Dynamic Generative R-CNN

Rasha Saffarini[1], Faisal Khamayseh[2], Yousef Daraghmeh[3], Derar Elyan[4], and Muath Sabha*[5]

[1]PhD candidate in IT engineering, Arab American University
[2]Department of Computer Science,Palestine Polytechnic University
[3]Department of Computer System Engineering,Palestine Technical University
[4]Department of Applied Computing,Palestine Technical University
[5]Department of Multimedia Technology,Arab American University

Abstract

Different techniques have been developed for object detection and recognition. These techniques can be divided into single-shot and two-shot methods. The single-shot method processes an entire image, whereas the two-shot method divides the image into regions and analyzes them separately. However, different versions of these techniques produce limited results in terms of accuracy and speed or both. Therefore, this study proposes a novel model called Dynamic Generative R-CNN (DGR-CNN) that reduces the number of proposed regions using a dynamic programming model that applies the graph similarity method over graph-based image segmentation. Additionally, the proposed model employs DCGAN to improve detection performance. DGR-CNN reduces the overall detection and classification time and enhances the detection accuracy. The PASCAL dataset, specifically VOC2007, was utilized to evaluate the model. The results showed that DGR-CNN significantly reduces the number of candidate regions compared to the selective search algorithm employed in R-CNN and fast R-CNN. Although fast R-CNN utilizes 2000 regions and faster R-CNN utilizes 300 regions, DGR-CNN reduces the number of regions to approximately 130. The mean average precision of the proposed method was 75.1%, while fast and faster R-CNN scored 66.9% and 69.9%, respectively. This increase in accuracy is achieved without comprising the speed significantly compared with faster R-CNN.

Keywords: Object detection, computer vision, deep learning, image processing, Fast RCNN.

1 INTRODUCTION

Computer vision uses computational models to understand digital visual data, with many applications such as attendance monitoring, aerial vehicles, and medical imaging (1) (2). Object detection is a critical process in computer vision that identifies objects in images regardless of location and conditions (3) (4). Two types of object detectors are traditional and deep learning-based, with the latter using more robust and representative features(5). There are two categories of deep learning models for object detection: one-stage and two-stage object detectors. One-stage detectors, like YOLO (6) and SSD (7), are suitable for real-time applications (8) (9), while two-stage detectors divide the process into region proposal and feature extraction and classification stages (8). In the region proposal stage, windows are generated to detect objects, and only some windows are chosen for further processing

due to computational limitations. Objects are detected and recognized using meaningful information (features) extracted from proposed regions. SIFT (10), HOG (11) are some examples of good feature extractors, and SVM (12) is an example of a good classifier. There are many two-stage detectors such as R-CNN (13), Fast R-CNN (14), Faster R-CNN (15), SPP-net (16), mask R-CNN (17), and cascade R-CNN (18).

Object detection models such as R-CNN, Fast, Faster, Mask, and cascade R-CNN still face certain limitations that need to be addressed. These limitations include the high computational demand and slow inference speed which arise due to the two-stage architecture and the high number of proposed regions. The large number of region proposals leads to a requirement for colossal memory to store images and their features. This, in turn, causes difficulties in achieving the optimal balance between speed and accuracy. Lastly, previous models have difficulty in detecting small objects and objects that are positioned close to each other. (17) (13) (18).

This paper presents a novel model called Dynamic Generative R-CNN (DGR-CNN) that aims to improve the performance of fast and faster R-CNN models by reducing the number of region proposals and improving these regions. The proposed model replaces the greedy selective search algorithm with a combination of dynamic programming, graph similarity, and graph segmentation algorithms to decrease the number of region proposals. After that, DCGAN is implemented to enhance these regions and boost detection accuracy. DGR-CNN offers several advantages, including lower memory requirements, higher detection performance, and a high speed-accuracy trade-off.

DGR-CNN was trained and tested on the PASCAL VOC2007 dataset. The results indicated that compared to the selective search algorithm employed in R-CNN and fast R-CNN, DGR-CNN substantially decreases the number of candidate regions. While fast R-CNN employs 2000 regions and faster R-CNN employs 300 regions, DGR-CNN decreases the region number to around 130, knowing that mask and cascade R-CNN are slower than fast R-CNN. The mean average precision of the proposed approach was 75. 1%, while the fast and faster R-CNN scored 66.9% and 69.9%, respectively. This increase in accuracy is achieved without significantly compromising the speed compared to faster R-CNN.

We propose a new model to improve the performance of the previous two-shot detection algorithms in terms of speed and accuracy. The proposed model has several contributions:

1. Lower memory requirements.
2. Higher detection speed than the fast and faster R-CNN.
3. Higher detection performance (higher mean average precision MAP) than fast and faster R-CNN.
4. Higher chance of detecting and classifying small objects.
5. High speed-Accuracy trade-off.

2 LITERATURE REVIEW

Much research has been conducted in various fields for object detection using different machine-learning techniques. Recently, deep learning techniques such as CNN, RCNN, fast, faster, mask, cascade RCNN, and others have become the most commonly used technologies (19).

The Region-based Convolutional Neural Network (R-CNN) framework was proposed in (13). The goals of R-CNN were to improve the process of semantic segmentation and detection. The earlier

models of object detection, which proposed regions and identified objects found in these regions using deep learning techniques, have been significantly improved by the R-CNN framework. The model utilizes the selective search greedy technique to provide region proposals. Then a convolutional neural network (CNN) that has been trained is used to extract features from these regions. After that, the location of the bounding box is clarified, and the objects are classified from the extracted feature utilizing support vector machines (SVMs). The R-CNN system demonstrates its efficacy in precisely identifying objects in images by achieving good results on benchmark datasets. The average mean precision of R-CNN on PASCAL VOC2007 is 58.5%.

The R-CNN (Region-based Convolutional Neural Network) was a powerful model in object detection. However, it had some limitations. Using separate proposal and classification phases resulted in slow inference and high memory usage. R-CNN relied on an external proposal algorithm called selective search, which restricted the scalability and made it computationally inefficient. The selective search algorithm is not differentiable, making it challenging to optimize end-to-end detection and making the overall method slow (6). Another problem of the R-CNN model is the limited detection accuracy compared with newer algorithms due to the use of Pre-trained features and single-stage detection. Furthermore, the R-CNN processed each region individually without considering the contextual information from neighboring regions. As a result, this led to sub-optimal results in scenarios with overlapping objects(17) (15).

Different models were developed as an improvement to R-CNN, such as Fast R-CNN, Faster R-CNN, Mask R-CNN, Cascade R-CNN, and Meta R-CNN. Fast R-CNN, introduced by (14), significantly improved object detection by addressing computational inefficiencies present in the previous model. This enhanced method features an end-to-end trainable framework and integrates object detection and region proposal phases into a single network. Fast R-CNN shares the convolutional features across the entire image, unlike the sequential approach of extracting features separately for each region proposal in R-CNN, thus reducing computation time significantly. Additionally, it utilizes a Region of Interest (RoI) pooling layer to extract fixed-size feature maps for each region proposal, leading to faster and more efficient processing. Incorporating these enhancements, Fast R-CNN achieved better results in accuracy and speed.

Fast R-CNN with VGG16 as the backbone was tested on the PASCAL VOC2007 data set; it achieves a Mean Average Precision (MAP) of approximately 66.9%. Despite the remarkable advancement in the detection speed provided by fast R-CNN, Fast R-CNN relies on an external region proposal mechanism, which makes the overall architecture more complex, and the model is still considered slower than some of the newer models. Moreover, the use of selective search in the region proposal phase causes a computational bottleneck during training. Additionally, because of the fixed-size RoI pooling, the model may not handle overlapping instances well and may have a problem in detecting small objects. While it balances accuracy and speed, the trade-off may not be optimal for certain real-time applications(15) (17).

The research presented in (15) introduces a unique Region Proposal Network (RPN) to address the speed limitation of R-CNN object detection. By sharing convolutional features with the primary detection network. Their RPN significantly shortens the inference time and allows for almost cost-free proposals. Faster R-CNN achieves state-of-the-art accuracy on datasets like PASCAL VOC and MS COCO by combining RPN and detection into a single network, opening the door for real-time object identification applications and spurring major breakthroughs in the area. In the PASCAL VOC 2007 dataset, the Mean Average Precision (MAP) of the faster R-CNN with VGG16 as the backbone is claimed to be roughly 69.9%.

Faster R-CNN represents a significant advancement in object detection, although it is not without

flaws. Speed is still an issue because of its two-stage architecture, which requires a lot of resources and prevents real-time deployment. Proposal accuracy can vary: the fixed-size ROI pooling restricts flexibility in handling diverse object sizes, and the RPN struggles with objects at varying scales or with occlusion. Furthermore, adaptability can be difficult since Faster R-CNN is less flexible and more sensitive to new scenarios or data because it depends significantly on huge datasets and careful hyperparameter tuning. (20) (17).

The authors in (17) propose a powerful object detection model built upon Faster R-CNN called Mask R-CNN. Mask R-CNN includes a branch to predict instance segmentation masks and bounding boxes to overcome the drawback of Faster R-CNN, which offers bounding boxes. The model proves its efficacy in multiple cases by achieving state-of-the-art performance on tasks such as instance segmentation and object detection. The design of the Mask R-CNN model makes it possible to differentiate and localize items precisely, even when they overlap or are hidden. But, it is not suitable for applications that need speed detection, like real-time applications, because of its two-stage architecture. Although Mask R-CNN tried to outperform faster R-CNN in terms of detection accuracy, in some cases, faster R-CNN achieved almost the same accuracy with faster detection.

In order to overcome the problem of suboptimal performance when utilizing high-quality detectors on non-high-quality proposals, the study suggests a multistage object detection architecture. It accomplishes this by employing a series of R-CNN stages that are trained with progressively higher IoU thresholds, gradually eliminating false positives and improving predictions. This ultimately results in better accuracy relative to single-stage models by allowing the later stages to concentrate on truly difficult cases. However, the multi-stage architecture may be less appropriate for real-time applications due to its higher processing complexity.

Varying datasets, such as the MS COCO and PASCAL VOC2007 datasets with varying backbones, are used to test the Cascade R-CNN. Results showed a significant improvement in terms of detection accuracy such that the AP of Cascade R-CNN with VGG16 as a backbone reaches 79.1%. But faster R-CNN outperformed it in terms of test speed.

Another improvement to faster RCNN was proposed in (21). A multi-object detection and classification model was proposed using cross-layer fusion. The model's main architecture was a five-layer convolution of VGG16, and small convolution kernels were added to the first, third, and fifth layers. After applying pairwise fusion, a more semantic feature map was extracted and fed to the RPN (region proposal network) to generate region proposals to be entered into an ROI network for the recognition process. The proposed improvement of faster R-CNN focused on improving the accuracy of the original faster R-CNN, especially for small object detection and object localization.

In (22), an improvement in faster R-CNN called "Decoupled Faster R-CNN" was proposed for the detection of short-shot objects. In(23), researchers modified faster R-CNN for semantic segmentation for multiscale targets by obtaining not only location information but also semantic information and information fusion about objects using RGB-D images.

2.1 Research Gap

Although R-CNN, Fast R-CNN, Faster, mask, and cascade R-CNN have made significant strides in object detection, they still encounter some limitations that must be addressed.

- computations: All previous model architectures have the drawback of being computationally expensive in different scales because of their multi-stage nature.
- Speed: The R-CNN model, developed early on, experiences slow inference speeds due to its sequential region proposal and classification stages. Fast R-CNN speeds things up by shar-

ing features but still requires significant resources. Although Faster R-CNN includes the proposal stage within the network, it remains computationally demanding, especially for resource-constrained devices or real-time applications. Mask and cascade R-CNN are slower than faster R-CNN, although they are considered an improvement because they focus on accuracy, not speed.

- Small objects: Detecting small objects can be challenging for most of these models.
- Memory constraints: Although not as memory-intensive as R-CNN, Fast R-CNN, and Faster R-CNN still require significant memory during inference, which can cause problems on devices with limited memory resources.
- Accuracy-speed balance: Achieving optimal speed and accuracy is still difficult. Although faster R-CNN has brought about considerable speed enhancements compared to R-CNN, striving for even faster inference can sometimes compromise accuracy.

3 METHODOLOGY

This paper presents a novel approach called Dynamic Generative R-CNN (DGR-CNNto), improving the performance of the fast, faster, mask and cascade R-CNN algorithms for object detection by proposing a new region proposal approach using graph segmentation and graph similarity algorithms. Then DcGAN is used to enhance these regions So that the objects inside the image become known to the CNN.

In a series of object detection and classification algorithms based on the idea of a candidate region proposal, the Fast-RCNN model is superior to other network models in terms of accuracy and performance. Several modifications to the fast R-CNN algorithm are proposed in this paper to increase its applicability in real-time applications by replacing the greedy selective search algorithm (24) used in the original fast R-CNN for region proposals with a new dynamic programming algorithm.

The general object detection and recognition systems go through several steps to detect objects and recognize them, namely the proposed region proposal phase, feature extraction phase, and detection and classification phase. The performance of the original fast R-CNN model is improved in this research by developing a new region proposal method instead of the selective search algorithm.

3.1 Region proposal phase

A new algorithm is proposed to select candidate regions from the input image to be used as input to CNN for feature extraction, object detection, and object classification to improve the performance of fast R-CNN. For region proposal, the technique started by dividing the image into a set of regions/segments using a graph-based segmentation algorithm, applying the graph similarity technique to find similar regions, storing each group of similar segments in a list, and entering one graph/segment from each set into CNN for the feature extraction process.

3.2 Image segmentation

Graph–based image segmentation is an algorithm used to segment an image into different regions. Graph-based segmentation starts by representing the image as a graph G =(V, E) with vertices v ϵ V and set of edges (v_i, v_j) ϵ E with a weight w(v_i, v_j) on each edge represents the dissimilarity between the two connected vertices. For graph-based image segmentation, each vertex of V represents a pixel

of the image, each edge (v_i, v_j) represents a connection between pixel I and pixel j, and the weights on these edges is a measurement indicates the difference between the two adjacent connected pixels such as color, intensity, location or any other parameter. The segmentation S is segmenting V into different connected graphs/segments C ϵ S. The cutting process is achieved by calculating distances between each pixel and other pixels and then cutting dissimilar vertices. Finally, the whole graph is broken into several continuous graphs such that the weights of edges between nodes in the same graph are relatively low, and the weights between nodes in different graphs should be higher weights. In object detection, each resulting graph/region represents an object candidate in the image that can be studied by extracting its features using any feature extraction method (25). Graph-based segmentation results are shown in Figure 1

Figure 1. Graph-based segmentation results

3.3 Finding Similar Regions

The output of applying graph-based segmentation to the image is many graphs representing candidate objects. The problem when studying and analyzing these regions is that different regions could represent the same object or have the same content. This means that CNN might analyze the same objects several times, which will need more computations and cost more time. To overcome this issue, a new approach is proposed to minimize the number of proposal regions. This is achieved by merging overlapped regions to produce one region and finding similar regions far apart from each other.

Finding overlapped regions is the process of estimating the degree of intersection between two regions(OLA) by calculating the area of intersection area and dividing it by the area of the union as presented in equation 1:

$$OLA = \frac{intersectionArea}{((area1 + area2) - intersectionArea)} \tag{1}$$

Area 1 and Area 2 represent the areas of the overlapped regions (region 1 and region 2), and the intersection area is the area where the two regions overlap.

Finally, if the overlapping area exceeds a predefined threshold (70%), the two regions are merged and treated as one region. By applying this process, many regions will be merged, and consequently, the number of candidate graphs will be reduced considerably.

The second operation to reduce the number of proposed regions for the CNN is to find similar, not overlapped, candidate regions that exist far away from each other. Since each candidate region is represented by a graph (from graph-based segmentation), the graph similarity technique will be used to find the degree of similarity between these graphs. The Eigenvector Similarity method is used in this research to find the degree of similarity between two graphs (26).

Let G_1 and G_2 be two graphs; for eigenvector similarity, the Laplacian of the graphs is calculated as in equation 2 & 3:

$$L_1 = D_1 - A_1 \tag{2}$$

$$L_2 = D_2 - A_2 \tag{3}$$

Where A_1 and A_2, D_1 and D_2, L_1 and L_2 represent the adjacency matrix, diagonal matrix of degrees, and Laplacian of G_1 and G_2 respectively. The eigenvalues of each Laplacian are calculated, then find the smallest k such that the sum of the top k eigenvalues constitutes 90% of the sum of all the eigenvalues. In the case where the k values for both graphs are different, use the smallest one for both. After that, we calculate the similarity metric as in Equation 4:

$$sim = \sum_{i=1}^{k} (\lambda 1_i - \lambda 2_i)^2 \tag{4}$$

Which represents the sum of the squared differences between the largest k eigenvalues between the two graphs. The similarity metric is a value in the range [0, ∞]. Smaller values of the similarity metric indicate similar graphs.

3.4 Group similar regions

The graph similarity concept can be used to find similar regions of an image to reduce the number of regions to be entered into the CNN by grouping similar graphs/regions in a list and feeding only one of them to the CNN to be analyzed, which in turn reduces the time needed for overall classification or any analysis process.

Similar graphs/segments were grouped and stored in an adjacency list. The adjacency list consists of a number of nodes equal to the number of similar graph groups. Each node of the list points to a list of similar graphs as Figure 2, where each G represents a group of similar graphs, and R represents a graph/region from the image.

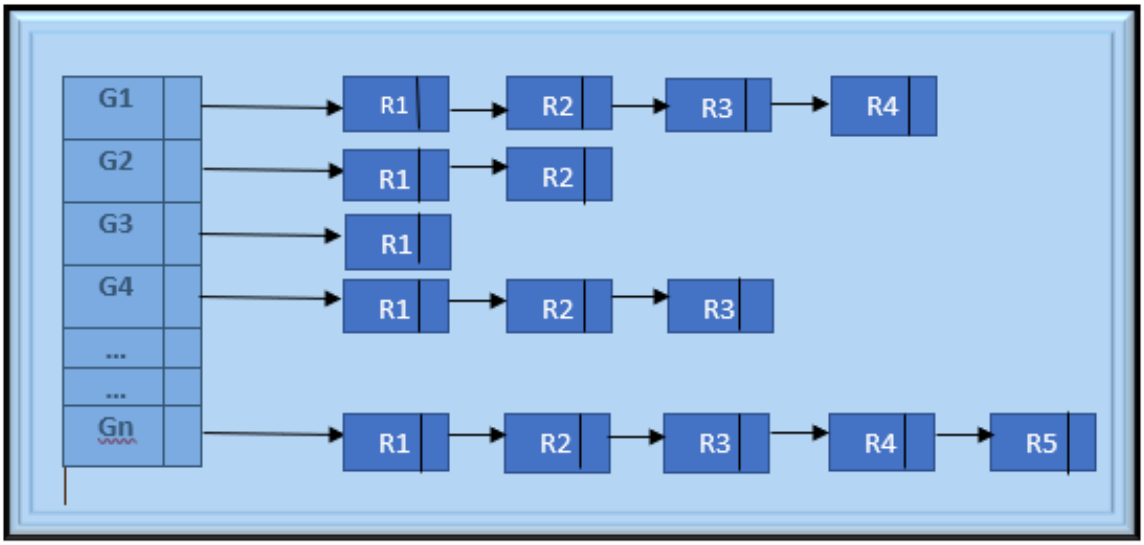


Figure 2. Adjacency list of similar graphs

3.5 Image enhancement

DCGAN model is utilized to enhance the accuracy of the proposed regions. The DCGAN model first converts the image to vector form by applying sequential CONV array filters (i.e., discriminator phase). This vector contains all the features within the image. The generated vector is used to generate a new image. This is done using the convolution transpose algorithm. This algorithm applies the CONV transpose array filter to predict the pixels within the image. This is performed for several epochs to produce a high-quality and more complete image compared with the original image (i.e., Generator phase). The DCGAN consists of two phases: the discriminator and generator phases.

Dicriminator Phase In this phase, the process starts with converting an image to the LAB color space and applying five sequential CONV array filters to extract essential features of the image. The resulting vector is then fed into an activation function to determine whether the image is real or fake. The Leaky Relu activation function is utilized to add nonlinearity to the network and allow pixels below the average intensity to build features based on significant combinations of pixels that have values greater than zero.

Batch normalization is used to standardize the inputs to each layer for each mini-batch, which stabilizes the learning process and decreases the number of training epochs needed to train deep networks. The process involves the application of the Leaky Relu activation function and the use of CONV array filters with different parameters, such as window size and stride, resulting in an image that is normalized and flattened to a vector form. Finally, the sigmoidal function is applied to decide whether the image is real or fake.

Generator Phase The generator model creates a new image from a given vector. The model consists of a fully connected layer, transpose convolution layers, batch normalization, and the leaky relu activation function. The generated image is sent to a discriminator model to determine if it is real or fake. If the discriminator returns a false value, the original vector is modified using back-propagation and entered into the generator to produce a new image. This process continues until a true value is obtained from the discriminator.

3.6 Feature extraction phase

Proposed regions from the previous stage are fed to a convolutional neural network CNN for extracting features and generating features vector. For this purpose, we used VGG16, which consists of 13 3*3 convolutional layers with different filters (64, 128, and 512), two fully connected layers with 4096 neurons each, and the output softmax FC layer with 1000 neurons, which will be used in the next phase. The architecture of the VGG16 network is shown in Figure 3.

3.7 Detection and classification phase

3.7.1 Detect and classify objects

The features vector from the previous stage is used to detect objects from the image and then classify them into different classes. This process is done using the softmax layer, which is used for the multiclass classification process by converting the output vector from the FC layers into a vector of probabilities. Each value is expected to range between 0 and 1, while the sum of all values equals 1

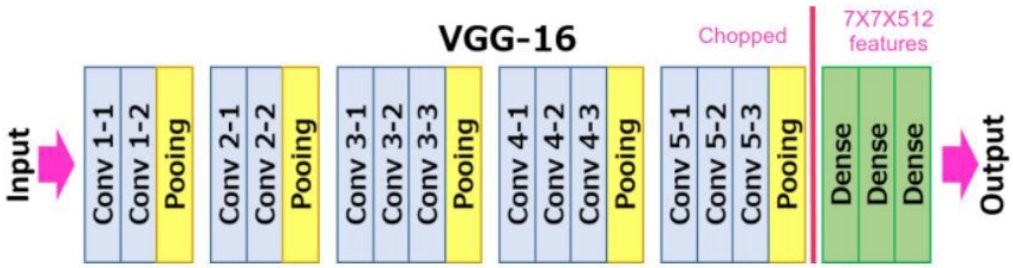


Figure 3. VGG16 architecture

(probability distribution). To increase classification accuracy and reduce the number of false predictions, we used a threshold value of the class probability as 85% to consider that it is a real object and belongs to that class.

3.7.2 Classify similar graphs

After classifying objects, the proposed model will take each object, and using a dynamic programming technique, the system returns to the adjacency list, finds the classified object in the list, and labels all connected graphs with the same class.

4 RESULTS

The overall detection and classification model DGR-CNN is tested in different conditions. Firstly, the proposed region proposal model is tested to show the considerable improvement of the detection speed over faster R-CNN, as shown later. The experiments are repeated by improving our system by adding the DcGAN model for the classification phase with some modifications on the overall model to enhance the detection accuracy and achieve a significant accuracy-speed trade-off. Experiments were conducted on the PASCAL VOC2007 dataset using the pre-trained VGG16 model, which was trained previously on the ImageNet dataset. The detection and classification model is evaluated with and without using DcGAN on different classes from the dataset, such as caw, coat, bottle, person, dog, cat, airplane, dining table, bus, chair, and others. Detection precision, recall, and F1-score for each class are calculated for the two scenarios. As depicted in tables 1, 2, and figures 5, 4 detection and classification gave better results when using DcGAN in the classification stage. Detection precision without using DcGAN ranges from 62% to 70.6% with a MAP of 66.7% and Mean accuracy MA of 66.56%, while using DcGAN improves the precision to range from 68.13% to 79.6% with a MAP of 75.104%, and MA of 76.2433%. The condition is the same for other results (Recall and F1-score).

The mean accuracy (MA), the mean average precision (MAP), the mean average recall (MAR), and the mean average F1-score of the object detection process are also calculated as shown in table 3 to be compared with previous models.

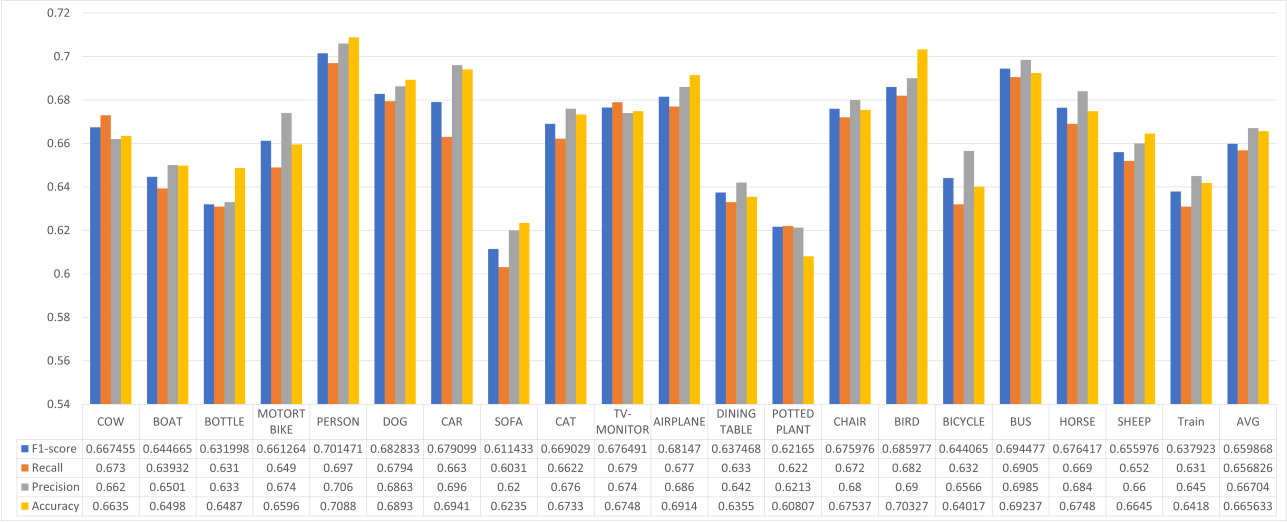
The number of proposed areas resulting from the proposed model is compared with the number of regions resulting from the selective search algorithm used in previous models. The proposed algorithm outperforms the selective search (Fast R-CNN) in detecting objects, as shown in table 4. The performed tests on various object classes, including Person, Dog, Chair, Bus, Bird, Train, and Car, show that the proposed algorithm significantly reduces the number of candidate regions proposed in

Table 1. Accuracy, precision, recall, and F1 score without using DCGAN

Class	**Accuracy**	**Precision**	**Recall**	**F1-score**
COW	0.6635	0.662	0.673	0.667454682
BOAT	0.6498	0.6501	0.63932	0.644664938
BOTTLE	0.6487	0.633	0.631	0.631998418
MOTORTBIKE	0.6596	0.674	0.649	0.661263794
PERSON	0.7088	0.706	0.697	0.701471133
DOG	0.6893	0.6863	0.6794	0.682832569
CAR	0.6941	0.696	0.663	0.679099338
SOFA	0.6235	0.62	0.6031	0.611433243
CAT	0.6733	0.676	0.6622	0.669028845
TV-MONITOR	0.6748	0.674	0.679	0.676490761
AIRPLANE	0.6914	0.686	0.677	0.681470286
DINING TABLE	0.6355	0.642	0.633	0.637468235
POTTED PLANT	0.60807	0.6213	0.622	0.621649803
CHAIR	0.67537	0.68	0.672	0.675976331
BIRD	0.70327	0.69	0.682	0.685976676
BICYCLE	0.64017	0.6566	0.632	0.644065187
BUS	0.69237	0.6985	0.6905	0.694476962
HORSE	0.6748	0.684	0.669	0.676416851
SHEEP	0.6645	0.66	0.652	0.65597561
Train	0.6418	0.645	0.631	0.637923197
AVG	0.6656325	0.66704	0.656826	0.659868448
	MA	MAP	MAR	MAF

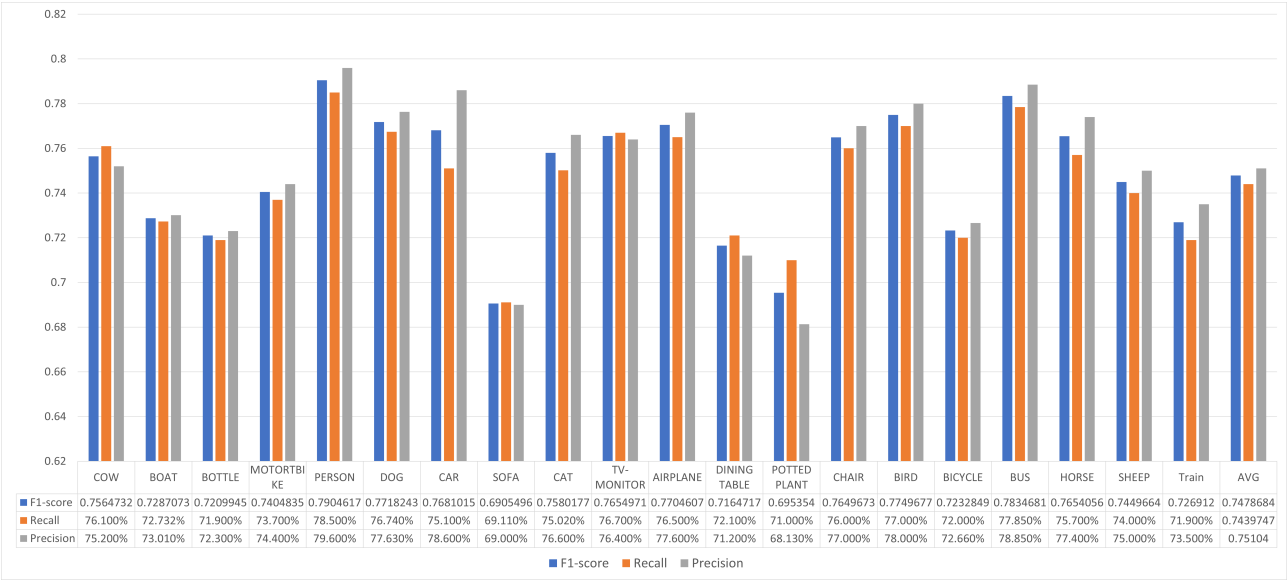
Table 2. Accuracy, precision, recall, and F1 score using DCGAN

Class	***Accuracy***	***Precision***	***Recall***	***F1-score***
COW	76.3500%	75.200%	76.100%	0.756473232
BOAT	74.9800%	73.010%	72.732%	0.728707349
BOTTLE	74.8700%	72.300%	71.900%	0.720994452
MOTORTBIKE	75.9600%	74.400%	73.700%	0.740483457
PERSON	***80.8800%***	79.600%	78.500%	0.790461733
DOG	78.9300%	77.630%	76.740%	0.771824344
CAR	79.4100%	78.600%	75.100%	0.768101496
SOFA	69.4500%	69.000%	69.110%	0.690549562
CAT	77.3300%	76.600%	75.020%	0.758017676
TV-MONITOR	77.4800%	76.400%	76.700%	0.765497061
AIRPLANE	79.1400%	77.600%	76.500%	0.77046074
DINING TABLE	71.3500%	71.200%	72.100%	0.716471738
POTTED PLANT	69.5070%	68.130%	71.000%	0.695353985
CHAIR	77.5370%	77.000%	76.000%	0.76496732
BIRD	80.3270%	78.000%	77.000%	0.774967742
BICYCLE	74.0170%	72.660%	72.000%	0.723284944
BUS	79.2370%	78.850%	77.850%	0.783468092
HORSE	77.4800%	77.400%	75.700%	0.765405617
SHEEP	76.4500%	75.000%	74.000%	0.744966443
Train	74.1800%	73.500%	71.900%	0.726911967
AVG	76.2433%	0.75104	0.743974737	0.747868448
	MA	***MAP***	***MAR***	***MAF***



	COW	BOAT	BOTTLE	MOTORT BIKE	PERSON	DOG	CAR	SOFA	CAT	TV-MONITOR	AIRPLANE	DINING TABLE	POTTED PLANT	CHAIR	BIRD	BICYCLE	BUS	HORSE	SHEEP	Train	AVG
F1-score	0.667455	0.644665	0.631998	0.661264	0.701471	0.682833	0.679099	0.611433	0.669029	0.676491	0.68147	0.637468	0.62165	0.675976	0.685977	0.644065	0.694477	0.676417	0.655976	0.637923	0.659868
Recall	0.673	0.63932	0.631	0.649	0.697	0.6794	0.663	0.6031	0.6622	0.679	0.677	0.633	0.622	0.672	0.682	0.632	0.6905	0.669	0.652	0.631	0.656826
Precision	0.662	0.6501	0.633	0.674	0.706	0.6863	0.696	0.62	0.676	0.674	0.686	0.642	0.6213	0.68	0.69	0.6566	0.6985	0.684	0.66	0.645	0.66704
Accuracy	0.6635	0.6498	0.6487	0.6596	0.7088	0.6893	0.6941	0.6235	0.6733	0.6748	0.6914	0.6355	0.60807	0.67537	0.70327	0.64017	0.69237	0.6748	0.6645	0.6418	0.665633

Figure 4. Accuracy, precision, recall, and F1-score without DcGAN



	COW	BOAT	BOTTLE	MOTORTBIKE	PERSON	DOG	CAR	SOFA	CAT	TV-MONITOR	AIRPLANE	DINING TABLE	POTTED PLANT	CHAIR	BIRD	BICYCLE	BUS	HORSE	SHEEP	Train	AVG
F1-score	0.7564732	0.7287073	0.7209945	0.7404835	0.7904617	0.7718243	0.7681015	0.6905496	0.7580177	0.7654971	0.7704607	0.7164717	0.695354	0.7649673	0.7749677	0.7232849	0.7834681	0.7654056	0.7449664	0.726912	0.7478684
Recall	76.100%	72.732%	71.900%	73.700%	78.500%	76.740%	75.100%	69.110%	75.020%	76.700%	76.500%	72.100%	71.000%	76.000%	77.000%	72.000%	77.850%	75.700%	74.000%	71.900%	0.7439747
Precision	75.200%	73.010%	72.300%	74.400%	79.600%	77.630%	78.600%	69.000%	76.600%	76.400%	77.600%	71.200%	68.130%	77.000%	78.000%	72.660%	78.850%	77.400%	75.000%	73.500%	0.75104

Figure 5. Accuracy, precision, recall, and F1-score using DcGAN

Table 3. MA, MAP, MAR, and MAF for all classes using proposed model

MA	MAP	MAR	MAF
76.2433%	775.104%	74.397%	0.7479

the first step. In fact, the proposed algorithm proposes only about 10% of the regions proposed by selective search. This reduction translates into faster computations in the subsequent phases of feature extraction, detection, and classification, enabling the object recognition process to be significantly faster than before. The proposed algorithm's performance is a testament to its superiority over the selective search (Fast R-CNN) algorithm.

From table 5, it is clear that the proposed region proposal model, even without the use of DcGAN, speeds up the detection process to be faster than faster R-CNN, which is considered one of the fastest

Table 4. Number of region candidates using our proposed method vs using selective search.

Region proposal method	Class						
	Car	Person	Dog	Bird	Train	Bus	Chair
Our proposed method	137	138	90	115	152	160	123
Selective search	1746	1793	1102	1417	1906	1725	1662

Table 5. Testig speed ad MAP with and without DcGAN

speed	*testing time/ per image*	*MAP*
RCNN	*47*	*66%*
FAST	*0.32*	*66.90%*
FASTER	*0.2*	*69.90%*
Our Model without DcGan	*0.1923*	*66.70%*
Our Model with DcGan	*0.2642*	*75.10%*

two-stage detectors. the proposed model took 0.1923s to accomplish the detection process while faster R-CNN took 0.2s. To achieve a significant speed-accuracy trade-off, DcGAN is used. Despite the detection speed becoming a little bit slower with a speed of 0.26s, the detection accuracy is significantly improved compared to R-CNN, fast, and faster R-CNN with an MAP of 75% 66%, 66.9%, and 69.9%, respectively. More details and graphical representations of these findings can be found in Figures 6 and 7.

Based on the results obtained from the comparative analysis, it is evident that the proposed model outperformed the existing models. First, in the region proposal phase, the difference in the number of region proposals between the proposed algorithm and the selective search algorithm is illustrated in Figure 8. In the first image, the number of candidate regions using a selective search was recorded at 2545, while the proposed methodology reduced it to approximately 270 regions. Similarly, in the second image, the number of candidate regions using selective search was 1984, and 248 using the proposed method. These findings highlight the effectiveness of the proposed model in generating proposed regions. After that, DcGAN is used to enhance each proposed region by generating a new image by completing what is missing from the original image or clarifying the elements in it so that the CNN can distinguish these objects; an example of enhancing one region from the first image of figure 8 is shown in figure 9. Finally, the enhanced regions are entered into the CNN to classify the objects inside it. Other examples of the use of DcGAN to enhance images are shown in Figure 10.

5 CONCLUSION

The detection of objects is a crucial task for many applications, and various techniques have been developed to accomplish it with differing degrees of accuracy and speed. Our research has introduced a novel approach to decrease the time required for object detection and recognition in images while also increasing detection accuracy. This is achieved by utilizing a dynamic programming model that utilizes the graph similarity technique over graph-based image segmentation. The effectiveness of our model was tested on a variety of classes included in the VOC2007 dataset, such as cars, people, dogs, birds, and others. The results demonstrated a significant difference between the number of

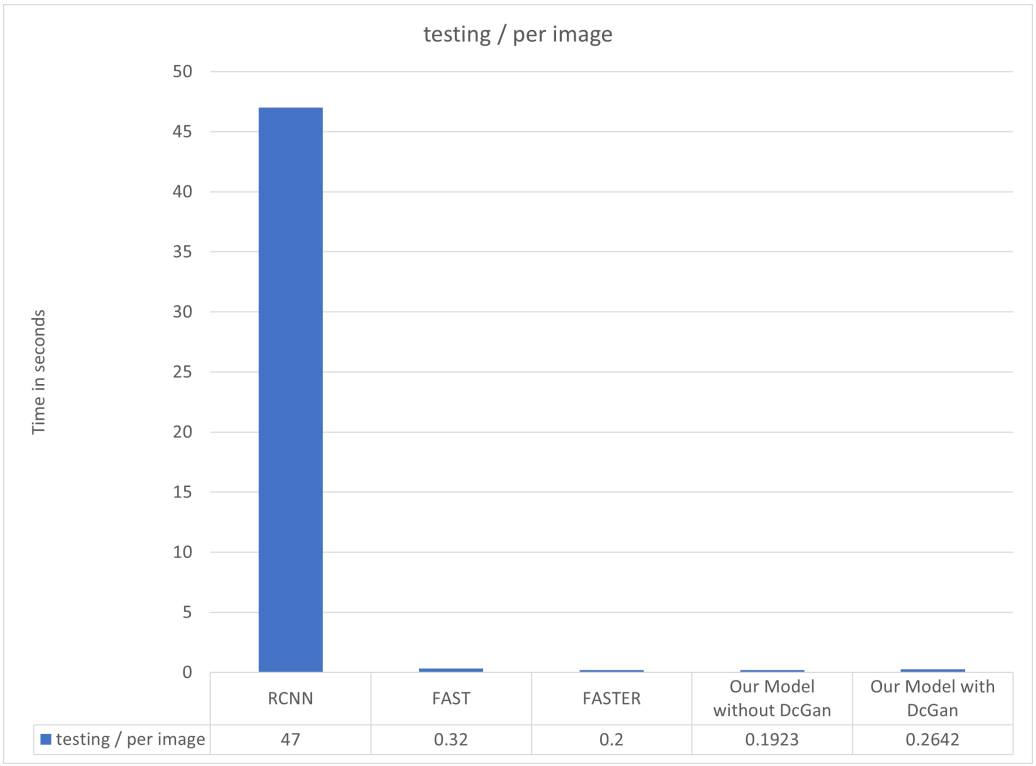


Figure 6. Comparison between RCNN, Fast R-CNN, Faster R-CNN, and the proposed model detection speed

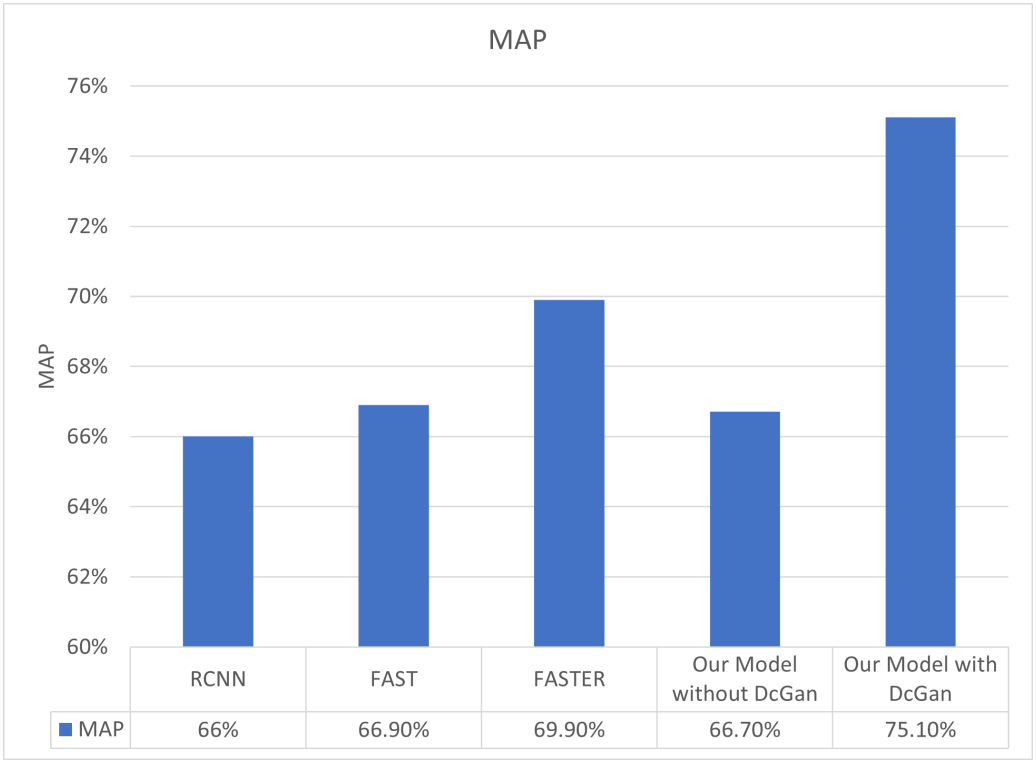


Figure 7. Comparison between RCNN, Fast R-CNN, Faster R-CNN, and the proposed model detection precision

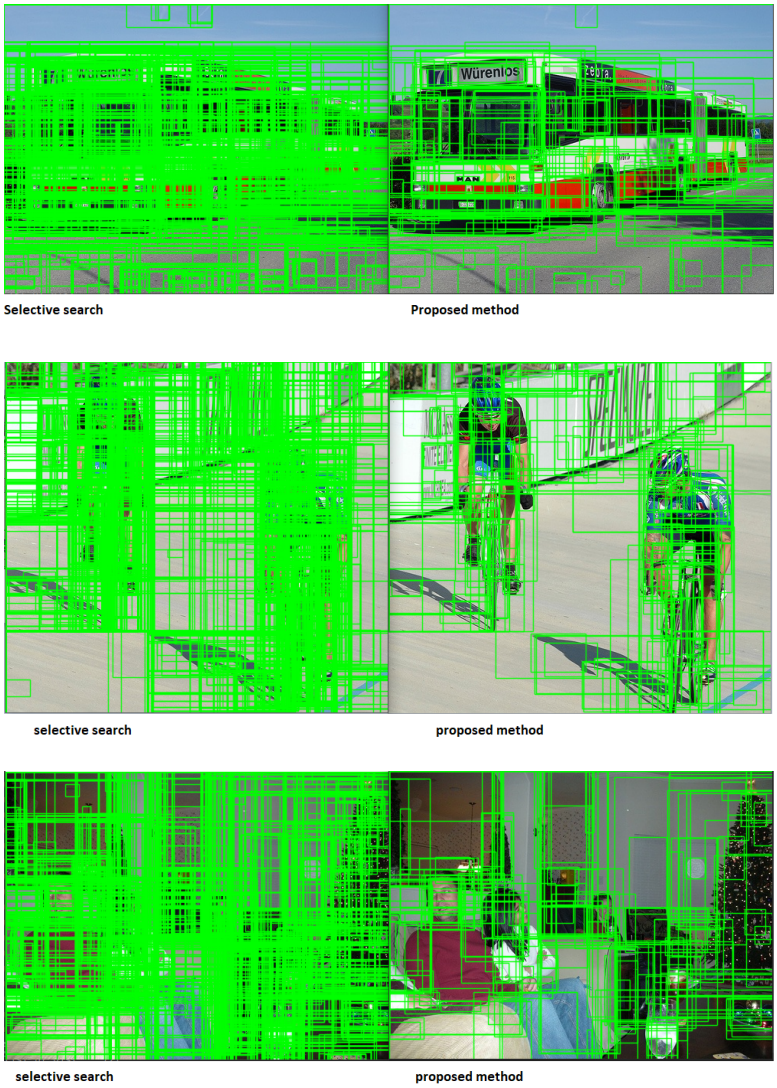


Figure 8. Number of candidate regions using selective search vs. candidate regions using proposed model

regions generated by our proposed method and other methods, including R-CNN and Fast R-CNN. Our proposed method was found to be faster than Fast, mask, and cascade R-CNN, almost the same as faster R-CNN. In terms of accuracy, the proposed method is more accurate than Fast and Faster R-CNN, with a MAP of 75.104%, 66.9%, and 69.9%, respectively.

REFERENCES

Khan, A.A., Laghari, A.A., Awan, S.A.: Machine learning in computer vision: a review. EAI Endorsed Transactions on Scalable Information Systems 8(32), e4–e4 (2021)

Xu, S., Wang, J., Shou, W., Ngo, T., Sadick, A.M., Wang, X.: Computer vision techniques in construction: a critical review. Archives of Computational Methods in Engineering 28, 3383–3397 (2021)

Zou, Z., Shi, Z., Guo, Y., Ye, J.: Object detection in 20 years: A survey. arXiv preprint arXiv:1905.05055 (2019)

Figure 9. Enhancement of a region from figure 8 using DcGAN

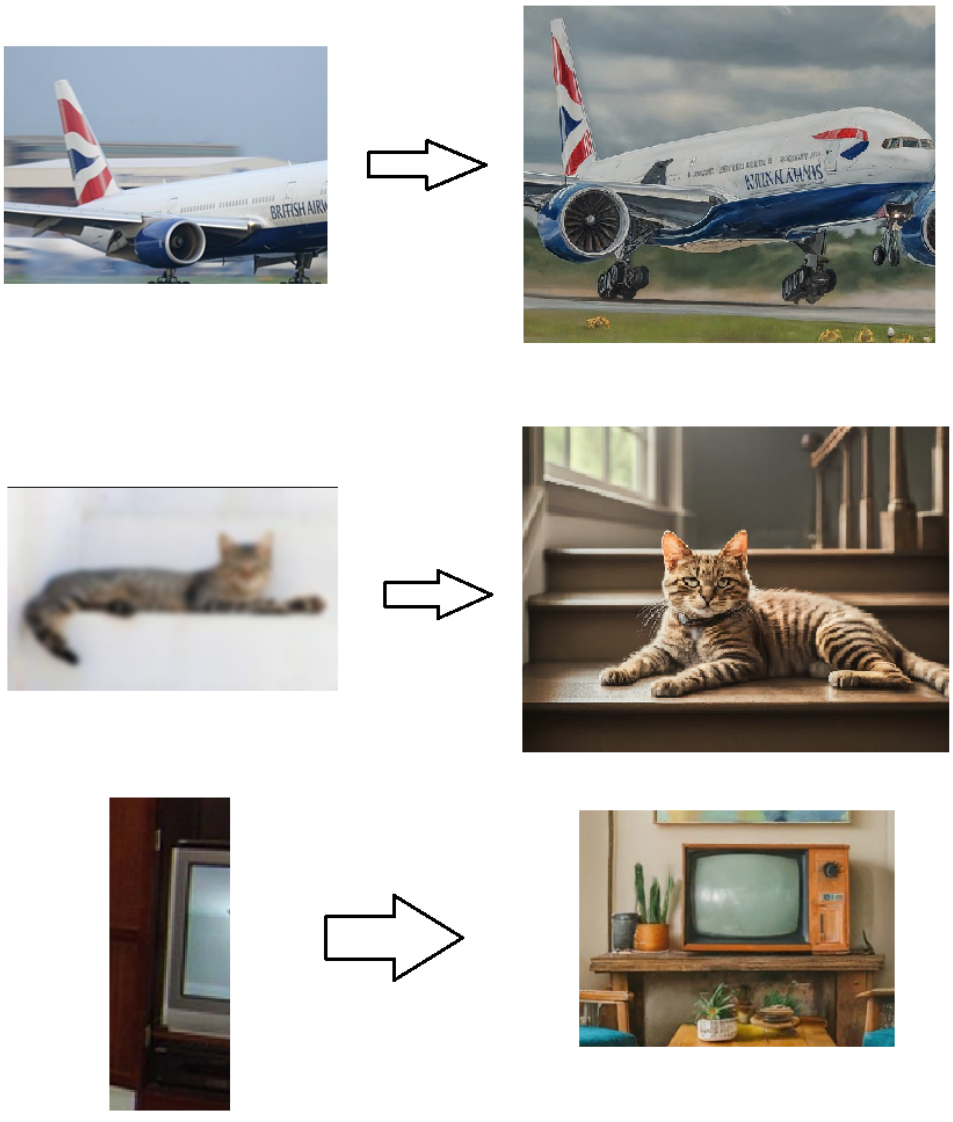


Figure 10. Examples of DcGAN enhancements

Verschae, R., Ruiz-del Solar, J.: Object detection: current and future directions. Frontiers in Robotics and AI 2, 29 (2015)

Walsh, J., O' Mahony, N., Campbell, S., Carvalho, A., Krpalkova, L., Velasco-Hernandez, G., Harapanahalli, S., Riordan, D.: Deep learning vs. traditional computer vision (04 2019)

Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)

Zhao, Z.Q., Zheng, P., Xu, S.t., Wu, X.: Object detection with deep learning: A review. IEEE transactions on neural networks and learning systems 30(11), 3212–3232 (2019)

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: A survey. International journal of computer vision 128(2), 261–318 (2020)

Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision 60(2), 91–110 (2004)

Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). vol. 1, pp. 886–893. Ieee (2005)

Pisner, D.A., Schnyer, D.M.: Support vector machine. In: Machine learning, pp. 101–121. Elsevier (2020)

Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 580–587 (2014)

Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015)

Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems 28 (2015)

He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence 37(9), 1904–1916 (2015)

He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)

Cai, Z., Vasconcelos, N.: Cascade r-cnn: High quality object detection and instance segmentation. IEEE transactions on pattern analysis and machine intelligence 43(5), 1483–1498 (2019)

Zaidi, S.S.A., Ansari, M.S., Aslam, A., Kanwal, N., Asghar, M., Lee, B.: A survey of modern deep learning based object detection models. Digital Signal Processing p. 103514 (2022)

Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)

Li, C.j., Qu, Z., Wang, S.y., Liu, L.: A method of cross-layer fusion multi-object detection and recognition based on improved faster r-cnn model in complex traffic environment. Pattern Recognition Letters 145, 127–134 (2021)

Qiao, L., Zhao, Y., Li, Z., Qiu, X., Wu, J., Zhang, C.: Defrcn: Decoupled faster r-cnn for few-shot object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8681–8690 (2021)

Jiang, D., Li, G., Tan, C., Huang, L., Sun, Y., Kong, J.: Semantic segmentation for multiscale target based on object recognition using the improved faster-rcnn model. Future Generation Computer Systems 123, 94–104 (2021)

Uijlings, J.R., Van De Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. International journal of computer vision 104(2), 154–171 (2013)

Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. International journal of computer vision 59(2), 167–181 (2004)

Koutra, D., Parikh, A., Ramdas, A., Xiang, J.: Algorithms for graph similarity and subgraph matching. In: Proc. Ecol. inference conf. vol. 17 (2011)