End-to-End Object Detection and Recognition in Forward-Looking Sonar Images with Convolutional Neural Networks

Matias Valdenegro-Toro
School of Engineering & Physical Sciences
Heriot-Watt University, EH14 4AS, Edinburgh, UK
Email: m.valdenegro@hw.ac.uk

***Abstract*—Object detection and recognition are typically stages that form part of the perception module of Autonomous Underwater Vehicles, used with different sensors such as Sonar and Optical imaging, but their design is usually separate and they are only combined at test time. In this work we present a convolutional neural network that does both object detection (through detection proposals) and recognition in Forward-Looking Sonar images and is trained with bounding boxes and class labels only. Convolutional layers are shared and a 128-element feature vector is shared between both tasks. After training we obtain 93% correct detections and 75% accuracy, but accuracy can be improved by fine-tuning the classifier sub-network with the generated detection proposals. We evaluated fine-tuning with a SVM classifier trained on the shared feature vector, increasing accuracy to 85%. Our detection proposal method can also detect unlabeled and untrained objects, and has good generalization performance. Our unified method can be used in any kind of sonar image, does not make assumptions about an object's shadow, and learns features directly from data.**

I. Introduction

Autonomous Underwater Vehicles (AUVs) are now commonly being used for multiple tasks that require advanced object detection and recognition capabilities, but for coastal and deep sea operations acoustic sensing is a requirement for acceptable results. Interpreting acoustic images is difficult due to shadows, noise, and reflections, which makes detecting and recognizing objects very challenging. Forward-Looking Sonar (FLS) can provide higher resolution images (compared to lower frequency sonar), however its interpretation and analysis is still a challenge.

Common approaches for object detection in sonar images include different forms of template matching (TM) [1] [2], while object recognition is performed with a variety of engineered features combined with a classifier [3]. Both stages are used together in a combined pipeline, but generally the design and training of the object detector and classifier are separate processes.

Combining the object detector and recognition stages could have computational benefits, since features could be shared and computed only once [4], as well as improve detection and/or recognition performance. Training an end-to-end system is usually easier as well.

*This work has been partially supported by the FP7-PEOPLE-2013-ITN project ROBOCADEMY (Ref 608096) funded by the European Commission.

Convolutional Neural Networks (CNNs) are state of the art for both object detection and recognition in optical images, with very good results in the ImageNet [5] and COCO [6] datasets, but their use in analysis of sonar images has been limited. CNNs provide an ideal theoretical framework to build a end-to-end system. If both detection and recognition tasks can be represented as layers or sub-networks of a complete neural network, then combined training using stochastic gradient descent would realize an end-to-end system.

In this work we propose a network architecture that combines object detection and recognition into one system, with one image input, two outputs, the class probabilities for recognition, and the objectness score for object detection. Convolutional layers are shared and the complete network is trained end-to-end from labeled examples.

Our contributions are an end-to-end system for object detection and recognition in sonar images that can easily be trained from labeled examples for both tasks are learned directly from data and no manual feature engineering is required. Our system does not make assumptions on object shape, or the presence of a shadow, and is able to generalize very well to unseen examples. In particular our object detector, based on detection proposals, can detect unlabeled and untrained objects without major issues and can be used in any kind of sonar image.

II. Related Work

There is a rich literature on both object detection and recognition in sonar images, but a small fraction of them can be considered end-to-end or unified systems.

Detectors based on Haar Features and boosted cascades can be considered end-to-end. This kind of methods use a cascade of weak classifier trained using Adaboost, typically with Haar features, since they are very fast to compute. Sawas et al. [7] uses such approach for mine-like object detection on synthetic aperture sonar, with 3 different classes, while Sawas et al. [8] uses the same technique for sidescan sonar imagery. In both cases detection recall is very high but the number of generated false positives is also large.

This kind of techniques can only be used with objects that have a large shadow, as it matches the Haar feature. Training with different objects require complete re-training

of the cascade, while a other methods only to re-train the classifier part [9].

Template matching methods [1] can also be considered end-to-end, as matching a template to an input image implicitly includes class information. Myers et al. [1] uses a custom similarity metric on a segmented (shadow, highlight and background) sonar image to detect mines with high precision and recall. Hurtos et al [2] uses cross-correlation similarity to detect and recognize submerged chain links. Typically TM is used when the number of classes is low, the objects are not deformable, and fine-grained recognition is not needed, but the generalization performance of such methods is low.

Convolutional Neural Networks have been used to construct end-to-end systems on optical images, with the most well known being Fast R-CNN [4] and Faster R-CNN [10]. Fast R-CNN uses selective search [11] to generate detection proposals over the input image. A set of convolutional feature maps (shared between all images) are used to generate a feature vector that can be used for multi-class classification and bounding box adjustment with respect to the proposal that generated it.

Faster R-CNN use Region Proposal Networks that generate detection proposals by predicting bounding boxes and objectness scores [12] directly with a sliding window CNN. Convolutional feature maps are shared between the detection and classification networks. Both Fast and Faster R-CNN are state of the art in ImageNet [10] and COCO object detection benchmarks [6].

A similar but different CNN-based model is YOLO [13], which is a unified (end-to-end) detection and classification system for optical images. The input image is divided into a 7×7 grid, and ground truth objects are assigned to cells. Each cell predicts a bounding box, objectness, and a probability distribution over classes. Objectness can then be thresholded to generate detections that include class information. This system has slightly worse performance than competing methods (like Faster R-CNN) but it is 100-1000 times faster, running in real-time on GPU.

Bounding box predicting methods using CNNs perform very well on optical images where large datasets are available (thousands of training samples per class), but they do not perform adequately or fail to train in smaller datasets, like the ones available in the underwater community. The dataset that we use for this paper has only around 100-200 training samples per class, and predicting bounding boxes directly with a CNN is not possible. Sonar images also contain less information than color images, as a typical sonar image is only one channel (acoustic return), while color images are three channels (red, green and blue).

These limitations motivate us to try different CNN-based models for end-to-end object detection and classification in sonar images. CNNs are ideal for the task, since they are well known for their ability to generalize outside of the training set, and for fine-tuning of the classification layers to target a different set of objects.

III. End-to-End Detection and Recognition with CNNs

A. Network Architecture

The basic design of our approach is shown in Fig. 1. A FLS image is input to a system that extracts convolutional features, which then are transformed into a feature vector that is used both by an object classifier and a detector to make their decisions.

The concept of objectness [12] is related to detection proposals. A proposal detector is an generic (or class agnostic) object detection system that can be used to detect many kinds of objects in an image without re-training [15]. We use a CNN to predict detection proposals in a sonar image, and we add a CNN-based classification system.

We have previously shown how CNNs can be used to generate detection proposals in FLS imagery [14]. We propose a similar model where we slide the CNN input across the sonar's field of view, and each sliding window has two decisions to be made:

- **Objectness**. This is defined as a score that measures the "membership" to an object class versus the background class [10]. We use objectness scores in the $[0, 1]$ range, where zero means the window contains just background, and one is interpreted as the window containing a complete object. Windows that contain partial objects get an objectness between zero and one. Our object detection system outputs an objectness score that can be thresholded to obtain a binary decision of whether the window contains an object or not. This is used for the object detection task.
- **Object Class** This is a prediction of which class the object belongs to. Windows with low objectness should be classified as background, while high objectness corresponds to one of the object classes. This is the object recognition task.

For each sliding window in the image (inside the sonar's field of view), we only output windows that have objectness greater or equal to the objectness threshold T_o. This parameter allows the operator to control the number of generated detections and their minimum confidence values. The system is trained to produce high objectness for ground truth objects, and low values for background.

We implement the design of Fig. 1 with a Convolutional Neural Network [16]. This kind of network is ideal for its well known performance in image recognition, and it can be easily designed to consider objectness and class information in an end-to-end fashion.

Our neural network architecture is shown in Fig. 2. This network has one input image and two outputs, objectness and a probability distribution over class labels, including background.

We use the following notation. Conv(N_f, $F_w \times F_h$) is a convolutional layers with N_f filters of width F_w and height F_h. MP(P_w, P_h) is a max-pooling layer with sub-sampling size of $P_w \times P_h$, and FC(n) is a fully connected layers with n output neurons.

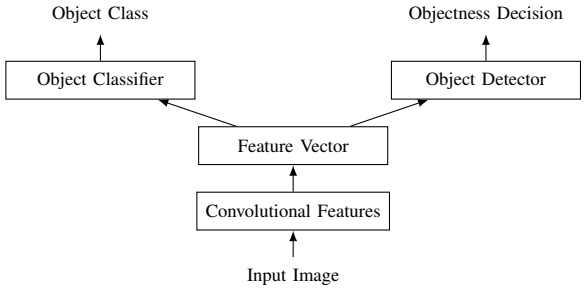


Fig. 1. Basic Architecture. An image is input to a classification system with two outputs. A shared set of convolutional features are computed, producing a feature vector that is used by both the object classifier and object detector to produce their decisions.

The architecture consists of three sub-networks. The main sub-network has shared convolutional layers, in a Conv(32, 5×5)-MP(2, 2)-Conv(32, 5×5)-MP(2, 2)-FC(128) configuration. The 128-element feature vector computed by this network is then used by the two smaller sub-networks that predict objectness and class probabilities. We call this feature vector the 128-element shared feature vector.

The input image is a 96×96 window of the original FLS input image. The configuration of the two output sub-networks is:

- **Detection Sub-Network**. This network's configuration is FC(96)-FC(1). The output layer uses a sigmoid activation (Eq. 1) tp predict objectness in the $[0, 1]$ range.
- **Classification Sub-Network**. This network's configuration is FC(96)-FC(N_c), where N_c is the number of output classes, also considering one class to represent background. The activation function of the output layer is the softmax function (Eq. 2) to predict a probability distribution over class labels. The output class can be obtained by taking the class with greatest probability.

$$f(x) = (1 + e^{-x})^{-1} \tag{1}$$

$$f(\mathbf{x}) = \left(\frac{e^{\mathbf{x}_i}}{\sum_j e^{\mathbf{x}_j}} \right)_i \tag{2}$$

All other layers use the ReLU activation function (Eq. 3) [17]. In total our network has 1.8 Million trainable weights, with most of the parameters coming from the fully connected layers.

$$f(x) = \max(0, x) \tag{3}$$

B. Data Preprocessing

Our objective is to train the proposed network in a end-to-end way. This requires special preprocessing of the training data, such as computing objectness scores and selecting image windows as input data. This process only requires a dataset with labeled FLS images. We only require bounding boxes with the corresponding class annotations in each image.

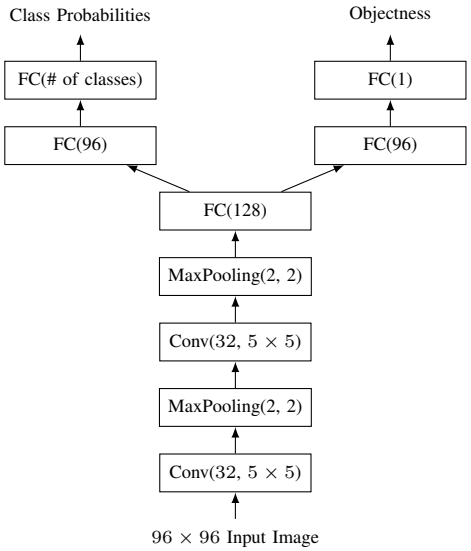


Fig. 2. Realization of the proposed architecture as a Convolutional Neural Network. This CNN has 1.8 Million trainable weights.

To generate the initial training set, for each image in a dataset, we run a 96×96 sliding window, and every window that has a intersection-over-union score (IoU, Eq. 4) with the ground truth greater than 0.5 is cropped and stored as input images during training.

$$\text{iou}(A, B) = \frac{\text{area}(A \cap B)}{\text{area}(A \cup B)} \tag{4}$$

Where A and B in Eq. 4 are rectangles. We estimate ground truth objectness with the IoU score. This is appropriate as the IoU of sliding window versus the ground truth increases as the window covers more parts of the ground truth bounding box, and very low (zero) objectness scores are produced for windows that are far away from the ground truth. Ground truth class labels come directly from labeled bounding boxes. We also randomly sample 20 windows per image that have IoU with ground truth smaller than 0.1 and label them as background.

C. Training

We now describe different aspects of the neural network architecture and the training process.

The network is trained end-to-end with both tasks of detection and recognition at the same time. This process requires the minimization of a multi-task loss function. In our case we use a linear combination of two losses. For classification we use the categorical cross-entropy loss (Eq. 5), while for objectness we use the mean absolute error (Eq. 6).

$$L_{cls}(y, \hat{y}) = -\sum_i \sum_c y_i^c \log \hat{y}_i^c \tag{5}$$

$$L_{obj}(y, \hat{y}) = n^{-1} \sum |y - \hat{y}| \tag{6}$$

The final loss function is shown in Eq. 7. The factors β and λ control the relative importance of each task in the overall

training process. We experiment with different values in the evaluation section.

$$L = \beta L_{obj} + \lambda L_{cls} \tag{7}$$

Batch normalization [18] is used after every trainable layer to prevent overfitting and accelerate training. The network is trained with stochastic gradient descent with a batch size of $b = 128$ images and the ADAM optimizer [19] with a starting learning rate of $\alpha = 0.1$. We train until the validation loss stops improving, normally after 20 epochs.

D. Fine-tuning

After initial experimentation, we found that the network can be trained to obtain acceptable recognition performance, but it was lower than initially expected. This is caused by differences between the training images generated by a simple sliding window, and the detection proposals generated by thresholding the objectness score.

We can easily bridge this gap by fine-tuning. This is the process of training a set of layers from neural network while keeping other layer's weights fixed, after initial training. No random initialization is performed and weights are initialized with previously obtained weights. This allows the tuning of some layers (usually classification) with different data or even completely retrain the classifier part of the network to target a different dataset or number of classes.

Unfortunately, we did not obtain good results with fine-tuning, most likely because of the small sample of our data (when compared to typical computer vision datasets like PASCAL VOC and ImageNet). An approach that produced much better results is to completely replace the classification sub-network with a multiclass linear SVM classifier trained on the 128-element shared feature vector. Tuning the parameters of a classification neural network is harder than just simply training a linear SVM, that only has a C as a regularization parameter.

It must be noted that the classification sub-network is required during training, if it is not included, then the 128-element shared feature vector would not contain any class information and classification performance would not be better than random chance. We explore the SVM classifier approach in the evaluation section.

IV. Experimental Evaluation

We have captured a set of 2000 FLS images with an ARIS Explorer 3000, containing 9 object classes: cans, bottles, a metal chain, drink cartons, a hook, a propeller, a tire, a valve and background. The dataset was captured in the Ocean Systems Lab's water tank and it only represents a laboratory setting. We split this dataset into 700 test images and 1300 training images. The full size FLS images in the train set are then cropped with the method described in Section III-B and a network training and validation sets are created, with again a 70%/30% split. Training data is augmented by left-right and up-down flips.

The dataset used to train the neural network consists of 157132 training 96×96 image crops, and 67343 validation images of the same size. All sliding windows were generated using a stride of $s = 8$ pixels.

We use two evaluation metrics:

- **Detection Recall**. Recall is defined as the fraction of correct detections to the number of ground truth bounding boxes. This is an indication of how well a detector can detect objects. Typically also precision is evaluated, but since our detection proposals also generate bounding boxes over unlabeled objects, recall is preferred for this kind of algorithm. We consider a proposal as a correct detection if the IoU score with any ground truth bounding box is greater than 0.5.
- **Classification Accuracy**. Accuracy is the fraction of correctly classified objects to the number of ground truth objects. Note that for an object to be correctly classified, it must be detected first, so classification accuracy is bound by the detection recall.

Both metrics are evaluated as a function of the objectness threshold $T_o \in [0, 1]$.

A. Quantitative Evaluation

We trained five different networks, varying the multi-task loss weights (β and γ) in order to measure its effect. Our principal results are shown in Fig. 3.

Recall is high in all configurations, but the classification accuracy varies considerably with the loss weights. Configuration $\beta = 3, \gamma = 1$ has the best accuracy, giving adequate performance of 75% accuracy, and 93% recall, both measured at $T_o = 0.5$. The number of detection proposals generated per image also varies considerably.

The best classification model generates around 100 proposals per image at the same $T_o = 0.5$ threshold. This large amount of proposals has two causes: detection of unlabeled and untrained objects, and clustering of detections around ground truth objects. Considering that our dataset has one to three objects per image, from Fig. 3 one can deduce that the number of false positives is pretty large. This is explained by multiple clustered detections around each object, and the generation of detection proposals in unlabeled objects present in our dataset. Real detections generated over background are rare.

Significant drops of recall and accuracy happen after $T_o = 0.6$, and very little detection proposals are generated past $T_o = 0.8$. Since we approximated objectness with the IoU score with ground truth, this shows the limitations of our model. The use of a single scale is also a root cause and this problem could be solved by using multiple scales or a scale invariant approach.

B. Fine-Tuning

We also explore the fine-tuning option with a SVM classifier. After training the $\beta = 3, \gamma = 1$ configuration, we run the trained detector on the original 1300 full-size FLS images and crop all detection proposals that have at least 0.5 IoU with ground truth. The 128-element shared feature vector is saved for all such instances and a multi-class SVM classifier is trained with $C = 1$. The one-versus-one approach is used to extend a binary SVM to a multi-class problem.

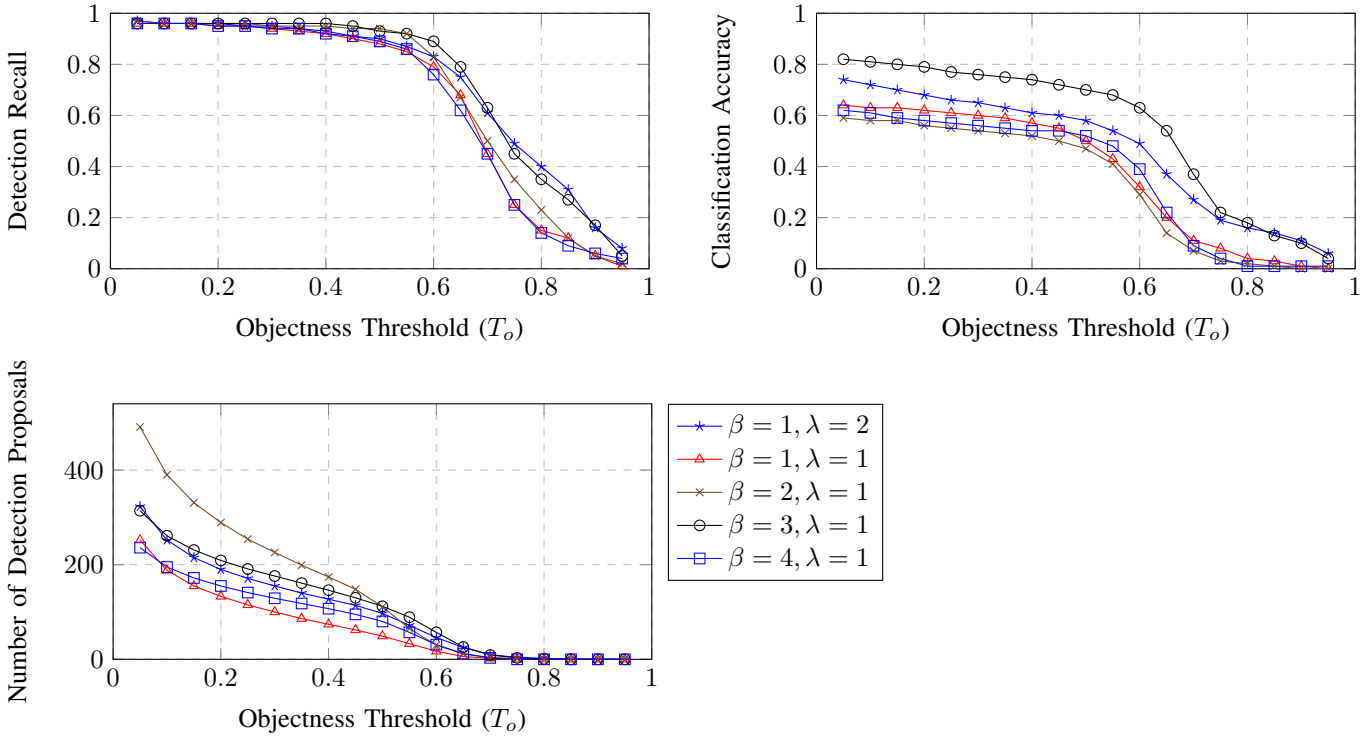


Fig. 3. Detection Recall, Classification Accuracy and Number of Detection proposals versus Objectness Threshold T_o. Different value combinations of the multi-task loss weights β and γ are shown.

Accuracy is then re-evaluated and results are shown in Fig. 4. Fine-tuning in this case has a considerable positive effect on classification accuracy, increasing from 75% to 85% at $T_o = 0.5$. But as T_o increases, accuracy drops in the same pattern as before fine-tuning. We were not able to fine-tune after $T_o = 0.65$ since some classes did not have enough correct detections, lowering the number of training samples available for fine-tune to less than one.

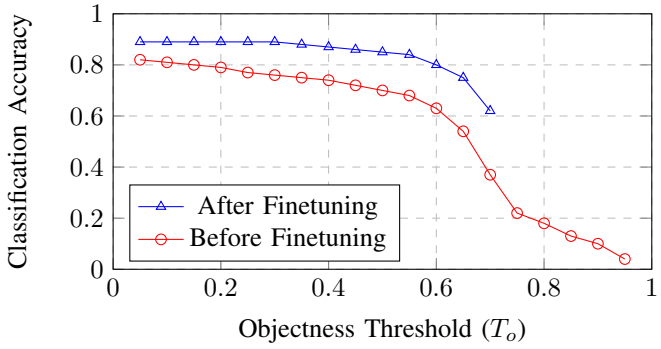
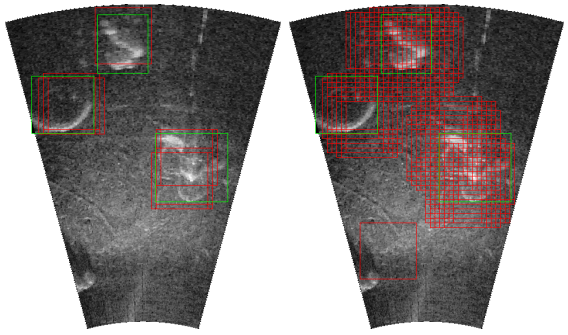


Fig. 4. Accuracy of model $\beta = 3, \lambda = 1$ before and after fine-tuning with a SVM classifier. Fine-tuning produces a considerable increase in accuracy, but increasing T_o has the effect of reducing the number of training samples. For $T_o > 0.65$ training fails and it is not shown.

C. Qualitative Evaluation

A set of matching detections in one test image is shown in Fig. 5a. In this example all objects are correctly detected



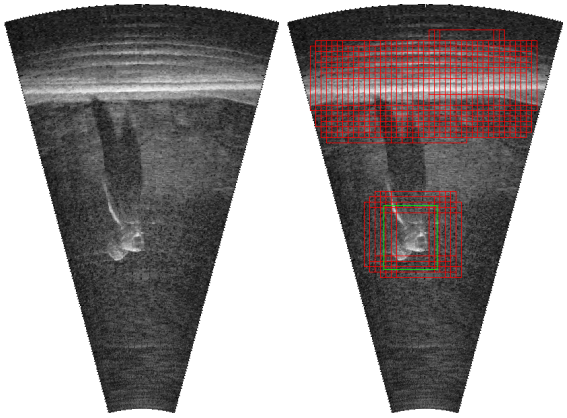
(a) Matching Detections (b) All Detection Proposals

Fig. 5. Detections on one sample image, generated at $T_o = 0.4$. Note that there is only one background false positive detection on the lower left corner.

and recognized as tire, hook and propeller. Fig 5b shows all detections that were generated in that image, with many overlapping detections that correspond to partial objects.

Fig. 7 shows detections in one test image as we vary the objectness threshold T_o. Large numbers of detections are generated around each ground truth object, some of them are duplicate detections, but many of these correspond to other kinds of objects that were unlabeled in our dataset since it was not clear which object class they belong to.

Detections usually cluster around ground truth objects, and post-processing methods, such as non-maximum suppression, can be used to remove duplicate detections. Such kind of detection clustering is also inherent to the low IoU threshold



(a) Input Image (b) All Detection Proposals

Fig. 6. Detection of Unlabeled and Untrained objects. In (a) wall reflections (top) and a propeller (middle) are shown, and (b) shows how our detection proposal system can generate detections on the wall without previous training on such kind of object.

(0.5) that was used to construct the network's training set. False detections over background are very rare (considering unlabeled objects). Fig. 5b shows one such false detection in the lower-left corner.

Fig. 6 shows a wall and its classic reflections, and we can successfully generate detections on the wall without training the detector with labeled wall images. Note that while detections are generated over the wall, predicted classes are wrong, as we did not train the classifier with that object class.

Detecting unlabeled and untrained objects is a powerful property of our detection proposal system, as only a better classifier (or more training data) is required to filter our undesired detections, but retraining the detector to target different object classes is not necessary.

V. Conclusions and Future Work

In this paper we have presented a CNN-based approach to build an end-to-end system for object detection and recognition in FLS images. We have made no hard assumptions on object shape, and our system even does not need objects to have shadows or to segment shadow and highlights. The only requirements from labeled data is object bounding boxes and class labels.

Typical object detection and recognition methods in FLS images take strong assumptions on the kind of object that can be detected, and have issues generalizing outside of the training set. Most techniques used on sonar images use detection and recognition stages that were trained separately. In this paper we propose a system where both stages can be represented as sub-networks in a bigger neural network, and thus trained together, sharing layers and a common feature vector. We have found no previous research that used CNNs for that purpose on this kind of data.

In our dataset of seafloor marine debris, we can obtain 93% detection recall with 75% classification accuracy, generating approximately 100 detection proposals per image. Classification accuracy can be further improved with fine-tuning by either re-training the classifier on images produced by the detection system, or replacing the classifier layers with a multi-class SVM. We evaluated the latter option and obtained an absolute 10% increment in accuracy, up to 85%.

Our method only uses a single scale, so detecting objects with considerable size variation would not be possible. We are currently researching how to include multiple scales in our end-to-end system. Improving accuracy is also important, and more training data will have that effect.

We also generate detection proposals for many unlabeled and untrained objects present in our dataset, which is an advantage as all objects present can be detected, with little background false positives. If a high precision detector is desired, then better discrimination between unlabeled objects and background is needed.

We believe that methods based on CNNs will help improve object detection and recognition in sonar images (of any kind) for Autonomous Underwater Vehicles, and this work is one step forwards in that direction.

Acknowledgments

The authors would like to thank Leonard McLean for his help in capturing data used in this paper.

References

[1] V. Myers and J. Fawcett, "A template matching procedure for automatic target recognition in synthetic aperture sonar imagery," *Signal Processing Letters, IEEE*, vol. 17, no. 7, pp. 683–686, 2010.

[2] N. Hurtós, N. Palomeras, S. Nagappa, and J. Salvi, "Automatic detection of underwater chain links using a forward-looking sonar," in *OCEANS-Bergen, 2013 MTS/IEEE*. IEEE, 2013, pp. 1–7.

[3] R. Fandos, A. M. Zoubir, and K. Siantidis, "Unified design of a feature-based adac system for mine hunting using synthetic aperture sonar," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 52, no. 5, pp. 2413–2426, 2014.

[4] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[7] J. Sawas and Y. Petillot, "Cascade of boosted classifiers for automatic target recognition in synthetic aperture sonar imagery," in *Proceedings of Meetings on Acoustics*, vol. 17, no. 1. Acoustical Society of America, 2013, p. 070074.

[8] J. Sawas, Y. Petillot, and Y. Pailhas, "Cascade of boosted classifiers for rapid detection of underwater objects," in *ECUA 2010 Istanbul Conference*, 2010, pp. 1–8.

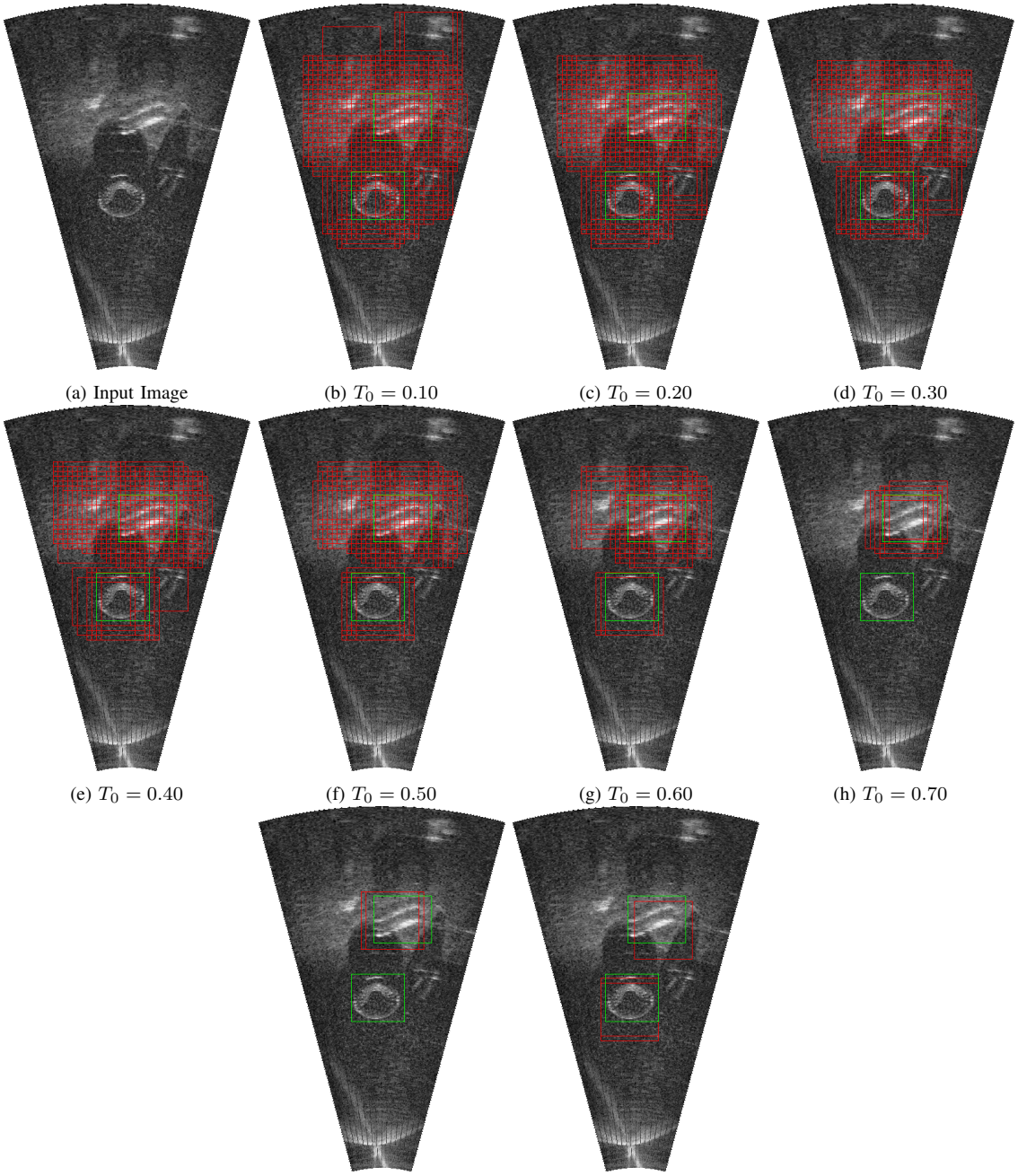
[9] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 806–813.

[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[11] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[12] B. Alexe, T. Deselaers, and V. Ferrari, "What is an object?" in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 73–80.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *arXiv preprint arXiv:1506.02640*, 2015.



(a) Input Image (b) $T_0 = 0.10$ (c) $T_0 = 0.20$ (d) $T_0 = 0.30$

(e) $T_0 = 0.40$ (f) $T_0 = 0.50$ (g) $T_0 = 0.60$ (h) $T_0 = 0.70$

(i) $T_0 = 0.80$ (j) Matches at $T_o = 0.5$

Fig. 7. Example Detections generated by model $\beta = 3, \lambda = 1$ as function of the objectness threshold T_o. Red bounding boxes represent our generated detection proposals, and green bounding boxes are the ground truth.

[14] M. Valdengro-Toro, "Objectness scoring and detection proposals in forward-looking sonar images with convolutional neural networks," in *To appear in Proceedings of the 7th IAPR TC3 Workshop on Artificial Neural Networks in Pattern Recognition*. Springer, 2016.

[15] J. Hosang, R. Benenson, and B. Schiele, "How good are detection proposals, really?" *arXiv preprint arXiv:1406.6962*, 2014.

[16] Y. Bengio, I. J. Goodfellow, and A. Courville, "Deep learning," *Available at http://www.iro.umontreal.ca/˜bengioy/dlbook*.

[17] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[19] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.