SHIP DETECTION FROM SAR REMOTE SENSING DATA USING CONVOLUTION NEURAL NETWORK

Balazs Krupinszki (s212902), Abdalmenem Owda (s210007), Ole Winther

DTU Compute, Technical University of Denmark

ABSTRACT

With the advent of satellite Synthetic Aperture Radar (SAR) data and its free-online availability for researchers and even industry, that enriches many applications such as maritime applications. The continuous and fast growth of SAR data is a huge potential to introduce deep learning in SAR data processing. In this research paper, we introduce three different Regional- Convolution Neural Networks (R-CNN), which has been used recently as object detectors, test the models with public SAR ship dataset, compare their performance based on common evaluation metrics and apply the model on random unlabeled dataset. With this study, we hope to exploit the best accurate model in the coastal applications to detect ships from SAR data. That is a major issue for offshore wind energy applications. Furthermore, stimulate more research either to improve the best model or develop our new model detector. The project code can be found on GitHub.

***Index Terms*—** SAR, ship, object detection, R-CNN, Fast R-CNN, Faster R-CNN

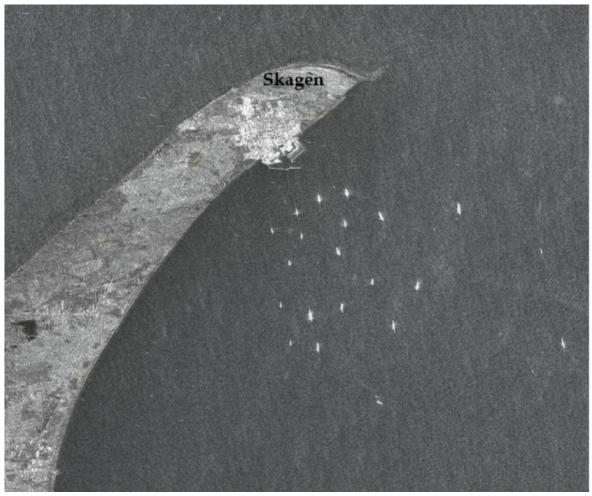

Fig. 1: SAR Sentinel 1B taken on 19/12/2021 at 16:35:51local time. The high brightness pixels refer most probably to ships

1. INTRODUCTION

Synthetic Aperture Radar (SAR) is a side-looking sensor equipped on a moving platform called "Satellite", transmits and receives signals in microwave frequencies. SAR systems are unique for many reasons, not to mention, acquisition of data is independent of weather conditions, cloud coverage, atmospheric parameters and able to work day-night time. Man-made objects,e.g, ships on oceans are visible in SAR data due to high backscattered signals "single or double bouncing scattering". In other words, the ratio of received to transmitted signal for metallic objects is relatively higher than sea surface roughness at moderate wind speed. Subsequently, the ships are visible in the scenes. The backscattered signals are recorded as Normalized Radar Cross Section (NRCS) values. Each pixel in the scene has its own NRCS value which refers to one of scattering mechanisms (surface, volume,and single). Figure 1 shows a bunch of ships are distributed close to Skaven haven in north Denmark.

Numerous applications, like marine surveillance, require accurate information about ships in terms of numbers, locations, types of ships. Furthermore, other applications need this information to mask these pixels and leave them out for any further post processing. Wind speed can be estimated by inferring all pixel brightness in the scene using a geophysical model function (GMF) with auxiliary information about wind direction and geometry of acquisition.This process is called "SAR wind retrieval". Existence of ships in SAR is a big obstacle for accurate wind speed measurements. Due to high brightness values of the ships, leads to overestimation to the wind speed values in the region where ships exist. Deep learning (DL) has met SAR in different fields and proves its robustness for object detection and classification. Several detectors have been proposed recently to detect objects (e.g, Region-Based CNN (R-CNN)[1], Fast R-CNN[2], Faster R-CNN[3] and many others. This project aims to exploit the previous mentioned models to predict ships from the SAR scenes, compare the used models based on their detection performance on a subset and propose ideas for future works.

2. DATA

Several SAR ship datasets have been published and collected for many different uses with a variety of image sizes and additional data. One of the biggest datasets available - Sar-Ship-Dataset[4] - seemed promising for our application. Not only does it contain large amounts of ship instances (59535 instances in 43819 images), but the relatively small image sizes (256 $\times$ 256 pixels) are also well suited for our use without any transformations needed.

The dataset is constructed from about 102 Chinese Gaofen-3 and 108 Sentinel-1 A/B images. The dataset is publicly available and can be downloaded from GitHub (latest accessed: 27/12/2021 6 pm). In this project, we have used a small subset of about 5000 images). To make the data applicable for the model, the dataset had to be preprocessed first. Image names, image sizes, and bounding boxes had to be extracted from the individual text files and collected in a common csv for further use.

3. THEORY

This section aims to break-down all the three used R-CNN modes, all these models are two-stage algorithms, in this project. Furthermore, it will explain further about the evaluation metrics.

3.1. R-CNN

R-CNN model is an object detector which consists mainly of a two-stage detection algorithm. The first stage aims to identify subsets of regions in an image that might contain an object using a dedicated region proposal algorithm such EdgeBoxes[5]. After that, the proposed regions (Region of Interest - ROIs) are cropped and resized to feed them in CNN. The second stage classifies the objects in each region based on the support vector machine (SVM) and the regressors are adopted to calculate the localization offset of the bounding boxes and refine the results . Figure 2 depicts the structure and the stages of the R-CNN detector.

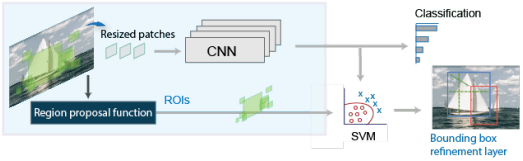

Fig. 2: Fast R-CNN flowchart taken from `https://se.mathworks.com/`

3.2. Fast R-CNN

It is similar to the R-CNN model and has the same concept of creating ROIs using a dedicated region proposal function. Unlike the R-CNN, it has less redundant computation and can be trained end to end. It first generates the shared feature map on the entire image and then pools CNN features from this shared map corresponding to each region proposal, instead of creating a feature map for every cropped ROI. Thanks to this optimization, it is much more efficient, as computations for overlapping regions are shared. Fast R-CNN can be up to 213* times faster at test-time than R-CNN [ref].

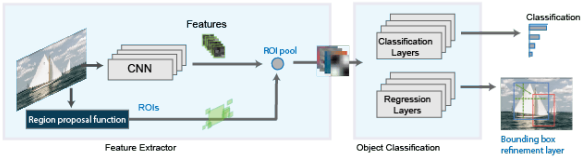

Fig. 3: R-CNN flowchart taken from `https://se.mathworks.com/`

3.3. Faster R-CNN

The Faster R-CNN shares a lot of it's arhitecture with the Fast R-CNN detector. The main difference is that it uses its own region proposal network (RPN) to generate ROIs directly instead of relying on an external algorithm like EdgeBoxes. The RPN utilizes the shared feature map generated by the CNN to make the region proposals, thus it is faster than using an external region proposal function. After the RPN generates the ROIs, they are cropped out of the shared feature map just like in the Fast R-CNN model. These cropped and resized features maps are then fed into the fully connected layer to classify the ROIs and to refine the bounding box positions.

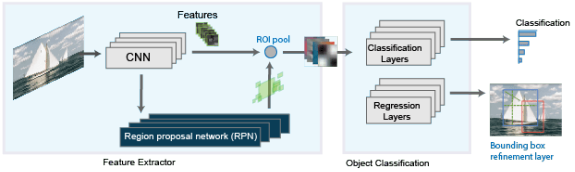

Fig. 4: Faster R-CNN flowchart taken from `https://se.mathworks.com/`

3.4. Evaluation Metrics

3.4.1. Intersection over union (IoU)

IoU is an evaluation metric used to measure the overlap between two bounding boxes (the ground truth and predicted). Specifically, IoU is given by the overlapping area of two bounding boxes divided by their union area (see figure 5 for more details).

Predefined IoU threshold value (usually between 0.5 - 0.95) used to classify the detection to True Positive (TP), False Negative(FN) indicates the ground truth is not detected,

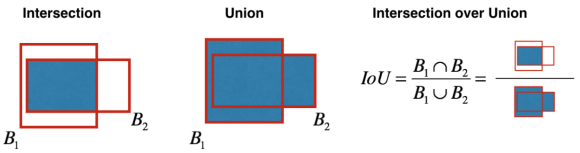

Fig. 5: Intersection over union calculation between two boxes.

False Positive (FP) indicates greater than threshold value, and True Negative (TN). Precision (P) and Recall (R) are used as common standard for binary classification and can be defined by the following equation:

$$P = \frac{TP}{TP + FP}$$
$$R = \frac{TP}{TP + FN}$$

As much as the precision stays high, the threshold value is changed and recall is increased; it is considered as a good detector for "precision-recall curve".

3.4.2. Average precision and mean average precision

Average precision (AP) is a metric that summarizes the precision-recall curve for detectors and it is the area under the precision-recall curve. In this project, we have only one class which is ship, then the AP can be defined as:

$$AP = \int P(R)dR$$

4. MODEL

We settled on utilizing the Faster R-CNN detector for our task, due to the advantages it provides over the other two models. A typical Faster R-CNN model can be broken down into four main parts:

- Backbone CNN that creates the shared feature map
- Region Proposal Network
- ROI pooling
- Fully connected classifier layer

However, several implementations of the Faster R-CNN model is possible depending on the CNN architecture used as the backbone model for feature map generation. We utilized 3 different (pre-trained) implementations of the Faster R-CNN detector:

- ResNet50 FPN: A 50 layer deep CNN with a Feature Pyramid Network (FPN) [6]
- MobileNetV3 Large [7] FPN: A deep and fast CNN with an FPN, optimized for slower hardware. Competitive accuracy compared to ResNet50, but faster execution.
- MobileNetV3 Large 320 FPN: An iteration of the MobileNetV3-Large FPN that uses reduced resolution, thus sacrifices accuracy for speed.

All three implementations utilize an FPN (Feature Pyramid Network) as well, which is a feature extractor that takes a single-scale image of an arbitrary size as input, and outputs proportionally sized feature maps at multiple levels, in a fully convolutional fashion. This process is independent of the backbone CNN and the multi-scale feature maps it generates usually have better quality information than the regular feature extraction, thus increasing the performance.

5. RESULTS

All three models were trained using a subset of 5000 images, for 3 epochs. Further training of these models is greatly encouraged, thus the trained models are publically available on GitHub.

5.1. Validation output

Region proposals of the three different architectures after 3 epochs of training can be seen in figure 6, 7, and 8. The green boxes represent the regions predicted by the models, while the red boxes represent the ground truth (target) bounding boxes. The thickness of the predicted bounding boxes depend on the confidence score for a specific ROI. If the score of a predicted bounding box is greater than 0.9, then the visualized box has thick outline, otherwise it has a thinner outline.

We can see that both the Resnet50 and MobileNetV3 models predict the 4 ship instances with great accuracy, but also produces a false positive box, although with a small confidence score.
The smallest model (MobileNetV3 320), which is much faster than the other two, produces more false positives, and also produces multiple boxes for single ship instances, thus the performance was sacrificed for speed.

5.2. Performance

After training all three models we calculated the mAP@[.5:.95] and AR (average recall) evaluation metrics on the validation dataset. The mAP@[.5:.95] metric refers to the average mAP (mean average precision) over different IoU (intersection over union) thresholds, from 0.5 to 0.95, with steps of 0.05. Table 1 shows the performance of the proposed models.

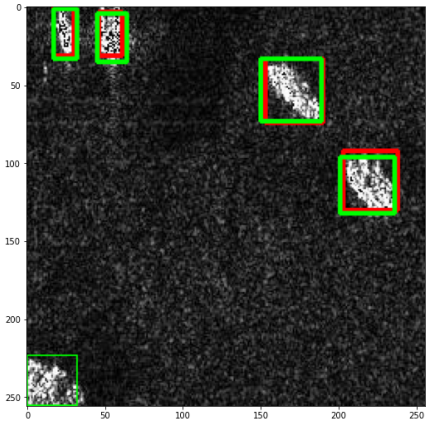

Fig. 6: ResNet50

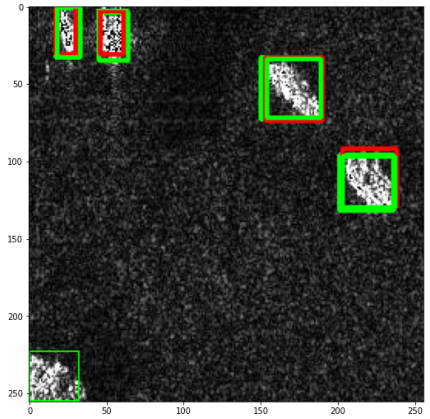

Fig. 7: MobileNetV3 Large

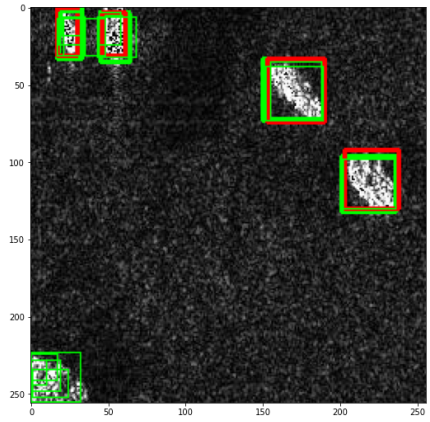

Fig. 8: MobileNetV3 Large 320

All three architectures yield great results considering the low number of epochs and should be considered for further optimization. While the two bigger models (ResNet and MobileNetV3 Large) might perform slightly better, the execution speed of the MobileNetV3 Large 320 can prove valuable in some applications.

Model	mAP@[.5:.95]	AR
ResNet50 FPN	0.872	0.810
MobileNetV3 Large FPN	0.898	0.762
MobileNetV3 Large 320 FPN	0.785	0.680

Table 1: Performance of the three architectures

6. APPLICATION

We used the model with the ResNet50 backbone to predict and mark the possible ship instances on a big (2560 x 2560 pixels) unlabelled image. To achieve this, we first split the big image up into smaller, 256x256 pixel subimages, which are then labelled by the network. Then we concat the output of these together to get the final output for the big image.
The final output of the model for a coastal SAR image can be seen in Figure 9.

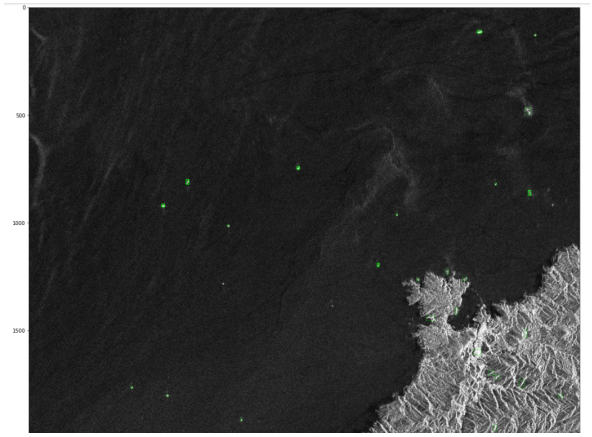

Fig. 9: Predictions of the ResNet50 model on a SAR image.

7. CONCLUSION

Ships are distributed randomly in SAR scenes with different densities in onshore and offshore areas. Subsequently, the existence of high brightness pixels (ships) are a major challenge in our PhD project. So far, it has become clear to us how to use the best architecture model in our main application. Detecting the locations of ships in SAR scenes and leaving them out of our wind retrieval process are the major accomplishment and will definitely improve the accuracy of wind speed by skipping the anomalous pixels.

Three different R-CNN models were proposed and tested to detect ships (high brightness pixels) in SAR images. The results in terms of mAP and AR were promising for the three different models. In regard to mAP, MobileNetV3 Large FPN performs slightly better than ResNet 50 and Large 320 FPN. Nevertheless, the AR of ResNet 50 is significantly

higher than the other two models. The smallest model (MobileNetV3 Large 320) performed worse than the other two, but yielded great improvement in running times, thus should be considered for application where execution speed is important. All three architectures performed well in our task considering the small number of epochs (3) and size of the dataset (subset of 5000 images) and should be considered for further optimization and improvements.

This study opens a new horizon for us in regard to how to improve these models and think beyond our main target to improve or develop new detector models based on our understanding, the strength and weakness of each model.

8. REFERENCES

References

[1] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," pp. 580–587, 2014.

[2] Ross Girshick, "Fast r-cnn," 2015.

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[4] Yuanyuan Wang, Chao Wang, Hong Zhang, Yingbo Dong, and Sisi Wei, "A sar dataset of ship detection for deep learning under complex backgrounds," *Remote Sensing*, vol. 11, no. 7, 2019.

[5] C. Lawrence Zitnick and Piotr Dollár, "Edge boxes: Locating object proposals from edges," in *Computer Vision – ECCV 2014*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, Eds., Cham, 2014, pp. 391–405, Springer International Publishing.

[6] Tsung-Yi Lin, Piotr DollÃ¡r, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie, "Feature pyramid networks for object detection," 2017.

[7] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam, "Searching for mobilenetv3," 2019.