Small Vessel Detection from Synthetic Aperture Radar (SAR) Imagery using Deep Learning
(Application Project - Computer Vision)

Jake Taylor
jakee417@stanford.edu

Toktam Mohammadnejad
toktammd@stanford.edu

1 Introduction

1.1 Description

Synthetic aperture radar (SAR) provides high-resolution, all-day, all-weather satellite imagery, which has become one of the most important means for high-resolution ocean observation and is well suited to better understand the maritime domain. We use the LS-SSDD-v1.0 open source SAR dataset to build and train a computer vision small vessel detection model which automatically generates bounding boxes around maritime vessels [1]. This type of automation would allow regulatory agencies to better conduct shipwreck rescue, fishery enforcement, and vessel traffic management. Our implementation is publicly available on Github. [1]

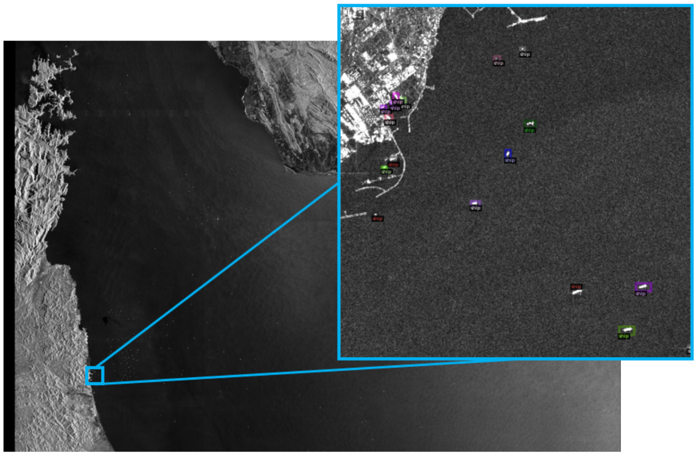

Figure 1: An example of large-scale image and sub-image from LS-SSDD-v1.0 SAR Imagery Dataset.

1.2 Challenges

Object detection from SAR imagery is challenging for a variety of reasons. Some ship targets and non-ship targets such as waves, dams, islands, icebergs, or reefs, have approximate backscattering intensity in SAR imagery that makes ship detection in SAR imagery difficult [2]. Moreover, many low-level and mid-level image features that have been widely used in object detection and classification applications cannot be introduced directly into ship detection via SAR imagery, which imposes an additional challenge. Lastly, detecting small objects in large-scale remote sensing images remains an unsolved problem within the computer vision literature.

[1] https://github.com/jakee417/LS-SSDD-v1.0-ShipDetectionComputerVision

1.3 Related Work

A survey of works related to **deep learning-based object detection** and specifically **small object detection** can be found in [3] and [4], respectively. For this project, we use the **dataset and benchmark results** given by [1], which has related research in [5; 6; 7]. **Data Augmentation for small object detection** approaches can be found in [8; 9].

2 Dataset

We use the Large-Scale SAR Ship Detection Dataset-v1.0 (LS-SSDD-v1.0) from Sentinel-1 [1]. LS-SSDD-v1.0 contains 15 large-scale 24,000 × 16,000 pixel, SAR images. The original large-scale SAR images are cut into 9000, 800×800 pixel sub-images shown in Figure 1. The authors use the first 10 of the 15, large-scale images as the training set (`train`). The last 5, large-scale images are used as the test set (`test`). `test` is further broken down into 2234 offshore sub-images (`test offshore`) and 766 inshore sub-images (`test inshore`). Differences between the datasets are shown below in Table 1 and Figure 2:

Dataset	# Imgs	# Ships	% Imgs w/ Ships	Ship/ *Img	Ships Pixel/ *Img Pixel
`train`	6000	3637	0.18	3.23	0.0016
`test`	3000	2378	0.24	3.23	0.0023
`test offshore`	2234	1495	0.27	2.41	0.0021
`test inshore`	766	883	0.15	7.54	0.0030

Table 1: LS-SSDD-v1.0 datasets. *Img denotes an image that has at least one ship.

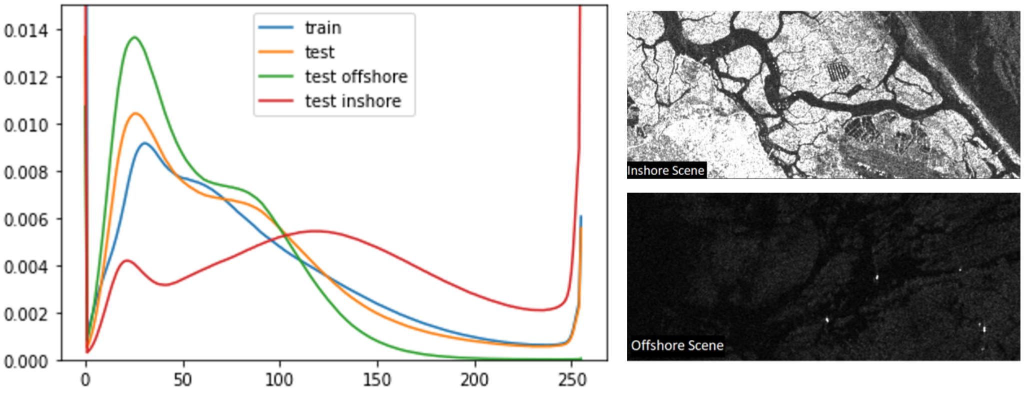

Figure 2: (Left) Sample of pixel intensity frequencies. (Right) Inshore and Offshore scenes.

3 Evaluation Metrics

For vessel detection, we use the score threshold of 0.5, intersection over union (IOU) threshold of 0.5, and non-maximum suppression (NMS) threshold of 0.5. To verify baseline model to [1], the following evaluation metrics are considered on `test`: Detection Probability (P_d), False Alarm (P_f), Missed Detection (P_m), Recall, Precision, Mean Average Precision (mAP), and F_1 score. To compare the baseline model results with further modelings, **only** mAP is used as a single evaluation metric.

4 Learning Methodology

4.1 Baseline Models

To simulate the results given in [1], we implemented two baseline models from Facebook's Detectron2 API [10; 11] . The two neural network architectures we used are (1) Faster Region-Based Convolutional Neural Network (Faster R-CNN) [12] and (2) RetinaNet [13], both pre-trained on the COCO dataset. For both of these models, we used ResNet-50 with Feature Pyramid Network (FPN) pre-trained on the ImageNet dataset as the backbone [14]. To ensure consistency with reference

[1], we trained on all of `train` (no validation dataset) using mini-batch gradient descent (MGD) with momentum for 12 epochs (36k iterations) and matching hyperparameters when possible given different APIs used. The comparison between Original Paper (OP) and Baseline Model (BM) results is shown in Table 2.

Dataset (Source)	Model	P_f	P_m	Recall	Precision	mAP	F_1
`test` (OP)	Faster R-CNN	26.26	22.29	77.71	73.74	74.80	0.76
`test offshore` (OP)	Faster R-CNN	17.18	8.09	91.91	82.82	89.99	0.87
`test inshore` (OP)	Faster R-CNN	44.04	46.32	53.68	55.96	46.76	0.59
`test` (OP)	RetinaNet	5.38	44.49	55.51	94.62	54.31	0.70
`test offshore` (OP)	RetinaNet	4.68	22.34	77.66	95.32	76.15	0.86
`test inshore` (OP)	RetinaNet	10.17	81.99	18.01	89.83	17.29	0.30
`test` (BM)	Faster R-CNN	26.00	25.23	74.76	73.99	71.32	0.74
`test offshore` (BM)	Faster R-CNN	18.06	8.36	91.63	81.93	89.00	0.86
`test inshore` (BM)	Faster R-CNN	44.18	53.79	46.20	55.81	38.64	0.50
`test` (BM)	RetinaNet	58.03	41.97	58.03	80.33	53.02	0.67
`test offshore` (BM)	RetinaNet	13.14	21.74	78.26	86.86	73.44	0.82
`test inshore` (BM)	RetinaNet	43.40	76.21	23.78	56.60	16.89	0.33

Table 2: Comparison between Original Paper (OP) and Baseline Model (BM) results.

When inference was conducted on `train`, a much lower level of performance was observed, which is summarized in Table 3. As reported in [1], when inference was conducted on `test offshore` and `test inshore`, we also observed that `test offshore` always has a much higher performance than `test inshore`, which is likely due to the absence of near-shore backscattering. Referring back to Table 1 and Figure 2, we have evidence to believe that `train` and `test` come from different distributions and that `train`'s distribution may be more similar with `test inshore`'s distribution. Thus, there is a need to introduce a new strategy to better estimate out-of-sample performance on `train`. For the rest of this paper, we will use `Baseline` to denote the Faster R-CNN baseline model.

Dataset	mAP @ 36k	mAP @ 72k	mAP @ 108k	mAP @ 144k
`train`	52.32	55.02	62.83	65.81
`test`	71.32	70.90	70.40	67.96

Table 3: `Baseline` performance on `train` versus `test` at various training iterations.

4.2 `Baseline` to `Improved`: Validation Set, Learning Rate Scheduling, and ResNext

To better estimate out-of-sample performance, we depart from [1] and adopt `train'` and `validation` datasets on an improved model called `Improved`. We first randomly shuffle `train` and then split 85% of `train` into `train'` and the remaining 15% of `train` into `validation`. `test` is kept unchanged so that `Improved` results are comparable with the `Baseline` and [1]. Another benefit from using `validation` is that we can use learning rate scheduling based on `validation` mAP. Specifically, we scale the learning rate by 0.1 if the current mAP score is within a relative threshold of 0.01 of the best mAP observed with one epoch patience. To address the issue shown in Table 3, we fit a Faster R-CNN with a ResNeXt-101-32x8d FPN backbone to benefit from its state of the art results [15]. A schematic of ResNeXt block architecture compared to ResNet block is shown in Figure 3.

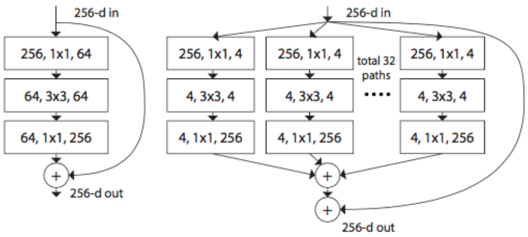

Figure 3: (Left) A ResNet block. (Right) A ResNeXt block with cardinality = 32.

4.3 Baseline to Improved: Data Augmentation

To avoid overfitting to train' on Improved, we incorporate standard data augmentation techniques such as; random rotation, brightness, contrast, lighting, and saturation. The other challenge in training small object detection models is ensuring that anchor boxes overlap with such small ground truth bounding boxes [8; 9]. To overcome this, we use a copy-paste procedure similar to [9], which augments train' with random copies of ship annotations:

1. For each annotation $\mathcal{A}$ in an image $\mathcal{I}$, make $\lceil C \rceil : C \sim U[0, \pi]$ copies of $\mathcal{A}$.
2. For $c \in C$, consider pasting c, $\lceil X \rceil : X \sim \mathcal{N}(\vec{0}, \sigma * I_2)$ pixels away from $\mathcal{A}$.
3. **Reject** c if: c intersects with another annotation $\mathcal{A}'$, the Paste-Mask, or is outside of $\mathcal{I}$.

We use Otsu's method, a histogram-thresholding algorithm, to create a Paste-Mask which prevents unrealistic paste locations from occurring [16]. Otsu's method excels for images that have areas of high contrast differences (containing land and sea) like those in test inshore. Although [9] copies templates of ships uniformly at random across an image, we found that simply copying each annotation to locations decided by a Gaussian Mixture Model (GMM) is sufficient to create a realistic augmented scene. We use $\pi = 5$ and $\sigma = 300$ on Improved. Examples of an original, masked, and augmented image are given in Figures 4 and 5.

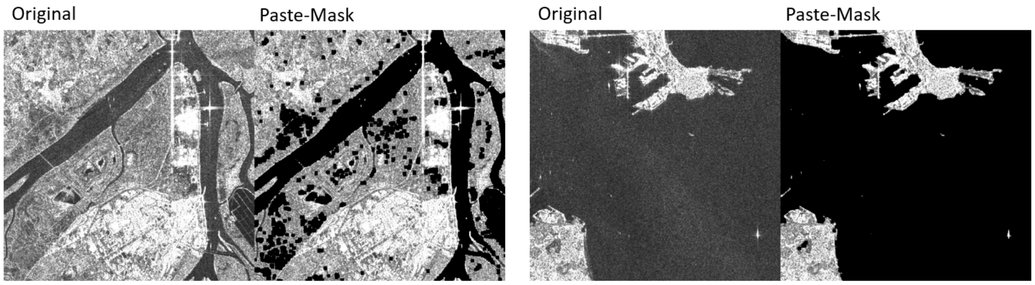

Figure 4: Examples of Paste-Mask. Black pixels show where legal pastes can occur.

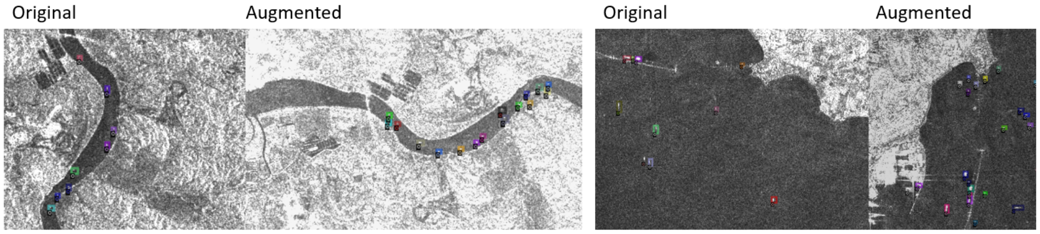

Figure 5: Examples of Copy and Paste Data Augmentation.

5 Results

By using train' and validation, now the model (Improved) reflects a better estimate of out-of-sample performance, which is shown in Table 4. Improved also shows higher model capacity and less signs of overfitting due to the learning rate scheduling and data augmentation.

Dataset	mAP @ 20k	mAP @ 25k	mAP @ 30k	mAP @ 35k
train'	56.83	56.90	56.65	57.13
validation	44.20	51.59	51.41	51.75

Table 4: Improved performance on train' versus validation at various training iterations.

Training curves on Improved for train' loss, validation loss, learning rate, and validation mAP are shown in Figure 6.

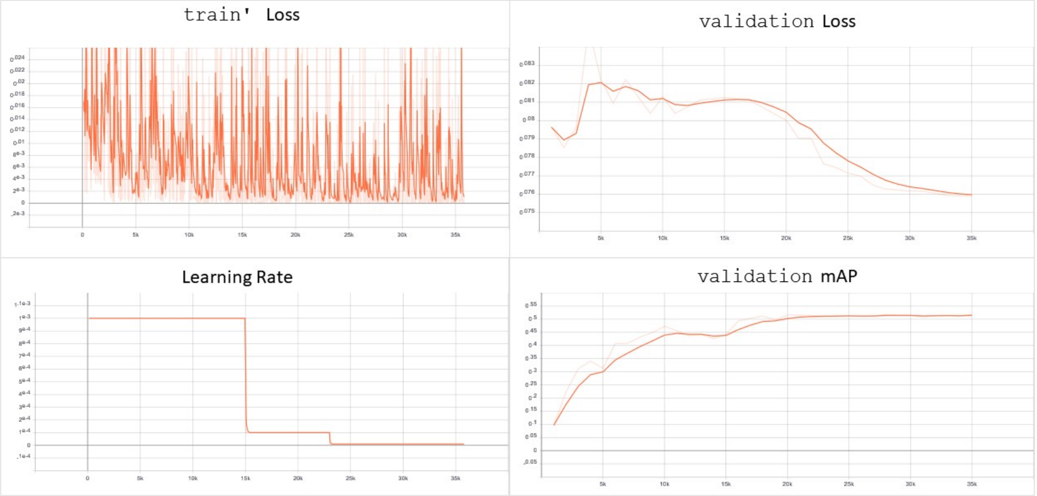

Figure 6: Training curves on Improved.

Finally, a comparison between Baseline and Improved based on mAP is given in Table 5. For completeness, we show results for Improved with and without data augmentation.

Dataset (Source)	Model	Data Augmentation	mAP
Validation	Baseline	No	44.87
Validation	Improved	No	46.67
Validation	Improved	Yes	51.75
test	Baseline	No	68.85
test	Improved	No	67.82
test	Improved	Yes	71.37
test inshore	Baseline	No	36.22
test inshore	Improved	No	36.40
test inshore	Improved	Yes	42.94

Table 5: Comparison between Baseline and Improved results.

A plot of various evaluation metrics and samples of output from Improved with data augmentation are included in Appendix A.

6 Discussion and Conclusion

Although there is an improvement on test inshore with Improved, we still observe a large difference in performance between the test inshore and test offshore. This observation is due to the fact that ships in inshore scenes are harder to detect than offshore scenes because of the interference of land and backscattering [2]. A better land masking strategy such as the methodology proposed in [17] or adding a bathymetry mask to the dataset [18] may help improve the performance of ship detection in inshore scenes. It appears that the combination of standard + copy-paste data augmentation as in Improved is an effective way for improving performance on test inshore. Despite these improvements, copy-paste data augmentation could also benefit from more accurate sea-land masking in future work. Lastly, the results on Improved further confirm that train and test do come from different distributions. Therefore, it is important to make sure that validation is drawn from the same distribution as test and that both reflect data we expect to get in the future.

7 Contribution

Both group members contributed equally to the project. We would like to thank CS230 Project TA Avoy Datta for project guidance throughout the quarter.

A `Improved` Output

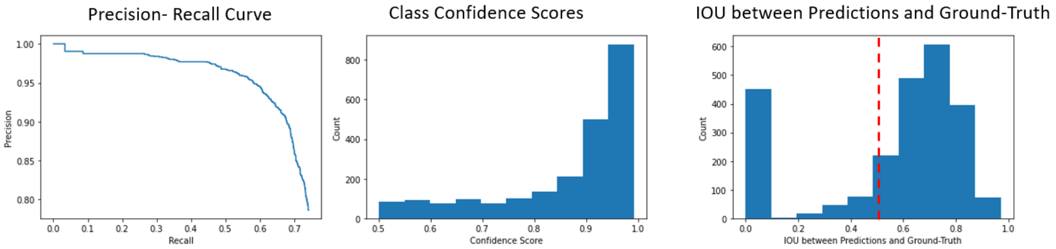

Figure 7: Precision-Recall Curve, Class Confidence Scores, and IOU between Predictions and Ground-Truth obtained from Inference on `test`.

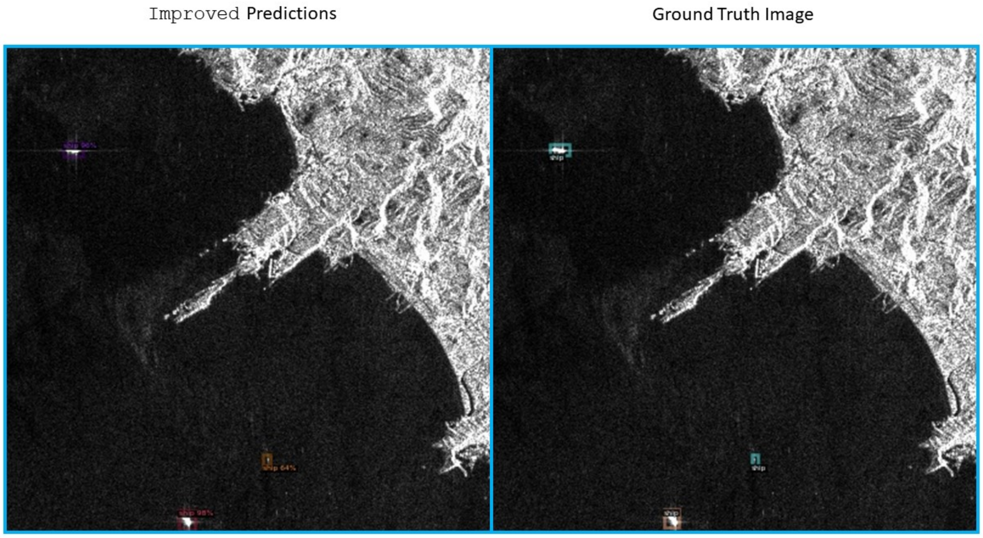

Figure 8: Comparison of Predictions and Ground Truth on `test inshore` without backscattering.

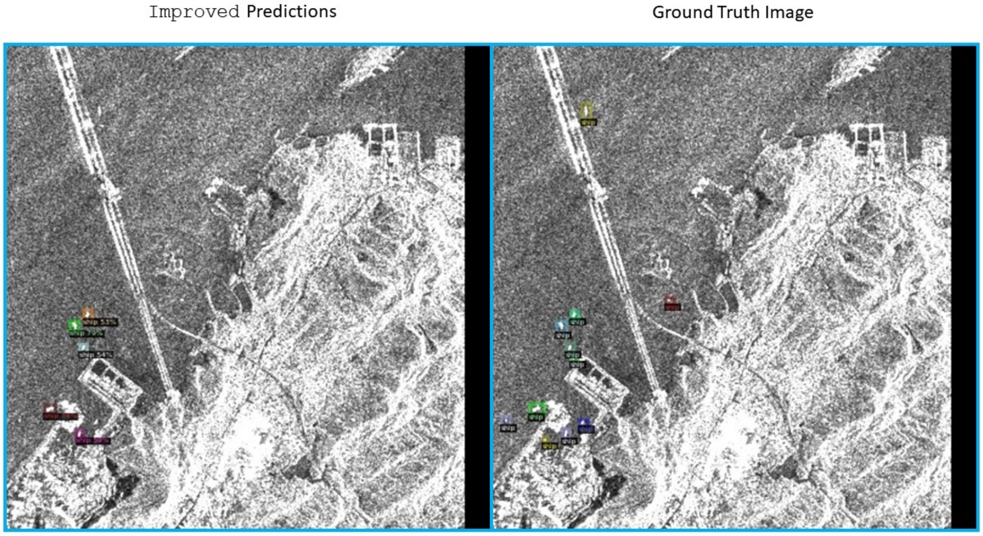

Figure 9: Comparison of Predictions and Ground Truth on `test inshore` with backscattering.

References

[1] T. Zhang, X. Zhang, X. Ke, X. Zhan, J. Shi, S. Wei, D. Pan, J. Li, H. Su, Y. Zhou, and D. Kumar, "Ls-ssdd-v1.0: A deep learning dataset dedicated to small ship detection from large-scale sentinel-1 sar images," *Remote Sensing*, 2020.

[2] "Sar images - principles of remote sensing." `https://crisp.nus.edu.sg/~research/tutorial/sar_int.htm`. Accessed: 2021-03-15.

[3] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *CoRR*, vol. abs/1907.09408, 2019.

[4] K. Tong, Y. Wu, and F. Zhou, "Recent advances in small object detection based on deep learning: A review," *Image and Vision Computing*, vol. 97, p. 103910, 2020.

[5] T. Zhang and X. Zhang, "High-speed ship detection in sar images based on a grid convolutional neural network," *Remote Sensing*, vol. 11(10), 2019.

[6] T. Zhang, X. Zhang, J. Shi, and S. Wei, "Hyperli-net: A hyper-light deep learning network for high-accurate and high-speed ship detection from synthetic aperture radar imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 167, pp. 123 – 153, 2020.

[7] T. Zhang and X. Zhang, "Shipdenet-20: An only 20 convolution layers and <1-mb lightweight sar ship detector," *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2020.

[8] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," *CoRR*, vol. abs/1902.07296, 2019.

[9] N. Mo and L. Yan, "Improved faster rcnn based on feature amplification and oversampling data augmentation for oriented vehicle detection in aerial images," *Remote Sensing*, vol. 12, no. 16, 2020.

[10] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2." `https://github.com/facebookresearch/detectron2`, 2019.

[11] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," 2020.

[12] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[13] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[15] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," 2017.

[16] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[17] Q. An, Z. Pan, and H. You, "Ship detection in gaofen-3 sar images based on sea clutter distribution analysis and deep convolutional neural network," *Sensors*, vol. 18, no. 2, 2018.

[18] T. Sagawa, Y. Yamashita, T. Okumura, and T. Yamanokuchi, "Satellite derived bathymetry using machine learning and multi-temporal satellite images," *Remote Sensing*, vol. 11, no. 10, 2019.

[19] Z. Cai and N. Vasconcelos, "Cascade R-CNN: delving into high quality object detection," *CoRR*, vol. abs/1712.00726, 2017.

[20] J. S. Z. X. H. F. Jiangmiao Pang, Cong Li, "R2-cnn: Fast tiny object detection in large-scale remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, pp. 5512 – 5524, 2019.

[21] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "Mmdetection: Open mmlab detection toolbox and benchmark," *CoRR*, vol. abs/1906.07155, 2019.

[22] B. Zoph, E. D. Cubuk, G. Ghiasi, T.-Y. Lin, J. Shlens, and Q. V. Le, "Learning data augmentation strategies for object detection," 2019.