CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

Deep Learning-Based Mosquito Quantification to determine Repellency of different compounds

A thesis submitted in partial fulfillment of the requirements

For the degree of Master of Science in Computer Science

By

Gaganpreet Kaur

December 2022

The thesis of Gaganpreet Kaur is approved:

Robert Mcilhenny, PhD	Date
George Wang, PhD	Date
Katya Mkrtchyan, PhD, Chair	Date

California State University, Northridge

Table of Contents

List of Figures

List of Abbreviations

DEET	N, N-Diethyl-meta-toluamide
BA	butyl anthranilate
4MPD	4-methylpiperidine
YOLO	You Only Look Once
SGD	Stochastic Gradient Descent
CSP	Cross-stage partial connections
FLOPS	Floating Point Operations Per Second
SVM	Support vector machine
CNN	Convolutional Neural Network
FPN	Feature Pyramid Network
PAN	Path Aggregation Network

Abstract

Deep Learning-Based Mosquito Quantification to determine Repellency of different compounds

By

Gaganpreet Kaur

Master of Science in Computer Science

Insect-borne diseases are a worldwide health problem, especially in tropical and subtropical climates. Mosquito repellents may effectively protect humans from mosquito borne diseases. This thesis presents a deep learning approach to test the efficacy of mosquito repellents. The compounds like DEET, acetone, 4-MPD are compared with solvents like water and acetone in experiments conducted by research lab under same conditions. The mosquito species used in this experiment is female Aedes aegypti. The setup uses similar heat and humidity conditions which attract mosquitoes to human.

In this thesis, mosquitoes are counted in regions sprayed with different compounds which is used to compare the repellency. Performance of the proposed model is evaluated using a custom mosquito dataset built upon image frames which are collected from the video dataset provided by the research laboratory. The dataset has been labelled and annotated manually using Roboflow. The quantification is done using YOLOv5 algorithm and comparison of different models like YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5l is done. YOLO algorithm is chosen because of its small object detection capability and less computing time.

The mean accuracy precision(mAP), precision and recall are used as performance measurement tools to evaluate the detection accuracy of the proposed model. . The inference time, training time and all other metrics for YOLOv5m are better for the provided dataset, therefore all the videos of experiment conducted are run on this model and compounds are compared. The experimental results show that accuracy of more than 90% is achieved with all types of YOLO models. 3% DEET shows the best repellency than all other tested compounds. There is not much difference in efficacy of 1% and 3% DEET but 4% MPD and 3% BA doesn't show great repellency against mosquitoes.

1. Introduction

The worldwide threat of mosquito transmitted diseases and their associated morbidity and mortality, underscores the need for effective insect repellents. Multiple chemical or plant based repellent products are marketed to consumers Repellants are being widely used by humans to protect themselves against insect borne diseases. DEET (N, N-Diethyl-meta-toluamide), an effective and safe repellant was developed during World War II. But this repellant has a limited spatial zone of protection.

Protection from insect bites is best achieved by avoiding infested habitats, wearing protective clothing, and using insect repellents. In many circumstances, applying repellent to the skin may be the only feasible way to protect against insect bites. Given that a single bite from an infected arthropod can result in transmission of disease, it is important to know which repellent products can be relied on to provide predictable and prolonged protection from insect bites. We sought to determine which products available provide reliable protection from mosquito bites.

Presently, there are only small number of active ingredients which are present in most of the commercial formulations that are available to protect humans from mosquito biting. DEET (N, N-diethyl-m-toluamide) is the most widely used repellent which is being used now for almost 70 years. But fear of possible side effects and less efficiency of this compound has resulted in development of new repellents. In this thesis, we will observe the Aedes aegypti mosquitoes (Orlando Strain i.e., mosquitoes found in Orlando) behavior in presence of various repellents in comparison to one another.

The experiment is conducted using different repellant compounds in various concentrations like DEET vs Acetone, BA vs Acetone, 4MPD vs water and data is collected. The collected video dataset will be used to quantify the mosquito behavior with different test compounds using image processing. These products were tested in a controlled laboratory environment in which the species of the mosquitoes, their age, their degree of hunger, the humidity, the temperature, and the light–dark cycle were all kept constant.

The objective of experimental dataset collected is to determine rankings of different mosquito repellant compounds, that is, to find out which one is the most effective by observing landing behaviors of the mosquitoes. We will show the mosquito behavior by quantifying mosquitoes in presence of different test compounds using deep learning algorithm. The data from video will be collected to form dataset of the mosquitoes. Feature extraction is required to extract the mosquitoes in region of interest. Counting the number of mosquitoes in ROI(region of interest) at different intervals will be plotted and average number will be calculated using YOLO algorithm.

Object counting is commonly used in many fields, but it becomes challenging when the objects in the images are small and are not easily distinguishable due to noisy background. In our experiment, the area with mosquitoes is surrounded by mesh which makes it difficult to subtract the noise reduction and mosquitoes are not clearly distinguishable. The model is trained in such a way that it detects the mosquitoes in that particular setup, as it detects blurry mosquitoes from the video, and it also affects accuracy to some extent too which is calculated in the result section. Our method can help the biologist to compare different compounds which was otherwise being done manually by counting the mosquitoes

YOLOv5 is an algorithm with high reliability and stability, and it is easy to deploy and train. At the same time, it is also one of the one-stage detection algorithms with the highest accuracy at present; therefore, in this thesis, we choose YOLOv5. The chosen YOLO algorithm provides the result which are better and faster than other object detection algorithms such as faster RCNN.

2. Background

Mosquitoes blood feed on humans and transfer pathogens from one host to another. They spread deadly diseases which cause epidemics that affect hundreds of millions of people and there are immense efforts taken internationally to combat this. Tropical and sub-tropical regions of the developing countries are at major risk.

Mosquitoes use combination of cues to search for the host. The host seeking behavior of mosquitoes primarily depends on their sense of smell. They sense the presence of human through odors emanating from their bodies through olfactory receptors. Heat and humidity are other factors responsible for attraction. Mosquitoes tend to land on surfaces that have temperature close to human bodies [3]. Mosquitoes also express different degree of preference for humans which is associated with odor profiles which differs between men and women as well as people of same sex [15,16]. Certain mosquito species like Aedes aegypti, Anopheles gambiae, have evolved a preference for humans, which makes them efficient vectors for disease transmission [1].

The sense of smell of mosquitoes and their behavior according to that has been studied extensively. Different studies have shown that if different odors are used on human skin or human skin like setup, the mosquito attraction or repulsion to odor is different. Understanding of different pathways and cues responsible for human attraction to mosquitoes have led to development of different compounds that can help to distract mosquitoes away from humans. Based on this behavior the substance was called “attractant” if it encourages mosquito to make flight toward source and “repellent” if mosquito moved away from source. Our experiment was conducted with Ae. aegypti in a caged assay where different compounds were tested.

Despite considerable efforts in recent years to control mosquito borne diseases, malaria alone produces 250 million cases per year and 800,000 deaths including 85% children under five years (WHO, 2010) [2]. Reducing host-vector interaction will reduce these vector-borne diseases. Therefore, the use of mosquito repellents on skin is highly recommended to prevent these diseases at first place. Mosquito repellents are widely used against protection from mosquito borne diseases. Although these compounds doesn’t guarantee you complete protection but still can lessen the chances of vector borne diseases. They are useful in outdoor activities like fishing, camping etc. where activity patterns of mosquitoes can be found.

Mosquito repellents are an effective way to prevent mosquito bites and reduce the spread of mosquito-borne diseases. In the early 90s, the U.S. Environmental Protection Agency (EPA) published a list of active ingredients that pose minimum risk to human health that can be used as pesticides or repellents without passing the EPA registration process [17]. There are only small number of active ingredients which are present in most of the commercial formulations are available to protect humans from mosquito biting. DEET (N,N-diethyl-m-toluamide) is the most widely used repellent which is being used now for almost 70 years. But fear of possible side effects and less efficiency of this compound has resulted in now development of new repellents. In this thesis, we will observe the mosquito behaviors in presence of two different repellents like DEET vs Acetone, BA (butyl anthranilate) vs Acetone, 4MPD(4-methylpiperidine) vs water. The experiment was perfomed on Aedes aegypti mosquitoes (Orlando Strain, collected from Orlando,Florida) .

DEET is widely used but it has to be applied over all exposed areas for protection against mosquitoes. There has been identification of strong agonist 4MPD(4-methylpiperidine) which has much higher vapor pressure than DEET. Research shows that even if it is applied to one part of human body, it will show repellency towards mosquitoes on other parts too.

Observing mosquito flights with these different substances, we will quantify the mosquito behavior and rank the compounds based on their repellency. This will be done using deep learning methods.

3. Related Work

It is a known fact that mosquitoes are attracted to humans and seek for reproduction. They can find humans through different cues. Many research have been conducted to find out what draws mosquitoes' attraction to humans. Research by Biomolecular Sciences Institute & Department of Biological Sciences has studied about multiple cues that attract mosquitoes to humans. According to their studies, mosquitoes use combination of cues like heat from human body, CO2 emissions, OR pathway receptors, humidity etc. to identify the host. Such studies have been helpful in finding the effective repellents against these insects[6].

There has been a lot of studies done to test for a better insect repellent and with different concentrations. One of the similar research has been done using DEET (N,N-diethyl-3-methylbenzamide, diethyltoluamide) and picardin ((2-(2-hydroxyethyl)-piperidinecarboxylic acid 1-methyl ester) to test for their efficacy based on the assessment of which repellent lasts longer in protection from mosquitoes. Both compounds were tested using arm in cage experiment and it was found that picardin showed better repellency but there wasn't much difference in comparison to DEET when applied at the same dosage. It showed better results when higher concentrations of the chemicals were used. This research said that if both DEET and picardin were applied in same concentration, they wouldn't differ in duration of protection they would provide[4].

In another research, comparison of eight different commercially available repellents with and without DEET were done with two different types of disease spreading mosquitoes. The repellent products were tested by applying on human hand and hand was put in a y-tube olfactometer with Aedes Aegypti and Aedes Albopictus mosquitoes in the tube. This study showed that different repellents were not equivalent in terms of duration and strength. Some of the DEET free repellents did not have any significant repellency. In summary, the results of this study show that not all commercially available mosquito repellents are effective in repelling mosquitoes and that efficacy is also dependent on the species of mosquito that is repelled. Overall, the results from this study confirm that DEET and p-menthane-3,8-diol repellents are the most effective mosquito repellents in the market [5].

The above all research are done by biologist from various research centers. There are research done in quantification of three different types of pests (aphid, thrips and whitefly). One of the research projects done by The Research Center for Coastal Environmental Engineering and

Technology of Shandong Province in recognition and quantification of pests using low computational methods. They used watershed algorithm for image extraction from background and used Mahala Nobis distance to identify different species by using color features of insect. The count was compared with human counting. This research is different from ours as we have to identify mosquitoes, but we can follow the similar approach for feature extraction for building our dataset [7].

For automatic deep learning-based classification of insects, a lot of other studies have also been used which used different architectures of CNN which has given accurate performances of more than 90%. There is one similar research done in National Institute of Technology, Tiruchirappalli, India, in which they conducted experiments for classification of different insects and comparing various machine learning models like artificial neural networks (ANN), support vector machine (SVM), k-nearest neighbors (KNN), naïve bayes and convolutional neural network (CNN) model. For insect detection they used foreground extraction and contour identification. They achieved highest classification rates of around 90% using CNN models [8]. Very similar research has been done using faster region based convolutional neural network (faster R-CNN) for recognition of insects based on cloud computing. It was developed for famers as a smart agriculture approach to automatically classify pest using deep learning approach and look for methods to protect the crops. This study implemented algorithm which is real time and can be used online too for recognition of agricultural pests. This proposed algorithm showed recognition results of 99% for all tested images [9].

Mosquitoes are small species and very difficult to detect unlike other insects. They don't even have visible features that can be extracted for recognition. There is one research done in detection of mosquitoes using their wing movements. This study uses the fact that wingbeats of different mosquito species is different which can be used to identify them. The identification is carried out using devices which can record the wing movement patterns of mosquitoes. In this research they have compared multi-layer CNN, resnet (residual neural network), denseNet (densely connected convolutional network) and XGboost (extreme gradient boosting) machine learning models for classification. The best result in achieved by multi-layer CNN with accuracy of 86%. [10]. Another research with same method used to classify mosquitoes and comparing CNN and denenet has also been done [11].

Another research done in agricultural sector for locating and counting insect pests on crop using deep learning methods is based on self-saliency feature maps. It uses combination of CNN, region proposal network (RPN) and non-maximums suppression to remove overlapping insect detection. The method used in this study proposes deep learning system which can be used to identify very small pests too. It also explores different feature extraction methods like AlexNet, ResNet and ZFNet. The mean accuracy precision achieved was 0.885[12].

4. Experimental Setup

A pair of heat source (Figure 2) is prepared using hand warmers (Hot hands® Hand warmers HH2; Heat Max, Dalton, GA). Temperature of 37 degree Celsius is maintained as it is the normal human body heat temperature to attract mosquitoes. These heat sources are covered with 100 x 15mm petri dish (Figure 1 (a)) and then covered with polyester netting from top. The heat source is covered using another 150 X 15 mm petri dish (Figure 1 (d)) base and window opening of 10 x 7.5 cm is cut out of it. Another pair of petri dish 150 X 15mm is also placed on side which is used to support the cage. Pieces of net treated with solvent and test compound is placed between two 10 x 7.5 cm flexible magnets (Figure 1(b)) with 7.5 x 6.2 cm window frame. The test cage of mosquitoes in then gently set over the 2-choice arena. The heat was the attractant to mosquitoes, so they were exposed to one of the test or solvents in the cage. The setup is shown in detail in Figure 1 and Figure 2.

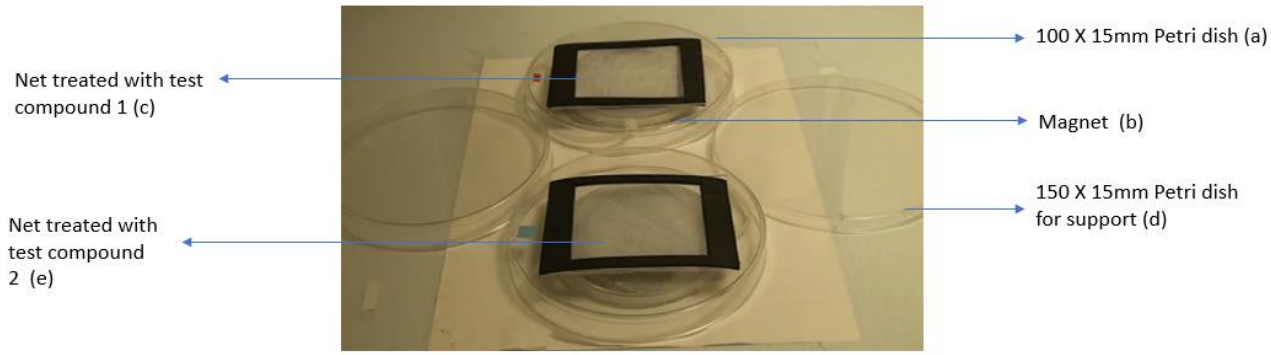


Figure 1: 2-choice arena placed in test cage with 40 females Aedes aegypti mosquitoes

The experiment setup has been done with keeping in mind all the cues and circumstances under which mosquitoes are attracted to their host. The heat and humidity conditions are tried to match as that of humans. Also the setup has been enclosed with lower density of mosquitoes which mimics the outside conditions for mosquitoes to search for their host.

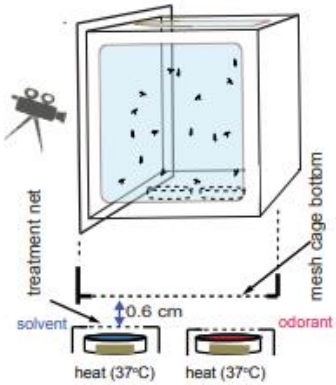


Figure 2: Schematic of 2-choice heat attraction experiment where two Petri dishes each other with approximately 37 degrees heat source and a net covering which is treated with test chemicals, is placed 6mm below a cage containing approximately 40 mosquitoes

5. Methodology

The process consists of collecting the data first, which is already provided with experiment. We have different video data sets with different tests compounds. We will need to do image acquisition first to get the mosquito images from our dataset and form a training dataset. Our algorithm should detect landed mosquitoes in that specific time frame, which needs the feature extractor model to identify the mosquito from background. Finally, we need to classify and count the total number and average it over the total video frames.

The main steps are as follow:

Image acquisition: We need to get a clear image of mosquito so that our model can recognize the mosquito.

Detection and Training Dataset: The detection algorithm should have ability to eliminate the interference that might be caused in a video dataset due to various reasons like, out of focus, illumination, or any other impurity. Our deep learning method should be detecting the mosquitoes with accuracy and provide it for classification, which will be used to form the training dataset.

Feature extraction: It is used to find key features that can reflect mosquitoes. Extracted features should have some similarity while identifying mosquitoes, like shape, texture, color etc. It is possible that different part of image might have same features, so it's important to import variety of features.

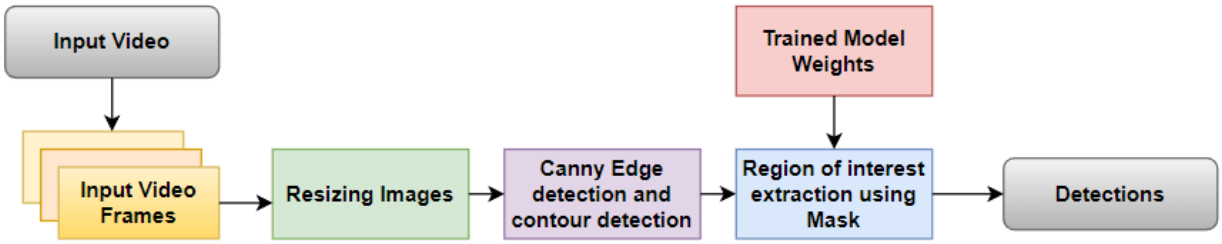


Figure 3: Framework of proposed method

5.1 Detection Algorithm

In this section, we represent the implementation procedure of the algorithm. This research is aimed at training the YOLOv5 object detection classifier for mosquito detection. The diagram of

workflow for complete mosquito detection procedure is presented in Figure 4 which consists of the following blocks: data pre-processing (data acquisition, data annotation), data processing, training and testing the deep classifier.

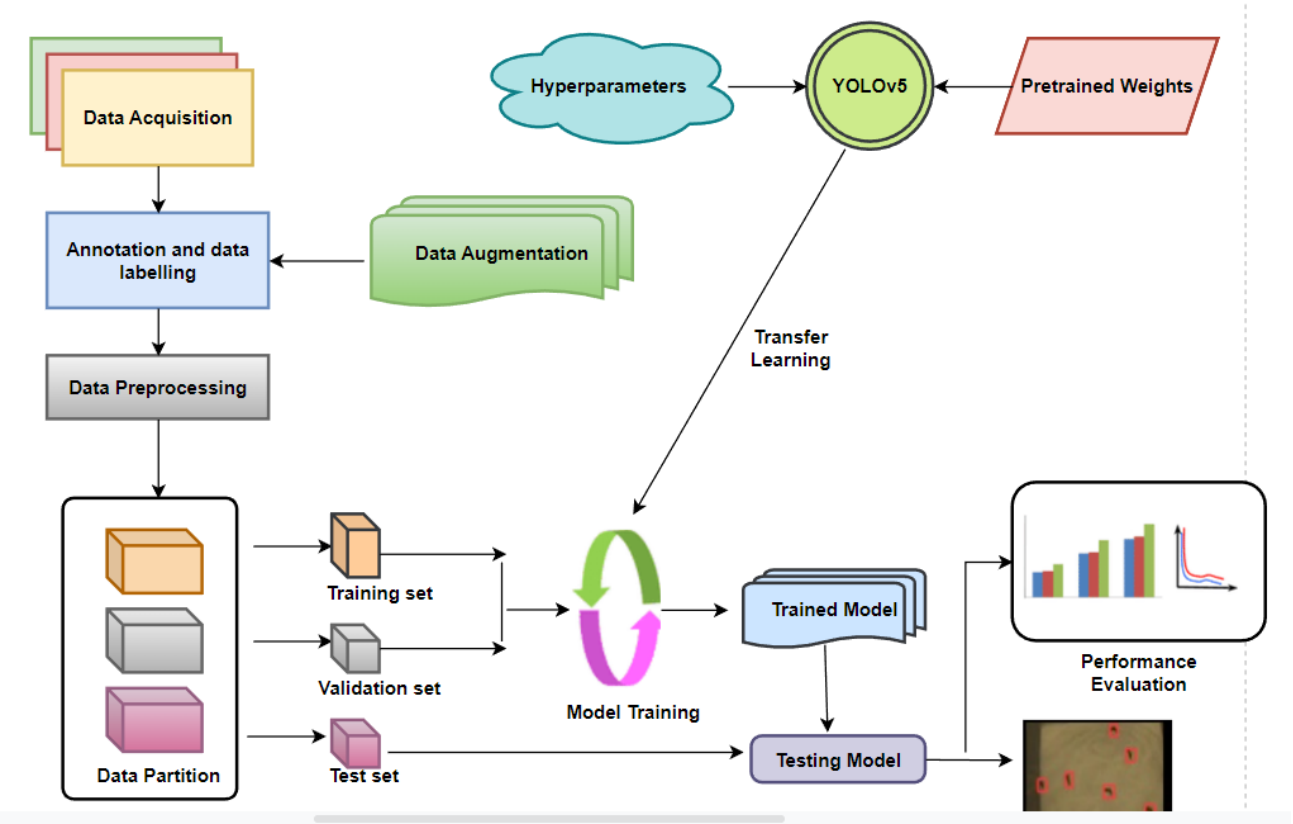


Figure 4: Diagram of workflow training model of whole detection procedure

5.1.1 Data Acquisition

Data has been obtained from the videos shared by the research laboratory. The videos are transformed into images to collect mosquito dataset. As mosquitoes are moving fast in the video and the picture quality is also not good enough to identify mosquitoes from their features, images of mosquitoes used are same as they appear in the video. To make our model learn properly about mosquito features in our video, 1000 images are annotated with different positions of sitting or flying mosquitoes. Samples of the dataset is shown in Figure 5. After the data acquisition process, the images have been reformatted to a predefined size of 416 X 416.

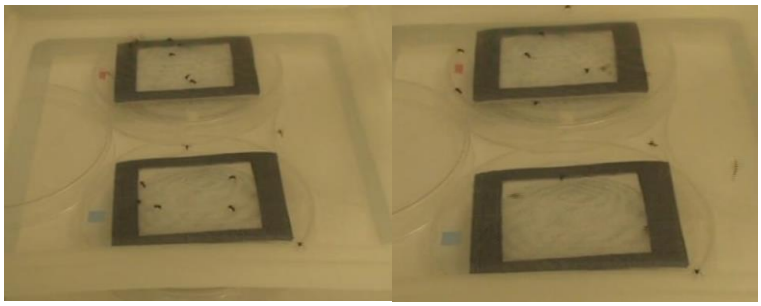


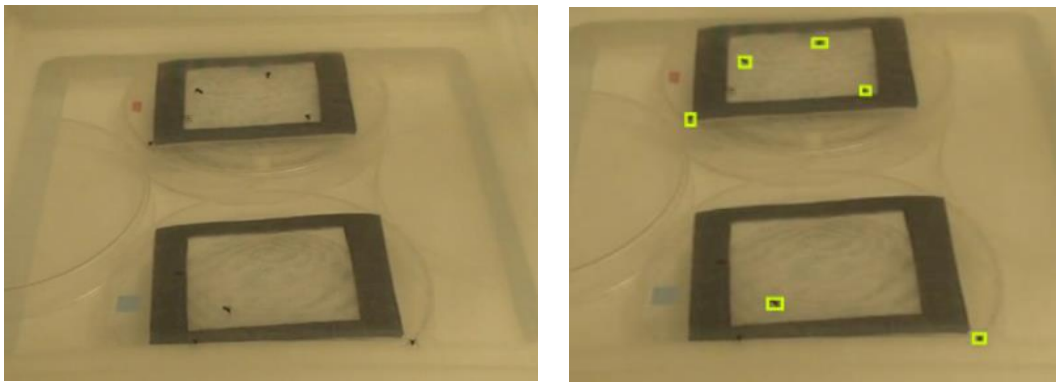
Figure 5: Sample dataset images collected from video dataset

5.1.2 Data Annotation

This is a method of identifying the appropriate data in different formats, such as text, video, or images. Roboflow has been used to create dataset that has been used in this project. Roboflow is a computer vision developer framework where custom data can be uploaded, annotated with different labels which can then be converted into various formats like COCO JSON, YOLO, Pascal VOC etc. after data preprocessing. There are public datasets available in the framework too which comes with annotations.

The images in this project have been extracted from video dataset available and labelled using Roboflow. Labelling is basically drawing bounding boxes to get the notations of exactly where the object is in the image. Mosquitoes is the label class used here to construct bounding boxes around mosquitoes manually.

Figure 6 below shows the annotated image of dataset.



(a) (b)

Figure 6:(a) Sample image (b) annotated portion of sample image

5.1.3 Data Preprocessing

Initially, data is required to implement the pre-processing task collected in compliance with the desired condition. Detailed data pre-processing procedures are depicted in Figure 7 below and described in the next sections.

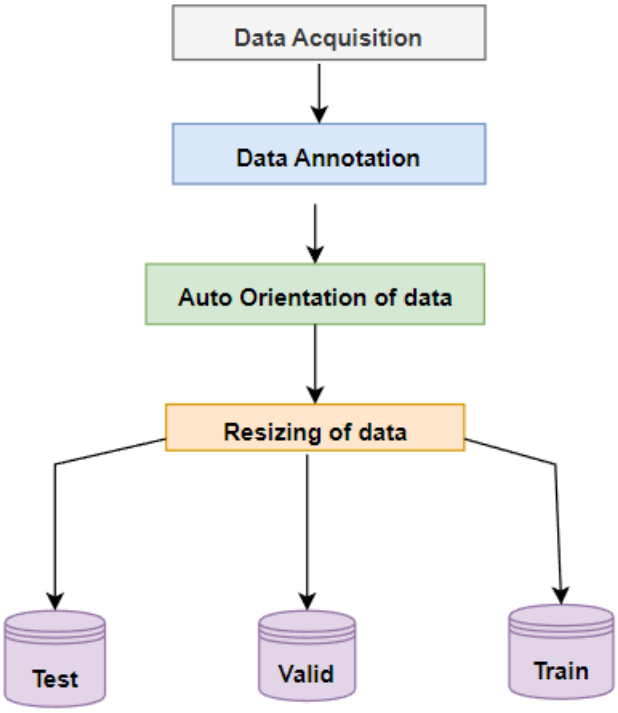


Figure 7: Data Preprocessing steps

5.1.4 Data Processing

Data processing is a method of transforming the data from a given process into a significantly more usable and preferred form by making it relevant and informative.

After finishing the dataset labelling, Roboflow performs data preprocessing steps such as image orientations, resizing, contrasting etc. After the needful steps, data can be converted into required

format and exported as a zip file or code.The data is converted into text files for all the dataset images which contains information about the bounding boxes for labelled mosquitoes.

Once the data processing is done, data is split into test, train and valid set. There are 721 training images, 174 images in valid dataset and remaining 105 in testing dataset. This train dataset consists of images directly from the video which are labelled. Once the training is complete, we follow the below workflow for our final detections.

5.2 YOLOv5 architecture

YOLO is a famous object detection algorithm which is easy to implement and is fast in terms of training. It has faster processing time than faster R-CNN as it doesn't apply separate network for extracting candidate region. The basic structure of the previous YOLOv5 is largely divided into the backbone network part, the neck part, and the head part, as shown in Figure 1 [18].

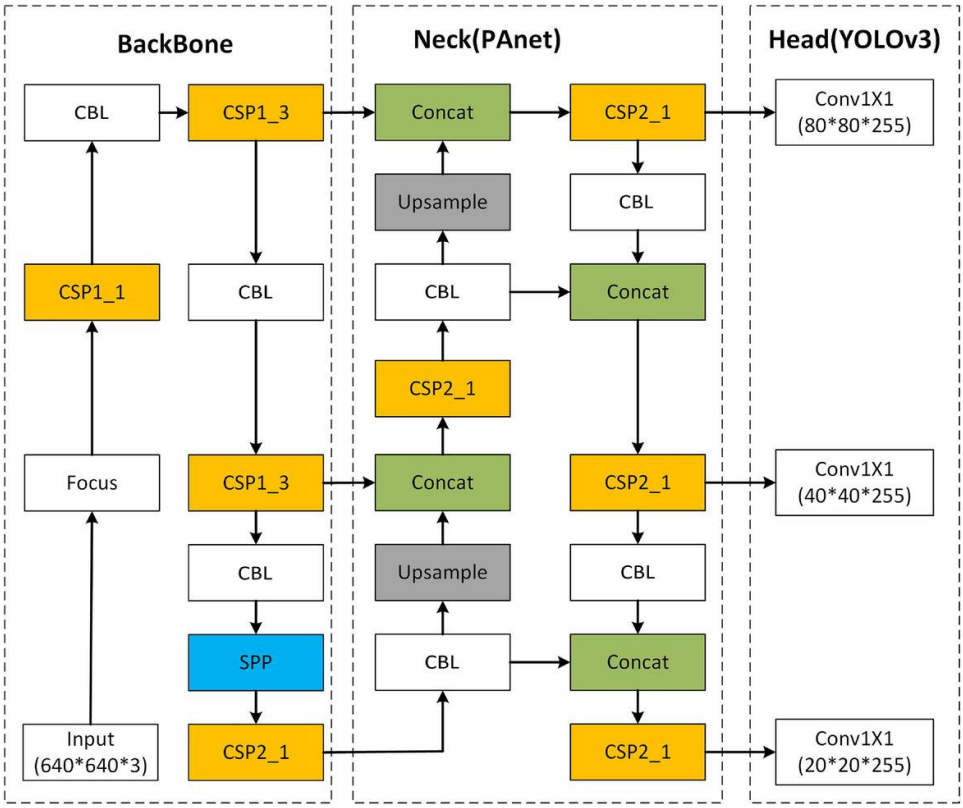


Figure 8: YOLOv5 Network Structure

YOLOv5 divides the model into three parts: the backbone network part, the feature enhancement part and the head part. Each part has different functions.

Backbone is a convolutional neural network formed by aggregating image features in various particle sizes. Neck is a series of layers that mix and combine image features to deliver prior to prediction, and Head consumes features from Neck (PAnet) and takes box and class prediction steps.

The backbone network part is used to extract image features. The biggest feature of YOLOv5 is that it has Focus and CSP (cross-stage partial connections) [21] layer. First, the focus structure is used to extract pixels from high-resolution images periodically and reconstruct them into low-resolution images. That is to say, the four adjacent positions of the images are stacked, and the information of WH dimension is focused into the *C* channel space to improve the receptive field of each point and reduce the loss of original information. The focus layer was created to reduce layers, parameters, FLOPS, and CUDA memory and improve forward and backward speed while minimizing the impact of mAP. Then, drawing on the design idea of CSPNet [19], the CSP1_x and CSP2_x modules are designed. The module first divides the feature mapping of the basic layer into two parts, and then combines them through the cross-stage hierarchical structure, which reduces the amount of calculation and ensures accuracy. The CSP layer extends to shallow information in the focus layer to maximize functionality .In the last part of the backbone network, using the SPP network, this module can further expand the receptive field and help to separate contextual features.

The feature enhancement part is used to further improve the feature extraction ability.The feature extraction module is iterated to extract detailed information and functions more thoroughly It uses the idea of PANet [20] to design the structure of FPN+PAN. First, it uses the FPN structure to convey strong semantic features from top to bottom, and then uses the feature pyramid structure constructed by the PAN module to convey strong positioning features from bottom to top. Through this method, it is used to fuse features between different layers.

The head part inherits the head structure of YOLOv3, which has three branches. The prediction information includes object coordinates, category and confidence. The main improvement is to use complete intersection over union (CIOU) loss as the bounding box region loss.

The basic principle of YOLOv5 is similar to YOLOv4. YOLOv5 is an improvement to YOLOv4. Among Faster R-CNN, YOLOv3 and YOLOv4, YOLOv5 has the best performance in precision, recall and average precision. It is available in 4 models s,m,l and x, each one of them offering different accuracies. The comparison chart of these models is shown in Figure 9. YOLOv5x has the largest storage size and YOLOv5s has the smallest storage size.

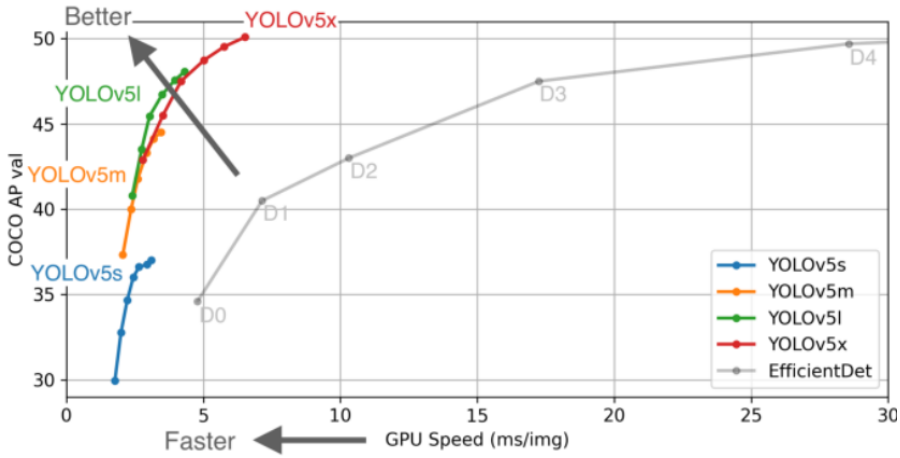


Figure 9: Comparison of YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x

5.2.1 Activation Function

An activation function defines the output of a neural network with any given set of inputs.They are useful as they add non-linearity in the neural network.

The choice of activation functions is most crucial in any deep neural network. Recently introduced activation functions are Leaky ReLU, mish, swish etc.

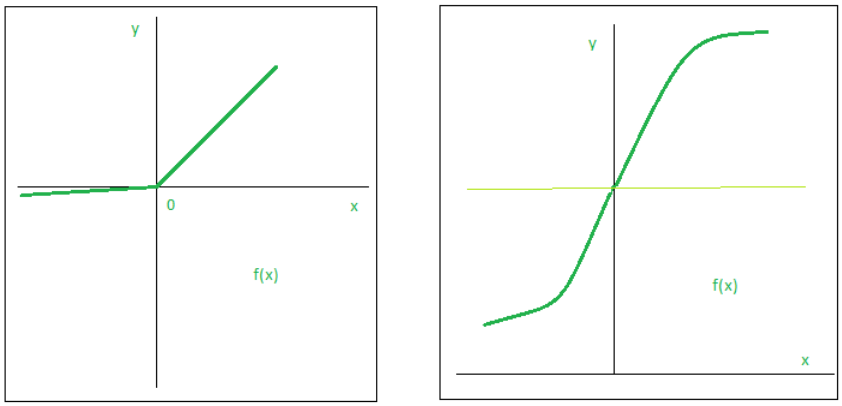
YOLOv5 has Leaky ReLU and sigmoid activation functions. Leaky ReLU is used in middle/hidden layers and sigmoid is used in final detection layer.

Leaky ReLU: It is an improved version of ReLU function. Instead of defining the ReLU function as 0 for x less than 0, we define it as a small linear component of x.

$$f(x) = \begin{cases} ax, x < 0 \\ x, x \geq 0 \end{cases} \quad (1)$$

Sigmoid: it is widely used activation function. It is defined as:

$$f(x) = \frac{1}{1+e^{-x}} \tag{2}$$



(a) (b)

Figure 10: (a) Leaky ReLU graph (b) Sigmoid Function

5.2.2 Optimizers

Any deep learning model tries to generalize the data using an algorithm and tries to make predictions on the unseen data. We need an algorithm that maps the examples of inputs to that of the outputs and an optimization algorithm. An optimization algorithm finds the value of the parameters(weights) that minimize the error when mapping inputs to outputs. These optimization algorithms or optimizers widely affect the accuracy of the deep learning model. They as well as affect the speed training of the model. When deep learning model is trained, each epoch's weight needs to be modified and loss function is minimized. The optimizer is a function that modifies the attributes of neural network such as weights and learning rate. There is a need for an optimization algorithm that can choose right weights for the model.

SGD and Adam are the common optimizers used in YOLOv5 models.

SGD(Stochastic Gradient Descent) Deep Learning optimizer:

The term stochastic means randomness on which the algorithm is based upon. In stochastic gradient descent, instead of taking the whole dataset for each iteration, we randomly select the batches of data. That means we only take few samples from the dataset. The procedure is first to

select the initial parameters w and learning rate n. Then randomly shuffle the data at each iteration to reach an approximate minimum.

$$\omega = \omega - \eta \nabla Q_i(\omega) \tag{3}$$

The procedure is first to select the initial parameters w and learning rate n. Then randomly shuffle the data at each iteration to reach an approximate minimum.

Since we are not using the whole dataset but the batches of it for each iteration, the path took by the algorithm is full of noise as compared to the gradient descent algorithm. Thus, SGD uses a higher number of iterations to reach the local minima. Due to an increase in the number of iterations, the overall computation time increases. But even after increasing the number of iterations, the computation cost is still less than that of the gradient descent optimizer.

Adam Deep learning optimizer:

The name adam is derived from adaptive moment estimation. This optimization algorithm is a further extension of stochastic gradient descent to update network weights during training. Unlike maintaining a single learning rate through training in SGD, Adam optimizer updates the learning rate for each network weight individually.

The adam optimizer has several benefits, due to which it is used widely. It is adapted as a benchmark for deep learning papers and recommended as a default optimization algorithm. Moreover, the algorithm is straightforward to implement, has faster running time, low memory requirements, and requires less tuning than any other optimization algorithm.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \left[\frac{\delta_L}{\delta \omega_t}\right] v_t \tag{4}$$

5.2.3 YOLO labelling format

Each file has one bounding box annotation for each of the objects in the image[13]. The annotations are normalized to the image size and lie within range of 0 to 1. They are represented in the following format

< object-class-ID> <X center> <Y center> <Box width> <Box height>

If there are two objects in the image, the content of the YOLO annotations text file might look like this in Figure 11.

```
0  0.383  0.439  0.183  0.628

0  0.507  0.454  0.191  0.713
```

Figure 11: YOLO format annotated file

5.3 Region of Interest Extraction

As we need to count mosquitoes only on two regions of the frame and compare the count, therefore we extract those regions using opencv functions and canny edge detection.

The first step is detecting the edges of the segments we want to extract. This is a multi-step process as mentioned below:

1. Convert the RGB image to gray-scale using “cvtColor()”
2. Remove noise from the gray-scale image by applying a blurring function “GaussianBlur()”
3. Finally applying the “Canny ()” function to the blurred image to obtain the edges.

The output after those steps looks like this. Although the regions are identified, there are a lot of unwanted edges which need to be eliminated and some of the edges have gaps in between which need to be closed.

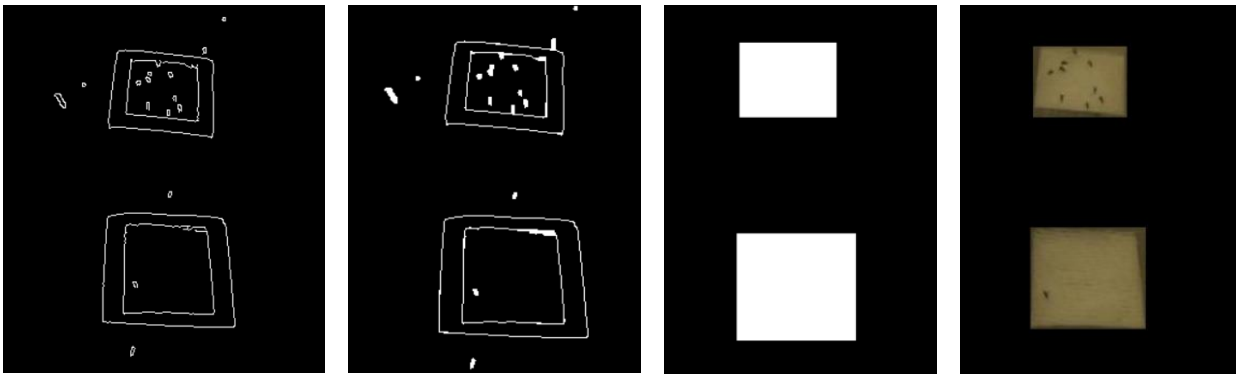
A common method applied for such purpose is Morphological Transformation which involves using a succession of dilations and erosions on the image to remove unwanted edges and close gaps. We use OpenCV functions “dilate ()” and “erode()” over multiple iterations to get an output as shown in Figure 12(a).

As you can see, the edges are now complete and much smoother than before. In order to implement a smooth extraction of the table, we will find the bounding rectangle, using OpenCV “boundingRect()” function, of the table contour and use its coordinates to extract the sub-image from the original image containing only the object of interest, in this case it will be two rectangular areas with different compounds as shown in Figure 12(b).

Now that we have segments identified, we need to build the image mask which will allow us to pull out the desired features from the original image.

Applying this mask on the original image gets us the desired segments over a background of our choice (e.g., Black or White) as shown in Figure 12(c).

For a black background we create a black canvas and then draw upon it using the OpenCV function “bitwise_and()” with the previously obtained mask.
The final output looks like as in Figure 12(d)



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

Figure 12: (a) Image after canny edge detection (b) Dilation and erosion of image (c) Generating mask using boundingRect() (d)Final image after performing bitwise_and() on image and the mask generated

6. Experimental Results

6.1 Evaluation Metrics

In our experiments, precision, recall, mean average precision (*mAP*), and processing time are employed as evaluation metrics to investigate the performance of network. Detections were evaluated via ground-truthing. Before introducing these metrics, the following concepts are introduced: True Positives (TP) refer to the number of positive samples correctly assigned by the classifier; True Negatives (TN) refer to the number of negative samples correctly assigned; False Positives (FP) refer to the number of positive samples misclassified; False Negatives (FN) refer to the number of negative samples misclassified. Intersection over Union (IOU) measures the degree of overlap between the candidate bound and the ground truth bound.

Precision is the proportion of correct positive samples in the anticipated data set to the number of positive samples predicted by the model. The recall is the proportion of right positive samples in the predicted data set to the actual number of positive samples in the predicted data set. The calculation formulas of precision and recall are given in Equations (5) and (6) respectively.

$$Precision = \frac{TP}{TP+FP} \tag{5}$$

$$Recall = \frac{TP}{TP+FN} \tag{6}$$

The precision represents the usefulness of the detection; a high precision indicates that the trained model returns a truly-detected object rather than a falsely detected one, while the recall defines the truly-detected object that the model returns [25]. A high precision shows a low value of FP, while the recall is usually related to a small number of FPs. Therefore, a higher percentage of precision and recall indicates that a model performs better [22]. The model's performance was evaluated using the F1-score given by Equation (7).

$$F1\ Score = 2\frac{Precision\ X\ Recall}{Precision + Recall} \tag{7}$$

This F1-score represents the harmonic mean of precision and recall, and a higher F1-score shows a better performance. All models were then evaluated with the mean average precision (mAP) using Equation (9).

Average Precision (*AP*): the region of the graph is bounded by the P-R curve and the coordinate axis. The calculation formula is as in equation (7).

$$AP = \sum_{i=1}^{n-1}(r_{i+1} - r_i)p(r_{i+1}) \tag{8}$$

Mean Average Precision (*mAP*): the mean of all categories in the dataset's average precision. The calculation formula is as in equation (8).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{9}$$

Averaging the average precision (AP) for all classes involved in the trained model yields mAP. A sufficiently well-performed model tends to produce a higher accuracy. Finally, the performance of processing rate was calculated using frame per second (FPS) with equation (10) to evaluate the model speed in processing the input in real-time applications.

$$FPS = \frac{Number\ of\ frames}{Total\ detection\ time(s)} \tag{10}$$

In general, the higher the FPS, the faster the model in detecting an object. Upon all training, a robust model was selected via the comparative performance.

6.1.1 Loss Functions

It is very important to check if your model is able to generalize or not. For this purpose, we make use of the loss function so as to check the performance of the model, how good and how bad the model is performing.YOLO uses sum-squared error between the predictions and the ground truth to calculate loss. The loss function composes of:

- classification loss - To determine which category the object in the prediction box belongs to. If an object is detected, the classification loss at each cell is the squared error of the class conditional probabilities for each class.
- box regression loss-. The box regression loss measures the errors in the predicted boundary box locations and sizes. We only count the box responsible for detecting the object.
- objectness loss - To determine whether there are objects in the predicted bounding box. This loss function helps the model to distinguish the background and foreground areas.

6.2 Training

It is essential to the understand the following when training any deep learning model.

- Epoch is when an entire dataset is passed forward and backward through the neural network only once[14].
- Batch Size is the total number of training examples present in a single batch [14]. And it goes a long with python generate mention previously
- Iterations (steps_per_epoch) is the number of batches needed to complete one epoch [14].

1)Epochs-

One Epoch is when an entire dataset is passed forward and backward through the neural network only once
Since one epoch is too big to feed to the computer at once we divide it in several smaller batches. One epoch leads to underfitting of the curve in the graph.

As the number of epochs increases, a greater number of times the weight are changed in the neural network and the curve goes from underfitting to optimal to overfitting curve.

2)Batch Size-

Batch size is total number of training samples present in a single batch. You can't pass the entire dataset into the neural net at once. So, you divide dataset into Number of Batches or sets or parts.

3)Iterations

Iterations is the number of batches need to complete one epoch. The number of batches is equal to number of iterations for one epoch.Let's say we have 2000 training examples that we are going to use. We can divide the dataset of 2000 examples into batches of 500 then it will take 4 iterations to complete 1 epoch.

6.2.1 Finding best batch size for our model

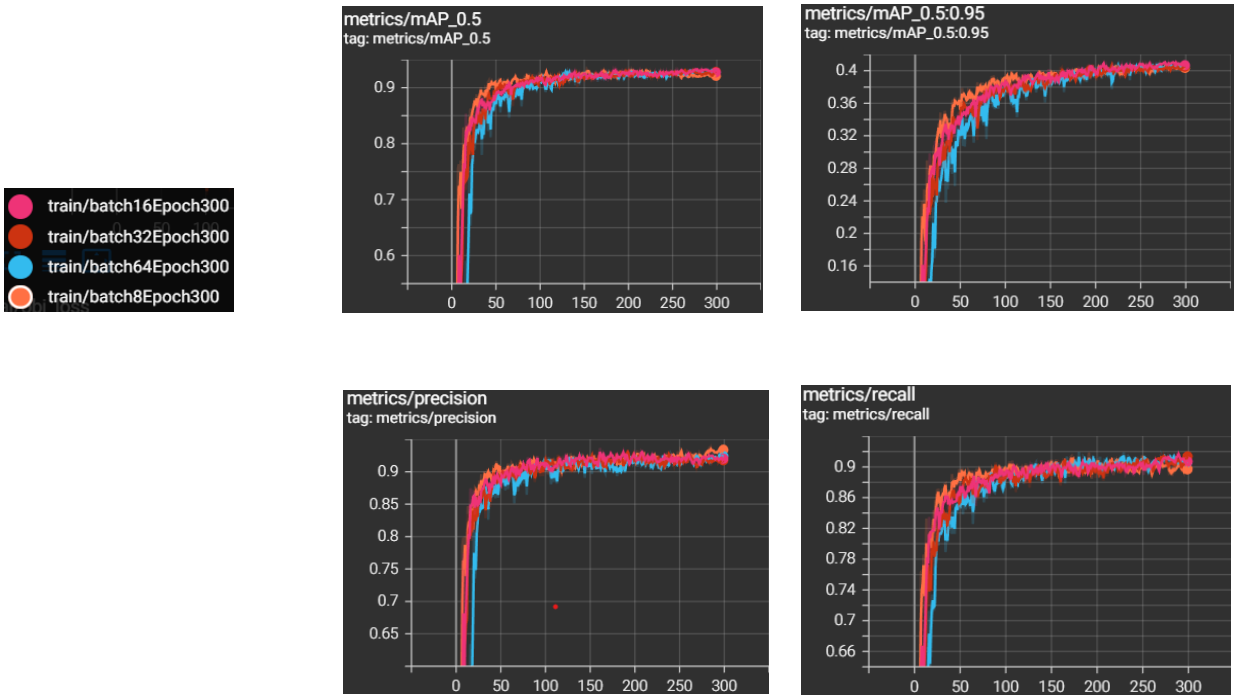
One common perception is that you should not use large batch sizes, because this will only cause the model to overfit, and you might run out of memory. if our batch size is small, there will be a

lot of noise present, and we might train our model only on noise. Nonetheless, after a certain size, if your gradient is already accurate, there is no point in making the batch size even bigger**,** because it will just be a computational waste as there will be little gain in accuracy.

Moreover, by using bigger batch sizes (up to a reasonable amount that is allowed by the GPU), we speed up training, as it is equivalent to taking a few big steps, instead of taking many little steps. Therefore, with bigger batch sizes, for the same amount of epochs**,** we can sometimes have a 2x gain in computational time. Moreover, **t**he noisier our gradient is, the bigger batch size we want.

Batch Size	Precision	Recall	mAP50	Time(hrs)
8	93.3	89.8	92.9	1.094
16	92.1	90.6	92.7	0.7
32	91.9	89.8	92.9	0.502
64	92.4	90.8	93.1	0.493

Figure 13 : Comparing performance of YOLOv5l model on different batch sizes and fixed Epoch value of 300.



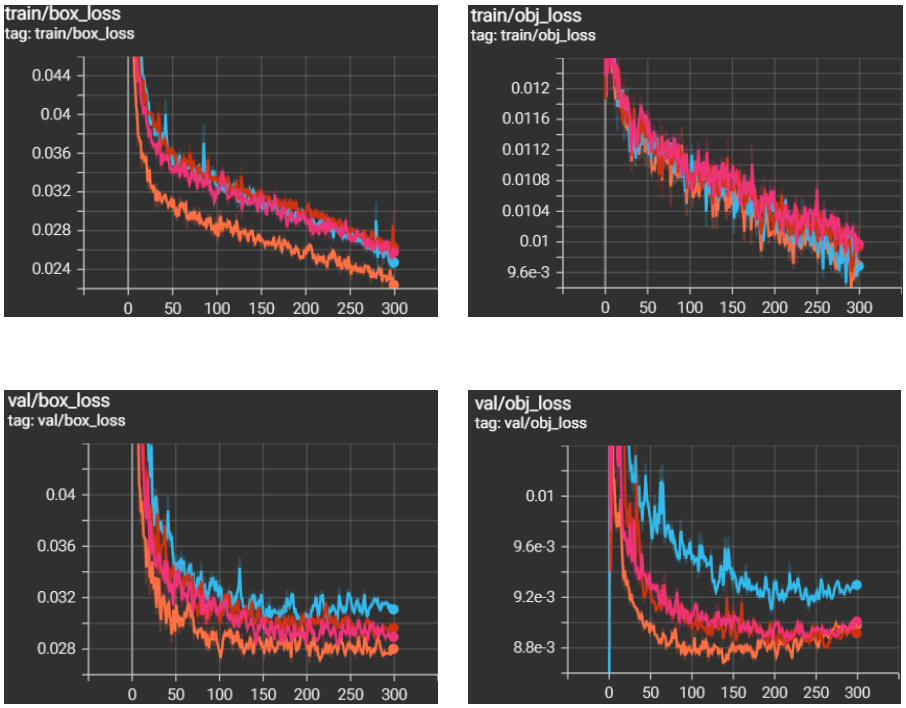


Figure 14 : Comparison of metrics, training losses and validation losses with different batch sizes and fixed epoch value of 300.

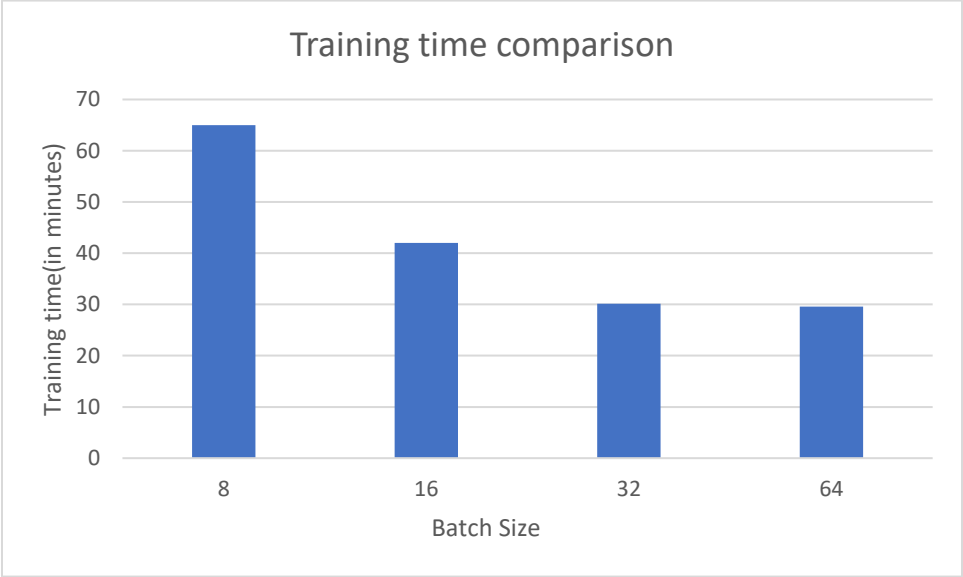


Figure 15 : Training time comparison with different batch sizes

As we see, the performance metrics like precision, recall and mAP are not significantly varying with different batch sizes but only difference is the time consumption. Also if we compare the training loss and validation losses, it is observed that as the batch size increases, the gradient becomes noisy. The smoothest gradient is achieved with lowest batch size of 8 and it gets rougher with batch size 64. The GPU and memory in google colab didn't allow us to run on batch size of 128 as it required much higher RAM. Also, as the batch size increases from 8 to 16 to 32, there is not much difference in gradient, but the training time is also reduced to half of what is achieved with batch size of 8. Although the objectness loss in validation dataset and box regression loss in training dataset increases a little, but precision, accuracy and recall remains almost same. The losses are not increased that much so we can choose 32 as our batch size which makes our training faster. Batch size of 64 is not chosen because it makes the gradient rougher and losses greater and there is not much of a time difference observed in training.

There is no magic batch size number, such as 32, it depends on the complexity of your data, and the GPU constraints you have. We saw that small batch sizes can help regularize through noise injection, but that can be detrimental if the task you want to learn is hard. Moreover, it will take more time to run many small steps. On the opposite, big batch size can really speed up your training, and even have better generalization performances.

6.2.2 Transfer Learning

There are various ways to train the model and different approaches might be considered in different situations. We have chosen transfer learning for our model because of the relatively smaller dataset.

When having a large enough dataset, the model will benefit most by training from scratch. The weights are randomly initialized by passing an empty string (' ') to the weights argument. But with transfer learning , pretrained model is used.

Since used dataset is relatively small (~1000 images), transfer learning is expected to produce better results than training from scratch. Ultralytic's default model was pre-trained over the COCO dataset, though there is support to other pre-trained models as well (VOC, Argoverse, VisDrone, GlobalWheat, xView, Objects365, SKU-110K). COCO is an object detection dataset with images from everyday scenes. The model will be initialized with weights from a pre-trained COCO model,

by passing the name of the model to the ‘weights’ argument. The pre-trained model will be automatically downloaded.

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems. The comparison of performance with and without using pretrained model is shown in Figure 13 [23].

Three possible benefits to look for when using transfer learning:

- **Higher start**. The initial skill (before refining the model) on the source model is higher than it otherwise would be.
- **Higher slope**. The rate of improvement of skill during training of the source model is steeper than it otherwise would be.
- **Higher asymptote**. The converged skill of the trained model is better than it otherwise would be.

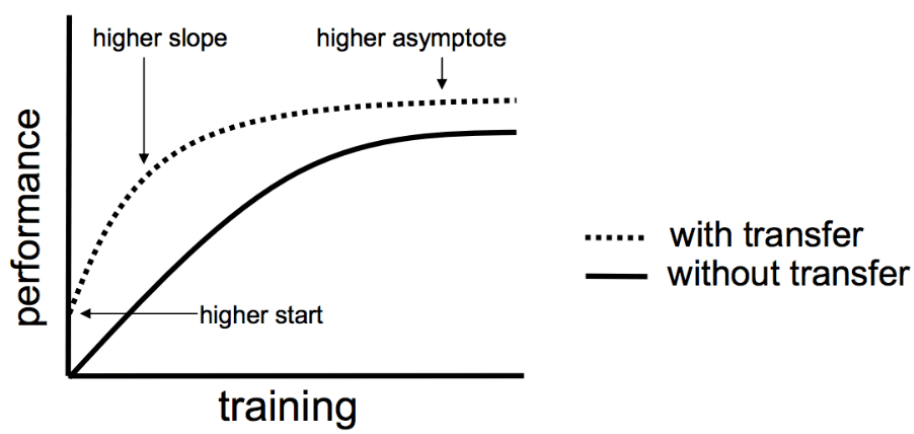


Figure 16 : Comparison of performance graph of a model with and without transfer learning

It is an approach to try if you can identify a related task with abundant data and you have the resources to develop a model for that task and reuse it on your own problem, or there is a pre-trained model available that you can use as a starting point for your own model.

On some problems where you may not have very much data, transfer learning can enable you to develop skillful models that you simply could not develop in the absence of transfer learning.

Training is induced by the following command:

```
!python train.py --img 416 --batch 16 --epochs 500 --data {dataset.location}/data.yaml --
weights yolov5l.pt --cache
```

- img – Image size in pixels (416 X 416 here)
- batch – batch size(use the largest batch size your hardware allows)
- epochs- number of epochs
- data – path to data-configuration files
- weights-path to initial weights

6.2.3 Overfitting and Underfitting

Your model is *underfitting* the training data when the model performs poorly on the training data. This is because the model is unable to capture the relationship between the input examples (often called X) and the target values (often called Y). Your model is *overfitting* your training data when you see that the model performs well on the training data but does not perform well on the evaluation data. This is because the model is memorizing the data it has seen and is unable to generalize to unseen examples.

Overfitting is a scenario where your model performs well on training data but performs poorly on data not seen during training. This basically means that your model has memorized the training data instead of learning the relationships between features and labels.

Overfitting is easy to diagnose with the accuracy visualizations you have available. If "Accuracy" (measured against the training set) is very good and "Validation Accuracy" (measured against a validation set) is not as good, then your model is overfitting.

Underfitting is the opposite counterpart of overfitting wherein your model exhibits high bias. This situation can occur when your model is not sufficiently complex to capture the relationship between features and labels (or if your model is too strictly regularized). Underfitting is a bit harder

to diagnose. If Accuracy and Validation Accuracy are similar but are both poor, then you may be underfitting.

One of the ways to prevent overfitting is by training with more data. Such an option makes it easy for algorithms to detect the signal better to minimize errors. Users should continually collect more data as a way of increasing the accuracy of the model. However, this method is considered expensive, and, therefore, users should ensure that the data being used is relevant and clean. Overfitting can also be combated by reducing the complexity of the model (i.e. reducing the number of trainable parameters). The specifics of how this is accomplished vary depending on the learning algorithm and the domain. For neural networks, we can use fewer layers (shallower networks), fewer neurons per layer, sparser connections between the layers (as in convolutional nets), or regularization techniques like dropout.

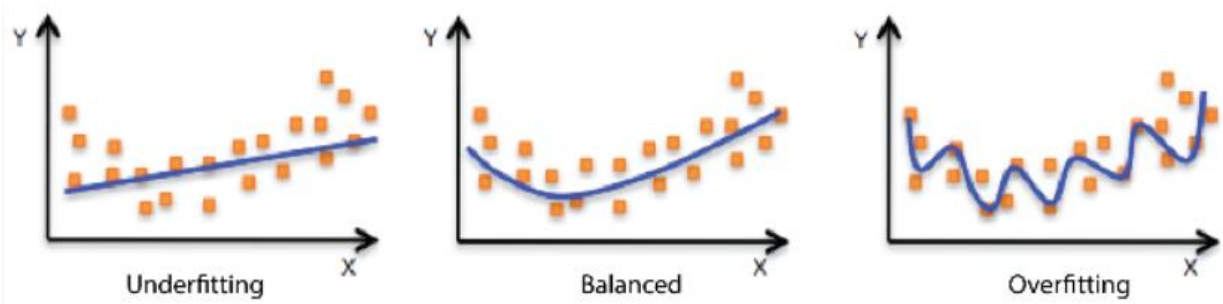


Figure 17 : Difference between Underfitting, Balanced and Overfitting models graphically

Here, we observe that when we use Epochs more than 200 , it causes overfitting as validation loss starts increasing as depicted below which shows validation loss of model trained with batch size of 8 and epochs 300.

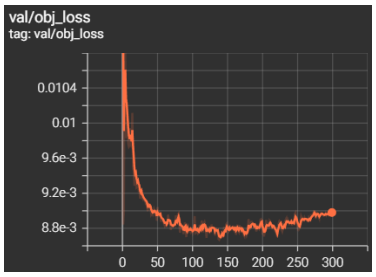


Figure 18: Objectness Loss in validation data showing overfitting in a model with Epochs 300 and batch size of 8

6.2.4 Choice of Optimizer

Adam optimizer shows really rough graphs for performance metrics as shown in Figure 16. The training is performed at batch size of 32 and Epochs value 200. The values are compared to those in SGD optimizer in Figure 19 and Figure 20. The values are seen to be increasing but better performance is observed in SGD with same batch size and Epoch and in less time, therefore SGD optimizer is chosen for our model.

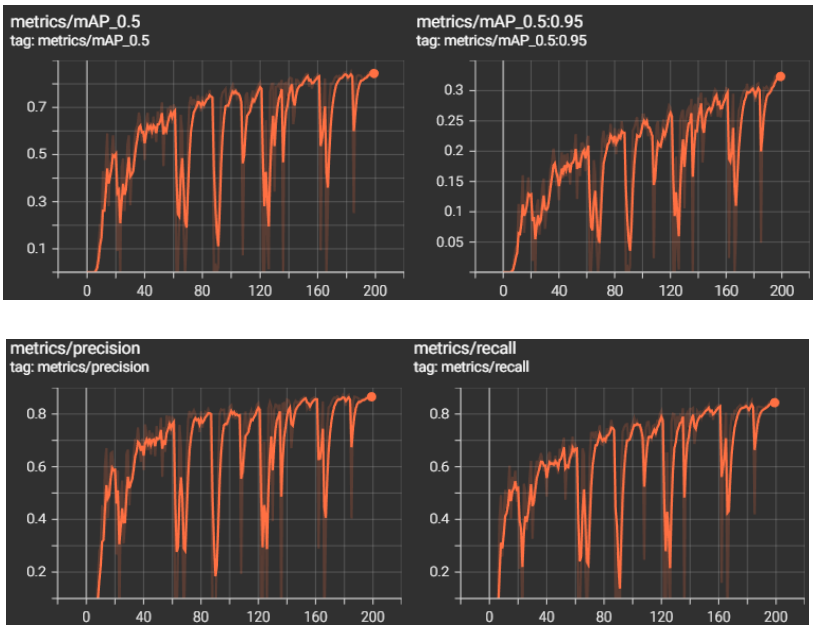
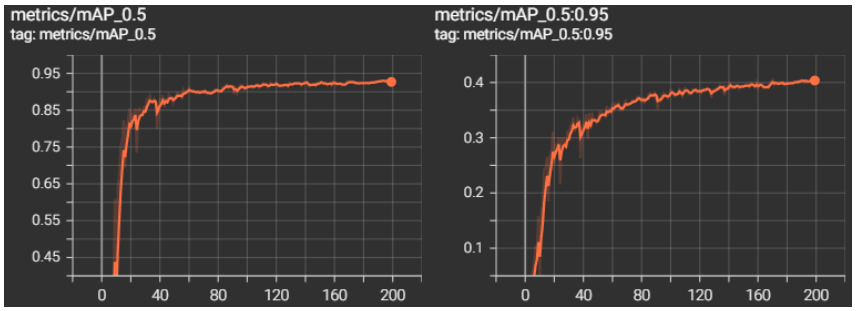


Figure 19 : Metrics for Adam optimizer



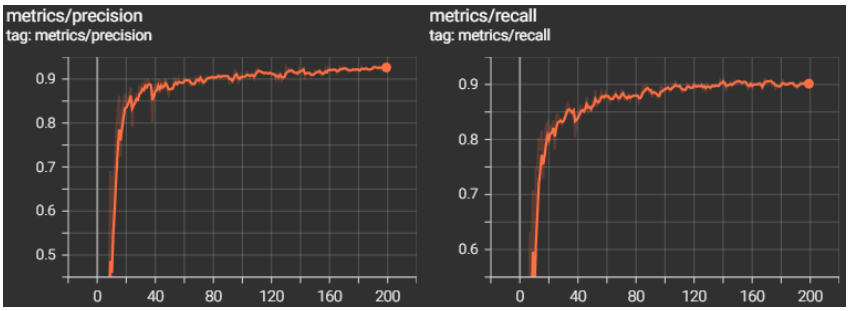


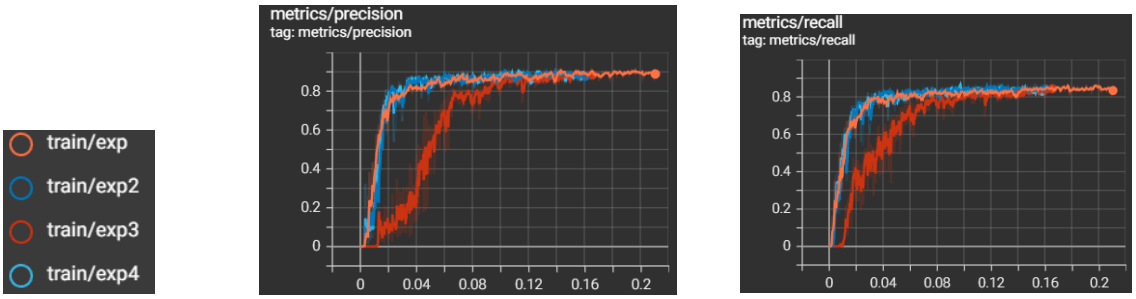
Figure 20 : Metrics for SGD optimizer

Optimizer	Precision(%)	Recall(%)	mAP_0.5(%)	mAP_0.5:0.95	Time(in hrs)
Adam	86.5	84.6	84.4	0.324	0.85
SGD	92.6	90.1	92.5	0.403	0.85

Figure 21 : Comparison table for Adam and SGD optimizers on Batch size 32 and Epoch 200.

6.3 Training results

The model was trained using Google Colab, which provides free access to powerful GPUs and requires no configuration. Model was trained with different combinations of batch size and epochs. Training a model for 500 epochs takes about 30 min. Different performance metrics for training and validation sets is shown in the graphs in Figure 20 and 21. Exp(experiment 1) is with batch size 16, exp 2 is 32, exp 3 is 64 and exp 4 is 8.



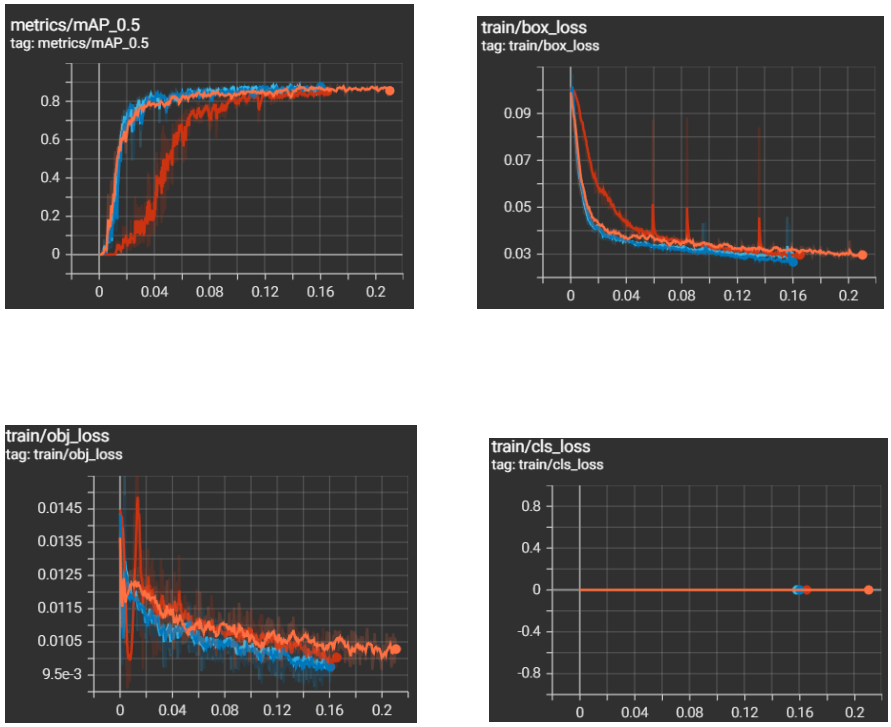


Figure 22 : Plots of Precision, Recall, mean Average Precision (mAP), Box loss, Objectness loss and Classification loss over the training epochs for the training and validation set.

There are three different types of loss shown in Figure 22 box loss, objectness loss and classification loss. The box loss represents how well the algorithm can locate the centre of an object and how well the predicted bounding box covers an object. Objectness is essentially a measure of the probability that an object exists in a proposed region of interest. If the objectivity is high, this means that the image window is likely to contain an object. Classification loss gives an idea of how well the algorithm can predict the correct class of a given object.

Model Name	Precision	Recall	mAP @ 0.5	mAP@0.5:0.95	Parameters(M)	FLOPS(G)	Time(h)
YOLOv5s	90.4	88.7	91.4	39.2	7.012	15.8	0.430
YOLOv5m	92.3	90.6	92.7	40.6	20.85	47.9	0.541
YOLOv5l	92.1	90.5	92.7	40.8	46.10	107.6	0.697
YOLOv5x	92.0	90.6	92.9	41.1	86.17	203.8	0.908

Figure 23: Comparison of YOLO models with Batch Size 16 and Epochs 300

Model Name	Precision	Recall	mAP @ 0.5	mAP@0.5:0.95	Parameters(M)	FLOPS(G)	Time(h)
YOLOv5s	90.6	87	91.1	38.4	7.012	15.8	0.304
YOLOv5m	92.3	90.6	92.7	40.6	20.85	47.9	0.541
YOLOv5l	92.6	90.2	92.8	40.5	46.10	107.6	0.85
YOLOv5x	92.0	90.6	92.9	41.1	86.17	203.8	0.908

Figure 24 : Comparison of YOLO models with Batch Size 32 and Epochs 200

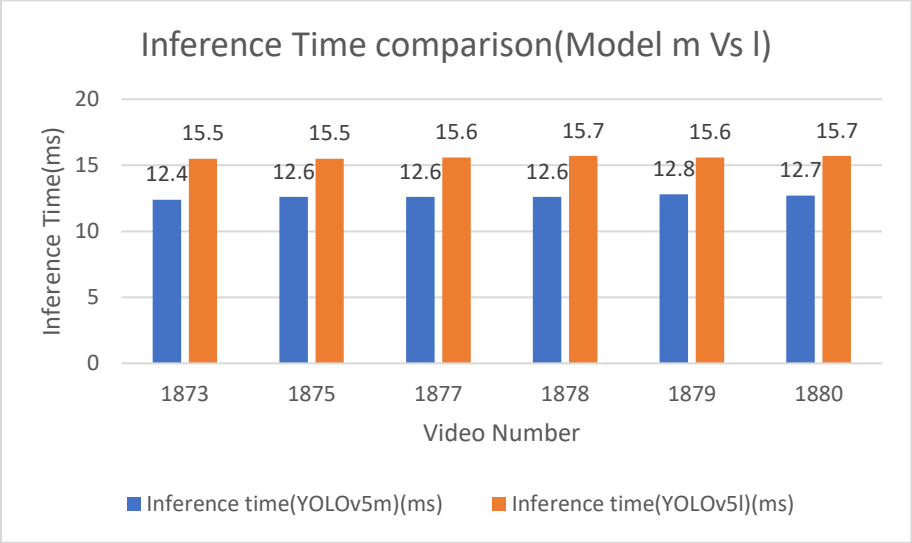


Figure 25 : Comparison of Inference time for 4031 frames between YOLOv5m and YOLOv5

7. Results and discussion

To evaluate the effectiveness of repellent activity against mosquito, we performed preparatory experiments with 40 Aedes Aegypti Female mosquitoes in a cage, however, is not ideal for the repellency test in the laboratory setting because it feeds on human. On the other hand, biting activity can't be tested throughout the experiment setting as actual human skin is not used, which is not optimal to quantify the biting rate to assess the effect of repellants.

The training model YOLOv5l with batch size of 32 and Epoch 200 is chosen as the best one as discussed above. The video dataset of 6 videos is run on this training model and following results are produced as shown below graphically in Figure 26. The comparison among all the repellents is done by combining results from all videos, which shows DEET provides better protection than others. Both 1% and 3% concentration showed almost same results with 3% DEET being slightly better.3% BA (butyl anthranilate) doesn't show any difference when compared to acetone, whereas 3% 4MPD shows a little repellency when compared with water on the side.

In comparison, this study used a lower density of mosquitoes, with 40 mosquitoes in a cage, because the lower-density environment more accurately mimics the biting pressures during outdoor activities.

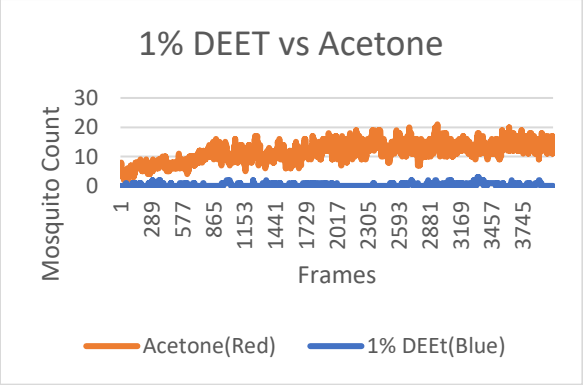


Figure 26(a)

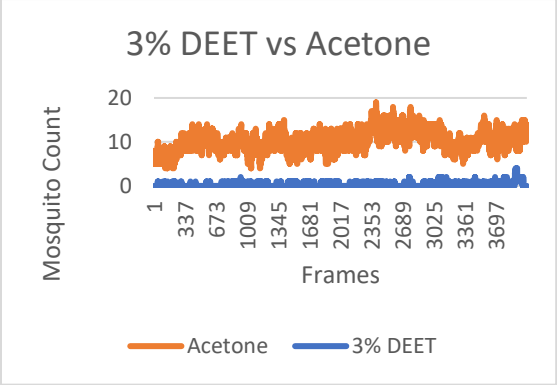


Figure 26(b)

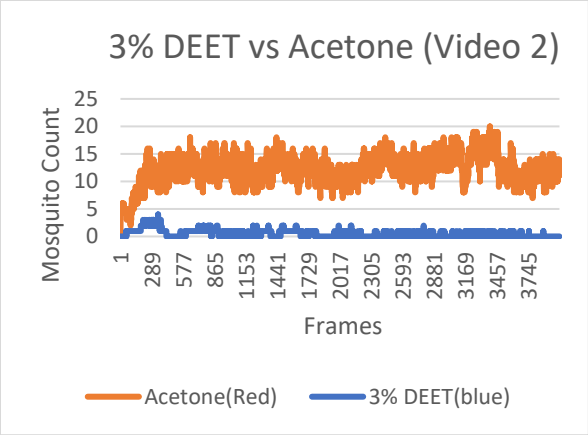


Figure 26(c)

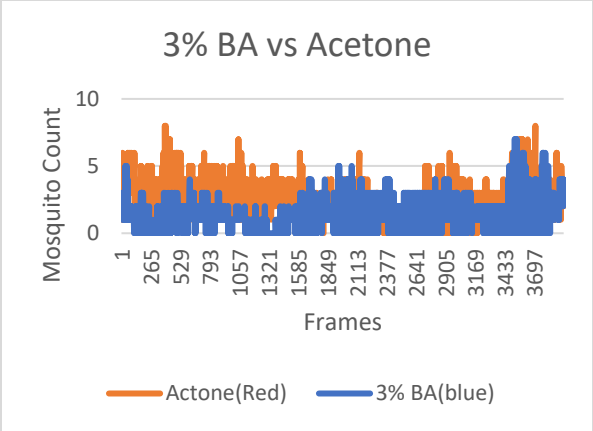


Figure 26(d)

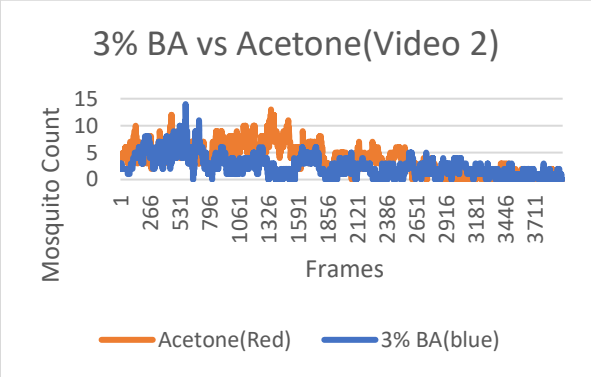


Figure 26(e)

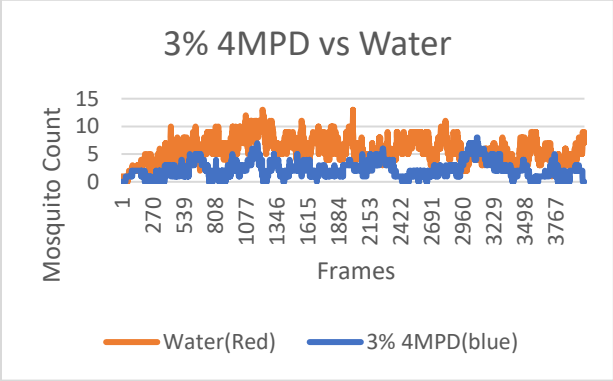


Figure 26(f)

Figure 26(a) : Comparison graph of 1% DEET and Acetone (b) : Comparison graph of 3% DEET and Acetone (c) Comparison graph of 3% DEET and Acetone in second video (d) : Comparison graph of 3% BA and Acetone (e) : Comparison graph of 3% BA and Acetone in second video, (f) : Comparison graph of 3% 4MPD and water

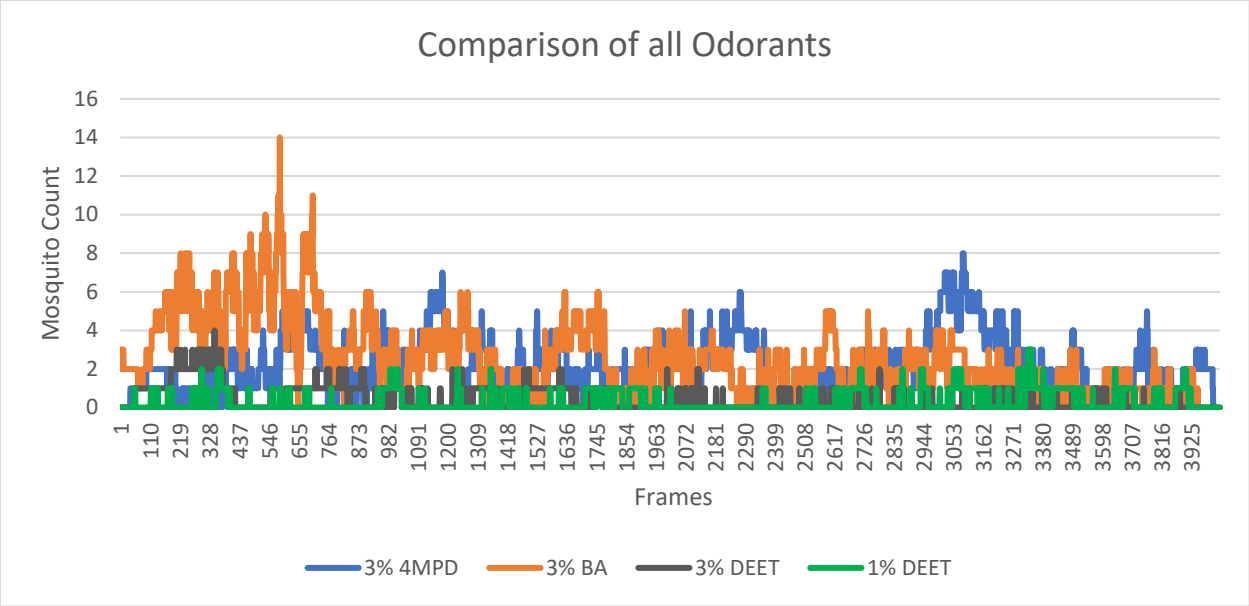


Figure 27 : Comparison graph of all odorants

The results plotted in Figure 26 and Figure 27 show that while DEET and 4MPD inhibited mosquitoes from flying towards the odor, BA had no such effect. There have been previous studies that butyl anthranilate (BA) had been used to repel Aedes aegypti mosquitoes from landing on human skin, but it doesn't prove that from experiment. Howerever, DEET with both 1% and 3% concentration is shown to be really effective against mosquitoes in the cage with it's repellency topping the others.

These results have been obtained using YOLOv5m model which has been chosen because of it's accuracy and less training time observed on this dataset with small object detection. YOLOv5s,m,l and x when compared with each other in this thesis model, showed that model s and m uses relatively smaller number of parameters but model l and x showed slightly better mAP values and less losses. There was a tradeoff done as YOLOv5m takes almost half as time as model l during training.

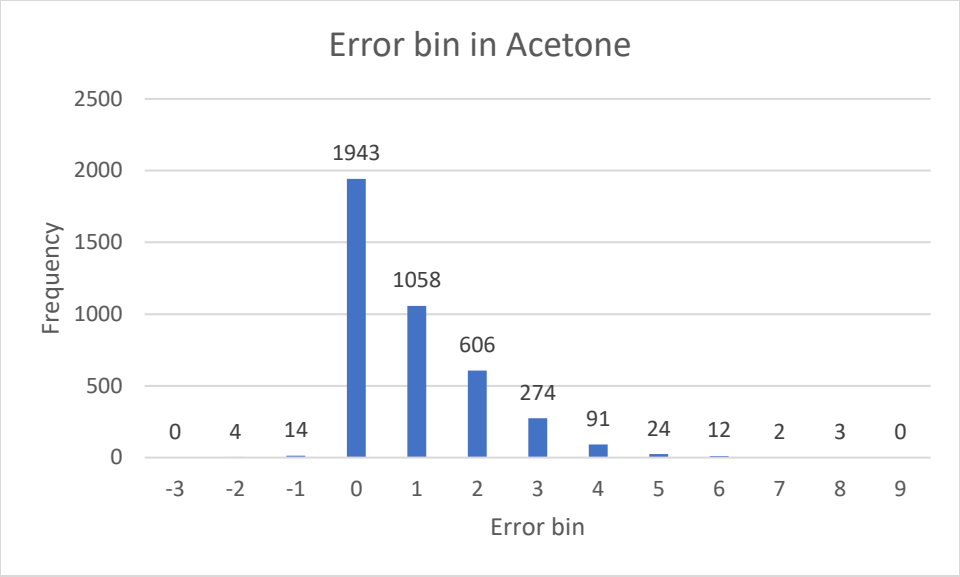


Figure 28 : Error difference ground truth and predicted values in one of the video dataset(3% DEET vs Acetone)

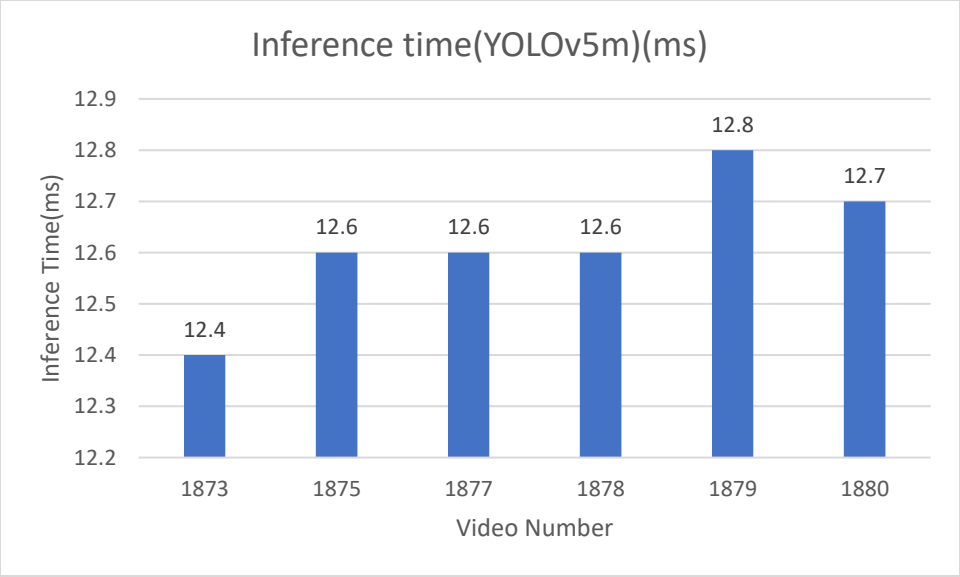


Figure 29 : Comparison of Inference time in all the videos.

The average inference time taken by YOLOv5m model is 12.6ms for every frame as shown in Figure 29. The total time taken for detections is calculated to be approximately 92.68 seconds on 4031 frames. Therefore, these results show that FPS for the built model is 44.

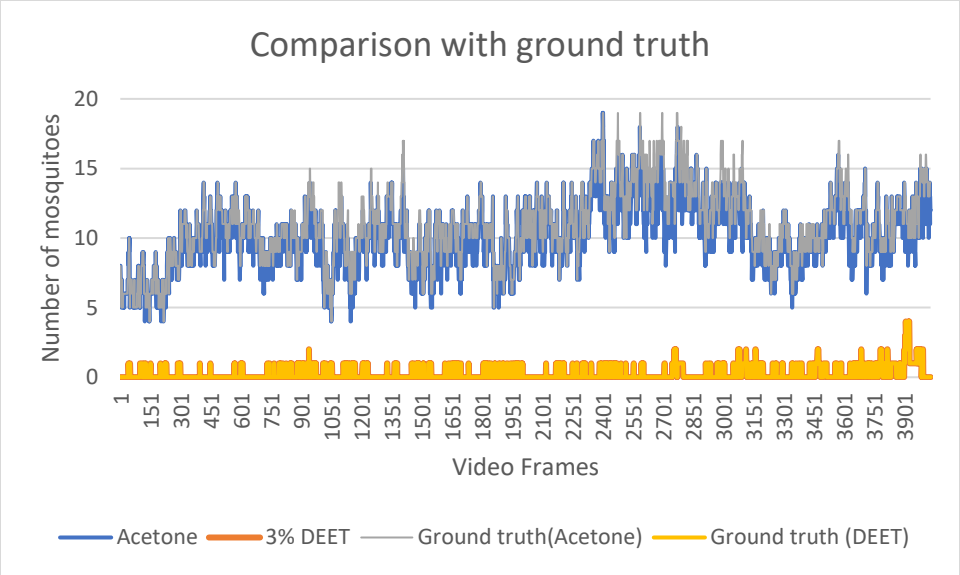


Figure 30 : Comparison of ground truth and predicted values in one of the video dataset.

Errors <0 shows the detections of some blurry mosquitoes which havenot landed but detected by the model(blurry mosuqitoes). Error > 0 indicates that there are some frames where mosquitoes are not detected properly because of cluster or accuracy reasons.

8. Conclusion

Mosquito quantification on all the video dataset provided by researchers is done using YOLOv5 model with version ‘m’. Medium version is chosen after calculating and comparing all the metrics like precision, recall, mAP values and loss functions. Although some tradeoff is done with metrics when training time was compared. Main goal is to choose the model which would give best performance in less inference time. More the parameters, better is the performance observed with our dataset. But as the metrics evaluated showed only 0.1% difference between model m and l, therefore model with less training and inference time is chosen which is ‘m’. As this project is run on Google colab, the best batch size for provided GPUs and given RAM is observed to be 32 and the best Epoch value of 200 for training. Choosing Epoch value of above 200 caused overfitting in model which is observed by increasing objectness loss in validation dataset. The quantification results are comparable to ground truth. F1 score is equal to 91.6 and FPS for this model is 44.

After running this model on all videos, it is observed that DEET shows the best repellency as compared to 4% MPD and 3% BA. 3% and 1% DEET are shown to repel mosquitoes with almost same efficacy with 3% DEET being better by only 0.5%. 4% MPD showed very little repellency, whereas BA is ineffective against mosquitoes. Although, biting activity can’t be tested throughout the experiment setting as actual human skin is not used, which is not optimal to quantify the biting rate to assess the effect of repellants.

This model can be used to run on any video with similar experiment setup as it is able to detect and count mosquitoes with 92.7% of accuracy.

References

1. Takken W, Verhulst NO. Host preferences of blood-feeding mosquitoes. Annu Rev Enstool. 2012; 58(1):433–453. [PubMed: 23020619]
2. C. Lavialle-Defaix, V. Apaire-Marchais, C. Legros et al.2011., “Anopheles gambiae mosquito isolated neurons: a new biological model for optimizing insecticide/repellent efficacy,” Journal of Neuroscience Methods, vol. 200, no. 1, pp. 68–73
3. De Jong R, Knols BGJ. 1995. Selection of biting sites on man by two malaria mosquito species. Cell. Mol. Life Sci. 51:80–84
4. Larry Goodyer, Steven Schofield, Mosquito repellents for the traveller: does picaridin provide longer protection than DEET?, Journal of Travel Medicine, Volume 25, Issue suppl_1, May 2018, Pages S10–S15, https://doi.org/10.1093/jtm/tay005
5. Stacy D. Rodriguez, Lisa L. Drake, David P. Price, John I. Hammond, Immo A. Hansen, The Efficacy of Some Commercially Available Insect Repellents for Aedes aegypti (Diptera: Culicidae) and Aedes albopictus (Diptera: Culicidae) , Journal of Insect Science, Volume 15, Issue 1, 2015, 140, https://doi.org/10.1093/jisesa/iev125
6. Raji JI, DeGennaro M. Genetic Analysis of Mosquito Detection of Humans. Curr Opin Insect Sci. 2017 Apr;20:34-38. doi: 10.1016/j.cois.2017.03.003. PMID: 28428935; PMCID: PMC5393449
7. Chunlei Xia, Tae-Soo Chon, Zongming Ren, Jang-Myung Lee, Automatic identification and counting of small size pests in greenhouse conditions with low computational cost, Ecological Informatics, Volume 29, Part 2, 2015, Pages 139-146, ISSN 1574-9541, https://doi.org/10.1016/j.ecoinf.2014.09.006.
8. Thenmozhi Kasinathan, Dakshayani Singaraju, Srinivasulu Reddy Uyyala, Insect classification and detection in field crops using modern machine learning techniques, Information Processing in Agriculture, Volume 8, Issue 3, 2021, Pages 446-457, ISSN 2214-3173, https://doi.org/10.1016/j.inpa.2020.09.006
9. Mohamed Esmail Karar, Fahad Alsunaydi, Sultan Albusaymi, Sultan Alotaibi, A new mobile application of agricultural pests recognition using deep learning in cloud computing system, Alexandria Engineering Journal, Volume 60, Issue 5, 2021, Pages 4423-4432, ISSN 1110-0168, https://doi.org/10.1016/j.aej.2021.03.009

10. Mulchandani, Pratik & Siddiqui, Muhammad & Kanani, Pratik. (2019). Real-Time Mosquito Species Identification using Deep Learning Techniques. 2249-8958.
11. E. Fanioudakis, M. Geismar and I. Potamitis, "Mosquito wingbeat analysis and classification using deep learning," *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 2410-2414, doi: 10.23919/EUSIPCO.2018.8553542.
12. Li, W., Chen, P., Wang, B. *et al.* Automatic Localization and Count of Agricultural Crop Pests Based on an Improved Deep Learning Pipeline. *Sci Rep* 9, 7024 (2019). https://doi.org/10.1038/s41598-019-43171-0
13. Lihi Gur Arie .2022. The Practical guide for object detection with YOLOv5 algorithm. https://towardsdatascience.com/the-practical-guide-for-object-detection-with-yolov5-algorithm-74c04aac4843
14. Sagar Sharma.2017. Epoch vs Batch Size vs Iterations. https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9
15. Havlicek J, Roberts S, Flegr J. 2005. Women's preference for dominant male odour: effects of menstrual cycle and relationship status. Biol. Lett. 1:256–59
16. Wedekind C, Furi S. 1997. Body odour preferences in men and women: Do they aim for specific MHC combinations or simply heterozygosity? Proc. R. Soc. Lond. B 264:1471–79
17. Soumi Mitra, Stacy D Rodriguez, Julia Vulcan, Joel Cordova, Hae-Na Chung, Emily Moore, Yashoda Kandel, Immo A Hansen, Efficacy of Active Ingredients From the EPA 25(B) List in Reducing Attraction of *Aedes aegypti* (Diptera: Culicidae) to Humans, *Journal of Medical Entomology*, Volume 57, Issue 2, March 2020, Pages 477–484, https://doi.org/10.1093/jme/tjz178
18. Jia, W.; Xu, S.; Liang, Z.; Zhao, Y.; Min, H.; Li, S.; Yu, Y. Real-time automatic helmet detection of motorcyclists in urban traffic using improved YOLOv5 detector. IET Image Process. 2021, 10, 1049.
19. Wang, C.-Y., et al.: CSPNet: A new backbone that can enhance learning capability of CNN. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020
20. Wang, K., et al.: PANet: Few-shot image semantic segmentation with prototype alignment'no date. 10

21. Wang, C.-Y.; Mark Liao, H.-Y.; Wu, Y.-H.; Chen, P.-Y.; Hsieh, J.-W.; Yeh, I.-H. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. CVF 2020, 14–19
22. Jason Brownlee.: A Gentle Introduction to Transfer Learning for Deep Learning. In Deep learning for Computer Vision. December 20,2017, https://machinelearningmastery.com/transfer-learning-for-deep-learning/