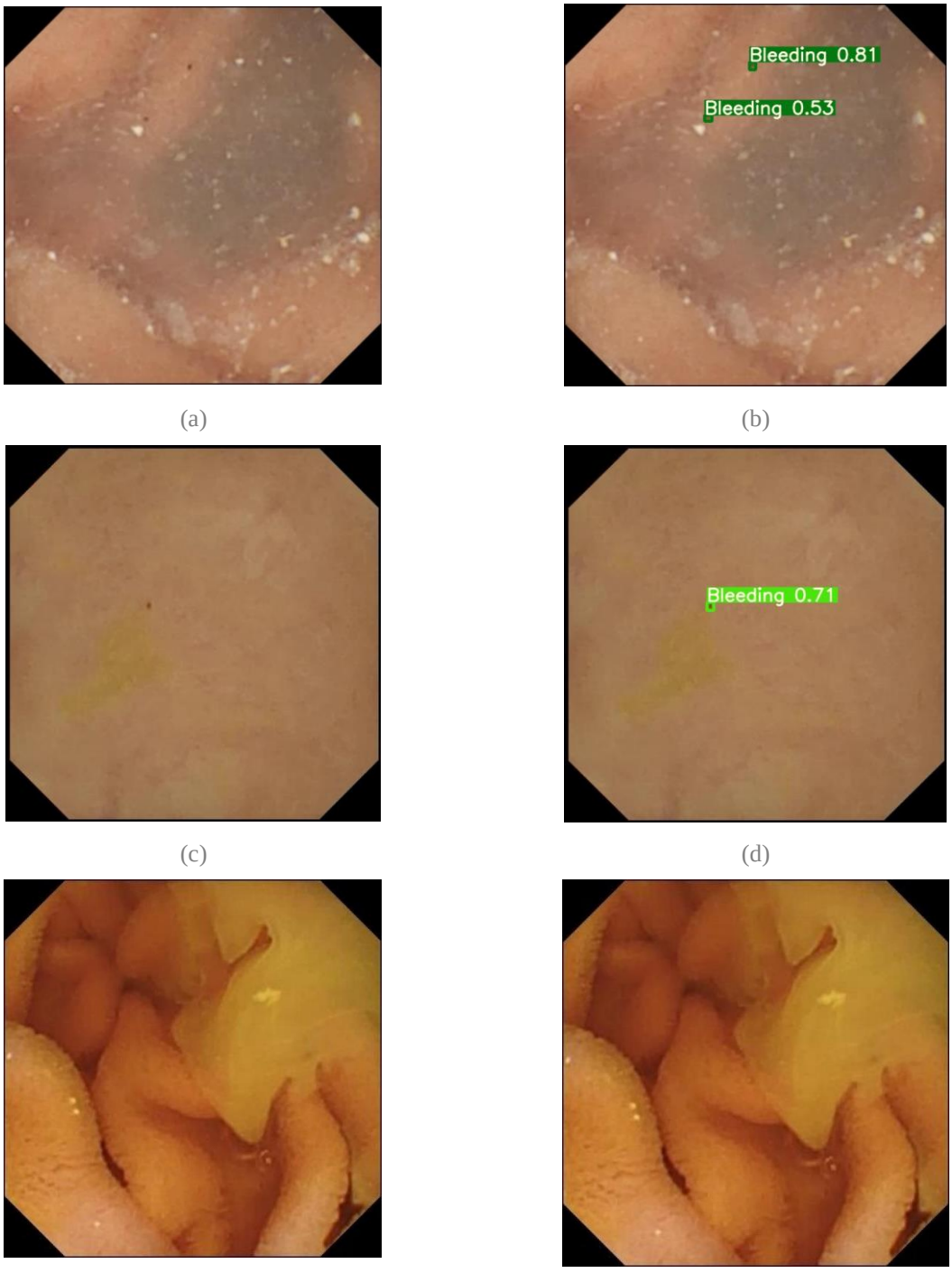
### 7.4.2. Discussion

According to [Fig. 47], configuration 4 provides the best performance, while configuration 3 provides the worst performance. In particular, configuration 4 sets the learning rate to 0.001, while configuration 3 sets the learning rate to 0.1. Thus, the learning rate of 0.1 yielded the overshoot of the model. The obtained results are a mAP of 0.873 and an IoU of 0.822 when using configuration 4 on the validation set. Table 10 depicts the performance results on the validation set and both test sets. There is no significant drop on YOLOv7 performance when using the test sets. Moreover, while training the model, a logical prevention response prevent overfitting. We can therefore infer that the learned model is not overfitted. Furthermore, the validation data and the test sets exhibit the same number of FLOPs.

[Fig.48] shows three examples of YOLOv7 [12] results. It displays the bounding boxes surrounding the objects of focus along with the confidence score. We can notice that YOLOv7 is able to recognize the MBS. Moreover, the results exhibit high confidence score with precise localization of the pattern.

**Table 10. YOLOv7 [12] Performance Results**

| | mAP | IoU | FLOPs |
|---|---|---|---|
| **Validation Set** | 0.873 | 0.822 | 188.9G |
| **Test Set 1** | 0.84 | 0.86 | 188.9G |
| **Test Set 2** | 0.86 | 0.8 | 188.9G |




(a) (b)

(c) (d)

(e) (f)

Fig. 48. Three sample results illustrating the results obtained when using YOLOv7 model. (a) Sample image 1 with multiple bleeding spots, (b) The output of sample image 1, (c) Sample image 2 with a bleeding spot, (d) The output of sample image 2, (e) Sample image 3 without any bleeding spots, (f) The output of sample image 3.

## 7.5. Performance Comparison

[Fig. 49] shows a comparison between the performances of the considered YOLO models on test set 2 in terms of both mAP and IoU. As illustrated in [Fig. 49], YOLOv3 performs better than YOLOv4 and YOLOv5 in terms of recognition with mAP equal to 0.828. This is an expected outcome since the architecture of YOLOv3 consists of residual blocks. One of them is exploited specifically for small object detections which concord with the small pattern of the bleeding spots. Moreover, in terms of IoU, YOLOv4 achieves an IoU of 0.736 which is better than 0.589 for YOLOv3 and 0.727 for YOLOv5. In fact, YOLOv4 is better in localizing bleeding spots since it incorporates two stage detectors. The first one is called the one stage object detector and the second one is the two-stage object detector. Nevertheless, YOLOv5 exploits path aggregation network that enhances the model localization ability. Alternatively, YOLOv7 achieved the highest IoU and mAP equal to 0.8 and 0.86 respectively. This makes YOLOv7 the most appropriate model to design the proposed approach.

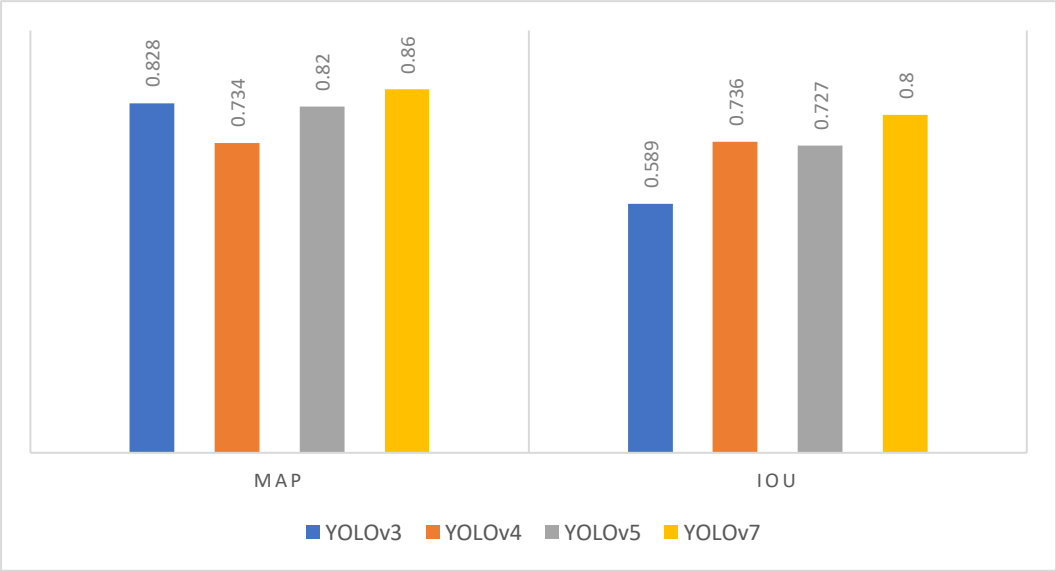
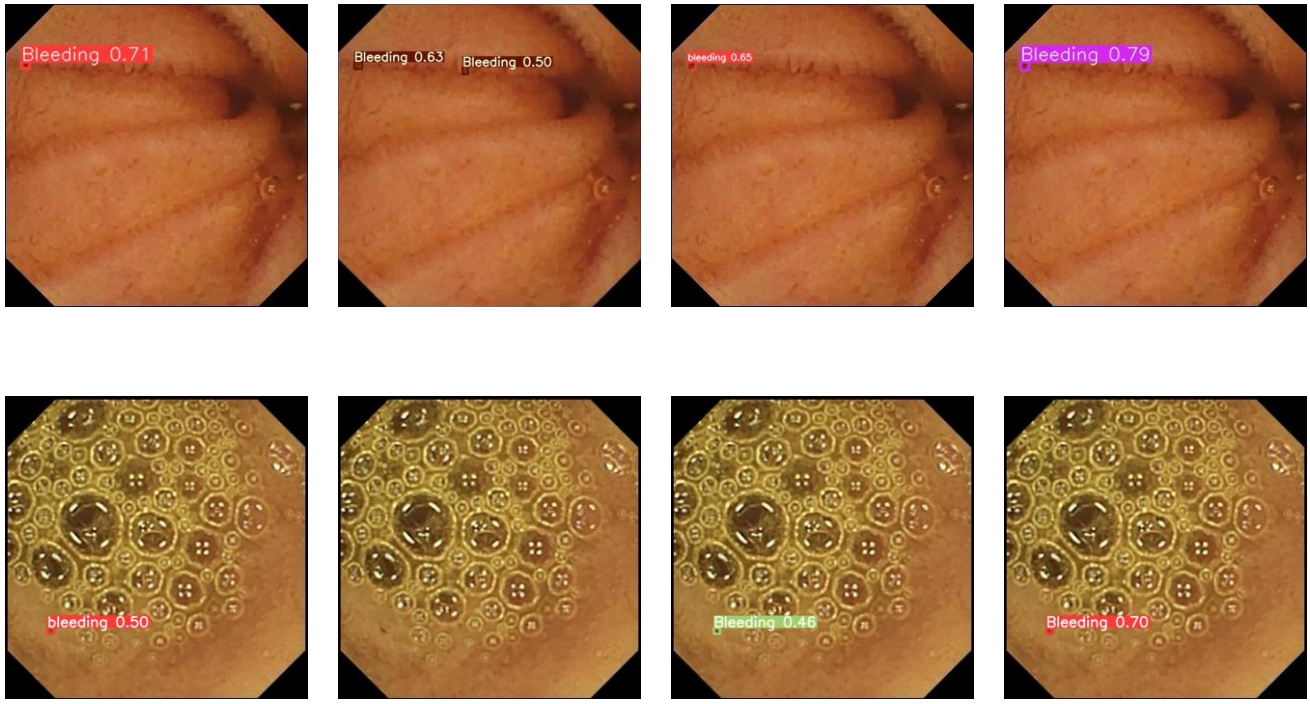



Fig. 49. Performance comparison of YOLOv3, YOLOv4, YOLOv5, and YOLOv7 in terms of mAP and IoU.

Regarding the confidence score, it is generally high for YOLOv7 [12]. Yet, there are some counter examples that show low confidence score. This may be caused by the variation of the spots' color. For instance, as shown in figure [Fig. 48, b], one of the

bounding boxes annotated with confidence score of 0.53 as the color of the spot is lighter than other spot with higher confidence score that have darker color.




(a) (b) (c) (d)

Fig. 50. Comparison of the results on two sample images. Single bleeding spot output result using (a) YOLOv3, (b) YOLOv4, (c) YOLOv5, (d) and YOLOv7.

[Fig. 50] shows 2 sample image results obtained using the considered YOLO models. As shown in [Fig. 50 b] YOLOv4 detected a False Positive object on the first sample, and failed to detect the bleeding spot on the second sample image.

## 7.6. Data Augmentation

Data augmentation is employed to increase the size of the training data set conveyed to the best performant model, namely YOLOv7 [12] . This is achieved by adding more images to train the model. These images were created by flipping and rotating existing training images. In particular, we implemented vertical flip, horizontal flip, 90° clockwise rotation, 90° counter-clockwise rotation, 45° rotation, and -45° rotation. The augmented

dataset contains “1056” images. [Fig. 51] depicts sample images from the augmented dataset. The performance of YOLOv7 without using the augmented data is compared with its performance when training the model with additional data. Table 11 depicts the performance of YOLOv7 when including and excluding data augmentation. As it can be seen, the augmented dataset improved YOLOv7 performance in terms of mAP.

**Table 11. Performance Comparison of YOLOv7 [12] when using data augmentation and without using it**

| | **mAP** | **IoU** | **FLOPs** |
|---|---|---|---|
| **Test results using data augmentation** | 0.883 | 0.81 | 188.9G |
| **Test results without data augmentation** | 0.86 | 0.8 | 188.9G |

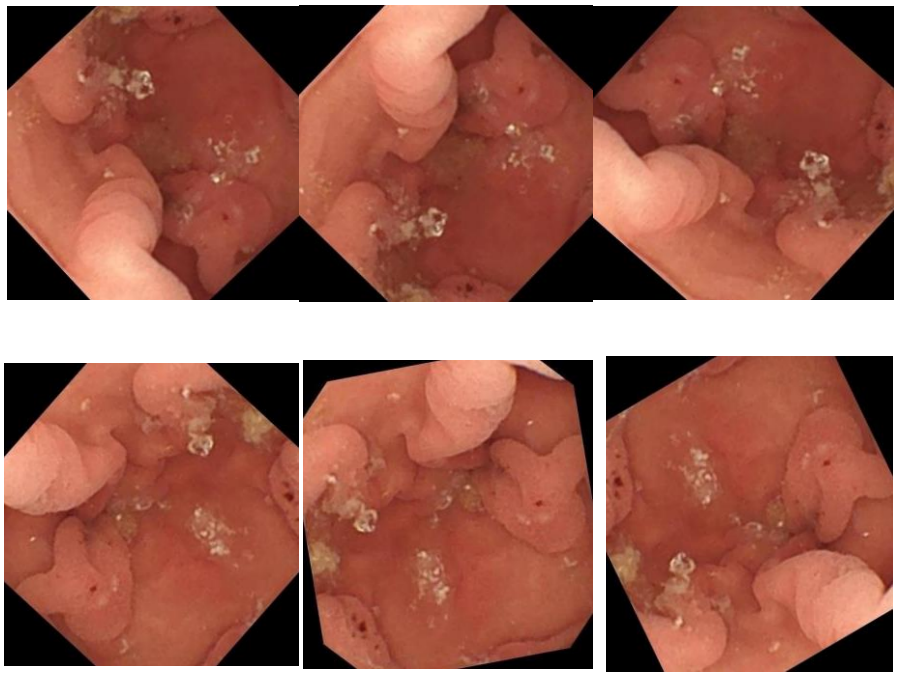


Fig. 51. Sample images obtained using data augmentation.

## 7.7. Performance Analysis

### 7.7.1. Space Complexity

We compare YOLOv3 [9], YOLOv4 [10], YOLOv5 [11] and YOLOv7 [12] in terms of space complexity. It refers to the space needed to store and train the model. Table 12 shows the space memory for each model. As depicted, YOLOv5 requires less space memory due to its optimized implementation, while YOLOv4 needs more space memory.

**Table 12. Performance Analysis in Terms of Space Complexity**

| Model | Space |
|---|---|
| YOLOv3 [9] | 123.5 MB |
| YOLOv4 [10] | 491.6 MB |
| YOLOv5 [11] | 14.4 MB |
| YOLOv7 [12] | 142 MB |

### 7.7.2. Time Complexity

As illustrated in in [Fig. 49], YOLOv7 exceeds the other models in terms of test result, yet there is no significant increase in term of time complexity. It is noticeable that YOLOv4 consumed more time when training the model. On the other hand, when training YOLOv5 it took the least time, and that is predictable since YOLOv5 uses less floating-point operations. Regarding the time considered to train all four models, Table 13 reports the training and testing times per image when using Google Collaboratory to train all models.

**Table 13. Performance Analysis in Terms of Training and Testing Time Complexity**

| Model | Training Time (s) | Testing Time (ms) |
|---|---|---|
| YOLOv3 [9] | 6.5295 | 0.00026 |
| YOLOv4 [10] | 23.07 | 0.00837 |
| YOLOv5 [11] | 3.8103 | 0.00031 |
| YOLOv7 [12] | 11.0554 | 0.00124 |

## 7.8. Conclusion

This chapter presents and discusses the results of the conducted experiments. According to the analysis and comparison of these results, YOLOv7 is the most appropriate model to use for the proposed system. Moreover, data augmentation enhanced the performance of YOLOv7 in terms of recognition.

# Chapter 8: Conclusions and Future Works

The arduousness of MBS diagnosis through the burdensome visualization of an 8-hour WCE video of the GI tract has led to the development of aided-diagnosis system. They are based on pattern recognition techniques to detect MBS.

In this project, we proposed to design an aided- diagnosis for MBS detection from WCE video. It is based on deep learning pattern recognition model. In particular, different versions of YOLO model are investigated.

For this purpose, the background related to deep learning and pattern recognition is studied. More specifically, CNN model and YOLO based architecture are detailed. Moreover, transfer learning and data augmentation techniques are presented.

The exploration of the related works showed that the reported works can be categorized into classification-based approach and detection-based approach. While the former approaches classify the video frames, the latter ones also localize the bleeding spots within the frames. Nevertheless, most of the detection approaches require segmenting the frames than classifying the region of interest.

Based on the study of the background and the exploration of the related works, a methodology is proposed to design the proposed system. It is based on comparing four versions of YOLO in terms of recognition performance and time efficiency. These are YOLOv3, YOLOv4, YOLv5, and YOLOv7.

The four considered YOLO models are trained and tested. The comparison and the analysis of the obtained results yielded the selection of the most suitable YOLO model for MBS recognition from WCE videos of the GI tract. Namely, YOLOv7 outperformed the other models.

As future works, the proposed system can be implemented to an applicable and more convenient user-friendly system that can be used by physicians. Additionally, the performance of the proposed system can be further enhanced by collecting more WCE data to train the model.

# References


[1] A. D. Sperber *et al.*, "Worldwide Prevalence and Burden of Functional Gastrointestinal Disorders, Results of Rome Foundation Global Study," *Gastroenterology*, vol. 160, no. 1, pp. 99-114.e3, Jan. 2021, doi: 10.1053/j.gastro.2020.04.014.

[2] R. L. Siegel *et al.*, "Colorectal cancer statistics, 2020," *CA. Cancer J. Clin.*, vol. 70, no. 3, pp. 145–164, May 2020, doi: 10.3322/caac.21601.

[3] "Definition & Facts of GI Bleeding | NIDDK," *National Institute of Diabetes and Digestive and Kidney Diseases*. https://www.niddk.nih.gov/health-information/digestive-diseases/gastrointestinal-bleeding/definition-facts (accessed Oct. 10, 2022).

[4] T. Wilkins, B. Wheeler, and M. Carpenter, "Upper Gastrointestinal Bleeding in Adults: Evaluation and Management," *Am. Fam. Physician*, vol. 101, no. 5, pp. 294–300, Mar. 2020.

[5] "Prof. M. S. Khuroo, MD, DM, FRCP, FACP, MACP (Emeritus), Director, Digestive Diseases Centre, Dr. Khuroo Medical Clinic, Srinagar, Kashmir, India. Former Director, Professor & Head Gastroenterology, Chairman, Dept. of Medicine, Sher-I-Kashmir Institute of Medical Sciences, Soura, Srinagar; Former Consultant & Head Gastroenterology and Liver Transplantation, King Faisal Specialist Hospital & Research Centre, Riyadh, Saudi Arabia." http://www.drkhuroo.in/search_results.php?key=a5e1c8c0ecf32d8cbba390b28cb72452c6dd7f90&cat_id=1&string=multiple+bleeding+spots (accessed Oct. 10, 2022).

[6] P. C. Khun, Z. Zhuo, L. Z. Yang, L. Liyuan, and L. Jiang, "Feature selection and classification for Wireless Capsule Endoscopic frames," in *2009 International Conference on Biomedical and Pharmaceutical Engineering*, Dec. 2009, pp. 1–6. doi: 10.1109/ICBPE.2009.5384106.

[7] S. Alotaibi, S. Qasim, O. Bchir, and M. M. Ben Ismail, "Empirical Comparison of Visual Descriptors for Multiple Bleeding Spots Recognition in Wireless Capsule Endoscopy Video," in *Computer Analysis of Images and Patterns*, Berlin, Heidelberg, 2013, pp. 402–407. doi: 10.1007/978-3-642-40246-3_50.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[9] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement." arXiv, Apr. 08, 2018. Accessed: Oct. 16, 2022. [Online]. Available: http://arxiv.org/abs/1804.02767

[10] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection." arXiv, Apr. 22, 2020. Accessed: Oct. 16, 2022. [Online]. Available: http://arxiv.org/abs/2004.10934

[11] "Releases · ultralytics/yolov5," *GitHub*. https://github.com/ultralytics/yolov5/releases (accessed Oct. 17, 2022).

[12] 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." arXiv, Jul. 06, 2022. Accessed: Oct. 08, 2022. [Online]. Available: http://arxiv.org/abs/2207.02696
[13] L. Deng and D. Yu, "Deep Learning: Methods and Applications," *Found. Trends® Signal Process.*, vol. 7, no. 3–4, pp. 197–387, Jun. 2014, doi: 10.1561/2000000039.
[14] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Comput. Sci.*, vol. 2, no. 6, p. 420, Aug. 2021, doi: 10.1007/s42979-021-00815-1.
[15] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, vol. 9, no. 4, Art. no. 4, Aug. 2018, doi: 10.1007/s13244-018-0639-9.
[16] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way," *Medium*, Dec. 17, 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (accessed Oct. 10, 2022).
[17] "Full Text PDF." Accessed: Oct. 10, 2022. [Online]. Available: https://www.researchgate.net/profile/A-Sufian/publication/337401161_Fundamental_Concepts_of_Convolutional_Neural_Network/links/5ed612a7299bf1c67d3292e6/Fundamental-Concepts-of-Convolutional-Neural-Network.pdf
[18] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks." arXiv, Dec. 02, 2015. doi: 10.48550/arXiv.1511.08458.
[19] Arc, "Convolutional Neural Network," *Medium*, Dec. 26, 2018. https://towardsdatascience.com/convolutional-neural-network-17fb77e76c05 (accessed Oct. 10, 2022).
[20] H. J. Jie and P. Wanda, "RunPool: A Dynamic Pooling Layer for Convolution Neural Network," *Int. J. Comput. Intell. Syst.*, vol. 13, no. 1, pp. 66–76, Jan. 2020, doi: 10.2991/ijcis.d.200120.002.
[21] A. Zaafouri and M. Sayadi, "Medication Code Recognition using Convolutional Neural Network," in *2020 5th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, Sep. 2020, pp. 1–5. doi: 10.1109/ATSIP49331.2020.9231801.
[22] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Prog. Artif. Intell.*, vol. 9, no. 2, pp. 85–112, Jun. 2020, doi: 10.1007/s13748-019-00203-0.
[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
[24] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition." arXiv, Apr. 10, 2015. Accessed: Oct. 17, 2022. [Online]. Available: http://arxiv.org/abs/1409.1556
[25] C. Szegedy *et al.*, "Going Deeper with Convolutions." arXiv, Sep. 16, 2014. Accessed: Oct. 17, 2022. [Online]. Available: http://arxiv.org/abs/1409.4842

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition." arXiv, Dec. 10, 2015. Accessed: Oct. 17, 2022. [Online]. Available: http://arxiv.org/abs/1512.03385
[27] L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification using Deep Learning." arXiv, Dec. 13, 2017. Accessed: Oct. 17, 2022. [Online]. Available: http://arxiv.org/abs/1712.04621
[28] A. Asperti and C. Mastronardo, "The Effectiveness of Data Augmentation for Detection of Gastrointestinal Diseases from Endoscopical Images." arXiv, Dec. 11, 2017. Accessed: Oct. 17, 2022. [Online]. Available: http://arxiv.org/abs/1712.03689
[29] I. Goodfellow *et al.*, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems*, 2014, vol. 27. Accessed: Oct. 17, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html
[30] Y. Liu and Y. F. Zheng, "One-against-all multi-class SVM classification using reliability measures," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, Jul. 2005, vol. 2, pp. 849–854 vol. 2. doi: 10.1109/IJCNN.2005.1555963.
[31] O. Bchir, M. M. Ben Ismail, and N. AlZahrani, "Multiple bleeding detection in wireless capsule endoscopy," *Signal Image Video Process.*, vol. 13, no. 1, pp. 121–126, Feb. 2019, doi: 10.1007/s11760-018-1336-3.
[32] J. M. Martinez, R. Koenen, and F. Pereira, "MPEG-7: the generic multimedia content description standard, part 1," *IEEE Multimed.*, vol. 9, no. 2, pp. 78–87, Jun. 2002, doi: 10.1109/93.998074.
[33] M. Verma, B. Raman, and S. Murala, "Multi-resolution Local extrema patterns using discrete wavelet transform," in *2014 Seventh International Conference on Contemporary Computing (IC3)*, Aug. 2014, pp. 577–582. doi: 10.1109/IC3.2014.6897237.
[34] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Comput. Geosci.*, vol. 10, no. 2, pp. 191–203, Jan. 1984, doi: 10.1016/0098-3004(84)90020-7.
[35] R. Shahril, A. Saito, A. Shimizu, and S. Baharun, "Bleeding Classification of Enhanced Wireless Capsule Endoscopy Images using Deep Convolutional Neural Network," p. 18.
[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Ha, "Gradient-Based Learning Applied to Document Recognition," p. 46, 1998.
[37] A. Caroppo, A. Leone, and P. Siciliano, "Deep transfer learning approaches for bleeding detection in endoscopy images," *Comput. Med. Imaging Graph.*, vol. 88, p. 101852, Mar. 2021, doi: 10.1016/j.compmedimag.2020.101852.
[38] S. Mukherjee, "The Annotated ResNet-50," *Medium*, Aug. 18, 2022. https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758 (accessed Oct. 21, 2022).
[39] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision." arXiv, Dec. 11, 2015. doi: 10.48550/arXiv.1512.00567.

[40] R. Mostafiz, M. M. Rahman, A. Islam, and S. Belkasim, "machine learning & knowledge extraction Focal Liver Lesion Detection in Ultrasound Image Using Deep Feature Fusions and Super Resolution," *Mach. Learn. Knowl. Extr.*, vol. 2, Jul. 2020, doi: 10.3390/make2030010.
[41] "Inception V3 Model Architecture," *OpenGenus IQ: Computing Expertise & Legacy*, Sep. 05, 2021. https://iq.opengenus.org/inception-v3-model-architecture/ (accessed Oct. 21, 2022).
[42] F. Rustam *et al.*, "Wireless Capsule Endoscopy Bleeding Images Classification Using CNN Based Model," *IEEE Access*, vol. PP, pp. 1–1, Feb. 2021, doi: 10.1109/ACCESS.2021.3061592.
[43] A. Pujara, "Image Classification With MobileNet," *Analytics Vidhya*, Jul. 15, 2020. https://medium.com/analytics-vidhya/image-classification-with-mobilenet-cc6fbb2cd470 (accessed Oct. 21, 2022).
[44] X. Jia and M. Q.-H. Meng, "A deep convolutional neural network for bleeding detection in Wireless Capsule Endoscopy images," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Orlando, FL, USA, Aug. 2016, pp. 639–642. doi: 10.1109/EMBC.2016.7590783.
[45] P. Sivakumar and B. M. Kumar, "A novel method to detect bleeding frame and region in wireless capsule endoscopy video," *Clust. Comput.*, vol. 22, no. S5, pp. 12219–12225, Sep. 2019, doi: 10.1007/s10586-017-1584-y.
[46] S. Suman *et al.*, *Detection and Classification of Bleeding Region in WCE Images using Color Feature*. 2017. doi: 10.1145/3095713.3095731.
[47] F. Wu, C. Zhu, J. Xu, M. W. Bhatt, and A. Sharma, "Research on image text recognition based on canny edge detection algorithm and k-means algorithm," *Int. J. Syst. Assur. Eng. Manag.*, vol. 13, no. S1, pp. 72–80, Mar. 2022, doi: 10.1007/s13198-021-01262-0.
[48] P. V. V. Kishore, A. S. C. S. Sastry, A. Kartheek, and Sk. H. Mahatha, "Block based thresholding in wavelet domain for denoising ultrasound medical images," in *2015 International Conference on Signal Processing and Communication Engineering Systems*, Guntur, India, Jan. 2015, pp. 265–269. doi: 10.1109/SPACES.2015.7058262.
[49] K. Pogorelov *et al.*, "Bleeding detection in wireless capsule endoscopy videos — Color versus texture features," *J. Appl. Clin. Med. Phys.*, vol. 20, no. 8, pp. 141–154, 2019, doi: 10.1002/acm2.12662.
[50] C. Sri Kusuma Aditya, M. Hani'ah, R. R. Bintana, and N. Suciati, "Batik classification using neural network with gray level co-occurence matrix and statistical color feature extraction," in *2015 International Conference on Information & Communication Technology and Systems (ICTS)*, Surabaya, Sep. 2015, pp. 163–168. doi: 10.1109/ICTS.2015.7379892.
[51] S. Kalmegh, "Analysis of WEKA Data Mining Algorithm REPTree, Simple Cart and RandomTree for Classification of Indian News," vol. 2, no. 2, p. 9.
[52] E. K. Sahin, I. Colkesen, and T. Kavzoglu, "A comparative assessment of canonical correlation forest, random forest, rotation forest and logistic regression methods for landslide susceptibility mapping," *Geocarto Int.*, vol. 35, no. 4, pp. 341–363, Mar. 2020, doi: 10.1080/10106049.2018.1516248.

[53] M. Abedini, B. Ghasemian, A. Shirzadi, and D. T. Bui, "A comparative study of support vector machine and logistic model tree classifiers for shallow landslide susceptibility modeling," *Environ. Earth Sci.*, vol. 78, no. 18, p. 560, Sep. 2019, doi: 10.1007/s12665-019-8562-z.
[54] T. Ghosh and J. Chakareski, "Deep Transfer Learning for Automated Intestinal Bleeding Detection in Capsule Endoscopy Imaging," *J. Digit. Imaging*, vol. 34, no. 2, pp. 404–417, Apr. 2021, doi: 10.1007/s10278-021-00428-3.
[55] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation." arXiv, Oct. 10, 2016. Accessed: Oct. 21, 2022. [Online]. Available: http://arxiv.org/abs/1511.00561
[56] P. Nepal, "AlexNet Architecture Explained," *Analytics Vidhya*, Aug. 05, 2020. https://medium.com/analytics-vidhya/alexnet-architecture-explained-5d19e3dca2bb (accessed Oct. 21, 2022).
[57] P. Coelho, A. Pereira, A. Leite, M. Salgado, and A. Cunha, "A Deep Learning Approach for Red Lesions Detection in Video Capsule Endoscopies," in *Image Analysis and Recognition*, vol. 10882, A. Campilho, F. Karray, and B. ter Haar Romeny, Eds. Cham: Springer International Publishing, 2018, pp. 553–561. doi: 10.1007/978-3-319-93000-8_63.
[58] L. Lan, C. Ye, C. Wang, and S. Zhou, "Deep Convolutional Neural Networks for WCE Abnormality Detection: CNN Architecture, Region Proposal and Transfer Learning," *IEEE Access*, vol. 7, pp. 30017–30032, 2019, doi: 10.1109/ACCESS.2019.2901568.
[59] M. Z. Alom, C. Yakopcic, M. Hasan, T. M. Taha, and V. K. Asari, "Recurrent residual U-Net for medical image segmentation," *J. Med. Imaging*, vol. 6, no. 01, p. 1, Mar. 2019, doi: 10.1117/1.JMI.6.1.014006.
[60] "pillcam.com." http://www.pillcam.com/ (accessed Oct. 31, 2022).
[61] Medtronic, "Future of healthcare." https://www.medtronic.com/us-en/index.html (accessed Oct. 31, 2022).
[62] A. Koulaouzidis *et al.*, "KID Project: an internet-based digital video atlas of capsule endoscopy for research purposes," *Endosc. Int. Open*, vol. 5, no. 6, pp. E477–E483, Jun. 2017, doi: 10.1055/s-0043-105488.
[63] "PillCam™ SB 3 System | Medtronic." https://www.medtronic.com/covidien/en-us/products/capsule-endoscopy/pillcam-sb-3-system.html (accessed Oct. 31, 2022).
[64] D. K. Iakovidis and A. Koulaouzidis, "Automatic lesion detection in wireless capsule endoscopy — A simple solution for a complex problem," in *2014 IEEE International Conference on Image Processing (ICIP)*, Paris, France, Oct. 2014, pp. 2236–2240. doi: 10.1109/ICIP.2014.7025453.
[65] D. K. Iakovidis, S. V. Georgakopoulos, M. Vasilakakis, A. Koulaouzidis, and V. P. Plagianakos, "Detecting and Locating Gastrointestinal Anomalies Using Deep Learning and Iterative Cluster Unification," *IEEE Trans. Med. Imaging*, vol. 37, no. 10, pp. 2196–2210, Oct. 2018, doi: 10.1109/TMI.2018.2837002.
[66] "Official Portal University Malaya Medical Centre." https://www.ummc.edu.my/department/department_sub.asp?kodjabatan=9z2Q3t6k&keyid= (accessed Oct. 31, 2022).

[67] I. Tziortziotis, F.-M. Laskaratos, and S. Coda, “Role of Artificial Intelligence in Video Capsule Endoscopy,” *Diagnostics*, vol. 11, no. 7, p. 1192, Jun. 2021, doi: 10.3390/diagnostics11071192.
[68] T. Diwan, G. Anirudh, and J. V. Tembhurne, “Object detection using YOLO: challenges, architectural successors, datasets and applications,” *Multimed. Tools Appl.*, Aug. 2022, doi: 10.1007/s11042-022-13644-y.
[69] “(9) (PDF) KVASIR: A Multi-Class Image Dataset for Computer Aided Gastrointestinal Disease Detection.” https://www.researchgate.net/publication/316215961_KVASIR_A_Multi-Class_Image_Dataset_for_Computer_Aided_Gastrointestinal_Disease_Detection (accessed Oct. 31, 2022).
[70] Ju, Luo, Wang, Hui, and Chang, “The Application of Improved YOLO V3 in Multi-Scale Target Detection,” *Appl. Sci.*, vol. 9, no. 18, p. 3775, Sep. 2019, doi: 10.3390/app9183775.
[71] N. Kwak and D. Kim, “Object detection technology trend and development direction using deep learning,” *Int. J. Adv. Cult. Technol.*, vol. 8, no. 4, pp. 119–128, Dec. 2020, doi: 10.17703/IJACT.2020.8.4.119.
[72] V. Rajput, “YOLO v4 explained in full detail,” *AIGuys*, May 19, 2022. https://medium.com/aiguys/yolo-v4-explained-in-full-detail-5200b77aa825 (accessed Oct. 16, 2022).
[73] M. Sozzi, S. Cantalamessa, A. Cogato, A. Kayad, and F. Marinello, “Automatic Bunch Detection in White Grape Varieties Using YOLOv3, YOLOv4, and YOLOv5 Deep Learning Algorithms,” *Agronomy*, vol. 12, no. 2, Art. no. 2, Feb. 2022, doi: 10.3390/agronomy12020319.
[74] T.-K. Nguyen, L. Vu, V. Vu, T.-D. Hoang, S.-H. Liang, and M.-Q. Tran, “Analysis of Object Detection Models on Duckietown Robot Based on YOLOv5 Architectures,” vol. 4, pp. 17–12, Mar. 2022.
[75] X. Xu, X. Zhang, and T. Zhang, “Lite-YOLOv5: A Lightweight Deep Learning Detector for On-Board Ship Detection in Large-Scene Sentinel-1 SAR Images,” *Remote Sens.*, vol. 14, no. 4, p. 1018, Feb. 2022, doi: 10.3390/rs14041018.
[76] K. Patel, C. Bhatt, and P. L. Mazzeo, “Deep Learning-Based Automatic Detection of Ships: An Experimental Study Using Satellite Images,” *J. Imaging*, vol. 8, no. 7, p. 182, Jun. 2022, doi: 10.3390/jimaging8070182.
[77] G. Boesch, “YOLOv7: The Most Powerful Object Detection Algorithm (2022 Guide),” *viso.ai*, Aug. 11, 2022. https://viso.ai/deep-learning/yolov7-guide/ (accessed Oct. 08, 2022).
[78] P. H. Smedsrud *et al.*, “Kvasir-Capsule, a video capsule endoscopy dataset,” *Sci. Data*, vol. 8, no. 1, Art. no. 1, May 2021, doi: 10.1038/s41597-021-00920-z.
[79] “how to use Background images in training? · Issue #2844 · ultralytics/yolov5.” https://github.com/ultralytics/yolov5/issues/2844 (accessed Feb. 07, 2023).
[80] “heartexlabs/labelImg.” Heartex, Oct. 30, 2022. Accessed: Oct. 30, 2022. [Online]. Available: https://github.com/heartexlabs/labelImg
[81] Naoki, “Object Detection: Intersection over Union (IoU),” *Medium*, Oct. 08, 2022. https://naokishibuya.medium.com/object-detection-intersection-over-union-iou-f7b91555eb5f (accessed Oct. 26, 2022).

[82] B. Wang, “A Parallel Implementation of Computing Mean Average Precision.” arXiv, Jun. 19, 2022. Accessed: Oct. 22, 2022. [Online]. Available: http://arxiv.org/abs/2206.09504
[83] “Floating-Point Operation - an overview | ScienceDirect Topics.” https://www.sciencedirect.com/topics/computer-science/floating-point-operation (accessed Oct. 30, 2022).
[84] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, Jul. 2009, doi: 10.1016/j.ipm.2009.03.002.
[85] A. Anwar, “What is Average Precision in Object Detection & Localization Algorithms and how to calculate it?,” *Medium*, May 13, 2022. https://towardsdatascience.com/what-is-average-precision-in-object-detection-localization-algorithms-and-how-to-calculate-it-3f330efe697b (accessed Oct. 22, 2022).
[86] “Roboflow: Give your software the power to see objects in images and video.” https://roboflow.com/ (accessed Feb. 07, 2023).
[87] “numpy: Fundamental package for array computing in Python.” Accessed: Feb. 07, 2023. [MacOS, Microsoft :: Windows, POSIX, Unix]. Available: https://www.numpy.org
[88] “opencv/opencv.” OpenCV, Feb. 07, 2023. Accessed: Feb. 07, 2023. [Online]. Available: https://github.com/opencv/opencv
[89] “Pillow,” *Pillow (PIL Fork)*. https://pillow.readthedocs.io/en/stable/index.html (accessed Feb. 07, 2023).
[90] K. Simonov, “PyYAML: YAML parser and emitter for Python.” Accessed: Feb. 07, 2023. [OS Independent]. Available: https://pyyaml.org/
[91] L. Zhu, “thop 0.1.1.post2209072238 : A tool to count the FLOPs of PyTorch model.” Accessed: Feb. 07, 2023. [Online]. Available: https://github.com/Lyken17/pytorch-OpCounter/