 doi:10.20944/preprints202003.0313.v2

Article

Small-Object Detection in Remote Sensing Images with End-to-End Edge-Enhanced GAN and Object Detector Network

Jakaria Rabbi [1,*], Nilanjan Ray [1], Matthias Schubert [2], Subir Chowdhury [3] and Dennis Chao [3]

[1] Department of Computing Science, 2-32 Athabasca Hall, University of Alberta, Edmonton, AB T6G 2E8, Canada; nray1@ualberta.ca
[2] Institute for Informatic, Ludwig-Maximilians-Universität München, Oettingenstraße 67, D-80333 Munich, Germany; schubert@dbs.ifi.lmu.de
[3] Alberta Geological Survey, Alberta Energy Regulator, Edmonton, Alberta, Canada; subir.chowdhury@aer.ca; dennis.chao@aer.ca
* Correspondence: jakaria@ualberta.ca

Abstract: The detection performance of small objects in remote sensing images is not satisfactory compared to large objects, especially in low-resolution and noisy images. A generative adversarial network (GAN)-based model called enhanced super-resolution GAN (ESRGAN) shows remarkable image enhancement performance, but reconstructed images miss high-frequency edge information. Therefore, object detection performance degrades for small objects on recovered noisy and low-resolution remote sensing images. Inspired by the success of edge enhanced GAN (EEGAN) and ESRGAN, we apply a new edge-enhanced super-resolution GAN (EESRGAN) to improve the image quality of remote sensing images and use different detector networks in an end-to-end manner where detector loss is backpropagated into the EESRGAN to improve the detection performance. We propose an architecture with three components: ESRGAN, Edge Enhancement Network (EEN), and Detection network. We use residual-in-residual dense blocks (RRDB) for both the ESRGAN and EEN, and for the detector network, we use the faster region-based convolutional network (FRCNN) (two-stage detector) and single-shot multi-box detector (SSD) (one stage detector). Extensive experiments on a public (car overhead with context) and a self-assembled (oil and gas storage tank) satellite dataset show superior performance of our method compared to the standalone state-of-the-art object detectors.

Keywords: Object detection; faster region-based convolutional neural network (FRCNN); single-shot multi-box detector (SSD); super-resolution; remote sensing imagery; edge enhancement; satellites

1. Introduction

1.1. Problem Description and Motivation

Object detection on remote sensing imagery has numerous prospects in various fields such as environmental regulation, surveillance, military [1,2], national security, traffic, forestry [3], oil and gas activity monitoring and other domains. There are many methods on detecting and locating objects from images that are captured using satellites or drones, but detection performance is not satisfactory on noisy and low-resolution images, especially when the objects are small [4]. Even on high-resolution imagery, the detection performance of small objects is lower compared to large objects [5].

Current state-of-the-art detectors have excellent accuracy on benchmark datasets such as ImageNet [6] and Microsoft Common Objects in Context (MSCOCO) [7]. These datasets consist of everyday images with distinguishable features and comparatively large object sizes.

On the other hand, there are various objects in satellite images like vehicles, small houses, small oil and gas storage tanks etc., only covering a small area [4]. The state-of-the-art detectors [8–11] show

a significant performance gap between low-resolution images and their high-resolution counterparts due to a lack of input features for small objects [12]. In addition to general object detectors, researchers have proposed specialized methods, algorithms, and network architectures to detect particular types of objects from satellite images such as vehicles [13,14], buildings [15], and storage tanks [16]. These methods are object-specific and use fixed resolution for feature extraction and detection.

To improve detection accuracy on remote sensing images, researchers have used deep convolutional neural network (CNN)-based super-resolution techniques to generate artificial high-resolution images and then detect objects [5,12]. Deep CNN-based super-resolution techniques such as image super-resolution using deep convolutional networks (SRCNN) [17] and accurate image super-resolution using very deep convolutional networks (VDSR) [18] show excellent results on generating realistic high-resolution imagery from low-resolution input data. Generative Adversarial Network (GAN)-based [19] methods such as super-resolution GAN (SRGAN) [20] and enhanced super-resolution GAN (ESRGAN) [21] show remarkable performance in enhancing low-resolution images with and without noise. These models have two subnetworks: a generator and a discriminator. Both subnetworks consist of deep CNNs. Datasets containing high-low resolution image pairs are used for training and testing the models. The generator generates high-resolution images from low-resolution input images, and the discriminator tries to predict whether a high-resolution image is real or an upscaled low-resoultion image. After sufficient training, the generator generates high-resolution images that are similar to the ground truth high-resolution images, and the discriminator cannot discriminate between real and fake images correctly anymore.

Although the resulting images look realistic, the compensated high-frequency details such as image edges may cause inconsistency with the high-resolution ground truth [22]. Some works show that this issue negatively impacts land cover classification results [23,24]. Edge information is an important feature for object detection [25], and therefore, this information is needed to be preserved in the enhanced images to get good detection accuracy.

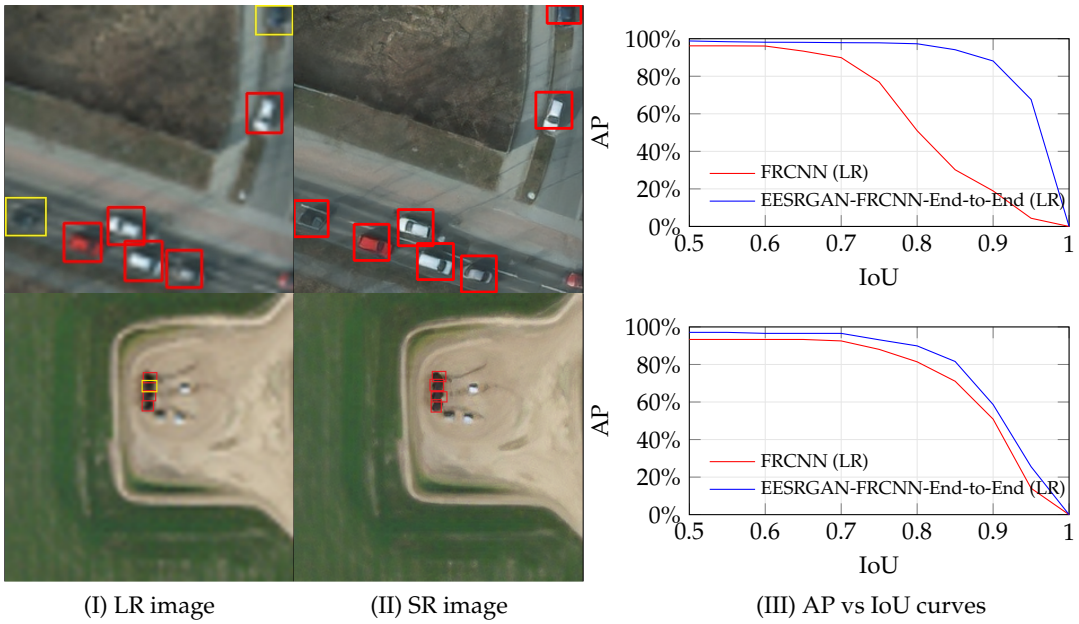
In order to obtain clear and distinguishable edge information, researchers proposed several methods using separate deep CNN edge extractors [26,27]. The results of these methods are good for everyday images, but the performance degrades on low-resolution and noisy remote sensing images [22]. Recent methods, such as [22] uses the GAN-based edge-enhancement network (EEGAN) to generate a visually pleasing result with sufficient edge information. EEGAN uses two subnetworks for generator. One network generates intermediate high-resolution images, and the other network generates the sharp and noise-free edges from the intermediate images. The method uses the Laplacian operator [28] to extract edge information and in addition, it uses a mask branch to get noise-free edges. This approach preserves sufficient edge information, but sometimes the final output images are blurry compared to a current state-of-the-art GAN-based super-resolution method [21] due to some noises introduced in the enhanced edge information. Therefore, this noise might hurt object detection performance.

Another important issue with small-object detection is the huge cost of very high-resolution imagery for large areas. Many organizations are using very high-resolution satellite imagery to fulfill their purposes. When it comes to continuous monitoring of a large area for regulation or traffic purposes, it is costly to buy high-resolution imagery frequently. Publicly available satellite imagery such as Landsat-8 [29] (30 m/pixel) and Sentinel-2 [30] (10 m/pixel) are not suitable for detecting small objects due to the high ground sampling distance (GSD). Detection of small objects (e.g., oil and gas storage tanks and buildings) is possible from commercial satellite imagery such as 1.5-m GSD SPOT-6 imagery but the detection accuracy is low compared to very high-resolution imagery, e.g., 30-cm GSD DigitalGlobe imagery in Bing map.

We identify two main problems to detect small-objects from satellite imagery. First, the accuracy of small-object detection is lower compared to large objects, even in very high-resolution imagery due to sensor noise, atmospheric effects, and geometric distortion. Secondly, we need to have access to very high-resolution imagery, which is very costly for a vast region with frequent updates. Therefore,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

we need a solution to increase the accuracy of the detection of smaller objects from low-resolution imagery. To the best of our knowledge, no work employs both super-resolution network with edge enhancement and object detector network in an end-to-end manner to detect small remote sensing objects.



(I) LR image (II) SR image (III) AP vs IoU curves

Figure 1. Detection performance on low and high-resolution images is shown in this figure. Detection on low-resolution (LR) images (60cm/pixel) is shown in (I); in (II), we show the detection on generated super-resolution (SR) images (15 cm/pixel). The first row of this figure represents the car overhead with context (COWC) dataset [31], and the second row represents the oil and gas storage tank (OGST) dataset [32]. Average precision (AP) values versus different intersection over union (IoU) values for the LR test set and generated SR images (from the LR images) are shown in (III) (for both the datasets). We use Faster R-CNN (FRCNN) on LR images for detection. Then instead of using LR images directly, we use our proposed end-to-end edge-enhanced super-resolution GAN and FRCNN (EESRGAN-FRCNN) architecture to generate SR images and simultaneously detect objects from the super-resolution images. The input is an LR image, and output is the detected bounding boxes on an SR image. Here the red bounding boxes represent true positives, and yellow bounding boxes represent false negatives, and IoU=0.75 is used for the detections.

In this paper, we propose an end-to-end architecture where object detection and super-resolution is performed simultaneously. Figure 1 shows the significance of our method. While state-of-the-art detectors misses objects, when trained on the low-resolution (LR) images, our method can detect those objects. The detection performance improves when we use super-resolved (SR) images for the detection of objects from two different datasets. Average precision (AP) versus different intersection over union (IoU) values (for both LR and SR) are plotted to visualize overall performance on test datasets. From Figure 1, we observe that for both the datasets, our proposed end-to-end method yields significantly better IoU values for the same AP. In section 4.2, we discuss AP and IoU in more detail, and details of these results are discussed in section 4.

1.2. Contributions of Our Method

Our proposed architecture consists of two parts: EESRGAN network and a detector network. Our approach is inspired by EEGAN and ESRGAN super-resolution networks and shows a remarkable improvement over EEGAN to generate visually pleasing super-resolved satellite images with enough edge information. We use a generator subnetwork, a discriminator subnetwork, and

an edge enhancement subnetwork [22] for the super-resolution network. For the generator and edge-enhancement network, we use Residual-in-Residual Dense Blocks (RRDB) [21]. These blocks use multi-level residual networks with dense connections that show good performance on image enhancement.

We use a relativistic discriminator [33] instead of a normal discriminator. Besides GAN loss and discriminator loss, we use Charbonnier loss [34] for the edge-enhancement network. Finally, we use different detectors [8,10] to detect small objects from the super-resolved image. The detectors act like the discriminator as we backpropagate the detection loss into the super-resolution network and, therefore, it improves the quality of the super-resolved images.

We create the oil and gas storage tank (OGST) dataset [32] from satellite imagery (Bing map), which has 30 cm and 1.2 m GSD. The dataset contains labeled oil and gas storage tanks from the Canadian province of Alberta, and we detect the tanks on super-resolved images. Detection and counting of the tanks are essential for the Alberta Energy Regulator (AER) [35] to ensure safe, efficient, orderly, and environmentally responsible development of energy resources. Therefore, there is a potential usage of our method for detecting small objects from low-resolution satellite imagery. The OGST dataset is available on Mendeley [32].

In addition to the OGST dataset, we apply our method on the publicly available car overhead with context (COWC) [31] dataset to compare the performance of detection for varying use-cases. During training, we use high and low-resolution image pairs but only require low-resolution images for testing. Our method outperforms the standalone state-of-the-art detectors for both datasets.

The remainder of this paper is structured as follows. We discuss related work in section 2. In section 3, we introduce our proposed method and describe every part of the method. The description of datasets and experimental results are shown in section 4, final discussion is stated in section 5 and section 6 concludes our paper with a summary.

2. Related Works

Our work consists of an end-to-end edge enhanced image super-resolution network with an object detector network. In this section, we discuss some research related to our method.

2.1. Image Super-Resolution

There are many works on super-resolution using deep CNNs. Dong et al. propose super-resolution CNN (SRCNN) [17] to enhance low-resolution image in an end-to-end training outperforming previous super-resolution techniques. The deep CNNs for super-resolution evolved rapidly later on, and researchers introduce residual blocks [20], densely connected networks [36], and residual dense block [37] for improving super-resolution results. He et al. [38] and Lim et al. [39] use deep CNNs without the batch normalization (BN) layer and observe significant performance improvement and stable training with a deeper network. These works are done on everyday images.

Liebel et al. [40] propose deep CNN-based super-resolution network for multi-spectral remote sensing imagery. Jiang et al. [22] propose a new super-resolution architecture for satellite imagery that is based on GAN. They introduce an edge-enhanced network to acquire smooth edge details in the final super-resolved image.

2.2. Object Detection

Deep learning object detectors are widely used today for any object detection task. These detectors can be categorized into two subgroups, region-based CNN (R-CNN) models that employ two-stage detection and uniform models using single stage detection [41]. Two-stage detectors comprise R-CNN [42], Fast R-CNN [43], Faster R-CNN [8] and the most used single stage detectors are SSD [10], You only look once (YOLO) [11] and RetinaNet [9]. In the first stage of a two-stage detector, regions of interest are determined by selective search or a region proposal network. Then, in the second stage, the selected regions are checked for particular types of objects and minimal bounding boxes for the

 doi:10.20944/preprints202003.0313.v2

detected objects are predicted. In contrast, single-stage detectors omit the region proposal network and run detection on a dense sampling of all possible locations. Therefore, single-stage detectors are faster but might be a little bit lower in accuracy. The RetinaNet [9] uses a focal loss function to deal with the data imbalance problem caused by many background objects and often shows similar performance as the two-stage approaches.

There are many works on the usage of deep CNNs to detect and count small objects in remote sensing imagery such as vehicles [13,44,45]. In [13], the authors introduce a convolutional regression neural network to detect vehicles from satellite imagery. Furthermore, a deep CNN-based detector is proposed in [44] to detect multi oriented vehicles from remote sensing imagery. A method combining a deep CNN for feature extraction and a support vector machine (SVM) for object classification is proposed in [45]. Ren et al. [46] modify the faster R-CNN detector to detect small objects in remote sensing images. They change the region proposal network and incorporate context information to the detector. Another modified faster R-CNN detector is proposed by Tang et al. [47]. They use a hyper region proposal network to improve recall and use a cascade boosted classifier to verify candidate regions. This classifier can reduce false detection by mining hard negative examples.

An SSD-based end-to-end airplane detector with transfer learning is proposed in [48]. In the paper, the authors use a limited number of airplane images for training. They also propose a method to solve the input size restrictions by dividing a large image into smaller tiles, then detect on smaller tiles and finally, map each image tile to the original image. The authors show that their method performs better than the SSD model. In [49], the authors show that finding a suitable parameter setting can boost the object detection performance of convolutional neural networks on remote sensing imagery. They use YOLO [11] as object detector to optimize the parameters and infer the results.

In [3], the authors detect conifer seedlings along recovering seismic lines from drone imagery. They use a dataset from different seasons and use faster R-CNN to infer the detection accuracy. There is another work [50] related to plant detection, where authors detect palm trees from satellite imagery using sliding window techniques and an optimized convolutional neural network.

Some works produce excellent results in detecting small objects. Lin et al. [51] propose feature pyramid networks, which is a top-down architecture with lateral connections. The architecture can build high-level semantic feature maps at all scales. These feature maps boost the object detection performance, especially for small object detection, when used as a feature extractor for faster R-CNN. Inspired by the receptive fields in human visual systems, Liu et al. [52] propose a receptive field block (RFB) module that uses the relationship between the size and eccentricity of receptive fields to enhance the feature discriminability and robustness. Hence, the module increases the detection performance of different sized objects when used as the replacement of the top convolutional layers of SSD.

A one-stage detector called single-shot refinement neural network (RefineDet) [53] is proposed to increase the detection accuracy and also enhance the inference speed. The detector works very well for small object detection. There are two modules for this architecture: A anchor refinement module to remove negative anchors and an object detection module that takes refined anchors as the input. The refinement helps to detect small objects efficiently than previous methods. In [54], feature fusion SSD (FSSD) is proposed where features from different layers with different scales are concatenated together, and then some downsampling blocks are used to generate new feature pyramids. Finally, the features are fed to multibox detector for prediction. The feature fusion in FSSD increases the detection performance for both large and small objects. Zhu et al. [55] train single-shot object detectors from scratch and get state-of-the-art performance on various public datasets. They remove the first downsampling layer of SSD and introduce root block (with modified convolutional filters) to exploit more local information from an image. Therefore, the detector can extract powerful features for small object detection.

All of the works discussed above are on everyday images. There is a work related to small object detection on remote sensing imagery propose by Yang et al. [56]. They use modified faster R-CNN

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

to detect both large and small objects. Authors propose rotation dense feature pyramid networks (R-DFPN), and the use of this network help to improve the detection performance of small objects.

There is an excellent review paper by Zhao et al. [57], where the authors show a thorough review of object detectors and also show the advantages and disadvantages of different object detectors. The effect of object size is also discussed in the paper. Another survey paper about object detection in remote sensing images by Li et al. [58] shows review and comparison among different methods.

2.3. Simultaneous Super-resolution with Object Detection

The positive effects of super-resolution on object detection tasks is discussed in [5] where the authors also use remote sensing datasets for their experiments. Simultaneous CNN-based image enhancement with object detection using single-shot multi-box detector (SSD) [10] is done in [59]. In [60], authors propose a GAN-based generator to generate a high-resolution image from a low-resolution image and then use a multi-task network as a discriminator and also for localization and classification of objects. These works are done on everyday normal images, and low and high-resolution image pairs are required. In another work [12], a method using simultaneous super-resolution with object detection on satellite imagery is proposed. The super-resolution network in this approach is inspired by the cycle-consistent adversarial network [61]. A modified faster R-CNN architecture is used to detect vehicles from enhanced images which were produced by the super-resolution network.

3. Method

In this paper, we aim to improve the detection performance of small objects on remote sensing imagery. Therefore, we propose an end-to-end network architecture that consists of two modules: A GAN based super-resolution network and a detector network. The whole network is trained in an end-to-end manner and high-low resolution image pairs are needed for training.

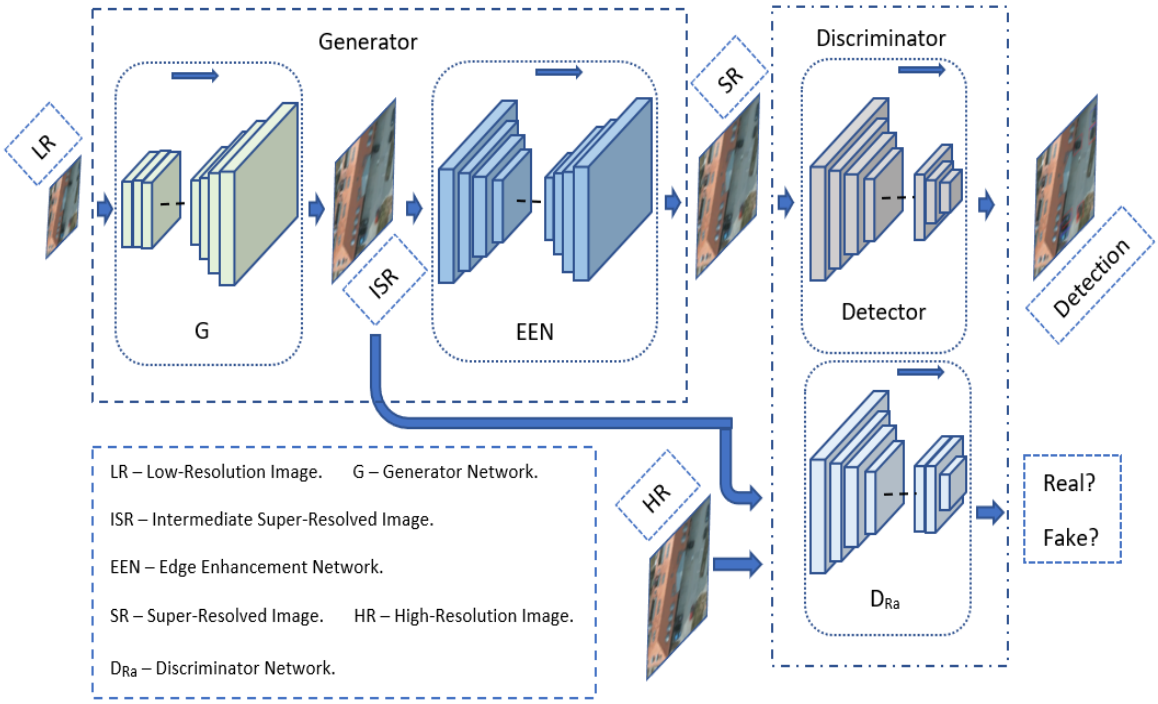


Figure 2. Overall network architecture with a generator and a discriminator module.

The super-resolution network has three components: generator (G), discriminator (D_{Ra}), and edge enhancement network (EEN). Our method uses end-to-end training as the gradient of the detection loss from the dectector is backpropagated into the generator. Therefore, the detector also works like a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

discriminator and encourages the generator G to generate realistic images similar to the ground truth. Our entire network structure can also be divided into two parts: A generator consisting of the EEN and a discriminator, which includes the D_{Ra} and the detector network. In Figure 2, we show the role of the detector as a discriminator.

The generator G generates intermediate super-resolution (ISR) images, and then final super-resolution (SR) images are generated after applying the EEN network. The discriminator (D_{Ra}) discriminates between ground truth high-resolution images and ISR. The inverted gradients of D_{Ra} are backpropagated into the generator G in order to create high-resolution images allowing for accurate object detection. Edge information is extracted from ISR, and the EEN network enhances these edges. Afterwards, the enhanced edges are again added to the ISR after subtracting the original edges extracted by the Laplacian operator and we get the output SR image with enhanced edges. Finally, we detect objects from the SR image using the detector network.

We use two different loss functions for EEN: one compares the difference between SR and ground truth images, and the other compares the difference between the extracted edge from ISR and ground truth. We also use the VGG19 [62] network for feature extraction that is used for perceptual loss [21]. Hence, it generates more realistic images with more accurate edge information. We divide the whole pipeline as a generator, and a discriminator, and these two components are elaborated in the following.

3.1. Generator

Our generator consists of a generator network G and an edge enhancement network EEN. In this section, we describe the architectures of both networks and the corresponding loss function.

3.1.1. Generator Network *G*

We use the generator architecture from ESRGAN [21], where all batch normalization (BN) layers are removed, and RRDB is used. The overall architecture of generator *G* is shown in Figure 3, and the RRDB is depicted in Figure 4.

Inspired by the authors of ESRGAN, we remove BN layers to increase the performance of the generator *G* and to reduce the computational complexity. The authors of ESRGAN also state that the BN layers tend to introduce unpleasant artifacts and limit the generalization ability of the generator when the statistics of training and testing datasets differ significantly.

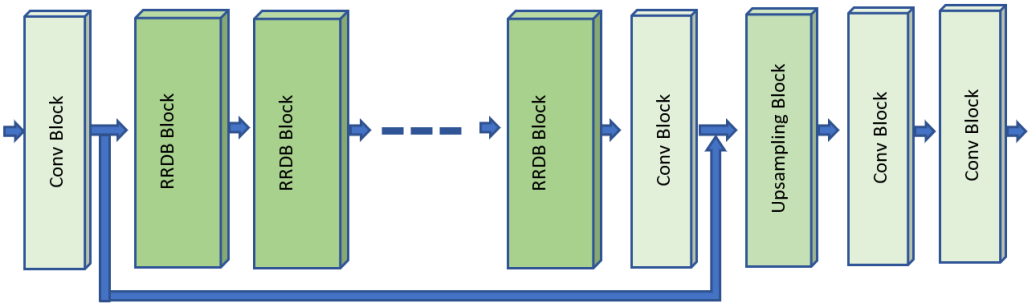
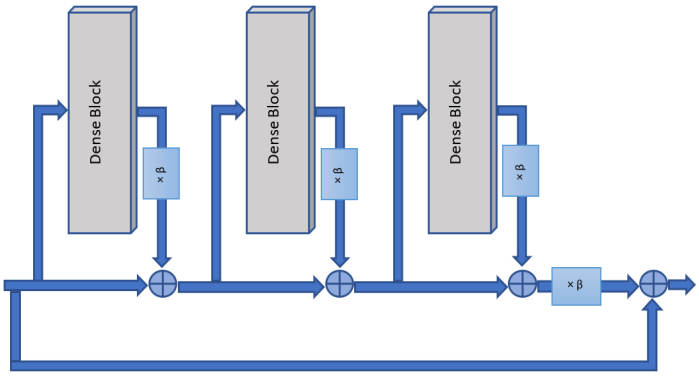


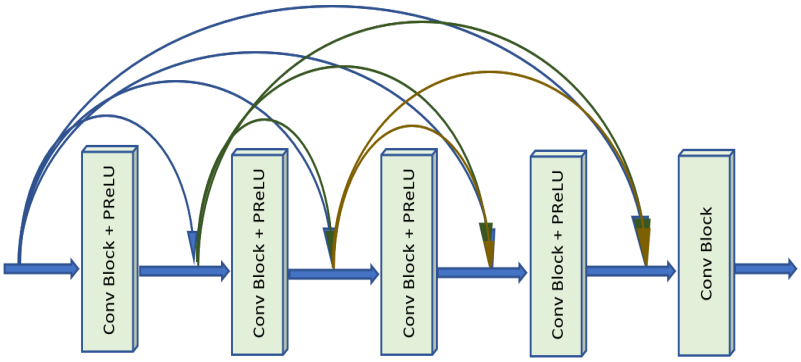
Figure 3. Generator *G* with RRDB, convolutional and upsampling blocks.

We use RRDB as the basic blocks of the generator network *G* that uses a multi-level residual network with dense connections. Those dense connections increase network capacity, and we also use residual scaling to prevent unstable conditions during the training phase [21]. We use the parametric rectified linear unit (PReLU) [63] for the dense blocks to learn the parameter with the other neural network parameters. As discriminator (D_{Ra}), we employ a relativistic average discriminator similar to the work represented in [21].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2



(a) RRDB from generator.



(b) Dense block from RRDB.

Figure 4. Internal diagram of RRDB.

In equation 1 and 2, the relativistic average discriminator is formulated for our architecture. Our generator G depends on the discriminator D_{Ra}, and hence we briefly discuss the discriminator D_{Ra} here and then, describe all details in section 3.2. The discriminator predicts the probability that a real image (I_{HR}) is relatively more realistic than a generated intermediate image (I_{ISR}).

$$D_{Ra}(I_{HR}, I_{ISR}) = \sigma(C(I_{HR}) - \mathbb{E}_{I_{ISR}}[C(I_{ISR})]) \rightarrow 1 \quad \text{More Realistic than fake data?} \tag{1}$$

$$D_{Ra}(I_{ISR}, I_{HR}) = \sigma(C(I_{ISR}) - \mathbb{E}_{I_{HR}}[C(I_{HR})]) \rightarrow 0 \quad \text{Less realistic than real data?} \tag{2}$$

In equation 1 and 2, σ, $C(\cdot)$ and $\mathbb{E}_{I_{ISR}}$ represents the sigmoid function, discriminator output and operation of calculating mean for all generated intermediate images in a mini-batch. The generated intermediate images are created by the generator where $I_{ISR} = G(I_{LR})$. It is evident from equation 3 that the adversarial loss of the generator contains both I_{HR} and I_{ISR} and hence, it benefits from the gradients of generated and ground truth images during the training process. The discriminator loss is depicted in equation 4.

$$L_G^{Ra} = -\mathbb{E}_{I_{HR}}[\log(1 - D_{Ra}(I_{HR}, I_{ISR}))] - \mathbb{E}_{I_{ISR}}[\log(D_{Ra}(I_{ISR}, I_{HR}))] \tag{3}$$

$$L_D^{Ra} = -\mathbb{E}_{I_{HR}}[\log(D_{Ra}(I_{HR}, I_{ISR}))] - \mathbb{E}_{I_{ISR}}[\log(1 - D_{Ra}(I_{ISR}, I_{HR}))] \tag{4}$$

We use two more losses for generator G: one is perceptual loss (L_{percep}), and another is content loss (L_1) [21]. The perceptual loss is calculated using the feature map ($vgg_{fea}(\cdot)$) before the activation layers

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

of a fine-tuned VGG19 [62] network, and the content loss calculates the 1-norm distance between I_{ISR} and I_{HR}. Perceptual loss and content loss is shown in equation 5 and equation 6.

$$L_{percep} = \mathbb{E}_{I_{LR}} ||vgg_{fea}(G(I_{LR}) - vgg_{fea}(I_{HR})||_1 \quad (5)$$

$$L_1 = \mathbb{E}_{I_{LR}} ||G(I_{LR}) - I_{HR}||_1 \quad (6)$$

3.1.2. Edge Enhancement Network EEN

The edge enhancement network EEN removes noise and enhances the extracted edges from an image. An overview of the network is depicted in Figure 5. In the beginning, Laplacian operator [28] is used to extract edges from the input image. After the edge information is extracted, it is passed through convolutional, RRDB, and upsampling blocks. There is a mask branch with sigmoid activation to remove edge noise as described in [22]. Finally, the enhanced edges are added to the input image where the edges extracted by the Laplacian operator were subtracted.

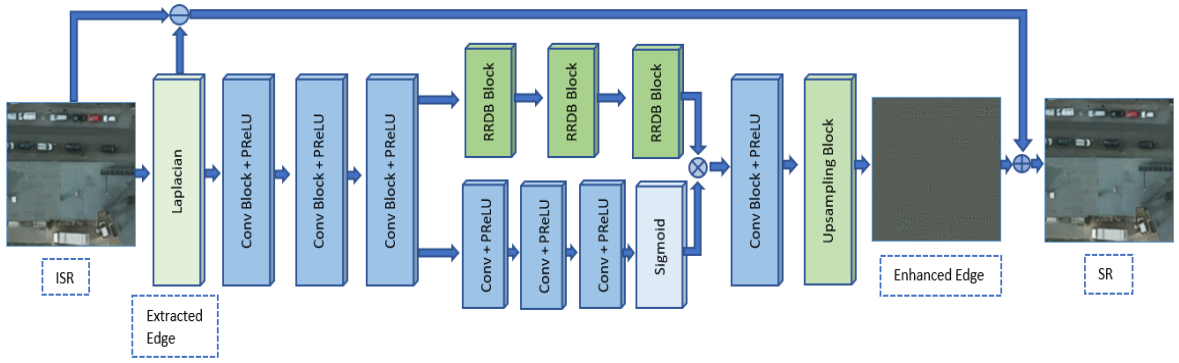


Figure 5. Edge enhancement network where input is an intermediate super-resolved image (ISR) and output is a super-resolved image (SR).

The EEN network is similar to the edge-enhancement subnetwork proposed in [22] with two improvements. First, we replace the dense blocks with RRDB. The RRDB shows improved performance according to ESRGAN [21]. Hence, we replace the dense block for improved performance of the EEN network. Secondly, we introduce a new loss term to improve the reconstruction of the edge information. In [22], authors extract the edge information from I_{ISR} and enhance the edges using an edge enhancement sub-network which is afterwards added to the edge-subtracted I_{ISR}. To train the network, [22] proposed to use Charbonnier loss [34] between the I_{ISR} and I_{HR}. This function is called consistency loss for images (L_{img_cst}) and helps to get visually pleasant outputs with good edge information. However, sometimes the edges of some objects are distorted and produce some noises and consequently, do not give good edge information. Therefore, we introduce a consistency loss for the edges (L_{edge_cst}) as well. To compute L_{edge_cst} we evaluate the Charbonnier loss between the extracted edges (I_{edge_SR}) from I_{SR} and the extracted edges (I_{edge_HR}) from I_{HR}. The two consistency losses are depicted in equation 7 and equation 8 where $\rho(\cdot)$ is the Charbonnier penalty function [64]. The total consistency loss is finally calculated for both images and edges by summing up the individual loss. The loss of our EEN is shown in equation 9.

$$L_{img\_cst} = \mathbb{E}_{I_{SR}}[\rho(I_{HR} - I_{SR})] \quad (7)$$

$$L_{edge\_cst} = \mathbb{E}_{I_{edge\_SR}}[\rho(I_{edge\_HR} - I_{edge\_SR})] \quad (8)$$

$$L_{een} = L_{img\_cst} + L_{edge\_cst} \quad (9)$$

Finally, we get the overall loss for the generator module by adding the losses of the generator G and the EEN network. The overall loss for the generator module is shown in equation 10 where λ_1, λ_2, λ_3,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

and λ_4 are the weight parameters to balance different loss components. We empirically set the values as $\lambda_1 = 1$, $\lambda_2 = .001$, $\lambda_3 = .01$, and $\lambda_4 = 5$.

$$L_{G\_een} = \lambda_1 L_{percep} + \lambda_2 L_G^{Ra} + \lambda_3 L_1 + \lambda_4 L_{een} \tag{10}$$

3.2. Discriminator

As described in the previous section, we use the relativistic discriminator D_{Ra} for training the generator G. The architecture of the discriminator is taken from ESRGAN [21] which employs the VGG-19 [62] architecture. We use Faster R-CNN [8] and SSD [10] for our detector networks. The discriminator (D_{Ra}) and the detector network jointly act as discriminator for the generator module. We briefly describe these two detectors in the next two sections.

3.2.1. Faster R-CNN

The Faster R-CNN [8] is a two-stage object detector and contains two networks: a region proposal network (RPN) to generate region proposals from an image and another network to detect objects from these proposals. In addition, the second network also tries to fit the bounding boxes around the detected objects.

The task of the RPN is to return image regions that have a high probability of containing an object. The RPN network uses a backbone network such as VGG [62], ResNet, or ResNet with feature pyramid network [51]. These networks are used as feature extractors, and different types of feature extractors can be chosen based on their performance on public datasets. We use ResNet-50-FPN [51] as a backbone network for our faster R-CNN. We use this network because it displayed a higher precision than VGG-19 and ResNet-50 without FPN (especially for small object detection) [51]. Even though the use of a larger network might lead to a further performance improvement, we chose ResNet-50-FPN due to its comparably moderate hardware requirements and more efficient convergence times.

After the RPN, there are two branches for detection: a classifier and a regressor. The classification branch is responsible for classifying a proposal to a specific object, and the regression branch finds the accurate bounding box of the object. In our case, both datasets contain objects with only one class, and therefore, our classifier infers only two classes: the background class and the object class.

3.2.2. SSD

The SSD [10] is a single shot multibox detector that detects objects in a single stage. Here, single-stage means that classification and localization are done in a single forward pass through the network. Like Faster R-CNN, SSD also has a feature extractor network, and different types of networks can be used. To serve the primary purpose of SSD, which is speed, we use VGG-16 [62] as a feature extractor network. After this network, SSD has several convolutional feature layers of decreasing sizes. This representation can seem like a pyramid representation of images at different scales. Therefore, the detection of objects happens in every layer, and finally, we get the object detection output as class values and coordinates of bounding boxes.

3.2.3. Loss of the discriminator

The relativistic discriminator loss (L_D^{Ra}) is already described in the previous section and depicted in equation 4. This loss is added to the detector loss to get the final discriminator loss.

Both Faster R-CNN and SSD have similar regression/localization losses but different classification losses. For regression/localization, both use smooth L_1 [8] loss between detected and ground truth

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

bounding box coordinates (t_*). Classification (L_{cls_frcnn}) and regression loss (L_{reg_frcnn}) and overall loss (L_{det_frcnn}) of Faster R-CNN are given in the following:

$$L_{cls\_frcnn} = \mathbb{E}_{I_{LR}}[-\log(Det_{cls\_frcnn}(G_{G\_een}(I_{LR})))] \tag{11}$$
$$L_{reg\_frcnn} = \mathbb{E}_{I_{LR}}[smooth_{L1}(Det_{reg\_frcnn}(G_{G\_een}(I_{LR})), t_*)] \tag{12}$$
$$L_{det\_frcnn} = L_{cls\_frcnn} + \lambda L_{reg\_frcnn} \tag{13}$$

Here, λ is used to balance the losses, and it is set to 1 empirically. Det_{cls_frcnn} and Det_{reg_frcnn} are the classifier and regressor for the Faster R-CNN. Classification (L_{cls_ssd}), regression loss (L_{reg_ssd}) and overall loss (L_{det_ssd}) of SSD are as following:

$$L_{cls\_ssd} = \mathbb{E}_{I_{LR}}[-\log(softmax(Det_{cls\_ssd}(G_{G\_een}(I_{LR}))))] \tag{14}$$
$$L_{reg\_ssd} = \mathbb{E}_{I_{LR}}[smooth_{L1}(Det_{reg\_ssd}(G_{G\_een}(I_{LR})), t_*)] \tag{15}$$
$$L_{det\_ssd} = L_{cls\_ssd} + \alpha L_{reg\_ssd} \tag{16}$$

Here, α is used to balance the losses, and it is set to 1 empirically. Det_{cls_ssd} and Det_{reg_ssd} are the classifier and regressor for the SSD.

3.3. Training

Our architecture can be trained in separate steps or jointly in an end-to-end way. We discuss the details of these two types of training in the next two sections.

3.3.1. Separate Training

In separate training, we train the super-resolution network (generator module and discriminator D_{Ra}) and the detector separately. Detector loss is not backpropagated to the generator module. Therefore, the generator is not aware of the detector and thus, it only gets feedback from the discriminator D_{Ra}. For example, in equation 11, no error is backpropagated to the G_{G_een} network (the network is detached during the calculation of the detector loss) while calculating the loss L_{cls_frcnn}.

3.3.2. End-to-End Training

In end-to-end training, we train the whole architecture end-to-end that means the detector loss is backpropagated to the generator module. Therefore, the generator module revceives gradients from both detector and discriminator D_{Ra}. We get the final discriminator loss (L_{D_det}) as following:

$$L_{D\_det} = L_D^{Ra} + \eta L_{det} \tag{17}$$

Here, η is the parameter to balance the contribution of the detector loss and we empirically set it to 1. Finally, we get an overall loss ($L_{overall}$) for our architecture as follows.

$$L_{overall} = L_{G\_een} + L_{D\_det} \tag{18}$$

4. Experiments

As described above our architecture can be trained separately and in an end-to-end manner. For separate training, we first train the super-resolution until convergence and then train the detector networks based on the super-resolved images. For end-to-end training, we also employ separate training as pre-training step for weight initialization. Afterwards super-resolution and object detection networks are jointly trained, i.e., the gradients from the the object detector are propagated into the generator network.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

In the training process, the learning rate is set to 0.0001 and halved after every $50k$ iterations. The batch size is set to 5. We use Adam [65] as optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and update the whole architecture weights until convergence. We use 23 RRDB blocks for the generator G and 5 RRDB blocks for the EEN network. We implement our architecture with the PyTorch framework [66] and train/test using two NVIDIA Titan X GPUs. The end-to-end training with COWC takes 96 hours for 200 epochs. The average inference speed using faster R-CNN is approximately 4 images/second and 7 images/second for SSD. Our implementation can be found in GitHub [67].

4.1. Datasets

4.1.1. Cars Overhead with Context Dataset

Cars overhead with context (COWC) dataset [31] contains 15 cm (one pixel cover 15 cm distance at ground level) satellite images from six different regions. The dataset contains a large number of unique cars and covers regions from Toronto in Canada, Selwyn in New Zealand, Potsdam and Vaihingen in Germany, Columbus and Utah in the United States. Out of these six regions, we use the dataset from Toronto and Potsdam. Therefore, when we refer to the COWC dataset, we refer to the dataset from these two regions. There are 12651 cars in our selected dataset. The dataset contains only RGB images, and we use these images for training and testing.

(a) Low-resolution (b) High-resolution (c) Bounding box

Figure 6. COWC dataset: low/high-resolution image pairs are shown in (a) and (b) and ground truth bounding boxes for cars are in (c).

We use 256 pixels to 256 pixels as image tile size, and every image tile contains at least one car. The average length of a car is between 24 to 48 pixels, and the width is between 10 to 20 pixels. Therefore, the area of a car is between 240 to 960 pixels, which can be considered as a small object relative to the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

other large satellite objects. We use bi-cubic downsampling to generate low-resolution images from the COWC dataset. The downscale factor is 4x, and therefore, we have 64 pixels to 64 pixels size for low-resolution images. We have a text file associated with each image tile containing the coordinates of the bounding box for each car.

The dataset contains only one class which is car and does not consider any other type of object. Figure 6 shows examples from the COWC dataset. We have a total of 3340 tiles for training and testing. Our train/test split is 80%/20%, and the training set is further divided into a training and a validation set by an 80% to 20% ratio. We train our end-to-end architecture with an augmented training dataset with random horizontal flips and ninety-degree rotations.

4.1.2. Oil and Gas Storage Tank Dataset

The oil and gas storage tank (OGST) dataset is complied in Alberta Geological Survey (AGS) [68], a branch of the Alberta Energy Regulatory (AER) [35]. AGS provides geoscience information and support to AER's regulatory functions on energy developments to be carried out in a manner to ensure public and environmental safety. To assist AER with sustainable land management and compliance assurance [69], AGS is utilizing remote sensing imagery to identify the number of oil and gas storage tanks inside well pad footprints in Alberta.

(a) Low-resolution (b) High-resolution (c) Bounding box

Figure 7. OGST Dataset: low/high-resolution image pairs are shown in (a) and (b) and ground truth bounding boxes for oil and gas storage tanks are in (c).

While the SPOT-6 satellite image at 1.5 m pixel resolution provided by the AGS has sufficient quality and details for many regulatory functions, it is difficult to detect small objects within well pads, e.g., oil and gas storage tanks with ordinary object detection methods. The diameter of a typical storage tank is about 3 m and their placements are usually vertical and side-by-side with less than

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

2 m. To train our architecture for this use-case, a dataset is needed providing pairs of low and high resolution images. Therefore, we create the OGST dataset using free imagery from the Bing map [70].

The OGST dataset contains 30 cm resolution remote sensing images (RGB) from the Cold Lake Oil Sands region of Alberta, Canada where there is a high level of oil and gas activities and concentration of well pad footprints. The dataset contains 1671 oil and gas storage tanks from this area.

We use 512 pixels to 512 pixels for image tile size, and there is no image without any oil and gas storage tank. The average area covered by an individual tank is between 800 to 1600 pixels. Some industrial tanks are large, but most of the tanks cover small regions on the imagery. We downscale the high-resolution image using bi-cubic downsampling with the factor of 4x, and therefore, we get a low-resolution tile of 128 pixes to 128 pixels. Every image tile is associated with a text file containing the coordinates of the bounding boxes for the tanks on a tile. We show examples from the OGST dataset in figure 7.

The dataset contains one unique class: tank, and we have a total of 760 tiles for training and testing. We use a 90%/10% split for our train/test data. The training data is further divided by 90%/10% for the train/validation split. The percentage of training data is higher here compared to the previous dataset to increase the training data because of the smaller size of the dataset. The dataset is available at [67].

4.2. Evaluation Metrics for Detection

We get our detection output as bounding boxes with associated classes. To evaluate our results, we use average precision (AP), and we calculate intersection over union (IoU), precision, and recall to get AP.

We denote the set of correctly detected objects as true positives (TP) and the set of falsely detected objects of false positives (FP). The precision is now the ratio between the number of TPs relative to all predicted objects:

$$precision = \frac{|TP|}{|TP| + |FP|} \tag{19}$$

We denote the set of objects which are not detected by the detector as false negatives (FN). Then, the recall is defined as the ratio of detected objects (TP) relative to the number of all objects in the data set:

$$Recall = \frac{|TP|}{|TP| + |FN|} \tag{20}$$

To measure the localization error of predicted bounding boxes, intersection over union (IoU) measures the overlap between two bounding boxes: the detected and the ground truth box. IoU corresponds to the area of overlap between both boxes relative to the joint area of both boxes. If we take all the boxes that have an IoU $\geq \tau$ as TP and consider all other detections as FP, then we get the precision at τ IoU. If we now vary τ from 0.5 to 0.95 IoU with a step size of 0.05, we receive ten different precision values which can be combined into the average precision (AP) at IoU=0.5:0.95 [8]. Let us note that in the case of multi-class classification, we would need to compute the AP for object each class separately. To receive a single performance measure for object detection, the mean AP (mAP) is computed which is the most common performance measure for object detection quality.

In this paper, both of our datasets only contain single class, and hence, we use AP as our evaluation metric. We mainly show the results of AP at IoU=0.5:0.95 as our method performs increasingly better compared to other models when we increase the IoU values for AP calculation. We show this trend in section 4.3.4.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

4.3. Results

4.3.1. Detection without Super-Resolution

We run the two detectors to document the object detection performance on both low-resolution and high-resolution images. We use SSD with vgg16 [62] network and Faster R-CNN (FRCNN) with ResNet-50-FPN [51] detector. We train the two models with both high-resolution and 4x-downscaled low-resolution images. Testing is also done with both high-low resolution images.

In table 1, we show the results of the detection performance of the detectors with different train/test combinations. When we use only low-resolution images for both training and testing, we see 64% AP for Faster R-CNN. When training on high-resolution images and testing with low-resolution images, the accuracy drops for both detectors. We also add detection results (using low-resolution images for training/testing) for both the datasets using SSD with RFB modules (SSD-RFB) [52], where accuracy slightly increases from the base SSD. The last two rows depict the accuracy of both detectors when training and testing on high-resolution images. We achieve up to 98% AP with the Faster R-CNN detector. This, shows the large impact of the resolution to the object detection quality and sets a natural upper bound on how close a super-resolution-based method can get when working on low-resolution images. In the next sections, we will show that our approaches considerably improve the detection rate on low-resolution imagery and gets astonishingly close to the performance of directly working on high-resolution imagery.

Table 1. Detection without super-resolution

Model	Resolution of Images to Train and Test the Detector (Train - Test)	COWC Dataset (Test Results, AP) (IoU=0.5:0.95) (single class - 15 cm)	OGST Dataset (Test Results) (IoU=0.5:0.95) (single class - 30 cm)
SSD	Low - Low	61.9%	76.5%
	High - Low	58%	75.3%
FRCNN	Low - Low	64%	77.3%
	High - Low	59.7%	75%
SSD-RFB	Low - Low	63.1%	76.7%
SSD	High - High	94.1%	82.5%
FRCNN	High - High	98%	84.9%

4.3.2. Detection with Super-Resolution (Separate Training)

In this experiment, we create 4x upsampled images from the low-resolution input images using bicubic upsampling and EESRGAN. Let us note that no training is needed for applying bicubic upsampling since it is a parameter free function. We use the super-resolved images (SR) as test for two types of detectors. The first is trained on real high-resolution images whereas the second is trained on the SR data as well. In addition, we compare three GAN architectures for generating super-resolved images, our new EESRGAN architecture, ESRGAN [21] and EEGAN [22]. Each network is trained separately on the test set before the object detector was trained. For the evaluation, we again compare detectors being trained on the super-resolved images from the particular architecure and an detector being directly trained on the high-resolution images.

In table 2, the detection output of the different combinations of super-resolution methods and detectors is shown with the different combinations of train/test pairs. As can be seen, our new EESRGAN architecture displays the best results already getting close to the detection rates which could be observed when working with high-resolution images only. However, after training EESRGAN can be

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

directly applied to low-resolution imagery where no high-resolution data is available and still achieves very good results. Furthermore, we can observe that both other super-resolution methods EEGAN and ESRGAN already considerably improve the AP when used for preprocessing the low-resolution images. However, for both data sets, EESRGAN outperformed the other two methods.

Table 2. Detection with separately trained super-resolution network

Model	Resolution of Images to Train and Test the Detector (Train - Test)	COWC Dataset (Test Results, AP) (IoU=0.5:0.95) (single class - 15 cm)	OGST Dataset (Test Results) (IoU=0.5:0.95) (single class - 30 cm)
Bicubic + SSD	SR - SR	72.1%	77.6%
	High - SR	58.3%	76%
Bicubic + FRCNN	SR - SR	76.8%	78.5%
	High - SR	61.5%	77.1%
EESRGAN + SSD	SR - SR	86%	80.2%
	High - SR	83.1%	79.4%
EESRGAN + FRCNN	SR - SR	**93.6%**	**81.4%**
	High - SR	92.9%	80.6%
ESRGAN + SSD	SR - SR	85.8%	80.2%
	High - SR	82.5%	78.9%
ESRGAN + FRCNN	SR - SR	92.5%	81.1%
	High - SR	91.8%	79.3%
EEGAN + SSD	SR - SR	86.1%	79.1%
	High - SR	83.3%	77.5%
EEGAN + FRCNN	SR - SR	92%	79.9%
	High - SR	91.1%	77.9%

4.3.3. Detection with Super-Resolution (End-to-End Training)

Table 3. Detection with end-to-end super-resolution network

Model	Resolution of Images to Train and Test the Detector (Train - Test)	COWC Dataset (Test Results, AP) (IoU=0.5:0.95) (single class - 15 cm)	OGST Dataset (Test Results) (IoU=0.5:0.95) (single class - 30 cm)
EESRGAN + SSD	SR - SR	89.3%	81.8%
EESRGAN + FRCNN	SR - SR	**95.5%**	**83.2%**
ESRGAN + SSD	SR - SR	88.5%	81.1%
ESRGAN + FRCNN	SR - SR	93.6%	82%
EEGAN + SSD	SR - SR	88.1%	80.8%
EEGAN + FRCNN	SR - SR	93.1%	81.3%

We train our EESRGAN network and detectors end-to-end for this experiment. The discriminator (D_{Ra}), and the detectors jointly act as a discriminator for the entire architecture. Detector loss is backpropagated to the super-resolution network, and therefore, the loss contributes to the enhancement of low-resolution images. At training time, low-high resolution image pairs are used to train the

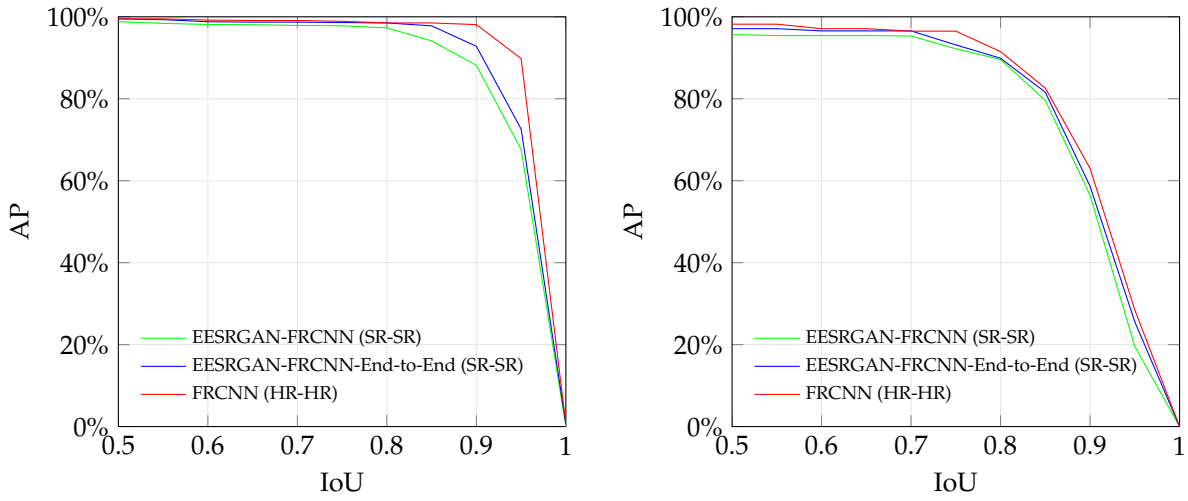
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

EEGAN part, and then the generated SR image are sent to the detector for training. At test time, only the low-resolution images are fed to the network. Our architecture first generates a super-resolved image of the low-resolution input before object detection is performed.

We also compare our results with different architectures. We use ESRGAN [21] and EEGAN [22] with the detectors for comparison. Table 3 clearly shows that our method delivers superior results compared to others.

4.3.4. AP versus IoU curve

We plot the AP values on different IoUs. In figure 8, we plot the AP versus IoU curves for our datasets. The performance of EESRGAN-FRCNN, end-to-end EESRGAN-FRCNN, and FRCNN is shown in the figure. The end-to-end EESRGAN-FRCNN network performs better than the separately trained network. The difference is most evident for the higher IoUs on the COWC dataset.



(a) COWC dataset **(b)** OGST dataset

Figure 8. AP-IoU curves for the datasets

Our results indicate excellent performance compared to the highest possible AP values (from FRCNN trained and tested on HR images)

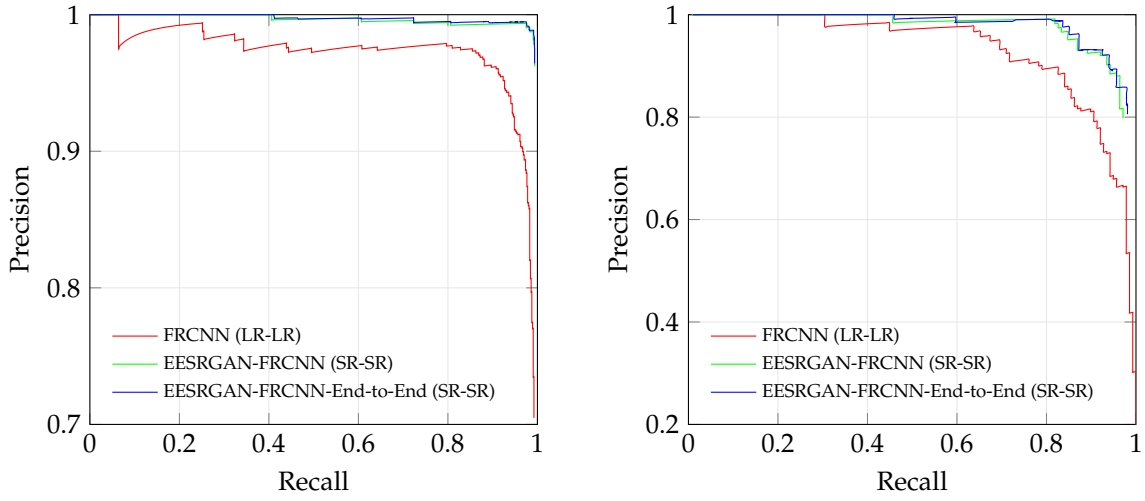
The OGST dataset displays less performance variation compared to the COWC dataset. The object size of the OGST dataset is larger than the COWC dataset. Therefore, the performance difference is not similar to the COWC dataset when we compare between standalone FRCNN and our method on the OGST dataset. To conclude, training our new architecture in an end-to-end manner displays an improvement for both datasets.

4.3.5. Precision versus Recall

In figure 9, the precision-recall curves are shown for both of our datasets. Precision-recall curve for COWC dataset is depicted in 9a and 9b represents the curve for OGST dataset. For each dataset, we plot the curves for standalone faster R-CNN with low-resolution training/testing images, and our method with/without end-to-end training. We use IoU=0.5 to calculate the precision and recall.

The precision-recall curves for both datasets show that our method has higher precision values in higher recall values compared to the standalone faster R-CNN models. Our model with end-to-end training performs better than our model without end-to-end training. In particular, the end-to-end model detects more than 99% of the cars with 96% AP in the COWC dataset. For the OGST dataset, our end-to-end model detects more than 81% of the cars with 97% AP.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2



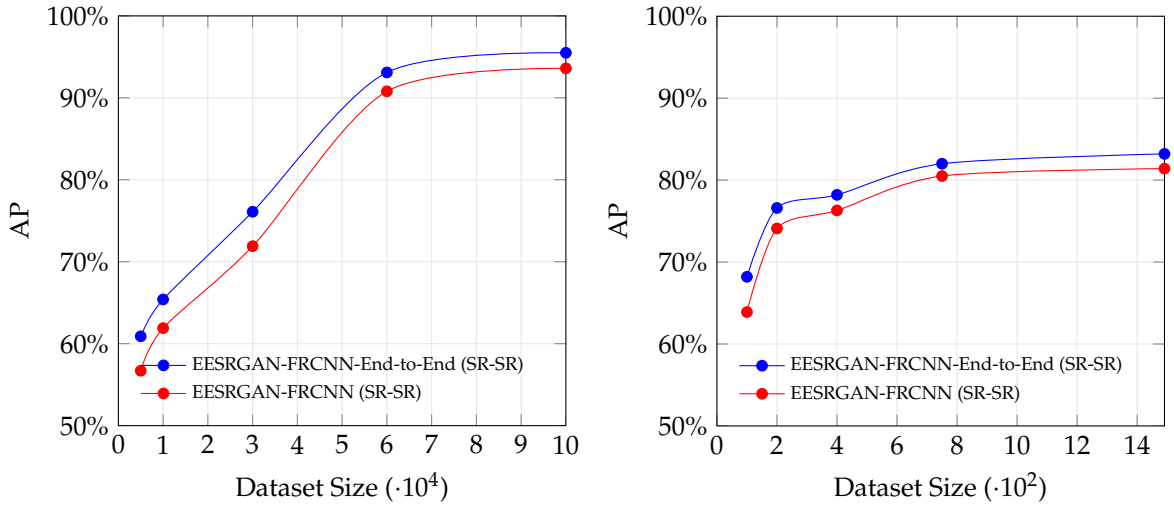
(a) COWC dataset **(b)** OGST dataset

Figure 9. Precision-recall curve for the datasets

4.3.6. Effects of Dataset Size

We train our architecture with different training set sizes and test with a fixed test set. In figure 10, we plot the AP values (IoU=0.5:0.95) against different numbers of labeled objects for both of our datasets (training data). We use five different dataset sizes: $\{500, 1000, 3000, 6000, 10000(cars)\}$ and $\{100, 200, 400, 750, 1491(tanks)\}$ to train our model with and without the end-to-end setting.

We get the highest AP value of 95.5% with our full COWC training dataset (10000 cars), and we use the same test dataset (1000 cars) for all combinations of the training dataset (with end-to-end setting). We also use another set of 1000 labeled cars for validation. Using 6000 cars, we get AP value near to the highest AP. The AP value decreases significantly when we use only 3000 labeled cars as training data. We get the lowest AP using only 500 labeled cars, and the trend of AP is further decreasing. Therefore, we can infer that we need around 6000 labeled cars to get precision higher than 90% for the COWC dataset. We have slightly lower AP values for all sizes of COWC datasets when we do not use the end-to-end setting, and we observe higher differences between the two settings (with and without end-to-end) when we use less than 6000 labeled cars.



(a) COWC dataset **(b)** OGST dataset

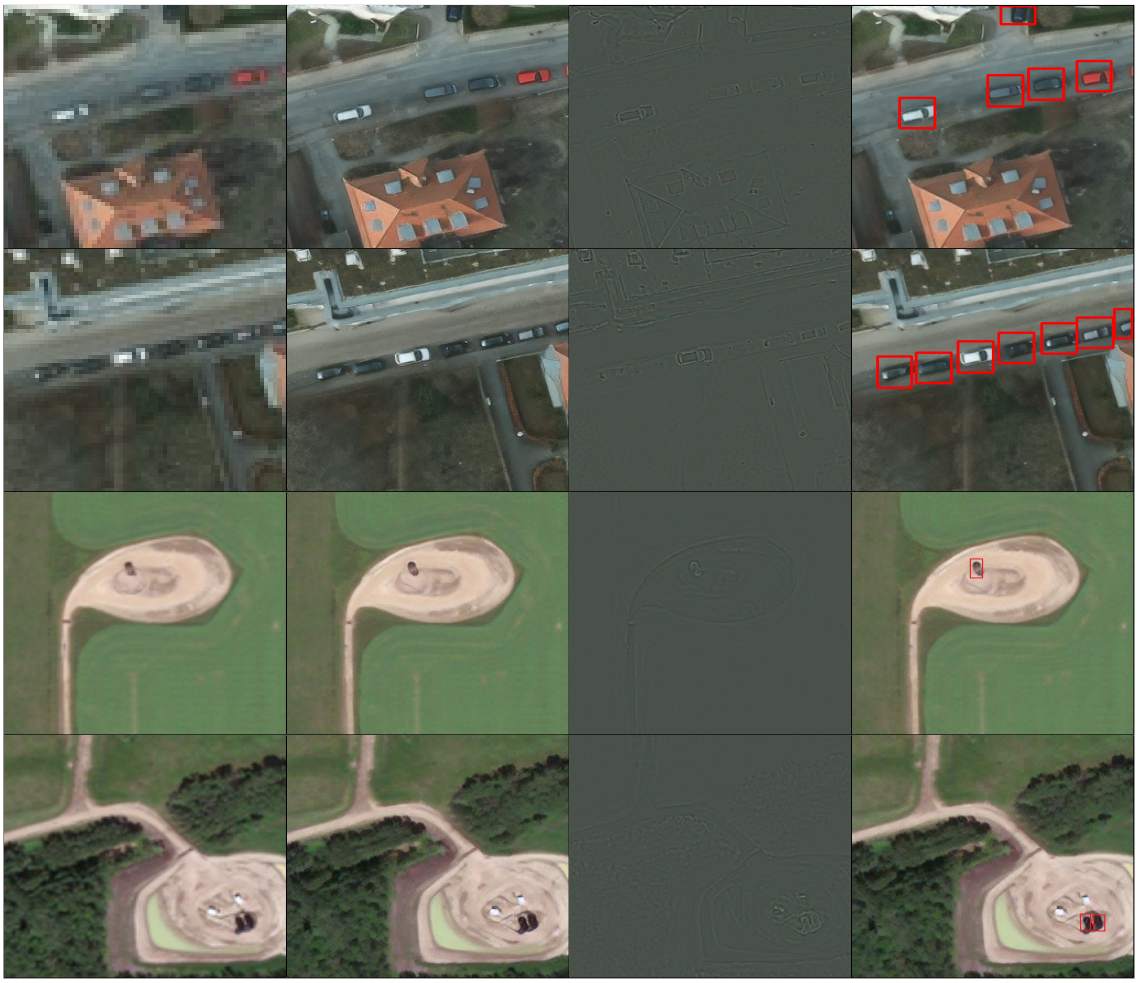
Figure 10. AP with varying number of training datasets from COWC dataset.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

The OGST dataset gives 83.2% AP (with end-to-end setting) using the full training dataset (1491 tanks), and we use 100 labeled tanks as test and same amount as validation data for all combinations of the training dataset. We get high AP values with 50% of our full training dataset. AP values drops below 80% if we further decrease number of training data. Similar to the COWC datasets, we also get comparatively lower AP values for all sizes of OGST datasets, and we also observe slightly higher differences between the two settings (with and without end-to-end) when the dataset consists of less than 400 labeled tanks.

We use 90% of the OGST dataset for training while we use the 80% of the COWC dataset for the same purpose. The accuracy of the testing data (OGST) slightly increases when we add more training data, as depicted in figure 10. Therefore, we use more percentage of training data for the OGST dataset than the COWC dataset, and it slightly helps to improve the relatively low accuracy of the OGST test data.

4.3.7. Enhancement and Detection



(a) Input LR image (b) Generated SR image (c) Enhanced edge (c) Detection

Figure 11. Examples of super-resolution (SR) images that are generated from input low-resolution (LR) images are shown in (a) and (b). The enhanced edge and detection results are shown in (c) and (d).

In figure 11, we show our input low-resolution (LR) images, corresponding generated super-resolution (SR) image, enhanced edge information and final detection. The image enhancement helps the detectors to get high AP values and also makes the images visually good enough to identify the objects easily. It is evident from the figure that the visual quality of the generated SR images is

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 April 2020 doi:10.20944/preprints202003.0313.v2

quite good compared to the corresponding LR images, and the FRCNN detector detects most of the objects correctly.

4.3.8. Effects of Edge Consistency Loss (L_{edge_cst})

In EEGAN [22] paper, only image consistency loss (L_{img_cst}) is used for enhancing the edge information. This loss generates edge information with noise, and as a result, the final SR images are blurry. The blurry output with noisy edge using only L_{img_cst} loss is shown in figure 12a. The blurry final images give lower detection accuracy compared to sharp outputs.

Therefore, we introduce edge consistency loss (L_{edge_cst}) in addition to L_{img_cst} loss that gives noise-free enhanced edge information similar to the edge extracted from ground truth image and the effects of the L_{edge_cst} loss is shown in figure 12b. The ground truth HR image with extracted edge is depicted in figure 12c.

(a) Final SR image and enhanced edge with L_{img_cst} loss

(b) Final SR image and enhanced edge with L_{img_cst} and L_{edge_cst} losses

(c) Ground truth HR image with extracted edge

Figure 12. Effects of edge consistency loss (L_{edge_cst}) on final SR images and enhanced edges.

5. Discussion

The detection results of our method presented in the previous section indicate that our end-to-end super-resolution-detector network improves detection accuracy compared to several other methods. Our method outperforms the standalone state-of-the-art methods such as SSD or faster R-CNN when implemented in low-resolution remote sensing imagery. We use EESRGAN, EEGAN, and ESRGAN as the super-resolution network with the detectors. We show that our EESRGAN with the detectors performs better than the other methods. We also show that the edge-enhancement helps to improve the detection accuracy. The AP improvement is higher in high IoUs and comparatively less improvement in low IoUs. We also show that the precision increases with the increase in resolution. Improvement of AP values for the OGST dataset is lower than for the COWC dataset because the area covered by a tank is slightly bigger than a car, and tanks sizes and colors are less diverse than cars.

Our experimental results indicate that AP values of the output can be improved slightly with the increase of training data. The results also demonstrate that we can use less training data for both the datasets to get a similar level of accuracy that we get from our total training data.

We have large numbers of cars from different regions in the COWC dataset, and we get high AP values using different IoUs. On the other hand, the OGST dataset needs more data to get a general

 doi:10.20944/preprints202003.0313.v2

detection result because we use data from a specific area and for a specific season. Therefore, more data from different regions and seasons would make our method more robust for the use-case of oil and gas storage tank detection.

The faster R-CNN detector gives us the best result, but it takes more time than an SSD detector. If we need detection results from a vast area, then SSD would be the right choice sacrificing some amount of accuracy.

We use low-high resolution image pairs to train our architecture, and the low-resolution images are generated artificially from the high-resolution counterparts. To our knowledge, there is no suitable public satellite dataset that contains both high-low resolution image pairs (real) and ground truth bounding boxes for detecting small objects. Therefore, we create the low-resolution images which do not precisely correspond to low-resolution images. However, improvement of resolution through deep learning always improves object detection performance on remote sensing images (for both artificial and real low-resolution images), as discussed in the introduction and related works section of this paper [5]. There are some impressive works [61,71] to create realistic low-resolution images from high-resolution images. For future work, we are looking forward to exploring the works to create more accurate low-resolution images for training.

6. Conclusions

In this paper, we propose an end-to-end architecture that takes low-resolution satellite imagery as input and gives object detection results as outputs. Our architecture contains a super-resolution network and a detector network. We use a different combination of super-resolution systems and detectors to compare the AP values for detection using two different datasets. The experimental results show that our proposed super-resolution network with faster R-CNN yields the best results for small objects on satellite imagery. However, we need to add more diverse training data in the OGST dataset to make our model robust in detecting oil and gas storage tanks. We also need to explore the techniques to create more realistic low-resolution images. In conclusion, our method combines different strategies to provide a better solution to the task of small-object detection on low-resolution imagery.

Author Contributions: Conceptualization, J.R., N.R. and M.S.; methodology, J.R., N.R. and M.S.; software, J.R.; validation, J.R.; formal analysis, J.R.; investigation, J.R.; resources, N.R.; data curation, J.R., S.C. and D.C.; writing–original draft preparation, J.R.; writing–review and editing, J.R., N.R., M.S., S.C. and D.C.; visualization, J.R.; supervision, N.R. and M.S.; project administration, N.R.; funding acquisition, N.R., S.C. and D.C.

Funding: This research was partially supported by Alberta Geological Survey (AGS) and NSERC discovery grant.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this paper:

SRCNN	Single image Super-Resolution Convolutional Neural Network
VDSR	Very Deep Convolutional Networks
GAN	Generative Adversarial Network
SRGAN	Super-Resolution Generative Adversarial Network
ESRGAN	Enhanced Super-Resolution Generative Adversarial Network
EEGAN	Edge-Enhanced Generative Adversarial Network
EESRGAN	Edge-Enhanced Super-Resolution Generative Adversarial Network
RRDB	Residual-in-Residual Dense Blocks
EEN	Edge Enhancement Network
SSD	Single-Shot multi-box Detector
YOLO	You Only Look Once
CNN	Convolutional Neural Network
R-CNN	Region-based Convolutional Neural Network
FRCNN	Faster Region-based Convolutional Neural Network
VGG	Visual Geometry Group
BN	Batch Normalization
MSCOCO	Microsoft Common Objects in Context

OGST	Oil and Gas Storage Tank
COWC	Car Overhead With Context
GSD	Ground Sampling Distance
G	Generator
D	Discriminator
ISR	Intermediate Super-Resolution
SR	Super-Resolution
HR	High-Resolution
LR	Low-Resoluton
FPN	Feature Pyramid Network
RPN	Region Proposal Network
AER	Alberta Energy Regulator
AGS	Alberta Geological Survey
AP	Average Precision
IoU	Intersection over Union
TP	True Positive
FP	False Positive
FN	False Negative

References

1. Colomina, I.; Molina, P. Unmanned aerial systems for photogrammetry and remote sensing: A review. *ISPRS Journal of photogrammetry and remote sensing* **2014**, *92*, 79–97.
2. Zhang, F.; Du, B.; Zhang, L.; Xu, M. Weakly supervised learning based on coupled convolutional neural networks for aircraft detection. *IEEE Transactions on Geoscience and Remote Sensing* **2016**, *54*, 5553–5563.
3. Fromm, M.; Schubert, M.; Castilla, G.; Linke, J.; McDermid, G. Automated Detection of Conifer Seedlings in Drone Imagery Using Convolutional Neural Networks. *Remote Sensing* **2019**, *11*, 2585.
4. Pang, J.; Li, C.; Shi, J.; Xu, Z.; Feng, H. $\mathcal{R}^2$ -CNN: Fast Tiny Object Detection in Large-Scale Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing* **2019**, *57*, 5512–5524. doi:10.1109/TGRS.2019.2899955.
5. Shermeyer, J.; Van Etten, A. The effects of super-resolution on object detection performance in satellite imagery. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2019, pp. 0–10.
6. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A.C.; Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge, 2014, [arXiv:cs.CV/1409.0575].
7. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. European conference on computer vision. Springer, 2014, pp. 740–755.
8. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2017**, *39*, 1137–1149. doi:10.1109/tpami.2016.2577031.
9. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollar, P. Focal Loss for Dense Object Detection. *2017 IEEE International Conference on Computer Vision (ICCV)* **2017**. doi:10.1109/iccv.2017.324.
10. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. *Lecture Notes in Computer Science* **2016**, p. 21–37. doi:10.1007/978-3-319-46448-0_2.
11. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **2016**. doi:10.1109/cvpr.2016.91.
12. Ji, H.; Gao, Z.; Mei, T.; Ramesh, B. Vehicle Detection in Remote Sensing Images Leveraging on Simultaneous Super-Resolution. *IEEE Geoscience and Remote Sensing Letters* **2019**, pp. 1–5. doi:10.1109/LGRS.2019.2930308.
13. Tayara, H.; Soo, K.G.; Chong, K.T. Vehicle detection and counting in high-resolution aerial images using convolutional regression neural network. *IEEE Access* **2017**, *6*, 2220–2230.
14. Yu, X.; Shi, Z. Vehicle detection in remote sensing imagery based on salient information and local shape feature. *Optik-International Journal for Light and Electron Optics* **2015**, *126*, 2485–2490.
15. Stankov, K.; He, D.C. Detection of buildings in multispectral very high spatial resolution images using the percentage occupancy hit-or-miss transform. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **2014**, *7*, 4069–4080.

16. Ok, A.O.; Başeski, E. Circular oil tank detection from panchromatic satellite images: A new automated approach. *IEEE Geoscience and Remote Sensing Letters* **2015**, *12*, 1347–1351.
17. Dong, C.; Loy, C.C.; He, K.; Tang, X. Image Super-Resolution Using Deep Convolutional Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2016**, *38*, 295–307. doi:10.1109/tpami.2015.2439281.
18. Kim, J.; Lee, J.K.; Lee, K.M. Accurate Image Super-Resolution Using Very Deep Convolutional Networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **2016**. doi:10.1109/cvpr.2016.182.
19. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. Advances in neural information processing systems, 2014, pp. 2672–2680.
20. Ledig, C.; Theis, L.; Huszar, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; et al.. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **2017**. doi:10.1109/cvpr.2017.19.
21. Wang, X.; Yu, K.; Wu, S.; Gu, J.; Liu, Y.; Dong, C.; Qiao, Y.; Loy, C.C. ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. *Computer Vision – ECCV 2018 Workshops* **2019**, p. 63–79. doi:10.1007/978-3-030-11021-5_5.
22. Jiang, K.; Wang, Z.; Yi, P.; Wang, G.; Lu, T.; Jiang, J. Edge-Enhanced GAN for Remote Sensing Image Superresolution. *IEEE Transactions on Geoscience and Remote Sensing* **2019**, *57*, 5799–5812. doi:10.1109/TGRS.2019.2902431.
23. Jiang, J.; Ma, J.; Wang, Z.; Chen, C.; Liu, X. Hyperspectral Image Classification in the Presence of Noisy Labels. *IEEE Transactions on Geoscience and Remote Sensing* **2019**, *57*, 851–865. doi:10.1109/TGRS.2018.2861992.
24. Tong, F.; Tong, H.; Jiang, J.; Zhang, Y. Multiscale union regions adaptive sparse representation for hyperspectral image classification. *Remote Sensing* **2017**, *9*, 872.
25. Zhan, C.; Duan, X.; Xu, S.; Song, Z.; Luo, M. An improved moving object detection algorithm based on frame difference and edge detection. Fourth International Conference on Image and Graphics (ICIG 2007). IEEE, 2007, pp. 519–523.
26. Mao, Q.; Wang, S.; Wang, S.; Zhang, X.; Ma, S. Enhanced image decoding via edge-preserving generative adversarial networks. 2018 IEEE International Conference on Multimedia and Expo (ICME). IEEE, 2018, pp. 1–6.
27. Yang, W.; Feng, J.; Yang, J.; Zhao, F.; Liu, J.; Guo, Z.; Yan, S. Deep Edge Guided Recurrent Residual Learning for Image Super-Resolution. *IEEE Transactions on Image Processing* **2017**, *26*, 5895–5907. doi:10.1109/tip.2017.2750403.
28. Kamgar-Parsi, B.; Kamgar-Parsi, B.; Rosenfeld, A. Optimally isotropic Laplacian operator. *IEEE Transactions on Image Processing* **1999**, *8*, 1467–1472. doi:10.1109/83.791975.
29. Landsat 8. https://www.usgs.gov/land-resources/nli/landsat/landsat-8. Accessed: 2020-02-11.
30. Sentinel-2. http://www.esa.int/Applications/Observing_the_Earth/Copernicus/Sentinel-2. Accessed: 2020-02-11.
31. Mundhenk, T.N.; Konjevod, G.; Sakla, W.A.; Boakye, K. A large contextual dataset for classification, detection and counting of cars with deep learning. European Conference on Computer Vision. Springer, 2016, pp. 785–800.
32. Rabbi, J.; Chowdhury, S.; Chao, D. Oil and Gas Tank Dataset. Mendeley Data, V3, 2020. doi:10.17632/bkxj8z84m9.3.
33. Jolicoeur-Martineau, A. The relativistic discriminator: a key element missing from standard GAN, 2018, [arXiv:cs.LG/1807.00734].
34. Charbonnier, P.; Blanc-Féraud, L.; Aubert, G.; Barlaud, M. Two deterministic half-quadratic regularization algorithms for computed imaging. *Proceedings of 1st International Conference on Image Processing* **1994**, *2*, 168–172 vol.2.
35. Alberta Energy Regulator. https://www.aer.ca. Accessed: 2020-02-05.
36. Tai, Y.; Yang, J.; Liu, X.; Xu, C. MemNet: A Persistent Memory Network for Image Restoration. *2017 IEEE International Conference on Computer Vision (ICCV)* **2017**. doi:10.1109/iccv.2017.486.
37. Zhang, Y.; Tian, Y.; Kong, Y.; Zhong, B.; Fu, Y. Residual Dense Network for Image Super-Resolution. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* **2018**. doi:10.1109/cvpr.2018.00262.

38. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *2015 IEEE International Conference on Computer Vision (ICCV)* **2015**. doi:10.1109/iccv.2015.123.
39. Lim, B.; Son, S.; Kim, H.; Nah, S.; Lee, K.M. Enhanced Deep Residual Networks for Single Image Super-Resolution. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* **2017**. doi:10.1109/cvprw.2017.151.
40. Liebel, L.; Körner, M. Single-image super resolution for multispectral remote sensing data using convolutional neural networks. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **2016**, *41*, 883–890.
41. Tayara, H.; Chong, K. Object detection in very high-resolution aerial images using one-stage densely connected feature pyramid network. *Sensors* **2018**, *18*, 3341.
42. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition* **2014**. doi:10.1109/cvpr.2014.81.
43. Girshick, R. Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV)* **2015**. doi:10.1109/iccv.2015.169.
44. Li, Q.; Mou, L.; Xu, Q.; Zhang, Y.; Zhu, X.X. R3-Net: A Deep Network for Multioriented Vehicle Detection in Aerial Images and Videos. *IEEE Transactions on Geoscience and Remote Sensing* **2019**, *57*, 5028–5042. doi:10.1109/tgrs.2019.2895362.
45. Ammour, N.; Alhichri, H.; Bazi, Y.; Benjdira, B.; Alajlan, N.; Zuair, M. Deep learning approach for car detection in UAV imagery. *Remote Sensing* **2017**, *9*, 312.
46. Ren, Y.; Zhu, C.; Xiao, S. Small object detection in optical remote sensing images via modified faster R-CNN. *Applied Sciences* **2018**, *8*, 813.
47. Tang, T.; Zhou, S.; Deng, Z.; Zou, H.; Lei, L. Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining. *Sensors* **2017**, *17*, 336.
48. Chen, Z.; Zhang, T.; Ouyang, C. End-to-end airplane detection using transfer learning in remote sensing images. *Remote Sensing* **2018**, *10*, 139.
49. Radovic, M.; Adarkwa, O.; Wang, Q. Object recognition in aerial images using convolutional neural networks. *Journal of Imaging* **2017**, *3*, 21.
50. Li, W.; Fu, H.; Yu, L.; Cracknell, A. Deep learning based oil palm tree detection and counting for high-resolution remote sensing images. *Remote Sensing* **2017**, *9*, 22.
51. Lin, T.Y.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **2017**. doi:10.1109/cvpr.2017.106.
52. Liu, S.; Huang, D.; others. Receptive field block net for accurate and fast object detection. Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 385–400.
53. Zhang, S.; Wen, L.; Bian, X.; Lei, Z.; Li, S.Z. Single-shot refinement neural network for object detection. Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4203–4212.
54. Li, Z.; Zhou, F. FSSD: feature fusion single shot multibox detector. *arXiv preprint arXiv:1712.00960* **2017**.
55. Zhu, R.; Zhang, S.; Wang, X.; Wen, L.; Shi, H.; Bo, L.; Mei, T. ScratchDet: Training single-shot object detectors from scratch. Proceedings of the IEEE conference on computer vision and pattern recognition, 2019, pp. 2268–2277.
56. Yang, X.; Sun, H.; Fu, K.; Yang, J.; Sun, X.; Yan, M.; Guo, Z. Automatic ship detection in remote sensing images from google earth of complex scenes based on multiscale rotation dense feature pyramid networks. *Remote Sensing* **2018**, *10*, 132.
57. 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* **2019**, *30*, 3212–3232.
58. Li, K.; Wan, G.; Cheng, G.; Meng, L.; Han, J. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS Journal of Photogrammetry and Remote Sensing* **2020**, *159*, 296–307.
59. Bai, Y.; Zhang, Y.; Ding, M.; Ghanem, B. Sod-mtgan: Small object detection via multi-task generative adversarial network. Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 206–221.

60. Haris, M.; Shakhnarovich, G.; Ukita, N. Task-driven super resolution: Object detection in low-resolution images. *arXiv preprint arXiv:1803.11316* **2018**.
61. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. Proceedings of the IEEE international conference on computer vision, 2017, pp. 2223–2232.
62. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition, 2014, [arXiv:cs.CV/1409.1556].
63. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *2015 IEEE International Conference on Computer Vision (ICCV)* **2015**. doi:10.1109/iccv.2015.123.
64. Lai, W.S.; Huang, J.B.; Ahuja, N.; Yang, M.H. Deep Laplacian Pyramid Networks for Fast and Accurate Super-Resolution. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **2017**. doi:10.1109/cvpr.2017.618.
65. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**.
66. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché Buc, F.; Fox, E.; Garnett, R., Eds.; Curran Associates, Inc., 2019; pp. 8024–8035.
67. Rabbi, J. Edge Enhanced GAN with Faster RCNN for end-to-end object detection from remote sensing imagery. https://github.com/Jakaria08/Filter_Enhance_Detect, 2020.
68. Alberta Geological Survey. https://ags.aer.ca. Accessed: 2020-02-05.
69. Chowdhury, S.; Chao, D.K.; Shipman, T.C.; Wulder, M.A. Utilization of Landsat data to quantify land-use and land-cover changes related to oil and gas activities in West-Central Alberta from 2005 to 2013. *GIScience & Remote Sensing* **2017**, *54*, 700–720.
70. Bing Map. https://www.bing.com/maps. Accessed: 2020-02-05.
71. Bulat, A.; Yang, J.; Tzimiropoulos, G. To learn image super-resolution, use a gan to learn how to do image degradation first. Proceedings of the European conference on computer vision (ECCV), 2018, pp. 185–200.