

COLLEGE OF COMPUTER STUDIES

Perception and Computer Vision

Gagalang, Mark Lester A.
Galzote, Gilela Anne M.
BSCS 4A

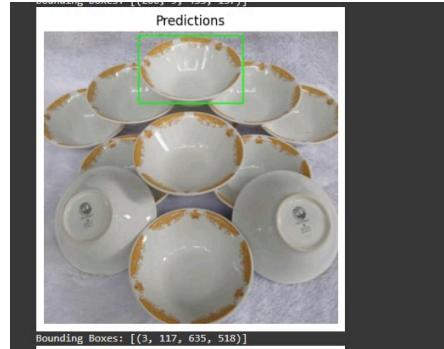
SELECTION OF DATASET & ALGORITHM

To use the algorithms which are Histogram of Oriented Gradients with Support Vector Machine (HOG-SVM) and You Only Look Once (YOLO), the programmers will be needing a dataset in order for it to hold information to be used by a program running on the system. The selected dataset used was found in roboflow. In this activity, the programmers' dataset object was a bowl that was specifically trained that has different varieties of designs. It identifies the size of the bowl in order to classify what size of a box is accurately suitable for the bowl that was detected.

With the usage of Histogram of Oriented Gradients with Support Vector Machine (HOG-SVM) have made the programmers workload light, as the algorithm has only extracted the features as it was being applied to the chosen dataset. It is automatically being processed as well as continuous which lessen the struggles that they may encounter. In addition, HOG-SVM doesn't have any training model process since the chosen data is pre-trained already.

```
Liblinear failed to converge, increase the number of iterations.  
Liblinear failed to converge, increase the number of iterations.  
Best parameters found: {'C': 0.1, 'max_iter': 1000}  
HOG-SVM model trained successfully in 21.50 seconds.  
Validation Accuracy: 0.9886363636363636  
Precision: 0.9777777777777777  
Recall: 1.0  
Testing Accuracy: 0.8  
Precision: 0.7142857142857143  
Recall: 1.0  
Detection time for test set: 0.002869129180908203
```

There are no any struggles that the programmers have noticed besides with images that have 3 or more bowls being included. It can only detect one bowl during the testing model process.



Meanwhile, in YOLO (You Only Look Once), the process of its data preparation has met some errors. The programmers have figured out that the extension needed to be renamed. The images can't be read due to its file format. Instead of .jpg, it only reads .JPG. It provides a long training time, since the pre-trained datasets undergo a new training process although it has a low percentage of epoch. Throughout the usage of the algorithm, YOLO, the programmers haven't encountered any problems during its testing process.

EVALUATION

Histogram of Oriented Gradients with Support Vector Machine (HOG-SVM) and You Only Look Once (YOLO) both reached a high percentage of accuracy. Though both have no problems during the training process as well as the results that were provided.

COMPARISON

Histogram of Oriented Gradients with Support Vector Machine (HOG-SVM) and You Only Look Once (YOLO) ran well in Google Colab. The difference was HOG-SVM has its fastest training time compared to YOLO. Based on the programmers' observation, YOLO trained the pre-trained dataset wherein it caused the slow progress of output using the algorithm as it involves fine-tuning weights across multiple layers. HOG-SVM requires fewer computational resources, as it focuses on detecting shapes and edges; it had gained the most accurate result due to the effect of training ten (10) epochs although the suggested minimum number of epochs was fifty (50).

```

print("Testing Accuracy:", test_accuracy)
print("Precision:", precision)
print("Recall:", recall)

# Speed Evaluation
start_time = time.time()
_ = grid.predict(test_features) # Just for timing
detection_time = time.time() - start_time
print("Detection time for test set:", detection_time)

Liblinear failed to converge, increase the number of iterations.
Liblinear failed to converge, increase the number of iterations.
Best parameters found: {'C': 0.1, 'max_iter': 1000}
HOG-SVM model trained successfully in 21.50 seconds.
Validation Accuracy: 0.9886363636363636
Precision: 0.9777777777777777
Recall: 1.0
Testing Accuracy: 0.8
Precision: 0.7142857142857143
Recall: 1.0
Detection time for test set: 0.002869129180908203

```

```

# Access and calculate mean values for evaluation metrics
precision = results.box.p.mean() # Mean Precision across classes
recall = results.box.r.mean() # Mean Recall across classes
map50 = results.box.map50.mean() # Mean mAP@0.5 across classes
map50_95 = results.box.map.mean() # Mean mAP@0.5:0.95 across classes

print("Evaluation Results:")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"mAP@0.5: {map50:.4f}")
print(f"mAP@0.5:0.95: {map50_95:.4f}")

Ultralytics 8.3.28 Python-3.10.12 torch-2.5.0+cu121 CPU (Intel Xeon 2.28GHz)
val: Scanning /content/Deteksi-Bowl-1/valid/labels.cache... 21 Images, 0 backgrounds
Class Images Instances Box(P R mAP50 mAP50
all 21 21 0.675 0.905 0.795
Speed: 3.2ms preprocess, 251.3ms inference, 0.0ms loss, 6.8ms postprocess per image
Results saved to runs/detect/train33
Evaluation Results:
Precision: 0.6749
Recall: 0.9048
mAP@0.5: 0.7959
mAP@0.5:0.95: 0.4203

```