Deep Learning Model with Multi-Object Detection for Recyclable Waste Management

Priya Khandelwal

Department of Applied Data Science, San Jose State University

DATA 270: Data Analyst Process

Dr. Eduardo Chan

2022, December 5

Modeling

At every level of society, a timely, swift, and effective response is needed to properly manage waste accumulated over decades of population and economic growth.

Waste management, in addition to being an urgent global concern, is also a costly one. Recycling is one of the most effective ways to reduce waste and improve waste management because it involves repurposing and reusing trash.

An economy based on sustainability and circularity relies heavily on recycling. Although recycling has obvious and abundant benefits, it also poses several challenges. The wide variety of waste makes it hard for consumers and waste management employees to distinguish between recyclable and non-recyclable materials due to their wide variety. Therefore, our goal was to create a recycling classification and quantification instrument that households and waste management centres can use for categorising recyclable waste.

Using the sorting approach, This study trained the models to classify recyclable objects into different types, such as paper, plastic, glass, metal, and cardboard. In this study, the data was collected from two open-source datasets that include recyclable or non-recyclable datasets that are trashnet and trashbox, where the sizes of images vary across both data sources. That has been balanced as part of data pre-processing. Where trashnet captured Images using phones and trashbox images are downloaded and manually modified. This study used a few images from the trashbox and most of the images from the trashnet to train the model.

Data preprocessing performed on datasets that included merging datasets, Class Imbalance, Removing duplicate images and reducing the noise in datasets. This study included data transformation to increase model accuracy, including normalization, dimensionality reduction and augmentation.

Model Proposals

YOLO

Technology and Solution Survey. In a paper by Redmon et al. (2016), YOLO (You Only Look Once) is proposed for object detection to achieve a near real-time object detection as robust and accurate as a human vision that could be a more significant contribution to the current autonomous vehicle age. They have considered it a single regression problem rather than the other's previous considerations as an image classification problem. Most earlier approaches before the YOLO model were repurposing the available classifiers to identify the images. This is in contrast to sliding window detection, where several iterations are done on the image. Instead, YOLO, as the name suggests, looks once at the image and constructs the bounding boxes and the classification probabilities. Each image is divided into SxS grid cells.

Bounding boxes and confidence scores are given by each cell showing its prediction accuracy. A five-coordinate prediction is provided for each bounding box: [x, y, w, h, confidence], whereby (x,y) are the coordinates of the center of an object, w is width, h is height relative to the image, and confidence refers to the intersection over union (IOU).

The problem of overlapping bounding boxes is solved by predicting the highest IOU between current bounding boxes to ground truth. Pascal Visual Object classes dataset (VOC 2007) by Everingham et al. (2016), Picasso Dataset by Ginosar et al. (2014) and the People-Art datasets by Cai et al. (2015) were used for their use case of general object detection and artwork. A comparison is made with other model's performance.

Table 1 shows that R-CNN has high Average Precision(AP) for VOC 2007 but drops down when applied to artwork. However, DPM maintains an average AP for all the datasets. Out of all, YOLO is seen to perform well for VOC 2007 and the artwork.

Though YOLO can predict a single object; it fails in the case of multiple objects within a single grid.

Table 1

Quantitative Comparison of Models Performance over VOC 2007, Picasso and People-Art Datasets

	VOC 2007	Picasso		People-Art
	AP	AP	Best F_1	AP
YOLO	**59.2**	**53.3**	**0.590**	**45**
R-CNN	54.2	10.4	0.226	26
DPM	43.2	37.8	0.458	32
Poselets [2]	36.5	17.8	0.271	
D&T [4]	-	1.9	0.051	

A proposed ensemble model of YOLO with regions with convolutional neural networks (R-CNN) optimize overlapping object detection. A conclusion can be made that YOLO is a better-performing image detector that is simple yet efficient enough to be applied to different domains.

Yang et al. (2021) propose a model that performs garbage identification and detection using YOLOv5 and data enhancement procedures to make the model more accurate. The team compares various versions of the YOLO model and other object detection algorithms using the COCO dataset and comparatively tracks the enhancements. To improve accuracy, the model workflow goes through multiple stages like image enhancement, image segmentation, and image recognition. They used the open-source TACO dataset, labelled manually and classified hierarchically, to train and test the proposed model. The labels are organised into about seven different categories. The model provided the average recognition result as 94.5%. The proposed application of this model is in conveyor belt recognition and segregation of garbage waste through video surveillance.

Kundu et al. (2021) implement deep learning in seed segregation and packaging. The existing models in this area are challenged with new models created. ‘Mixed Cropping Seed Classifier and Quality Tester (MCSCQT) system was proposed using the YOLOv5 for object detection to perform seed classification and quality testing. The trained model achieved a precision call of 99% in classifying the seeds into maise and pearl millet.

Jiang et al. (2022) compare multiple object detection models—YOLO, YOLOv3, YOLOv5, Faster R-CNN, and Single-Shot Detector (SSD)—in terms of real-time detection of domestic waste. The author compares YOLO, Faster R-CNN, and SSD, then compares YOLOv3 and YOLOv5. The authors highlight four of their contributions in this work: 1) the introduction of the attention combination mechanism 2) the addition of a new small target detection layer to the head network to improve the model’s ability to detect small objects; 3) the use of Complete IoU (CIoU) loss function to reach optimal output prediction bounding box, and 4) the selection of Adam optimization algorithm to improve the model’s convergence speed during training while reducing the amount of variation in loss value. YOLOv5s was observed to be the more efficient and accurate model in computation intensity than other models.

Sozzi et al. (2022) compared various Convoluted Neural Network models in viticulture to estimate and predict the yield of white grapes. Six different versions of You Only Look Once (YOLO) models have been implemented here to compare, and they are YOLOv3, YOLOv3-tiny, YOLOv4, YOLOv4-tiny, YOLOv5x, and YOLOv5s. The aim is to be able to detect the bunch and provide the count in real time. The models were trained on self-acquired data from the fields and open-sourced databases.

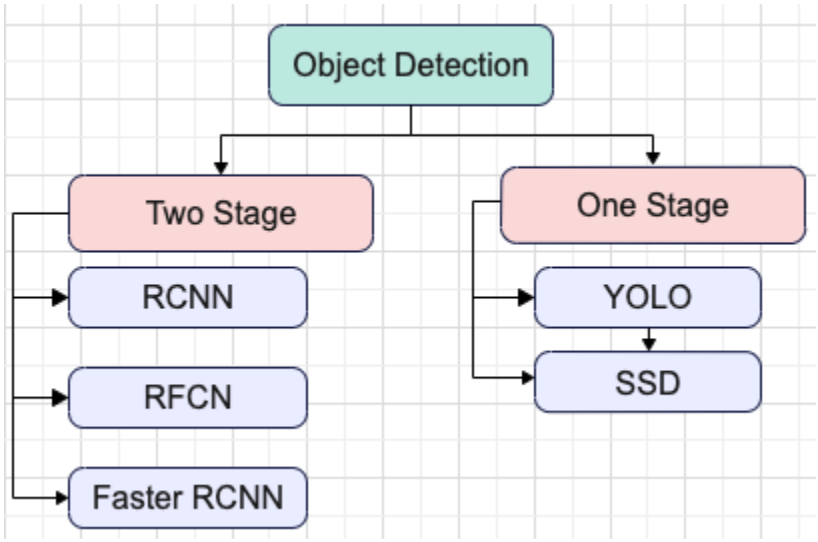
The performance comparison has been made based on the F-1 score and detection speed in FPS. A better performance was observed among the models YOLOv5x and YOLOv4 with an

F-1 score of 0.76 and 0.77 and detection time of 31 and 32 FPS, respectively. Moreover, YOLOv5x estimated the bunch counts with an average error of 13.3%.

After reviewing multiple research papers, YOLO was the best-performing model in speed and accuracy, while YOLOv5 is the newer and enhanced version of YOLO. So, The study selected YOLO for the research. The YOLO algorithm relies on its small size and fast calculation speed to detect targets. YOLO has a straightforward structure. Figure 1 depicts that YOLO is the one-stage algorithm, while the two-stage algorithm includes RCNN, RFCN etc. It can directly output the position and category of the bounding box through the neural network. YOLO's speed is fast because it only requires the picture to be put into the network to provide the final detection result. In YOLO, global images are directly used for detection. This allows the global information to be encoded, reducing the error of detecting the background as the object (Jiang et al., 2022).

Figure 1

Levels of Object Detection Models



Faster. In YOLO, the training network used is based on GooleNet. There are 24 convolutional layers in GooleNet V1 and two complete connection layers. The number of

convolution and convolution operations in Darknet-19 is, therefore, less than in GoogleNet. To reduce the amount of calculation, YOLO is the key. A final step involves replacing the entire connection layer with the average pooling layer (Jiang et al., 2022).

Better. For training, the original Yolo network used 224x224 pixels, while 448x448 pixels were used for detection. Adapting to image classification is possible when switching from the classification model to the detection model.

Figure 2

Feature Extraction Pseudocode

Algorithm 1 The pseudocode of feature extraction
1. For $X = 1$; $X <= (total\ number\ of\ training\ images)$; $X++$
2. Read photos of doors and Windows
3. Divide the picture into $n \times n$ areas for use
4. Search for areas with possible target centers
5. Generate bounding boxes in possible regions
6. Predict target width and height
7. According to anchor boxes size and object size adjust bounding boxes size
8. Predict the target category
9. Calculate the confidence score of bounding boxes
10. Output the center coordinates, width and height, and object category of the bounding box with the highest confidence
11. end for

YOLO Pseudocode. Zhang et al., (2022) represented the YOLO architecture and pseudocode to describe how YOLO works. The YOLO algorithm's flow is shown in Figure 2. The feature extraction pseudocode for Algorithm 1 is responsible for extracting the feature from the image and in Figure 3 Algorithm 2 for the training model is shown in this section.

YOLO Algorithm. According to YOLO, the problem of object detection is now a regression issue as opposed to a categorization issue. A convolutional neural network forecasts

class probabilities and picture-bounding boxes. By looking at the image only once, this algorithm identifies objects and their positions using bounding boxes.

Figure 3

Training Model Pseudocode

Algorithm 2 The pseudocode of training model.
Input: I: set of n training images
M: the width of anchor boxes
N: the height of anchor boxes
Output: P: the target category
(X, Y): coordinates of the center of the bounding boxes
W: the width of the bounding boxes
H: the height of the bounding boxes
1. For $n = 1$ to I do:
2. According to the feature extraction algorithm to extract the training picture: P, X, Y, W, H
3. Calculate the error between the target center coordinates predicted by the model (X, Y) and the real training images (X_n, Y_n)
4. Calculate the error between the width-height coordinates of the model predicted detection box (W, H) and the real detection box (W_n, H_n)
5. Calculate the confidence errors of the objects in the predictive detection boxes of models C_n
6. Calculate the confidence errors of the objects is not found in the predictive detection boxes of models
7. Calculate the error between the prediction categories P_n and P
8. The training model adaptively adjusts learning according to the loss function
9. Convergence of loss function
10. Obtain the training model with the minimum loss function to achieve target detection

This object identification algorithm uses S*S grid cells to partition the picture or frame into objects. Each grid cell forecasts the locations and dimensions of the B bounding boxes as well as the likelihood that an item will appear in the underlying grid and conditional class probabilities. A grid cell may identify an object since its center is located there. Using a sufficient bounding box, this grid cell is in charge of finding the item. Figure 4 illustrates how YOLO divide the image in a bounding box and the probability of the respective box class. Where

pc represents the probability of containing an object in the grid by the underlying bounding box, (*bx*, *by*), indicates the center of the predicted bounding box, (*bh*, *bw*) represents the predicted dimension of the bounding box, *p*(*ci*) means the conditional class probability that the object belongs to *ith* class for the *pc* and *n* is the number of classes/categories.

Figure 4

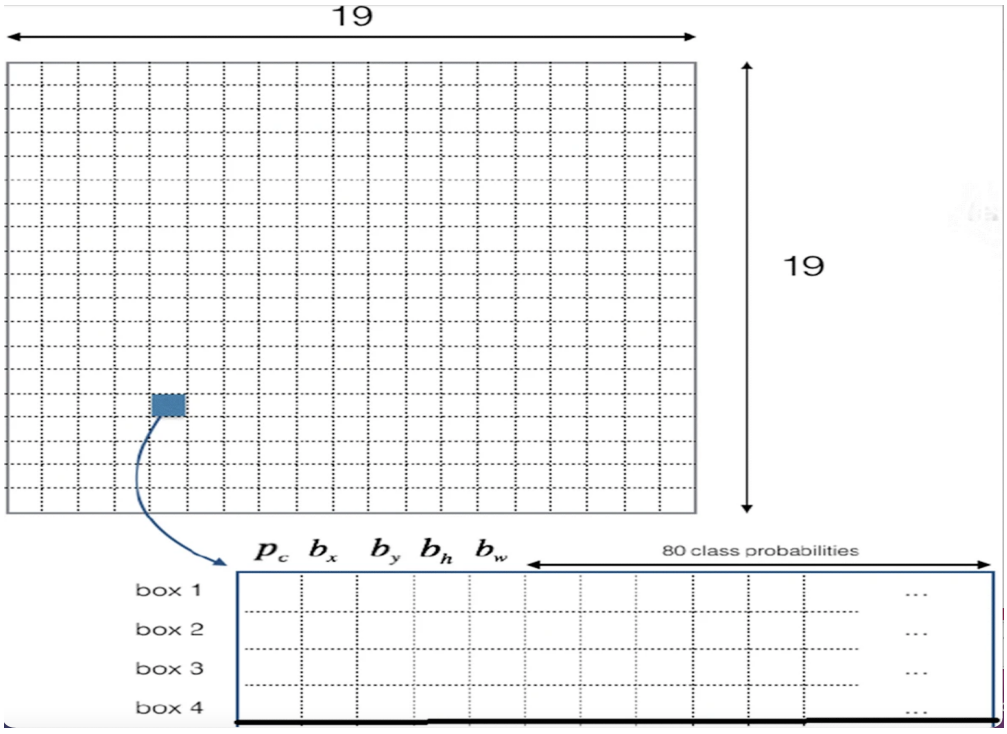
Probability and Bounding Box Division

p_c	b_x	b_y	b_w	b_h	$p(c_1)$	$p(c_2)$	...	...	...	...	...	...	...	$p(c_n)$

Figure 5 depicts the final schematic image divided into grid cells, and the supplied image is split into 19*19 grids. For each grid, four bounding boxes are projected. For a certain grid, class probabilities are distributed among the projected bounding boxes.

Figure 5

The Image Grid Cells and Predict Corresponding to a Cell



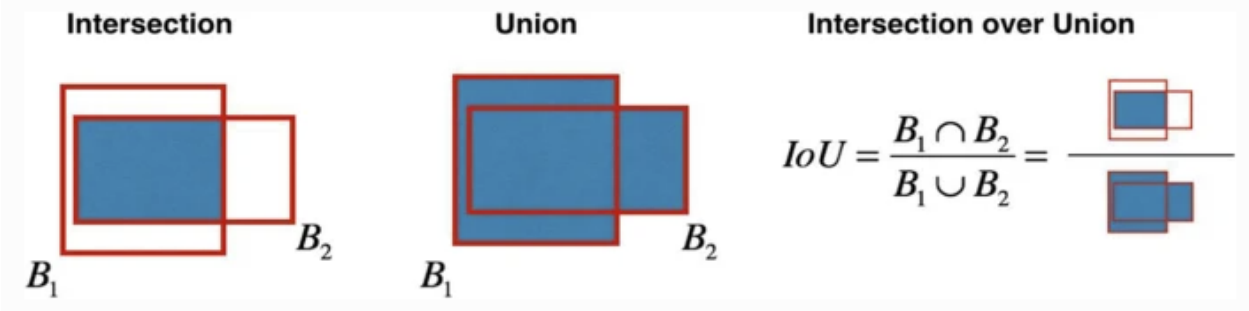
By multiplying pc by the intersection over union (IoU) between the ground truth and predicted bounding box, the confidence score (cs) is calculated for each bounding box per grid.

The confidence score would be zero if the object were not present in the grid cell. These bounding boxes typically vary in size while taking into account various shapes for capturing the objects, also known as anchor boxes.

Now, each grid's bounding box will be assigned a class-specific score, box coordinates, and classification output category. We will have a total of S2 B projected boxes, and boxes with class scores below a specific threshold will be removed. In most object detection algorithms, this threshold is typically fixed at 0.5. However, it may change based on the dataset and its properties. Low chance of an object being present in that grid or a low probability of any class category that optimises the class score may cause a low score.

Figure 6

Computer-Aided Design for Intersection Over Union (IoU)



as shown using Figure 6 as an example. Start by choosing the box with the highest class score. All additional bounding boxes that overlap the selected box will be ignored if their IoU exceeds a specific cutoff. Proceed through these phases until there are no bounding boxes with confidence ratings (Diwan et al., 2022).

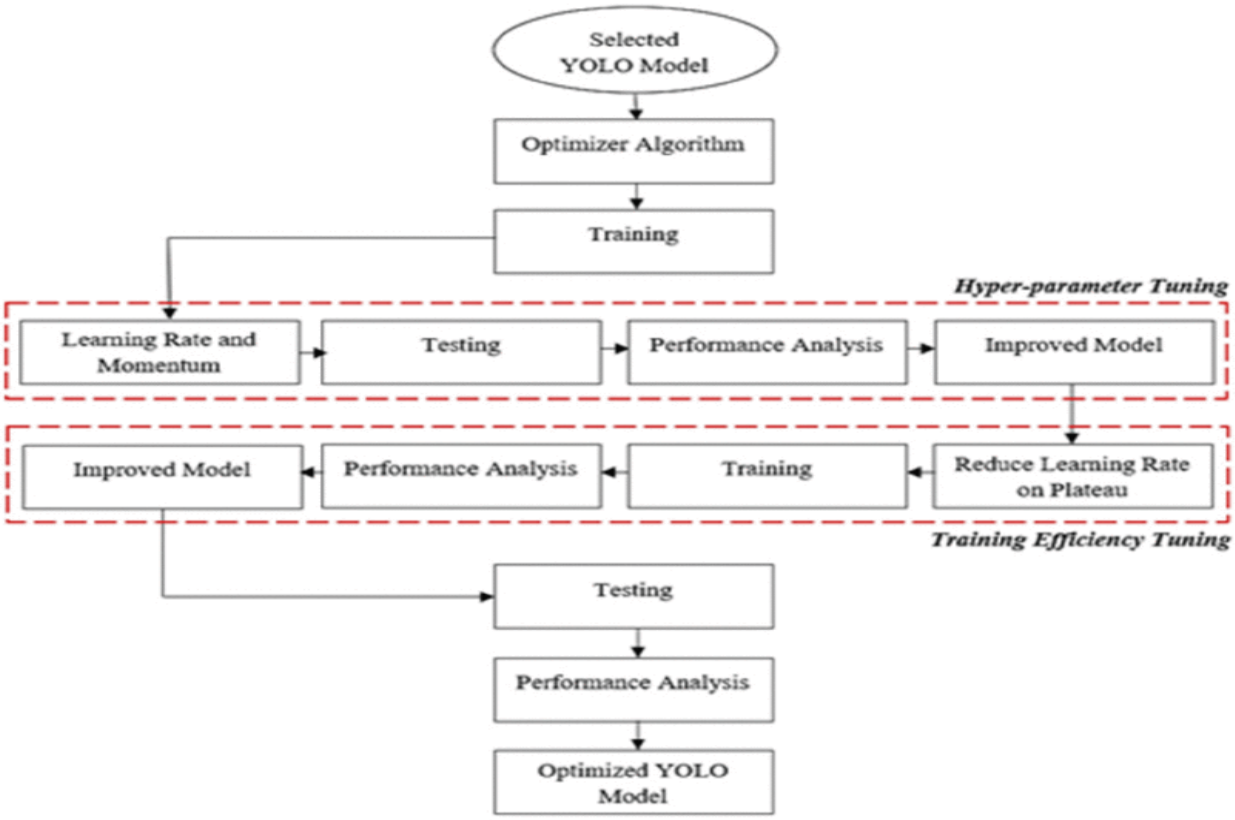
Optimisation of the model. Tuning the hyperparameter and learning rate of the YOLO v5 algorithm is the best way to get maximum performance. Figure 7 illustrates the

hyper-parameter in the optimiser, and the learning rate on the plateau is tuned in the suggested strategy. Based on the optimiser method, the chosen YOLO model was enhanced, focusing on the learning rate and momentum, as both attributes were essential for performance accuracy and processing speed.

This study examined two optimisers for training the chosen YOLO model (YOLOv5) with an emphasis on the model's optimisation. The implemented ADAM optimiser was contrasted with the default SGD optimiser. The best-fit parameters for model training were adjusted throughout the first ADAM execution stage. The learning rate and momentum throughout the training impacted ADAM's behaviour. The training was done for various values of both parameters as part of the tuning.

Figure 7

Hyperparameter Tuning Strategy of YOLO



To find the ideal combo value for improved training performance for the chosen YOLO, various learning rates and momentums were tested using ADAM. The momentum and typical values utilised in practice were 0.5, 0.9, and 0.99, whereas the learning rate varied between 106 and 1.0.

This work enhanced the behaviour of the learning rate on each epoch during the training utilising the approach of reducing learning rate-on-plateau in addition to employing ADAM as an optimiser to adjust the learning rate for each weight. By keeping track of the loss during training, this method increased the model's accuracy by moving down into regions with lesser loss. If the loss remained constant for several epochs, the learning rate would be slower (Isa et al., 2022).

Model Supports

Environment, Platform, Tools

Hardware Configuration. Object detection models often have strict hardware specifications. For processing massive volumes of data, Random-Access Memory (RAM) was utilized in our situation, using 32 GB. Additional faster graphics processing units (GPUs), such as the NVIDIA Tesla P100 or NVIDIA Tesla T4, were employed. Because of our significant computational requirements, we also needed two virtual CPUs (vCPUs). The Google Colab Pro tool satisfied each and every one of these hardware specifications. In addition, we utilized 15 GB of storage for our unprocessed and preprocessed data. GCS Standard Storage, Los Angeles, is used to complete this (us-west2). A summary of all the hardware specifications we utilized is shown in table 2.

Software Configuration. All software utilized in this study is managed by MacOS Big Sur 11.5.2, where Roboflow is used for Data preprocessing as well as a source for modelling. Jira is used for communication.

Tools. Table 3 depicts the list of tools and methods that have been used to implement and evaluate the method.

Table 2

Hardware Configurations and Purpose

Hardware	Configuration	Purpose
RAM	32 GB	Short-term memory for the processor
CPU	QTY 2 vCPUs	Processors
GPU	GPU to NVIDIA P100 or T4	Specialized processor to accelerate graphic rendering
GCS - Standard Storage, Los Angeles (us-west2)	15 GB	Datastore

Model Architecture and Data Flow

As seen in Figure 8, YOLO has 24 convolution layers, followed by two ultimately linked layers. Only four of the 24 convolutional layers, or "max-pooling layers," are followed by further layers. The highlights of this version of the technique include (1*1) convolution and global average pooling. Before passing the input picture through the convolutional network, it is resized to 448x448. The model was trained and fine-tuned using the first twenty layers, an average pooling layer, and a fully connected layer. The model is then fine-tuned for the item recognition task after integrating four more convolutional layers and two fully connected layers with randomly initialized weights. All convolutional and dense layers feature a Leaky Rectified

Linear Unit activation function, with the exception of the last layer, which has a linear activation function (LReLU).

Table 3

Tools, Methods and Purpose

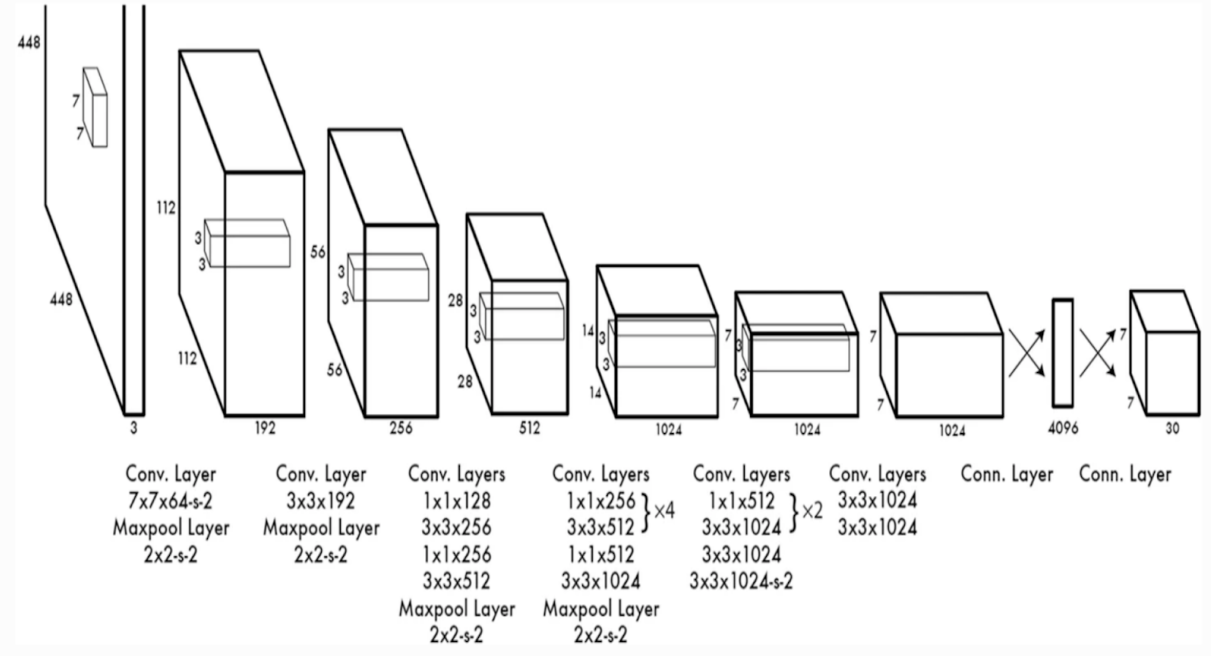
Library		Methods	Usage
PyTorch	YAML	open	To open the configuration file
Roboflow	roboflow	Roboflow	Fetch the code from roboflow
google.colab	drive	mount	To access the google drive
utils	downloads	attempt_download	A jumble of tools and Functions of use
	plots	plot_results	To plot the logs in the form of graph
	glob	glob	To access the image file from storage
TensorBoard	tensorboard	tensorboard	Model performance monitoring
Ipython	Display	Image, clear_output, display	To display images and it does not block output until the thread is clear
	magic	register_line_cell_magic	To start an interactive debugger

The final layer predicts class probabilities and bounding box coordinates.

Figure 9 illustrates the architecture or data flow of the YOLO v5 model where it can be seen mainly three layers backbone, neck, and head that is playing a vital role in detecting objects from images. The image has to go through these three layers to perform object detection.

Figure 8

YOLO Architecture of Detecting Objects



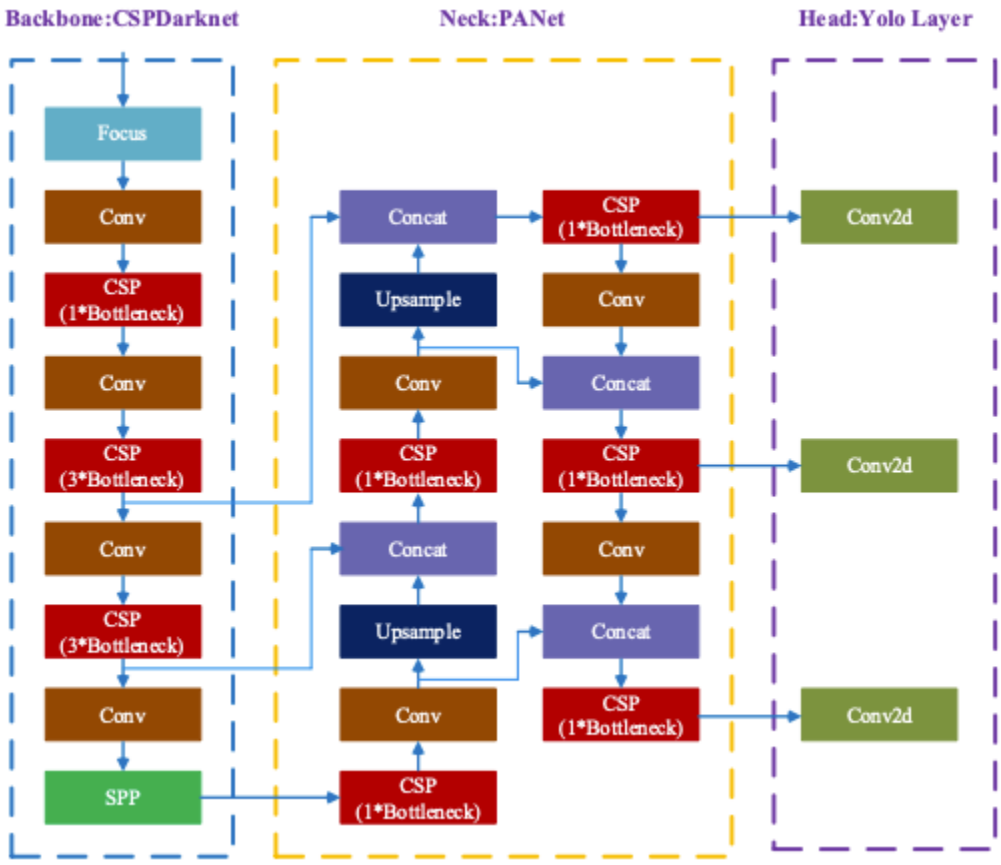
The gradient flow is separated in CSP's feature map of the base layer to propagate through several routes. Implementing the CSP link in a deep network would boost CNN's capacity for learning and, as a result, will increase accuracy while requiring less processing power.

Additionally, CSP eliminates the computational bottleneck by evenly dividing the processing across all of CNN's layers. Additionally, CSP will reduce memory costs by compressing the feature maps during the creation of the feature pyramid through the model neck. This is done by employing cross-channel pooling. These feature pyramids make it easier to

distinguish between similar items of various scales and sizes. In this regard, YOLOv5 employs the path aggregation network (PANet) as the model neck, which is especially advantageous for instance segmentation in maintaining the accuracy of the geographical information. This precision contributes to precisely locating the pixels.

Figure 9

Architecture and Dataflow of YOLOv5



The bottom-up method offered by PANet uses clear horizontal connections to connect the lower layers to the upper ones. The shortcut link is the only one on the trail, and it only has 10 levels. YOLOv5 employs a dense prediction in the section head. The label and a vector with the expected bounding box's coordinates confidence level make up the final prediction. In order to analyze the output, low-scoring boxes (those with scores below the confidence level) are

eliminated, and just one box is chosen from a group of overlapping boxes to identify the same item based on architectural variance. YOLOv5 consists of numerous versions, including YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, each of which has a different depth and width. However, only the YOLOv5s devices were employed in this investigation. In general, two configurations of depth multiple and width multiples were calibrated for each model and scaled for the network segments separately. The width multiple served as the layer channel multiple utilized to scale up the model's backbone and feature network, while the depth multiple represented the model's depth factor (Fang et al., 2021).

Model Comparison and Justification

Table 4 illustrates the comparison of Faster-RCNN and YOLO v5. Faster-RCNN is a two-stage object detection network technique. Compared to one-stage algorithms like YOLO, two-stage algorithms have more complex network architectures, but they also provide relatively high detection precision at the tradeoff of slower sample detection times. When applied to a GPU server, the recommended method enables the identification of a single sample in about 0.2 seconds. This detection period meets the requirements for clinical usage (Zhu et al., 2022).

Despite the fact that YOLO and Faster R-CNN both employ CNN as their foundation and that this is one of their primary goals, their conceptual frameworks are very different from one another. Faster R-CNN maintains the fundamental principles of R-CNN, including regional proposal and ROL pooling while utilizing CNN to deal with the entire input picture initially and divide proposals subsequently. However, the use of the Regional Proposal Network, which significantly speeds up the computation of processing proposals, is its main contribution. YOLO separates the entire image in the beginning before processing it with CNN.

Faster-RCNN was created based on R-CNN and Fast-RCNN. The backbone feature extraction network, RPN (regional proposal network), and prediction & localization network make up its core architecture. Whereas the main architecture of YOLO is made up of a backbone, neck (PANet), and head network. Faster-RCNN is a Region Proposal Based Algorithms on the other hand YOLO is a Regression-Based Algorithms.
Although, Both models can process the images and videos input image size for training images and FPS processing is different. Faster R-CNN models perform worse than YOLO v5 object detection when it comes to seeing small objects in images.

Three types of augmentations are carried out automatically by the YOLO-v5 data loader: scaling, color space changes, and mosaic augmentation. Where mosaic is a brand-new and powerful data augmentation strategy that joins four training pictures into one using certain ratios. It is advantageous to enhance the training dataset in order to improve the detector's performance and prevent overfitting. Whereas Faster RCNN used traditional CNN methodology to avoid overfitting. Both algorithms avoid underfitting using provided dataset.

YOLO v5 and Faster RCNN require images with annotation to train the model as part of preprocessing and multiple preprocessing can be performed for both models to enhance the quality of data and increase the accuracy of the model but, where YOLO v5 uses internal augmentation of images before start training the data and augmentation can be done externally before pass the image data to train in Faster RCNN to enhance the performance of the model. On a Linux computer with an Intel 5930K, two NVIDIA GeForce GTX 1080 GPUs, and 32GB of RAM, training and testing will be used for 5000 pictures in both models. Whereas Faster RCNN takes 155% more time than YOLO v5 to train the model. The weight file size of Faster-RCNN is almost 3 times larger than the YOLO v5. The study conducted by the author found that

confidence coefficients obtained by YOLO v5 were quite higher than the Faster RCNN. In terms of training and inference, YOLO v5 is known to operate at a faster rate. According to the author's analysis, training the model took less than half the time of Faster RCNN, however, the size of the weight file was more than twice as large for YOLO v5 (Fang et al., 2021).

The strength of YOLO v5 is the speed of training and inference. A region of interest is provided by faster RCNN for convolution to enhance the accuracy of the model prediction. The Limitation of Faster RCNN is slower in training and inference and YOLO places severe spatial restrictions on bounding box predictions, including grouping together of tiny objects. It still has trouble extrapolating to things with novel or unique aspect ratios or arrangements.

Model Evaluation Methods

F1-Score

The research makes reference to a wide range of evaluation metrics. To objectively assess the performance, the output of the detectors was compared to the manual annotations. The F-Score for overall accuracy, recall rate, and precision were determined.

Recall

The model's capacity to identify every pertinent item or identify every bounding box that was identified in the training set. The recall is calculated mathematically as the product of the number of true positives divided by the sum of the true positives and false negatives. (1) depicts the formula of Recall.

Precision

The model precision refers to its capacity to recognize just the pertinent items. (2) is the formula for calculating the recall. The first harmonic mean between recall and accuracy is the

F1-score which can be seen in (3). The accurate detections of the ground truths are known as True Positives (TPs).

Table 4

Comparison of YOLO v5 and Faster RCNN

Characteristic	YOLO v5	Faster RCNN
Architecture	one-stage object detection	two-stage object detection
Data Type	Image, Video	Image, Video
Small Objects	Can be detected	Fail to detect
Overfitting/Underfitting	Uses Internal augmentation	Not used Augmentation
Preprocessing	Resize, Annotation etc.	Resize, Augmentation, Annotation etc.
Training time	Three times less than Faster-RCNN	Higher than YOLO
CPU	QTY 2 vCPUs	QTY 2 vCPUs
GPU	GPU to NVIDIA P100 or T4	GPU to NVIDIA P100 or T4
Size of the weight file	Higher than Faster RCNN	Less than YOLO v5
Strength	High speed	Uses region of interest
Limitation	Fail to detect tiny object	Slower

False Positives (FPs) are things that were incorrectly identified as being present. The undiscovered items are False Negatives (FNs). The total of the TPs and FNs (TP + FN) can be

used to calculate the number of ground truths, and the sum of the TPs and FPs (TP + FP) can be used for calculating the number of detections.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (1)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2)$$

$$\text{F}_1 = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

The confidence rate was still a free parameter for all the models at the conclusion of the training process. The certainty of the made forecast is reflected in the confidence rate, which is a number. If a prediction has a 50% confidence level, it means that the network has a 50% confidence level in the identified or categorized item. As a result, improving the confidence score is crucial for improving network performance. To optimize this number, the cross-validation method is helpful. We eliminated all augmentations from the validation set and calculated the F1-score in increments of 1% for all confidence criteria ranging from 0% to 100%.

A confidence threshold takes into account all confidence rates that are greater than or equal to the given value. For the model's typical functioning, the confidence level that optimizes the F1 score is used (Magalhães et al., 2021).

Confusion Matrix

By computing statistical metrics such as the True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives, one may determine the accuracy of document categorization systems (FN). As seen in Figure 8, these elements combine to generate the Confusion Matrix.

A table that may be created for a classifier on a binary data set and used to describe the performance of the classifier is known as a confusion matrix (Maria et al., 2016).

Figure 10

Confusion Matrix

	Predicted: No	**Predicted: Yes**
Actual: No	TN	FP
Actual: Yes	FN	TP

Loss Function

Figure 11 shows the loss in logarithms. The real class's intended output, which can be either 0 or 1, is compared to each class' projected probability, and a score/loss is calculated that penalizes the probability depending on how much it deviates from the actual expected value. Significant differences close to one receive a big score due to the penalty's logarithmic nature, but tiny differences close to zero receive a small score. During training, the cross-entropy loss is used to modify the model weights. Since loss reduction is the main objective, a better model will result in a lower loss. A perfect model has a cross-entropy loss of 0. (Stoyanov et al., 2018).

Figure 11

Cross Entropy or Loss Function

$$\mathcal{C}(\theta) = \mathbb{E}_{x \in \mathbf{X}} \left[-y \log(p) + (1-y)\ log(1-p) \right],$$

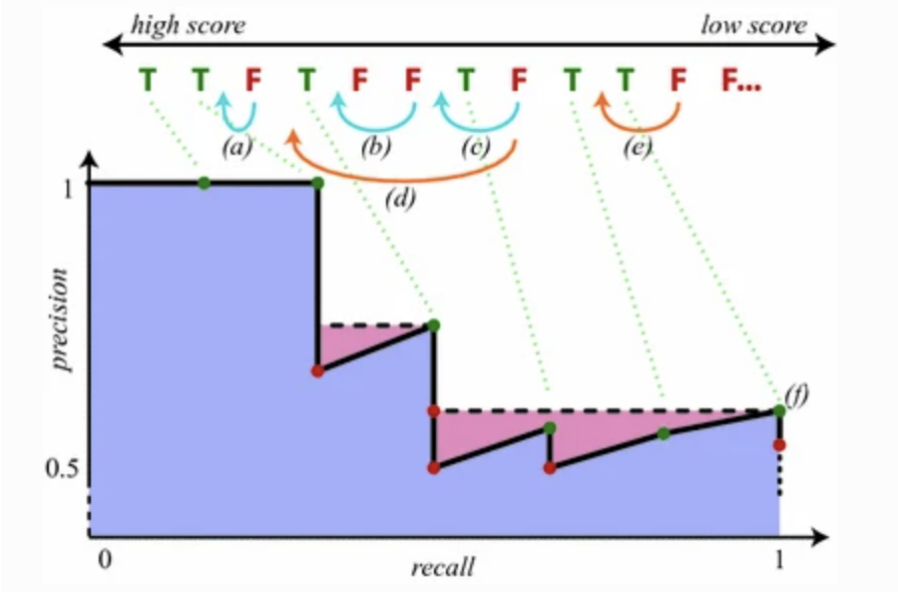
Precision/Recall Curve

Figure 12 describes that, for some object classes with six ground-truth instances, the precision/recall curve (bottom) for a series of true-positive (TP) and false-positive (FP) detections are ranked by score (top). The black curve results from plotting the accuracy and

recall numbers in order. The pink area displays the outcome of substituting the maximum recall at the same or higher precision for each precision. The combined area of the pink and blue zones is known as AP. The arrows (a-e) depict how positive perturbations affect the results of FP detection scores (Henderson et al., 2017).

Figure 12

Precision/Recall Curve



mAP

Likely abbreviated as mAP, mean Average Precision The AP, also known as the area under the precision-recall curve, gives a measure of quality for single class categorization across all recall levels. The mean of the APs in a multi-class classification is thus known as the mAP.

Model Validation and Evaluation

The optimal confidence threshold for the models needed to be established initially. Because it achieves the optimum compromise between accuracy and recall, maximizing the number of TPs while minimizing the FPs and FNs, this value is the confidence threshold that maximizes the F1 score. Figure 13, 14, and 15 show how the F1 score, precision, and recall

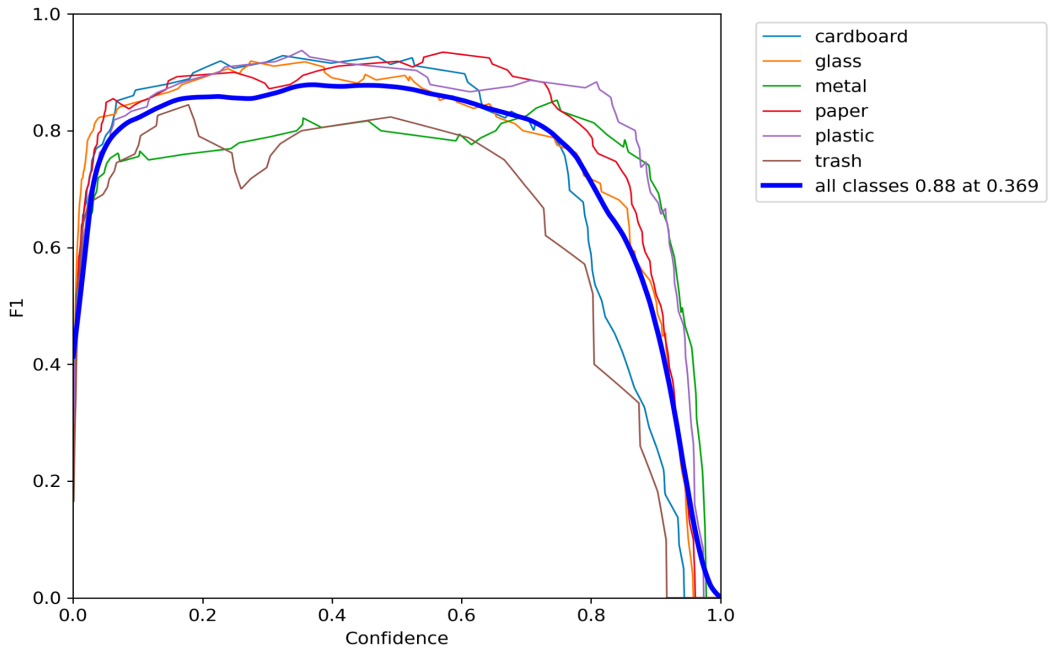
changed when the confidence level changed. Each color curve represents the specific class and the dark blue curve is representing all class average performance during validation. We may readily conclude from the given evaluation that certain classes behave better than others. Classes with flattened curves exhibit a low number of FPs and FNs and better confidence in their predictions.

Figure 13 illustrates that the trash class has the lowest F1 score across all classes. Figure 11 shows that, in comparison to other classes, the metal class had the lowest accuracy. Figure 14 demonstrates that the trash class has the lowest accuracy and confidence.

A smooth accuracy recall curve was constructed using all the predictions for finding classes in pictures of waste in Figure 15. With the development of the prediction confidence score, this curve created a compromise between the recall rate and the accuracy rate. Higher confidence levels often have more accurate forecasts with a lower recall rate.

Figure 13

Evolution of the F1-score with the Variation of the Confidence Threshold for all Classes.



The tradeoff between accuracy and recall for 0.5 thresholds is depicted by the precision-recall curve in Figure 16. High accuracy is correlated with a low false positive rate,

Figure 14

Evolution of the precision with the Variation of the Confidence Threshold for all Classes.

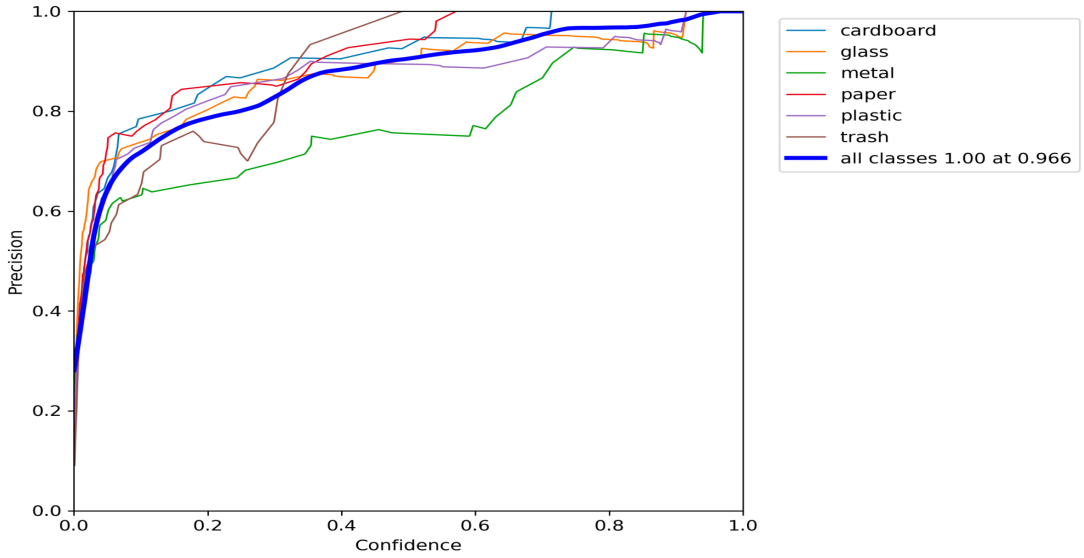
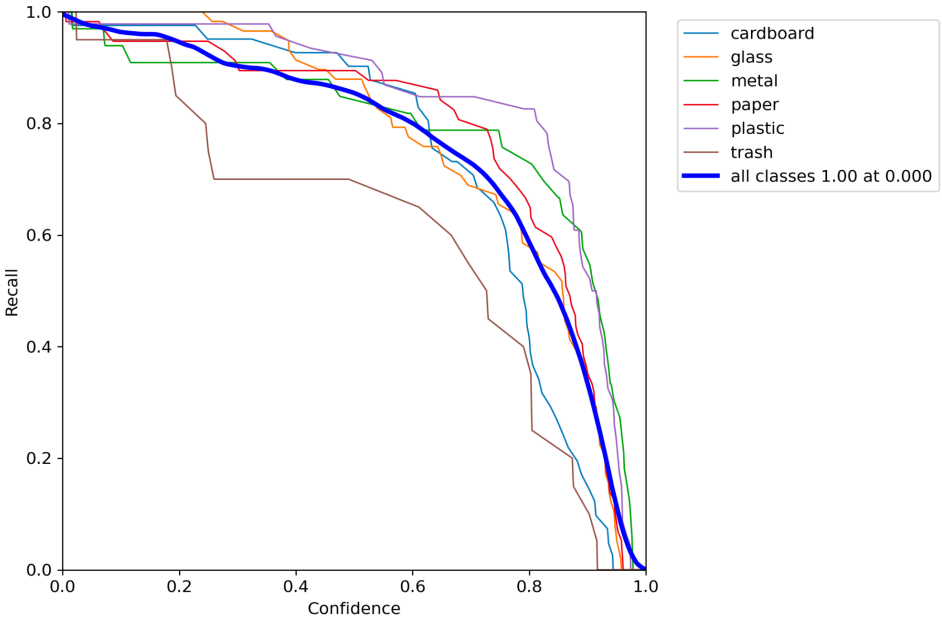


Figure 15

Evolution of the Recall with the Variation of the Confidence Threshold for all Classes.



while big recall is correlated with a low false negative rate. A high area under the curve denotes both high recall and high precision.

Figure 16

Precision-recall Curve in the Test Set Using the Calibrated Confidence Threshold.

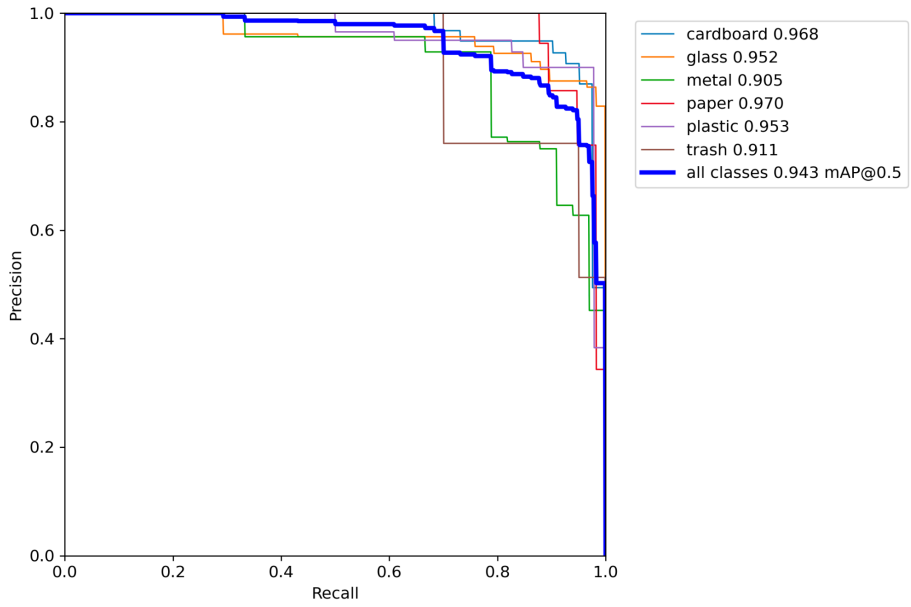
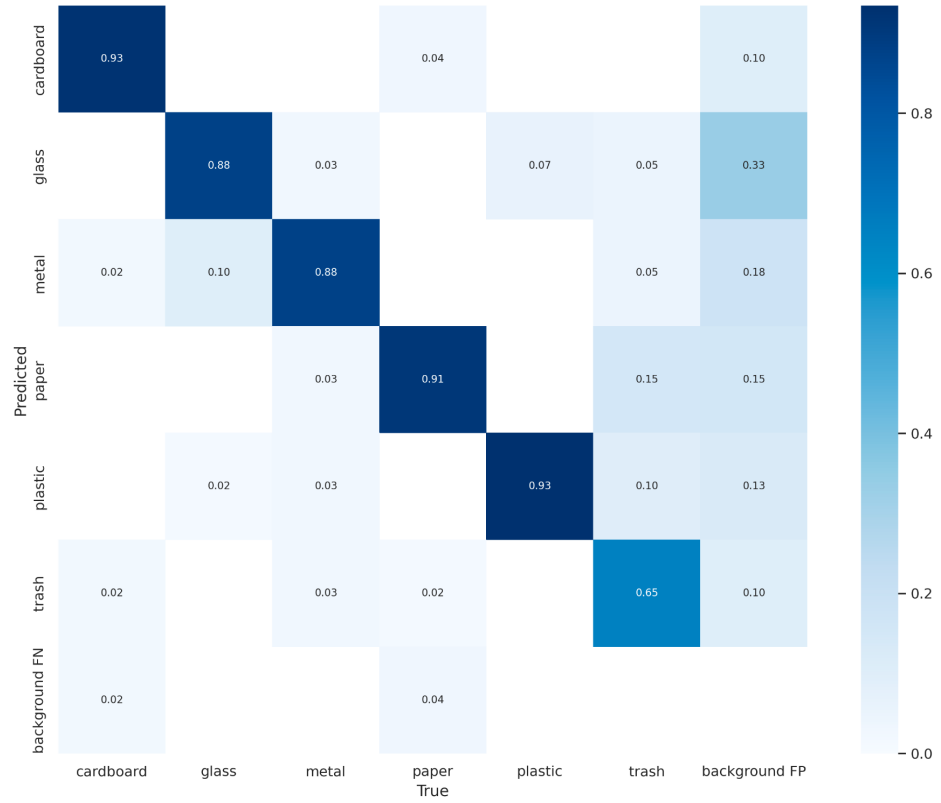


Figure 17 illustrates the test results are examined using the confusion matrix. A confusion matrix is a list of data classes, with the real data being categorized into each class so that we can see which categories of samples in the modified network are most likely to be confused. The rows in the confusion matrix correspond to the actual categories for the test images. The columns display the test picture classes that were split up by the network during the real test. The network model's confusion matrix for the provided dataset is shown in Figure 17. It shows that cardboard is incorrectly classified as paper and the original paper samples are classified as trash. glass is classified as 0.03% waste, followed by 0.07% plastic and 0.05% paper. When trash makes up 0.05%, cardboard is 0.10%, glass is 0.02%, and metal is 0.02%.

Plastic is seen as waste and metal. Figure 17 shows that trash has the lowest accuracy and is identified as cardboard and paper. As a consequence, the result reveals that cardboard has the highest real predicted class, whereas trash has the lowest.

Figure 17

Confusion Matrix of Classes



In order to change the model weights during training, the cross-entropy loss is utilized. The goal is to reduce loss; hence, the better the model, the less the loss. A cross-entropy loss of zero indicates a flawless model. Figure 18 illustrates the loss during training and validation in terms of the box, object, and class loss. Where the box_loss graph is depicting the losses of the box or can be called a bounding box of the object in the image through which the model can detect the position of the object in the image. Adjacent to that Obj_loss graph is representing the loss of an object in the image inside the bounding box. Lastly, the cls_loss graph is representing

the loss of the class of the bounding box in the image that is mainly responsible for training the model and classifying the object as a class.

Figure 18

Object, Class, and Box loss in Training and Validation

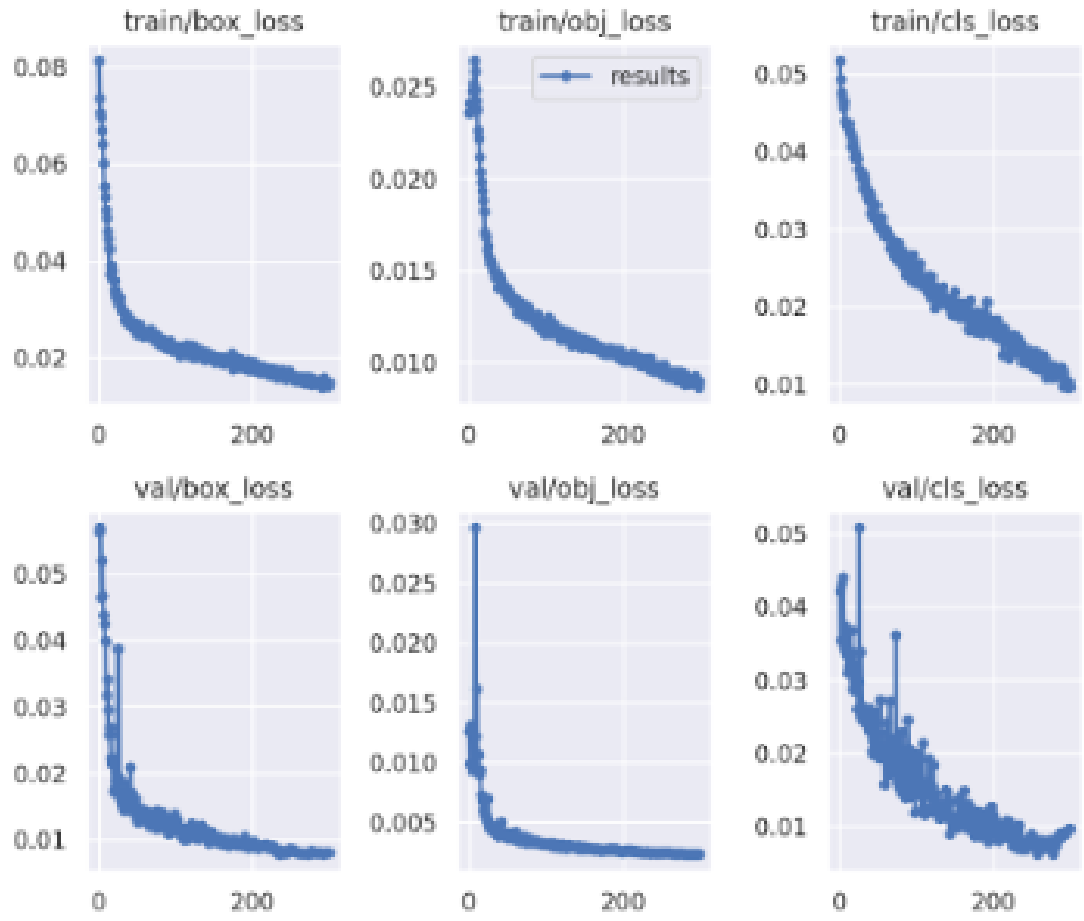


Figure 19 show the mAP of the YOLO v5 model where it is depicting the mAP of 95 % up to 300 epochs. It shows a measure of quality for single-class categorization across all recall levels.

Figure 20 shows the data flow during training of the model that is captured by tensorboard, which is used in this study to evaluate the results. It is depicting the flow of the dataset through input to output. Where it is showing all the levels of YOLO v5 and layers

including SPP, Convolution, and concatenation that is discussed in the YOLO architecture section of this paper.

Figure 19

YOLO v5 Graph of mAP

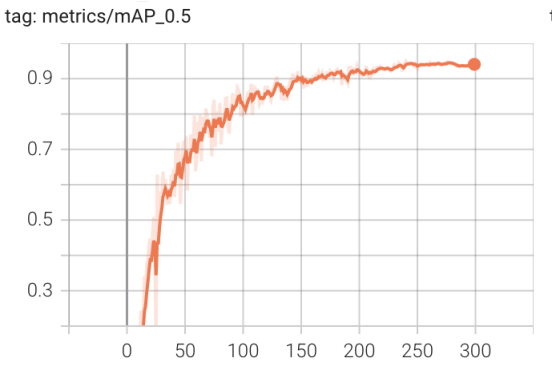


Table 5 illustrates the comparison of the accuracy achieved by YOLO v5 and Faster RCNN with different level parameters. Figure 21 is depicting the logs of YOLO v5 with 100 epochs and the result represented in table 5 for Epoch 100 is calculated from that.

Table 5

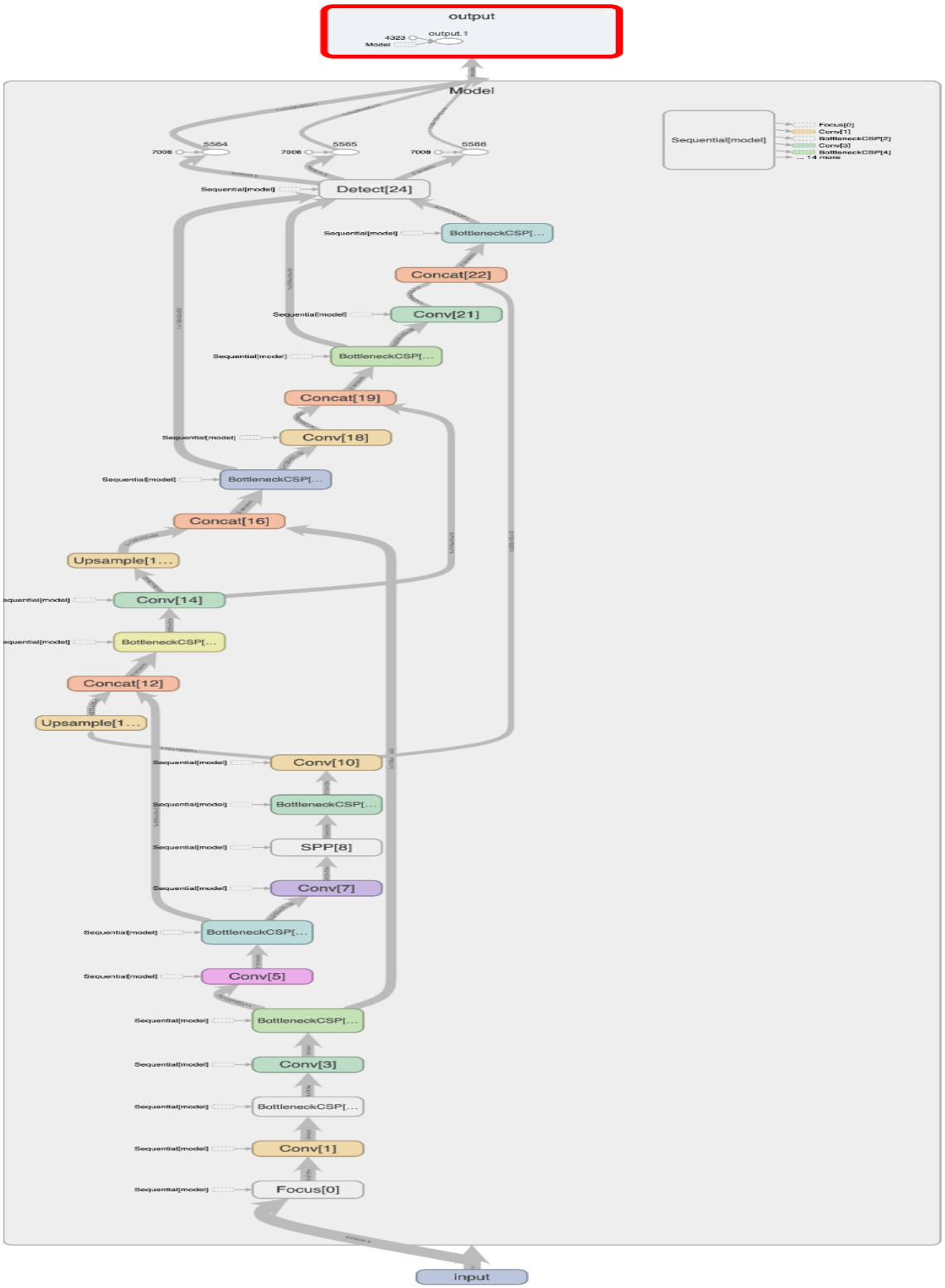
Comparison of YOLO v5 and Faster RCNN

Model	mAP	Precision	Recall	F1-Score
YOLO v5	95%	96%	94%	88%
Faster RCNN	74.2%	73%	70%	67%
100 Epoch YOLO v5	80%	72%	86%	76%
300 Epoch YOLO v5	94%	95%	94%	87%

Figure 21 illustrates some of the images predicted by the YOLOv5 model after training and validation. It can be seen in Figure 21, that model is predicting images with higher accuracy.

Figure 20

Data flow of Model During Training



As Figure 21 depicts the glass image is predicted by the model with 91% accuracy the paper image model was predicted with 94% accuracy and the metal object gave 95 % accuracy.

Figure 21

Logs for 100 Epoch

```
custom_YOLOv5s summary: 232 layers, 7260003 parameters, 0 gradients, 16.8 GFLOPs
                 Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 8/8 [00:03<00:00,  2.36it/s]
                   all        252        255      0.722      0.859      0.862       0.72
             cardboard        252         41      0.825      0.927      0.915      0.722
                 glass        252         58      0.727      0.897      0.883      0.726
                 metal        252         33      0.683       0.97      0.851      0.804
                 paper        252         57      0.706      0.877      0.874      0.725
               plastic        252         46      0.719       0.87      0.898      0.788
                 trash        252         20      0.671      0.611      0.754      0.557
Results saved to runs/train/yolov5s_results
```

Figure 22

Classes predicted by the model

Conclusion. This study proposed an automated waste identification framework utilizing deep learning algorithms and image processing techniques to reduce the effect caused by improper garbage disposal. With a huge collection of photos, training methods, and predicative patterns for object recognition and classification, the framework was able to be put into practice. In this study, we show how to use the Faster R-CNN algorithm to classify waste items into six categories (Cardboard, Metal, Glass, Paper, Plastic, and Trash) based on a number of objects in a single image. Object detection includes the fundamental processes of object segmentation, localization, and classification. This study employed the single-stage YOLO v5 algorithm to

identify recyclable and non-recyclable garbage and compared the results with the two-stage Faster-RCNN method. The architectural advances of single-stage object detectors, in particular YOLO v5s, as well as the underlying pretrained CNN architectures were thoroughly examined in this paper. This research also covered the YOLOv5 dataflow comprehensive investigation. Therefore, this study also provides the list of hardware, software, and tools used to perform this research according to the requirements of YOLO v5. This study explained the various evaluation methods to evaluate the models. After evaluating the model it clearly can be concluded that In terms of detection accuracy and inference time, YOLO v5 greatly outperforms their two-stage equivalent object detectors. Compared to the Faster RCNN model, YOLOv5s produced the highest mAP at 96% with a respective FPS of 106.4 which remarks an outstanding model for the detection of garbage objects in images.

Limitations. The dataset for this research, which contains photographs of regional waste products that are slightly diverse, was the primary problem. This is the cause of the model's inaccurate predictions on a few local garbage photos. Images of the trash and other waste items in the training dataset that are unclean and appear dirty must be attached. This will aid the model's ability to anticipate local waste materials, which primarily consist of filthy domestic goods. In doing so, it may be possible to obtain improved categorization with a greater accuracy level. So that the framework can easily be trained to predict many things in a single image without giving any mistakes in the detection process, the dataset should also comprise many photos that contain images of various waste products.

Future Scope. The inclusion of other sorts of other bulky trash categories in the dataset should also be considered for future studies on this topic. This framework will become more

developed and unquestionably contribute to the enhancement of an effective waste management procedure.

Applying each model's various classification methods to the framework independently may be done for this analysis, which can then be used to determine which model is optimal for obtaining faster and more accurate item identification and prediction. In order to safeguard the environment and lessen pollution, future work also seeks to integrate this technology into a mobile platform so that it is extremely simple for the user to categorize waste items and dispose of the waste materials in the appropriate disposal bin.