BOHR International Journal of Smart Computing and Information Technology
2023, Vol. 4, No. 1, pp. 7–16
DOI: 10.54646/bijscit.2023.32
www.bohrpub.com

METHODS

Object detection and ship classification using YOLOv5

Sean Brown*, Caitlin Hall*, Raffaele Galliera* and Sikha Bagui*

Department of Computer Science, University of West Florida, Pensacola, FL, United States

***Correspondence:**
Sean Brown,
sab138@students.uwf.edu
Caitlin Hall,
ceh59@students.uwf.edu
Raffaele Galliera,
rg101@students.uwf.edu
Sikha Bagui,
bagui@uwf.edu

Received: 08 February 2023; **Accepted:** 03 March 2023; **Published:** 18 March 2023

Using a public dataset of images of maritime vessels provided by Analytics Vidhya, manual annotations were made on a subsample of images with Roboflow using the ground truth classifications provided by the dataset. YOLOv5, a prominent open source family of object detection models that comes with an out-of-the-box pre-training on the Common Objects in Context (COCO) dataset, was used to train on annotations of sub-classifications of maritime vessels. YOLOv5 provides significant results in detecting a boat. The training, validation, and test set of images trained YOLOv5 in the cloud using Google Colab. Three of our five subclasses, namely, cruise ships, ROROs (Roll On Roll Off, typically car carriers), and military ships, have very distinct shapes and features and yielded positive results. Two of our subclasses, namely, the tanker and cargo ship, have similar characteristics when the cargo ship is unloaded and not carrying any cargo containers. This yielded interesting misclassifications that could be improved in future work. Our trained model resulted in the validation metric of mean Average Precision (mAP@0.5) of 0.932 across all subclassification of ships.

Keywords: object detection, image classification, maritime, ship classification, YOLOv5.

Introduction

In the military maritime realm, there is a desire to understand the environment passively without using emitters such as RADAR and LiDAR. A vessel can keep a low profile and detect navigational hazards or threats. In the commercial maritime realm, ships can use images and videos in conjunction with RADAR to add another layer of confidence to hazard avoidance. These industries need reliable object detection using cameras. Hence, this study will explore the use of electro-optical cameras taking still images to be used in a dataset to retrain a YOLOv5 model, a deep learning object detector, on such images and evaluate the performances.

Object detection is a technique for locating instances of objects in either images or key frames from videos by leveraging machine learning algorithms with the goal of replicating recognition intelligence using a computer. Object detection can be applied to many domains as long as a sufficient domain-based dataset is available. But, object detection algorithms must also consider condition factors applicable to that domain, such as poor weather or lighting conditions.

To study the application of object detection using still images and videos, an applicable dataset of maritime images from Analytics Vidhya (1, 2) was used, and five different classifications of ships were defined for the algorithm to detect. This study also takes into consideration various factors that can affect the results and attempts to improve the training dataset and algorithmic configurables. A subsample of images has been annotated using Roboflow (3) and trained through an existing off-the-shelf object detection algorithm called YOLOv5 (4).

YOLOv5 was born due to the improvements made by Jocher (5), who ported YOLOv3's Darknet weights

(6) to PyTorch (7). PyTorch is an open-source machine learning framework that allows users to implement powerful computational functions that are accelerated with a GPU's processing power. YOLOv5 version 6.1 was released on February 22, 2022, which featured the YOLOv5n model for ultralight edge devices. At the time of this report, the latest version, 6.1, was utilized. The YOLOv5s model was used due to the constraints of using free cloud-based tools to facilitate our training and detection, as described later in the section "Methodology."

This study is organized as follows. The section "Literature Review" reviews works related to the use of machine learning techniques on images of maritime ships, specifically with the YOLO family of algorithms and similar machine learning algorithms. The section "Methodology" describes the dataset and methods of pre-processing, annotation, training, and validation. The section "Results and Discussion" discusses the results and the key metrics of mAP, precision, recall, AUC, and F1 scores. Finally, the section "Conclusion and Future Work" discusses the conclusions and future ideas.

In the literature review, we will compare and contrast previous works with this study and determine the uniqueness of the dataset and methodology, as well as increase the fundamental knowledge on the subject of object detection and classification. It should be noted that the demand for advanced video surveillance and perception capability has been requested by the United States Department of Defense (DoD). The DoD has set aside millions of dollars to procure and field innovative technologies from non-traditional vendors, making this research in-demand and valuable.

Many studies discuss machine learning in the application of ship classification (8), but to date, none of these works address the problem with the off-the-shelf application of YOLOv5 and the dataset from Analytics Vidhya. The uniqueness of this study is that it applies YOLOv5 to a large dataset consisting of various types of ships, in addition to the varying quality of images from multiple viewpoints.

Literature review

Research has been done for ship classification using alternative algorithms and methods and different datasets. Kim et al. (9) used a different dataset that includes images and is focused on the improvement of a preexisting classification with different defining subclasses of ships including "Boat," "Speed Boat," "Vessel Ship," "Ferry," "Kayak," "Buoy," "Sail Boat," and "Others." They achieved a mean average precision (mAP) (0.5) value of 0.898 and an mAP (0.5:0.95) value of 0.528. Li et al. (8) presented a combination of real-time ship classification, Ship Detection from Visual Image, and YOLOv3 to achieve an mAP value of 0.741.

Tang et al. (10) compared different versions of YOLO for datasets of Synthetic Aperture Radar images and traditional satellite camera images of ships. While similar in nature in terms of using YOLO and object detection of ships, the datasets are quite different in terms of image capture angle. The dataset chosen for this study, from Analytics Vidhya, contains much closer images of various horizontal profiles with five distinct classifications, namely, "Cargo," "Military," "Carrier," "Cruise," and "Tankers." Using YOLOv3, Liu et al. (11) took steps to improve the algorithm's accuracy of large pixel dense satellite images by reducing the original networks 32 times down-sampling to four times, as well as using a sliding window method to cut down large images to many smaller images.

The structure of YOLOv5 can be split into input, backbone, neck, and prediction. Some research has been done on creating new backbones, improving existing backbones (12), or swapping YOLOv5's backbone for another existing backbone. Ting et al. (13) swapped the exiting backbone for Hauawei's GhostNet (14) and stacked two of these GhostNets into what they call a Ghostbottlenet. Using the Ghostbottlenet instead of YOLOv5's original backbone, they were able to improve feature extraction and reduce the overall model size. Zhou et al. (15) also replaced the original backbone with Mixed Receptive Field Convolution (MixConv), where MixConv makes use of multiple convolution kernels to improve feature extraction by increasing attention to pixel coordinates in horizontal and vertical channels. Qiao et al. (16) attempted to re-identify maritime vessels that the model has already seen even at other orientations, using a Global-and-Local Fusion-based Multi-view Feature Learning by replacing the backbone with ResNet-50 for global and local feature extraction.

Orientation recognition is the focus of another study where the researchers use a single shot detector (SSD) for both multiclass vessel detection and five defined orientations (i.e., front, front side, side, backside, and back) (17, 18). SSD is a feedforward ConvNet that explores the presence of an object instance in the predefined default bounding boxes, followed by a non-maximum suppression stage to produce the final detection. Tang et al. (19) explored the use of an SSD with hue, saturation, and value pre-processing to improve the Intersection Over Union. The hue, saturation, and value pre-processing operation are used to extract regions of interest to feed to the YOLO network.

There are other studies in the same field that cross compare different object detection algorithms, such as Faster R-CNN(20), R-FCN (21), SSD, and EfficientDet (22), while still attempting to detect maritime vessels. Iancu et al. (23) found that in small to medium size objects greater than 162 pixels, Faster R-CNN with Inception-Resnet v2 outperforms the others except in detecting large objects where EfficientDet does a better job (23). It is interesting to note that all of the convolutional neural network (CNN) based detectors were also pre-trained on the COCO (24) dataset, similar to YOLO, which is also a CNN (23).

The COCO (24) dataset by Microsoft houses 3,30,000 images and 1.5 million object instances, and 80 object

categories. The richly annotated dataset contains objects in their natural context and depicts complex everyday scenes. The dataset focuses on segmenting individual object instances rather than what other object recognition datasets support, such as image classification, object localization, or segmentation. Since one of the 80 object categories is "boat," we can utilize transfer learning for our five sub-classifications of "boat" since YOLOv5 is pretrained on COCO.

Methodology

Pre-processing

Before this study gets into the explanation of the implemented methods, it would be beneficial to give some background on the dataset and some design decisions and difficulties found while taking the first steps with the dataset. This dataset of maritime vessels, provided by Analytics Vidhya (1, 2), came with over 8,000 images that were already ground truth labeled for the classification of "Cargo," "Military," "Carrier," "Cruise," and "Tanker." While the images did have a ground truth label, they did not have individual bounding boxes for labeled objects inside the image. Specifically, it was found that images of cruise ships would often have two or more cruise ships in the frame of the image.

Cargo ship was renamed Container ship, and Carrier ship was renamed "RORO," which stands for Roll-On-Roll-Off ships. ROROs have a very particular shape and size, as shown in **Figure 1**.

Labels were renamed to distinctly identify the difference between a "Carrier" and "Cargo." Other difficulties with this dataset were similarities in the images of container ships and tankers. While annotating and drawing bounding boxes by hand progressed, it became challenging to determine the difference between the two, specifically when a container ship was empty and did not have any containers loaded on the deck. Some tankers had piping and other mechanical features on the deck, while some had a flat deck and looked like an empty cargo container ship.

The data were separated into a training set, validation set, and testing set, to be used by YOLOv5. Roboflow was utilized to assist the project in pre-processing the data. Using Roboflow, there were 1,500 annotated images, dividing the classifications into 300 annotations per classification. Controlling the number of annotations evenly boosted the performance of training YOLOv5. Other Roboflow pre-processing steps were taken to orient any flipped or rotated images automatically and also to resize the images to 416 pixels by 416 pixels.

FIGURE 1 | Roll-on-roll-off ship classification example.

Training an object detection model

To train a model on the data, a cloud environment, Google Colab (25), was utilized. Google Colab offers free GPU runtime and leverages a python-based Jupyter notebook. From this environment, you can clone the public Github repository for YOLOv5 and install all the necessary python package requirements.

Roboflow's export to a Jupyter notebook is seamless. There are two options for online and offline imports to a project notebook. For an offline configuration, the images and annotations can be downloaded and placed in a notebook. Through the internet, Roboflow's API can be used in a few lines of auto-generated code to bring in the dataset. Python, the programming language of choice for data science engineers, comes loaded with tools for transforming results into usable graphs and using the trained YOLOv5 model in action to produce images with classification bounding boxes and confidence levels as shown in **Figure 2**.

In the first attempt at training a YOLOv5 model, we passed the following arguments to the train.py script:

```
python train.py -img 416 -batch 16 -epochs 150 -data {dataset.location}/data.yaml -weights yolov5s.pt -cache
```

The "img" flag defines the size in pixels of the input images by length and width. The "batch" flag determines the batch size, that is, how much can be loaded into memory; this is dependent on the hardware.

In total, 16 was chosen as a recommended default, but the YOLOv5 documentation warns not to use small batch sizes as they can produce poor batchnorm statistics. The "epochs" flag is the number of complete passes the training dataset

FIGURE 2 | Container ship classification example.

makes through the algorithm; 150 was a recommended default. The number of epochs is a crucial parameter to tweak to look for overfitting or underfitting. Overfitting negatively impacts the model's performance and ability to generalize new data.

The "data" flag is the file location of the training data. The "weights" flag is the small COCO pre-trained checkpoint of YOLOv5s.pt. This checkpoint was selected to take into consideration the cloud-based training environment duration requirement of free use for less than 24°h. Lastly, the "cache" flag was used to cache images for faster training.

Post-processing

Our first attempt at training a custom YOLOv5 model was with some recommendations for the parameters of train.py. After viewing some of the charts produced by TensorFlow, it was determined that our model was overfitting. An undesirable positive slope occurred at the end of the bounding box regression loss graph (Mean Squared Error) or box_loss graph. To correct this overfitting problem, the epochs were lowered from 150 to 100 to reduce the number of times the training data goes through the algorithm.

Results and discussion

Dataset quality

The dataset was obtained from Analytics Vidhya for the Game of Deep Learning: Computer Vision Hackathon (1, 2). It was essential to gather a wide variety of photos with a few key factors to test the performance of the machine learning model. The first type of image contained in the dataset is a simple image of one ship with that ship being the focus of the image. The second type of image is a single image containing multiple same classifications of the ship. The third type of image is a single image containing numerous vessels of different classifications. The fourth type is an image of a ship, but it is not the main focus of the photograph. The ship may be in the background or blend into the environment more so than in other photographs.

The dataset also contains images both in color and black and white, as well as blurry images and clear images. Including all of these types of images in the dataset ensures that it challenges the machine learning model to predict classifications that are not inherently obvious and pushes the limits further of how well the model can predict given less than ideal images.

Object detection

As mentioned in the section "Methodology," the machine learning model was run through Google Colab using YOLOv5 and Python in a Jupyter notebook. The model ran 100 epochs and produced images in which it made predictions on classifications of ships in multiple images. Some of these images were classified correctly, and some of them were not. In a few of the predictions, there were various ships within the images for added complexity to the model. **Figures 1–7** show examples of the algorithm's output. The model performs differently on different images for object detection. It is possible to correctly classify an object, misclassify a ship, or classify background objects as a ship, and it can be possible to miss the detection entirely.

Figure 3 shows the model successfully classifying multiple ships within the same image of type "Cruise Ship," so it is understood that the model has the capability to identify more than one object per image.

Figures 2–4 all show successful classifications of three different types of ships with images of varying quality. It can be seen that **Figures 2**, **3** have a higher confidence in classification than **Figure 4**, likely due to the poor image quality of **Figure 4**. However, it is important to note that the model can detect an object even when the image quality is not ideal.

Figure 5 is an example of the model missing the detection entirely. It can be seen that there are two cruise ships in the image, but the model identifies only one.

Figure 6 shows another capability of the model in that it can identify overlapping ships. In this image, the tanker was located directly in the foreground of the container ship, and the model was still able to detect both ships successfully. Likely the container ship suffers low confidence due to the model not observing the bow and stern of the container ship, which are very distinct features.

Finally, **Figure 7** shows an example of the model misclassifying a background object and a ship. In this image,

FIGURE 3 | Cruise ship classification example.

FIGURE 4 | Military ship classification example.

the model confused a bridge for a tanker, most likely due to the large towers of the bridge resembling a mast erected vertically above the hull of a tanker. The false positive of the large bridge could be mitigated by adding additional images containing bridges and maritime ships that do not have bounding boxes containing the bridge. There are a few aircraft carriers mixed into the dataset, but adding more would improve precision.

FIGURE 5 | Missing classification example.

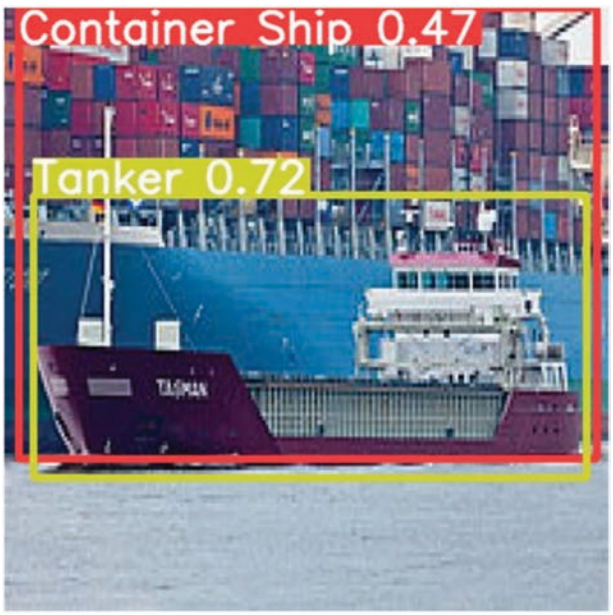

FIGURE 6 | Overlay classification example.

Post-processing

As described in the "Post-processing" subsection of the "Methodology" section, the method of rectifying the model's apparent overfitting observed in the box_loss graph was to reduce the epochs from 150 to 100. As seen in **Figure 8**, the data for the box_loss appears to be trending in a healthy manner and does not indicate overfitting, as seen in the previous training attempt.

The phenomenon of overfitting occurs when the model has been trained for too long, becomes too specific to the training set, and performs poorly with new data. An ideal learning

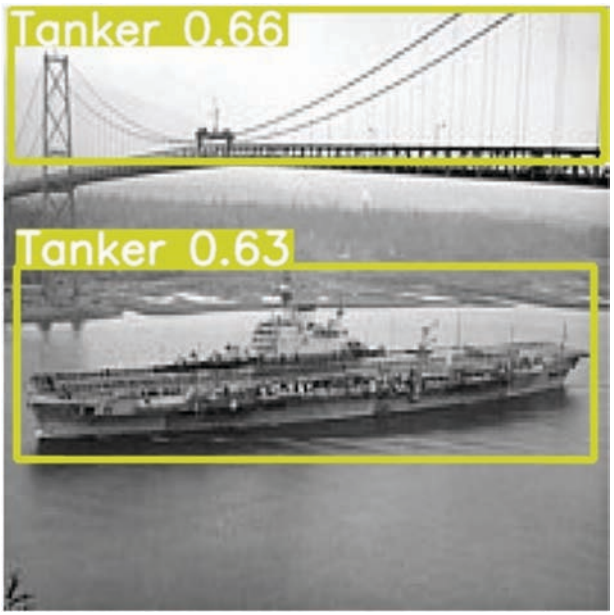

FIGURE 7 | Misclassification example.

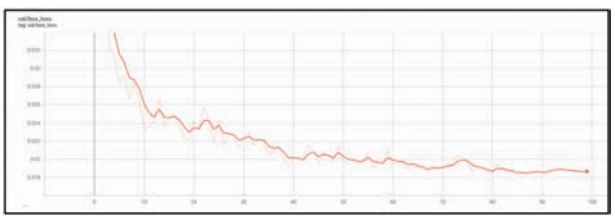

FIGURE 8 | Validation box loss chart.

curve graph is identified when training and validation loss decreases to the point of stability.

Performance

The concept of algorithm performance is interpreted as the quality of results the machine learning model produced.

For this project's scope, the focus will be on the model's performance to accurately determine the correct classification of ships. To do this, a variety of key performance indicators are used. The most important of these indicators is the mAP metric. However, there are an additional four other metrics worth discussing as a part of the algorithm results.

The mAP metric achieved for this project was 0.929 at the 0.5 level and 0.598 at the 0.95 level. The mAP metric relies on the precision, recall, precision-recall curve, and intersection over union, which is discussed in detail later in the paper. To give a brief definition of each, intersection over union is the measure of how much the bounding boxes on the object overlap. This value will be 1.0 for exactly lined up or 0.0 if there is no overlap at all.

TABLE 1 | Validation summary results.

Class	Images	Labels	P	R	mAP@0.5	mAP@0.5:0.95
All	292	322	0.892	0.902	0.929	0.598
Container	292	69	0.845	0.844	0.920	0.567
Ship Cruise	292	59	0.871	0.913	0.902	0.568
Ship Military	292	65	0.921	0.938	0.961	0.583
Ship RORO	292	60	0.993	1	0.995	0.740
Tanker	292	69	0.830	0.777	0.870	0.532

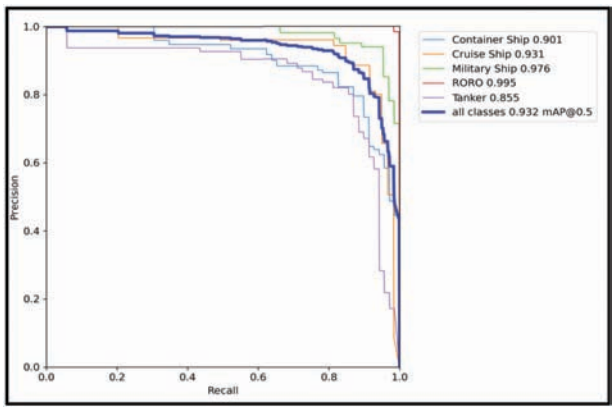

FIGURE 9 | Precision-recall chart.

Precision is the measure of identifying relevant objects; recall is the measure of the truth within the bounding boxes. The mAP metric is the mean of the AP metrics. An AP metric is the area under the precision-recall curve, shown in **Figure 9**. This calculation is shown in Equation (1):

$$AP = \sum [\text{Recalls (k)} - \text{Recalls (k} + 1)] * \text{Precisions (k)} \quad (1)$$

The larger the area under the curve (AUC), the higher the AP, which indicates a better model. Our mAP metric is higher at the 0.5 Intersection Over Union level, known as the traditional level, meaning that when there is half overlap over the union area of the two bounding boxes, the model has an accuracy of 92.9% as seen in **Table 1**.

The four other main charts are shown in **Figures 9–12**. These figures visualize the key performance indicators. To better understand these key performance indicators and interpret the results, it is crucial to first discuss the confusion matrix in **Figure 13** and the four key metrics used in the calculation of the key performance indicators. The four key factors used in these equations (with example definitions in terms of a RORO classification) are as follows:

- True Positive (TP): The ship was classified as a RORO and it was actually a RORO.
- True Negative (TN): The ship was classified as not a RORO and it was actually not a RORO.

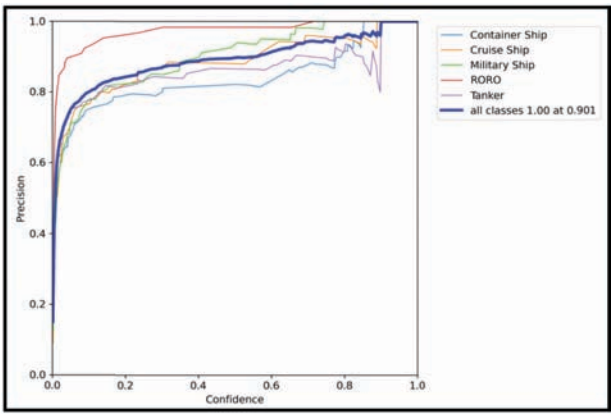

FIGURE 10 | Precision chart.

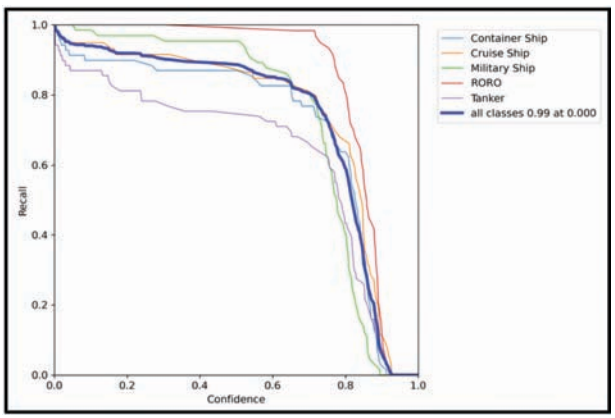

FIGURE 11 | Recall chart.

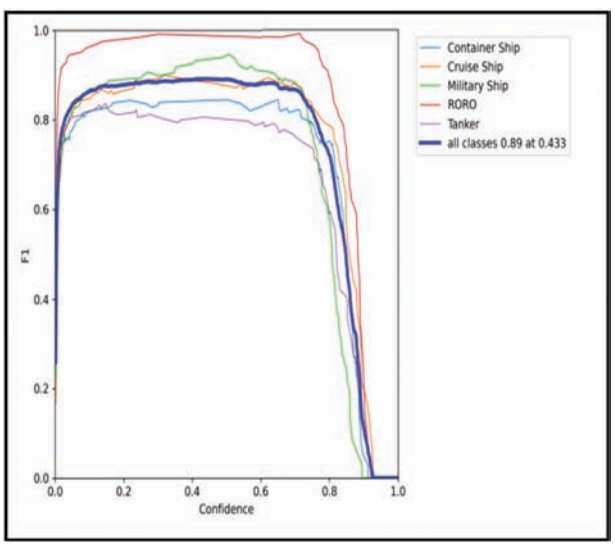

FIGURE 12 | F1 curve chart.

- False Positive (FP): The ship was classified as a RORO and it was not actually a RORO.
- False Negative (FN): The ship was classified as not a RORO and it actually was a RORO.

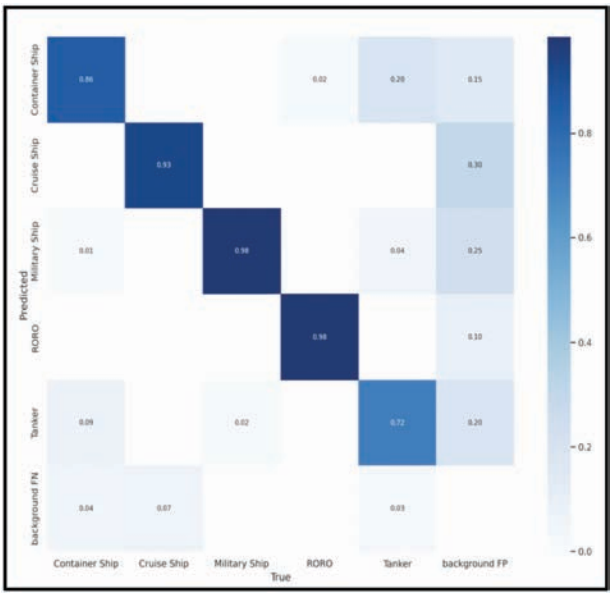

FIGURE 13 | Confusion matrix.

While the concept of these metrics is essential to the indicators below, the confusion matrix will help to understand the performance of the machine learning model, how the model has made the predictions, and the area containing the most prevalent errors of the model predictions.

The confusion matrix in **Figure 13** shows the true classification compared to the predicted classification. It can visualize where the model is getting confused in classifying or differentiating between two different classifications. This is visualized through a two-by-two matrix with one axis being the actual truth, or ground truth, and the other axis being the prediction, or the model's truth. In a perfect situation, 1.00 would be seen across the diagonal from the top left to the bottom right of the matrix. However, the model is not perfect. The model gets pretty close to perfect as the correct classification percentage for each category of ship is as follows:

- Container ship: 86%
- Cruise ship: 93%
- Military ship: 98%
- RORO: 98%
- Tanker: 72%

In addition to showing the percentage of the algorithm correctly classified, the breakdown of incorrect classifications can be seen as well. The most common misclassification occurred when the algorithm mistook the background of the image for a cruise ship. Confusion between classifications of ships arose the most between incorrectly classifying tankers as container ships 20% of the time.

One of the performance indicators is precision, shown in **Figure 10**. Precision measures the accuracy of the predictions or the percentage of the predictions that are correct. The

precision can be calculated by dividing the number of true positives by the summation of true positives and false positives. This calculation is shown in Equation (2):

$$\frac{TP}{TP + FP} \tag{2}$$

The higher the precision, the more accurate the model was at predicting the correct classification of vessels. **Figure 10** shows precision over confidence. In this case, confidence is a value between 0.0 and 1.0, which indicates how confident the model is that the prediction is correct. In general, a confidence of 0.7 and above is strong, between 0.3 and 0.7 is okay, and below 0.3 is weak and probably not a good prediction.

Figure 10 shows that as the confidence grows, the precision grows as well. It is interesting that the precision grows logarithmically and not linearly with confidence. It starts with a precision of around 20% when the confidence is 0.0, then rockets to approximately 75% by a confidence of 0.1. It is evident from this chart that the tanker classification contains the most variability and is the most incorrectly classified vessel in this model because its precision over confidence has the most drastic spikes at a high confidence value while the other classifications remain somewhat stable.

The second performance indicator is recall, shown in **Figure 11**. Recall is an indicator of how well all the positives (or correct classifications) are identified. This indicator is calculated by dividing the number of true positives by the summation of true positives and false negatives.

According to **Figure 11**, the recall decreases as the confidence increases. This calculation is shown in Equation (3):

$$\frac{TP}{TP + FN} \tag{3}$$

The third performance indicator is the precision-recall rate, shown in **Figure 9**. A precision-recall (PR) curve plots the precision values on the y-axis and the recall values on the x-axis for each classification in the model. Precision is measured by Eq. (2), and recall is measured by Eq. (3).

It would be ideal to see that the algorithm has both high recall and high precision, but in most cases, this is not possible, so there is usually a trade-off between the two in machine learning algorithms. To analyze this specific graph, it is known that a good machine learning model produces a high AUC.

In **Figure 9**, it can be seen that the RORO classification has the greatest AUC with a PR value of 0.995. The tanker classification had the lowest AUC with a PR value of 0.855. Overall, the results show that on average, of all classes, the AUC is significant with a PR value of 0.932, indicating a good result.

Finally, the fourth performance indicator is the F1 score, shown in **Figure 12**. The F1 score combines the precision and recall metrics and is a measure of the accuracy of the dataset.

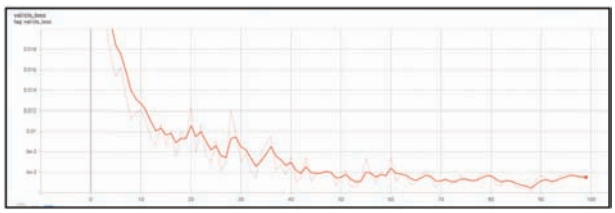

FIGURE 14 | Validation CLS loss chart.

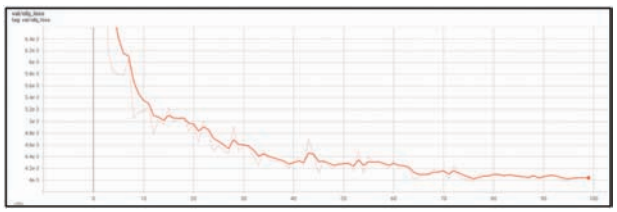

FIGURE 15 | Validation object loss chart.

Generally, an F1 score above 0.9 is excellent, between 0.8 and 0.9 is good, between 0.5 and 0.8 is okay, and below 0.5 is not good. In this study, the F1 score is highest at a confidence of 0.433 with an F1 score of 0.89.

This score rating is just bordering the excellent range, but remains in the good range. One of the factors for this score is class balancing, in which the goal is to have an even amount or fairly distributed dataset of the different classification types contained in the data, and a large amount of them for more accurate learning.

Validation

The goal of validation is to validate the model's performance by how well it correctly predicted the correct classification of the ship and that the object is accurately detected. There can be, and are, instances where the model identifies a background image as some classification of a vessel or classifies a ship incorrectly. **Figures 8**, **14**, **15** represent the performance of the validation set. In each of these figures, it is important to look at the dark orange line, which is the validation, compared to the faded orange line, which is the training data.

The validation box loss chart in **Figure 8** shows the mean square error (MSE) of the validation data vs. training data. MSE depicts how close the regression line is to a set of points by calculating the distance from the points to the regression line and squaring the error. You can observe from the graph that the MSE consistently declines and never trends in the positive direction. Box loss shows how well the predicted box overlaps with the validation bounding box.

Figure 14 shows the classification loss or cross-entropy. Entropy measures the average amount of information needed to represent a random event drawn from a probability for a random variable. Cross-entropy is the measure of the difference between two probability distributions of random

sets and can be used as a loss function when optimizing classification models. **Figure 14** shows the validation of the model to classify the object given a set of possible classifications correctly. Lastly, in **Figure 15**, the object loss chart shows binary cross entropy or the ability of the model to detect an object of interest or not accurately.

In each of these charts, it would be evident if the model was overfitting or underfitting. If overfitting, the algorithm would do well on the training dataset but poorly on validation data. This scenario would visually be identifiable by the validation line being consistently above the training data line even though the loss on the training data is low.

Underfitting occurs when the algorithm performs poorly not only on the validation data, but the training data as well. It is visually identifiable by looking at the validation charts if both validation and training data lines are separated, and the training line is above the validation line. This may mean that the algorithm is too complex for the given dataset in comparison to underfitting, where the model is not complex enough for the given dataset. All three of our validation charts show healthy loss functions. **Table 1** shows training and validation results, showing precision, recall, and mAP. The standard for comparing object detectors is mAP, and for our classifications, we are pleased with the results and will compare them with other related projects in the section "Conclusion."

Conclusion and future work

With the methods chosen, using a public dataset, annotating images from the dataset with Roboflow, and training an off-the-shelf machine learning algorithm YOLOv5 in a cloud-based environment, the application of commercial and military passive perception of maritime ships is achievable. Object detection and classification of maritime ships have many options of machine learning algorithms but our results prove that YOLOv5 is a competitive CNN, as indicated by our mAP value of 0.929 and healthy validation curves presented in the section "Validation."

We found that our custom-trained model using the Analytics Vidhya dataset performed better in terms of mAP of different Intersection Over Union thresholds from 0.5 to 0.95 in 0.05 increment steps compared to the related works of Kim et al. (9), where their model resulted in a mAP@0.5:0.95 of 0.528 and our model performed at 0.598. Another comparison is of the related works of Li et al. (8) using YOLOv3 to achieve an mAP value of 0.741, and our custom YOLOv5 model produced an mAP value of 0.929. Iancu et al. (23) at best produced an mAP value of 55.48% while cross-comparing four other competitors to YOLOv5 compared to our model's mAP of 92.9%.

In future work, to rectify the misclassifications of cargo container ships and tankers with similar features, additional images with varying characteristics would have to be added to the datasets. The algorithm would benefit from annotating images where a container ship may be empty and the same for tankers and various stages depending on cargo load. Other future work that could be useful to commercial and military customers is estimating the distance and bearing of the target after properly detecting and classifying an object of interest.

We plan to investigate improving our results by increasing the number of hand-annotated images in our training and validation datasets. Increasing the number of images per classification will also further refine our results. There is also news of future versions of YOLOv6 and YOLOv7 that could be utilized, as well as changing to larger pre-trained weights to compare results. With the growing fleet of commercial and military ships, there is demand for research like this study, and we think the future will use cameras and machine learning as a passive perception system for maritime ships.

Author contributions

SB and CH prepared the dataset by annotating the images and also worked on applying the machine learning algorithms, and wrote the initial draft of the manuscript. RG and SB provided the guidance and supervised the whole project. SB and CH. RG provided comments and SB did the overseeing and final edits of the manuscript to bring it to publication form. All authors contributed to the article and approved the submitted version.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Analytics Vidhya. *Game of Deep Learning: Computer Vision Hackathon*. (n.d.). Available online at: https://datahack.analyticsvidhya.com/contest/game-of-deep-learning/ (accessed June 5, 2022).
2. Kaggle. *Game of Deep Learning: Ship Datasets*. (n.d.). Available online at: https://www.kaggle.com/datasets/arpitjain007/game-of-deep-learning-ship-datasets (accessed June 5, 2022).
3. Roboflow Inc. *Roboflow Annotate [Computer Software]*. Des Moines, IA: Roboflow Inc (2020).
4. Ultralytics. *GitHub - ultralytics/yolov5: YOLOv5*. (n.d.). Available online at: https://github.com/ultralytics/yolov5 (accessed June 5, 2022).
5. Roboflow Inc. *YOLOv5 New Version Explained [May 2022]*. (n.d.). Available online at: https://blog.roboflow.com/yolov5-improvements-and-evaluation/ (accessed June 12, 2022).
6. Ultralytics. *GitHub - ultralytics/yolov3: YOLOv3*. (n.d.). Available online at: https://github.com/ultralytics/yolov3 (accessed June 12, 2022).
7. PyTorch. *GitHub - pytorch/pytorch: PyTorchv1.11*. (n.d.). Available online at: https://github.com/pytorch/pytorch (accessed June 12, 2022).

8. Li H, Deng L, Yang C, Liu J, Gu Z. Enhanced YOLO v3 tiny network for real-time ship detection from visual image. *IEEE Access Pract Innov Open Solut.* (2021) 9:16692–706. doi: 10.1109/ACCESS.2021.3053956
9. Kim J-H, Kim N, Park Y, Won CS. Object detection and classification based on YOLO-V5 with improved maritime dataset. *J Math Sci Educ.* (2022) 10:377. doi: 10.3390/jmse10030377
10. Tang G, Zhuge Y, Claramunt C, Men S. N-YOLO: a SAR ship detection using noise-classifying and complete-target extraction. *Remote Sens.* (2021) 13:871. doi: 10.3390/rs13050871
11. Liu R, Wang T, Zhou Y, Wang C, Shan G, Snoussi H, et al. A satellite image target detection model based on an improved single-stage target detection network. *Proceedings of the 2019 Chinese Automation Congress (CAC).* Hangzhou: (2019). p. 4931–6. doi: 10.1109/CAC48633.2019.8997495
12. Zhang X, Yan M, Zhu D, Guan Y. Marine ship detection and classification based on YOLOv5 model. *J Phys Conf Ser.* (2022) 2181:012025. doi: 10.1088/1742-6596/2181/1/012025
13. Ting L, Baijun Z, Yongsheng Z, Shun Y. Ship detection algorithm based on improved YOLO V5. *Proceedings of the 2021 6th International Conference on Automation, Control and Robotics Engineering (CACRE), Shanghai, China, September 23-25, 2022.* Shanghai: (2021). p. 483–7. doi: 10.1109/CACRE52464.2021.9501331
14. Han K, Wang Y, Tian Q, Guo J, Xu C, Xu C, et al. GhostNet: more features from cheap operations. *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, June 13-19, 2020.* Seattle, WA: (2020). p. 1577–86. doi: 10.48550/arXiv.1911.11907
15. Zhou S, Yin J. YOLO-Ship: a visible light ship detection method. *Proceedings of the 2022 2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE), January 14-16, 2022, Guangzhou, China.* Guangzhou: (2022). p. 113–8. doi: 10.1109/ICCECE54139.2022.9712768
16. Qiao D, Liu G, Dong F, Jiang S-X, Dai L. Marine vessel re-identification: a large-scale dataset and global-and-local fusion based discriminative feature learning. *IEEE Access Pract Innov Open Solut.* (2020) 8:27744–56. doi: 10.1109/ACCESS.2020.2969231
17. Ghahremani A, Kong Y, Bondarev E, de With PHN. Multi-class detection and orientation recognition of vessels in maritime surveillance. *Econ Inst.* (2019) 31:266–261. doi: 10.2352/ISSN.2470-1173.2019.11.IPAS-266
18. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C, et al. SSD: single shot multibox detector. *ArXiv [Preprint]* (2015):doi: 10.48550/arxiv.1512.02325
19. Tang G, Liu S, Fujino I, Claramunt C, Wang Y, Men S, et al. H-YOLO: a single-shot ship detection approach based on region of interest preselected network. *Remote Sens.* (2020) 12:4192. doi: 10.3390/rs12244192
20. Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans Pattern Anal Mach Intellig.* (2017) 39:1137–49. doi: 10.1109/TPAMI.2016.2577031
21. Dai J, Li Y, He K, Sun J. R-FCN: object detection via region based fully convolutional networks. *ArXiv [Preprint]* (2016):doi: 10.48550/arxiv.1605.06409
22. Tan M, Pang R, Le Q. EfficientDet: scalable and efficient object detection. *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, June 13-19, 2020.* Seattle, WA: 10778-10787 (2020). doi: 10.1109/CVPR42600.2020.01079
23. Iancu B, Soloviev V, Zelioli L, Lilius J. ABOships—An inshore and offshore maritime vessel detection dataset with precise annotations. *Remote Sens.* (2021) 13:988. doi: 10.3390/rs13050988
24. Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft COCO: common objects in context. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T editors. *Proceedings of the European Conference on Computer Vision (ECCV), Zurich, Switzerland, September 6-12, 2014.* (Vol. 8693), Zurich: (2014). p. 740–55. doi: 10.1007/978-3-319-10602-1_48
25. Google. *GoogleColaboratory(Version2022/5/20)[Computer software].* Mountain View, CA: Google (2019).