

# **DEVELOPMENT OF DEEP LEARNING MODEL FOR BUILDING OPENING DETECTION AS THE APPLICATION OF UNMANNED AERIAL VEHICLES (UAV) IN URBAN SEARCH AND RESCUE MISSION**

ALI SUROJAYA

June 2023

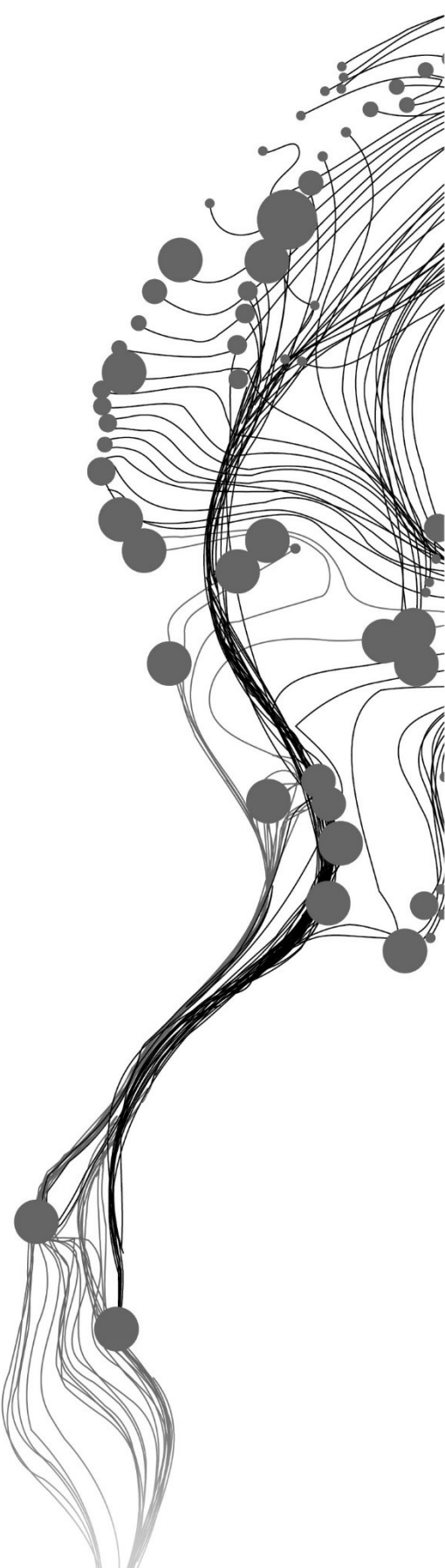
## **SUPERVISORS:**

Prof. Dr.Ing. Francesco Nex (First Supervisor)

Dr. John Ray Bergado (Second Supervisor)

Ning Zhang (Scientific Advisor)





# **DEVELOPMENT OF DEEP LEARNING MODEL FOR BUILDING OPENING DETECTION AS THE APPLICATION OF UNMANNED AERIAL VEHICLES (UAV) IN URBAN SEARCH AND RESCUE MISSION**

**ALI SUROJAYA**

Enschede, The Netherlands, June 2023

Thesis submitted to the Faculty of Geo-Information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.

Specialization: Geoinformatics

## **SUPERVISORS:**

Prof.Dr.Ing. Francesco Nex (First Supervisor)

Dr. John Ray Bergado (Second Supervisor)

Ning Zhang (Scientific Advisor)

## **THESIS ASSESSMENT BOARD:**

Prof.Dr.Ir. M.G. Vosselman

Prof.Dr. Norman Kerle

#### DISCLAIMER

This document describes work undertaken as part of a programme of study at the Faculty of Geo-Information Science and Earth Observation of the University of Twente. All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the Faculty.

## ABSTRACT

Automatic Unmanned Aerial Vehicle (UAV) is a promising technology to minimise human involvement in dangerous activities like urban search and rescue missions (USAR). For this purpose, the ability of a UAV that can map and locate the openings in the damaged building is needed to be able to transit between outdoor and indoor environments. This study focuses on developing a deep learning model for real-time damaged building opening detection by comparing the performance of single and multi-task learning-based detectors. This study consists of four phases: dataset development, single-task network training, multi-task network design, and model performance testing and comparison. This work successfully provides a novel damaged building opening dataset containing images and mask annotations. The deep learning-based detector used in this study is based on YOLOv5 as the most stable current state-of-the-art of real-time object detection model. First, this study compares the different versions of YOLOv5 (i.e. small, medium, and large) to perform damaged building opening detections. Second, multi-task learning YOLOv5 is trained on the same dataset and compared with the single-task detector. This study found that multi-task learning (MTL) based YOLOv5 can improve detection performance by combining detection and segmentation losses. The YOLOv5s-MTL trained on the damaged building opening dataset obtained 0.648 mAP, an increase of 0.167 from the single-task-based network. Regarding the inference speed, the YOLOv5s-MTL can run inference in 73 frames per second on the tested platform. Overall, this study provides a novelty by contributing to developing an initial damaged building opening dataset and detection model. It also demonstrates that a multi-task learning method can improve detection accuracy.

**Keywords:** damaged building opening, multi-task learning, YOLOv5, object detection, image segmentation

# ACKNOWLEDGEMENTS

I would like to express my deepest appreciation and gratitude to all those who have contributed to completing this thesis. Above all, I would like to take this opportunity to express my profoundly grateful for the trust and confidence the LPDP RI placed in me by awarding me this scholarship. This scholarship was a turning point in my life, opening doors to countless opportunities and enabling me to focus wholeheartedly on my studies.

I would also like to sincerely thank my supervisors and my thesis assessor, Prof. Francesco, Dr. JR, and Prof. Vosselman. Their guidance, expertise, and constant support have been instrumental in shaping this research and ensuring its successful completion. Francesco's expertise in photogrammetry, deep learning, and robotics has undoubtedly improved my skills and interest in those fields. JR's expertise in applying deep learning with geospatial technologies helped me navigate the complexities of the research process. Prof. Vosselman's suggestions gave me a new perspective and helped me spot things for improvement.

Furthermore, I would also like to acknowledge the support provided by Ning Zhang for opening the door and guiding me into the world of computer science. I am grateful for the opportunities I have had to learn from him. I am genuinely unsure if I would have been able to conduct research in the field of deep learning and computer vision without his support.

I am deeply grateful to my family and friends for their unwavering support and understanding throughout this demanding process. Their encouragement, love, and belief in my abilities have constantly motivated and strengthened me. To my father, thank you for always believing in me, and to my mother, thank you for your endless love and prayers. To Archita, Arivia, Aulia, Hafidz, Nuzul, Wildan, Mas Yan, Mas Aufar, Mas Andhika, and Mas Deva, I am glad to have you guys as part of my first time living aboard far away from my family.

Lastly, I would like to honour my sister, Mbak Nilam, by dedicating this thesis to her. I remember how she always cared for me as her little brother since we were kids. I am incredibly fortunate to have her as my sister with all of the struggles we have been through together. She is the one who brings inspiration for me to dream big and start this miles-away journey. Without her, I doubt I would have had the courage and opportunity up to this point in my life.

# TABLE OF CONTENTS

---

1.	INTRODUCTION.....	1
1.1.	Background.....	1
1.2.	Problem Statement.....	3
1.3.	Research Novelty.....	3
1.4.	Research Objectives and Questions.....	4
2.	LITERATURE REVIEW.....	5
2.1.	Convolutional Neural Network.....	5
2.2.	Object Detection.....	5
2.3.	Image Segmentation.....	8
2.4.	Multi-Task Learning.....	11
2.5.	Existing Method of Building Opening Detection.....	12
3.	METHODOLOGY.....	14
3.1.	Dataset.....	14
3.2.	Research Workflow.....	14
3.3.	Network Training Settings.....	22
3.4.	Evaluation Metrics.....	22
4.	RESULTS.....	23
4.1.	Damaged Building Opening Dataset Development.....	23
4.2.	Single Task Object Detection.....	26
4.3.	Multi-Task Learning.....	30
4.4.	Model Performance Comparison.....	35
5.	DISCUSSION.....	37
5.1.	Quality of Dataset for Damaged Building Opening Detection Model.....	37
5.2.	Single-Task Model Performance on Damaged Building Opening Detection Task.....	38
5.3.	Multi-Task Model Performance on Damaged Building Opening Detection Task.....	39
5.4.	Limitations.....	40
6.	CONCLUSION AND RECOMMENDATION.....	41
6.1.	Conclusion.....	41
6.2.	Recommendation.....	42

# LIST OF FIGURES

---

Figure 2.1. CNN Architecture by LeCun et al. (1998) .....	5
Figure 2.2. Stages of R-CNN.....	6
Figure 2.3. The architecture of SPP-net .....	7
Figure 2.4. YOLOv1 Architecture.....	8
Figure 2.5. The Architecture of Deconvolutional Segmentation Networks.....	9
Figure 2.6. SegNet Architecture.....	10
Figure 2.7. Mask R-CNN Architecture.....	10
Figure 2.8. Insta-YOLO architecture based on YOLOv3.....	11
Figure 2.9. The architecture of hard parameter sharing (left) and soft parameter sharing (right).....	11
Figure 2.10. Door detection and opening status classification model by Ramoa et al. (2021) .....	13
Figure 3.1. Damaged Building Images Samples .....	14
Figure 3.2. Dataset Development Workflow.....	15
Figure 3.3. YOLOv5 Object Detection Model Architecture (Jocher et al, 2022) .....	17
Figure 3.4. YOLOv5 Performances Compared to EfficientDet Model Trained on COCO Dataset.....	19
Figure 3.5. YOLOv5-MTL Network Architecture (modified from Jocher et al. (2022)).....	20
Figure 4.1. Type of openings: damaged walls, roofs, open doors and windows (from left to right).....	23
Figure 4.2. Sample Images with Annotations .....	24
Figure 4.3. Horizontal and Vertical Shear 15 degrees .....	25
Figure 4.4. Original Brightness (left), -25% (middle), and +25% Brightness (right).....	25
Figure 4.5. Grayscale, Blur, and Noise Images (Left to Right) .....	26
Figure 4.6. Training-Validation Box Loss (left) and Object Loss (right) .....	27
Figure 4.7. Training Precision and Recall of YOLOv5s.....	28
Figure 4.8. YOLOv5 Mean Average Precision (mAP).....	28
Figure 4.9. Example of Damaged Building Detection on Test Data.....	29
Figure 4.10. False Negative (left image) and False Positive (middle and right images) Detections.....	30
Figure 4.11. Box, Segmentation, and Object Losses .....	32
Figure 4.12. YOLOv5s-MTL Precision, Recall, and mAP .....	32
Figure 4.13. Ground Truth Image and YOLOv5-MTL Detection and Segmentation Prediction Results ..	34
Figure 4.14. YOLOv5-MTL Predictions on Different Types of Damaged Building Opening.....	34
Figure 4.15. Ground Truth Image, Single-task, and Multi-task Output (Left to Right) .....	35
Figure 4.16. Ground Truth Image, Single-task, and Multi-task Output (Left to Right) .....	36



## LIST OF TABLES

---

Table 1. Compound-scales Variations of YOLOv5 versions.....	18
Table 2. CSP structure of YOLOv5 versions .....	18
Table 3. The number of neurons in YOLOv5 versions.....	18
Table 4. Performance Comparison of YOLOv5 Detection Models.....	29
Table 5. Performance Comparison of YOLOv5 MTL Models .....	33
Table 6. Detection Performance Comparison.....	35

## LIST OF ABBREVIATIONS

---

AP	:	Average Precision
CNN	:	Convolutional Neural Network
CSP	:	Cross Stage Partial
DPM	:	Deformable Part-based Model
FCN	:	Fully Convolutional Network
FLOPS	:	Floating Point Operations
FN	:	False Negative
FP	:	False Positive
FPN	:	Feature Pyramid Network
FPS	:	Frame per Second
GSD	:	Ground Sampling Distance
HOG	:	Histogram Oriented Gradients
IoU	:	Intersection over Union
LiDAR	:	Light Detection and Ranging
mAP	:	Mean Average Precision
MTL	:	Multi-task Learning
PAN	:	Path Aggregation Network
PR	:	Precision-Recall
RGB	:	Red Green Blue
SGD	:	Stochastic Gradient Descent
SPM	:	Spatial Pyramid Matching
SPP	:	Spatial Pyramid Pooling
TP	:	True Positive
UAV	:	Unmanned Aerial Vehicles
USAR	:	Urban Search and Rescue
YOLOl	:	YOLO large
YOLO <sub>m</sub>	:	YOLO medium
YOLO <sub>s</sub>	:	YOLO small

# 1. INTRODUCTION

## 1.1. Background

Disaster is a destructive phenomenon that can happen anytime and anywhere. According to the Emergency-event Database (EMDAT), in 2021, 432 disaster events occurred worldwide, and more than 100 million people were affected, causing around USD 252.1 billion in economic losses (EMDAT, 2022). In the disaster management cycle, disaster response is one component carried out immediately after a disaster (Carter, 2008). Search and rescue are the first things done in the disaster response process. Search and rescue missions in urban areas or Urban Search and Rescue (USAR) are time-critical and dangerous activities (Mittal et al., 2019). In urban disasters like earthquakes, it aims to save people trapped in the rubble of buildings, isolated due to collapsed buildings or other conditions.

Efforts currently being made in USAR are by inspecting the affected buildings and continuing to search for victims, most of which is still done manually by the disaster response team. This effort is considered less effective and dangerous, so other alternatives are needed. Due to the challenging conditions and the high risks rescue workers face, it is necessary to apply relevant technologies for post-disaster scenarios (Delmerico et al., 2019). The idea is to obtain the search and rescue mission done with less human involvement and reduce additional risks. Automated tools are currently being developed to lessen the requirement for human involvement in the USAR operation, one of which is the application of Unmanned Aerial Vehicles (UAV). UAVs can be applied as a tool that can replace human involvement, one of which is in search and rescue missions (Gomez & Purdie, 2016).

UAVs can be used for area exploration and locating victims (Goian et al., 2019). The development of UAVs that allow the application of the concept of semi-controlled pilot systems has been widely implemented as a disaster management and response tool. Nex et al. (2019) have researched the development of a UAV that can automatically map and detect collapsed buildings in near real-time timescales using a deep learning algorithm. This technology can help the USAR mission better locate the impacted area due to catastrophic disasters like earthquakes. A UAV search and rescue mission scenario was developed by Zhang et al. (2022) to train a detection network based on harmonious composite images for a low-light indoor environment. The detection network is helpful for the USAR scenario, especially for indoor search and rescue missions.

Mittal et al. (2019) have studied UAVs that can navigate and land autonomously using vision technology for search and rescue due to collapsed buildings. The combination of stereo-cameras and bio-radar was used to detect a safe landing area. This automatic system is crucial because most USAR UAVs still need to be manually remote-controlled, thus requiring a highly skilled operator (Khatib & Siciliano, 2008). With this

vision-based UAV, the UAV system can operate and navigate automatically. However, the technology developed by Mittal et al. (2019) is still limited to automation only to be able to land outside of the collapsed building. Other (or most USAR) use cases require a UAV to enter the building to detect trapped people inside the damaged buildings.

Most autonomous UAVs can only work outside or inside the building, but the capability to navigate in the entering process is still underdeveloped. The difficulty in entering the collapsed building lies in the structure of the building, which is not easily identified which areas can be entered by UAV. A technology that can map the openings in the damaged building is needed. With the ability to detect openings in collapsed structures, UAVs can enter buildings automatically without the need to be manually teleoperated. This ability is required to help the UAV transition between outdoor and indoor environments and vice-versa. Thus, a safer and faster search and rescue process can be done.

The combination of a depth camera, light detection and ranging (LiDAR), and a priori information developed by Pritzl et al. (2021) has been state-of-the-art for autonomous UAVs. This technology is applied in firefighting scenarios that enable UAVs to recognize and estimate the window's location, orientation, and edges to enter a building autonomously. Their approach needs at least one combination between depth camera plus LiDAR or LiDAR with a priori information to support the automatic navigation for UAVs. In some cases, the UAV equipped with a depth camera and LiDAR are unavailable due to cost and other limitations. Thus, the possibility of only using an RGB camera sensor for UAV navigation will be studied in this research.

Ramôa et al. (2021) studied three approaches to detecting and classifying door openings using RGB and depth information. The developed algorithm is used to classify a door into different opening classes, i.e., open, semi-open, and closed. This algorithm successfully classifies doors in real time on a limited-power device and can be applied for indoor navigation purposes to transit between rooms. The network is based on a sequential method that combines object detection or semantic segmentation with a classification network. However, the algorithm developed by Ramôa et al. (2021) was not trained to detect the door opening in the damaged building scenario. The performance of the opening detection can be different due to the unideal condition of doors in damaged building scenes. Also, compared to the current state of the art of real-time object detection networks, the inference time of this door detection and classification network is still lagging behind.

This research will try to overcome the above problem by proposing a detection algorithm for opening space in damaged buildings. Opening space in the damaged building can be damaged walls, holes, opened windows or doors. All these kinds of openings can be the access point for the UAV into the building. To the best of our knowledge, no deep learning model was developed specifically for detecting the opening space in damaged buildings. Therefore, there is a need to design a deep learning model to detect opening space in damaged buildings and try to implement this real-time algorithm for UAV navigation. This research aims to

apply a method for real-time object detection on aerial images based on Convolutional Neural Network (CNN) to detect openings in buildings affected by disasters. The opening information gained from UAV can be used by the USAR team to enter the collapsed building.

The proposed solution in this research is to develop a multi-task learning-based network by combining object detection and segmentation task. The idea is to utilize the segmentation task to improve the detection performance by classifying pixels of the opening space. Contributions that will be given through the results of this research include (1) the preparation and annotation of a collapsed building opening dataset; (2) the development of a real-time object detection model on aerial images to detect openings in disaster-affected buildings; and (3) a modified combination of object detection and segmentation based on multi-task learning network algorithm to detect openings in the damaged building. With the results of this research, it is hoped that this study can be applied to UAV devices that are useful in the search and rescue mission.

## **1.2. Problem Statement**

The capability of an autonomous UAV to enter the building for the Urban Search and Rescue (USAR) mission is crucial. The difficulty in entering a collapsed building lies in the structure of the building, which is not easily identified which areas can be entered by UAV. Thus, a technology that can map the openings in the damaged building is needed. With the ability to detect openings in collapsed structures, UAVs can enter buildings automatically without the need to be manually teleoperated. This ability is required to help the UAV transition between outdoor and indoor environments.

Opening detection is used to detect and locate the opening space on the building façade. The detection network must be able to work in real-time on low-powered devices. Although there have been several works of research in the field of autonomous UAVs for search and rescue missions, no single study exists which develop real-time opening detection in damaged building scene. There is also a need to provide a damaged building opening dataset which is not available yet.

This work will only focus on damaged building opening detection using a computer vision algorithm without discussing the after-process the UAV will take regarding the detection result. The proposed scenario can also be used to inform the USAR team where the opening can be entered.

## **1.3. Research Novelty**

This research's main novelty is developing a deep learning model for real-time damaged building opening detection. The opening detection in damaged buildings is essential for an autonomous urban search and rescue mission system. In the case of UAVs for USAR, this model can help the autonomous system to map and locate the possible openings as the entering space for the UAV. To the best of our knowledge, there is still no detection model for damaged building opening detections.

Initially, there is still no available dataset for this specific purpose. Thus, as part of this study's contributions, a damaged building opening detection dataset consisting of damaged building images and the corresponding openings annotation is developed. A comprehensive comparison study on multi-task learning and single-task-based detection model is also conducted. The proposed multi-task-based deep learning model will be examined to improve the performance of a single-task damaged building opening detection model. This comparison study between these two deep learning model techniques is necessary to prove the initial hypotheses. Furthermore, the findings of this study will contribute to the scientific understanding of deep learning, especially in scene understanding and object detection applications.

#### **1.4. Research Objectives and Questions**

The main objective of this study is to develop a deep learning model for real-time detection of the opening space in damaged buildings. This research will use deep learning to study the combination of object detection and segmentation algorithm based on multi-task learning methods. The main objective can be divided into sub-objectives with the following research questions.

**1.4.1.** Develop a damaged building opening dataset for real-time opening detection task.

- What is the available dataset that can be used for building opening detection?
- What is the procedure for developing a damaged building opening dataset?
- What are the augmentation methods to improve the damaged building opening dataset?

**1.4.2.** Develop a single-task learning based object detection model for real-time damaged building opening detection.

- What is the architecture of the single-task learning model for damaged building opening detection?
- How is the performance of a single-task learning model for damaged building opening detection?
- How different single-task model configurations can affect the detection performance?

**1.4.3.** Develop a multi-task learning model for real-time detection of the opening space in the damaged building.

- What is the architecture of the multi-task learning model for damaged building opening detection?
- How is the performance of the multi-task learning model for damaged building opening detection?
- How different multi-task learning model configurations can affect the detection performance?

**1.4.4.** Assess the deep learning model performance on real-time damaged building opening detection test scenarios.

- How are the results from the single-task model compared to those from the multi-task model for real-time damaged building opening detection?
- What makes the multi-task model perform differently compared to the single-task detector?

## 2. LITERATURE REVIEW

### 2.1. Convolutional Neural Network

Convolutional Neural Network or CNN is the most widely used architecture in deep learning for computers. The idea of CNN was first introduced by Fukushima (1980) with the basis of the hierarchical receptive field model. Further study by LeCun et al. (1998) developed a CNN-based model for image recognition. Figure 2.1 shows the architecture of CNN used for document recognition.

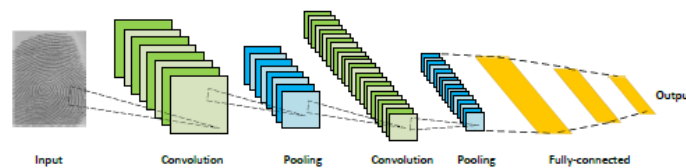


Figure 2.1. CNN Architecture by LeCun et al. (1998)

CNN is mainly constructed by three types of layers, i.e. convolutional layers, nonlinear layers, and pooling layers. The convolutional layer extracts the features using a kernel or filter with defined weights. Then activation function is applied by the nonlinear layers to enable the modelling of nonlinear functions. Last, using some statistical operations, the pooling layer replaces the small neighbourhood of a feature map (Minaee et al., 2022).

### 2.2. Object Detection

One of the major tasks in computer vision is object detection. The main idea of the object detection algorithm is to develop a computational model to present information about what objects are and where (Zou et al., 2023). An object detection model needs to be able to define where is the location of the object (localization) and which class should be the property of each object (classification). Thus, object detection workflow can be divided into three main stages, i.e. region selection, feature extraction, and object classification (Zhao et al., 2018). Many applications of object detection have been applied in different real-world use cases, such as driverless cars (Ghasemieh & Kashef, 2022), vehicle counting (Tayara et al., 2017), face recognition (Rahouma & Mahfouz, 2021), disease detection (Harakannanavar et al., 2022), etc.

There are two main phases in the development of object detection, traditional detectors and Convolutional Neural Network (CNN) based detectors. Both development phases are described below.

### 2.2.1. Traditional Detectors

The initial development of object detection algorithms was developed based on handcrafted features. This means the feature properties are derived using the information presented in the image. Viola & Jones (2001) developed the first real-time face detection algorithm, known as Viola-Jones Detectors, without concern for image differencing and skin colour detection. The **Viola-Jones Detectors** have massively improved their speed hundreds of times faster than the current state-of-the-art of face detection at that time. This can be achieved by combining several methods: integral image, feature selection, and detection cascades.

Furthermore, the Histogram of Oriented Gradients (**HOG**) feature descriptor was introduced by (Dalal & Triggs, 2005) to solve the problem of detection in multi-size objects. This algorithm was designed for pedestrian detection that rescales the input image multiple times while maintaining the size of the sliding window. The study showed that implementing a locally normalized histogram of gradient orientation features in a high-density overlapping grid can reduce the false positive rates for person detection.

As the further development of HOG, a Deformable Part-based Model (**DPM**) was proposed by (Felzenszwalb et al., 2008). DPM used a general framework for training support vector machine (SVMs) with a latent structure to develop a detection system based on multiscale and deformable models. The development of DPM algorithm becomes the final era of traditional object detectors before the popularization of convolutional neural network-based object detection.

### 2.2.2. CNN-based Detectors

With the emergence of the Convolutional Neural Network (CNN) for image classification introduced by Krizhevsky et al. (2012), further development of CNN-based detectors for object detection is also getting popular. The deeper architecture of a convolutional network enables the model to learn more complex features compared to the traditional methods. The frameworks of CNN-based object detectors can be divided into two categories. First is following the conventional object detection workflow by generating region proposals, continued by classifying every proposal based on the categories of the objects. Second, consider object detection as a regression or classification problem and apply a unified framework to gain the final output: locations and categories.

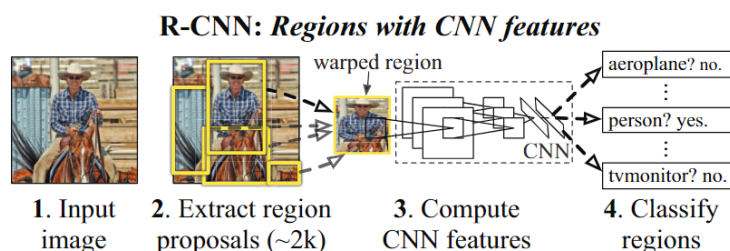


Figure 2.2. Stages of R-CNN



The region proposal-based techniques, also known as two-stage object detectors, mimic the attentional mechanism of the human brain in identifying and locating objects. This algorithm scans the whole image first and then focuses on the region of interest. **R-CNN** (Girshick et al., 2014) was developed based on a two-stage object detector, with the flowchart shown in Figure 2.2.

The workflow of R-CNN can be divided into three stages, (1) region proposal generation; (2) CNN deep feature extraction; (3) classification and localization.

One of the disadvantages of R-CNN is that it is prone to content losses due to region proposal cropping. To overcome this problem, He et al. (2014) combined the concept of SPM or Spatial Pyramid Matching (Lazebnik et al., 2006) into a new CNN architecture called **SPP-net**. SPM handles finer to coarser scales for dividing and aggregating local into mid-level features. SPP-net utilizes feature maps of the fifth convolution layer to project region proposals with the architecture shown in Figure 2.3. The final convolution layer is then defined as spatial pyramid pooling (SPP) layer. SPP-net obtains better detection and improves detection efficiency by sharing computation costs among different proposals.

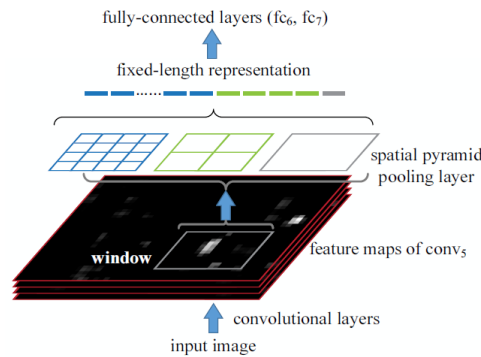


Figure 2.3. The architecture of SPP-net

**Fast R-CNN** (Girshick, 2015) was developed to improve the detection capability from the previous detection network. This network introduced simultaneous training on classification and bounding box regression. Fast R-CNN can obtain better mean average precision at 200 times faster than R-CNN. But the performance of Fast R-CNN still depends on proposal detection. Thus, **Faster R-CNN** (Ren et al., 2015) introduced a nearly cost-free region proposal based on Region Proposal Network (RPN). RPN is based on a fully-convolutional network with the ability to predict object boundaries and positions at the same time. The detection algorithm performed by Faster R-CNN was counted as the first near-real-time detection network with a frame rate of 5 frames per second on a GPU. But, this network is very time-consuming for the training process and not robust in dealing with different scales and shapes of objects.

Time spent to deal with different components in two stages detectors become the bottleneck in real-time application. Thus, single-stage detectors based on global regression or classification can decrease the processing time by mapping directly from image pixels to bounding boxes and class probabilities.

**YOLO**, or You Only Look Once (Redmon et al., 2015) was the first one-stage detector network. The idea of YOLO is to divide the input image into grids and predict the centre object in every grid cell. The network will predict each grid's bounding boxes and their corresponding probabilities. YOLO can run the detection in real-time with the simplified version of YOLO can reach 155 FPS with better performance results. However, YOLO has limitations in dealing with a group of small objects caused by the spatial constraint on bounding box prediction.

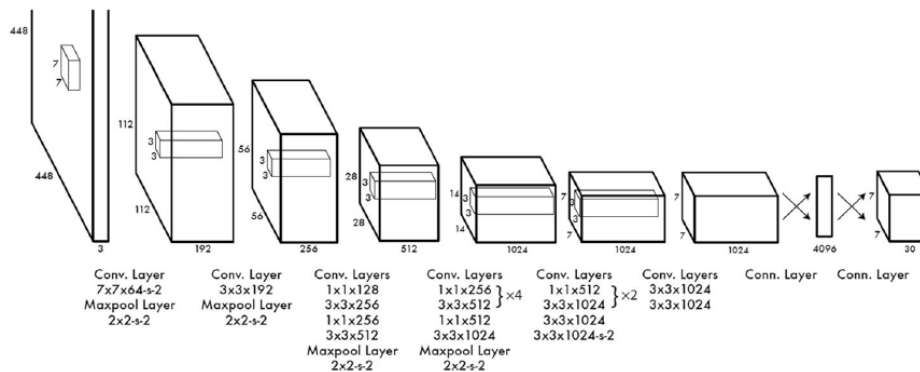


Figure 2.4. YOLOv1 Architecture

Figure 2.4 shows the architecture of YOLOv1 (Redmon et al., 2015). YOLOv1 consist of 24 convolution layers and two fully connected layers. The last layer of this network predicts the object class probability and bounding box coordinates. Compared to two-stage object detection, YOLOv1 has lower recall and large localization error that become the main focus of the current development of one-stage object detection.

To solve this problem, a Single Shot MultiBox Detector or **SSD** was proposed by Liu et al. (2016). The application of multi-reference and multi-detection methods in SSD increases the detection accuracy of a one-stage detector. Aiming to handle objects in different sizes, SSD fuses predictions from multiple feature maps with various resolutions. Recently, the development of YOLO successors has shown that YOLOs are performing better in accuracy and inference speed (Diwan et al., 2022).

### 2.3. Image Segmentation

Image segmentation is part of the scene understanding tasks dealing with pixel-based classification. The main task of image segmentation is to divide the image into multiple segments of objects (Abdulateef & Salman, 2021). There are two types of image segmentation: semantic segmentation and instance segmentation. Image segmentation that works on pixel labelling for a set of object classes is called semantic segmentation. Instance segmentation is the extension of semantic segmentation that labels each object of interest in the image (Minaee et al., 2020). Over decades, different image segmentation methods have been developed, such as thresholding (Otsu, 1979), graph cut (Boykov et al., 2001), region-growing (Nock & Nielsen, 2004), k-means clustering (Dhanachandra et al., 2015), etc.

With the emergence of deep learning (DL), the performance of image segmentation methods also improved. Based on the primary technique and the architecture, several DL-based image segmentation can be categorized as follows.

### 2.3.1. Fully Convolutional Networks

A fully convolutional network, or FCN (Long et al., 2015), is the first DL technique implemented for image segmentation tasks. FCN uses convolutional layers that can produce a segmentation map of an image of the same size. The main concept of the FCN is to replace fully-connected layers in CNN with fully-convolutional layers that enable the network to create a spatial segmentation map instead of only classification scores. FCN has become the milestone of image segmentation based on DL and has been implemented in further research, such as W-Net (Xia & Kulis, 2017) and dual path adversarial learning networks based on FCN (Bi et al., 2018). However, FCN still has several drawbacks, i.e., it is inefficient for real-time inference and is ineffective in handling global context information.

### 2.3.2. Encoder-decoder based Models

Noh et al. (2015) introduce an image segmentation model that consists of two parts, an encoder and a decoder. The encoder used convolution layers from VGG16, and the decoder consisted of deconvolution and unpooling layers. The architecture of this network can be seen in Figure 2.5.

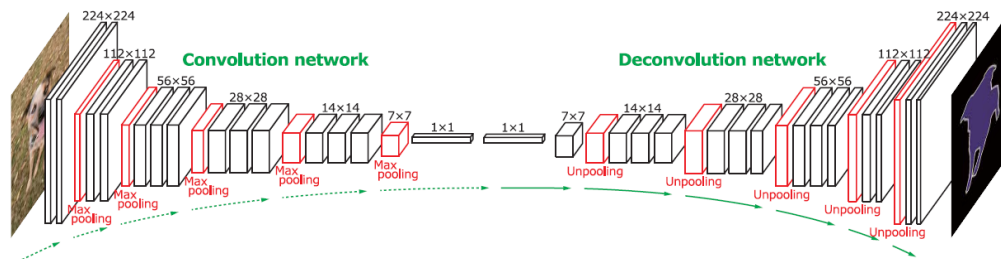


Figure 2.5. The Architecture of Deconvolutional Segmentation Networks

Another work conducted by Badrinarayanan et al. (2017) designed an encoder-decoder network for image segmentation called SegNet. As shown in Figure 2.6, SegNet consists of 13 VGG16 convolutional layers as the encoder and corresponding decoder networks. SegNet proposed a new technique for upsampling from its lower-resolution feature maps by using pooling indices computed from the max-pooling step to do nonlinear upsampling. This makes up-sample learning unnecessary. Thus SegNet is smaller in the number of training parameters compared to other models.

The Loss of high-resolution representation through the encoding process can also cause information loss which becomes the main drawback of SegNet. Also, when unpooling from low-resolution feature maps, surrounding information will likely be lost (Yamanakkanavar & Lee, 2021).

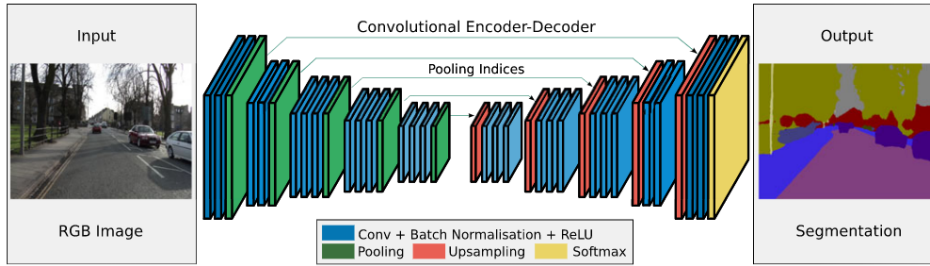


Figure 2.6. SegNet Architecture

### 2.3.3. R-CNN based Segmentation Models

Several studies in the computer vision domain utilize the ability of R-CNN (Girshick et al., 2014) to perform both object detection and image segmentation simultaneously. One is Mask R-CNN (He et al., 2017), an object detection model that simultaneously generates a segmentation mask for each object. This network is based on a Faster R-CNN (Ren et al., 2015) that has been modified to obtain three outputs, i.e. bounding box, object class, and segmentation mask. Another segmentation network that uses R-CNN as the basis is the Path Aggregation Network or PAN (Liu et al., 2018). This network is based on Mask R-CNN and Feature Pyramid Network (FPN) models. FPN is used as the feature extractor with a new augmented bottom-up pathway. As shown in Figure 2.7, the first two fully connected layers generate the object class and bounding box, while the third processes the region of interest to predict the object mask using FCN.

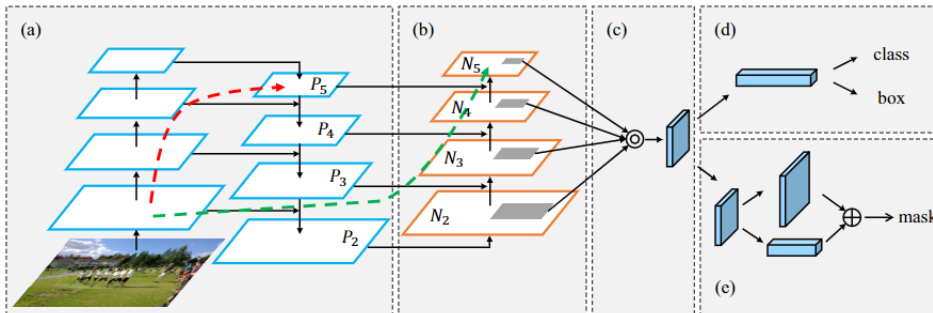


Figure 2.7. Mask R-CNN Architecture

### 2.3.4. YOLO based Segmentation Models

The segmentation model that builds based on a two-stage detection pipeline suffers from several weaknesses, i.e. computational complexity, slow processing, and does not fit tasks with oriented boxes like in remote sensing applications. Thus, Insta-YOLO (Mohamed et al., 2021), a deep-learning model for real-time instance segmentation, was proposed to solve these problems. Insta-YOLO used YOLOv3 as the backbone with the architecture shown in Figure 2.8. This model can run in real-time at 35 frames per second (FPS) with an average precision of 78.16%.

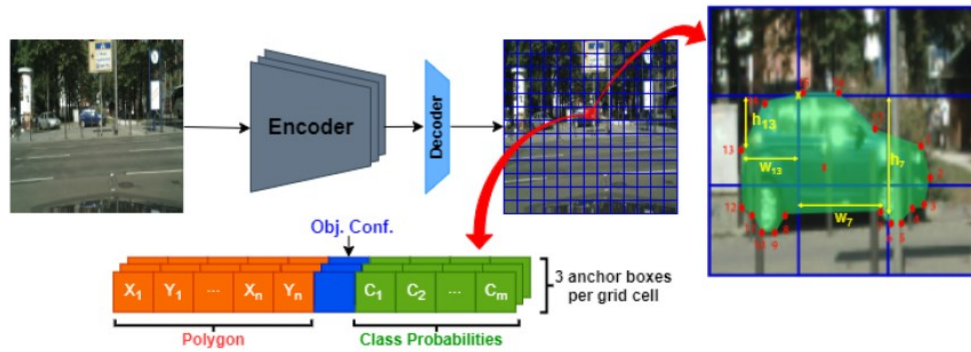


Figure 2.8. Insta-YOLO architecture based on YOLOv3

Current development of YOLO successors, i.e. YOLOv5 (Jocher et al., 2022) and YOLOv7 (Wang et al., 2022), are also equipped to work in image segmentation tasks by utilizing one-stage object detection as the basic model.

## 2.4. Multi-Task Learning

Multi-task learning is a training method for machine learning models that combine multiple tasks simultaneously (Crawshaw, 2020). The shared network in multi-task learning can increase the model's efficiency and learning speed. These characteristics of multi-task learning will benefit the designed opening detection model to work in a real-time and low-powered device. In the case of this study, the concept of multi-task learning can be applied to jointly trained semantic segmentation and object detection. Simultaneously learning both tasks could benefit the network in terms of speed, feature sharing, and simplicity for training (Dvornik et al., 2017).

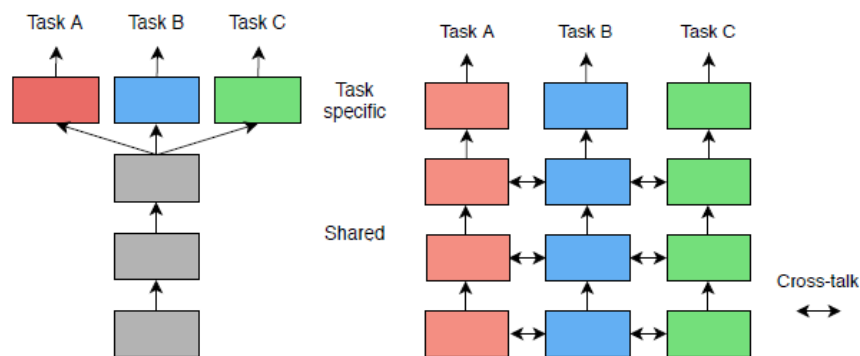


Figure 2.9. The architecture of hard parameter sharing (left) and soft parameter sharing (right)

Multi-task learning was classified into hard and soft parameter-sharing methods. Shared and task-specific parameters construct hard parameter-sharing architecture. While in soft parameter-sharing, every task is assigned its parameters, and the mechanism of feature sharing is by cross-task talk, as shown in Figure 2.9 (Vandenhende et al., 2020). This study will apply the concept of hard parameter-sharing by designing a

network that consists of a shared encoder and branches out into task-specific decoding heads. A shared encoder followed by segmentation and object detection heads will be the main idea of the proposed network.

Based on the study conducted by Balamuralidhar et al. (2021), semantic segmentation has been shown to strengthen object detection capability by improving category recognition, location accuracy, and context embedding. Improving category recognition means that the ability of semantic segmentation to delineate object boundaries could help differentiate the object's category captured in the image. Also, by encoding these boundaries very well, semantic segmentation can enable the detection to improve localization accuracy. In terms of scene understanding context, object and surrounding backgrounds pattern that are learned in semantic segmentation can improve the confidence accuracy of object detection.

The multi-task learning method was applied in MultiNet (Teichmann et al., 2018) by using a similar feature map in an encoder network for object detection and semantic segmentation. BlitzNet (Dvornik et al., 2017) integrate semantic segmentation and object detection by sharing the network weights until the last layer. There are two main techniques for jointly trained segmentation to improve detections. The first is by using segmentation to extract fixed features and then integrate them into the detection branch. But the weakness of this technique is the high computation cost due to executing both segmentation and detection during inference.

The second training technique is implementing a segmentation head on top of the detection head. This method overcomes the previous weakness because the computation to generate the segmentation map is unnecessary. Thus the detection speed will not be affected. The search space for parameters in the backbone also can be reduced when the task is related or similar. Several studies implementing this method, such as StuffNet (Brahmbhatt et al., 2016) and Mask R-CNN (He et al., 2017), are examples of multi-task learning that could improve object detection.

## **2.5. Existing Method of Building Opening Detection**

A sequential-based detection and classification model was studied by Ramôa et al. (2021) for real-time door detection and opening status classification. The sequential approach runs the task separately and in sequence order. This method uses object detection and segmentation to detect and segment doors, then continue by cropping the images. The cropped images will be used as input to be classified as open, semi-open, or closed doors. This model adopted BiSeNet (Yu et al., 2018) to segment the door and AlexNet to classify the opening status of the doors. Figure 2.10 shows the workflow of the door detection and opening status classification network.

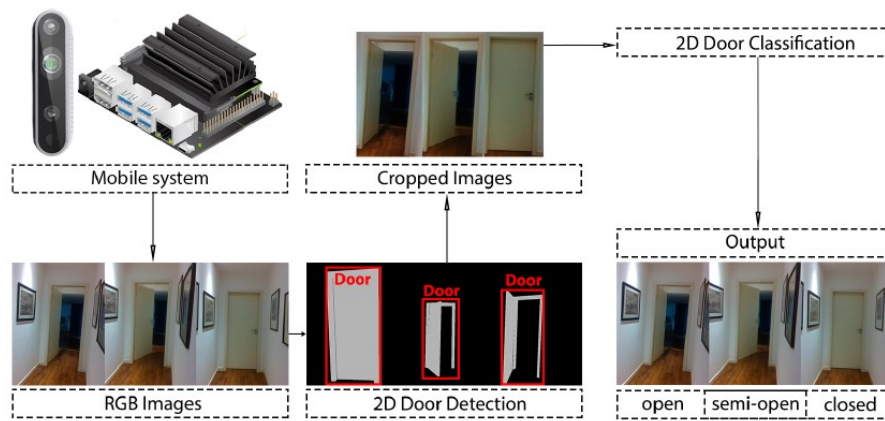


Figure 2.10. Door detection and opening status classification model by Ramoa et al. (2021)

This sequential method shows the capability of a deep learning model to detect building openings, i.e. doors opening status. However, this method still needs to be improved in terms of speed performance. The sequential procedure is inefficient regarding the inference time. The total time of the model is around 1-2 frames per second which are relatively slow. Also, this model is not suitable for the post-disaster opening detection scenario because it is only trained to detect doors in normal buildings (undamaged). No existing network still focuses on detecting the opening of damaged buildings. Thus, developing a damaged building opening detection model is needed and will be developed in this study.

## 3. METHODOLOGY

### 3.1. Dataset

This study used the RGB images dataset containing a scene of a damaged building obtained from previous work by Chachra et al. (2022) as the primary data to train the object detection model. The dataset contains images of earthquake events in Indonesia and Nepal with different scenes, such as damaged buildings, roads, bridges, and other collapsed structures. The 2579 images provided in this dataset were acquired from crowdsourcing through Twitter and Getty Images platforms. All images from Twitter are subject to the policy of free use for non-commercial usage, and the images from Getty Images are only used for training purposes. Figure 3.1 shows the sample of damaged building images used in this study.



Figure 3.1. Damaged Building Images Samples

Not all of the images in this dataset can be used for this study because we only focus on opening in damaged buildings. Thus, image selection is needed to select images containing the opening scene in damaged buildings. A total of 738 images are selected and divided into training, validation, and testing datasets with the composition 70%, 20%, and 10%, respectively. The size of the images is 640x407 pixels and will be resized to 640x640 pixels to get the square dimension of the input images for model training purposes. Because this data only contains RGB images, the annotation process must be conducted with the detailed workflow in section 3.2.1.

### 3.2. Research Workflow

This research mainly consisted of four phases: dataset development, single-task network training, multi-task learning network design and training, and model performance evaluation. As part of the contribution of this study, the damaged building opening dataset is developed from the available open-source image dataset. The dataset development consists of image collection, selection, and annotation process. Two deep learning-based detection model approaches are studied to provide the best detection model for real-time damaged building opening detection.



First, a single-task detection network is trained on the dataset and evaluated to obtain the network's performance. Second, a multi-task learning network is designed by adding a segmentation head to the detection network architecture. The segmentation task in the multi-task learning approach is expected to improve the detection performance. Both detection network approaches will be compared in terms of accuracy and inference speed. YOLOv5 is used as the base network in building the damaged building opening detection network. The following sections provide a more detailed description of each stage, including the justifications and workflow.

### 3.2.1. Dataset Development

The supervised learning method in the deep learning model requires training, validation, and testing dataset. This section explains the dataset development workflow to provide a damaged building opening dataset for this study. To the best of our knowledge, no available dataset contains annotation and images for damaged building opening detection. Thus, this study will provide this dataset. Figure 3.2 shows the workflow of the damaged building opening dataset.

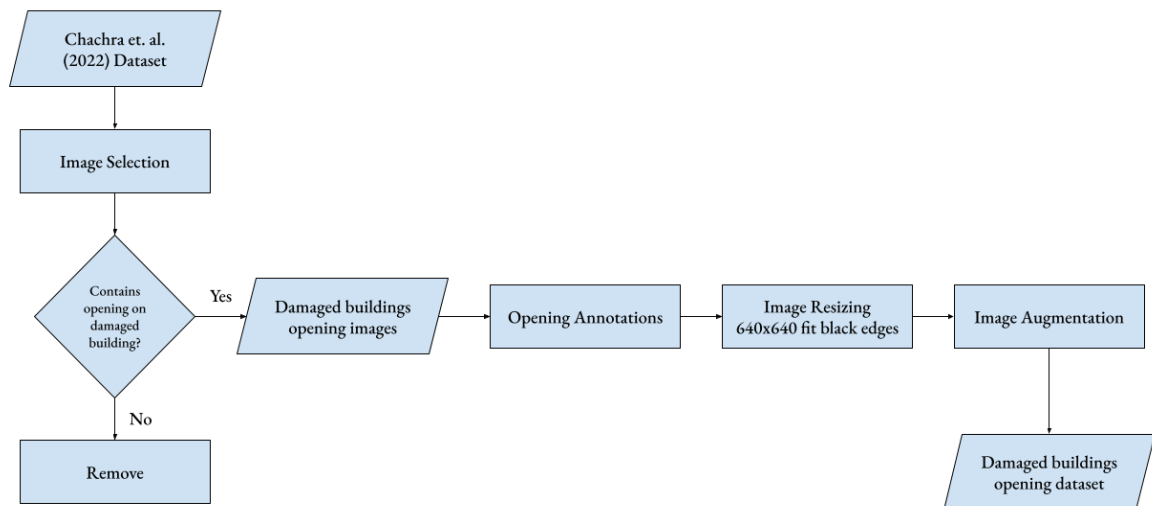


Figure 3.2. Dataset Development Workflow

The damaged buildings opening dataset development is built upon the previous Chachra et al. (2022) dataset. The dataset containing a different scene of earthquake events will then be manually filtered to select only images that have openings on damaged buildings. Other scenes of damaged roads and bridges or collapsed structures that do not contain building openings are removed. The set of damaged building opening images is obtained from this selection process.

The following procedure is the image annotation process by annotating the opening on the damaged buildings. The opening, in this case, means all the opening space on the damaged building that can be an access for UAV to enter the building. Because this study only focuses on developing the opening detection model, so there is only one class, i.e. opening. The image annotation process follows the procedure of

segmentation annotation by annotating objects such as open doors or windows, damaged roofs, and wall holes as the opening class. The annotation tool used in this study is Labelme due to its practicality and complete features.

The images are then converted into 640x640 pixels. The size conversion is done without affecting the aspect ratio by implementing image fits with black edges. This method resizes the image to the desired dimension without changing the image's aspect ratio and filling the empty pixels with black colour. Once all the data is the same size, dataset augmentation techniques are used to improve the model generalizing ability. The following image augmentation techniques are applied to prepare the dataset for training:

- Horizontal flip: to improve the training image variety, a horizontal flip is applied by rotating the image on the vertical axis.
- Random brightness: an arbitrary pixel intensity level changed across the images from -25% to +25% of the original brightness level.
- Probabilistic grayscale: 25% of the training images are randomly chosen and converted to grayscale.
- Noise: apply 1% noise to the images to improve the model's resilience to camera artefacts.
- Random blur: randomly apply 1 pixel of Gaussian blur to the training images.
- Mosaic: combine several images to improve the model in handling small objects problems.
- Shear: add variability to perspective to help the model be more resilient to the camera, subject pitch, and yaw. This is done by applying 15 degrees of vertical and horizontal shear.

Through all these processes, the damaged building opening dataset is obtained and ready to be used for training, validation, and testing of the model.

### **3.2.2. Single-task Detection Network**

Conventional object detection networks are based on a single-task approach. Thus it is necessary also to train the single-task model to obtain the benchmark for the proposed model approach (multi-task learning). In this phase, YOLOv5 (Jocher et al., 2022) model is trained on the damaged building opening dataset and evaluated to get the model performance. YOLOv5 provides four model architecture variants with different model sizes, i.e. YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x (small, medium, large, and extra large, respectively). All the models are available with pre-trained model weights trained on the COCO dataset. YOLOv5 small, medium, and large detection models were trained on the damaged building opening dataset and compared to get the best model performance in terms of accuracy and speed. The details of a YOLOv5 model architecture are as follows.

#### **Model Architecture**

YOLOv5 is a one-stage detector currently one of the most stable and well-performing real-time object detection models. This network is pre-trained on the COCO dataset with the fastest detection speed of up

to 140 frames per second (Yan et al., 2021). YOLOv5 model architecture consists of several layers: the backbone, neck, and head. Figure 3.3 shows the detailed architecture of the YOLOv5 single-task detector by Jocher et al. (2022).

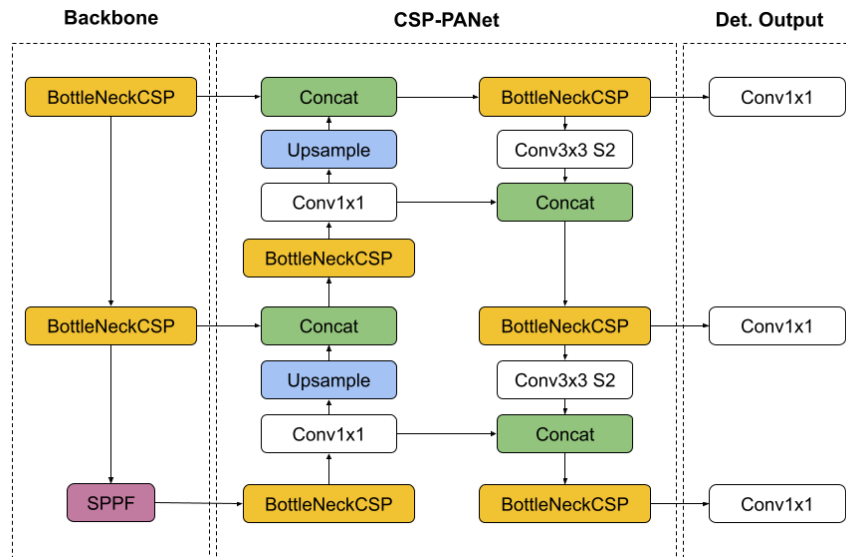


Figure 3.3. YOLOv5 Object Detection Model Architecture (Jocher et al., 2022)

### Backbone

In a deep learning model, convolutional networks that extract features are called backbones (Elharrouss et al., 2022). The backbone used in YOLOv5 version 6.0/6.1 is a New CSP-Darknet53. This backbone is constructed based on Cross Stage Partial Network or CSPNet (C. Y. Wang et al., 2020) and Darknet53 (Redmon & Farhadi, 2018). The combination of CSP-Darknet53 is done by reducing and replacing the convolutional residual blocks of Darknet53 with BottleneckCSP modules. This backbone is shown as the optimal network for a detector by providing a large receptive field size and parameter number (Bochkovskiy et al., 2020).

### Neck and Detection Head

YOLOv5 use SPPF and New CSP-PAN as a neck to combine different scales of features extracted from different levels of the backbone. SPPF is a modified version of Spatial Pyramid Pooling or SPP (He et al., 2015) with fewer floating point operations (FLOPs). The CSP-PAN network is used to aggregate the features. The detection head will generate detections from these multi-level aggregated features. The complete overview of YOLOv5 architecture is shown in Figure 3.3.

Different versions of YOLOv5 use the same network architecture as in Figure 3.3 but apply different depth and width scales. Table 1 shows the compound scales variations in YOLOv5 versions. The depth scale is a variable that defines the number of layers of a network. While the width scale is a variable that determines the size of the output channel of the network's hidden layers. These two variables are varied in the different

versions of YOLOv5. The higher network's depth and width scales, the more complex the model is. These versions provide the model configuration options to find the most optimal model for specific applications.

Table 1. Compound-scales Variations of YOLOv5 versions

Model	Depth Scale	Width Scale
YOLOv5s	0.33	0.5
YOLOv5m	0.67	0.75
YOLOv5l	1	1

The depth scale variations will vary the number of layers in a YOLOv5 network. The depth of the CSP structure in every YOLOv5 model variation is different. The value of the depth scale is used to define the structure of CSP layers. The width scale determines the output channel size of the network layer. Thus, the thickness at each layer will be affected and influence the learning ability of the feature extractor layers. Chao et al. (2022) calculated and compared the effect of different depths and width scales on the various versions of the YOLOv5 networks configuration, as shown in Table 2 and Table 3.

Table 2. CSP structure of YOLOv5 versions (Chao et al., 2022)

	YOLOv5s	YOLOv5m	YOLOv5l
Depth scale	0.33	0.67	1
The first CSP1	CSP1_1	CSP1_2	CSP1_3
The second CSP1	CSP1_2	CSP1_4	CSP1_6
The third CSP1	CSP1_3	CSP1_6	CSP1_9
The first CSP2	CSP2_1	CSP2_2	CSP2_3
The second CSP2	CSP2_1	CSP2_2	CSP2_3
The third CSP2	CSP2_1	CSP2_2	CSP2_3
The fourth CSP2	CSP2_1	CSP2_2	CSP2_3
The fifth CSP2	CSP2_1	CSP2_2	CSP2_3

YOLOv5 uses two types of CSP networks, CSP1 and CSP2, for the backbone and the neck, respectively. Table 2 shows that in YOLOv5s, there is only one residual unit, so it is CSP1\_1. The depth scale in YOLOv5m is increased and results in two residual units, so it is CSP1\_2. For YOLOv5l, there are three residual units, so the first CSP1 is CSP1\_3. This notion also applies to the structure of CSP2; in YOLOv5s, one set of convolutions is used, i.e. CSP2\_1.

Table 3. The number of neurons in YOLOv5 versions (Chao et al., 2022)

	YOLOv5s	YOLOv5m	YOLOv5l
Width scale	0.5	0.75	1
Focus	32	48	64
The first Conv	64	96	128
The second Conv	128	192	256
The third Conv	256	384	512
The fourth Conv	512	768	1024

Different width scales value for the YOLOv5 model versions determines the number of neurons in the convolution layers. As shown in Table 3, YOLOv5s has 32 neurons in the Focus layer. Then, the second neuron changes to 64 due to the width scale. This multiplication continued till the fourth convolution layer. Detailed illustrations are shown in Annex 1. These versions of model configurations make the YOLOv5 model more flexible and scalable. Due to the difference in the model's complexity, different versions will also result in various performances.

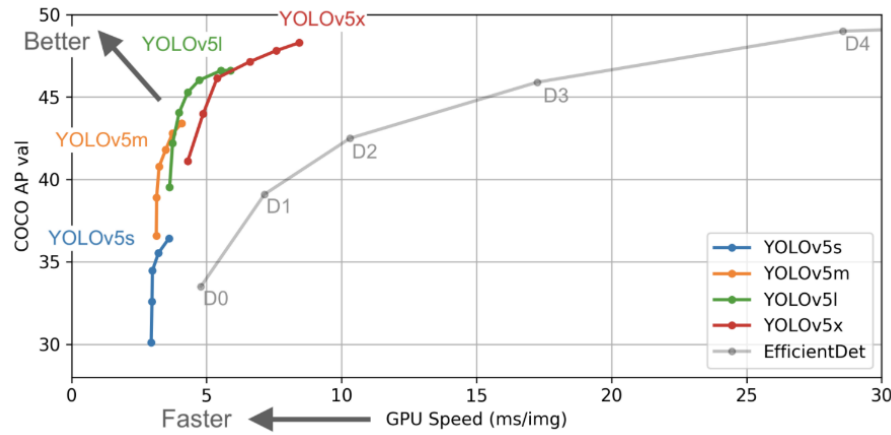


Figure 3.4. YOLOv5 Performances Compared to EfficientDet Model Trained on COCO Dataset

Figure 3.4 illustrates the performance comparison between four versions of YOLOv5 and EfficientDet (Tan et al., 2019) trained and evaluated on the same dataset, i.e. COCO dataset. All the versions of YOLOv5 outperform the EfficientDet regarding inference speed. Increasing the model's complexity will improve the accuracy but sacrifice the inference speed. This study compares the detection performances of three different versions of YOLOv5 for damaged building opening detection to find the optimal model configuration.

### 3.2.3. Multi-task Learning Network Design

This phase focus on applying a multi-task learning (MTL) based network to improve the performance of the real-time damaged building opening detection model. The idea is to use the segmentation task to enhance the performance of the detection model. The hard-parameter sharing method will be implemented to build the architecture of the damaged building opening detection model. This means that the multi-task model will consist of a shared backbone followed by task-specific heads, i.e. detection and segmentation head. This sharing method is beneficial for regularisation and reduces the risk of overfitting.

The multi-task learning is developed based on the YOLOv5 object detection architecture by adding a segmentation branch jointly with the detection head. The small, medium, and large versions of YOLOv5-MTL were trained on the damaged building opening dataset and evaluated using the evaluation metrics to get the model's performance. The details of a YOLOv5-MTL model architecture are as follows.

## Model Architecture

The architecture of YOLOv5-MTL is based on hard parameter sharing by utilizing one backbone with two heads (detection and segmentation). The hard parameter-sharing technique can also benefit the model by reducing the risk of overfitting (Balamuralidhar et al., 2021). The hard parameter-sharing in the MTL network divides the parameter into shared and task-specific parameters. These task-specific parameters are trained separately for each task but use the cumulative loss. The main architecture of YOLOv5-MTL is based on the YOLOv5 detection network with added functionality for image segmentation. The idea of adding the segmentation branch to the last output from the CSP-PANet, as shown in Figure 3.5, is inspired by the Mask R-CNN (He et al., 2017), which is proven as the optimal multi-task architecture of object detection and image segmentation.

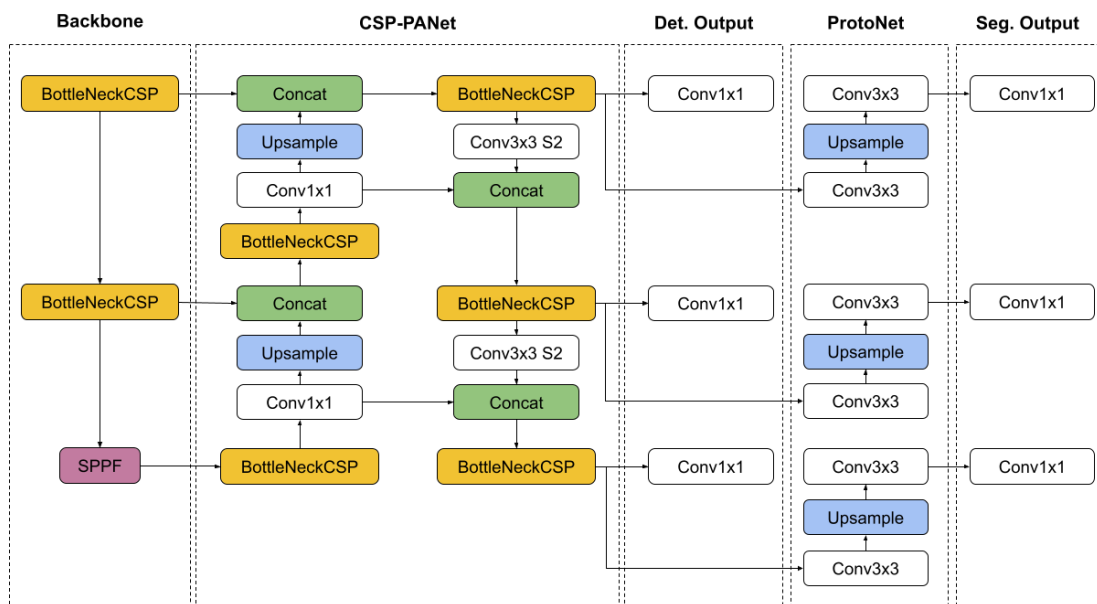


Figure 3.5. YOLOv5-MTL Network Architecture (modified from Jocher et al. (2022))

### Backbone and Neck

The backbone used in this network is the same as in YOLOv5, which is the new CSPDarkNet53. This backbone has shown good performance in the features extraction process to provide information for the object detection algorithm. The extracted features from different levels of backbones are combined using SPPF and New CSP-PAN as the Neck network.

### Detection and Segmentation Heads

YOLOv5-MTL is a modification of the detection architecture by adding the ability to do instance segmentation. ProtoNet (Bolya et al., 2019), a fully connected neural network, is added along with the YOLOv5 object detection head. ProtoNet is constructed by a three-layer network of 2D convolutions with SiLU activation functions to produce prototype masks. This ProtoNet is used in the final instance segmentation head along with the final 2D convolutional layers.

This prototype generation branch is attached to a backbone layer to predict a set of prototype masks for the whole image scene. The ProtoNet that is linked to deeper backbone features can produce better robustness of the mask and good performance on smaller objects (Bolya et al., 2019). This design can benefit the damaged building opening network to detect small openings. The detailed architecture of YOLOv5-MTL can be illustrated in Figure 3.5.

The main difference of the multi-task learning version of YOLOv5 is adding additional segmentation branches based on ProtoNet. This additional network is responsible for predicting the segmentation mask. The combined loss of detection and segmentation losses calculates the loss of this model with a detailed explanation as follows.

### Loss Function

The idea of developing a multi-task learning based detection model in this study is to improve the performance of the detection task by parallelly performing a segmentation task. The losses of each task are combined to represent the model loss. The combined loss is calculated based on objectness loss, class probability loss, bounding box regression loss, and mask loss. The object and bounding box loss are responsible for showing the bounding box score prediction and coordinates, respectively. While the class probability loss is to get the object classification score, and the mask loss is responsible for the segmentation mask prediction. Regression loss for bounding box coordinates prediction is calculated based on mean intersection over union (IoU) ranging from 0-1. Object and class losses are calculated using the binary cross-entropy with logits or BCEWithLogits Loss developed by PyTorch to calculate the losses. BCEWithLogits loss function can be described as:

$$l(x, y) = L = \{l_1, \dots, l_N\}^T, l_n = -\omega_n [y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log(1 - \sigma(x_n))] \quad (1)$$

Where  $x$  is the input,  $y$  is the target,  $\omega$  is the weight,  $N$  represents the batch size, and  $n$  is the sample number in the batch. BCEWithLogits Loss combines a Sigmoid layer with BCE loss in one class to obtain more numerical stability than separating the Sigmoid layer. This loss function resulted in the class probability prediction ranged 0 to 1. Based on this prediction, the model will calculate the difference between the prediction and the ground truth, resulting in a loss value.

#### 3.2.4. Model Performance Comparison

The final phase of this study is to compare the model's performance and conduct hyperparameter optimization. The best model version of single and multi-task learning networks was compared to obtain the best learning approach in developing damaged building opening models. There are two comparison approaches to assess the performance of each model, i.e. based on a quantitative approach by comparing the performance based on the evaluation metrics, and based on visual interpretation.

### 3.3. Network Training Settings

Single and multi-task learning object detection models are trained with the following settings to obtain comparable detection model performances. These setting values are chosen based on the standard YOLOv5 detection model training method.

- Batch Size: 16
- Number of Epochs: 100
- Optimizer: Stochastic Gradient Descent (SGD)
- Loss Function: Binary Cross-Entropy with Logits Loss
- Learning Rate: 0.01
- Training Platform Specification: Google Colab (12GB NVIDIA Tesla T4 GPU)

### 3.4. Evaluation Metrics

The trained network is evaluated by calculating Precision, Recall, and Mean Average Precision (mAP) on the test datasets. These evaluation metrics are measured to obtain the model's performance scores in performing the object detection or image segmentation task. Precision measures the model's performance in getting true positives from all positive predictions. In comparison, recall measures how good the model is at getting true positives from all the ground truth positive. The formula for calculating precision and recall is explained below.

$$Precision = \frac{TP}{TP + FP}; \quad Recall = \frac{TP}{TP + FN} \quad (2)$$

TP stands for true positive, FP is false positive, and FN is false negative. The mAP is the average precision over classes. This average precision (AP) is given by the area under the precision-recall curve for the detectors (Henderson & Ferrari, 2016). Precision-recall (PR) curve is a plotted graph that shows the recall change for a given precision and vice versa. The general formula of mAP is formulated as follows:

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (3)$$

N represents the total number of classes in the detection task. In this study, N equals one as in one class object detection task. In the object detection domain, Intersection over Union (IoU) provides a metric to measure the amount of predicted bounding box that overlaps with the ground truth bounding box divided by the total area of both bounding boxes. This IoU is used to calculate the mAP as a threshold value, for instance, 0.5 in mAP50. That means detection is classified as true positive if have IoU above 0.5. This study used mAP50 to evaluate the performance of the damaged building opening detection mode.



## 4. RESULTS

### 4.1. Damaged Building Opening Dataset Development

Training, validation, and testing data are needed to develop a damaged building opening detection model. The supervised learning method in deep learning models requires many good training samples representing the context variety to generalize the task and prevent overfitting in the data. There is still no dataset for the task-specific detection of damaged building openings. Thus, dataset development is also conducted in this research.

This study uses images of the post-earthquake event in Nepal and Indonesia collected from previous research by Chachra et al. (2022). The selection procedure was conducted to select images containing damaged building scenes, continued by pre-processing and several augmentation processes. This study focuses on utilizing images that have a set of openings on the damaged building. Figure 4.1 shows the sample of different types of damaged building openings that are included in this research.



Figure 4.1. Type of openings: damaged walls, roofs, open doors and windows (from left to right)

Catastrophic disasters such as earthquakes can damage buildings and lead to unstructured building openings. These openings can be open doors or windows, collapsed walls, and damaged roofs. All these types of openings can be the entrance space for a victim-searching device, like a UAV, to enter the building. This study aims to develop a deep-learning model to detect these openings. The model must be able to recognize and locate the position of the openings. This requires a dataset that contains images and annotation to train the model. A total of 738 images are selected and ready to be annotated.

#### 4.1.1. Image Annotation

The annotation follows the image segmentation procedure by outlining the opening and covering the object entirely. This annotation can be used to train both segmentation and object detection tasks. Annotation labels will be converted as bounding boxes for object detection training purposes. There is no need to differentiate the opening for the real-case scenario based on the types. Thus, all the opening types are categorized into one object class, i.e. opening.



Figure 4.2. Sample Images with Annotations

There are some limitations to the image annotation result. First, The annotation quality depends on the interpretation to determine which openings can be entered by a UAV. As shown in Figure 4.2, the annotation (in red) covers all the spaces that are defined as openings. Some spaces that a UAV cannot enter are not annotated. Second, the balance composition of different opening types instances is not considered yet. There would be better if the number of open doors, windows, collapsed walls and roof instances were equally represented. However, this dataset is enough to be used as the initial stage of developing the damaged building opening detection model. A total of 738 images and the corresponding annotation are divided into training, validation, and dataset with the composition 70%, 20%, and 10%, respectively.

#### 4.1.2. Image Augmentation

A large amount of data is required to achieve satisfying results from a deep learning model. Due to the limitation in providing damaged building images and annotations, sufficient amounts of data to represent real-world problems are not possible. Hence this limitation can cause the model to be overfitting. One of the approaches to address this issue is data augmentation. It can help improve the model performance by creating various and diverse samples within the training dataset (Alomar et al., 2023). A total of 1500 training images are generated from the augmentation process as part of the training data quality improvement. The justification and result of several augmentation techniques that are applied to the training dataset are as follows.

##### Geometric Transformation

The first augmentation method applied to the dataset uses basic geometric operations such as flipping and shear. Flipping is one of the most intuitive methods to increase data size or diversity. Horizontal flipping is applied to add a variety of training samples by mirroring the images on the vertical axis. There is no specific pattern of the geometric form of the damaged building opening. The space is usually unstructured and irregular. Thus, applying horizontal flipping can improve the object variety.

Another geometric transformation used as the augmentation method is shear. Shear adds variability to perspective to help the model be more resilient to the camera and subject pitch and yaw (Nanni et al., 2021). Figure 4.3 shows the 15 degrees of horizontal and vertical shear applied to the sample image. This share augmentation is randomly applied to the training data. Shear helps handle random geometric forms of

damaged building openings by modifying the aspect ratio to present the variety of the openings; therefore model's generalization is improved.



Figure 4.3. Horizontal and Vertical Shear 15 degrees

Other geometry transformations like vertical flip, crop-resizing, and rotation were not implemented. In the actual case, the UAV for search and rescue is not flying upside-down. Therefore, the vertical flip is unnecessary because it is not realistic. Due to the small instance of the opening in one image, the crop method is not used. Missing small opening object instances and leaving only the background is the unwanted effect of crop and resizing. Applying image rotation also can cause the same disadvantage. Thus, rotation was not implemented as part of data augmentation.

#### Photometric Transformation

Photometric transformation is a data augmentation method that applies modification to change the pixel's value (Alomar et al., 2023). This transformation includes various methods, such as changing colour, brightness, or contrast. This study implements random brightness and probabilistic grayscale for the damaged building opening dataset development. Random brightness ranging from -25% to +25% is applied to improve the resiliency of the model in different lighting and camera settings. This augmentation is needed because search and rescue activities can be conducted anytime and under any weather conditions. This augmentation will help add training image variations in lower or brighter light conditions, as shown in Figure 4.4. For the grayscale, random 25% of the training images are converted into grayscale to improve the model in detecting the openings.

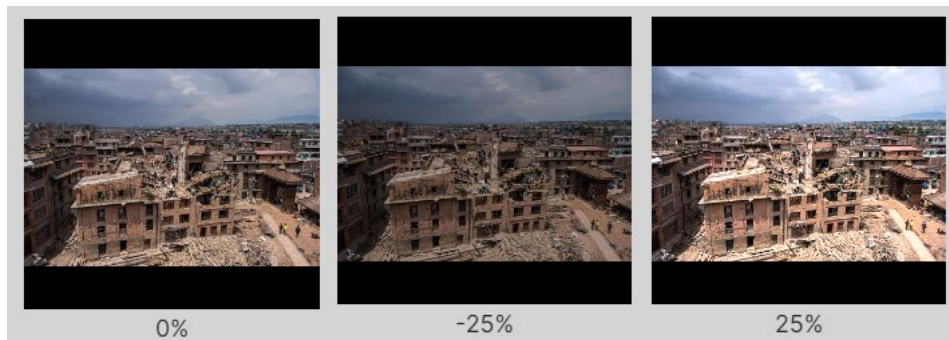


Figure 4.4. Original Brightness (left), -25% (middle), and +25% Brightness (right)

Other photometric transformation methods, such as colour casting and vignetting, are not applied. Colour casting shifts the RGB channels' intensities, while vignetting makes the image's edge darker (Wu et al., 2015). The main consideration is that both methods are irrelevant to the use case for damaged building opening detection. Adding these augmentations will probably only increase the computing memory but not significantly affect the model performances.

### Kernel or Filtering

In the deep learning application, the kernel plays a significant role in the feature extraction process. Besides, kernels also can be used in data augmentation by generating specific images with different images condition (Alomar et al., 2023). Blurring is an image-filtering technique that applies a kernel to the image. This study applied blur augmentation to increase the model resiliency with different camera focus settings. A 1-pixel Gaussian blur is applied to the training images randomly.

### Noise Transformations

Most of the real-case data is imperfect. Noise in the dataset can impair the model accuracy and decrease the generalization capability. Thus, data augmentation by introducing a different type of noise can improve the model's robustness. Spike noise is applied to the training images to improve the model's resilience to camera artefacts. This noise is distributed randomly to cover 1% of the pixels in the image. Figure 4.5 shows the sample images of grayscale, blur, and noise applied in the training dataset. Samples of training, validation, and test images are shown in Annex 2.

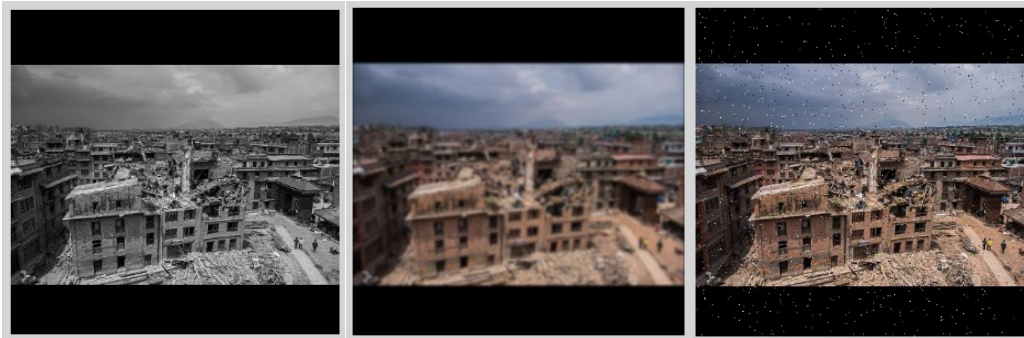


Figure 4.5. Grayscale, Blur, and Noise Images (Left to Right)

## 4.2. Single Task Object Detection

Based on the task capability, a network that works only on one task is called a single-task network. This study defines a regular detection network as single-task object detection to distinguish it from a multi-task learning network. The recent development of object detection networks is focused on improving detection speed while maintaining accuracy. Hence, the concept of one-stage detectors was proposed (Zhang et al., 2021). One-stage detectors have shown exemplary performance in obtaining good accuracy and low-inference time to obtain fast and good detection accuracy. This light detection network is useful for low-

powered device applications such as UAVs. As one of the current state of the art of real-time object detection networks, YOLOv5 is used as the main model framework for damaged building opening detection.

#### 4.2.1. Model Architecture

YOLOv5 model architecture consists of the backbone, neck, and head. The backbone network is used as the feature extractor of the deep learning model. The neck is a set of layers that combine the image features and then pass through the head to obtain bounding box prediction. The detailed descriptions of the YOLOv5 model architecture are already discussed in Section 3.2.2.

As one of the current state-of-the-art real-time object detection models, YOLOv5 shows good performance on both accuracy and inference speed. Three versions of YOLOv5 are trained on this dataset to provide better options for the damaged building opening detection model. Small, medium, and large versions of YOLOv5 are trained and evaluated on the damaged building opening dataset.

#### 4.2.2. Model Training

The training procedure of YOLOv5 is started by initializing the model configuration and pre-trained network. Using a pre-trained network will help the model define the initial weight parameters that are not random. As shown in Figure 4.6, different YOLOv5 versions are already trained and tested on the COCO dataset. These pre-trained weights were used to train the YOLOv5 on the damaged building opening dataset. No freezing layers are applied during the model fine-tuning from the COCO pre-trained model.



Figure 4.6. Training-Validation Box Loss (left) and Object Loss (right)

Figure 4.6 shows the training log of the YOLOv5s damaged building opening detection to understand the training performance better. Train/box\_loss shows the bounding box prediction losses on the training data, while the val/box\_loss shows the bounding box prediction losses on the validation data. Box prediction is responsible for finding the bounding box coordinates. We can see from the graph there is no overfitting in

the box prediction, as shown with the same curve pattern on both training and validation box losses. Object loss in the YOLOv5 model represents the bounding box score prediction losses. The YOLOv5 model predicts the bounding box coordinates (box) and gives information about the box score prediction (object). The object loss shows the different patterns on the train and validation dataset. The trained model worked well in predicting the object loss, shown as a declining loss curve, but contrary, the object loss on validation is increasing. This means that the model is overfitting when predicting objectness. The class loss is ignored because there is only one object class or single class detection.

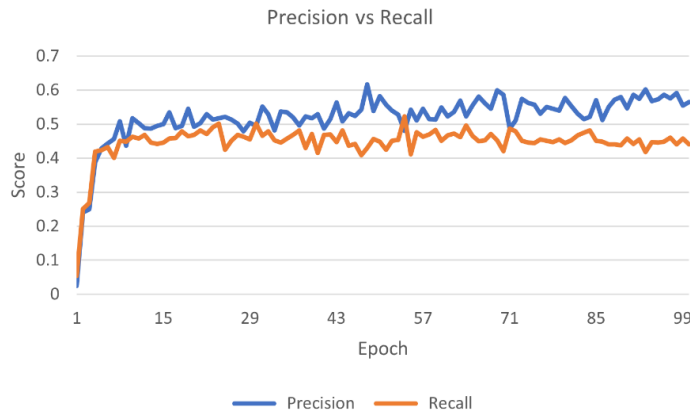


Figure 4.7. Training Precision and Recall of YOLOv5s

The model training log in Figure 4.7 also shows the increasing pattern of model precision. The same pattern happens for the model's recall but is lower than the model's precision. The precision represents the model's ability to detect true positives among all detections. For the opening detection scenario, higher precision is preferred. False detection will risk the UAV failing to enter the building. Thus better model precision is better. In comparison, less recall can be tolerated for this application case. The recall score shows how many openings are detected from all ground truth openings on the scene. Less recall means that some openings are not detected and will not bring any risk because the UAV will only navigate to the detected openings, not the false ones. On the other hand, from a first responder perspective, poor recall will give incomplete information.

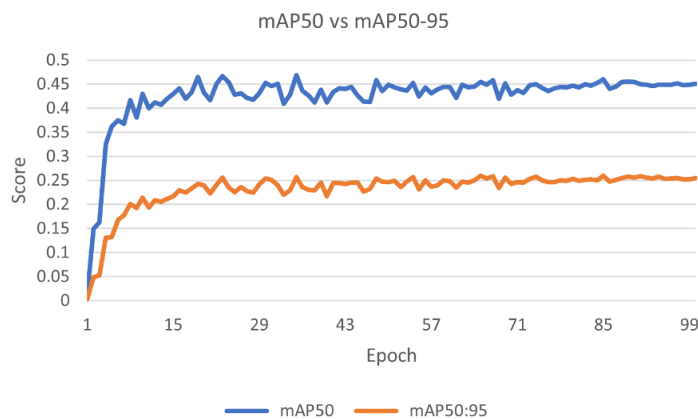


Figure 4.8. YOLOv5 Mean Average Precision (mAP)

Both mAP50 and mAP50-90 show incremental patterns concerning the training epochs, as shown in Figure 4.8. This indicates that increasing the training epoch will also increase the model's mean average precision (mAP). It can be shown from the mAP50 graph that from the 50<sup>th</sup> epoch and afterwards, the mAP tends to be constant. This indicates that increasing the training epoch over 100 will not significantly increase the model's mAP. The mAP50:90 shows lower precision because of the strict threshold applied to the model. Usually, mAP50 is more suitable to represent the model's detection performance.

#### 4.2.3. Model Performance

The models are trained using the same training-validation-test dataset and training configuration as in Section 3.3. The backbone is initially loaded with pre-trained weight from YOLOv5 trained on the COCO (Lin et al., 2014) dataset. Three versions of YOLOv5, i.e. small, medium, and large, are compared to find the best-performed model based on speed and accuracy in damaged building opening detection.

Table 4. Performance Comparison of YOLOv5 Detection Models Trained on Damaged Building Opening Dataset

Model	Precision	Recall	mAP50	Speed (FPS)
YOLOv5s	0.609	<b>0.477</b>	0.481	<b>107</b>
YOLOv5m	<b>0.656</b>	0.450	<b>0.501</b>	52
YOLOv5l	0.616	0.455	0.490	32

The YOLOv5 models trained on damaged building opening dataset are evaluated on test data which were not used in the training process. The models are tested on the Google Colab platform with NVIDIA Tesla T4 GPU. Table 4 shows the model's performance evaluated by calculating Precision, Recall, Mean Average Precision (mAP), and speed in frame per second.

The small, medium, and large versions of YOLOv5 show slightly similar performance results in terms of precision, recall, and mAP. YOLOv5s has a smaller mAP compared to the other two but has the fastest inference speed. This result shows that the small version of the model is more efficient in terms of processing time.

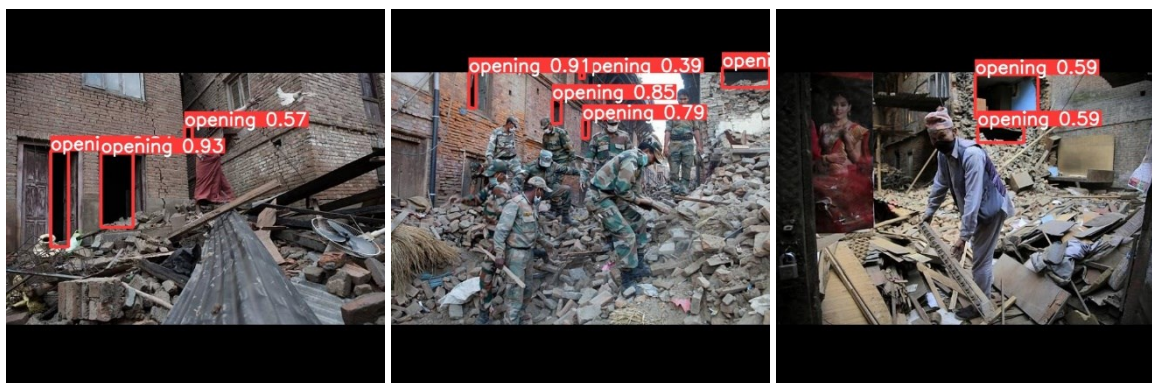


Figure 4.9. Example of Damaged Building Detection on Test Data

Figure 4.9 and Figure 4.10 shows the example of detection results from YOLOv5s on the test images. From the visual interpretation, as illustrated in Figure 4.9, we can see that this model successfully detected several types of openings. Open doors, windows, and collapsed walls were detected on the scene. This result shows that the YOLOv5 model can detect damaged building openings by providing relevant training data.

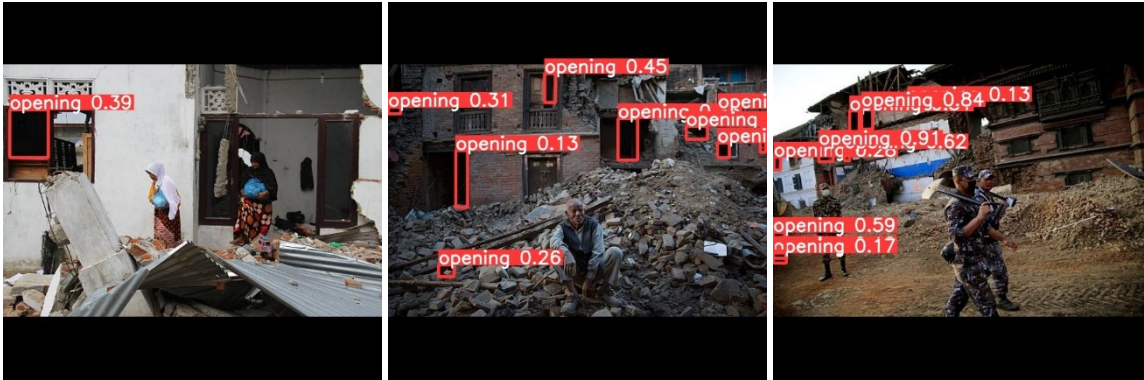


Figure 4.10. False Negative (left image) and False Positive (middle and right images) Detections

However, inference results on the test dataset also show several detection errors. On the left images in Figure 4.10, some openings are not detected as an opening (false negative). Furthermore, some non-building openings are detected as openings (false positive), as shown in the middle and right images in Figure 4.10. Similar visual characteristics of some non-opening and real opening objects can lead to this wrong detection result. The main challenge in detecting damaged building openings is the complexity of the opening's form, size, and shape. There are no unique characteristics in the size and form of the opening. However, the opening pattern can still be detected by better understanding the spatial context relative to other objects like walls and related objects like windows and doors.

A better understanding of contextualization is needed to reduce the false positives in the detection. Image segmentation provides more detailed information and excellent performance in contextualizing the scene (Singha et al., 2023). Therefore, the idea of multi-task learning by combining object detection and segmentation is proposed to obtain better damaged building opening detection.

### 4.3. Multi-Task Learning

Multi-task learning (MTL) is a machine learning model training method combining multiple tasks simultaneously (Crawshaw, 2020). MTL can potentially improve the network's performance for related tasks shared by complementary information and act as a regularizer for each task (Vandenhende et al., 2020). In damaged building opening detection, object detection and image segmentation models can be used to map the location of the openings. Both tasks can detect and locate the position of the openings. Thus, combining these tasks is expected to improve the model's performance by sharing combined losses. The development of MTL based damaged building opening detection model in this study is based on this idea. After knowing



the performance of the damaged building opening detection model trained on YOLOv5 single-task object detection networks, this study examines the application of YOLOv5-MTL for this detection purpose. YOLOv5-MTL is a network based on the latest YOLOv5 that combines object detection and segmentation heads as multi-task learning. The YOLOv5-MTL model architecture and performance are described in the sections as follows.

#### **4.3.1. Model Architecture**

The architecture of YOLOv5-MTL is based on hard parameter sharing by utilizing one backbone with two heads (detection and segmentation). The hard parameter-sharing technique can also benefit the model by reducing the risk of overfitting (Balamuralidhar et al., 2021). The hard parameter-sharing in the MTL network divides the parameter into shared and task-specific parameters. These task-specific parameters are trained separately for each task but use the cumulative Loss. The main architecture of YOLOv5-MTL is based on the YOLOv5 detection network with added functionality for image segmentation. The detailed model architecture of YOLOv5-MTL is already discussed in Section 3.2.3.

#### **4.3.2. Model Training**

The training process of the YOLOv5-MTL model started with initializing the model configuration. Following the standard configuration model in YOLOv5, the multi-task learning model was also initialized using the yaml configuration file format. This configuration consists of the model layers and parameters as in the network architecture (Figure 3.5). Using the data loader function, the model loads images with corresponding labels (bounding box coordinates) and masks (segmentation mask coordinates). The dataset was prepared and saved in YOLO annotation format. The dataset consists of JPG images and annotation masks in TXT format. For training purposes, the training and validation data were loaded.

The multi-task learning YOLOv5 model was trained using pre-trained weight on the COCO dataset. This pre-trained weight is used as the parameter initialization. Using pre-trained weights trained on extensive datasets like COCO helps the model initialize the parameter and fine-tune using the custom dataset, which is a damaged building opening dataset. This option is better than training the new model from scratch.

The input data then propagates to the hidden units for each model layer and generates the prediction output. Given the inputs, the network will calculate and store the initial variables for the neural network from the input to the output layer. An activation function plays a role in calculating the weighted sum of input and biases. YOLOv5 uses Leaky ReLU (Xu et al., 2015) and Sigmoid as the activation function for the hidden and final detection layers, respectively. During network training, the learning process tries to minimize the losses. In this case, the detection and segmentation losses are combined.

The output from forward propagation is then used for backpropagation. Backpropagation calculates the parameters' gradient in reverse ways of forward propagation. The model's parameter is then updated using a gradient calculated by backpropagation. Both forward and backpropagation are connected and contribute to getting the best-fit model.

Figure 4.11 and Figure 4.12 show the training log of the YOLOv5s-MTL damaged building opening detection to understand the training performance better. From the training and validation losses graph, the model performance during the training process can be observed.

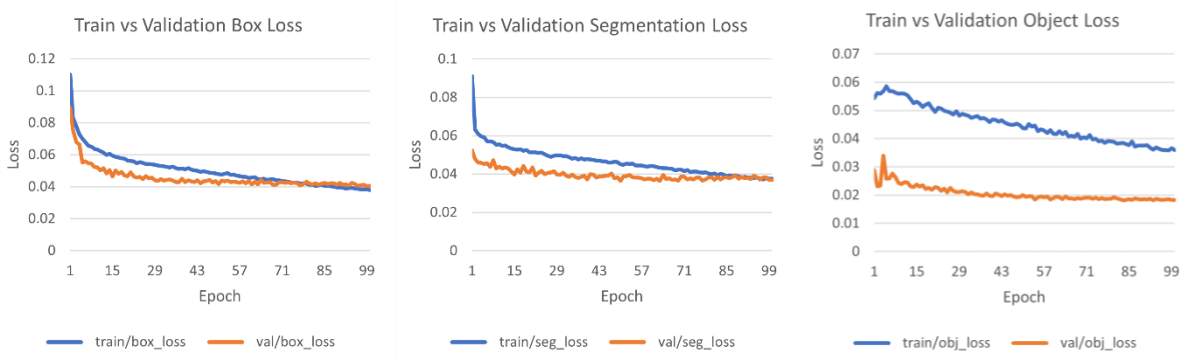


Figure 4.11. Box, Segmentation, and Object Losses

Four losses are computed in the multi-task learning based YOLOv5 model by adding segmentation loss, while box, object, and class loss are the same as in single-task YOLOv5. Since this is a single-class detection, this report does not report the training class loss. Both training and validation box losses show the same decreasing pattern. The same also happened for the segmentation and object losses. These indicate that the trained model can learn the task well and decrease the loss with respect to the number of epochs. The object loss overfitting on the single-task model is not an issue for this multi-task learning based model. From Figure 4.11, we can observe that no overfitting happened on the model.

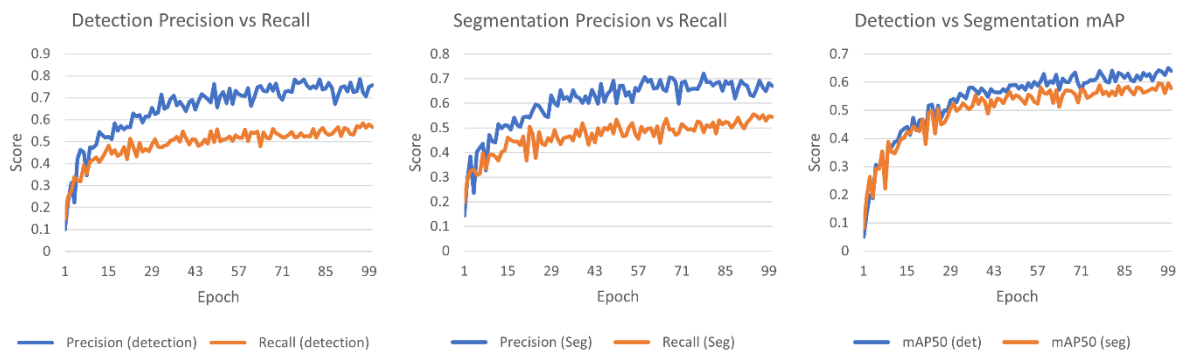


Figure 4.12. YOLOv5s-MTL Precision, Recall, and mAP

The precision, recall, and mAP score for every epoch is recorded and visualized in Figure 4.12 to monitor the accuracy performance of the model during training. The graph shows that the YOLOv5s-MTL has a

similar increment pattern of precision, recall, and mAP50 on detection and segmentation tasks. Interestingly, this chart shows that the continual growth of the model accuracies still tends to increase. This indicates that increasing the training epoch (more than 100 epochs) can increase the model's accuracy.

#### 4.3.3. Model Performance

The multi-task learning version of YOLOv5 is trained using the same training-validation-test dataset and training configuration as in Section 3.3. The backbone is initially loaded with pre-trained weight from YOLOv5 trained on the COCO (Lin et al., 2014) dataset. They were then trained on the damaged building opening dataset on three versions of YOLOv5-MTL, i.e. small, medium, and large. These multi-task learning models were evaluated using the same metrics in Section 3.4 on detection and segmentation performance. The models are tested on the Google Colab platform with NVIDIA Tesla T4 GPU.

Table 5. Performance Comparison of YOLOv5 MTL Models Trained on Damaged Building Opening Dataset

Model	Detection			Segmentation			Speed (FPS)
	P	R	mAP50	P	R	mAP50	
YOLOv5s-MTL	0.747	0.577	0.648	0.675	0.539	0.585	<b>73</b>
YOLOv5m-MTL	0.753	0.606	0.664	0.671	0.540	0.577	30
YOLOv5l-MTL	<b>0.799</b>	<b>0.629</b>	<b>0.698</b>	<b>0.732</b>	<b>0.556</b>	<b>0.611</b>	27

Table 5 shows the performances of the YOLOv5-MTL with three different architecture versions. The large version of YOLOv5-MTL shows better mAP for detection and segmentation than small and medium models. In terms of inference speed, the small version of the model gives the fastest speed than the medium and large versions with more than two times. This condition shows that the bigger model size will result in more model parameters and more time for image inference.

YOLOv5l-MTL is not able to perform real-time inference ( $< 30\text{FPS}$ ). Whereas YOLOv5m-MTL barely meets the real-time requirement. The complexity of the model influences these inference speed performances. As in Table 2 and Table 3, YOLOv5l has a deeper and larger neuron size. The difference mAP of small and medium models is only about 0.02. Thus, considering the inference speed, the YOLOv5s-MTL model is the optimum version for the damaged building opening detection model. Unlike the single-task detection networks, the YOLOv5s-MTL provide openings detection output as bounding boxes and segmentation masks. The prediction output provides information about the class name, class probability, bounding box, and masks, as shown in Figure 4.13. Same as in the detection output of a single-task detector (Figure 4.9), the MTL version of YOLOv5 is able to correctly detect the openings with additional information, i.e. segmentation mask.

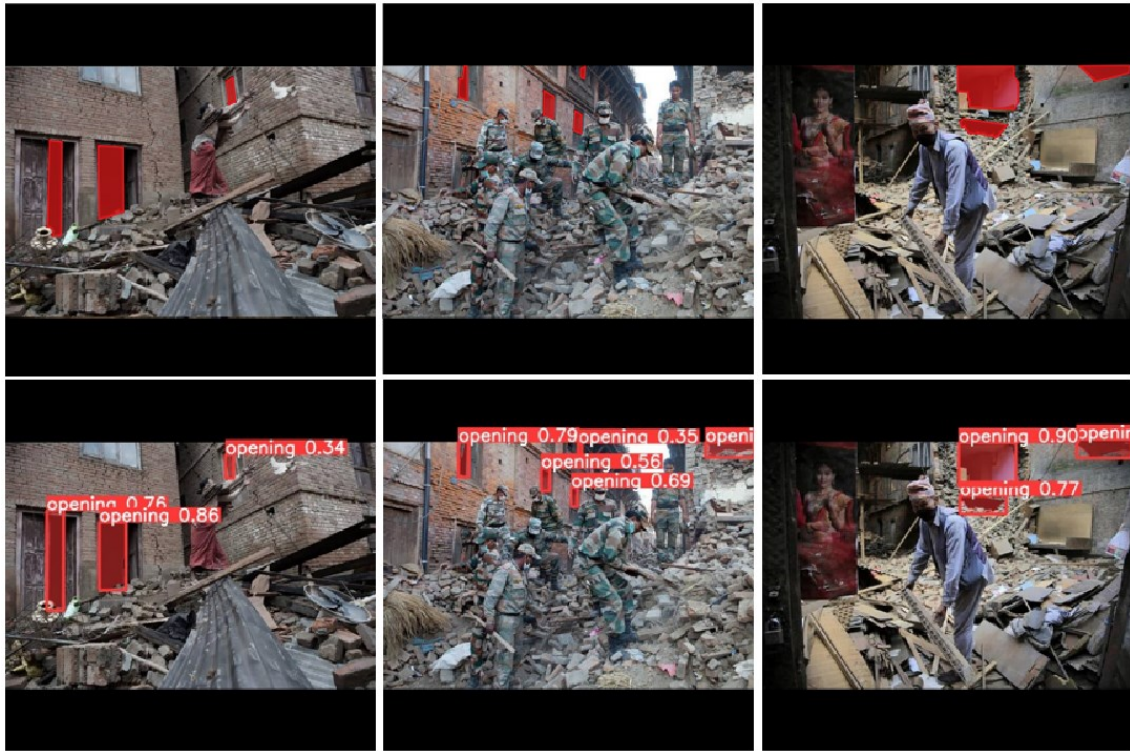


Figure 4.13. Ground Truth Image and YOLOv5-MTL Detection and Segmentation Prediction Results

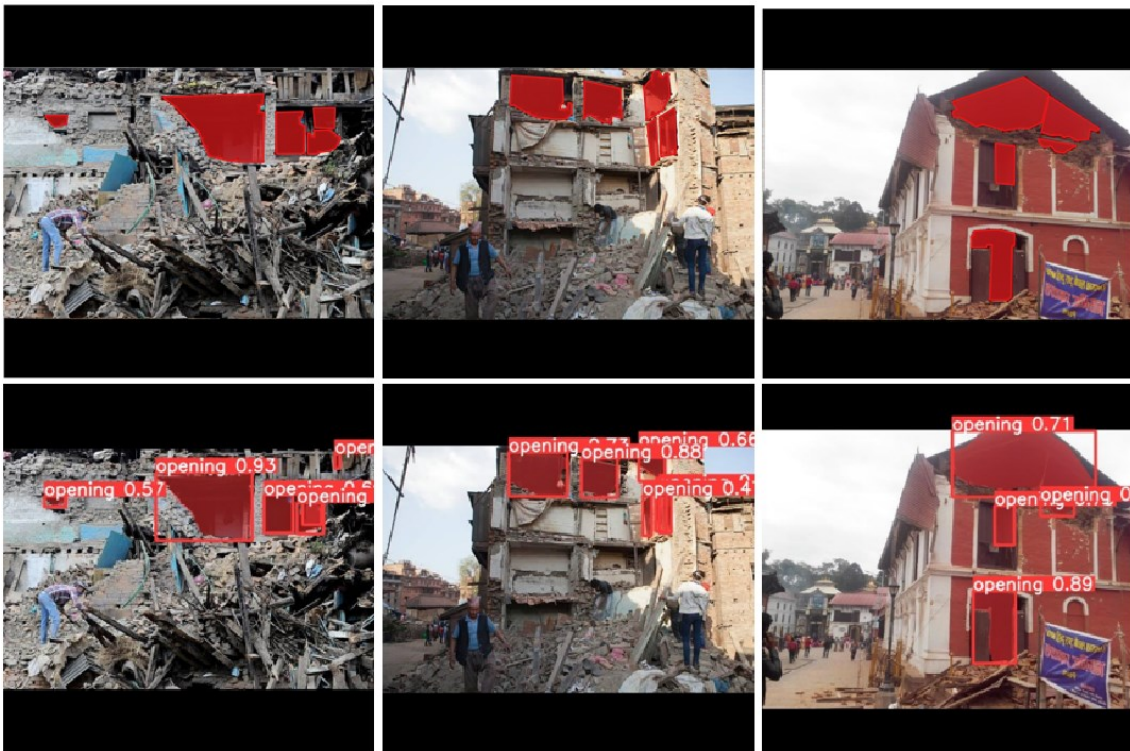


Figure 4.14. YOLOv5-MTL Predictions on Different Types of Damaged Building Opening

Collapsed walls or roofs are detected and segmented quite well by this model. As shown in Figure 4.14, different sizes and forms of openings due to collapsed walls and roofs can be detected and segmented by

this YOLOv5s-MTL model. Open doors and windows are also detected and segmented. The predicted mask on the test images shows that the model correctly delineates the opening space area. Qualitatively, the multi-task learning YOLOv5 model also shows better detection results based on visual interpretation.

#### 4.4. Model Performance Comparison

In the previous sections, single and multi-task learning-based detection networks were trained on the damaged building opening dataset. This section will compare both learning approaches and evaluate each task's benefits and drawbacks for the damaged building opening detection scenario. The model performance comparison is discussed as follows.

Table 6. Detection Performance Comparison

Model	Precision	Recall	mAP50	Speed (FPS)
YOLOv5s	0.609	0.477	0.481	107
YOLOv5s-MTL	0.747 (+0.138)	0.577 (+0.1)	0.648 (+0.167)	73 (-34)

The detection performance of single-task (YOLOv5s) and multi-task (YOLOv5s-MTL) are listed in Table 6. YOLOv5s-MTL shows improvement in precision, recall, and mean-average precision. While the single task YOLOv5s outperforms in terms of inference speed. Despite slower speed than single-task, the YOLOv5s-MTL still satisfies the FPS requirement for real-time application on low-powered devices.

The results show that adding the segmentation head on the network can improve detection accuracy but decrease the inference speed. The idea of combining detection and segmentation loss is proven to be able to improve the model's mAP by about 0.167. Multi-task learning tends to have a slower inference speed than single-task models due to a larger number of parameters. However, the inference speed of YOLOv5s-MTL can still be categorized as real-time (>30 FPS).



Figure 4.15. Ground Truth Image, Single-task, and Multi-task Output (Left to Right)

Figure 4.15 compares the ground truth image, detection output from the YOLOv5s model, and output from YOLOv5s-MTL, respectively (a clear comparison image can be seen in Annex 4). This figure shows that multi-task learning has improved the model's accuracy. The openings not fully detected on single-task are comprehensively detected and segmented by the multi-task learning based network.



Figure 4.16. Ground Truth Image, Single-task, and Multi-task Output (Left to Right)

In terms of precision, YOLOv5s-MTL is better than YOLOv5s. From Figure 4.16, it can be observed that multi-task learning has less false-positive detection than single-task networks. The detection errors are highlighted in orange circles. Overall, the multi-task model has fewer errors than the single-task. However, the multi-task model prediction result also shows some detection errors. The interesting fact from this finding indicates that better contextualization in image segmentation helps the detection task to reduce the non-opening detected as an opening (false-positive). The validation and test prediction results samples are shown in Annex 3.

## 5. DISCUSSION

This chapter discusses four points of the findings. Firstly, it discusses the quality of the damaged building opening dataset. Secondly, it discusses the performance of single-task networks in performing damaged building opening detections, with the following discussion on the performance of the multi-task networks. Lastly, it investigates the limitations of this study.

### 5.1. Quality of Dataset for Damaged Building Opening Detection Model

This study has developed a dataset of images and corresponding annotations for a damaged building opening detection model. This dataset covers several opening types: open doors/windows, collapsed walls, and damaged roofs. To simplify the detection model, all the opening types are categorized into one class, i.e. openings. In the real application scenario, there is no need to classify the opening types. Therefore single-class detection scenario is enough. The annotation quality depends on the manual annotation process. The annotator needs to provide detailed and correct annotations of the openings carefully. The challenge during the annotation process is to define the consistent criteria for the openings. Most of the space of the openings is not perfectly exposed, often covered by small objects like cables, ropes, sticks, or other debris. The definition of the opening space also can be various. Holes on collapsed walls or roofs can be defined as an opening if there is enough size (large enough for UAVs to enter). Other openings that are not big enough should not be annotated as openings. Thus, providing standard guidance for the annotation procedure is necessary.

A big size of dataset is needed to train a deep-learning model. One of the advantages of deep learning is the ability to handle large volumes of data. Besides the data quality, the bigger the training data, the better the deep learning model's performance. Based on this notion, providing a comprehensive training dataset for developing damaged building opening detection is crucial. Several augmentation techniques are applied to the annotated dataset to help increase the model's robustness. The choice of data augmentation that is used in the dataset is by considering the real-case scenario application. A horizontal flip is applied to add variations to the training data. Due to the unstructured form of the damaged building openings, flipping the image horizontally will increase the variations of the targeted objects. Shear is applied to increase the view perspective. This will help the model understand different perspectives that usually happen in real-case scenarios. The mosaic augmentation is used to improve the model in handling small objects. Other augmentation techniques like grayscale, random brightness, blurring, and noise are applied to enhance the model's robustness in handling various image qualities. Quite often, in different post-disaster conditions, the image quality captured by the UAVs is also varying. It can be darker due to low illumination and blur or contain noise due to the sensor's low quality.

The dataset developed in this study is sufficient for developing a damaged building opening detection model. However, several aspects are not covered in this dataset. First, the ground sampling distance (GSD) of the images in this dataset is not uniform. In an ideal training scenario, having training images with similar GSD is essential to ensure the constant object ratio. This will affect the model's performance in detecting various sizes of openings. Images used in this study are obtained from open data sources and captured from different cameras with different settings. Moreover, there is no available open data source for damaged building facade scenes captured using UAV. Nevertheless, the images used in this dataset are filtered to select the most relevant images representing the real-case point of view. Second, the opening types class balance is not considered in this study. Having a balance size of instances for every opening type can ensure the model becomes more optimal.

## **5.2. Single-Task Model Performance on Damaged Building Opening Detection Task**

The study found that a single-task YOLOv5 object detection network trained on a damaged building opening dataset can perform the detection quite well. The pre-trained weight of YOLOv5 trained on the COCO dataset helps the model initialize the weight so that model only needs to be fine-tuned. This training procedure is more efficient than training the model from scratch. From the experiments training on three different versions of YOLOv5, we found that the model's performances vary, as shown in Table 4. One interesting finding is that the bigger version of YOLOv5 does not promise better accuracy. We can see from the performance of the YOLOv5l (large) is not better than the medium version (YOLOv5m). The increase in the width and depth affects the complexity of the network. Intuitively, a larger model is expected to obtain better accuracy. The mAP50 of YOLOv5m is slightly better than YOLOv5l, with a 0.011 difference. This could happen because of the effect on the number of convolutional kernels in YOLOv5l. As shown in Table 3, the convolution kernel neurons numbers of the large model version are much more due to the larger width scale. Too many convolution kernels on the network can sometimes affect the model's overfitting. In the small version of YOLOv5, the bounding box score prediction is already detected as overfitting; this became worse in deeper networks, like in the large version of YOLOv5. For this reason, we found that the medium version of YOLOv5 is the optimal model configuration for a single-task damaged building opening detector.

For the inference speed, YOLOv5s perform fastest in predicting damaged building openings on an image compared to the other two versions, i.e., medium and large. This speed performance result is related to the complexity of the model. The bigger depth and width scales of the YOLOv5 model will increase the parameter number and the model's complexity. Furthermore, the more the number of model parameters is, the higher the inference time. The fastest inference speed is obtained on YOLOv5s with 107 FPS. In comparison, the inference speed of YOLOv5m is decreased by about half lower from the small version. However, YOLOv5m still meets the speed requirement for a real-time detection model (>30 FPS).



Considering the excellent accuracy while still being able to perform real-time inference, it can be concluded that YOLOv5m is the optimal model for the single-task damaged building opening detection model.

Besides the good performance in detecting the opening space of a damaged building, several errors can still be found in the detections. As illustrated in Figure 4.10, false negatives and false positives still appear in the detection results. The false negatives mean that opening spaces are not detected as openings. It seems possible that these results are due to the background classification mechanism. The false negative due to background classification mechanisms happens because the object blends with the texture of the image background (Miller et al., 2022). This could be the case on the damaged building opening dataset due to the similar texture between the openings and the background images. For instance, holes due to damaged walls can look identical to the background objects beneath the holes. Whereas false positive means that background objects are detected as openings. This error is highly avoided for the application case of damaged building opening detection. If the model detects non-opening as an opening, there will be a failure to enter the damaged building. Thus, it becomes crucial to minimize false positive detections. This effort is elaborated in the proposed multi-task learning detection model.

### **5.3. Multi-Task Model Performance on Damaged Building Opening Detection Task**

There is room for improvement in the previous results of the single-task damaged building opening detection model. The idea of combining image segmentation and object detection tasks into a multi-task learning model is proposed in this study. Image segmentation models tend to have better contextualization compared to object detectors. Detailed annotation in segmentation by masking the object boundaries benefits the model in understanding the object context. The ability to better understand the object's context helps minimize false positive detections. Using the same framework of YOLOv5, the multi-task learning version of YOLOv5 is trained to handle multi-task, i.e. object detection and segmentation. The computed losses from detection and segmentation are combined to get the total loss.

The proposed multi-task learning based model for damaged building opening detection is observed to outperform the single-task-based model. The YOLOv5-MTL was trained using the same training configuration to obtain comparable performance with the single-task network. From the experiments training on three different versions of YOLOv5-MTL, we found that the model's performances vary, as shown in Table 5. From these results, we can see that the more complex the model is in YOLOv5-MTL, the model's accuracy increases. YOLOv5l-MTL shows higher mAP50 among others two smaller model versions. Regarding speed, YOLOv5s-MTL performs the fastest inference time with 73 FPS tested on NVIDIA Tesla T4 GPU. Although the YOLOv5l-MTL gives the best accuracy, the inference time of this model is slow, with only 27 FPS. This means that the large version of YOLOv5-MTL cannot perform real-time inference (<30 FPS). Conversely, YOLOv5m-MTL is barely meeting the real-time inference requirements with 30 FPS. This multi-task learning model tends to have a slower inference time than single-

task detectors. This happened because working with two tasks will increase the model's complexity. Thus require more processing time. Even so, the small version of YOLOv5-MTL meets the requirements of a real-time detector. Considering the inference speed, YOLOv5s-MTL is the optimal model with better detection accuracy than the single-task detection model.

Based on the visual interpretation of the detection results, the YOLOv5s-MTL shows improvement by decreasing the false positive and false negative in the single-task detectors. This MTL network solves the background classification mechanism problems in the single-task model (Figure 4.16). The segmentation mask provides fitted boundaries that separate objects and backgrounds. Thus, the model can better in learning to differentiate between openings and non-openings. To decrease false positive detections, the image segmentation task helps the model understand contextualization better. As we observe from Figure 4.16, fewer false positives are detected on the image. The findings of this study show that the multi-task learning model performs better than the single-task detector while still having real-time inference.

#### **5.4. Limitations**

Even though this study meets all its research objectives, the findings and conclusions of this study should be interpreted cautiously. First, while the damaged building opening dataset showed decent data with completeness and variety to represent the actual use case, the images were not captured using a UAV camera. Most of the images on the dataset were captured on the ground with people eye level view. The point of view (POV) from a scene captured on the ground can be slightly different from oblique/near nadir POV. This will not have any significant implications for the model, but training the model using images captured from a UAV camera is preferred in the ideal case. Apart from that, the dataset developed in this study is still helpful by contributing initial data for the damaged building opening detection model.

Second, the model inference is not tested on the embedded platform due to resource and time limitations. Running the model on a high-performance GPU device could be preferred as the model can be run without limitations and give maximum performance. This study used an NVIDIA Tesla T4 GPU for network training and testing. However, such devices cannot be installed on mobile platforms like UAVs due to the minimum energy/power. Embedded platforms with limited resources and energy usually perform different performances in terms of inference speed. One example of an embedded GPU platform is Jetson Xavier NX. This device has 6 CPU cores and a GPU with 384 CUDA cores. Compared with the GPU used in this study, the Tesla T4 has 2560 CUDA cores. This difference shows that inference performance on the embedded platform could result in different performances. Thus, a series of experiments need to be conducted to examine the model performance in real-time damaged building opening detection tasks. Nevertheless, the findings of this study are still essential to compare the performance of single and multi-task-based networks for a specific detection application.

## 6. CONCLUSION AND RECOMMENDATION

### 6.1. Conclusion

The main goal of this study is to develop a deep learning detection model for real-time damaged building opening detection. Open doors or windows, damaged walls, and collapsed roofs are defined as a damaged building openings. To train the detection model, a damaged building opening dataset development was conducted. The process of dataset development consists of image selection, annotation, pre-processing, and augmentations. A total of 738 images were annotated by providing images with the corresponding mask labels of the opening objects. Horizontal flip, shear, random brightness, grayscale, blur, and noise were applied to the training data to improve the robustness of the model resulting in 1500 training images. The dataset developed in this study shows a decent dataset with completeness and variety to represent the actual use case.

As the most stable current state of the art of real-time object detection model, YOLOv5 is used as the main model architecture in this study. YOLOv5 was trained and evaluated on the damaged building opening dataset as a benchmark of a single-task object detection model. Three different versions of YOLOv5, i.e. small, medium, and large, were compared as detector models for damaged building openings. These YOLOv5 versions have different model complexity based on the total number of layers and the size of the output channel per layer. Overfitting is observed during the training of this single-task model in terms of object loss. Although, this study found that YOLOv5m has the best performance with the mAP50 of 0.501 and speed of 52 FPS. This single-task detector shows quite good detection results. However, the false positive detection of the model needs to be reduced.

This study tried to improve the detection performance by implementing a multi-task learning method by adding the segmentation head to the YOLOv5 model. Similar to in single-task YOLOv5 model, the architecture of the YOLOv5-MTL uses a New CSPDarkNet53 as the backbone and SPPF and New CSP-PAN as the neck layers. ProtoNet is added parallelly with the YOLOv5 object detection head to perform instance segmentation. The YOLOv5-MTL is observed to outperform the single-task model. The prediction output of this model provides information about the object masks along with the class name, class probability, and the bounding box. The proposed multi-task learning model sacrificed the processing time slightly but achieved accuracy improvements. YOLOv5s-MTL is the optimal network configuration for this damaged building opening detection application. This model can perform the detection with 0.648 mAP50 and a speed of 73 FPS. Compared to the single-task detector, the YOLOv5-MTL improves the model's mAP50 by about 0.167. Adding a segmentation head to the object detection model is proven to improve the model contextualization and reduce detection errors.

Overall, this study provides a novelty by contributing to developing an initial damaged building opening dataset and detection model. It also demonstrates that a multi-task learning method can improve detection accuracy. And at last, it broadens our knowledge in developing a real-time deep learning-based detection model algorithm.

## **6.2. Recommendation**

With the more rapid development of UAV applications in terms of system and data acquisitions, there is a potential for better improvement for disaster response applications. Several possible improvements can be considered as future research development, such as:

### **Training Dataset Improvement**

The disaster dataset in Chachra et al. (2022) was the only publicly available dataset that fit the criteria to train a deep learning model for damaged building opening detection during the study period. However, this dataset only contains crowd-sourced images without standard sensor and instrument settings. Further, standard images are needed to fit the application scenario of the model. The model will be applied to a UAV platform; hence, the source of training images captured using a UAV camera is preferred. Uniform ground sampling distance (GSD) is also essential for this case. By having the same GSD, the model will be trained better to deal with the opening size. Since not every opening on the damaged building can be entered by the UAV, considering the space size, training images with uniform GSD will facilitate this size difference issue in defining the opening class. This dataset improvement is expected to increase the model performance in real-use-case scenarios.

### **Add Image Depth Estimation**

Depth estimation is a computer vision task to measure the distance between pixel relative to the camera. This depth information is usually helpful for autonomous systems and navigation applications. In terms of damaged building opening detection, hypothetically, introducing the depth estimation task to the multi-task learning will help the model to detect the opening correctly. One of the characteristics of the opening is having deeper depth surrounded by objects with relatively similar depth (e.g. walls). This characteristic will help the model better understand and detect the openings. Thus, future research is interesting to assess the introduction of depth estimation head on the multi-task learning approach for the damaged building opening detection model.

## LIST OF REFERENCES

- Abdulateef, S., & Salman, M. (2021). A Comprehensive Review of Image Segmentation Techniques. *Iraqi Journal for Electrical and Electronic Engineering*, 17(2), 166–175. <https://doi.org/10.37917/ijeee.17.2.18>
- Alomar, K., Aysel, H. I., & Cai, X. (2023). Data Augmentation in Classification and Segmentation: A Survey and New Strategies. *Journal of Imaging* 2023, Vol. 9, Page 46, 9(2), 46. <https://doi.org/10.3390/JIMAGING9020046>
- Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12), 2481–2495. <https://doi.org/10.1109/TPAMI.2016.2644615>
- Balamuralidhar, N., Tilon, S., & Nex, F. (2021). MultEYE: Monitoring System for Real-Time Vehicle Detection, Tracking and Speed Estimation from UAV Imagery on Edge-Computing Platforms. *Remote Sensing* 2021, Vol. 13, Page 573, 13(4), 573. <https://doi.org/10.3390/RS13040573>
- Bi, L., Feng, D., & Kim, J. (2018). Dual-Path Adversarial Learning for Fully Convolutional Network (FCN)-Based Medical Image Segmentation. *Visual Computer*, 34(6–8), 1043–1052. <https://doi.org/10.1007/S00371-018-1519-5/FIGURES/7>
- Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. <https://arxiv.org/abs/2004.10934v1>
- Bolya, D., Zhou, C., Xiao, F., & Lee, Y. J. (2019). YOLACT: Real-time Instance Segmentation. <https://arxiv.org/abs/1904.02689v2>
- Boykov, Y., Veksler, O., & Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11), 1222–1239. <https://doi.org/10.1109/34.969114>
- Brahmbhatt, S., Christensen, H. I., & Hays, J. (2016). StuffNet: Using “Stuff” to Improve Object Detection. *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, 934–943. <https://doi.org/10.1109/WACV.2017.109>
- Carter, W. Nick. (2008). *Disaster Management: a Disaster Manager's Handbook*. Asian Development Bank.
- Chachra, G., Kong, Q., Huang, J., Korlakunta, S., Grannen, J., Robson, A., & Allen, R. M. (2022). Detecting damaged buildings using real-time crowdsourced images and transfer learning. *Scientific Reports* 2022 12:1, 12(1), 1–12. <https://doi.org/10.1038/s41598-022-12965-0>
- Chao, W., Jingjing, F., Zhuang, L., & Kuanwei, L. (2022). Research on the Influence of the Depth and Width of YOLOv5 Network Structure on Traffic Signal Detection Performance. *2022 6th CAA International Conference on Vehicular Control and Intelligence, CVCI 2022*. <https://doi.org/10.1109/CVCI56766.2022.9964985>
- Crawshaw, M. (2020). *Multi-Task Learning with Deep Neural Networks: A Survey*. <https://doi.org/10.48550/arxiv.2009.09796>
- Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, I, 886–893. <https://doi.org/10.1109/CVPR.2005.177>
- Delmerico, J., Mintchev, S., Giusti, A., Gromov, B., Melo, K., Horvat, T., Cadena, C., Hutter, M., Ijspeert, A., Floreano, D., Gambardella, L. M., Siegwart, R., & Scaramuzza, D. (2019). *The Current State and Future Outlook of Rescue Robotics*.
- Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm. *Procedia Computer Science*, 54, 764–771. <https://doi.org/10.1016/J.PROCS.2015.06.090>

- Diwan, T., Anirudh, G., & Tembhurne, J. V. (2022). Object detection using YOLO: challenges, architectural successors, datasets and applications. *Multimedia Tools and Applications*, 82(6), 9243–9275. <https://doi.org/10.1007/S11042-022-13644-Y/TABLES/7>
- Du, H. (2023). *General Object Detection Algorithm Yolov5 Comparison and Improvement* [CALIFORNIA STATE UNIVERSITY]. scholarworks.calstate.edu
- Dvornik, N., Shmelkov, K., Mairal, J., & Schmid, C. (2017). BlitzNet: A Real-Time Deep Network for Scene Understanding. *Proceedings of the IEEE International Conference on Computer Vision, 2017-October*, 4174–4182. <https://doi.org/10.1109/ICCV.2017.447>
- Elharrouss, O., Akbari, Y., Almaadeed, N., & Al-Maadeed, S. (2022). *Backbones-Review: Feature Extraction Networks for Deep Learning and Deep Reinforcement Learning Approaches*. <https://arxiv.org/abs/2206.08016v1>
- EMDAT. (2022). *2021 Disaster in Numbers*. <https://doi.org/10.1787/eee82e6e-en>
- Felzenszwalb, P., McAllester, D., & Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. <https://doi.org/10.1109/CVPR.2008.4587597>
- Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4), 193–202. <https://doi.org/10.1007/BF00344251/METRICS>
- Ghasemieh, A., & Kashef, R. (2022). 3D object detection for autonomous driving: Methods, models, sensors, data, and challenges. *Transportation Engineering*, 8, 100115. <https://doi.org/10.1016/J.TRENG.2022.100115>
- Girshick, R. (2015). Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1440–1448. <https://doi.org/10.1109/ICCV.2015.169>
- Girshick, R., Donahue, J., Darrell, T., Malik, J., Berkeley, U. C., & Malik, J. (2014). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1, 5000. <https://doi.org/10.1109/CVPR.2014.81>
- Goian, A., Ashour, R., Ahmad, U., Taha, T., Almoosa, N., & Seneviratne, L. (2019). Victim localization in USAR scenario exploiting multi-layer mapping structure. *Remote Sensing*, 11(22). <https://doi.org/10.3390/rs11222704>
- Gomez, C., & Purdie, H. (2016). UAV- based Photogrammetry and Geocomputing for Hazards and Disaster Risk Monitoring – A Review. In *Geoenvironmental Disasters* (Vol. 3, Issue 1). Springer. <https://doi.org/10.1186/s40677-016-0060-y>
- Harakannanavar, S. S., Rudagi, J. M., Puranikmath, V. I., Siddiqua, A., & Pramodhini, R. (2022). Plant leaf disease detection using computer vision and machine learning algorithms. *Global Transitions Proceedings*, 3(1), 305–310. <https://doi.org/10.1016/J.GLTP.2022.03.016>
- He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. *Proceedings of the IEEE International Conference on Computer Vision, 2017-October*, 2980–2988. <https://doi.org/10.1109/ICCV.2017.322>
- He, K., Zhang, X., Ren, S., & Sun, J. (2014). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8691 LNCS(PART 3), 346–361. [https://doi.org/10.1007/978-3-319-10578-9\\_23](https://doi.org/10.1007/978-3-319-10578-9_23)
- Henderson, P., & Ferrari, V. (2016). End-to-end training of object class detectors for mean average precision. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10115 LNCS, 198–213. [https://doi.org/10.1007/978-3-319-54193-8\\_13](https://doi.org/10.1007/978-3-319-54193-8_13)
- Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., NanoCode012, Kwon, Y., Michael, K., TaoXie, Fang, J., imyhxy, Lorna, Yifu), 曾逸夫(Zeng, Wong, C., V, A., Montes, D., Wang, Z., Fati, C., Nadar, J.,

- Laughing, Jain, M. (2022). *YOLOv5 SOTA Realtime Instance Segmentation*. <https://doi.org/10.5281/ZENODO.7347926>
- Khatib, O., & Siciliano, B. (2008). *Springer Handbook of Robotics* (B. Siciliano & O. Khatib, Eds.). Springer.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, 25. <http://code.google.com/p/cuda-convnet/>
- Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2, 2169–2178. <https://doi.org/10.1109/CVPR.2006.68>
- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2323. <https://doi.org/10.1109/5.726791>
- Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8693 LNCS(PART 5), 740–755. [https://doi.org/10.1007/978-3-319-10602-1\\_48](https://doi.org/10.1007/978-3-319-10602-1_48)
- Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path Aggregation Network for Instance Segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 8759–8768. <https://doi.org/10.1109/CVPR.2018.00913>
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9905 LNCS, 21–37. [https://doi.org/10.1007/978-3-319-46448-0\\_2/FIGURES/5](https://doi.org/10.1007/978-3-319-46448-0_2/FIGURES/5)
- Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June-2015*, 431–440. <https://doi.org/10.1109/CVPR.2015.7298965>
- Miller, D., Goode, G., Bennie, C., Moghadam, P., & Jurdak, R. (2022). Why Object Detectors Fail: Investigating the Influence of the Dataset. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2022-June*, 4822–4829. <https://doi.org/10.1109/CVPRW56347.2022.00529>
- Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., & Terzopoulos, D. (2020). Image Segmentation Using Deep Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7), 3523–3542. <https://doi.org/10.48550/arxiv.2001.05566>
- Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., & Terzopoulos, D. (2022). Image Segmentation Using Deep Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7), 3523–3542. <https://doi.org/10.1109/TPAMI.2021.3059968>
- Mittal, M., Mohan, R., Burgard, W., & Valada, A. (2019). *Vision-Based Autonomous UAV Navigation and Landing for Urban Search and Rescue*. <http://arxiv.org/abs/1906.01304>
- Mohamed, E., Shaker, A., El-Sallab, A., & Hadhoud, M. (2021). *INSTA-YOLO: Real-Time Instance Segmentation*. <https://arxiv.org/abs/2102.06777v2>
- Nanni, L., Paci, M., Brahmam, S., & Lumini, A. (2021). Comparison of Different Image Data Augmentation Approaches. *Journal of Imaging*, 7(12), 254. <https://doi.org/10.3390/JIMAGING7120254>
- Nex, F., Duarte, D., Steenbeek, A., & Kerle, N. (2019). Towards real-time building damage mapping with low-cost UAV solutions. In *Remote Sensing* (Vol. 11, Issue 3). MDPI AG. <https://doi.org/10.3390/rs11030287>
- Nock, R., & Nielsen, F. (2004). Statistical region merging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11), 1452–1458. <https://doi.org/10.1109/TPAMI.2004.110>

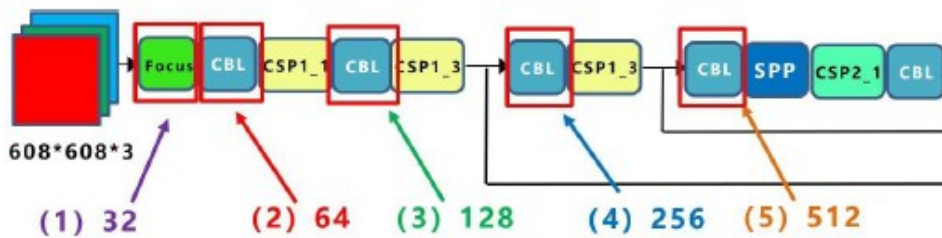
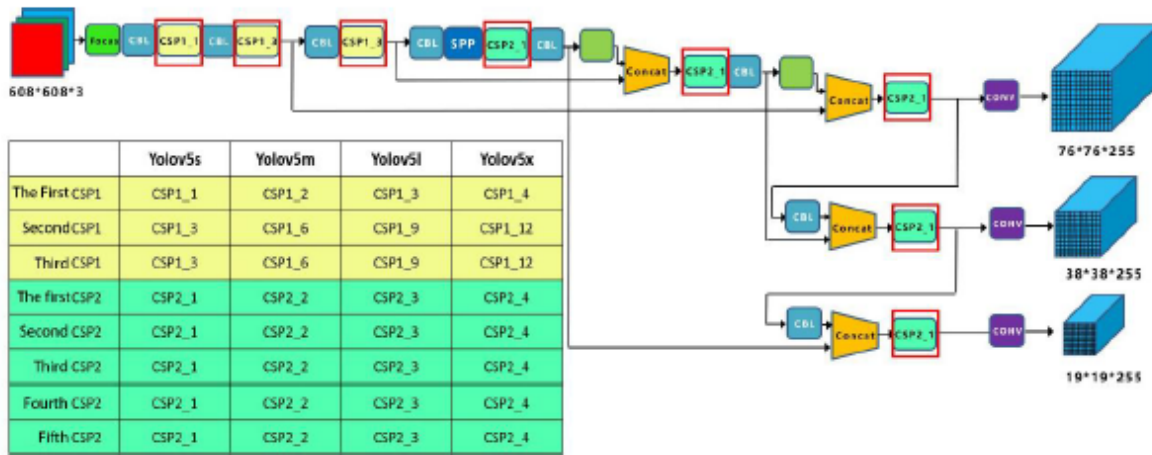
- Noh, H., Hong, S., & Han, B. (2015). Learning Deconvolution Network for Semantic Segmentation. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1520–1528. <https://doi.org/10.1109/ICCV.2015.178>
- Otsu, N. (1979). THRESHOLD SELECTION METHOD FROM GRAY-LEVEL HISTOGRAMS. *IEEE Trans Syst Man Cybern, SMC-9*(1), 62–66. <https://doi.org/10.1109/TSMC.1979.4310076>
- Pritzl, V., Stepan, P., & Saska, M. (2021). Autonomous Flying into Buildings in a Firefighting Scenario. *IEEE International Conference on Robotics and Automation (ICRA)*, 239–245. <http://mrs.felk.cvut.cz/projects/dofec>
- Rahouma, K. H., & Mahfouz, A. Z. (2021). Design and Implementation of a Face Recognition System Based on API mobile vision and Normalized Features of Still Images. *Procedia Computer Science*, *194*, 32–44. <https://doi.org/10.1016/j.procs.2021.10.057>
- Ramôa, J. G., Lopes, V., Alexandre, L. A., & Mogo, S. (2021). Real-time 2D–3D door detection and state classification on a low-power device. *SN Applied Sciences*, *3*(5). <https://doi.org/10.1007/s42452-021-04588-3>
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December*, 779–788. <https://doi.org/10.48550/arxiv.1506.02640>
- Redmon, J., & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*. <https://arxiv.org/abs/1804.02767v1>
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems*, *28*. <https://github.com/>
- Singha, T., Pham, D. S., & Krishna, A. (2023). A real-time semantic segmentation model using iteratively shared features in multiple sub-encoders. *Pattern Recognition*, *140*, 109557. <https://doi.org/10.1016/J.PATCOG.2023.109557>
- Tan, M., Pang, R., & Le, Q. V. (2019). EfficientDet: Scalable and Efficient Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 10778–10787. <https://doi.org/10.1109/CVPR42600.2020.01079>
- Tayara, H., Soo, K. G., & Chong, K. T. (2017). Vehicle Detection and Counting in High-Resolution Aerial Images Using Convolutional Regression Neural Network. *IEEE Access*, *6*, 2220–2230. <https://doi.org/10.1109/ACCESS.2017.2782260>
- Teichmann, M., Weber, M., Zöllner, M., Cipolla, R., & Urtasun, R. (2018). MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving. *IEEE Intelligent Vehicles Symposium, Proceedings, 2018-June*, 1013–1020. <https://doi.org/10.1109/IVS.2018.8500504>
- Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D., & Van Gool, L. (2020). Multi-Task Learning for Dense Prediction Tasks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(7), 3614–3633. <https://doi.org/10.1109/TPAMI.2021.3054719>
- Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *1*. <https://doi.org/10.1109/CVPR.2001.990517>
- Wang, C. Y., Mark Liao, H. Y., Wu, Y. H., Chen, P. Y., Hsieh, J. W., & Yeh, I. H. (2020). CSPNet: A new backbone that can enhance learning capability of CNN. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2020-June*, 1571–1580. <https://doi.org/10.1109/CVPRW50498.2020.00203>
- Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. <https://arxiv.org/abs/2207.02696v1>
- Wu, R., Yan, S., Shan, Y., Dang, Q., & Sun, G. (2015). *Deep Image: Scaling up Image Recognition*. <http://arxiv.org/abs/1501.02876>



- Xia, X., & Kulis, B. (2017). *W-Net: A Deep Model for Fully Unsupervised Image Segmentation*. <https://doi.org/10.48550/arxiv.1711.08506>
- Xu, B., Wang, N., Kong, H., Chen, T., & Li, M. (2015). *Empirical Evaluation of Rectified Activations in Convolutional Network*. <https://arxiv.org/abs/1505.00853v2>
- Yamanakkanavar, N., & Lee, B. (2021). A novel M-SegNet with global attention CNN architecture for automatic segmentation of brain MRI. *Computers in Biology and Medicine*, *136*, 104761. <https://doi.org/10.1016/J.COMPBIOMED.2021.104761>
- Yan, B., Fan, P., Lei, X., Liu, Z., & Yang, F. (2021). A Real-Time Apple Targets Detection Method for Picking Robot Based on Improved YOLOv5. *Remote Sensing 2021, Vol. 13, Page 1619, 13(9)*, 1619. <https://doi.org/10.3390/RS13091619>
- Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., & Sang, N. (2018). BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11217 LNCS*, 334–349. <https://doi.org/10.48550/arxiv.1808.00897>
- Zhang, N., Nex, F., Vosselman, G., & Kerle, N. (2022). Training a Disaster Victim Detection Network for UAV Search and Rescue Using Harmonious Composite Images. *Remote Sensing*, *14(13)*, 2977. <https://doi.org/10.3390/rs14132977>
- Zhang, Y., Li, X., Wang, F., Wei, B., & Li, L. (2021). A Comprehensive Review of One-stage Networks for Object Detection. *Proceedings of 2021 IEEE International Conference on Signal Processing, Communications and Computing, ICSPCC 2021*. <https://doi.org/10.1109/ICSPCC52875.2021.9564613>
- Zhao, Z. Q., Zheng, P., Xu, S. T., & Wu, X. (2018). Object Detection with Deep Learning: A Review. *IEEE Transactions on Neural Networks and Learning Systems*, *30(11)*, 3212–3232. <https://doi.org/10.48550/arxiv.1807.05511>
- Zou, Z., Chen, K., Shi, Z., Guo, Y., & Ye, J. (2023). Object Detection in 20 Years: A Survey. *Proceedings of the IEEE*, 1–20. <https://doi.org/10.1109/jproc.2023.3238524>

# ANNEX

## Annex 1. Model Structure and Number of Convolutional Kernel of YOLOv5



	Yolov5s	Yolov5m	Yolov5l	Yolov5x
(1) Neurons	32	48	64	80
(2) -	64	96	128	160
(3) -	128	192	256	320
(4) -	256	384	512	640
(5) -	512	768	1024	1280

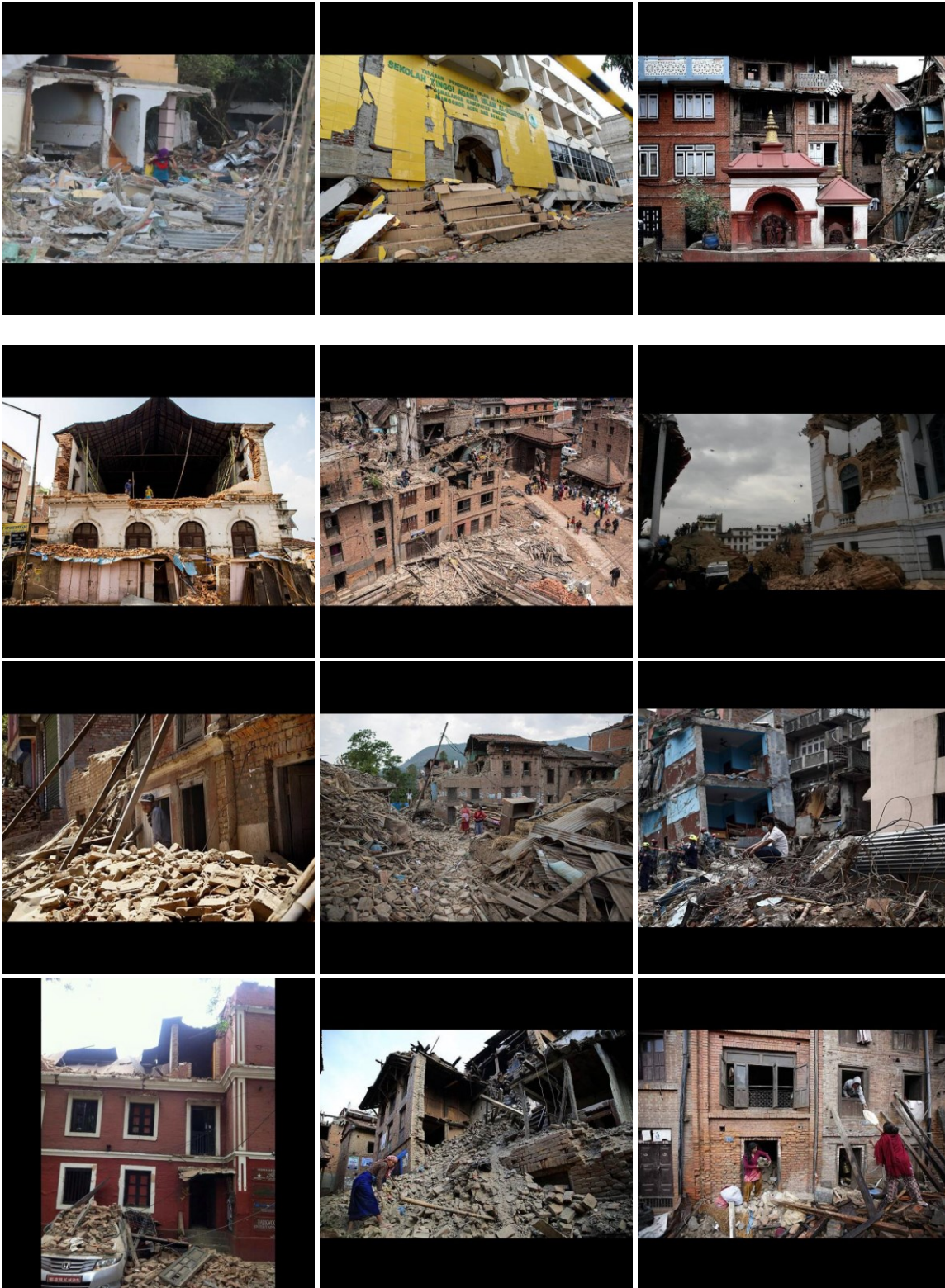
Source: Du (2023)

## Annex 2. Damaged Building Opening Dataset Samples

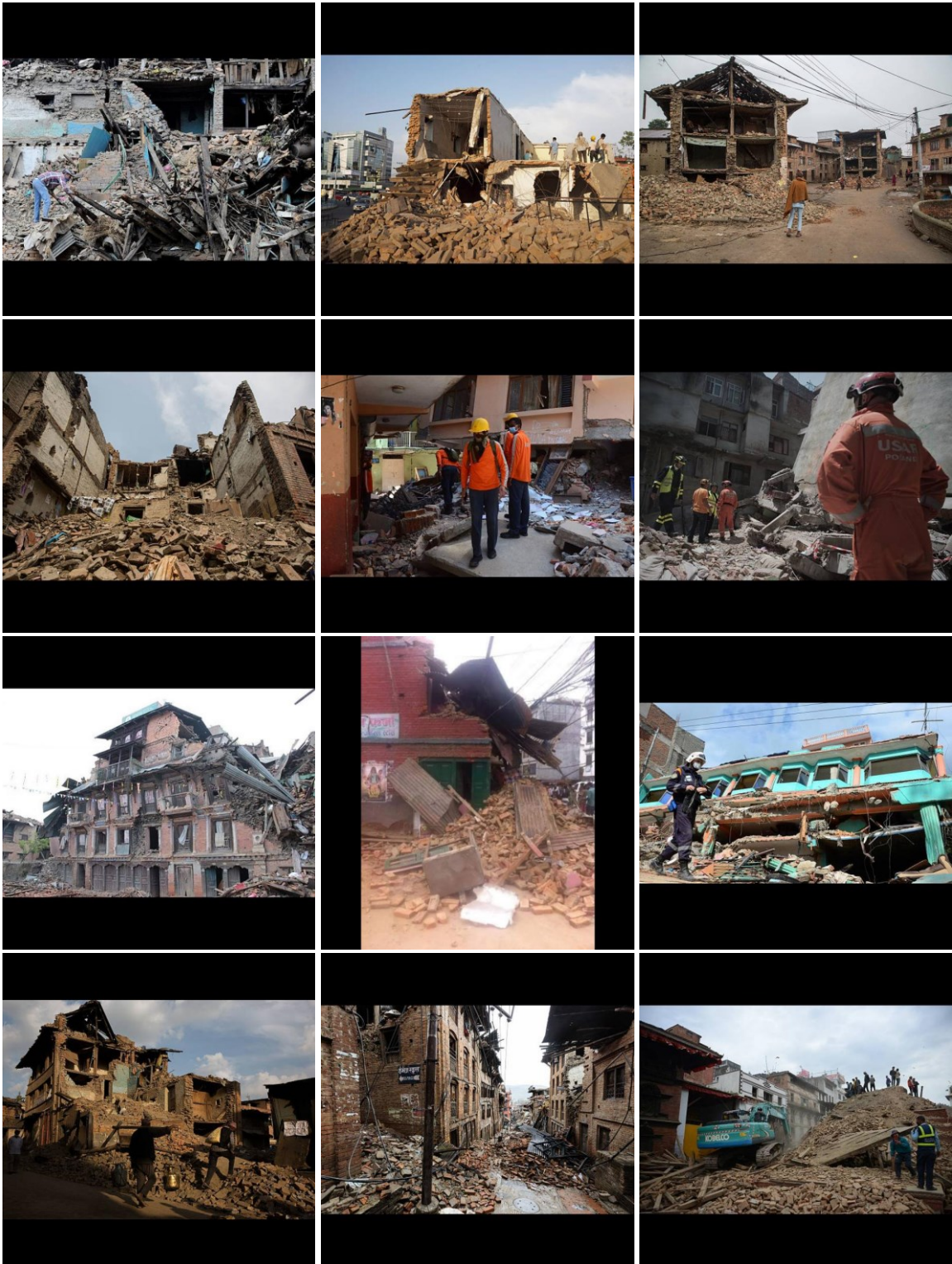
Training images sample:



Validation images sample:

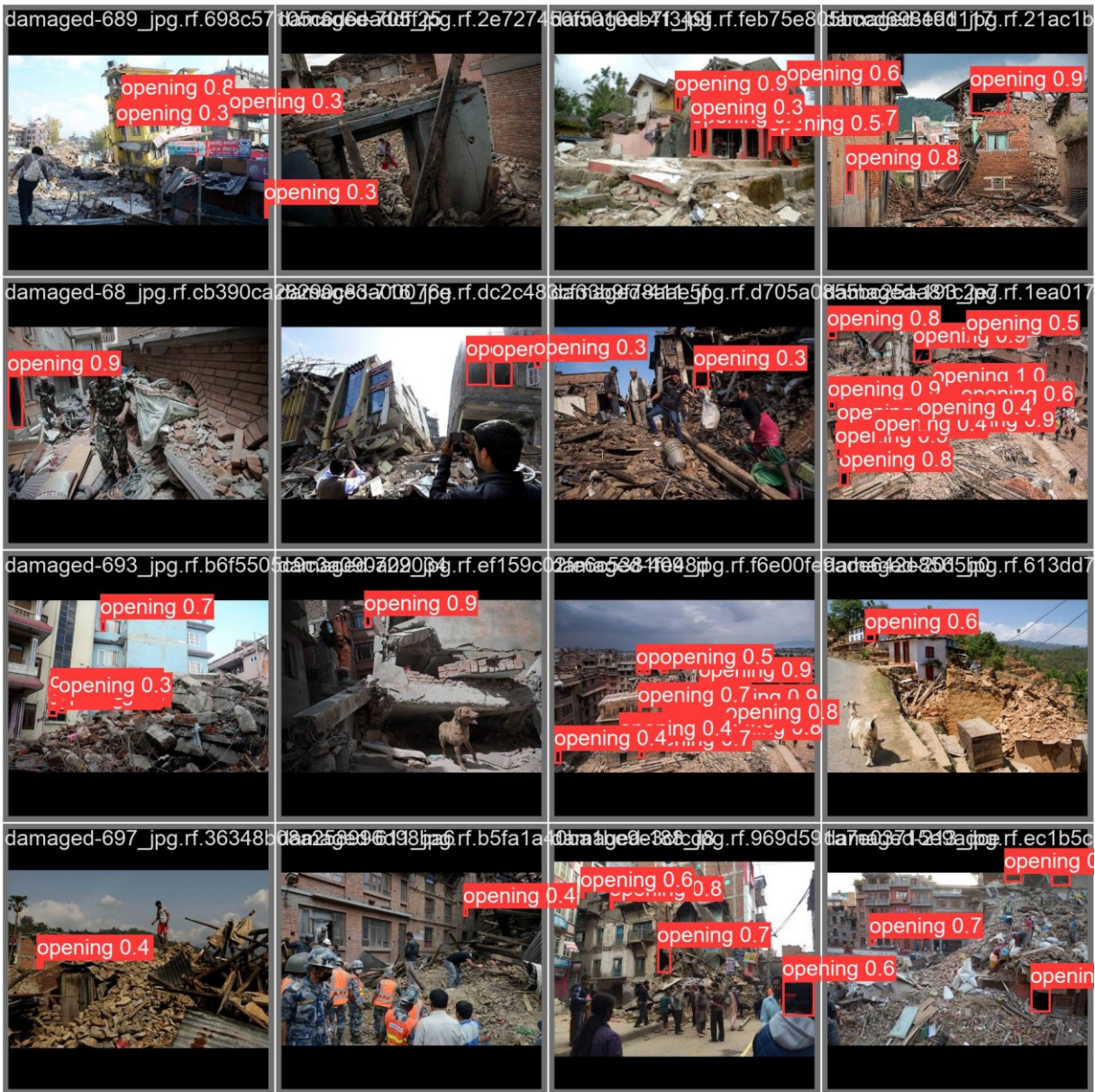


Test images sample:



### Annex 3. Train, validation, and test prediction image result samples

YOLOv5s (single-task) validation prediction results:



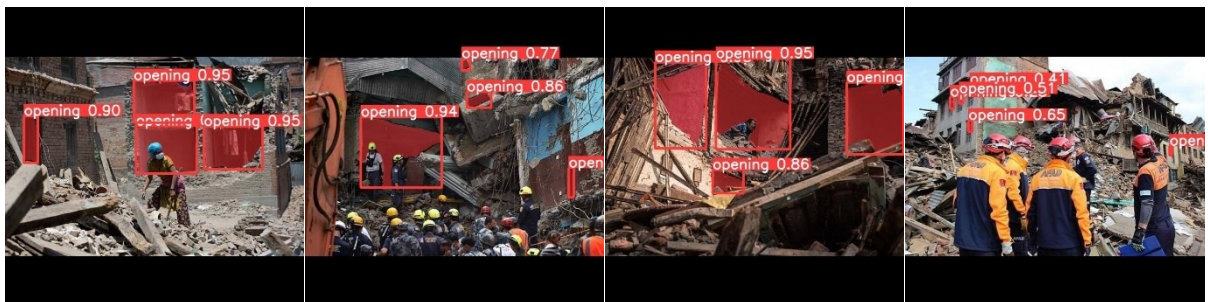
YOLOv5s prediction result on test dataset:



YOLOv5s-MTL validation prediction results:



YOLOv5s-MTL prediction result on test dataset:



Annex 4. Ground Truth Image and YOLOv5s-MTL Prediction Results on Test Dataset





