Target Localization in Synthetic Aperture Sonar Imagery using Convolutional Neural Networks

Thibaud Berthomier, David P. Williams, Samantha Dugelay
NATO STO Centre for Maritime Research and Experimentation (CMRE), La Spezia, Italy
thibaud.berthomier@cmre.nato.int

Abstract—Automatic target recognition (ATR) in synthetic aperture sonar (SAS) is usually performed in two stages: object detection and target classification. The detector aims to localize all the potential targets whereas the classifier distinguishes between real targets and false alarms. The probability of detection at this first stage must be the highest as possible to ensure that targets are not missed. Unfortunately, this generally implies a significant false alarm rate. Therefore, the challenge of the second stage, classification, is to drastically reduce the number of false alarms while keeping the detected targets. Using a large database of SAS images, efficient CNN classifiers have been demonstrated for underwater target classification tasks. In this paper, we suggest applying a pretrained classification CNN for localizing targets in SAS images. In so doing, we show the feasibility of target detection and classification in one-step using CNNs.

Index Terms—Object Detection, Classification, Convolutional Neural Networks (CNNs), Synthetic Aperture Sonar (SAS)

I. INTRODUCTION

Convolutional neural networks (CNN) have recently achieved state-of-the-art performance on a wide range of object detection tasks [1]. Object detection networks are capable of localizing and identifying multiple objects in an image. To do so, two methods are generally used: two-stage (Figure 1a) and single-stage (Figure 1b) detectors. The former approach, in the first stage, finds a set of regions of interest and, in the second stage, classifies each proposal region into object categories [2]. The single-stage detector divides the image into a rectangular grid of cells and determines if any part of a target falls within the cells [3], [4]. Nowadays, object detection networks are used in many applications such as real-time image analysis [2]–[4], facial recognition [5], autonomous driving [6], [7] and license plate recognition [8]. In this paper, we address object detection in the context of underwater target localization using synthetic aperture sonar (SAS) imagery.

The principle of SAS consists of combining successive pings coherently along a known track to produce high-resolution imagery of underwater environments [9]. Indeed, this technique produces images down to centimeter resolution out to hundreds of meters in range. The produced images of the seafloor can be exploited for many underwater applications such as mine countermeasures, pipeline inspection, archaeology and imaging of wrecks. Automatic target recognition (ATR) using SAS imagery is usually performed in two stages: region proposal and object classification algorithms. In previous works [10], the first stage is performed using a detection algorithm: the cascaded, integral-image-based [11] or the Mondrian [12] detector. Applied to SAS images, the detector generates a set of potential objects of interest and a “mugshot” (i.e. a small image) of each object is extracted from the SAS image as shown in Figure 1a. During the second stage, the obtained mugshot images become the inputs of a classification algorithm. Using deep convolutional neural networks as the classifier, a significant improvement in performance was achieved over the traditional feature-based classification approaches [10].

In this paper, we propose to skip the region proposal stage and simply apply a CNN over the whole SAS image. Therefore, we develop a single-stage detector based on transfer learning of previously trained CNN classifiers. For the experiment, we consider a four-layer CNN followed by a fully-connected layer (see Figure 1a). By exploiting the computational properties of the convolution operator, we apply the same convolutional layers (i.e. same weights and biases) directly on the SAS image (see Figure 1b). Then, the obtained feature maps (four in our case) are fused into a single map using the weights and the bias terms of the fully-connected layer. The output of the new CNN is a prediction map where each pixel indicates the probability that a region belongs to each class under consideration (here, targets and clutter). Finally, the presence of targets and their positions in the image are estimated by applying a threshold to the obtained map. This one-stage approach is faster and simpler.

The remainder of this paper is organized as follows. Section II describes our datasets, the deep convolutional neural network architecture and the training procedure. Performance indicators and experimental results are presented in Section III. As image quality and environmental conditions on the seafloor have the largest impact on target detection performance [11]–[13], Section III-C proposes an analysis of the performance as a function of seabed type. Finally, Section IV provides concluding remarks and directions for future research.
II. CONVOLUTIONAL NEURAL NETWORKS FOR SAS IMAGES

A. General overview

A CNN is a deep learning approach which is particularly efficient for image analysis. The power of CNNs derives from its great representational capacity when a sufficient amount of labeled data is available. In the computer-vision community, their recent success is partly due to the emergence of large scale annotated databases, such as ImageNet [14] for image classification and Microsoft COCO [15] for object detection. Compared to these popular databases (several hundreds of thousands of images), the sizes of mine countermeasures (MCM) sonar datasets are relatively limited (several tens of thousands of images).

An intuitive solution would be to consider efficient existing object detection networks, such as Faster R-CNN [2], Single Shot MultiBox Detector (SSD) [3] or You Only Look Once (YOLO) [4]. These networks are already trained on large datasets (e.g. Microsoft COCO [15]) containing annotated optical images of common objects. But, the physics behind SAS imagery and the specificity of underwater targets are fundamentally different. Therefore, we propose a different approach that consists of adapting a pretrained classification CNN to our objective. The next subsection describes the CNN architecture and the training phase. Then, the following subsection explains how target detection is performed using the same CNN and presents the test dataset.

B. Classification CNN

1) Architecture:

A typical CNN architecture consists of two main parts: feature extraction and classification. The feature extraction part alternates blocks of convolution layers, non-linear activations and pooling operations. Each layer of the network produces a feature map which corresponds to an intermediate representation of the input data. Examples of feature maps produced from a SAS image are displayed in the Appendix. The output of the last layer is used as the input to one or more fully-connected layers, called dense or classification layers. In the context of underwater target classification, the CNN input is a mugshot (i.e. an output of a detection algorithm) and the output is the probability of belonging to each class under consideration (e.g. target and non-target).
Considering the size of our datasets, we design a relative small CNN (in comparison with popular CNNs used in the computer-vision community [16]–[18]) with four feature extraction blocks and one dense layer as illustrated in Figure 1a. The CNN input is assumed to be a $267 \times 267$ pixel SAS mugshot with a 15 mm resolution in each dimension. A feature extraction block consists of a convolutional layer, a Rectified Linear Unit activation (ReLU) and average pooling. The last block contains a fully-connected layer and a final (sigmoid) activation function in order to obtain the probabilities that a mugshot contains a target. The detailed architecture is given in Table I.

2) Datasets:

SAS provides high-resolution imaging of underwater scenes by combining sonar ping returns [9]. The principle is to move a sonar-equipped platform while illuminating the same spot on the sea floor with several pings. By coherent reorganization of the ping returns, a synthetic aperture image is produced with improved range and along-track resolution. A large database of SAS data has been collected by CMRE’s MUSCLE autonomous underwater vehicle (AUV) during many sea trials conducted between 2007 and 2015 in various geographical locations (see Table II). The datasets are quite diverse in terms of seabed characteristics (sand, silt, clay, ripples, vegetation, rocks, etc.) and mine-like objects (e.g. truncated cones, cylinders, spheres and rocks).

The SAS system has a center frequency of 300 kHz and a bandwidth of 60 kHz. It provides complex-valued imagery with an along-track resolution of 25 mm and a range resolution of 15 mm. As shown in Figure 1, a scene-level SAS image typically spans 50 m in the along-track direction and 110 m in the range direction. Firstly, the Mondrian detection algorithm [12] was applied to all the training data in order to generate a set of alarm mugshots to be classified (see Figure 1a). The detector inputs are the magnitude images interpolated by converting pixels into squares covering 15 mm in each dimension and then normalized between 0 and 40. To satisfy the requisite CNN inputs, a second normalization step transforms the pixel values from [0, 40] to [-1, 1].

3) Training phase:

Training the network simply means learning the filters, and the associated bias terms, of the convolution and the dense layers. In our case, the CNN contains 1509 trainable parameters. Using the TensorFlow software library [19], the training process was performed with the RMSprop optimizer combined with the binary-cross-entropy loss function. As the two classes are unbalanced, 32 mugshots of each class were selected to form a batch. Data augmentation (that respects the data properties) was applied on each chosen mugshot to improve the robustness of the CNN. The presented CNN was trained for 100 epochs of 1000 batches.

More details on the training process and the CNN performance are contained in [20]. This pretrained CNN serves as the basis for the design of our object detection algorithm described in the following subsection.

C. Object detection CNN

Object detection aims to localize and classify existing objects in a larger image by labeling them with rectangular bounding boxes and associated class probabilities [1]. To do so, the developed method first computes a prediction map using the classification CNN and then provides the positions of potential targets in the scene-level SAS image.

1) Prediction map:

Like the classification CNN, the new CNN is designed with four feature extraction blocks and a classification one. Despite the input size being much larger than before (typically $3335 \times 7333$ pixels), the first four blocks remain unchanged. Indeed, convolutional, pooling and activation operators are independent of the input size as long as it is larger than the initial one ($267 \times 267$ pixels). By contrast, the last block,
the classification one, has to be slightly customized in order to produce a prediction map instead of a probability score. Initially, the output of the convolutional blocks is a tensor with four elements ($1 \times 1 \times 4$) but now it will be a tensor $T$ with four features maps (each of which typically contains $48 \times 111$ pixels). The goal of the dense layer is thus to combine the four feature maps into a single one.

A dense layer is a linear operation where each output neuron is connected to all the neurons from the previous layer. The layer is determined by a weight matrix $W$ and a bias vector $b$. In our case, the dense layer returns a single output by linearly combining the four values produced by the convolutional blocks. Therefore, the weight $W$ is a vector of size 4 (see Table I) and the bias $b$ is a scalar. To compute a prediction map $Y$ using the same weight $W$ and bias $b$, the tensor $T$ of size $M \times N \times 4$ is reshaped into a matrix $X$ of size $L \times 4$ where $L = M \times N$:

$$Y = X \cdot W + b$$

where $\cdot$ denotes the matrix product. The output $Y$ of the dense layer is then reshaped into a matrix of the desired dimensions $(M \times N)$. Finally, the sigmoid activation function is applied to obtain the final prediction map.

2) Resolution:

The CNN was initially built for an input of size $267 \times 267$, so each pixel of the obtained prediction map indicates the probability that a cell of size $267 \times 267$ contains a target. Moreover, the pooling operations in the CNN decrease the resolution of the SAS image (15 mm $\times$ 15 mm) by a factor of 64 ($4 \times 4 \times 2 \times 2$). In other words, the described CNN will return the same prediction map that a map built using the initial CNN, a sliding window of size $267 \times 267$ and a stride of 64 pixels would. The default resolution of a prediction map is thus 96 cm $\times$ 96 cm.

During the training phase, the input data are small images centered on the detection position (data augmentation with a random translation of maximum 1 m was applied). With cell sizes of 96 cm $\times$ 96 cm, a target in the SAS image may not be located exactly in the center of a cell; the associated score will therefore likely be lower than the one obtained for a cell centered on the target. In order to improve the prediction score and thus the probability of detection, we propose a simple method to increase the resolution of the prediction map. The principle is illustrated in Figure 2.

To increase the default resolution by a factor of two, for instance, the CNN is applied four times on the same SAS image with a different offset each time. The offset is fixed as half of the stride arising from the pooling operation (64 pixels). Then, the four obtained prediction maps are rearranged in order to build the final map. The same process can be performed to increase the resolution by a power of 2. Figure 3 shows the prediction maps at higher resolution (48 cm $\times$ 48 cm and 12 cm $\times$ 12 cm) for the image shown in Figure 1b. We can observe in Figure 3 that the number and the intensity of the pixels corresponding to the two targets are now higher than on the map at the basic resolution. The target will thus be easier to detect. The main drawback is that this method is time consuming: it takes 4 or 64 times longer to obtain the maps in Figure 3a and in Figure 3b.
3) Localization:

The previous subsections explain how we compute a prediction map using a classification CNN on a SAS image. The obtained map represents the probability that a target exists in each cell of the original SAS image. As we can observe in Figure 1b or 2, the presence of an object on the seafloor will probably activate more than one cell (i.e., multiple cells will have a probability higher than a fixed threshold). To localize the targets in the image, we implement an iterative method whose principle is given by the pseudo-code in Algorithm 1 below:

**Algorithm 1:** Iterative maximum suppression for object localization using a prediction map.

**Inputs:**
- map → Prediction map
- Θ → Threshold
- bbsize → Bounding box size in image
- ratio → Ratio between image and map
- offset → Half of initial input size of CNN

**Outputs:**
- scores → Target probability
- pos → Alarm positions in the image

Output initialization:
Get bounding box size in map: bbsize × ratio;

while max(map) > Θ do
  Add max(map) to scores;
  Get max(map) position P_map in map;
  Get position in image: P_image = P_map / ratio + offset;
  Add P_image in pos;
  Define a box B around P_map in map;
  Set to 0 all pixels inside B;
end

For a given threshold and bounding box size, Algorithm 1 returns the positions (and the associated probabilities) of all potential targets present in a SAS image from its prediction map. In practice, the size of the bounding box is fixed to be the size of the initial CNN input (i.e., 267 × 267 pixels). The offset defined as an input is due to the absence of padding on the SAS image: the first pixel of the prediction map corresponds to the center of the first cell (or “window”). Figure 4 shows the five highest-scoring alarms obtained from the prediction map of Figure 1b. In this example, the localization algorithm would return both true targets if the threshold was 0.5. A lower threshold, such as 0.2, will generate some false alarms due to sand ripples. The next section is devoted to the performance evaluation of our object localization CNN.

**III. EXPERIMENTAL RESULTS**

The performance of the object localization network is evaluated on sonar data collected by the same system during a more recent sea trial than the training datasets (see Tables II and III). For the experiments presented here, only SAS images containing at least one true target were considered. The first subsection introduces the performance indicators used in the second and third subsections.

**Table III: Test dataset details.**

<table>
<thead>
<tr>
<th>Trial</th>
<th>Year</th>
<th>Location</th>
<th>Scenes with targets</th>
<th>Targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAN2</td>
<td>2014</td>
<td>Bonassola, Italy</td>
<td>422</td>
<td>469</td>
</tr>
</tbody>
</table>

**A. Performance indicators**

As our approach aims to perform detection and classification in one stage, the performance indicators are a combination of the metrics used to evaluate detectors and classifier in ATR. The proposed metrics, based on the parameters detailed in Table IV, are given by:

\[
\text{Precision} = \frac{\text{Hits}}{\text{Predicted targets}} = \frac{\text{TP}}{\text{TP} + \text{FP}}
\]
\[
\text{Recall} = \frac{\text{Hits}}{\text{True targets}} = \frac{\text{TP}}{\text{TP} + \text{FN}}
\]
\[
\text{False alarm rate} = \frac{\text{False alarms (f.a.)}}{\text{Maximum number of f.a.}} = \frac{\text{FP}}{\text{TN} + \text{FP}}
\]
\[
\text{False alarms per image} = \frac{\text{False alarms}}{\text{SAS images}} = \frac{\text{FP}}{422}
\]

**Table IV: Confusion matrix for detection and classification in one stage.**

<table>
<thead>
<tr>
<th>True classes</th>
<th>( C_1: ) Target</th>
<th>( C_0: ) Clutter</th>
</tr>
</thead>
<tbody>
<tr>
<td>( H_1: ) Target</td>
<td>Hit or True Positive (TP)</td>
<td>Miss or False Negative (FN)</td>
</tr>
<tr>
<td>( H_0: ) Clutter</td>
<td>False alarm or False Positive (FP)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>
Fig. 5: Object detection performance depending on the map resolution $R_i = R_0 \times 2^{-2i}$ with $R_0 = 96 \times 96$ cm$^2$: the stars and the circles on the curves indicate the performances obtained with a threshold of 0.5 and 0.7, respectively.

Fig. 6: Impact of the environment (benign or complex seabed) on performance at two different resolutions $R_0$ and $R_3$ (see Figure 5): $C_i$ corresponds to the label of the curve $i$. 

(a) Precision versus thresholds.

(b) Recall (or detection probability) versus thresholds.

(c) False alarms per image (total number of false alarms divided by the number of images and the proportion of the seabed category) versus thresholds.

(d) ROC-like curve: recall versus false alarms per image.
The precision shows the model’s ability to localize relevant targets, while the recall (also called probability of detection) reflects the ability to localize successfully a target (i.e. to not miss a target). The maximum number of false alarms is computed using Algorithm 1 with a threshold fixed to zero (minus the number of true targets). Precision, recall and false alarm rate are calculated for various thresholds on the prediction map. However, instead of the false alarm rate, the number of false alarms per image is plotted as it is more useful in our context. Finally, the accuracy of our model is also quantified by calculating the area under the curve (AUC), where the curve in question is the receiver operating characteristic (ROC) curve. The ROC curve is obtained by plotting the false alarm rate and the probability of detection as the threshold is varied.

B. Detection/classification performance

Figure 5 presents the experimental results computed for four resolutions of the prediction maps: $R_0 = 96 \times 96$, $R_1 = 48 \times 48$, $R_2 = 24 \times 24$ and $R_3 = 12 \times 12$ cm$^2$. The obtained AUC scores are close to the AUC of a perfect detector/classifier (i.e. close to 1). Nevertheless, considering the number of false alarms per image instead of the false alarm rate, we can observe that our model still has potential for improvement. For instance, with a fixed threshold of 0.5 (indicated by a star on the plots), the CNN has a recall of 0.84 and a precision of 0.23 (about 3 false alarms per image). The results also confirm that a higher resolution improves the recall (Figure 5b) but at the expense of more false alarms (see Figure 5c), and thus a lower precision (see Figure 5a). At a fixed threshold of 0.5, doubling the resolution in both dimensions ($R_0$ to $R_1$) improved the recall by 0.05 but doubled the number of false alarms. At the highest computed resolution (i.e. $R_3$) and the same threshold, the recall reaches 0.94 with an average of 15 false alarms per image.

C. Impact of the environment

It is well-known that environmental conditions impact target detection performance [11]–[13]. Figure 6 shows the performance achieved by the model as a function of the seabed type. The first dataset, collected in the bay of Bonassola, is mainly composed of fine sediments (silt or sand) and sand ripples but it also contains areas covered by rocks and vegetation. All SAS images with detection opportunities were manually segmented into four classes: fine sediments, ripples, vegetation and rocky area. The segmentation results are summarized in Table V. However, mine-like objects were only deployed on fine sediments (12 targets) and on ripples (8 targets) during the experiment. Therefore, the performance metrics were not computed for each type of seabed, but rather for two categories based on the complexity: (i) benign seabed (i.e. fine sediments) and (ii) complex seabed, which combines ripples, vegetation and rocky areas.

The experimental results illustrated in Figure 6 confirm the influence of the seabed composition and cover on the localization performance. As expected, the precision is much higher in benign areas than in complex seabed where the risks of a false alarm or missing a target are increased. In benign seabed, the CNN achieves a high level of performance (more than 0.90 recall for less than 2 false alarm per image). Additionally, we have observed that the majority of false alarms were due to objects or rocks with a similar shape as the targets considered for the experiments. In complex seabed, at a similar threshold, the probability of detection is lower and the number of false alarms higher, as one would expect. For instance, at the reference threshold of 0.5, the recall is around 0.73 at the basic resolution (versus 0.90 in fine sediments) which is too low for a ATR system. However, by increasing the resolution, it is possible to obtain almost the same recall in complex environments as in benign environments. The cost of this operation will be the number of false alarms (and the time to compute the map) but a decent compromise can be found using a higher threshold. For instance, with a threshold of 0.7, the CNN achieves the same recall of 0.87 with an average of 1 false alarm in benign seabed and 10 per image in complex seabed.

IV. Conclusion

In this paper, we proposed a novel way to exploit CNNs for underwater target localization tasks with SAS images. Our approach aims to detect and classify (i.e. to localize), in a single stage, using a CNN previously trained for classification tasks. A simple iterative algorithm is applied subsequently to return the positions of potential targets from the CNN outputs, called prediction maps. Our approach achieved promising performance in benign and complex seabeds. Logically, the detection ability of the model is higher in fine sediments than in sand ripples, vegetation and rocky areas. However, in complex seabed, using a higher threshold and finer map resolutions, the recall of the model can be increased while maintaining an acceptable number of false alarms per image. For future work, we will investigate methods, such as new CNN architectures, or pre- or post-processings to improve our model capacity by taking into account the environmental complexity in the prediction. Additionally, further experiments are in progress to compare our approach with the state-of-art two-stage approaches (Mondrian detector combined with a CNN classifier).
ACKNOWLEDGMENT

This work was supported by the NATO STO Centre for Maritime Research and Experimentation (CMRE) and the French Government Defence. Sonar data used for the experiments were collected by the CMRE with funding provided by the NATO Allied Command Transformation (ACT).

APPENDIX

As the filters in a CNN are often a source of mystery, this appendix proposes to examine the learned filters and the intermediate responses. Figure 7 shows each convolution filter (without the bias terms) learned during the training phase. Then, Figure 8 displays the output of each layer (convolution, pooling and activation) for two different mugshots: one with a target and one with seabed only. We can observe in Figure 8 that the fourth column of filters (shown in Figure 7) seems to extract features related to the target. In fact, instead considering a mugshot of a cylinder would show that the second column of filters is also activated by target highlights. The first column appears to be more sensitive to pixels of low intensity such as seafloor background or in the shadow of a target.

REFERENCES


Fig. 7: Learned filters of the four convolution layers of the CNN: each subfigure has a common colorscale in which the yellow corresponds to one and dark blue to zero.
Fig. 8: For the two mugshots at the top, each intermediate response at each layer of the CNN is given with the same colorscale as Figure 7: the final probabilities of belonging to the target class are 0.8172 for the true target, and 0.0013 for the clutter alarm.