Space Analytics Engine for Ship Detection

Christopher Hamilton
a1766121

Supervised by: Tat-Jun Chin

ABSTRACT

Ship detection in satellite images is a task that allows for surveillance and gathering information for defence or other purposes. With recent advancements in machine learning technology, it is possible to detect ships in the ocean from optical imagery or other sensors on satellites. A machine learning method called transfer learning is very beneficial to the training time and accuracy of machine learning models, and it is well explored. A machine learning framework called Mask R-CNN will be used for ship detection and segmentation due to its ability to conduct instance segmentation in images.

The ability to analyse images on satellites rather than on the ground will reduce the time taken to get results as well as allow processed data to be transferred directly to the end-user, rather than needing to be processed first on the ground. The use of model pruning will be explored to reduce the storage and memory requirements as well as the power consumed when running a machine learning model. This is essential to running machine learning models from smaller devices such as satellites, allowing for a reduced cost of space missions.

1 INTRODUCTION AND MOTIVATION

Satellites have traditionally been used as a method to gather data before transferring it to the ground for further processing and analysis. However, it may also be possible to perform this data analysis on-board the satellite that captures the information. This idea includes many benefits, such as improving the cost-benefit ratio of space missions to gather data and improve the time it takes to get results from such tasks. It could also be useful to update the model loaded onto the satellite for a new purpose. Currently, many machine learning models require substantial amounts of storage and memory. This, in turn, means that when the model is running inference, it consumes large amounts of power which is not always ideal for running the model on mobile or small devices. It will be explored if there is a reliable method to prune the model, reducing the storage and memory requirements and, in turn, the power consumption while still achieving good results.

With the rise of machine learning algorithms and frameworks to more accurately classify and segment images based on objects in them, it is now possible to use these algorithms to detect ships from satellite imagery. The location and movement of ships in the waters around a country is an essential piece of information to analyse for the defence of the country. The use of machine learning in this context would make it a lot easier to analyse this data and notice any activity of interest.

A machine learning model, as described, could be run from a satellite which also captures images to analyse. This would allow for the capture and analysis of images directly on the one system for information to be sent directly back to the end-user. This is possible with current technology, however, if the machine learning model that is based on the satellite was particularly deep or complex, it may require a very large amount of memory, storage, and power to run. For a satellite based in space, this may not be ideal due to limited resources. It will be investigated if a large and complex model such as this can be compressed to a form where it is still functional with similar accuracy, but also uses far less memory and power.

This paper will review the current research and literature in this area in section 2, outline the methodology used for the project in section 3, explain the experimental setup that was used in section 4, show the results in section 5, followed by the conclusions and potential further work that could be investigated. A link to the GitHub repository containing what is needed to reproduce the results can be found in section 6.

2 LITERATURE REVIEW

2.1 CURRENT RESEARCH DATASETS

The current research into the field of ship detection by neural networks mostly includes the use of optical imagery and SAR (Synthetic Aperture Radar) imagery. The use of transfer learning that is taking a model that has been pre-trained for a specific goal and using it to initialise the weights of the new model that will be trained is also well researched, with many papers exploring this idea. The difference between the SAR images and the optical images in the use of detecting ships is that SAR images are better visualisations for any time of day or even bad weather conditions. However, the SAR imagery does not include enough detail to classify the ships clearly and can also be ineffective for ships near each other or near ports. Optical imagery includes much more detail, but it is much less clear if the weather is cloudy.

The current method for SAR detection checks the pixel levels and searches for pixels that are brighter than their surroundings. This model is only suitable for the open sea as it cannot distinguish ships from other objects, due to the surroundings being at a similar brightness. (Li, Ding, Zhang, Wang, & Chen, 2019) Due to the lack of detail in the SAR based models, generally, they do not classify ships into different categories. This is mainly due to the lack of features on the ships themselves that would be needed to predict what type of ship it is accurately. Although, even in many optical image ship detection models, the ships are often only put into one category due to some difficulty in differentiating them. (Chen, Wu, Liu, & Zhang, 2020) The FGSD (Fine Grain Ship Detection) Dataset gathered from the Pattern Recognition and Intelligent System Lab and University of Posts and Telecommunications in Beijing, China, hopes to allow for better classification on ships depending on their type.

2.2 RESNET

In the development of the FGSD dataset, the researchers gathered thousands of images from Google Earth and created 43 classes of different ships and a class for the dock that a ship may be docked at or near. Once the photos were categorised, they had both horizontal and rotating bounding boxes put over the ships as annotations for the machine learning model. The horizontal bounding box was created using Faster-RCNN, and R2CNN created the rotating one. Their dataset ended up including 1917 images for the training dataset, 268 images for the validation set and 427 images for the test set. ResNet101 was used as the backbone for training and testing on this model due to its fast training time and accurate results. (Fung, 2017) ResNet101 is itself a very deep network, with 101 layers, making it highly accurate, and due to its idea of “identity shortcut connections,” it can still train quickly.

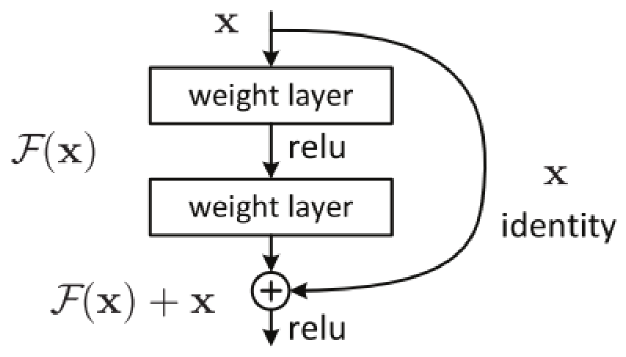
x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 1: An example of a residual block in ResNet (He, Zhang, Ren, & Sun, 2015)

The residual blocks that ResNet is made up of are included to reduce the problem that is caused by a large number of layers, that is, a vanishing gradient, “as the gradient is back-propagated to earlier layers, repeated multiplication may make the gradient infinitely small.” (Fung, 2017) Due to this vanishing gradient, if the network has many layers, it may start to degrade rapidly. However, the use of residual blocks in ResNet means that one or more layers can be skipped and then the element-wise sum of the original weights and the outputs of the residual layers is taken to get the residual block as shown in Figure 1. (He, Zhang, Ren, & Sun, 2015) These residual blocks are needed for the model to continue to operate accurately and with small error.

2.3 MASK R-CNN

A machine learning framework called Mask R-CNN can be used in many problems requiring instance segmentation. This process involves the task of object detection, where each object is located and bounded by a bounding box and the task of semantic segmentation in which each pixel of an object is classified as such. (He, Gkioxari, Dollár, & Girshick, 2017) Mask R-CNN extends the Faster R-CNN framework by including the use of masks for semantic segmentation.

Due to the instance segmentation used in Mask R-CNN, it is a desirable option for the problem of ship detection from satellite images, as the end user would likely want to have each instance of a ship detected as well as the pixels that make up the image selected. Current research already uses Mask R-CNN for this purpose for the reasons outlined. (Zhao & Li, 2020) One paper outlines the use of Mask R-CNN for the detection and targeting of ships from a video feed. The Mask R-CNN framework was compared with a YOLO framework, and their accuracies were tested. The results of those tests were that the Mask R-CNN framework was over two times more accurate while still being able to detect at 23 frames per second, meeting their requirements. (Zhao & Li, 2020)

Using the results from this project, Mask R-CNN would be an appropriate framework for the use of detecting ships from satellite imagery, a problem similar to targeting ships from other ships in the ocean. Since the backbone of Mask R-CNN can be chosen, that is, the architecture used for feature extraction over an entire image, from the previous research into ResNet, this may be the best architecture to choose for this project. The fast training without losing accuracy in the results would be ideal since one goal of the project is to update the machine learning model with re-trained weights for a new task or with improved accuracy.

2.4 STAGE-WISE TRAINING

The idea of updating a trained model is well defined with the idea of transfer learning being a common method due to the fact that “features that are transferred from other tasks are better than randomly initialised weights for the network, even when the features are of distant objects.” (Zou & Zhong, 2018) From previous research into transfer learning on convolutional neural networks, the most accurate model produced came from fine-tuning the layers conv1-fc8 or conv5-fc8. (Zou & Zhong, 2018) Considering that the layers in conv1-fc8 include all the layers in conv5-fc8, and it takes a much shorter amount of time to train only conv5-fc8, it would be ideal just to train these layers for high

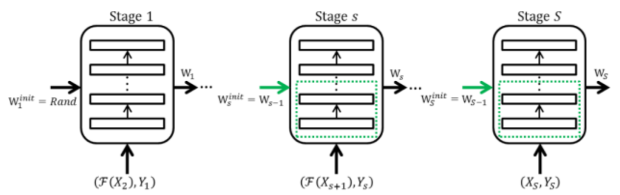

Figure 2: Stage-wise training showing the initialisation of the weights in different stages on different layers

accuracy, rather than taking more time to yield very similar results. This method of training is known as stage-wise training as only specific layers are trained at a time, in separate stages. (Hu, Dollár, He, Darrell, & Girshick, 2018)

The use of stage-wise training is mainly to deal with the issue of shared features between objects. For example, in a typical method of training a neural network, "these networks aim at learning features that are *unique* to each class, as opposed to the *typical* features of a class." (Barshan & Fieguth, 2015) Stage-wise training, however, allows the data to be presented to the network gradually meaning that the network initially only learns to make predictions on the available parts of the data. In later layers, more refined information is provided to the network to allow it to make better predictions, as shown in Figure 2.

2.5 NETWORK PRUNING

A large number of weights and connections in a deep neural network result in the consumption of large amounts of storage and memory usage as well as energy consumption. To fit the model onto a satellite, the neural network may need to be compressed, and one way of achieving this is through pruning the weights and connections. (Han, Mao, & Dally, Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, 2016)

The idea behind network pruning is that the connections that have only a small effect on the output are removed leaving only the most informative connections, reducing the storage, memory and power needed to run the model. There have been several different methods for pruning neural networks including filter pruning (Li, Kadav, Durdanovic, Samet, & Graf, 2017), connection pruning (Han, Pool, Tran, & Dally, 2015), the random dropping of neurons (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) and adding noise outputs to the network to allow for more efficient pruning (Babaeizadeh, Smaragdis, & Campbell, 2017).

With many different options for pruning the network, due to the structure of the existing network, a focus should be put on the pruning of connections from a pre-trained model. This method of pruning is done iteratively, with the connections being initially trained, then the model is pruned and re-trained to ensure the accuracy is not lost, as shown in Figure 3.

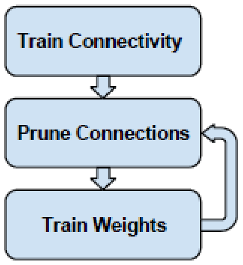

Figure 3: Structure for pruning and training the neural network

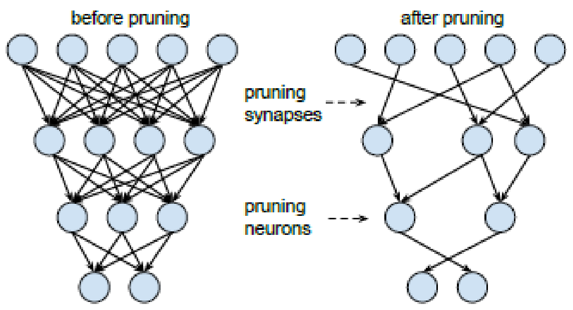

Figure 4: Diagram of neurons and connections before and after pruning

According to research already done into this method of pruning, the re-training of the network after pruning is essential. Otherwise the accuracy will be significantly impacted, and the network may not be able to perform as intended. (Han, Pool, Tran, & Dally, 2015)

Other research has also looked at pruning and quantisation of the model working together to reduce the size even further. In those experiments, it was found that the model they used was able to be pruned to 3% of its original size without a loss in accuracy. (Han, Mao, & Dally, Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, 2016)

To maintain simplicity and determine if it is feasible to effectively compress the ship detection model outlined here, only the method of pruning will be used. The layers that can be pruned through this method are the convolutional layers and the fully connected layers. (Han, Pool, Tran, & Dally, 2015) Their experiments found that the convolutional layers were more sensitive to pruning than the fully connected layers. That is, when the convolutional layers were pruned, they had a higher drop in accuracy than when the fully connected layers were pruned. Knowing this may allow us to choose to prune the less sensitive layers more to reduce the size as much as possible without affecting the accuracy.

3 METHODOLOGY

For this project, the current research will be used as a starting point to create an accurate model to detect ships in optical satellite imagery. This model will use a large dataset for training, Mask R-CNN for the framework and ResNet as the backbone for Mask R-CNN. Mask R-CNN is chosen due to its instance segmentation which allows each ship object in an image to be detected as well as each pixel that is included in a ship to be masked. ResNet is chosen for the backbone due to its useful residual blocks that do not degrade the performance of the neural network. A large dataset with high-resolution images, provided by Airbus for ship detection, will be used in the training of the network since this will allow for the best performance of the neural network.

Once the neural network is trained, the model will be evaluated by comparing the actual location of the ships in the images with the generated masks and the instances of ships in each image compared with the actual number of ships. The mAP (mean average precision) and mAR (mean average recall) will be calculated to ensure that the model is accurate enough for ship detection. The number of false negatives, that is, the ships that should have been detected but were not, and false positives, that is, other parts of the image that were masked but were not ships will also be examined for the accuracy of the model.

With an accurate model, the model pruning method will be implemented to attempt to reduce the size as much as possible without affecting the accuracy. This differs from the initial aim of the project to use transfer learning and stage-wise training to determine if only certain stages of the neural network can be updated to accurately perform a new task or increase the accuracy for the initial ship detection task. However, the model pruning will be necessary to run a large model such as efficiently on a satellite, as is the intended use case. Ideally, the pruned model would not need as much storage or memory, as well as using less power.

Model pruning could also be practical for the other initial aim of the project, to determine if it is practical to transfer a trained model from the ground to a satellite. With a reduced number of parameters, the pruned model may be able to be transferred much faster than if it was simply re-trained, allowing for the possibility of training the model for new tasks.

4 EXPERIMENTAL SETUP

4.1 MACHINE LEARNING MODEL SETUP

To prune the trained model to reduce its size, the trained model needs to be processed, and the layers to prune need to be selected. In each experiment to be run, the model will be pruned on particular layers, and then re-trained for a specific amount of time to determine the optimal result. The initial pre-trained model used for each experiment will be the same to ensure a fair test. The pruning will be done using TensorFlow 2.2.0, Keras 2.4.3 and CUDA 10.1 using an NVIDIA RTX 2060 GPU.

The training, testing, and pruning of the model will all be separated out to allow for a clear flow of the process. The model training will be done on the Airbus Ship Detection Challenge Dataset provided for use by the public on Kaggle. (Airbus, 2018) For the training method, it was decided to train the model on this dataset until the mAP and mAR results were high enough to serve a practical purpose. The model was pre-loaded with weights that were pre-trained on the COCO dataset, allowing for the use of transfer learning to help reduce the training time. The pre-trained weights used were provided with the Mask R-CNN model. (Abdulla, 2017) After training, the newly trained weights were saved before being fed to the testing methods to find their mAP and mAR to see how accurate the model is.

The inference of the Mask R-CNN implementation allows for a visualisation of the detections in each image, with the detected mask being shown on top of the image. By comparing the ground truth masks to the detected masks, we can visually see how accurate the model is.

For an accurate measurement of mAP and mAR, large sample size is needed. Rather than using all the images to test this, to reduce computation time, a sample of 500 images was chosen. This sample was large enough to not affect the accuracy numbers by a large amount but also small enough to not take too long in processing. For each image that the inference was run on, the results produced contain the regions of interest, the detected masks, the detected class IDs, and the scores. This data was then fed to a utility function provided with the Mask R-CNN model to calculate the AP (Average Precision) and recall for each image. (Abdulla, 2017) These values would then be used in turn to calculate the mAP and mAR for the model.

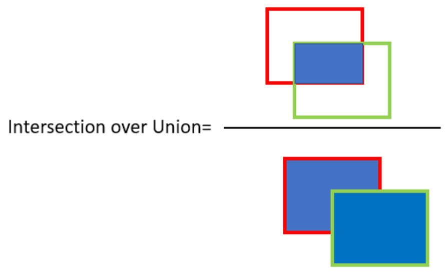

Figure 5: Visual explanation of Intersection over Union (IoU)

The mAP and mAR values are calculated based on the intersection over union of the ground truth masks and the detected masks.

The intersection over union (IoU) is calculated as the ratio of intersecting masks to the union of the two masks, determining how well the ground truth and detected masks overlap. (Balsys, 2020) The IoU threshold for determining if the detected image is a true positive is set to be 0.5 and if the IoU value is below this, the detection is classified as a false positive.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Equation 1: Formula to calculate precision for each image

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Equation 2: Formula to calculate recall for each image

Equations 1 and 2 show the formulas used for each image to calculate the precision and recall based on the IoU of the detections. Once all the images were processed, the mean of the precision and recall values was taken to find the mAP and mAR. The pruning of the model was chosen only to be done on specific layers as not to decrease the accuracy too far. From prior research, the convolution layers and fully connected layers were able to be pruned successfully without a loss of accuracy. This method will be used in this paper due to a large number of convolutional layers. Once the model was fully trained, it was then cloned with each convolutional layer being modified to be pruned during training. The input tensors of the model also needed to be specifically preserved to ensure that the model still performed as intended after it was re-trained.

The newly cloned model was then recompiled and set up to be trained to improve the accuracy that was lost by pruning the connections. The experiments done will look at different amounts of re-training, as well as different pruning sparsity to determine their effect on the Mask R-CNN model's accuracy and size.

It is also interesting to investigate how much re-training is necessary for the model to perform with similar results after the pruning has taken place. Ideally, little training will be needed, but this will need to be looked at during the experiments.

4.2 CHALLENGES

Due to compatibility issues between the different versions of the NVIDIA graphics driver used, TensorFlow, Keras and CUDA, setting up the Mask R-CNN machine learning model was not a trivial task. The Mask R-CNN framework that was used can be is modified from the initial implementation (Kelly, 2020) to work with TensorFlow 2 which needed to be used due to the CUDA version that was required for the GPU that was used.

Another challenge that was faced was the configuration of the machine learning model that would produce the best results for accuracy and recall. These parameters had to be changed individually for training the model as they each affect different aspects of the training. This process had to be repeated until the accuracy and recall were relatively high. During the training process and refining the parameters, another framework other than Mask R-CNN may be more ideal for ship detection. The Mask R-CNN framework works well for applying a mask over the detected ships; however, for the problem of merely detecting ships locations in images, other frameworks may be able to achieve higher accuracy.

Although other frameworks may have given better results for ship detection, the project was continued with Mask R-CNN. The Mask R-CNN model needed to be modified to work with the pruning APIs that were used. Due to the custom layers that are in the model architecture, the entire model could not be pruned but only individual layers. It would have been interesting to see how other model frameworks would compare if they were able to be pruned entirely, compared to pruning on only certain layers.

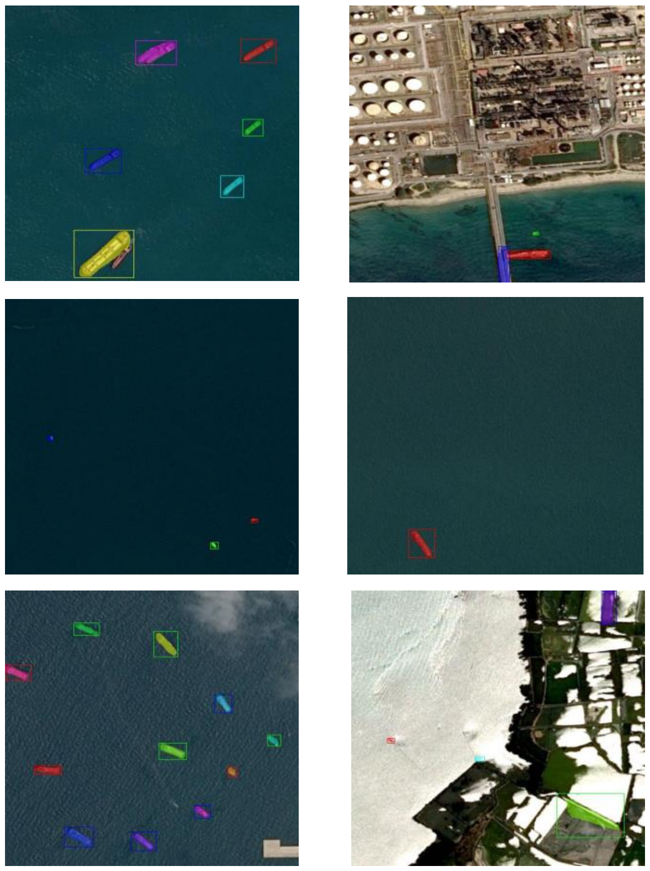

Figure 6: Examples of detections from the trained model

5 RESULTS

The Mask R-CNN model used is not proven to have the highest mAP and mAR compared to other machine learning image detection models, with the official paper declaring a mask mAP of 35.7%. Rather than aiming from a much higher mAP and mAR using this model for ship detection, the focus is instead put on reducing the model size as much as possible without decreasing the mAP or mAR by a large amount.

Some examples of the model detection are shown in Figure 6 including examples of false positives where land masses are predicted as being ships. It also shows a false negative case where two ships that are very close together results in one ship not being detected.

The mean average precision and mean average recall were calculated for the ship detection Mask R-CNN model and the results are outlined below in Table 1.

Mask R-CNN paper mAP	Ship detection model mAP	Ship detection model mAR
0.357	0.33	0.35

Table 1: Comparison of mAP and mAR with official Mask R-CNN paper

As outlined in the challenges section 4.2, the accuracy of the Mask R-CNN model precision is not significantly high enough to use in a practical sense for ship detection in the field of defence or other uses, however, for the task of model pruning, the difference in accuracy between the unpruned and pruned models can be investigated.

Model	mAP	mAR	Size (MB)	mAP decrease	mAR decrease	Size decrease
Not pruned	0.33	0.35	394.4			
Pruned	0.18	0.2	179.6	45%	43%	54%
Pruned + retrained for 7 epochs	0.25	0.25	179.6	24%	29%	54%

Table 2: Statistics of the pruned model compared to the unpruned model as well as fine tuning

Once the model was pruned, it was evaluated with respect to the original unpruned model to determine how the pruning affected the results. The results of this testing are shown in Table 2. The size of the pruned model was compared to the unpruned model, and it was 54% less. This means that the pruning of parameters that was undertaken was efficient in reducing the size of the model. The newly pruned model was given the same data for testing as the original model, and the mAP and mAR were calculated. The percentage decrease for these values was also calculated since the original accuracy of the model was not very high, and this may give a better indication of how the results changed. As expected, the mAP and mAR decreased immediately after pruning to 0.18 and 0.2, respectively. This works out to be a decrease of 45% for the mAP and 43% for the mAR from the original model. The pruned model was then copied and re-trained to determine if it would be able to get similar results to the model before being pruned. Once it was re-trained, both the mAP and mAR increased to 0.25. This is just 24% less than the original mAP and 29% less than the original mAR.

Considering that the size of the model decreased by 54%, the initial pruned model, which had an mAP decrease of 45%, is a good start before re-training. Ideally, more experiments need to be done with different pruning techniques to determine if this decrease would be consistent. However, due to the limited timeframe for this project, it is assumed that the effect that this pruning has would be consistent with other trials. Once the model was re-trained, the mAP was just 24% lower than the original unpruned model. This result shows that it is possible to improve the accuracy of a pruned model for ship detection after pruning.

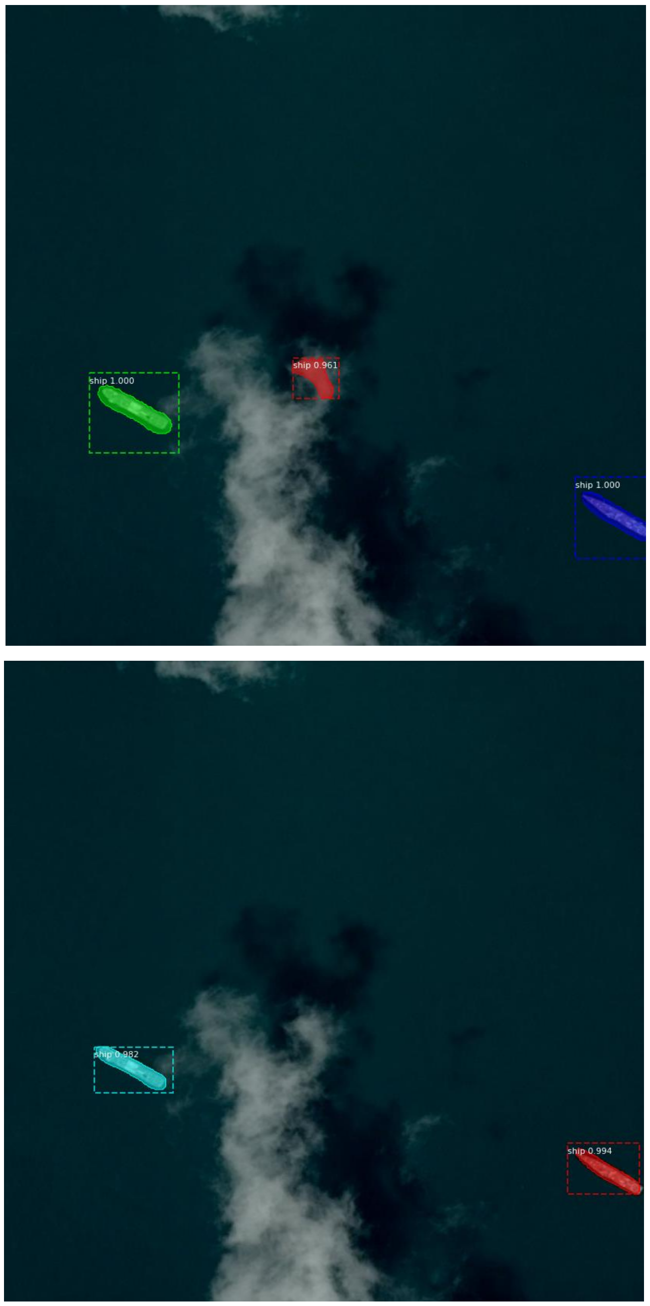

Figure 7: Example of detection from pruned model without fine tuning (top) and pruned model with fine tuning (bottom)

An example of the difference that re-training after pruning makes is shown in Figure 7. The images shown are of detections from the pruned model without re-training and with re-training on the same image. The model without re-training detects a cloud instead of a ship giving a false-positive result. Once the model is re-trained, it then only detects the ships in the image, which is one example of the precision increasing.

Overall, the increase in the mAP and mAR during training the pruned model shows that it is possible to improve a pruned model by a significant amount. The size decrease of the model after pruning proves that it is practical to use model pruning on a ship detection model that uses Mask R-CNN.

6 ACCESSING THE PROJECT REPOSITORY

The project can be accessed online for training, testing and pruning at: https://github.cs.adelaide.edu.au/a1766121/Space-Analytics-Engine. A description of the required software setup is included with the repository as well as some examples.

7 CONCLUSIONS

The reduction of the size of the model, as well as the decrease in the mAP and mAR, is expected based on previous research done in model pruning. (Han, Pool, Tran, & Dally, 2015) These results show that a space analytics engine based on a machine learning model, trained for ship detection, and loaded onto a satellite, could be an effective method, as it could be pruned to use less resources.

To improve the performance of the model, different architectures could be tested for ship detection rather than using Mask R-CNN. Other convolutional neural network architectures such as U-Net have been proven to also work for ship detection and segmentation, and these other architectures may give better results for mAP and mAR on the Airbus Ship Detection Challenge dataset. This, in turn, would allow for a better comparison between the pruned and unpruned models through more trials.

Another way to increase the number of trials is to test different pruning sparsity for each model. Previous research shows that as pruning sparsity increases, the mAP of the model will decrease (Zhu & Gupta, 2017), however, it would be interesting to see how the model size overall differs compared to the mAP. This would allow us to determine if compression could be increased while keeping accuracy close to the unpruned model.

Considering the analysis of the model in its pruned and unpruned forms, time constraints on this project meant that a method to determine the exact memory usage when running each model could not be determined. In a future investigation, calculating the memory required to run the model could be compared before and after pruning for another metric to compare between them.

Memory limitations also meant that the pruned model could not be trained for many epochs. Ideally, multiple tests could be run on the pruned model after it is trained for each epoch to determine how long it needs to be trained to perform well.

Due to time constraints for this project, the goal of running the pruned model on an NVIDIA Jetson Nano could not be achieved. However, the idea of model pruning to reduce memory, storage, and power use would be ideal for assisting in converting the model to run on such a device. This would simulate running the neural network model on a satellite which may have a similar form factor and limitations.

Another investigation would be to determine if it is worth training a large model to be accurate in this task before pruning rather than simply training a smaller model from the beginning to accomplish a similar goal. A smaller model may be able to fit the requirements needed to operate on a satellite and the pruning of a large model may not be necessary at all. However, this remains to be determined based on the accuracy of the smaller model compared to that of the larger model before and after pruning.

The research conducted in this project has allowed for the possibility of many different experiments and new methods. In a future investigation, the experiments outlined in the conclusions section should be investigated to prove further or contradict the results given in this paper.

REFERENCES

Abdulla, W. (2017). Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. *GitHub repository*. Retrieved from https://github.com/matterport/Mask_RCNN

Airbus. (2018). *Airbus Ship Detection Challenge*. Retrieved from Kaggle: https://www.kaggle.com/c/airbus-ship-detection/data

Babaeizadeh, M., Smaragdis, P., & Campbell, R. H. (2017). NoiseOut: A Simple Way to Prune Neural Networks. *ICLR*. Retrieved from https://arxiv.org/abs/1611.06211

Balsys, R. (2020, July 15). Understanding the mAP (mean Average Precision) Evaluation Metric for Object Detection. *Analytics Vidhya*. Retrieved from https://medium.com/analytics-vidhya/understanding-the-map-mean-average-precision-evaluation-metric-for-object-detection-432f5cca53b7

Barshan, E., & Fieguth, P. (2015). Stage-wise Training: An Improved Feature Learning Strategy for Deep Models. *JMLR: Workshop and Conference Proceedings 44*, 49-59. Retrieved from http://proceedings.mlr.press/v44/Barshan2015.pdf

Chen, K., Wu, M., Liu, J., & Zhang, C. (2020, March 15). FGSD: A dataset for fine-grained ship detection in high resolution satellite images. Retrieved from https://arxiv.org/pdf/2003.06832.pdf

Fung, V. (2017, July 16). *An Overview of ResNet and its variants*. Retrieved from Towards Data Science:

https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035

Han, S., Mao, H., & Dally, W. J. (2016). Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *ICLR*. Retrieved from https://arxiv.org/abs/1510.00149

Han, S., Pool, J., Tran, J., & Dally, W. J. (2015). Learning bothWeights and Connections for Efficient Neural Networks. *NIPS*. Retrieved from https://arxiv.org/abs/1506.02626

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. *IEEE International Conference on Computer Vision (ICCV)*. Retrieved from https://arxiv.org/abs/1703.06870

He, K., Zhang, X., Ren, S., & Sun, J. (2015, December 10). Deep Residual Learning for Image Recognition. Retrieved from https://arxiv.org/pdf/1512.03385.pdf

Hu, R., Dollár, P., He, K., Darrell, T., & Girshick, R. (2018). Learning to Segment Everything. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4233-4241. Retrieved from https://openaccess.thecvf.com/content_cvpr_2018/html/Hu_Learning_to_Segment_CVPR_2018_paper.html

Hui, Z., Na, C., & ZhenYu, L. (2019). Combining a Deep Convolutional Neural Network with Transfer Learning for Ship Classification. *2019 12th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, 16-19. doi:10.1109/ICICTA49267.2019.00011

Kelly, A. (2020, June 20). *Mask R-CNN*. Retrieved from Github: https://github.com/akTwelve/Mask_RCNN

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2017). Pruning Filters for Efficient Conv Nets. *ICLR*. Retrieved from https://arxiv.org/abs/1608.08710

Li, Y., Ding, Z., Zhang, C., Wang, Y., & Chen, J. (2019). SAR Ship Detection Based on Resnet and Transfer Learning. *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, 1188-1191. doi:10.1109/IGARSS.2019.8900290

Sagar, A. (2019, June 28). *Deep Learning for Ship Detection and Segmentation*. Retrieved from MC.AI: https://mc.ai/deep-learning-for-ship-detection-and-segmentation/

Shaodan, L., Chen, F., & Zhide, C. (2019). A Ship Target Location and Mask Generation Algorithms Base on Mask RCNN. *International Joual of Computational Intelligence Systems*, *12*(2), 1134-1143. doi:https://doi.org/10.2991/ijcis.d.191008.001

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 1929-1958. Retrieved from https://dl.acm.org/doi/10.5555/2627435.2670313

Vidyapith, B. (2019, September). Smart ship detection using transfer learning with ResNet. *International Research Journal of Engineering and Technology, 6*(9), 1870-1974. Retrieved from https://www.irjet.net/archives/V6/i9/IRJET-V6I9291.pdf

Zhao, D., & Li, X. (2020). Ocean ship detection and recognition algortim based on aerial image. *Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC)*, 218-222. doi:10.1109/IPEC49694.2020.9115112

Zhu, M. H., & Gupta, S. (2017). To prune, or not to prune: exploring the efficacy of pruning for model compression. Retrieved from https://arxiv.org/abs/1710.01878

Zou, M., & Zhong, Y. (2018, February 15). Transfer Learning for Classification of Optical Satellite Image. *Sens Imaging, 19*(6). doi:10.1007/s11220-018-0191-1