Object Recognition in Forward-Looking Sonar Images with Convolutional Neural Networks

Matias Valdenegro-Toro[1]

***Abstract*— Forward-looking sonars can provide high resolution images that can be used for different tasks in an underwater environment. However, image interpretation is still an open problem due to multiple issues inherent in acoustic imaging. In this work, we use Convolutional Neural Networks (CNN) for object recognition in forward-looking sonar images. We show that a CNN outperforms the state of the art for such kind of images by achieving an accuracy of 99.2%. While state of the art template matching methods have accuracies between 92.4% and 97.6%. We also compare the number of learnable parameters of CNNs and template matching that are required to achieve high performance. Our results show that CNNs require less parameters to provide better recognition capabilities that generalize well to unseen data.**

I. Introduction

Autonomous Underwater Vehicles (AUVs) are commonly being used for survey, inspection and exploration tasks, but object recognition capabilities of AUVs are not comparable with Land and Air vehicles due to limitations of the underwater domain, such as light scattering, attenuation, and poor visibility. Optical information is not reliable and other kinds of sensors must be used.

Forward-Looking Sonars (FLS) can capture high detail images of underwater objects and scenes at high frame rates (up to 15 Hz), independent of water turbidity and optical visibility. But the interpretation of sonar images is still a challenge due to different issues, such as acoustic shadows, noise, and reflections.

Classical approaches for object recognition in sonar images either use Template Matching (TM) [1] [2] with different similarity measures, or classifiers trained on engineered features, such as those based on shadow or highlight shape [3] [8]. Both kind of approaches can provide good performance but cannot generalize over types of objects that were not considered during feature development or are not present in the training dataset. This is especially problematic for small objects, such as bottles, cans, tires, shoes, and drink cartons. These kinds of garbage objects are present in the seafloor and are polluting the oceans. In general most methods's accuracy saturates around 95% and very rarely can reach 99%. This small gap has not been bridged, mostly due to diminishing returns.

Convolutional Neural Networks (CNNs) are state of the art for many object recognition on many benchmark datasets, such as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [4] and PASCAL VOC [5]. However these kinds of datasets consist only of optical images, and no research has been performed to evaluate the performance of CNNs on sonar imagery.

Two scientific questions motivate this paper: How well do CNNs work in FLS images? Was this kind of performance possible before, with classical Neural Networks? We explore an experimental answer to both questions.

The main contribution of this work is the design and experimental validation of a CNN that surpasses state of the art for object recognition in FLS images. We show that a CNN can recognize complex objects with human-like performance. We also show that CNNs require less parameters than template matching while having better accuracy, and being 40% faster, which allows for a real-time implementation.

II. Related Work

There is a large literature about object recognition. In the marine domain this task is also called automatic target recognition or object identification. Most approaches use either template matching [6] [1] [2] or an engineered feature extractor with a trained classifier to make the recognition decision [3] [7]. For sonar imagery, features based on shadow or highlight are popular [8].

Sawas et al. [7] proposes a cascade of classifiers based on Haar features. Adaboost is used to train such cascade and select the most relevant features. This method is very fast but recognition performance saturates around 95% and a large amount of false positives is generated.

Williams [9] uses integral images to build several kinds of maps that can be compared to produce detections. While the theoretical background of this method is unclear, its performance in synthetic aperture sonar is very good. Generalizing this kind of method to other kinds of sonar (such as FLS) is not possible.

Template Matching [6] requires the definition of a templates that indicate how objects of interest look like, and a similarity metric is used to compare the templates with the input image. A maximum similarity over the set or higher than a threshold indicates a successful recognition.

Myers and Fawcett [1] propose a template matching recognition system for synthetic aperture sonar. Templates are generated with a computer model, and both the template and input image are segmented into dark shadow, shadow, bright highlight, highlight or seabed (background). Segmentation is performed by comparing the pixel value with the image median scaled by a user-defined value. Then similarity is

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

[1]Matias Valdenegro-Toro is with Ocean Systems Laboratory, School of Engineering & Physical Sciences, Heriot-Watt University, EH14 4AS, Edinburgh, UK m.valdenegro@hw.ac.uk

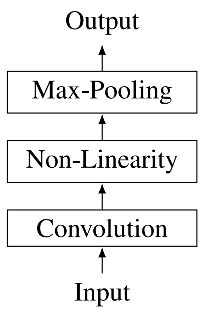


Fig. 1. Typical Convolutional Module

computed with a custom function based on cross-correlation. This system outperforms the common cross-correlation TM but in order to segment the image it takes strong assumptions about the input image.

Hurtos et al. [2] uses cross-correlation template matching to identify chain link corners, and clustering these detections allow for the detection of chain links. This system effectively recognizes chain links by matching its parts.

These methods, while performing close to 95% accuracy, are not enough. Current methods cannot achieve human level performance (close or surpassing 99% accuracy) and do not generalize between different types of sonar or even object shapes. Most of the research aims to recognize mine like objects, and little attention has been given to other objects of interest in the underwater domain.

It is well known from the Computer Vision community that feature engineering is not the best way to do object recognition [5]. Learning features directly from data works better and we explore such option in this work.

III. Convolutional Neural Networks on FLS Imagery

CNNs are a kind of neural network designed for image processing [10], where the key innovation was to learn convolutional filters from data, which are optimized for the given task. This is done by inserting convolutional layers that compute convolution with the input image instead of the typical weighted sums.

In order to apply CNNs on sonar images, we need to develop appropriate CNN architectures. We start with a popular approach that consists of ”stacking” predefined convolutional modules for feature extraction [4], followed by fully connected layers that work as a classifier [10].

A convolutional module is shown in Fig. 1. The input image is first convolved with a bank of f filters with size $W_f \times H_f$, but typically these filters are square ($W_F = H_f$). The filter weights are learned together with the rest of the neural network in a end-to-end fashion [10]. After convolution, f feature maps are output and a non-linear activation function is applied. The Rectified Non-Linear Unit (ReLU, Eq 1) is preferred due to its non-saturating properties [11], but other activations can be used. We explore the effect of activation functions in Sec. IV.

$$f(x) = \max(0, x) \tag{1}$$

After activations, a sub-sampling operation is applied. This is intended to reduce the amount of information in each feature map. Sub-sampling also introduces a small degree of translation invariance. The typical choice for sub-sampling is Max-Pooling, where an input feature map is split into $s \times s$ non-overlapping regions and the maximum value is output for each region. This reduces the size of a $w \times h$ feature map to $\lfloor \frac{w}{s} \rfloor \times \lfloor \frac{h}{s} \rfloor$. A common choice is $s = 2$, which discards approximately 75% of the information in a feature map.

Stacking several convolutional modules work as a feature extractor, where the features extracted by each module form a hierarchy. Modules close to the input image will extract low level features, while further modules will compute high level features.

A set of convolutional modules are followed by one or more fully connected layers. This is just a rewording of classic multilayer perceptrons (MLP), and a convolutional module's output is just flattened to a 1D array in order to be an acceptable input to an MLP. In this case the MLP just acts as a classifier, but the combination of convolution modules and MLP allow for learning features and classifier at the same time (end-to-end), which significantly outperform any kind of feature engineering.

Convolutional modules's key advantage is their low number of parameters. The only parameters to be learned are $f \times W_f \times H_f$ filter weights, usually with small values (up to 11) for filter sizes, and they do not depend on input sizes. In contrast, fully connected layers have $n \times m$ weights, where n is the number of neurons and m is the input feature dimensionality, but generally n is set to higher values than comparable CNNs, and m is also typically large to increase the chances of linear separability.

CNNs can be directly applied to FLS imagery. For object recognition, we have designed a CNN with 96×96 image input that correspond to object crops from a FLS image. Given that Conv*n*-*w*-*s* is a Convolutional module with *n* square filters of size $w \times w$ and Max-Pooling with subsampling size (s, s), and FC*n* is a fully connected layer with n output neurons, we propose the following CNN architectures:

- Conv32-5-2, FC64, FC10.
- Conv32-5-2, Conv32-5-2, FC64, FC10.
- Conv32-5-2, Conv32-5-2, Conv32-5-2, FC64, FC10.

For comparison purposes, we also evaluate the following MLP architectures:

- FC10.
- FC256, FC10.
- FC512, FC256, FC10.
- FC1024, FC512, FC256, FC10.

All architectures use the same activation function for all layers, except for the output layer that uses the softmax function (Eq 2). Softmax takes a vector **x** as input, in any range, and transforms it to a probability distribution.

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

The softmax function allows a neural network to output a probability distribution over the class labels, making it ideal for a multiclass classifier. The output class can be obtained by selecting the one with maximum probability. Neural networks are commonly trained with gradient descent based methods that minimize a loss function. For our classification networks we use the multinomial cross-entropy loss, as shown in Eq 3, where y is the ground truth labels and $\hat{y}$ is the predictions from the neural network, both encoded as probability distributions (one hot encoding).

$$L(y, \hat{y}) = -\sum_i y_i \log \hat{y}_i \quad (3)$$

To minimize the loss function we use mini-batch gradient descent with a batch size of $b = 32$ images , which is a variant of stochastic gradient descent (SGD). We use an initial learning rate of $\alpha = 0.1$, but since this hyperparameter is key for successful training, we also use ADAM [12], which adapts the learning rate during training automatically, based on a moving average of the gradient norms.

To prevent overfitting, we use Batch Normalization [13] after each convolutional and fully connected layer. We also evaluated the use of Dropout [14] for the same purpose, but Batch Normalization gave better generalization performance. We include a comparison of both methods in our evaluation.

IV. Experimental Evaluation

A. Data

We have captured FLS images with an ARIS Explorer 3000, at the 3.0 MHz frequency setting with ranges of 1 to 3 meters. 2000 images were captured inside our water tank containing different types of debris objects, as shown in Fig. 2. One sonar image is shown in Fig. 3, while a sample of each objects can be seen in Fig. 4. Our dataset contains 9 different classes plus one background class. Background samples are generated by randomly sampling and cropping image patches that do not intersect with ground truth.

We labeled 2627 relevant objects in this image set, and generated a dataset by cropping all object's bounding boxes and resize each crop to 96×96 pixels. We split this dataset into 70% for the training set, and 30% for the test set, which corresponds to 1838 images for training, and 789 for testing.

B. Baseline

We define our baseline as template matching (TM) [2] with cross correlation (CC) and sum of squared differences (SQD) as similarity scores. While there are better methods that are based on TM, like [1], these methods require extensive template design, and such comparison would not be fair, since CNNs learn the most appropriate features directly from training data and only require the design of an appropriate network architecture.

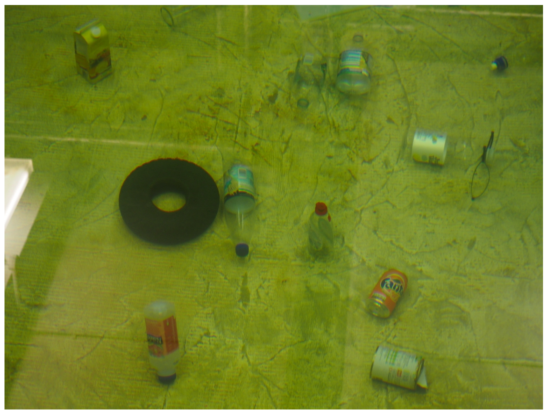


Fig. 2. Garbage objects in the our water tank

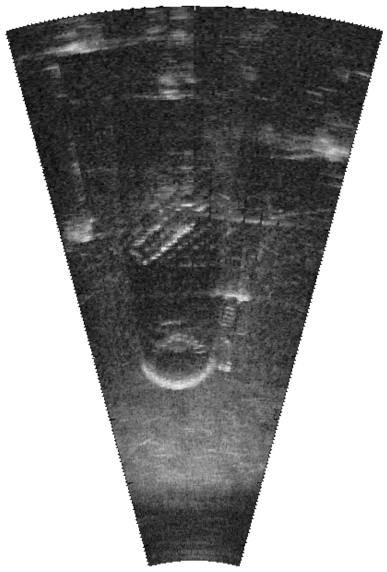


Fig. 3. FLS image of some objects (bottles and a tire)

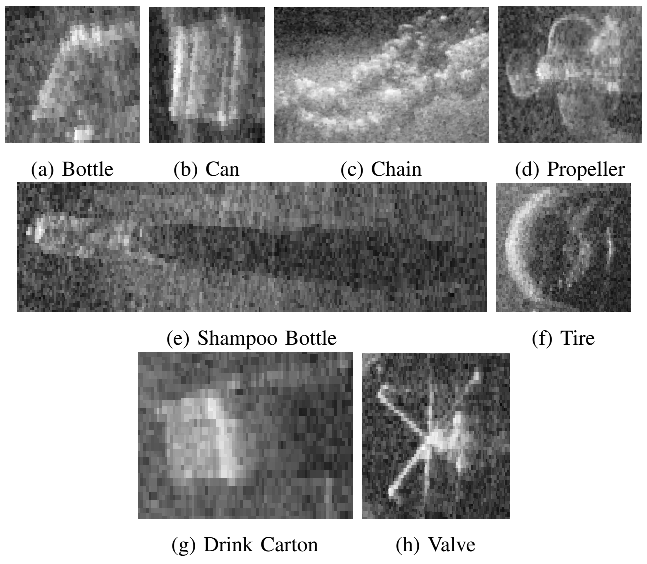


Fig. 4. Sample FLS images of objects in our dataset

$$S_{CC} = \frac{\sum(I - \bar{I}) \sum(T - \bar{T})}{\sqrt{\sum(I - \bar{I})^2 \sum(T - \bar{T})^2}} \quad (4)$$

$$S_{SQD} = n^{-1} \sum(I - T)^2 \quad (5)$$

CC similarity is defined by Eq 4, while SQD is shown in Eq 5. Here I is the input image, T is the template, $\bar{I}$ is the image average and $\bar{T}$ is the template average. As training data for TM, we randomly select a subset of the training set as templates along with their class labels. At test time, the similarity of the input image with each template is computed and the one with best similarity is selected as output class. For CC the maximum is best, while for SQD it is the minimum.

We compare template matching with a varying number of templates, since this parameter affects recognition accuracy. The number of templates is selected as the $l \in [1\%, 5\%, 10\%, 20\%, 30\%, 40\%, 50\%]$ percent of the training set. We use three comparison metrics: the test accuracy, the number of parameters and the time required to produce one prediction[1]. The number of parameters of a neural network can be directly obtained as the number of weights. For template matching, we compute the number of parameters as $n \times T_w \times T_h$, where n is the number of templates, and $T_w = 96$ and $T_h = 96$ are its width and height correspondingly.

C. Results

Table I shows our comparison of template matching versus different configurations of CNNs and MLPs. Changing the activation function produces wildly different accuracy results, but it is well known that ReLU is the best choice [4]. No MLP configuration can produce similar results to a CNN. This is due to the fact that features extracted by an MLP might not be the best for the problem and the number of parameters is too high, which leads to overfitting.

Our results show that using MLPs or CNNs with classical neural network methods will not match the accuracy produced by a modern CNN with Batch Normalization and ReLU activations on FLS images. The best CNN has an accuracy of 99.2%, which is significantly higher than what TM can provide, and it is much higher than what typically is reported on FLS images [2]. The only 7 misclassified samples are shown in Fig. 7. Two of them are background but actually correspond to a object (the ground truth is incorrect), and the rest show blurry objects that are very hard to recognize.

The difference between TM and CNNs can be explained by the number of parameters. Fig. 5 shows the relationship between accuracy and number of parameters. Our best performing CNN has one order of magnitude less parameters than the template matching methods. CNNs can provide very high accuracy with a relatively low parameter count, which we interpret that a CNN learns how to extract the most relevant features in order to successfully recognize objects, while there is a lot of redundancy in template parameters.

This redundancy cannot be reduced just by using less templates, since accuracy decreases as well, as shown in Fig. 5. MLPs need a large amount of parameters to perform well, which requires exponentially large training sets and leads to overfitting. An MLP just cannot learn the same kind of features than a CNN does, which limits their ability to generalize.

Comparing computation times (Table I), the best performing CNN also outperforms the best TM method by 40%. While both methods are very fast and allow real-time implementations, this improvement is welcome and might free resources for other tasks.

MLPs also show that more parameters do not always translate into better recognition performance. Fig. 6 shows number of parameters versus computation times, and it clearly shows that CNNs are the fastest method. MLPs have similar computational performance but too many parameters to get good recognition accuracy, as shown in Fig. 5.

We also evaluated the effect of network depth in Table II. Deeper convolutional networks have better accuracy, but the same is not true for MLPs, as accuracy increases or decreases as the network gets deeper.

Our results shows that using CNNs for object recognition in FLS images improves performance over the current state of the art and does not need to make assumptions or preprocess the image, such as segmentations done by other works. Feature engineering is replaced by feature learning and it considerably outperforms previous methods. A CNN can learn important features directly from the raw sonar data and generalize very well on unseen data.

V. Conclusions and Future Work

In this paper we have demonstrated the use of Convolutional Neural Networks on Forward-Looking Sonar images. We compared CNNs with the commonly used template matching methods for object recognition, and shown that CNNs can provide better performance, while keeping a low parameter count. The CNNs that we have designed are faster than most template matching methods. Due to the small size of our datasets (only 2000 images) and the high amounts of noise that are characteristic to sonar, we did not expect to obtain big improvements.

We have also shown that classical neural networks (Multilayer Perceptrons) cannot provide adequate performance on FLS data, and are easily surpassed by both template matching and CNNs. While this has been previously explored in the literature [15] for color images, it has never been evaluated for FLS images. This key finding signals that deep and convolutional neural networks are a clear direction for future research in sonar image processing.

CNN performance is not very sensitive to network parameters, and we confirm that the best performing networks should use Batch Normalization [13] and ReLU non-linearities [16]. Deeper CNNs provide better performance, an the choice of activation function and regularization method is key to obtain excellent performance. Due to diminishing returns, it is known that bridging the gap between 95% to 100% accuracy is very difficult. CNNs have left a very small room for improvement (only 0.8%) on our dataset.

[1]On a Core i7 6700HQ CPU with 32GB of RAM.

Method			Accuracy	# of Parameters	Computation Time
Network type	Regularizer	Activation			
CNN	BatchNorm	Sigmoid	95.0%	930K	8.0 ± 1.9 ms
CNN	BatchNorm	Tanh	89.9%	930K	8.0 ± 1.9 ms
CNN	BatchNorm	ReLU	**99.2 %**	930K	8.0 ± 1.9 ms
CNN	Dropout	Sigmoid	16.4%	930K	8.0 ± 1.9 ms
CNN	Dropout	Tanh	22.3%	930K	8.0 ± 1.9 ms
CNN	Dropout	ReLU	96.9%	930K	8.0 ± 1.9 ms
MLP	BatchNorm	Sigmoid	65.5%	10.1M	7.0 ± 1.5 ms
MLP	BatchNorm	Tanh	29.5%	10.1M	7.0 ± 1.5 ms
MLP	BatchNorm	ReLU	92.3%	10.1M	7.0 ± 1.5 ms
MLP	Dropout	Sigmoid	62.1%	10.1M	7.0 ± 1.5 ms
MLP	Dropout	Tanh	16.5%	10.1M	7.0 ± 1.5 ms
MLP	Dropout	ReLU	51.7%	10.1M	7.0 ± 1.5 ms
Template matching with CC			92.4%	8.5M	53.4 ± 8.8 ms
Template matching with SQD			97.6%	8.5M	13.4 ± 2.7 ms

TABLE I. Comparison of the accuracy, number of parameters and computation times of different configurations of CNNs and MLPs versus Template Matching. For Neural Networks we vary the regularization method and the activation function. CNN architecture corresponds to Conv32-5, MP2, Conv32-5, MP2, FC64, FC10, while the MLP architecture is FC1024, FC512, FC256, FC10.

Type	Configuration	Accuracy	# of Parameters	Computation Time
CNN	Conv32-5, FC64, FC10	96.7%	4.3M	6.4 ± 1.4 ms
CNN	Conv32-5, MP2, Conv32-5, MP2, FC64, FC10	**99.2 %**	930K	8.0 ± 1.9 ms
CNN	Conv32-5, MP2, Conv32-5, MP2, Conv32-5 FC64, FC10	99.1%	184K	8.1 ± 2.3 ms
MLP	FC10	91.8%	101K	0.1 ± 0.0 ms
MLP	FC256, FC10	80.6%	2.3M	1.8 ± 0.5 ms
MLP	FC512, FC256, FC10	92.8%	4.8M	3.4 ± 0.9 ms
MLP	FC1024, FC512, FC256, FC10	92.3%	10.1M	7.0 ± 1.5 ms

TABLE II. Comparison of neural networks with varying depth after training for 10 epochs.

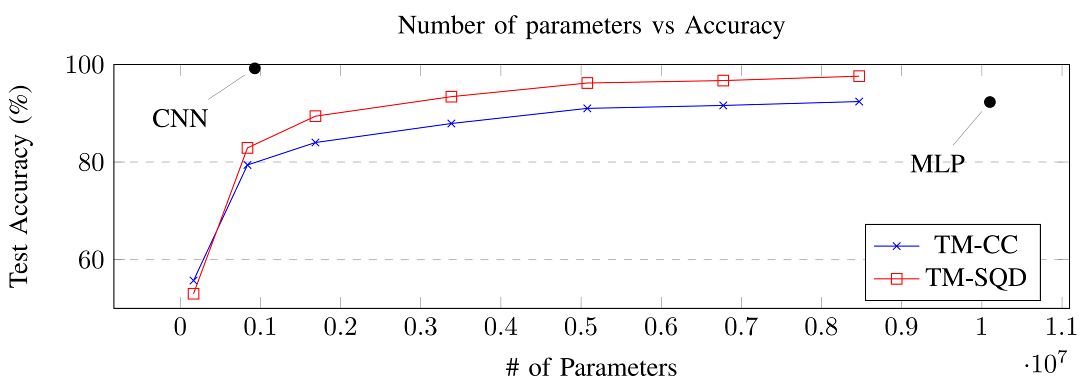


Fig. 5. Comparison of number of parameters versus accuracy on the test set. CNN refers to Conv32-5, MP2, Conv32-5, MP2, FC64, FC10, while MLP refers to FC1024, FC512, FC256, FC10 configuration.

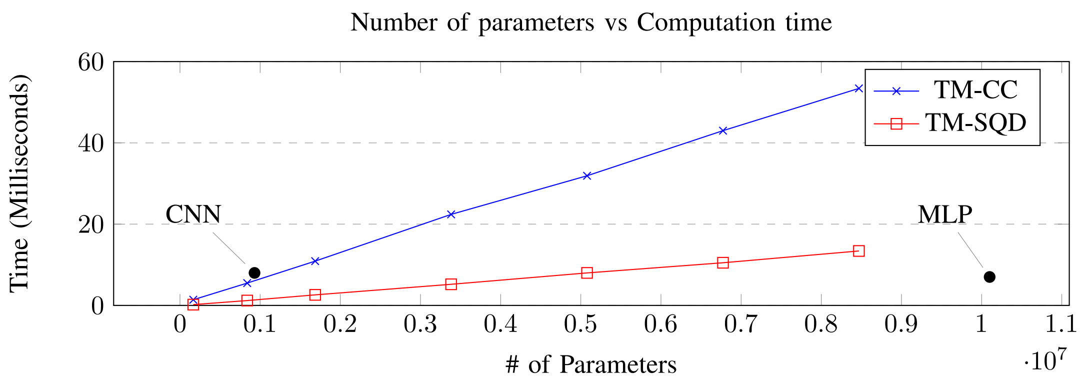
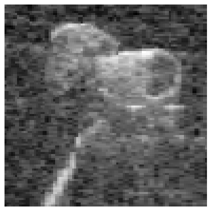
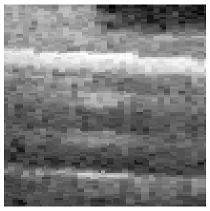
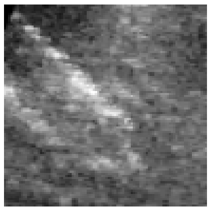


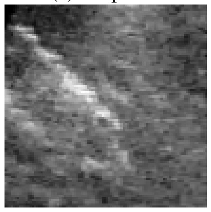
Fig. 6. Comparison of number of parameters versus computation times. CNN and MLP network configurations are the same as the ones shown in Fig. 5

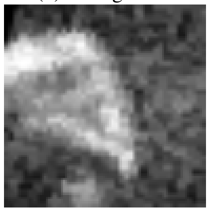
In terms of future work, we believe that CNNs can also be used for object detection in sonar images. A bigger challenge is to use CNNs for object detection and recognition in optical images. CNNs work very well in optical images, but it is

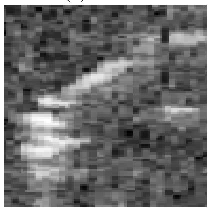

(a) Propeller


(b) Background


(c) Bottle


(d) Bottle


(e) Drink Carton


(f) Bottle

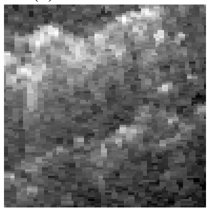

(g) Background

Fig. 7. Misclassified samples by our CNN, with their ground truth labels. Only these 7 samples from the test set are misclassified by our CNN.

unknown how they would perform with degraded underwater images.

Since CNNs do not require manual feature engineering and are completely data-driven, the current challenge is to collect big labeled datasets containing objects of interest. It is well known that machine learning methods improve with increasing amounts of labeled dataa. The computer vision community has already done it [17] and greatly profited in terms of results [4], but the Underwater community in general has not collected large datasets, mostly due to military secrecy and the small size of the field.

We believe that the use of Deep Learning techniques in Autonomous Underwater Vehicles will lead to significant improvements in the perception capabilities for this type of robots.

Acknowledgments

The authors would like to thank Leonard McLean for 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] 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.

[5] A. 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.

[6] H. Midelfart, J. Groen, and O. Midtgaard, "Template matching methods for object classification in synthetic aperture sonar images," in *Proceedings of the Underwater Acoustic Measurements Conference*, no. S S, 2009.

[7] 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.

[8] Y. Petillot, Y. Pailhas, J. Sawas, N. Valeyrie, and J. Bell, "Target recognition in synthetic aperture and high resolution side-scan sonar," in *Proceedings of the European Conference on Underwater Acoustics, ECUA*, 2010.

[9] D. P. Williams, "Fast target detection in synthetic aperture sonar imagery: A new algorithm and large-scale performance analysis," *IEEE Journal of Oceanic Engineering*, vol. 40, no. 1, pp. 71–92, 2015.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[11] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8609–8613.

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

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

[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[15] G. Urban, K. J. Geras, S. E. Kahou, O. Aslan, S. Wang, R. Caruana, A. Mohamed, M. Philipose, and M. Richardson, "Do deep convolutional nets really need to be deep (or even convolutional)?" *arXiv preprint arXiv:1603.05691*, 2016.

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

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.