COMP9517 group project report

Xiaomiao Yu z5438351

Boyang Pan z5380749

Jiayang Jiang z5319476

LinXing Jia z5382484

Xinrui Yao z5430157

I. Introduction

Detection and classification are two basic tasks in computer vision, which have a very wide range of requirements in practical applications. With the development of big data and artificial intelligence, human-made target detection and classification have become a thing of the past. Existing deep neural network based object detection and classifiers have achieved very promising performance and have been applied in various aspects. This project is mainly aimed at the above two basic computer vision needs, using the data of two types of animals, turtles and penguins, to carry out related research. The corresponding analysis will be carried out from the two aspects of task description and dataset.

4.1 Task Specification

The goal of this project is to first perform target detection on turtles and penguins, and then classify them. Two types of approaches are described in the task specification. The first is to use two independent machine learning models to accomplish the two goals of detection and classification respectively; the other is to use one machine learning model to realize the integration of detection and classification.

4.2 Dataset

This project uses turtle and penguin datasets to verify the performance of detection and classification models. The dataset has been split into training and testing sets, containing 500 and 72 images. It can be found that the category information and the ground truth of the object detection frame are provided in the annotation file, so this dataset can support the completion of the above two tasks.

Currently, there are a variety of object detection and recognition algorithms. The project uses HOG+SVM and KerasCV to solve object detection.

II. Literature Review

Object detection is a very important research direction of computer vision, its main tasks are target localization and Object classification. Object detection is crucial in vision processing tasks such as object tracking, digit recognition et al. At the same time, Object detection is also widely used in pedestrian detection, text detection, remote sensing Object detection and other fields. Traditional Object detection methods include region selection, manual feature extraction, classifier classification, et al. After the development of deep learning and machine learning, there are many detection methods based on neural networks with excellent performance. Object detection methods based on deep learning can be divided into three categories: two-stage object detection, single-stage object detection, and Transformer-based object detection. This paper mainly introduces the traditional detection method combined with the histogram of oriented gradient (HOG) and the support vector machine (SVM) algorithm and some mainstream methods in deep learning.

2.1 HOG+SVM

HOG algorithm is a feature extraction method used for image Object detection. This method extracts the contour information of the Object by calculating the gradient of each pixel in the image. The specific step is to first convert the color image into a grayscale image, and then divide the image into several small blocks, calculate the gradient and direction of the pixels in the small block, and then calculate the direction distribution of the gradient in each small block, and do normalization processing of the gradient histogram to reduce the influence of other factors. Finally, the feature vector is extracted. Hog usually with machine learning algorithms (such as support vector machine) for the combination of characteristic vectors for training and classification, Object detection and recognition.[1]

The advantage of HOG algorithm is that it has certain robustness to the factors such as illumination and shadow in the image, and can extract the Object features and retain the image details in complex scenes. Compared with traditional feature description algorithms, HOG algorithm does not need to extract feature points from images, so it has significant advantages in object detection algorithms.[2]

The support Vector Machine (support Vector Machine) is a supervised learning algorithm. This method is mainly to find hyperplanes that can divide data in n-dimensional space into two classes, where n represents the number of features of the data. Then separate different categories of samples.[1]

The advantage of this algorithm is that it has strong generalization ability and performs well on small sample data sets, but it needs more computing resources and time to compute large-scale data sets.

2.2 YOLO

YOLO (You Only Look Once) is a one-stage object detection algorithm in deep learning. Compared with traditional object detection algorithms, YOLO uses CNN convolutional neural network to extract features and identify categories and locations. The position and category of bounding box of target can be predicted directly on the image by a single neural network model.

The YOLO algorithm has been continuously optimized and improved since it was proposed in 2016, and the existing YOLO algorithms include YOLOv2, YOLOv3, YOLOv4 and YOLOv5, as well as the latest v7 version.YOLOv7 improves the shortcomings of the previous version by performing up-sampling and dimensional splicing operations in the last layers of neural network prediction to improve the detection effect of small targets.[3]

Compared with other neural network-based algorithms, YOLO uses the entire image as the input layer of the neural network and simultaneously predicts the location and category of all targets, which can more accurately locate and classify targets. It also has better adaptability to different scale targets.

2.3 TensorFlow and Keras Framework

Keras is an open source deep learning library originally developed by Francois Chollet. It provides an advanced neural network API that can run on a variety of deep learning frameworks including TensorFlow, Microsoft Cognitive Toolkit (CNTK), and Theano[4].

The emergence of Keras improves the efficiency of models and experiments built based on deep learning. Keras provides many commonly used neural network layers, such as full connection layer, convolutional layer, cyclic layer, et al. Users can use the Keras library to complete various neural network

tasks, such as object detection and image classification in this paper[5]. For example, in the detection and classification tasks in this article, the pre-trained neural network res-net is used for model training.

III. Method

There are two different methods used in this project. One is the traditional type, which is the histogram of oriented gradients with the support vector machine. One is the modern type with neural network training, that is KerasCV

3.1 HOG with SVM

a) Explanation of HOG

In order to use the HOG method, we need to pre-process pictures with kernel functions as such.

Fig 3.1

With given filtered information of the pictures, we can calculate the gradients with a cell with a fixed size of pixels. Here is an example:

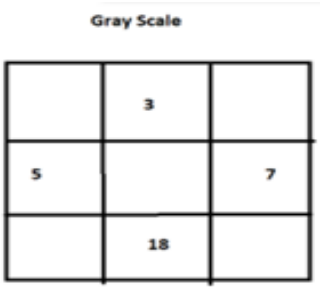


Fig 3.2

To calculate the gradients magnitude, we use the function:

$$\sqrt{|7-5|^2+|18-3|^2}$$

To calculate the gradient direction, we use the function:

$$\arctan\left(\frac{18-3}{7-5}\right)$$

Divide orientations into N bins and assign the gradient magnitude of each pixel to the bin corresponding to its orientation => cell histogram.

For example, if we set n to 9, each bin is separated by 20 degrees. Here is an example[6]:

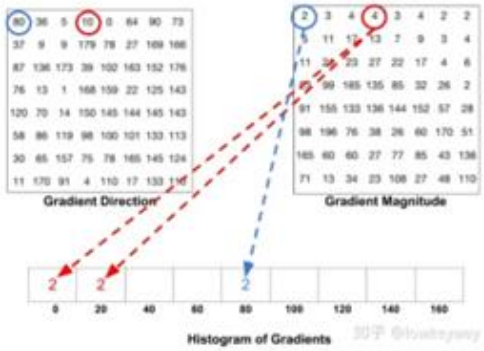


Fig 3.3

Fig 3.4

Then we combine several cells into blocks.

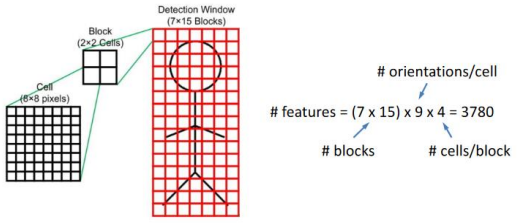


Fig 3.5

Then we can go through all pixels by shifting detection windows to get all

information of gradient magnitude and directions.

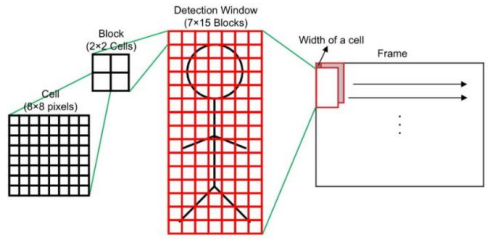


Fig 3.6

In our project, we used cells size of 32x32,3x3 cells as a block and 9 bins.

b) Explanation of SVM

In this project, we used liner SVM.

With the dataset we have from HOG. We try to find a linear function that separates two different groups of data.

There are two key functions to find the margin line:

$$\min_{w,b} \frac{1}{2}\|W\|_2^2 \ s.t. \ y_i\left(W^T x_i + b\right) \geq 1, \ \ \forall i$$

$$\text{Margin:} \frac{1}{\|W\|_2}$$

So that we can maximize the distance between the margin line and the closest example. Also, positive class and negative class samples are on each side of the margin line.

Classes are defined by function:

$$W^T x_i + b \geq 0 \text{ if } \ y_i > 0;$$

$$W^T x_i + b \leq 0 \text{ if } \ y_i < 0$$

3.2 KerasCV

Keras cv uses YOLO method to detect and classify objects. We will resize the input image of 484x484. Then the resized image will be divided into several grills. Each grid cell predicts B bounding boxes and confidence scores for those boxes. Scores would be created by method IoU. Each bounding box consists of 5 predictions: x, y, w, h, and confidence. (x,y) is the center of the gill box. Confidence value here means how confidence it thinks the box has an object and how accurate each box bound is. It is defined as follow:[7]

$$\Pr\left(Object\right) \times \text{IoU}_{pred}^{truth}$$

Each grid cell also predicts conditional class probabilities: [7]

$$\Pr\left(Object\right) \times \text{IoU}_{pred}^{truth}$$

Finally, we use the following algebra to calculate final confidence of each class in the image:

$$\Pr\left(Object\right) \times \text{IoU}_{pred}^{truth} \times \Pr\left(Class_i \mid Object\right)$$

The result of this step is as shown.[7]

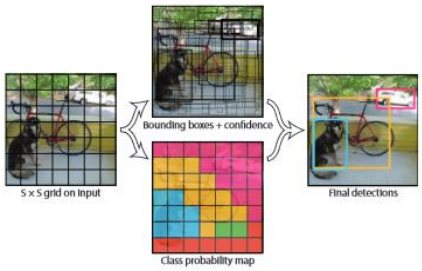


Fig 3.7

Then we need to make it go through our network, which contains 24 convolutional layers followed by 2 fully connected layers.[7]

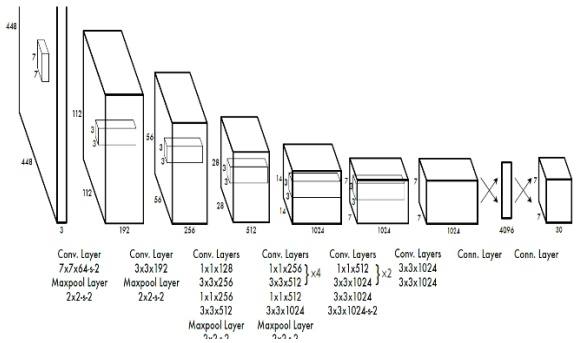


Fig 3.8

IV. Experimental Results

4.1 HOG+SVM

The model training is initially conducted utilizing the HOG+SVM method. The JSON file is read, and based on its annotated categories in train annotations, the images are duplicated into two distinct folders according to the classification of penguins and turtles. The feature.hog method from skimage is employed to extract HOG features from the input images. Three parameters are set as in Table 4.1.

Orientations	9	The quantity of segments for gradient direction. Within each segment, all gradients are considered as the same type. 180 degrees are divided into 9 segments, each segment being 20 degrees.
Pixels per cell	(32, 32)	The pixel size for each cell. Within each cell, a histogram of gradient directions is calculated. Each cell contains 32x32 pixels.
Cells per block	(3, 3)	The number of cells in each block. The histograms of all cells within the block are connected together to form the block's histogram. Each block contains 3x3 cells.

Table 4.1 Parameters in HOG

The images are resized to dimensions of (256, 256). For each positive and negative sample image, the image is read, converted into a grayscale image, resized, the HOG features are extracted, and then these features are stored in a file. All files in the train folder are iterated through, features are loaded and added to the data list, with corresponding labels added to the label list simultaneously. Joblib is used to load data and labels. The Linear Support Vector Machine classifier from the scikit-learn library is utilized to create a LinearSVC object. The model is trained using the data and labels, and the trained model is saved. For each image in the test set, it is similarly converted to grayscale, resized, HOG features are extracted, and then the model is used for prediction. The matching status between the prediction results and real labels is calculated. The classification effect results are listed in Table 4.2.

Precision	0.61	TP	22
Recall	0.92	FN	2
F1_score	0.73	FP	14
Accuracy	0.78	TN	34

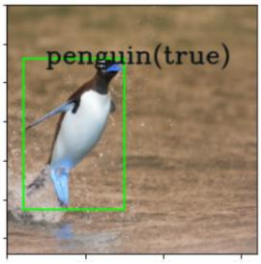
Table 4.2 Classification results of HOG+SVM

The SVM model trained as per the above-mentioned process continues to be employed. The input image is read and converted into a grayscale image. The image is subjected to scale pyramid processing. At each scale, the image is traversed using a sliding window.

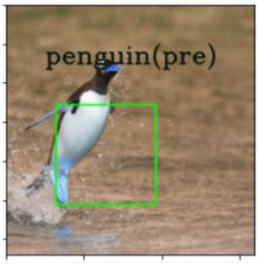
In this scenario, the box size is set as (256,256), the step size is (128,128), and the stride is 1. For each image fragment in the sliding window, its HOG features are extracted, and the model is used for prediction. If the model predicts the image fragment as a positive class (penguin) and the prediction confidence is greater than -0.5, then the position and size of this image fragment are added to the detection results. Non-maximum suppression is performed on all detection results to filter overlapping and redundant detection results.

The detection results, after non-maximum suppression, are drawn on the original image. The image with drawn detection results and the position and size of the last detection box are obtained.

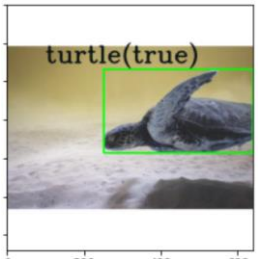
Examples of the performance are shown in Fig 4.1. The note Pre refers to performance of prediction.



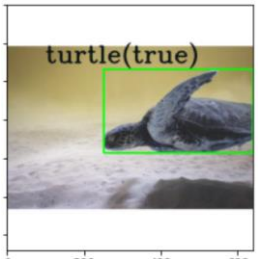
Correct Image Annotation(a)



Output images (a)



Correct Image Annotation(b)



Output images (b)

Fig 4.1 Classification and detection performance of HOG+SVM

The statistical data of all detection results with bounding boxes are listed in Table 4.3.

mean of the distances	199.32
standard deviation of the distances	140.01
mean of the IoU (Intersection over Union)	0.22
standard deviation of the IoU	0.17

Table 4.3 Detection results of HOG+SVM

4.2 KerasCV

KerasCV is also employed for model training and prediction. All input images are resized to (128,128), and a random horizontal flip is performed to increase the diversity of the training data.

A TensorFlow dataset containing images, labels, and bounding boxes is created for both training and test images. A custom Keras callback function is defined, intended for the computation of the COCO (Common Objects in Context) metrics at the end of each training epoch.

A RetinaNet model is created; RetinaNet is a single-stage object detection model, implying it manages the classification and localization (i.e., bounding box regression) of objects as a unified task. And a Stochastic Gradient Descent (SGD) optimizer is created and set with parameters:

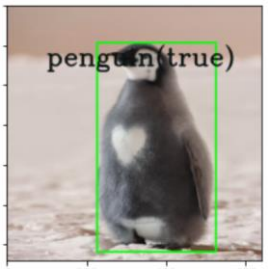
learning_rate=base_lr

momentum=0.9

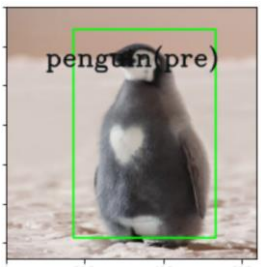
global_clipnorm=10.0

The model is compiled with specifying the classification loss function (used for predicting each object's category) as 'focal', the bounding box loss function (used for predicting each object's location) as 'smoothl1', along with the optimizer.

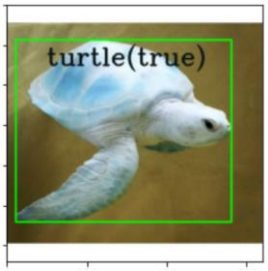
The model is trained using the training set. The training is performed for 20 epochs, and the COCO metrics are calculated at the end of each training epoch. Examples of the performance are shown in Fig 4.2.



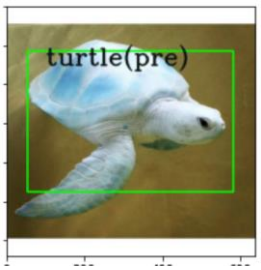
Correct Image Annotation(a)



Output images (a)



Correct Image Annotation(b)



Output images (b)

Fig 4.2 Classification and detection performance of KerasCV

The trained model is used to evaluate the test set, with classification results as follows:

Precision	0.86	TP	31
Recall	0.86	FN	5
F1_score	0.86	FP	5
Accuracy	0.86	TN	31

Table 4.4 Classification results of KerasCV

The detection results with bounding boxes are illustrated as follows:

mean of the distances	64.86
standard deviation of the distances	134.59
mean of the IoU(Intersection over Union)	0.61
standard deviation of the IoU	0.22

Table 4.5 Detection results of KerasCV

4.3 Environments

All simulations are operated by Jupyter notebook on devices with 12th Gen Intel i5 CPU cores with 2×8 GB RAM. On average, the entire process using the HOG+SVM method takes 45 seconds, and the one using the KerasCV method takes 2 hours and 43 minutes.

V. Discussion and Compare

Traditional object detection methods typically use image processing techniques to classify and locate corresponding targets, which are often based on manually designed features such as the most famous Viola Jones algorithm and the hog algorithm we use in our projects. The basic process of these methods is to first use the sliding window method to select candidate regions in the image, then customize design features based on different regions, and finally complete detection based on the customized underlying features.

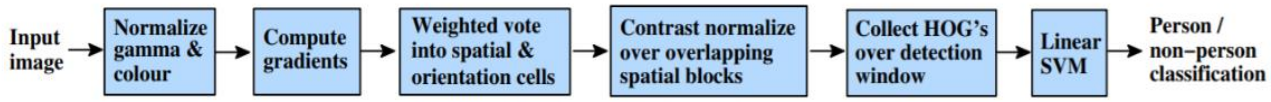


Table 5.1 steps to do object detection for HOG+SVM

In the early years, HOG+SVM model as a traditional method is popular which is the best choice to solve object detection. The core idea of Hog is that the shape of the detected local object can be described by the distribution of gradients or edge directions, and HOG can better capture local shape information, with good invariance to geometric and optical changes. Also, SVM has outstanding performance in solving machine learning and classification in small sample situations. Due to the use of sliding windows to detect images from left to right and from top to bottom in the HOG algorithm to locate targets in the image, even if non maximum suppression algorithms are used, many redundant bounding boxes will still be generated, resulting in relatively low precision(61.11%) and high recall(91.67%) in the HOG+SVM algorithm. The region selection strategy based on sliding windows is not targeted, with high Time complexity and redundant windows. In addition, the hog feature is a manually designed feature, which is not very robust to changes in diversity. In some very similar and occluded images, detection failures often occur. This also leads to significant deviation in bounding box prediction.

	mean of Distance	Standard Deviation of the Distances	mean of IoU	Standard Deviation of the IoU
HOG+SVM	199.3166	140.0076	0.2206	0.1723
kerasCV	64.8555	134.5883	0.6126	0.2199

Table 5.2 detection performance

By comparison, the accuracy and performance of kerasCV are relatively high. It is a deep learning based object detection method that provides a series of advanced APIs for solving object detection. These APIs include object detection specific data augmentation techniques, Keras native COCO metrics, bounding box format conversion utilities, visualization tools, and pre-trained object detection models. KerasCV uses MPL to train data, MPL refers to multiple layers composed of artificial neurons that can be Perceptron. The hierarchy of MPL is a Directed acyclic graph. Usually, each layer is fully connected to the next layer, and the output of each artificial neuron on a certain layer becomes the input of several artificial neurons in the next layer. It has multiple layers and a wide width, which can theoretically be mapped to any function and can solve many complex problems. Therefore, this neural network has strong learning ability, from training, testing, to validation, it has performed very well. However, there are also some cases of detection failures in KerasCV, which may be due to insufficient training and learning of the neural network due to insufficient dataset size, or due to too similar images and inappropriate parameter settings.

	Precision(%)	Recall(%)	F1_score(%)	Accuracy(%)	Training time
HOG+SVM	61.11	91.67	73.33	77.78	45 seconds
kerasCV	86.11	86.11	86.11	86.11	2 hours 43 mins

Table 5.3 detection performance

Finally, for training time, almost all deep learning based object detection methods use gradient descent algorithms to update parameters, and many times forward and backward are needed. Therefore, it will take a long time to train a model when the data volume is large. However, traditional methods do not need many parameters, only a hog feature vector needs to be generated for binary classification. Therefore, the training time of HOG+SVM much less than that of kerasCV.

VI. Conclusion

In conclusion, we successfully implemented object detection on ‘turtle vs penguin’ dataset using two methods, which are HOG+SVM and KerasCV. In terms of training speed, the traditional method of HOG+SVM(45 seconds) is significantly faster than Keras_ CV, but due to the limited number of training sessions, its performance and accuracy are relatively average. The feature descriptor acquisition process of the HOG algorithm is complex and has high dimensionality, resulting in poor real-time performance. This method is also difficult to handle occlusion issues, and it is also difficult to detect large amplitude of human posture movements or changes in object direction, which is also the main reason for the low accuracy of HOG+SVM. In addition, HOG+SVM is only suitable for target detection in small datasets. Once the dataset is too large or there are too many data volumes and types, its performance will be greatly reduced. At this point, kerasCV solves this problem very well. It is a computer vision library based on deep learning, which uses MPL to train data and has an internal data augmentation API. It could not only train very large datasets, but also greatly surpass HOG+SVM in terms of performance and accuracy and finally

accurately predict the position and classification of objects in the image, as well as depict corresponding bounding boxes. However, due to the high number of training times and the use of artificial neural networks, its training time is also very long (2 hours 43 minutes).

In the future, Deep learning methods will become the mainstream method for solving object detection, but traditional detection methods are still applicable for some simple object detection problems, such as pedestrian detection and cat dog classification. For the sake of traditional methods of intensive reading, we will try different image preprocessing methods, such as image enhancement, image denoising, Gaussian smoothing, etc. to better extract image features from the HOG algorithm. At the same time, DPM, as an improvement of the HOG algorithm, is also a good choice. Moreover, in order to improve Keras we will try different data augmentation methods and build my own neural network to improve the training speed of kerasCV.

References:

[1] Weifeng, W, Baobao, Z, ZhiQiang, W, FangZhi, Z & Qiang, L 2021, 'Garbage image recognition and classification based on hog feature and SVM-Boosting', Journal of physics. Conference series, vol. 1966, no. 1, p. 12002.

[2] Xing, W, Deng, N, Xin, B, Liu, Y, Chen, Y & Zhang, Z 2019, 'Identification of Extremely Similar Animal Fibers Based on Matched Filter and HOG-SVM', IEEE access, vol. 7, pp. 98603–98617.

[3] Wang, C-Y, Bochkovskiy, A & Liao, H-YM 2022, 'YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors'.

[4] Nguyen, G, Dlugolinsky, S, Bobák, M, Tran, V, López García, Álvaro, Heredia, I, Malík, P & Hluchý, L 2019, 'Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey', The Artificial intelligence review, vol. 52, no. 1, pp. 77–124.

[5] Harjoseputro, Y 2020, 'A Classification Javanese Letters Model using a Convolutional Neural Network with KERAS Framework', International journal of advanced computer science & applications, vol. 11, no. 10.

[6] "（四十六）OpenCV HOG+SVM 的物体检测."
https://zhuanlan.zhihu.com/p/94934407
Accessed 1 Aug. 2023

[7].Redmon, J. et al. (2016) You only look once: Unified, real-time object detection, arXiv.org. Available at: https://arxiv.org/abs/1506.02640 (Accessed: 01 August 2023).