Journal of Engineering
journal homepage: www.joe.uobaghdad.edu.iq
Number 8 Volume 27 August 2021

Mechanical and Energy Engineering

Practical comparation of the accuracy and speed of YOLO, SSD and Faster RCNN for drone detection

S. Alkentar *
Student at Al-Baath University
Homs-Syria
Saad.zgm@gmail.com

B. Alsahwa
Lecturer at HIAST
Damascus-Syria
bassems74@netcourrier.com

A. Assalem
Lecturer at Al-Baath University
Homs-Syria
assalem1@gmail.com

D. Karakolla
Lecturer at HIAST
Damascus-Syria
daoud.karakola@gmail.com

ABSTRACT

Convolutional Neural Networks (CNN) have high performance in the fields of object recognition and classification. The strength of CNNs comes from the fact that they are able to extract information from raw-pixel content and learn features automatically. Feature extraction and classification algorithms can be either hand-crafted or Deep Learning (DL) based. DL detection approaches can be either two stages (region proposal approaches) detector or a single stage (non-region proposal approach) detector. Region proposal-based techniques include R-CNN, Fast RCNN, and Faster RCNN. Non-region proposal-based techniques include Single Shot Detector (SSD) and You Only Look Once (YOLO). We are going to compare the speed and accuracy of Faster RCNN, YOLO, and SSD for effective drone detection in various environments. We have found that both Faster RCNN and YOLO have high recognition ability compared to SSD; on the other hand, SSD has good detection ability.

Keywords — Faster RCNN, YOLO, SSD, Drone

مقارنة عملية لدقة وسرعة خوارزميات YOLO, SSD و Faster RCNN لكشف الطائرات المسيرة الصغيرة Drones

سعد القنطار *
طالب مهندس في جامعة البعث
حمص- سوريا

باسم السهوة
دكتور محاضر في المعهد العالي للعلوم التطبيقية والتكنولوجيا
دمشق- سوريا

عبد الكريم السالم
دكتور محاضر في جامعة البعث
حمص- سوريا

داوود كركولله
دكتور محاضر في المعهد العالي للعلوم التطبيقية والتكنولوجيا
دمشق- سوريا

الخلاصة

يمكن تقسيم الخوارزميات المستخدمة للكشف والتعرف إلى خوارزميات يدوية التصميم hand-crafted وخوارزميات تعتمد على التعليم العميق Deep learning (DL). يمكن تقسيم أنظمة الكشف بالتعليم العميق إلى أنظمة كشف ثنائية المرحلة (طريقة المناطق المقترحة) وأنظمة كشف وحيدة المرحلة (بدون اقتراح مناطق). تهدف الكواشف وحيدة المرحلة إلى إزالة الحاجة إلى استخراج المناطق المقترحة عن طريق القيام بهذه العملية في نفس الشبكة. تتميز هذه الكواشف بسهولة التدريب وفعالية حسابية عالية. لعل ابرز الخوارزميات التي تعتمد على مرحلتين Faster RCNN ومن ابرز الخوارزميات التي تعتمد على مرحلة واحدة You only look once (YOLO) وSingle

*Corresponding author
Peer review under the responsibility of University of Baghdad.
https://doi.org/10.31026/j.eng.2021.08.02

Article received:27/11/2020
Article accepted: 12/5/2021
Article published:1/8/2021

shot detector (SSD). سنقوم في هذا البحث إلى بمقارنة أداء هذه الخوارزميات بهدف أيجاد الأفضل منها لكشف الطائرات الصغيرة Drone بشكل فعال في مختف البيئات.

الكلمات المفتاحية: طائرات مروحية صغيرة بدون طيار Drone , تعليم عميق, Faster RCNN, YOLO, SSD

I. INTRODUCTION

Feature extraction and classification algorithms can be either hand-crafted or DL-based. Hand-crafted methods for feature extraction are based on manually designed models that work on low-level features to propose Regions of Interest (ROIs) **(Sun, W., et al., 2018)**. Those models were based on techniques such as Background Subtraction (BS), the Histogram of Oriented Gradients (HOG) features **(Dalal, N. et al., 2005)** and **(Chavez-Garcia, R. O., et al., 2016)**, Local Binary Pattern (LBP) or Canny edge detection **(Hussain, B. and Hathal, M., 2020)**. Hand-crafted methods are not very strong since complex features are difficult to hand-craft. In DL techniques, the network extract features provide a higher level of abstraction.

After that comes the classifier such as Support Vector Machine (SVM) **(Sun, W., et al., 2018)**, decision tree **(Sun, W, et al., 2018)**, hybrid neural networks **(Al-Araji, A. and Al-Zangana, S., 2019)** or deep network **(Wang, X., et al., 2009) (Neagoe, V.E., et al., 2012)** to determine the type of the object (e.g. person, airplane) in the image or video frames. The progress of DL techniques and their development and success in detection and recognition tasks made it the subject of this research. Convolutional Neural Networks (CNNs) techniques are used to recognize objects as faces, handwriting, and vehicles. Being able to extract information directly from raw-pixel content makes it a good choice for unstructured data (like images and voices). CNNs extract information by performing various operations, typically combining filtering, pooling, and non-linear activation. The main advantage of using CNNs for feature extraction compared to hand-crafted methods is that CNNs learn features from images directly without explicit programming **(Tomè, D., et al., 2016)**. Detection and recognition DL approaches can be either two stages (region proposal approach) detectors or a single stage (non-region proposal approach) detectors. The single stage detectors are simpler to train with higher computational efficiency **(Ren, J., et al., 2017)**.

Drones detection is considered a great challenge. Drones come in different shapes and sizes. It also moves at different velocities, and there is always the possibility of complicated backgrounds. Drones are not included in common images datasets which required collecting special datasets to estimate the performance of DL detection and recognition algorithms.

This paper contains:

1. Briefing Faster RCNN, SSD, and YOLO and comparing previous studies results on common datasets to identify the most effective training parameters on the performance of the algorithm.
2. Comparing Faster RCNN, SSD, and YOLO performance for drone detection using evaluation data from our environment.

The rest of this paper is organized as follows: discussing common datasets and comparing parameters on section II. Section III briefs Faster RCNN, YOLO, and SSD. Section IV studies previous researches results to find the most effective training parameters. The training parameters and the practical algorithms results are used in section V then the conclusion was given by summing up the results.

II. DATASETS AND COMPARING PARAMETERS:

Datasets are groups of annotated images. Annotated means with object position and class. Each dataset contains a specific number of classes. Class images are divided into testing and training images groups. **Table 1.** shows the most common images datasets.

Table 1. Most common images datasets.

Dataset	Ref	Classes	Train/Val	Annotated
VOC 2007	**(Mark E., et al., 2009)**	20	9,963	**24,640**
VOC 2012	**(Mark E., et al., 2012)**	20	11,530	**27,450**
COCO	**(Tsung-Yi L., et al., 2015)**	80	330,000	**200,000**
ImageNet	**(Olga R., et al., 2015)**	27	14,197,122	**1,034,908**

VOC (**Mark E., et al., 2009**) represents Pascal Visible Objects Classes dataset, and COCO represents Microsoft Common Objects in Context dataset. The class column shows the number of classes. Train/Val column contains the number of images in the database. Those images are grouped into train sets and evaluation sets. Annotated column contains the number of annotated images in the database. Some databases like COCO and ImageNet contain some unannotated images which can be used for testing. It is also possible for any image to contain more than one class. Usually, mean average precession (mAP) is used to compare algorithms accuracy. The mAP represents the average area under the Precession/Recall curve. Precession can be calculated from: **(Mark E., et al., 2009)**

$$\text{precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \qquad (1)$$

True Positive is the number of correct class detections for the algorithm in the test set, while False Positive is the number of incorrect class detections for the algorithm in the test set.
Precession represents the algorithm recognition capability. It takes maximum value when there aren't any incorrect detections (False Positive = 0)
Recall can be calculated from **(Mark E., et al., 2009)**

$$\text{recall} = \frac{\text{True positive}}{\text{True positive} + \text{False negative}} \qquad (2)$$

False Negative is the number of missing class detections for the algorithm in the test set.
The recall represents the algorithm detection capability. It takes its maximum value when there aren't any missing targets (False Negative = 0)
We can combine both precession and recall in "F1 score", F1 score is the harmonic mean of precision and recall. F1 score can be calculated by

$$\text{F1} = 2 * \frac{\text{precesion} * \text{recall}}{\text{precision} + \text{recall}} \qquad (3)$$

Algorithms speed is estimated using a number of processed frames per second (fps) or the required time to process one frame

$$\text{fps} = \frac{1}{\text{frame processing time in seconds}} \qquad (4)$$

III. DETECTION ALGORITHMS:

Detection and recognition systems are considered to be of the most important image processing and computer vision systems. Their applications vary from smart supervision systems to people detection and recognition systems. Among many detection and recognition systems mentioned in the literals, those depending on DNN (deep neural networks) proved their superiority. SSD **(Wei Liu, et al., 2016)**, YOLO **(Joseph R., et al., 2016)** and Faster RCNN **(Shaoqing R., et al., 2015)** are considered the most important algorithms for detection and recognition. They use DNN and work in the real time (YOLO and SSD) or close to real time (Faster RCNN). We will go through each with some details.

A. Faster RCNN

Faster RCNN was an enhanced version of Fast RCNN **(Girshick, R., et al., 2015)** which was built on R-CNN. R-CNN uses selective search **(Uijlings, J.R., et al., 2013)** (a region proposal technique) to generate 2000 region proposals. These proposals are fed into a CNN for feature extraction and then SVM for classification. But classifying the 2000 region takes a long time, about 47s per image. To overcome the time issue, Ross Girshick (the author who proposed R-CNN) introduced Fast R-CNN. He suggested changing the process order to calculate features map first, then make proposals based on it. This was faster than the R-CNN technique as the convolution is computed once per image rather than for 2000 region proposals. Fast R-CNN was found to be approximately 9 times faster than R-CNN.

Experimental results showed that Fast R-CNN had 66.9% mAP while R-CNN of 66.0% on the PASCAL VOC2007 dataset. Training time dropped to 9.5 hours as compared to R-CNN with 84h. Fast RCNN with truncated SVD (0.32s) was 213x faster than R-CNN (47s). Used Nvidia k40 GPU on those experiments, which demonstrated that Fast RCNN did accolated object detection process

Studies showed that selective search was the bottleneck of the classification process. Therefore, Faster RCNN replaced selective search with RPN (Region Proposal Network). Similar to Fast RCNN, images are fed into a CNN to generate feature maps. But instead of using selective search, this sub-network learns region proposals using DL algorithms. As a result, RPN has an mAP of 75.9%, which is approximately 10.2% better than selective search results on the VOC (Visible Objects Classes) 2012 dataset.

RPN is a CNN functioning by predicting object bounds (region proposal) and scores for those bounds simultaneously. This design works at near real-time frame rates improving the quality and accuracy for general DL-based object detection.

B. YOLO

YOLO used a single neural network for both detection and position estimation. It uses features from the entire image to predict objects' positions. YOLO also predicts all bounding boxes for all classes simultaneously. The used network reasons globally about all objects in the image. Its design enables end-to-end training, real time speed, and high average precision. **Fig 1** shows the YOLO detection model.

YOLO authors proposed special networks structures to get the best of their proposed algorithm. YOLO suffers from poor multi-target detection; its model has difficulty with small objects that appear in groups, such as flocks of birds. It also suffers poor generalization performance for objects in new or unusual

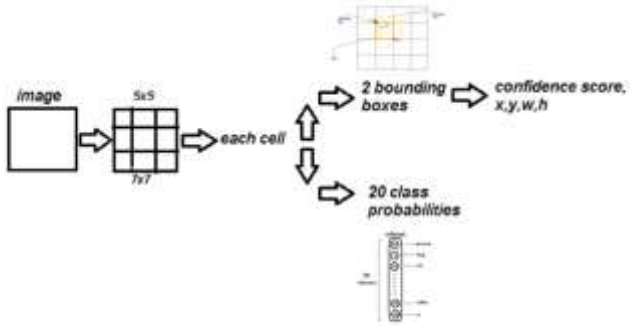

Figure 1. YOLO detection model.

aspect ratios or configurations.

YOLO network runs at 45 frames per second with no batch processing on a Titan x GPU as compared to Fast RCNN at 0.5fps and Faster RCNN at 7FPS

Training and testing on PASCAL VOC dataset. YOLO archived 63.4% mAP with 45 fps as compared to Fast RCNN (70.0%, 0.5 fps) while faster RCNN (73.2%, 7fps).

YOLO Authors took few design decisions to improve the original YOLO v1 calling it YOLO v2 (**Joseph R., et al., 2016**). The changes are summed up here.

a) Batch Normalization (BN): They added BN layer ahead of each convolutional layer which accelerated the network to get convergence and helped regularize the model. This increased the mAP by 2%.

b) High Resolution Classifier: They added a fine-tuning to the classification network at 448x448 for 10 epochs on ImageNet. This enhanced the mAP by 4%

c) Convolutional with anchor boxes: They used anchor boxes In V2 (similar to Faster RCNN) and predicted the class and object for each anchor box. This enhanced the recall value by 7%, while mAP was decreased by 0.3%.

d) Predicting the size and aspect ratio of anchor boxes using dimension clusters: They used K-means clustering on the training set bounding boxes. Using dimension clusters improves bounding box center prediction by almost 5% over the above version with anchor boxes.

e) Fine-Grained Features: V2 used both higher-resolution features and low-resolution features by stacking adjacent features from different channels, which improved the performance by 1%.

f) Multi-Scale Training. To improve network performance on different images sizes, the network changes images dimensions from {320, 352, …, 608} every 10 batches. This enhanced the network's ability to predict detections at different resolutions.

The authors also proposed a new classification network called Darknet 19. Using their new network at high resolution detection, YOLO V2 achieves 78.6% mAP and 40fps as compared to YOLO V1 with 63.4 mAP and 45fps on VOC 2007.

Authors further improved V2 to YOLO V3 **(Redmon and Farhadi, 2018)**. They used multi-label classification in V3 to match more complex datasets containing many overlapping labels (for example, the same object can be labeled 'Lamborghini' and 'car'). It also uses three feature maps with different scales to predict the bounding box (similar to SDD). The last convolutional layer predicts a 3d tensor encoding class predictions, objects, and bounding box. The authors also propose a new features extraction network called Darknet 53, inspired by ResNet. V3 achieved 57.9% mAP compared to V2 with 48.1 mAP on the COCO dataset.

C. SSD

The SSD is based on a feed-forward convolutional network (FFCN). It generates a fixed-size collection of bounding boxes and scores for the presence of object class in those boxes. Then it passes the boxes to a non-maximum suppression step to get the final detection. The first few layers in the network are based on a high-quality image classification architecture. We are going to call this 'base network'. The auxiliary structure is added to the network to generate detection with the following key features:

a) Multi-scale feature maps for detection, adding convolutional feature layers with progressively decrease in size to the end of the based network allowed predictions of detection at multiple scales. This generated multiple features maps with different scales

b) Convolutional predictors for detection. The added feature layers can predict a fixed number of detection (bounding boxes) using a group of convolutional filters. The bounding boxes offset output values are relative to the default box position, and the box position is relative to its feature map location.

c) Default boxes and aspect ratios set a default group of bounding boxes for each feature map cell for all different sized feature maps. Each box has a fixed position relative to its corresponding cell. Each feature map cell predicts the offsets relative to the default box shapes in the cell and the per-class scores that indicated the presence of a class instance in each of those boxes.

It is hard for SSD to classify small objects, especially if trained on a small dataset such as PASCAL VOC. To overcome this, they used a data augmentation strategy. It includes random crops (zoom in and zoom out) and image flipping. This operation increased the dataset size, which required doubling the training iterations.

IV. RELATED WORK

Table 2. represents previous studies' results for the three algorithms: Faster RCNN, YOLO, and SSD. The first column contains the algorithm name; the second column contains the reference number. The detection model is the used network structure. Train data and Test data are the used train set and test set. FPS is the number of processed frames per second; mAP is the mean average precession mentioned in section 2. The (-) was used for values that were not mentioned in the reference. VOC07 and VOC12 are Pascal VOC 2007 and Pascal VOC 2012. VOC07++12 means using both train and evaluation sets in Pascal VOC 2007 with the train set of Pascal VOC 2012 for training. The number after algorithm name (300, 512) represents the dimensions of network input, (*) after algorithm name mean using data augmentation (generating more data from the original by applying some image transformations like rotating, scaling, adding noise, and other methods) for training data, the increase in training data usually requires more training steps too.

The most significant training parameters based on table 2 are:

1. Training set: by comparing rows 9 and 10, the effect of the training set on the algorithm performance can be noticed. Using larger training set improved Faster RCNN performance by 10%.
2. Test set: by comparing rows 12 and 14, the effect of the test set on algorithms results can be noticed. Changing the test set affected YOLO performance by 5%.
3. Network structure: by comparing rows 12 and 13, the effect of network structure on algorithms results can be noticed. Using VGG16 improved YOLO performance by 3%. Also, the effect of network structure on algorithm processing time can be noticed by comparing rows 3 and 4. Faster RCNN speed increased by 11 frames per second when using ZF compared to its speed when using VGG16.
4. Network input dimensions: by comparing rows 18 and 22, the effect of network input dimensions on network performance can be noticed. Increasing SSD input dimensions from 300x300 to 512x512 affected mAP by 2%. The effect of network input dimensions on processing time can be seen by comparing rows 20 and 24. The number of processed frames per second decreased by 27 frames when increasing SSD input dimensions from 300x300 to 512x512
5. Number of training steps: by comparing rows 18 and 21 and checking their reference, increasing train set size using data augmentation required doubling the number of training steps can be noticed. SSD gained 3% on using data augmentation and doubling training steps.

TABLE 2. RELATED WORK RESULTS.

#	Algorithm	Ref	Detection Model	Train data	Test Data	FPS	mAP %
1	Faster RCNN	(Shaoqing R., et al., 2015)	VGG16	VOC 07++12 + COCO	VOC12	-	75.9
2	Faster RCNN	(Shaoqing R., et al., 2015)	VGG16	VOC 07 +12 + COCO	VOC07	-	78.8
3	Faster RCNN	(Joseph R., et al., 2016)	VGG16	VOC 07+12	VOC07	7	73.2
4	Faster RCNN	(Joseph R., et al., 2016)	ZF	VOC 07+12	VOC07	18	62.1
5	Faster RCNN	(Wei Liu, et al., 2016)	-	VOC 07++12	VOC12	-	70.4
6	Faster RCNN	(Wei Liu, et al., 2016)	-	VOC 07++12 + COCO	VOC12	-	75.9
7	Faster RCNN	(Dai, J., et al., 2016)	Resnet101	VOC 07+12	VOC12	2	76.4
8	Faster RCNN	(Dai, J., et al., 2016)	Resnet101	VOC 07+12 + COCO	VOC12	0.3	85.6
9	Faster RCNN	(Dai, J., et al., 2016)	Resnet101	VOC 07++12	VOC12	2	73.8
10	Faster RCNN	(Dai, J., et al., 2016)	Resnet101	VOC 07++12 + COCO	VOC12	0.3	83.8
11	Faster RCNN	(Jonathan, H.., et al., 2017)	Resnet101	ImageNet	ImageNet	-	32
12	YOLO1	(Joseph R., et al., 2016)	YOLO	VOC 07+12	VOC07	45	63.4
13	YOLO1	(Joseph R., et al., 2016)	VGG16	VOC 07+12	VOC07	21	66.4
14	YOLO1	(Joseph R., et al., 2016)	YOLO	VOC 07+12	VOC12	-	57.9
15	YOLO2	(Joseph R., et al., 2016)	Darknet19	VOC 07+12	VOC07	40	78.6
16	YOLO2	(Joseph R., et al., 2016)	Darknet19	VOC 07++12	VOC12	-	73.4
17	YOLO3	(Redmon and Farhadi, 2018)	Darknet53	COCO	COCO	19	57.9
18	SSD300	(Wei Liu, et al., 2016)	VGG16	VOC 07++12	VOC12	-	72.4
19	SSD300	(Wei Liu, et al., 2016)	VGG16	VOC 07++12 + COCO	VOC12	-	77.5
20	SSD300	(Wei Liu, et al., 2016)	VGG16	VOC 07+12	VOC07	46	74.3
21	SSD300*	(Wei Liu, et al., 2016)	VGG16	VOC 07++12	VOC12	-	75.8
22	SSD512	(Wei Liu, et al., 2016)	VGG16	VOC 07++12	VOC12	-	74.9
23	SSD512	(Wei Liu, et al., 2016)	VGG16	VOC 07++12 + COCO	VOC12	-	80
24	SSD512	(Wei Liu, et al., 2016)	VGG16	VOC 07+12	VOC07	19	76.8
25	SSD512*	(Wei Liu, et al., 2016)	VGG16	VOC 07++12	VOC12	-	78.5
26	SSD	(Jonathan, H., et al., 2017)	Inception V2	ImageNet	ImageNet	-	22

V. COMPARING ALGORITHMS RESULTS

Fig. 2. represents the working model. In order to find the best algorithm among YOLO, SSD, and Faster RCNN for drone detection and recognition, an accurate comparison between them was made. Based on section 4 results, the training parameters were choosen as mentioned in **Table 3**.

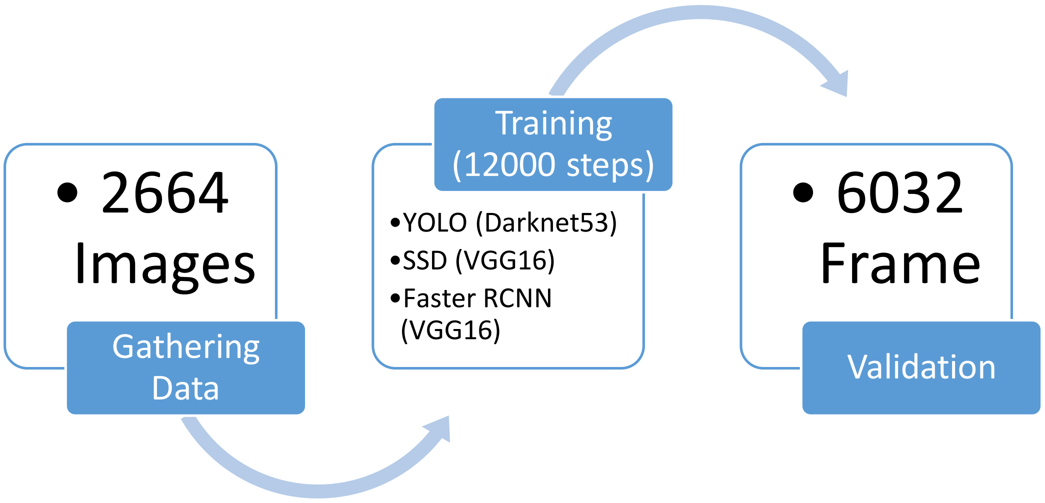

Figure. 2. Training model.

TABLE 3. TRAINING PARAMETERS

Train set	**2664**
Test Set	**6032**
Detection Model	**VGG16, Darknet53**
Training steps	**12000**

2664 images of different drones models were gathered for training. The transfer learning from the VOC model was used. VOC initial weights speeded up the training process. Those initial weights were extracted from the network trained on VOC07+12; this allowed the network to recognize VOC dataset classes in addition to drones. Each of the algorithms was trained for 12000 steps (Training steps in **Table III**).

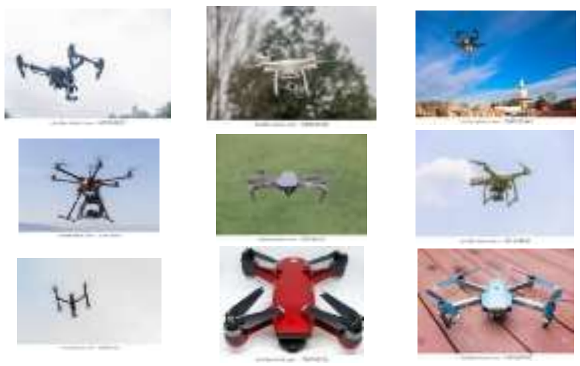

Figure. 3. Few images from the training dataset.

Fig. 3 shows few images of the used training set. Training the algorithms for different drones' models increase its generalization ability to detect and recognize any drone model. To estimate the algorithm performance, videos with a total frames count of 6032 were used (Test set in **Table 3**), 4752 frames containing drones. Those videos were taken in our environment with many different backgrounds such as clear sky, cloudy sky, trees, and mountains.

TABLE 4. EXPERIMENTAL RESULTS.

Method	FPS	mAP%
Faster 1000	7.69	**6.05**
Faster 544	16.10	**1.78**
Faster 300	27.18	**0.78**
SSD 300	27.93	**0.84**
YOLO 832	45.10	**1.74**
YOLO 312	208.48	**0.71**
YOLO 632	72.67	**1.13**
YOLO 544	87.22	**1.19**

Comparing Faster RCNN, YOLOv3 and SSD for drone detection

YOLO v3 paper recommended using their network structure, DarkNet53 **(Redmon and Farhadi, 2018)**, which Resnet101 inspired. Faster RCNN and SSD papers didn't have recommendations, so the author of this paper chose VGG16 network architecture (Detection model in **Table 4**). According to section 4 results, the network architecture effect is less than the training and testing dataset effect in general. We used a PC with those specifications for training and testing:

- Processor: Intel® Core™i7-7700 CPU @3.60GHz
- Ram: 8GB
- VGA: NVIDIA GeForce GTX 1060 6GB

All testing frames are 1280x720. Each frame has a slightly different processing time; therefore, the average processing time for all frames was calculated. The three algorithms with multiple input dimensions were tested. **Fig.4** shows some of the detection algorithms results on the test set, while **Table 4.** details them.

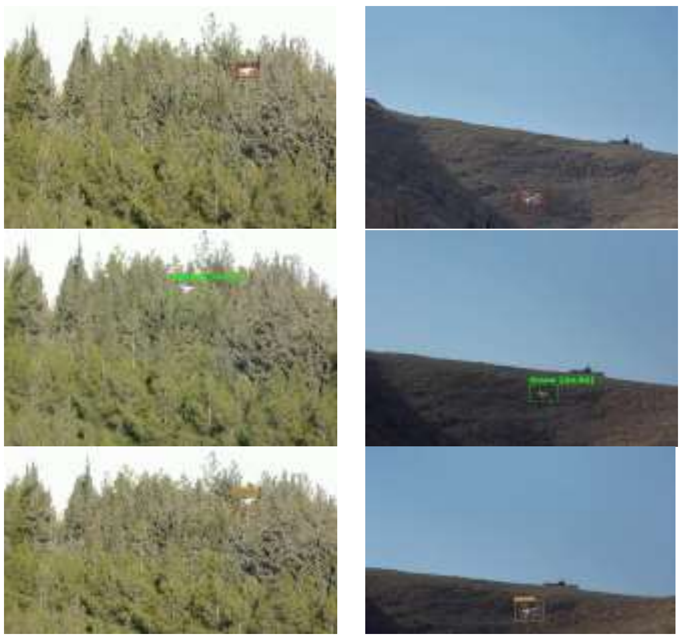

Figure.4 Detection algorithms results, red boxes for Faster RCNN, green boxes for YOLO, and orange boxes for SSD.

The number after the algorithm name represents the network input dimensions. Input images are scaled to fit network input dimensions. The increase in speed can be noticed when using networks with small input dimensions. Practical results are showing significant improvement in mAP when using network input dimensions close to the image dimensions. Rows 1 and 5 prove Faster RCNN and YOLO superiority with large input dimensions compared to small input dimensions. Other tests show the impracticality of using networks with large input dimensions when process small images (image dimensions are smaller than network input dimensions).
To get a better idea about the detection and recognition, data on **Table 5.** was checked. TP (True Positive) represents the number of true positive cases, FP (False Positive) represents the number of false positive cases, FN (False Negative) represents the number of false negative cases, and F1 represents "F1 score".

TABLE 5. EXPERIMENTAL RESULTS RAW DATA

#	Method	TP	FP	FN	F1
1	Faster 1000	1779	0	2973	**0.54**
2	Faster 544	502	0	4250	**0.19**
3	Faster 300	88	0	4664	**0.03**
4	SSD 300	662	677	4090	**0.217**
5	YOLO 832	562	0	4190	**0.211**
6	YOLO 312	137	0	4615	**0.05**
7	YOLO 632	289	0	4463	**0.07**
8	YOLO 544	203	0	4549	**0.08**

Equation 1 in section 2 shows that FP directly affects algorithm recognition performance. According to **Table 4**, both Faster RCNN and YOLO have high target recognition ability (FP=0). On the other hand, SSD suffers a higher number of false detections, which means low recognition performance.
Equation 2 in section 2 shows that FN directly affects algorithm detection performance. According to **Table 4**, SSD300 has good detection performance compared to YOLO312 and Faster RCNN 300

$$FN(SSD)<FN(YOLO)<FN(Faster)$$

VI. CONCLUSION

Choosing the 'best algorithm' depends on the used application and system requirement. Detecting drones requires high accuracy, long range, and low false positives. According to **Tables 3** and **4**, Faster RCNN is the best algorithm for drone detection, but the long frame processing time prevents using it in real time applications without any improvements. On the other hand, YOLO is capable of working directly in real time. But SSD suffers a high false positive ratio making it not suitable for this application.
Drone detection is a great challenge due to its different shapes and complex backgrounds; this study showed that Faster RCNN still has an edge on accuracy and detection ability over other algorithms but suffers slow speed. We can increase the speed by using a smaller network input size, but this will affect the accuracy.

REFERENCES

- Sun, W.; Zhu, S.; Ju, X.; Wang, D., 2018. Deep learning based pedestrian detection. In Proceedings of the Chinese Control And Decision Conference (CCDC), Shenyang, China, pp. 1007–1011.
- Dalal, N.; Triggs, B., 2005. Histograms of Oriented Gradients for Human Detection, Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 886–893.
- Chavez-Garcia, R.O.; Aycard, O. 2016. Multiple Sensor Fusion and Classification for Moving Object Detection and Tracking. IEEE Trans, pp. 525–534.
- Wang, X.; Han, T.X.; Yan, S. 2009. An HOG-LBP human detector with partial occlusion handling. IEEE 12th International Conference on Computer Vision, Kyoto, Japan, pp. 32–39.
- Neagoe, V.E.; Ciotec, A.D.; Ba˘rar, A.P. 2012. A Concurrent Neural Network Approach to Pedestrian Detection in Thermal Imagery. The 9th International Conference on Communications (COMM), pp. 133–136.
- Brazil, G.; Yin, X.; Liu, X. 2017. Illuminating Pedestrians via Simultaneous Detection and Segmentation. The IEEE International Conference on Computer Vision, Venice, Italy, pp. 4960–4969.
- Tomè, D.; Monti, F.; Baroffio, L.; Bondi, L.; Tagliasacchi, M.; Tubaro, S. 2016. Deep Convolutional Neural Networks for pedestrian detection. Signal Process. Image Communication. pp. 482–489.
- Ren, J.; Chen, X.; Liu, J.; Sun, W.; Pang, J.; Yan, Q.; Tai, Y.W.; Xu, L. 2017. Accurate Single Stage Detector Using Recurrent Rolling Convolution. The Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu.
- Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg. 2016. Single Shot MultiBox Detector ECCV.
- Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection, Computer Vision and Pattern Recognition (https://arxiv.org/abs/1506.02640)
- Shaoqing Ren∗ Kaiming He Ross Girshick Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, NIPS.
- Girshick, R. Fast R-CNN. 2015. the IEEE International Conference on Computer Vision(ICCV), pp. 7–13.
- Uijlings, J.R.; Van De Sande, K.E.; Gevers, T.; Smeulders, A.W. 2013. Selective search for object recognition. Int. J. Comput. pp. 154–171.
- Dai, J.; Li, Y.; He, K.; Sun, J., 2016, R-FCN: Object Detection via Region-based Fully Convolutional Networks. The IEEE Conference on Advances in Neural Information Processing, Barcelona, Spain, pp. 379–387.
- Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, Kevin Murphy. 2017, Speed/accuracy trade-offs for modern convolutional object detectors. CVPR.

- Joseph Redmon, Ali Farhadi. 2016, YOLO9000: Better, Faster, Stronger; Computer Vision and Pattern Recognition; Computer Vision and Pattern Recognition (https://arxiv.org/abs/1612.08242)
- J. Redmon and A. Farhadi, 2018, "Yolov3: An incremental improvement," CoRR. (https://arxiv.org/pdf/1804.02767.pdf)
- Mark Everingham·Luc Van Gool·Christopher K. I. Williams·John Winn·Andrew Zisserman; 2009, The PASCALVisual Object Classes (VOC) Challenge; Int J Comput Vis.
- Mark Everingham, John Winn; 2012, The PASCAL Visual Object Classes Challenge (VOC2012) Development Kit.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Doll´ar, 2015, Microsoft COCO: Common Objects in Context.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, Li Fei-Fei, 2015, ImageNet Large Scale Visual Recognition Challenge.
- Hussain, B. and Hathal, M., 2020. Development of Iraqi License Plate Recognition System based on Canny Edge Detection Method, *Journal of Engineering*, 26(7), pp. 115-126. DOI: 10.31026/j.eng.2020.07.08.
- Al-Araji, A. and Al-Zangana, S., 2019. Design of New Hybrid Neural Structure for Modeling and Controlling Nonlinear Systems, *Journal of Engineering*, 25(2), pp. 116-135. DOI: 10.31026/j.eng.2019.02.08.

ABBREVIATIONS AND SYMBOLS

DL	Deep Learning
CNN	Convolutional Neural Networks
SSD	Single Shot Detector
YOLO	You Only Look Once
ROI	Region of Interest
BS	Background Subtraction
LBP	Local Binary Pattern
SVM	Support Vector Machine
VOC	Visible Objects Classes
COCO	Common Objects in Context
mAP	Mean Average Precision
FPS	Frames Per Second
DNN	Deep Neural Network
RPN	Regional Proposal Network
BN	Batch Normalization
TP	True Positive
FP	False Positive
FN	False Negative
HOG	Histogram of Oriented Gradients

Drones dataset link: (YOLO style annotations)
https://drive.google.com/file/d/1ppvIMu1R25Mvrpl14HJxAIclcADjDQ-N/view?usp=sharing

XML style annotations: (for Faster RCNN and SSD training)
https://drive.google.com/file/d/1-h8usgcd_Q2aKZg9SiDDyHhQfK_SJDUJ/view?usp=sharing

Test Videos
https://drive.google.com/drive/folders/15DDR4IkUfh9EtQ5md6RcnwtmMFCS2nLx?usp=sharing

Results Videos
https://drive.google.com/drive/folders/1obTBCSBkTNL8MTBduDoV3cmsFnD51XWh?usp=sharing