Penguins versus Turtles

Arkapravo Nandi (z5412155), Haonan Yan (z5441308), Nazar Dilinazha(z5387957), Vishnu Birudavolu(z5280782), Mengyao Xiong(z5288298)

***Abstract*—Computer vision has revolutionized fields ranging from surveillance and robotics to medical diagnostics and biology. Object detection and classification are two core tasks in the field of computer vision, which is widely used in various fields such as surveillance, traffic monitoring, medical diagnosis and biological research. Although advances in deep learning have revolutionized these tasks, traditional methods still have their advantages, especially when computational resources and data are limited.**

***Index Terms*—Computer Version, Machine Learning, Deep Learning, Object detection and classification.**

I. Introduction

In this project, we focus on recognizing and classifying two specific animals in wildlife images: penguins and sea turtles. Our challenge is to develop an efficient and accurate method that can automatically detect the location of these animals and then classify them. The dataset used for this purpose is the Penguin and Turtle dataset from Kaggle, consisting of a training set of 500 images and a validation set of 72 images.

II. Literature Review

Object detection is a computer vision task that involves identifying and localizing multiple objects within an image or video frame.

The basic architecture of traditional object detectors is shown in **Fig. 1** [12]. Traditional methods include [16] Histogram of Oriented Gradients (HOG) [2], and Deformable Part Models (DPM) [4]. While these methods demonstrated some success in simple scenarios, they struggled to cope with complex scenes and lacked the ability to learn high-level features directly from data.

In recent years object detection had make impressive breakthroughs due to the application of Neural networks, especially Convolutional Neural Networks (CNNs) [5]. However, researchers are still facing some challenges from intra-class variation, number of categories, and efficiency. [19].

CNN based object detection method can be broadly categorized into one stage detector and two stage detector.

Two-stage detectors, such as Faster R-CNN [14],SPP-net [12],,FPN [7]. These models try to find an arbitrary number of objects proposals in an image during the first stage and then classify and localize them in the second.

One-stage detectors, like YOLO (You Only Look Once) [13] and SSD (Single Shot Detectors) [10], significantly improved the speed of object detection by directly predicting bounding boxes and class probabilities in a single pass through the network.

Fig. 1. Basic architecture of traditonal object detection method.

Fig. 2. An Example of Input Image with 3 Key points(from Left) and Detected Image(on Right)

III. Methods

A. Data Exploration

The dataset consists of a training set of 500 images and a validation set of 72 images. Each image is annotated to contain either penguins or turtles. The annotations provide the category labels and bounding box coordinates of the image. The r

B. Proposed Detection Method - 1

1) First Method: In the First Detection Method, the first Input Image is converted to a Negative Image. A negative image is obtained by subtracting 255 from each of the pixel values present in 3 channels i.e. Red, Green, and Blue. Following that Shift Detection Algorithm from Open CV is used to detect features from both the Input and Negative Images. For, the Shift Detection Algorithm, the parameters Number of features to be detected, number of Octave Layers, contrast threshold, edge threshold, and sigma are given the values 0, 4, 0.04, 50, and 1.67 respectively. The shift detection algorithm detects all the important edge features in both input and negative output images, but only 10 percent of prominent edge features [16], [17] are considered for our proposed detection model. The prominent key points [16] between the 2 images are matched using Brute Force Matching Method. The brute force method uses 'Manhattan' distance to measure the distance between the vectors (key points) of both Images. After receiving the important matching points from both the images it is checked if the number of matching points is greater than 55 or not. So, if the number of matching points is greater than 55, then the bounding box is drawn with the matching points got on the first image (Input) which were obtained by Brute Force Method. The Bounding Box is drawn with the top left (minimum x-value, minimum y-value) and bottom right coordinates (maximum x-value, maximum y-value) around the image.

Fig. 3. An Example of Input Image with 52 Key points(from Left) and Detected Image(on Right)

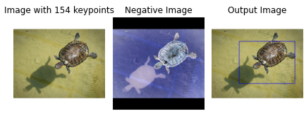

Fig. 4. Original Input Image (id: 103), Negative Image, and Detected Object in the Image

From the above images shown in **Fig. 2** and **Fig. 3**, it is seen that the Images detected by following the combination of Shift and Brute Force are not good i.e. does not contain enough information or features which will help the computer to classify as Penguins. For example, in **Fig. 3**, the Penguins' body is nicely detected but not its eyes and heads are coming in the detected portion. Also, in Fig. 1, the Brute Force Method can only find similarities with only the wing portion of the Penguin. Therefore, when the length of matching points is found to be less than 55, a custom algorithm (i.e. **Second Method**) has been implemented as an alternative approach to deal with the above examples as shown in **Fig. 2** and **Fig. 3**.

The Images shown in **Fig. 4**, and **Fig. 5** are the Input Images (shown from left), and the middle image is the Negative Image of the Input Image (the negative image is obtained by subtracting 255 from the input pixels of all three channels) and the last Output Images are obtained by **First Method**. The Output images shown in **Fig. 4** and **Fig. 5** were obtained by (i) brute force matching between the similar pixel coordinate vectors between the Input and Negative Image, and (ii) drawing bounding boxes on the image by taking the top left coordinates and bottom right coordinates of the prominent matching key points on the Input Image.

2) Second Method: The Second Method is used when the length of matching points obtained from the Input Image after Brute Force Matching is less than 55. In this method, first, the Input Image is converted to Gray-scale Image. Following the Gray-scale Conversion, the Gray-scale Image is converted to Inverted Image. It is converted to Inverted to make the object brighter and the background darker. Then, the Inverted Image is converted to a threshold image by setting a threshold equal to 180. Contour Algorithm [17] is applied to the threshold image to detect the contours of [16], [17] the Input Image.

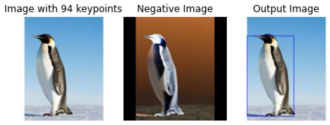

Fig. 5. Original Input Image (id: 118), Negative Image, and Detected Object in the Image

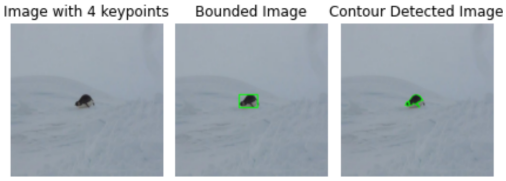

Fig. 6. Original Input Image(id: 88), Bounded Image and Contour Detected Image

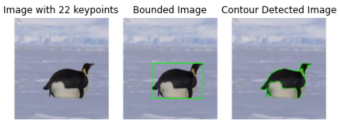

Fig. 7. Original Input Image(id: 186), Bounded Image and Contour Detected Image

Then a condition has been mentioned that if the length of contours detected on the input image is greater than 0, then the object contour is detected from the image by evaluating the maximum contour area among all the detected contours. From, the detected maximum contour area, a bounding box has been drawn on an image by taking minimum x, minimum y, maximum x, and maximum y values from the detected maximum contour area. Whereas, if no contour area can be detected the input image is given as output image.

The Images shown in **Fig. 6**, and **Fig. 7** are the Input Images (shown from left), the middle image is bounded by a Rectangular bounding box, and the last Output Images are obtained by following Contour Algorithm implemented in Second Method.

The Middle Bounded Images shown in **Fig. 6**, and **Fig. 7** are obtained by getting all the (i) contours detected in the image shown in the rightmost pictures and (ii) then evaluating the highest or maximum contour area from all the detected contours in the original input image. Contour detection algorithm implementation was found to be useful when the object occupies very little space in the image with brighter background and becomes impossible to be detected by Shift and Brute Force Matching.

C. Meanshift

The Meanshift method uses a region of interest point available from the annotation folder under 'bbox' category to create a mask containing the detected portion of the object (highlighted portion) in the image and uses a sliding window method with the help of mean shift algorithm [6] to detect the Object in the image. Therefore, the method starts with first getting the region of interest [9] of the Input Image from the annotation file. Then the region of interest portion of the image is converted to HSV format. Histogram is calculated for the region of interest of only the hue channel of HSV converted region of interest. Following this process, the 'Back Projection' Algorithm of Open CV is used to highlight the important portions of HSV converted original input image with the help of a histogram of the hue channel of the region of interest.

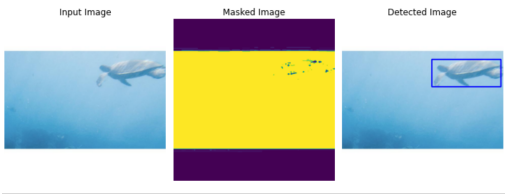

Fig. 8. Original Input Image (id: 38), Masked Image, and Detected Object in the Image

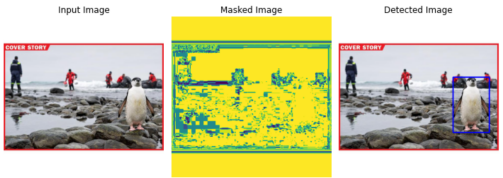

Fig. 9. Original Input Image (id: 341), Masked Image, and Detected Object in the Image

Thus, in highlighting the important portions of the HSV-converted Input Image, a mask is created shown in Fig. 8 and Fig. 9. After, then a detector window is used which starts from the initial position (i.e. coordinates of the region of interest got from the annotation file). The mean Shift Algorithm has been applied on the detector window to slide on the obtained HSV mask for 10 iterations and the window slides by 1 point at a time on the mask. A stopping criterion has been set which is a standard value obtained from the Open CV library. This means the window will stop sliding on the mask when the center of the detected window and the centroid of points produced in the mask differed by epsilon error. After, the end of 10 iterations, updated coordinates from the window are returned which are then used to detect objects in the original input image.

Images that are shown in **Fig. 8**, and **Fig. 9** consisting of the Original Input Image (from left), in the middle lies the Masked Image which is formed by applying Back Projection on the HSV-converted input Image with the region of interest histogram of the hue channel. The Rightmost Image displayed in **Fig. 8** and**Fig. 9** is the detected object in the image which is formed by the mean shifting operation of the window on the masked image for 10 iterations by shifting 1 point at a time.

D. HOG

Histogram of Orientation Gradients (HOG) is a feature descriptor that is particularly effective for object detection.HOG captures the structure of an object by calculating the gradient localized to the image. The idea behind HOG is that the shape of an object in an image can be inferred from its intensity gradient or edge orientation.

To compute the HOG features:

1) The image is divided into multiple small connected regions (called cells), each of which compiles a histogram in the gradient direction (or edge direction).
2) The histograms of cells within larger areas were then normalized to account for changes in light and contrast.
3) A feature vector is then obtained by flattening the histogram of all blocks of the image.

The resulting HOG features capture the appearance and shape of objects in an image, making them particularly suitable for tasks such as human detection.

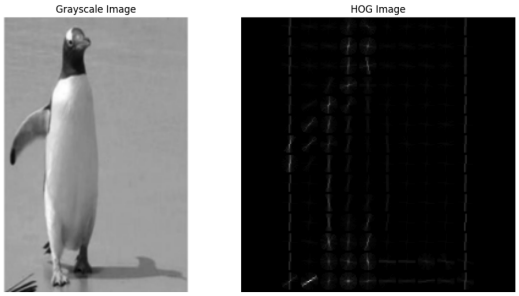

Fig. 10. An example showing the original image (left) and its HOG visualization (right). The HOG visualization captures the primary orientations of the gradients in the image, representing the object's structure.

E. ORB

Oriented FAST and Rotated BRIEF (ORB) is a feature descriptor that is a valid alternative to SIFT and SURF.ORB is essentially a fusion of the FAST keypoint detector and the BRIEF descriptor, with a number of modifications to improve its performance.

The primary steps involved in ORB feature extraction are as follows:

1) **Keypoint Detection using FAST:** ORB uses the FAST algorithm to detect key points in an image.FAST is known for its computational efficiency and ability to capture salient points.
2) **Orientation Assignment:** Each keypoint is assigned a direction based on the center point of intensity of the patch around it. This direction ensures that the rotation remains constant.
3) **BRIEF Descriptors:** The BRIEF descriptor is a binary feature descriptor. It generates a binary string for each keypoint by comparing the intensity of pixel pairs in the area around it. The exact pixel pairs are selected based on a pre-calculated pattern.
4) **Rotation Invariance:** The binary descriptors obtained from BRIEF are rotated according to the previously computed orientation, thus ensuring that the descriptors remain consistent even when the image is rotated.

The ORB descriptor is binary and is efficient in matching operations using the Hamming distance.ORBs are computationally efficient and resistant to rotation and illumination variations, making them popular in real-time applications.

F. PCA

Principal Component Analysis (PCA) is a statistical procedure that transforms raw features into a new set of orthogonal features, known as principal components. These components are designed to capture the variance in the data in decreasing

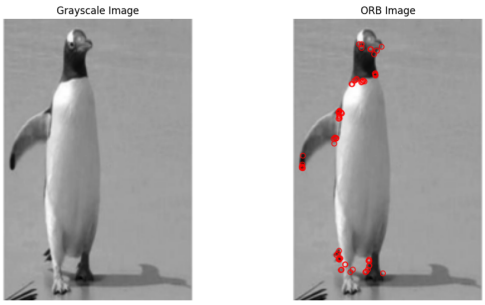

Fig. 11. An example showcasing the original image (left) and its ORB keypoints (right). The keypoints, represented by circles, capture significant structures in the image.

order.The basic idea behind PCA is to reduce the dimensionality of the data set while retaining as much information (variance) as possible.

In the context of this project, PCA serves multiple purposes:

- **Dimensionality Reduction:** By retaining only the highest principal components, the dimensionality of the data is greatly reduced. This not only reduces the computational burden during model training, but also allows for better generalization by alleviating the curse of dimensionality.
- **Noise Filtering:** PCA can help filter out noise in the data. The components with the least amount of variance tend to be noise or unwanted variation. By eliminating these components, the data becomes more robust.
- **Improved Visualization:** Visualization can be challenging for datasets with high-dimensional features.PCA can help project these features into two or three principal components, making visualization feasible and providing insights for data clustering or separation.

In the pipeline, PCA is combined with SVM classifier for animal detection. A grid search method was used to fine-tune the hyperparameters of PCA and SVM. Specifically, the number of components retained by PCA (as part of the total variance) and SVM parameters (e.g., regularization ”C”, ”gamma” of the RBF kernel, and kernel type) were optimized by cross-validation. The final model is then trained using the best combination of parameters to ensure the best performance on the validation data.

G. Support Vector Machine (SVM)

Support Vector Machines (SVMs) are a set of supervised learning methods recognized for their ability to handle high dimensional data. In this work, SVM classifiers are trained on the basis of features extracted from images.The training and optimization process of SVM involves several key steps:

1) **Initial Training:** SVMs are initially trained on the basis of feature vectors extracted from the training dataset.SVMs try to find the optimal hyperplane to distinguish between classes in the feature space.
2) **Hyperparameter Tuning:** SVMs have several hyperparameters that need to be fine-tuned for optimal performance. These include the regularization parameter (C), kernel coefficients (gamma), and kernel function type. In this project, a grid search is used to find the optimal hyperparameters. The lattice search involves training the SVM for each combination of parameters in the defined lattice and selecting the combination that performs best on the cross-validation dataset.
3) **Evaluation:** Once the optimal hyperparameters are found, the performance of the SVM is evaluated on the validation dataset. Evaluation metrics include the confusion matrix (which provides a summary of true positive, true negative, false positive, and false negative predictions) and the classification report (which includes precision, recall, and F1 scores for each category).

SVM provides flexibility through the use of different kernel functions, making it a powerful classifier for a variety of tasks including image classification. The ability to fine-tune the hyperparameters further ensures that the SVM can be optimized for the specific task at hand.

H. YOLOv5

YOLO, which stands for you only look once, is a popular and efficient object detection algorithm. The main principle of YOLO is, YOLO divide the image (or video) into discrete grids and run the algorithm(including predicting bounding boxes, objectness scores, and class probabilities) directly from each grid, which means the prediction are made in one pass through the neural network which make it exremly efficient compare to other deep learning models and capable of real time object detection on a standard GPU(which is our cheap laptop in this case). However, YOLO may not be as precise in localizing small objects or objects that are close to each other, but this won’t be a problem for our particular object detection task since there is only one object in each image and the object takes a big portion of the image, therefore, yolo is a perfect fit for our penguin-turtle detection task.

1) **Data Preprocessing:**
   - Data format change
     The given label data set in our task is a Jason file which needs to transfer to YOLO label file which is a single text file for each of the images, and the in the format of $[class_labelx_centery_enterwidthheight]$. To do this we write a simple program to read each of the label lists in the Jason dictionary and convert the bounding box to the preferred format and write it to a separate .txt file and rename each of those .txt files to the corresponding image name.
   - Data Augmentation:
     Considering we have a relatively small training data set, overfitting might easily appear during the training process. Therefore, we increased the number of images by flipping, scaling, and cutting off random images from the initial data set and adding it to our training set. By doing augmentation, we increased our image number to 800.
2) **Tunning parameters and adjusting model configuration:**

- Model Selection:
  YOLOv5 introduced various model sizes to accommodate diverse computational resources and speed requirements. As we have only one object per image, the model's efficiency in processing frames rapidly and making accurate predictions aligns perfectly with our requirements. Therefore, YOLOv5s stands out as the smallest and fastest model, making it an ideal choice for our specific task.
- Batch Size:
  Considering the limited size of our dataset, we experimented with smaller batch sizes. Smaller batches can sometimes aid in training on smaller datasets by reducing the risk of overfitting and enabling more frequent weight updates. This adjustment helped our model converge faster and achieve better generalization on the task.
- Learning Rate:
  The learning rate significantly influences the pace of model optimization during training. we slowly adjusted the learning rate to find an optimal balance between rapid convergence and avoiding overshooting the loss landscape.
- Confidence Threshold:
  Considering we expect only one object per image, we chose a higher confidence threshold makes the model more conservative in accepting predicted detections and reducing false positives.

3) **Training process Fig 12** shows the training process of 100 epochs.

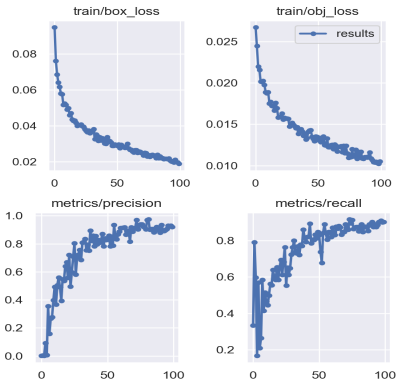

Fig. 12. Training process of YOLOv5

I. *ResNet-50*

ResNet-50 is a convolutional neural network (CNN) that was first introduced by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in their paper, "Deep Residual Learning for Image Recognition" [8]. For this project, we have implemented a modified simpler untrained model of ResNet50 to classify images.

1) **Motivation** The fundamental advantage of ResNet-50 is that it addresses the problem of vanishing and exploding gradients when training deep neural networks. With relatively deep neural networks, gradients can diminish or explode during back-propagation making the entire model quite difficult to train effectively. However, the ResNet-50 model consists of many residual blocks (hence its name "RESidual NETwork"), which enable the training of very deep neural networks. According to the paper "ResNet strikes back: An improved training procedure in timm" (Wightman et al., 2021) [18] ResNet-50 performed well in classifying images from the "ImageNet" [3] data-set, which is a mulitclass dataset containing a wide variety if objects and animals. Hence ResNet-50 was chosen as a method to perform the given task of classifying images of penguins and turtles.
2) **Data Pre-processing and Data Augmentation** To train the model, the images were loaded into test and training sets, both sets were resized to a height of 160 pixels and a width of 160 pixels for faster training and reduced memory usage. The images in the training set were then subjected to data augmentation techniques to increase the training dataset size, exposing the model to more diverse examples of the same image which may lead to better generalisation and increased performance. Furthemore, data augmentation acts as a form of regularisation and prevents overfitting as it introduced variation and noise in the training data making it harder for the model to recognise specific artifacts of the images. This makes the model less likely to overfit on the training data. The specific data augmentation techniques that were applied were:
   - Rotation
   - Translation
   - Flipping
   - Zooming
3) **Architecture**
   The architecture for the model implemented for this task consisted of these key components:
   - Residual Block: The distinguishing feature of the ResNet that solves the problem of vanishing/exploding gradients. Each residual block takes in an input tensor x the number of filters in the convolution layers and the strides for the convolution layer. Batch normalisation and ReLU activation is used at each layer.
   - ResNet: This builds the full ResNet model. It takes in an input shape of the input images, has an inital convolution layer that applies Batch normalisation, ReLU activation follower by a MaxPooling layer. Then, three residual blocks are stacked, each with 64 filters. After that, the model has three more blocks with 128 filters, followed by six blocks with 256 filters, and finally, three blocks with 512 filters.
   - Output: Finally, model performs global average pooling to reduce the spatial dimensions of the output to a single vector. A dropout layer with a dropout rate of 0.2 is applied to prevent overfitting. The paper by Srivastava et al. (2014) demonstrates

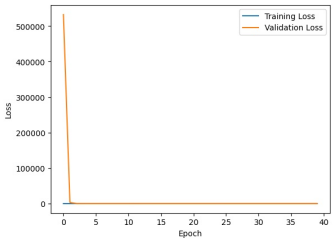

Fig. 13. Loss Graph while Training

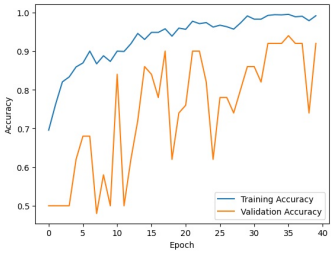

Fig. 14. Accuracy Graph while Training

how dropout is a simple and effective way to prevent overfitting in neural networks [15]. Lastly a Dense layer with a single neuron and sigmoid activation is used to produce a binary classification output since there are only two classes of images.

4) **Tuning** To achieve the optimal performance, these parameters were tuned:
   - Batch size: The batch size was initially arbitrarily set to 16. However, the batch size was later set to 64 as larger batch sizes provide better generalisation, regularisation and stabler gradients.
   - Learning Rate: The learning rate was initially arbitrarily set to 0.1. On further experimentation this proved to be too large a learning rate as the training accuracy started to plateau and the loss kept increasing. Hence, a much smaller learning rate of 0.001 was used and a exponential learning decay function was also added to prevent overshooting and faster initial convergence. This decay function uses a decay rate of 0.96 and decays every 2 epochs.
   - Loss Function: Since the data consisted of only two classes, binary cross-entropy was used as a loss function.
   - Epochs: The model was trained for 40 epochs.
5) **Performance** The trained model had validation accuracy of 0.8194 and a validation loss of with a loss of roughly 0.4. Considering that a fully fledged ResNet50 with an extensive data set achieved around 84% accuracy, these results were acceptable. Furthermore, as shown in Fig 14, after randomly selecting images from the validation set, the model was correctly able to classify them.

IV. Experimental Results

A. Experimental Results of Proposed Detection Method - 1 on Validation Data set

Table 1 represents the evaluated mean and standard deviation between the centers of the predicted and actual bounding boxes across all validation images in the data set. Euclidean distance formulae are used to measure the distances between centers of predicted and actual bounding boxes drawn on images present in the validation data set.

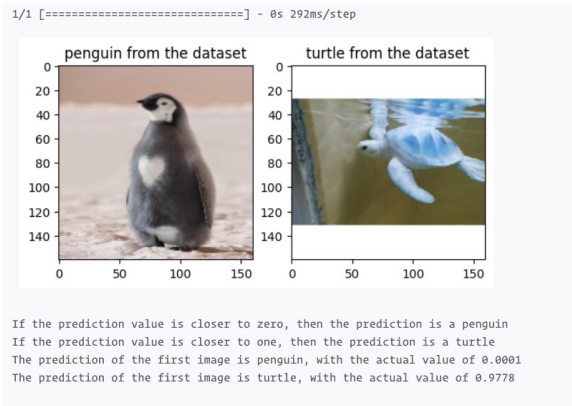

Fig. 15. Predicted classification result of ResNet on a random selection from the test set

TABLE I
Evaluation Metrics for Bounding Boxes by Proposed Detection Method - 1

Evaluation Metrics for Center of Bounding Boxes (Predicted - Actual)	Value
Mean Distance	77.555
Standard Deviation	56.191

The above screenshot of the Jupyter Notebook containing pictures with objects detected in bounding boxes is taken from the validation data set provided. The red-colored bounding box [as seen in **Fig. 11**] is the predicted bounding box i.e. the bounding box drawn by the **Proposed Detection Method - 1** mentioned above and the green-colored bounding box is the actual bounding box provided in the annotation file of the

Fig. 16. Predicted Detected Bounding Box by **Proposed Detection Method - 1**(shown in Red Colour) and Actual Detected Bounding Box (shown in Green Colour)

validation data-set.

In the Images, the left-hand top [as seen in **Fig. 16**]] side gives the IOU metric score between the predicted and actual bounding box drawn on the image, and on each of the Images below corresponding IOU score is provided for reference.

TABLE II
EVALUATION OF IOU METRICS FOR BOUNDING BOXES BY PROPOSED DETECTION METHOD - 1

Evaluation of IOU Metrics for Predicted and its Actual Bounding Boxes	Value
Mean IOU	0.4197
Standard Deviation IOU	0.2650

The values described in **Table 2** above provide the mean and standard deviation on the IOU(Intersection over Union) between the predicted bounding box and the actual bounding box drawn across all the images present in the validation dataset.

TABLE III
EVALUATION METRICS FOR BOUNDING BOXES OF MEANSHIFT METHOD

Evaluation Metrics for Center of Bounding Boxes (Predicted - Actual)	Value
Mean Distance	11.254702030004061
Standard Deviation	11.347344965734406

Table 3 represents the evaluated mean and standard deviation between the centers of the predicted and actual bounding boxes across all validation images in the data set. The predicted boxes are formed on the images using Meanshift algorithm. Euclidean distance formulae are used to measure the distances between centers of predicted and actual bounding boxes drawn on images present in the validation data set. In comparison to the metric values from **Table 1**, it is observed that the mean distance and standard deviation value are very less which means the distance between the center of predicted and actual bounding bounding boxes drawn on input images is very less.

The above screenshot of the Jupyter Notebook containing pictures with objects detected in bounding boxes is taken from the validation data set provided. The red-colored bounding box [as seen in **Fig. 17**] is the predicted bounding box i.e. the bounding box drawn by the **Meanshift** method mentioned above and the green-colored bounding box is the actual bounding box provided in the annotation file of the validation data set.

In the Images, the left-hand top [as seen in **Fig. 17**]] side gives the IOU metric score between the predicted and actual bounding box drawn on the image, and on each of the Images below corresponding IOU score is provided for reference.

In comparison to the images shown in **Fig. 16**, it is observed in **Fig. 17**, the predicted bounding boxes (shown in red color) formed by the Meanshift method in each of the images get almost overlapped with the actual boxes (shown in green color). Therefore, the IOU Scores recorded between each of the figures in **Fig. 17** are almost close to 1, which gives

Fig. 17. Predicted Detected Bounding Box by **Meanshift** Algorithm(shown in Red Colour) and Actual Detected Bounding Box (shown in Green Colour)

the conclusion that the images are nicely detected by the Meanshift algorithm.

TABLE IV
EVALUATION OF IOU METRICS FOR BOUNDING BOXES FORMED BY MEANSHIFT METHOD

Evaluation of IOU Metrics for Predicted and its Actual Bounding Boxes	Value
Mean IOU	0.9014799184790563
Standard Deviation IOU	0.13499728466406788

The values described in **Table 4** above provide the mean and standard deviation on the IOU(Intersection over Union) between the predicted bounding box and the actual bounding box drawn across all the images present in the validation dataset.

In comparison to the mean IOU value in **Table 3**, the mean IOU value in **Table 4** provides a high score close to 1. This means in across all the images present in the Validation Dataset, the objects in images are nicely detected by the Meanshift algorithm.

B. Experimental Results of Proposed Classification Method: HOG+SVM

We used a combination of Histogram of Oriented Gradients (HOG) and Support Vector Machine (SVM) to classify penguins and turtles [1]. The process and results of the method are detailed below.

1) Feature Extraction using HOG: Features were extracted from each image in the dataset using the HOG method. All images were resized to a standard size of 128x128 pixels and converted to grayscale before extracting features.The HOG method captures gradient information, which is critical for distinguishing penguins from turtles based on shape.

2) Model Training with SVM: Support Vector Machine (SVM) is trained based on the extracted HOG features. Initial evaluation was carried out by dividing the training data into a

smaller training set and a test set, where the test set comprised 20% of the data. The model achieved 86% accuracy on the test set.

3) Hyperparameter Tuning: In order to optimize the performance of the model, the hyperparameters of the SVM were fine-tuned using a grid search strategy. Parameters considered included the **C** value that regulates the regularization of the SVM and the **gamma** that affects the shape of the decision boundary. After determining the optimal parameters, the accuracy of the test set increased by 89%.

4) Validation: The fine-tuned SVM model is then applied to the validation set. This step is critical to evaluate the performance of the model on unseen data. The accuracy of the model on the validation dataset was 82%.

TABLE V
CLASSIFICATION REPORT FOR VALIDATION SET USING HOG+SVM

	Precision	Recall	F1-Score
Penguin	0.96	0.67	0.79
Turtle	0.74	0.97	0.84

C. Experimental Results of Proposed Classification Method: HOG + PCA + SVM

We chose a combination of Histogram of Orientation Gradients (HOG), Principal Component Analysis (PCA) and Support Vector Machine (SVM) to classify the images [11]. The process and results are detailed below.

1) Data Loading and Pre-processing: The dataset is divided into training and validation sets and the corresponding annotations are loaded from JSON files. Each image is resized to a uniform size of 128x128 pixels and converted to grayscale. To validate the loading process, a sample image is shown.

2) Feature Extraction using HOG: For each image, HOG features were extracted that captured the gradient information necessary to distinguish penguins from turtles based on shape.

3) Dimensionality Reduction with PCA and Model Training using SVM: We constructed a pipeline by first applying PCA for dimensionality reduction and then training an SVM classifier. A grid search was performed to find the optimal hyperparameters for PCA and SVM. The best parameters identified were a PCA component ratio of 0.8 and SVM parameters $C = 200$, $\gamma = 0.01$, and kernel type 'rbf'.

4) Evaluation on Test Set: The model achieved an accuracy of 87% on the test set. The classification report is summarized in Table VI.

TABLE VI
CLASSIFICATION REPORT FOR TEST SET USING HOG + PCA + SVM

	Precision	Recall	F1-Score
Penguin	0.93	0.81	0.87
Turtle	0.81	0.94	0.87

5) Evaluation on Validation Set: The model was further tested on a separate validation set, achieving an accuracy of 86.1%. The classification report is summarized in Table VII.

TABLE VII
CLASSIFICATION REPORT FOR VALIDATION SET USING HOG + PCA + SVM

	Precision	Recall	F1-Score
Penguin	0.93	0.78	0.85
Turtle	0.81	0.94	0.87

D. Experimental Results of Proposed Classification Method: ORB + SVM

We utilize Oriented Rapid and Rotated BRIEF (ORB) for feature extraction and Support Vector Machine (SVM) for classification. The following is the detailed classification:

1) Feature Extraction using ORB: The ORB algorithm was employed to extract robust keypoints and binary descriptors from the images. Each image was resized to a uniform dimension of 128x128 pixels and converted to grayscale. The algorithm was set to extract a maximum of 200 keypoints.

2) Model Training using SVM: After feature extraction, the SVM classifier was trained on the dataset. The model was initially trained with default parameters. The performance on the test set was evaluated, revealing the need for hyperparameter tuning.

3) Hyperparameter Tuning using Grid Search: To optimize the performance, a grid search was conducted on the SVM's hyperparameters including C, γ, and the kernel type. The best parameters identified were $C = [10]$, $\gamma = [0.0001]$, and kernel type 'rbf'.

4) Evaluation on Validation Set: The model was further evaluated on a separate validation set:

TABLE VIII
CLASSIFICATION REPORT FOR VALIDATION SET USING ORB + SVM

	Precision	Recall	F1-Score
Penguin	0.60	0.33	0.43
Turtle	0.54	0.78	0.64

The validation accuracy achieved was approximately 55.6%.

E. xperimental Results of Proposed Classification Method: Proposed Detection Method-1+HOG+SVM

We further explored another combined approach using a proposed detection method followed by feature extraction with HOG and classification using SVM.

Validation Results: The performance metrics from the validation set for this method are as follows:

TABLE IX
CLASSIFICATION REPORT FOR PROPOSED DETECTION METHOD-1+HOG+SVM

	Precision	Recall	F1-Score
Penguin	0.85	0.64	0.73
Turtle	0.71	0.89	0.79

The overall accuracy of this method on the validation set is 76.39%. This method also illustrates the potential of com-

bining detection and classification, but its accuracy is slightly lower than the "Mean Shift+HOG+PCA+SVM" method.

F. xperimental Results of Proposed Classification Method: Mean Shift + HOG + PCA + SVM

The method combines object detection (using mean shift) and classification (using HOG, PCA and SVM). The following steps were taken for the experimental setup:

1) A Mean Shift algorithm was used to detect the animals in the image.
2) Histogram of Oriented Gradients (HOG) features were extracted from the detected regions.
3) Principal Component Analysis (PCA) was applied to reduce the dimensionality of the HOG features.
4) A Support Vector Machine (SVM) was trained on the PCA-transformed HOG features.
5) The trained SVM was used to classify the validation images.

The performance of this method on the validation set is given as follows:

TABLE X
CLASSIFICATION REPORT FOR MEAN SHIFT + HOG + PCA + SVM

	Precision	Recall	F1-Score
Penguin	0.91	0.86	0.89
Turtle	0.87	0.92	0.89

The accuracy of the method on the validation set is 0.89, indicating its high performance in distinguishing penguin and turtle images.

G. Experimental Results of YOLOv5

YOLO is a one-stage detector that does object localisation (drawing bounding box) and classification in one run, therefore, instead of evaluating classification and detection separately, we showed the overall evaluation result as follows : We also draw the bounding box in the original image to visualise our experiment result and compare it to the ground truth in **Fig 18**. **Fig 19** shows final result curve for YOLOv5.

TABLE XI
EVALUATION RESULT OF YOLOV5

Class	Instances	Percision	Recall	mAP@50	mAP95
Penuine	36	0.959	0.917	0.959	0.764
Turtle	36	0.891	0.905	0.933	0.63

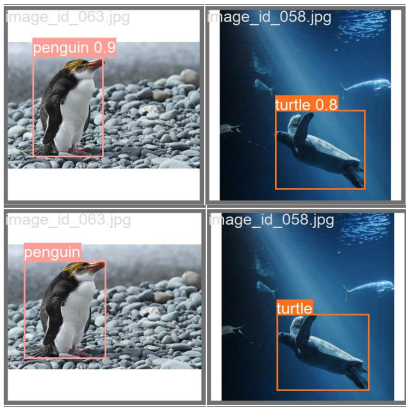

Fig. 18. First row: Predicted Detected Bounding Box by **YOLOv5**. Second row: Ground truth

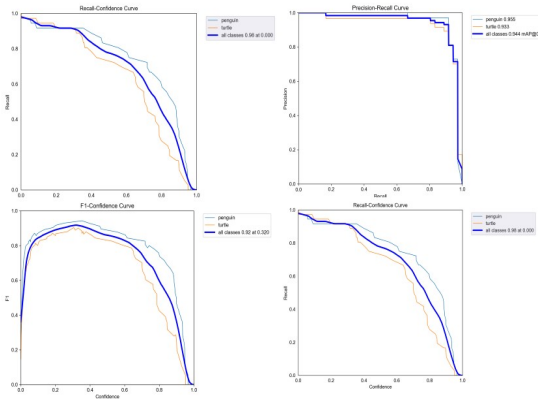

Fig. 19. Final result

.

TABLE XII
COMPARISON OF DETECTION PERFORMANCE

Methods	Deviation of Distance		Intersection Over Union	
	Mean	Standard	Mean	Standard
Proposed method-1	77.555	56.191	0.4197	0.2650
Meanshift	11.255	11.347	0.9014	0.1350

V. DISCUSSION

A. Comparison

For traditional methods in detection task, two main parameters are compared in **TABLE XII**. The proposed detection method-1 showed some limitations in terms of feature detection. Specifically, certain important features, like the eyes and head of the penguins, are not included in the detection result, affecting the overall performance of the model.

This proposed method is evaluated based on the distance between the predicted and actual bounding box centers. The mean distance and standard deviation are found to be 77.555 and 56.191, respectively. While these metrics can serve as a quantitative measure of detection performance, a lower value in both the mean distance and standard deviation is generally preferred for a detection model. This implies that the model could be further improved to achieve more accurate detection results. The Intersection over Union (IoU) is another important metric used for evaluating object detection models. It measures the overlap between the predicted bounding box and the actual bounding box. An IoU of 1 implies a perfect overlap, whereas an IoU of 0 indicates no overlap at all.

In the case of proposed detection method-1, the mean IoU value was found to be 0.4197 with a standard deviation of 0.2650. This indicates that there is some overlap between the predicted and actual bounding boxes, but it also suggests that the model has room for improvement in terms of the precision

of the bounding box predictions.

Considering the results obtained from the "meanshift" detection method, it appears to outperform the "proposed detection method-1" significantly. The mean distance between the center locations of the predicted and actual bounding boxes is considerably lower (11.255 vs 77.555), implying a closer alignment of predicted bounding boxes to the actual ones.

Moreover, the mean IoU is substantially higher (0.9014 vs 0.4197) which indicates that there's a stronger overlap between the predicted and actual bounding boxes using the "meanshift" method, reflecting an increased precision in the detection task.

For nontraditional methods, though it usually combined two task together, Mean Average Precision (mAP) is one of the most important metrics for object detection models in YOLOv5. For Resnet, By measuring Precision and Recall metrics at various thresholds, we can understand how well the model identifies objects and whether it tends to err on the side of making false positive or false negative errors. those will be discussed under classification table.

TABLE XIII
COMPARISON OF CLASSIFICATION PERFORMANCE FOR PENGUIN

Classification Performance			
Methods	Precision	Recall	F1-score
HOG+SVM	0.96	0.67	0.79
HOG+PCA+SVM	0.93	0.78	0.85
ORB+SVM	0.60	0.33	0.43
PDM-1+HOG+SVM	0.85	0.64	0.73
MEANSHIFT+HOG+PCA+SVM	0.91	0.86	0.89
RESNET50	0.8108	0.8164	0.8193
YOLOv5	0.959	0.917	–

TABLE XIV
COMPARISON OF CLASSIFICATION PERFORMANCE FOR TURTLE

Classification Performance			
Methods	Precision	Recall	F1-score
HOG+SVM	0.74	0.97	0.84
HOG+PCA+SVM	0.81	0.94	0.87
ORB+SVM	0.54	0.78	0.64
PDM-1+HOG+SVM	0.71	0.89	0.79
MEANSHIFT+HOG+PCA+SVM	0.87	0.92	0.89
RESNET50	0.8257	0.7946	0.8145
YOLOv5	0.891	0.905	–

TABLE XV
COMPARISON OF CLASSIFICATION ACCURACY

Methods	Accuracy
HOG+SVM	82%
HOG+PCA+SVM	86.1%
ORB+SVM	55.6%
PDM-1+HOG+SVM	76.39%
MEANSHIFT+HOG+PCA+SVM	88.8%
RESNET50	81.9%
YOLOv5	100%

The **TABLE XIII** incorporates seven different classification methods.All these methods are evaluated on two classes: Penguin and Turtle.

From the table XIII, it's evident that the "HOG+SVM" method exhibits high precision for both Penguin and Turtle classes but a relatively lower recall for the Penguin class. The "ORB+SVM" method appears to underperform compared to other methods across all metrics.

The "HOG+PCA+SVM" method seems to deliver the most balanced performance with both precision and recall scores being quite high for both classes. This is mirrored in the F1-score as well, which is the highest among all methods.

The "PDM-1+HOG+SVM" and "MEANSHIFT+HOG+PCA+SVM" methods also perform well on all the metrics and even exceed the "HOG+PCA+SVM" method in certain aspects.

The performance of RESNET50 is also quite steady, with precision, recall, and F1-score all above 80%, showing the advantages of deep learning models in image classification tasks.

YOLOv5 stands out with high levels of precision and recall for both classes, even achieving 100% accuracy.

All methods perform quite well on both classes, but some methods stand out in certain specific metrics. Moreover, even though YOLOv5 outperforms all other methods, the performance of other methods shouldn't be overlooked, especially those with balanced performance across all metrics, like "HOG+PCA+SVM" and "MEANSHIFT+HOG+PCA+SVM".

For further improvement, especially for those classes that lag slightly behind, model tuning or additional training data might be considered.

Overall, the YOLOv5 model demonstrates good detection and classification performance for both "penguin" and "turtle" classes. However, the model shows a slightly better performance on the "penguin" class in both aspects. For further improvement, especially for the "turtle" class, model tuning or additional training data might be considered.

VI. CONCLUSION

In the detection task, the "YOLOv5" method showed superior performance, achieving an accuracy rate of 100%, with precision for the Penguin and Turtle classes at 0.959 and 0.891, respectively. The recall for these classes was also high at 0.917 and 0.905, indicating that the method performed excellently in detecting objects in images.

The "MEANSHIFT+HOG+PCA+SVM" method also showed strong results, with an accuracy of 88.8%. It demonstrated high precision and recall for both classes, demonstrating a well-rounded performance across the board.

The MEANSHIFT+HOG+PCA+SVM method stood out with the highest F1-scores for both Penguin and Turtle classes in the classification task .

The HOG+PCA+SVM method showed balanced performance with high precision, recall, and F1-scores for both classes. While the HOG+SVM method had high precision, its recall, particularly for the Penguin class, was relatively lower. The ORB+SVM method, in contrast, performed the least favorably across all the metrics for both classes.

In conclusion, for the object detection task, the YOLOv5 method outperforms other methods, offering precise and accurate detection. In the classification task, the MEANSHIFT+HOG+PCA+SVM method showed superior performance, with the HOG+PCA+SVM method also displaying

high performance in terms of precision, recall, and F1-score. These insights suggest that the YOLOv5 method for detection and MEANSHIFT+HOG+PCA+SVM method for classification could be recommended for similar wildlife animal image detection and classification tasks in future studies.

ACKNOWLEDGMENT

We would like to acknowledge the UNSW COMP9517 course staff for their support in writing this document.

REFERENCES

[1] Hilton Bristow and Simon Lucey. Why do linear svms trained on hog features perform so well?

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[4] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[5] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

[6] Huimin Guo, Ping Guo, and Qingshan Liu. Mean shift-based edge detection for color image. In *2005 International Conference on Neural Networks and Brain*, volume 2, pages 1118–1122, 2005.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[9] Zengwei Ju, Jingli Zhou, Xian Wang, and Qin Shu. Image segmentation based on adaptive threshold edge detection and mean shift. In *2013 IEEE 4th International Conference on Software Engineering and Service Science*, pages 385–388. IEEE, 2013.

[10] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016.

[11] Mohd. Mustaqeem and Mohd. Saqib. Principal component based support vector machine (pc-svm): a hybrid technique for software defect detection. *Cluster Computing*, 24:2581–2595, 2021.

[12] C.P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 555–562, 1998.

[13] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[14] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[15] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[16] Hongzhi Wang and John Oliensis. Generalizing edge detection to contour detection for image segmentation. *Computer Vision and Image Understanding*, 114(7):731–744, 2010.

[17] Hongzhi Wang and John Oliensis. Generalizing edge detection to contour detection for image segmentation. *Computer Vision and Image Understanding*, 114(7):731–744, 2010.

[18] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *CoRR*, abs/2110.00476, 2021.

[19] Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, and Brian Lee. A survey of modern deep learning based object detection models. *Digital Signal Processing*, 126:103514, 2022.