Deep convolutional neural network target classification for underwater synthetic aperture sonar imagery

A. Galusha, J. Dale, J. M. Keller, A. Zare

A. Galusha, J. Dale, J. M. Keller, A. Zare, "Deep convolutional neural network target classification for underwater synthetic aperture sonar imagery," Proc. SPIE 11012, Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXIV, 1101205 (10 May 2019); doi: 10.1117/12.2519521

SPIE.

Event: SPIE Defense + Commercial Sensing, 2019, Baltimore, Maryland, United States

Deep Convolutional Neural Network Target Classification for Underwater Synthetic Aperture Sonar Imagery

A. Galusha*, J. Dale*, J. M. Keller*, A. Zare†

*University of Missouri, Electrical Engineering & Computer Science Department, Columbia, MO
†University of Florida, Electrical & Computer Engineering Department, Gainesville, FL

ABSTRACT

In underwater synthetic aperture sonar (SAS) imagery, there is a need for accurate target recognition algorithms. Automated detection of underwater objects has many applications, not the least of which being the safe extraction of dangerous explosives. In this paper, we discuss experiments on a deep learning approach to binary classification of target and non-target SAS image tiles. Using a fused anomaly detector, the pixels in each SAS image have been narrowed down into regions of interest (ROIs), from which small target-sized tiles are extracted. This tile data set is created prior to the work done in this paper. Our objective is to carry out extensive tests on the classification accuracy of deep convolutional neural networks (CNNs) using location-based cross validation. Here we discuss the results of varying network architectures, hyperparameters, loss, and activation functions; in conjunction with an analysis of training and testing set configuration. It is also in our interest to analyze these unique network setups extensively, rather than comparing merely classification accuracy. The approach is tested on a collection of SAS imagery.

Keywords: Synthetic Aperture Sonar, Deep Convolutional Neural Network, Deep Learning, Choquet Fuzzy Integral, Sugeno Fuzzy measure

1. INTRODUCTION

Synthetic aperture sonar imagery has continued to be an effective tool for observing seafloor sediment and sunken targets. With multi-frequency capabilities it is able to inspect both topical and buried objects with great clarity. The potential of this sensor has been employed extensively for use in underwater explosive hazard extraction.

Over the last several years, the field of deep learning has expanded rapidly into many areas relating to computer vision. Deep convolutional neural networks (CNN) have demonstrated efficiency at picking out complex features from imagery through the use of kernel-based filters. These large networks are most effective when provided with vast quantities of training data. Because images are so complex, a sparse data scope facilitates better global convergence. While deep learning approaches are relatively new to the area of underwater explosive hazard detection, they have proven to be a viable target classification method for differentiating explosive hazards from the seafloor, as seen in previous publications[6].

The goal of this study is to show the accuracy of deep CNNs in locating potentially hazardous targets in SAS imagery, and to show how specific workflows improve effectiveness over others.

Author contact links:
A.Galusha - apgqpc@mail.missouri.edu
J.Dale - jjdale@mail.missouri.edu
J.M.Keller - kellerj@missouri.edu
A.Zare - azare@ece.ufl.edu

Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXIV, edited by Steven S. Bishop, Jason C. Isaacs, Proc. of SPIE Vol. 11012, 1101205 · © 2019 SPIE
CCC code: 0277-786X/19/$18 · doi: 10.1117/12.2519521

2. METHODS

2.1 Data Preprocessing

The beamformed SAS data produces a large fine grain pixel matrix with 2 channels: high frequency and low frequency. To start, we want a target confidence measure at every pixel. Instead of applying an accurate but computationally expensive classification procedure to all possible coordinates, we use the combined RX algorithm[1], a quick anomaly detector, to narrow the imagery down to the most likely target locations. The detector computes a target likelihood value for every pixel based on a neighborhood around it, producing a confidence map of the same size as the original image. This confidence map allows only the ROIs surrounding the most likely pixels to be considered by the classifier. A summary is provided below of the combined RX algorithm, though the reader is referred to the study by Galusha et al. for more detail[1].

The Combined RX prescreener is a fusion of four Reed-Xiaoli anomaly detectors, each of which was found to show strength in differing areas of target detection. The standard RX detector consists of a sliding window and a surrounding square annulus, separated by a gap. This gap helps confidence values remain robust to odd target shapes by providing separation between the intended target and background regions. The RX left-right (RX-LR) detector operates in the same way, except the inner and outer annulus windows in standard RX are now adjacent to each other. RX-LR is tailored to capture the characteristic shadow that appears to the right of targets in SAS imagery. Pixel values within respective windows are plugged into the RX equation (1) to obtain a detector confidence.

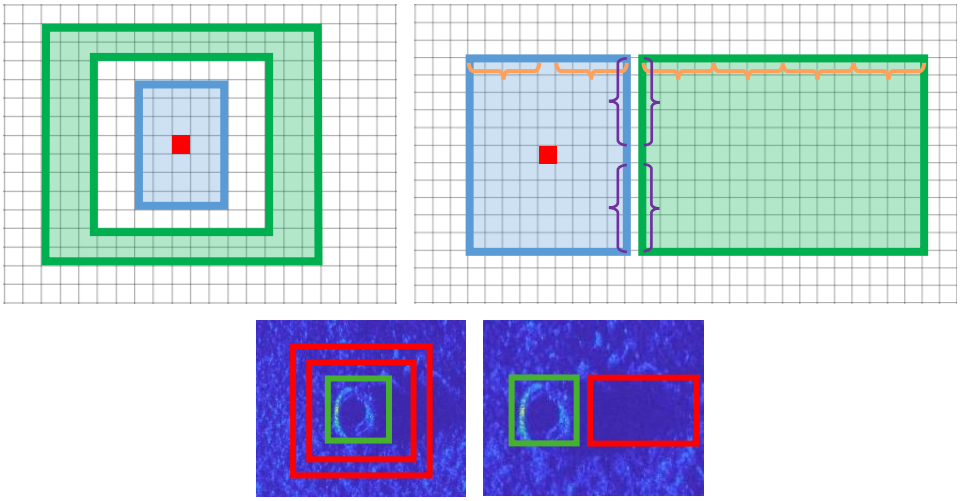

Figure 1: Left images show regular RX layout, right images show RX Left-Right

$$RX = \frac{(\mu_{in} - \mu_{out})^2}{\sigma_{out}^2} \tag{1}$$

Here the pixel mean μ_{in} and μ_{out} of respective inner and outer regions, and variance σ_{out}^2 of the outer region are used.

By considering the RX and RX-LR responses for both high and low frequency SAS channels, we have four confidence values for each pixel. The final combined RX detector confidence is the product of these four confidences. Other fuzzy methods of fusion, such as the Choquet Integral, have been shown to be less effective in relative target separation[1].

From the combined RX confidence map, non-maximal suppression segments the map into a subset of important regions of interest corresponding to high detector certainty. There are many renditions of suppression algorithms, and we tested a few fragile approaches before landing on a simple, robust approach. First, we find all peaks on the confidence map. Peaks are ideal because they are natural points of distinction from the background and a singular greater target potential than its neighbors. Then, discard peaks that are less extreme than neighboring peaks. By setting a radius defined by expected target size, smaller peaks within radius of bigger peaks are discarded. The result of this suppression method is a list of important coordinates alongside detector confidence values for each SAS image. There are regularly hundreds of these points per image.

While hundreds of ROI's are more manageable than millions of pixels, almost all of these suppressed points are false alarms. Fortunately, a look at the distribution of target and false alarm confidences reveals clear structure, which can be exploited to produce an adaptive confidence threshold. Preliminary experiments have shown promise, reducing the number of false positives per image by an order of magnitude without discarding any targets.

With a more feasible problem size, we extract two-channel SAS image tiles from each point. To account for expected shadow regions on the right, we use a roughly 2×3 tile aspect ratio, padded by one pixel so tiles have a unique center point, with potential targets centered slightly left within the tile. ROIs that extend past image boundaries are padded with zero values, as CNNs are largely robust to this alteration at the input layer. The set of false alarms and true target tiles are used to train the CNN classifiers.

2.2 Network Structure

Many variations to network architecture were evaluated, but all successful experiments had similarities to the structure described here. This network has two consecutive convolution layers, each followed immediately by a max pooling layer. After convolution and pooling, a fully connected layer with dropout feeds into a final two node output layer that defines target and false alarm prediction values. Loss on this network is computed as cross entropy with input batches of size 20 for 2000 steps. All networks were created with the TensorFlow deep learning framework. Figure 2 shows a visual depiction of the chosen network architecture.

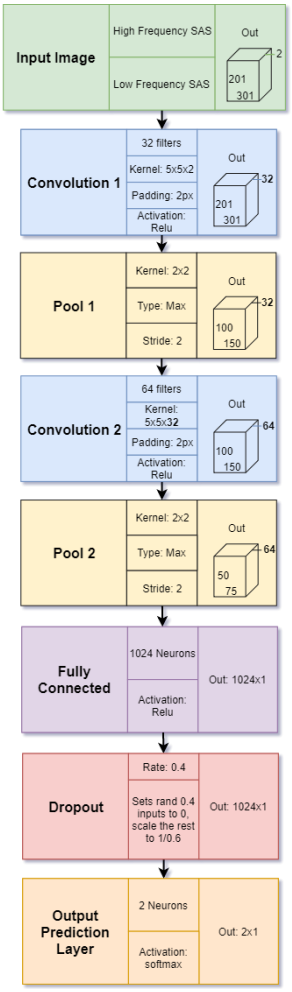

Figure 2: Diagram of the network architecture used throughout experiments showing blocks colored by operation.

Parameter tuning included adding/removing layers, changing activation functions, convolution kernel size adjustment, number of steps per training (similar to epochs), and altering learning/dropout rates. All subsequent results stem from this architecture which yielded the best results.

2.3 Data Set Manipulation

Most improvements in classification accuracy came from proper changes to the input data. To validate network performance, global location-based cross-validation is implemented. One factor that directly influenced network accuracy was how false tiles were chosen for training. While the training data is still widely unbalanced after the adaptive confidence threshold, neural networks are able to handle some degree of asymmetry, as opposed to other learning models. Basic data augmentation on the true tiles was effective in better balancing and expanding the dataset. The following subsections describe these aspects in detail.

2.4 Fold Cross Validation

Before a model can be trusted, it must be corroborated. One of the most widely used methods of doing so is to split our labeled data into smaller training and testing partitions. To extensively validate our results, we partition the labeled data by real world location. Ideally, partitioning the labeled imagery by geographic location would roughly correspond to partitioning by seafloor topography. The number of labeled images from each location vary drastically in size, facilitating further analysis of network performance in discrete training scenarios. This system lets us clearly see which locations house valuable data and how much training data is required to train effectively.

The training/testing configuration was to train on four of the five locations and test on the excluded. When we refer to "network 1", we are indicating the network that was trained on locations two through five and tested on location one. For fusion methods, it is important for proper analysis that no fusion combines training and testing data. This means that any two networks trained using the described five-fold cross-validation scheme cannot be fused with testing labels bleeding into training data. Networks used for fusion are trained only on their respective location. Fusion methods are described in detail in Section 2.4.

Figure 3: Cross-validation fold structure used for classification validation

2.5 Training Sample Sources

In preliminary training, there were discussions about what kind of false alarms would best help the network to differentiate between targets and non-targets. The form of a target is obvious to human operators, while most false alarms are flat and less distinct. Because low confidence false alarms are generally flat seafloor with little similarity to true targets, it would seem trivial for a network to differentiate these two. This led us to choose high confidence false alarms for training but resulted in decidedly worse accuracy than training with expected low confidence alarms. The results section discusses this phenomenon in further detail. The types of false alarms we considered were high confidence, low confidence, a combination of high/median/low confidence, and random choice.

All false alarm data was extracted from the same image as each ground truth target. By using nearby false tiles, the intent is that the seafloor texture with be consistent with our targets. Given vastly differing examples of targets and false alarms, a network may decide that the distinction is really the surface pattern rather than the existence of a target. This would undoubtedly lead to poor performance in the field.

2.6 Training Sample Sizes

In most machine learning scenarios, it is ideal for class partitions to be balanced. This is because imbalance persuades many learning models to accidentally favor the class of bigger size. If 99% of the data is false alarms, which is true of our SAS dataset, a classifier can call everything a false alarm and get 99% accuracy. Neural networks have been found to be more robust to this asymmetry[12], which works in our favor. The number of examples in each class in this context is highly unbalanced, even after further suppression via a dynamic confidence threshold Even CNNs suffer in performance with class imbalance of this severity. We decided on three false alarms per true target after experimentation.

2.7 Augmenting the Data

A popular way to compensate for an unbalanced dataset is to augment the smaller class with slightly adjusted copies of existing examples. Neural networks can sometimes result to correlating class labels to fine details of the inputs rather than the big picture. Augmenting the data serves to both grow the training set and force networks to examine the same inputs from a different perspective.

There are many methods of image augmentation[13], but due to the underlying physics behind SAS image formation, most of these augmentation techniques do not produce realistic images. The most prominent characteristic of SAS imagery is that the shadow must always face away from the sonar source, on the right in our imagery. Flipping the target along the horizontal axis, for example, does not violate this constraint. Another alteration we used was to rotate the target +/- 10 degrees, about the expected target location within the tile. While rotation is not generally viable for augmenting SAS data (because the shadow is no longer horizontal), this rotation is so slight that the shadow is still mostly horizontal. After these augmentation techniques, the number of examples in the target class grows six-fold. Other augmentations techniques, such as the introduction of noise, will be tested in the future. An example of these manipulations is shown in Figure 4.

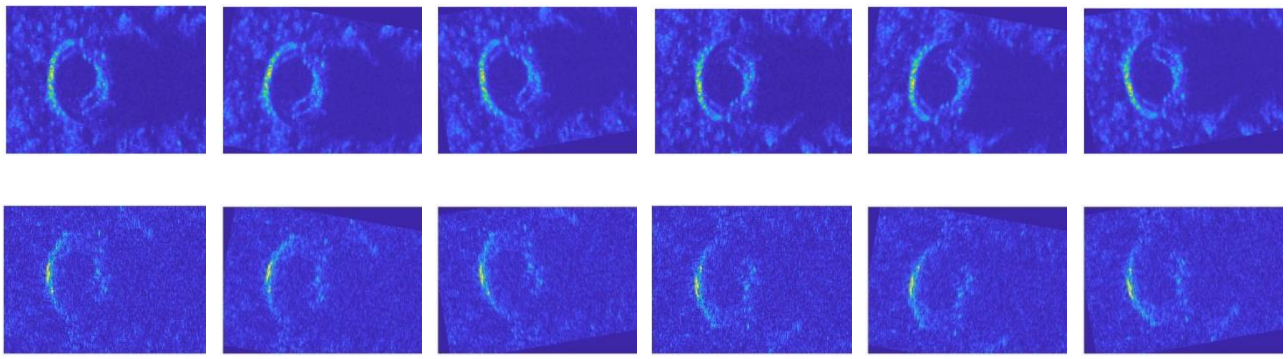

Figure 4: Target augmentations where top and bottom rows are high and low frequency respectively, augmentations here are none, rotate 10°, rotate -10°, flip horizontally, flip & rotate 10°, flip & rotate -10°.

2.8 Prediction Fusion

To further improve the accuracy of classification, we use classifier fusion techniques to generalize a tile's true-confidence rank beyond that of a single model. In theory, it is possible for this network to achieve optimal performance, but it is highly unlikely in non-trivial scenarios. It is also unlikely that the confidence assigned by each classifier will be directly comparable. The goal of fusion is that, by merging the grades of a tile in multiple models, the final fused value/rank will closer match our expectation of that tile's true class. Rank in this context refers to the position within the sorted list of all confidences in which a given tile is placed. True rank will have all target tiles with higher confidences than all false alarms. Fusion can be thought of as averaging the position of ten dart throws at a dart board. Ideally, the average of all throws will be closer to the center than any one throw.

As mentioned in Section 2.3.1, it is important in fusion that we do not indirectly expose testing labels to the training set. To ensure the integrity of the test set, all fused model outputs for a given test point must never have been exposed to the label of that test point. This means that fusion algorithms can only occur between models that test on same data. The two fusion methods shown in this paper are the combined classifier and the Choquet classifier.

2.9 Combined Classifier

While this fusion method is extremely simple, it has proven its effectiveness in many applications. The combined classifier is a simple multiplication of all confidence values computed in relation to a single tile. For the five networks trained on each individual location set, the four networks not trained on the current test location produce confidences, and a final certainty value results from the product of these confidences. In our detector fusion experiments, this method outperformed a static and dynamic Choquet integral fuzzy fusion algorithms[1]. This may potentially be attributed to detector confidence residing strictly in $\mathbb{R}^+$. For classifiers, there is no assistance from extremely confident model sources because all predictions reside in the [0,1] space.

Let $f_i(x)$ be the confidence of model i on point x.

$$C(x) = \prod_i f_i(x) \tag{2}$$

Where C is the combined classifier confidence.

2.10 Choquet Classifier

When considering multiple data sources, the optimal fusion algorithm should prioritize the most accurate evidence while never completely disregarding any contribution. Choquet fuzzy integrals attempt to achieve this through a non-linear weighted average of all data sources. From a defined fuzzy measure, incoming evidence is weighted by the pertinent fuzzy measure value and summed together to produce a single confidence. To define the fuzzy measure, we use the Sugeno λ-measure.

The Sugeno λ-measure needs an initial set of measures for the singleton data source subsets, which we can think of as the importance of each data source. These are called densities and are often defined either by experts or some metric. Here, a grading metric is implemented just like in our detector fusion[1]. The area under the curve (AUC) of ROC curves work well because the values are in [0,1] and the higher an AUC value is, the more accurate that classifier is toward the ground truth. It is also reasonable to assume that the more accurate the classifier, the more we want to prioritize its evidence. With these four densities (in our scenario) we can now solve for the nonzero λ value that will derive fuzzy measure components:

$$1 + \lambda = \prod_{i=1}^{n}(1 + \lambda g^i) \quad : \quad \lambda > -1 \tag{3}$$

Where g^i is the AUC density for data source i.

This unique λ value lets us solve for the measure attributed to all possible subsets of data sources:

Subset	**Measure**
$A \cup B$	$g(A \cup B) = g(A) + g(B) + \lambda g(A)g(B)$

(3)

Where $g(X)$ is the measure of source X, $g(A \cup B)$ is the measure of the joined subset $A \cup B$, and $A \cap B = \emptyset$.

From Equation 3, we can compute the whole fuzzy measure given only the densities. With the fuzzy measure, we can compute final Choquet fuzzy integral values for every tile. Every test tile is given four network confidence values from the opposing four location-trained networks. The confidence values are sorted in descending order and represented as $h(x_i)$ where x_i is the source and $i \in [1, n]$ as i pertains to the source index after a descending sort of the n sources (in our case, $n = 4$). This is done individually for sets of confidences attributed to each data tile. The final Choquet integral is computed with Equation 4.

$$\text{Choq}(h) = \sum_{i=1}^{n}\big(h(x_i) - h(x_{i+1})\big)g(A_i) \tag{4}$$

Where g is a measure, A_i is the current cumulative set of sources $\{x_1, \dots, x_i\}$, and i corresponds to sorted sources.

For each test set, a new fuzzy measure is determined from the four test classifiers not trained on that test set, and their ROC AUC values. There is a unique set of sources for each test, resulting in five unique fuzzy measures. The Choquet integral is then distinctly computed on each test set with Equation 4. For more information on how this algorithm is carried out, the reader is referred to the work of Keller[5].

3. RESULTS

3.1 Training Data Source Comparison

In all five folds, training with low confidence false alarms results in much higher accuracy. The intermediate curves trained on high/mid/low data also performs better than classifiers trained on high confidence data, which affirms that trending toward low confidence training data improves network understanding.

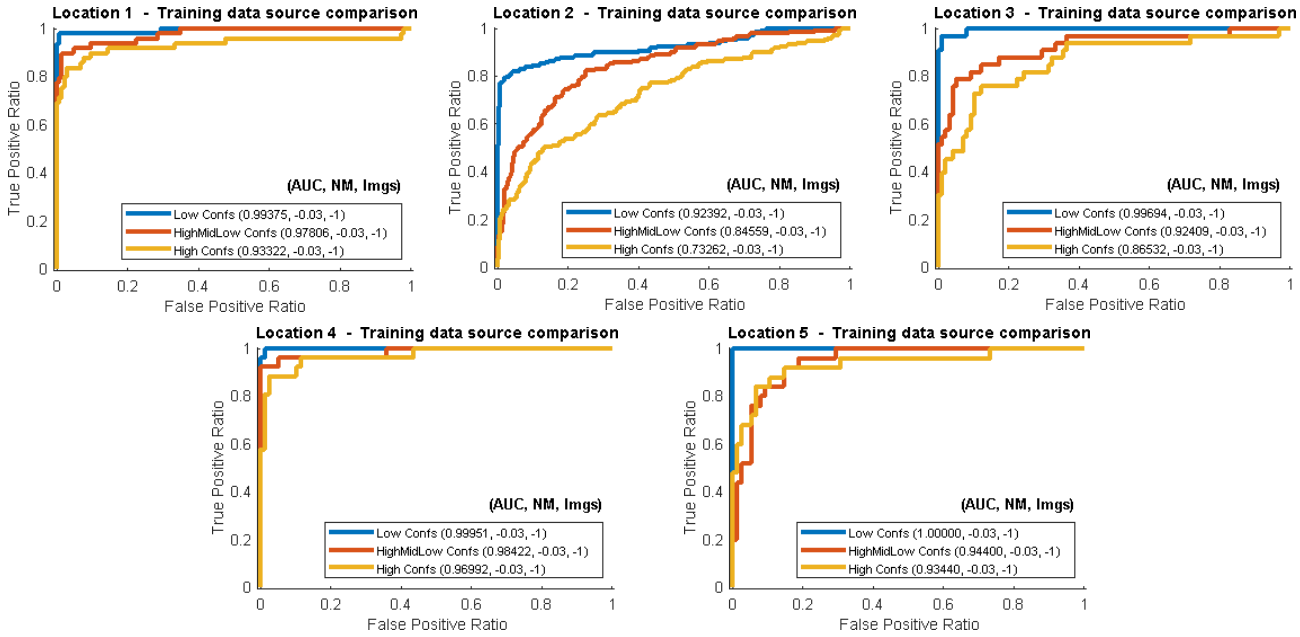

Figure 5: ROC curves for the 15 networks trained & tested on unique data sources and locations.

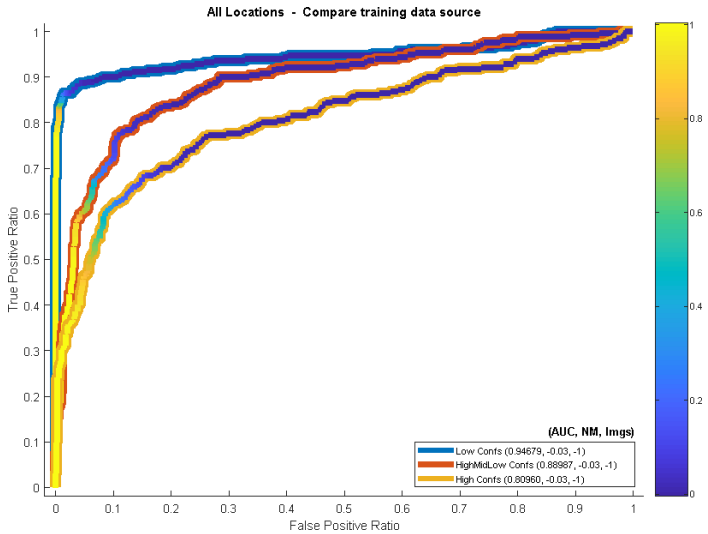

Figure 6: All locations test data concatenated by their training type, yielding test ROC curves of the whole dataset. Color within the curve designates prediction values between 0 and 1.

3.2 Augmented Data

The difference in accuracy between augmented and unchanged data is not striking, but in all location scenarios, except for location five, augmented data performs better than raw data. In location five, the difference in performance is mostly negligible. After concatenating the predictions into full dataset ROCs, we see a consistent improvement over the raw test results.

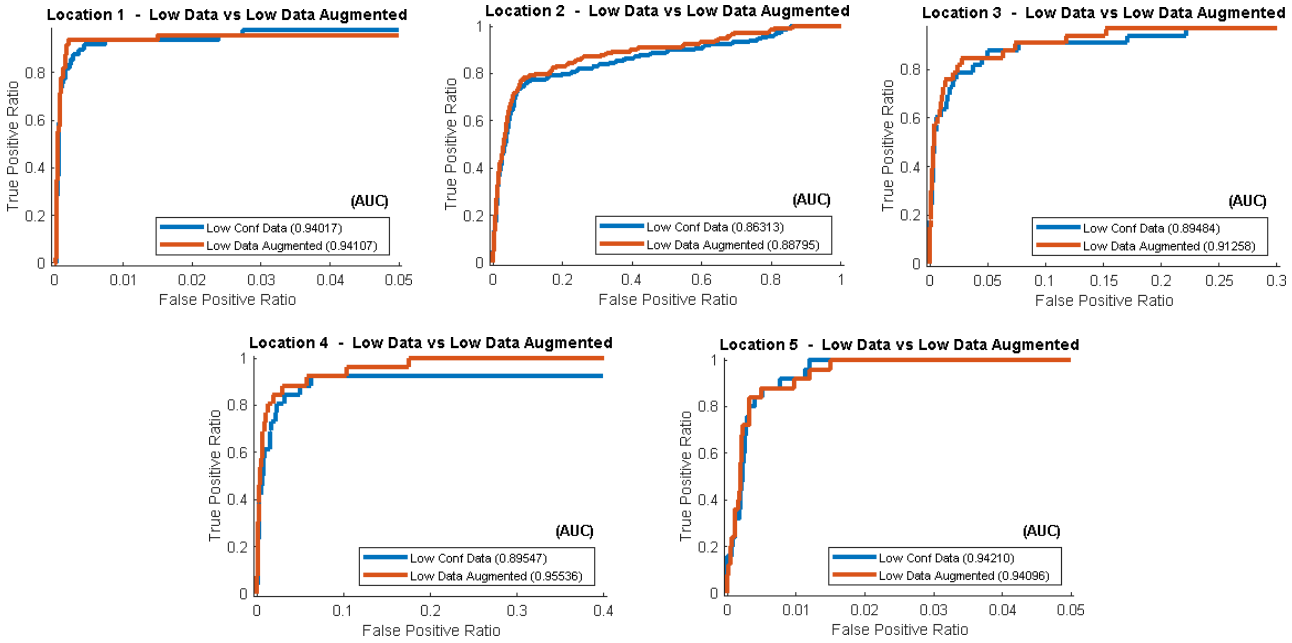

Figure 7: ROC curves plotted for the 5 augmented networks compared to their raw counterparts by location.

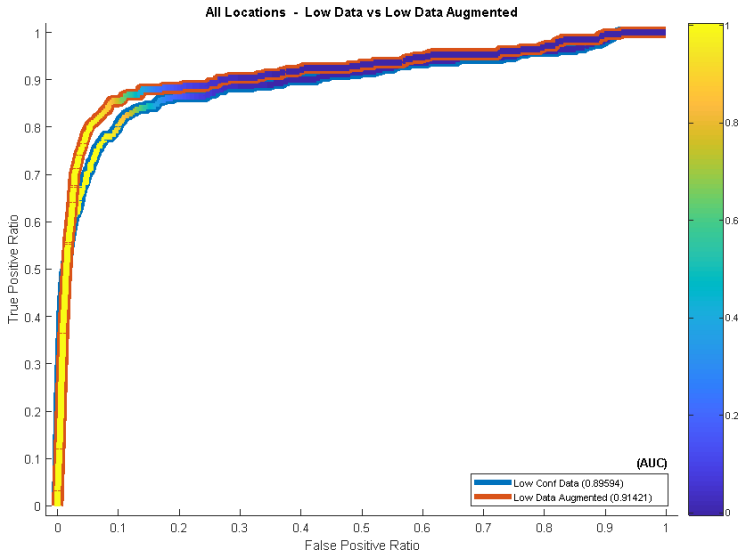

Figure 8: Low confidence original and augmented test data from all locations concatenated by their training type, yielding test ROC curves of the whole dataset. Color within the curve designates prediction values between 0 and 1.

3.3 Small Training Set Fusion

In each of the five location ROC plots, we see that the Choquet classifier consistently improved upon the results of the individual four networks. The combined classifier, while working well with detector fusion, did not perform very well here. In locations two and five, the combined classifier performed similarly to Choquet, but because the data in location two is much larger than the other four folds, a simple fusion was unable to account for this disparity. This size difference is also why network two performs well on all experiments except for location five.

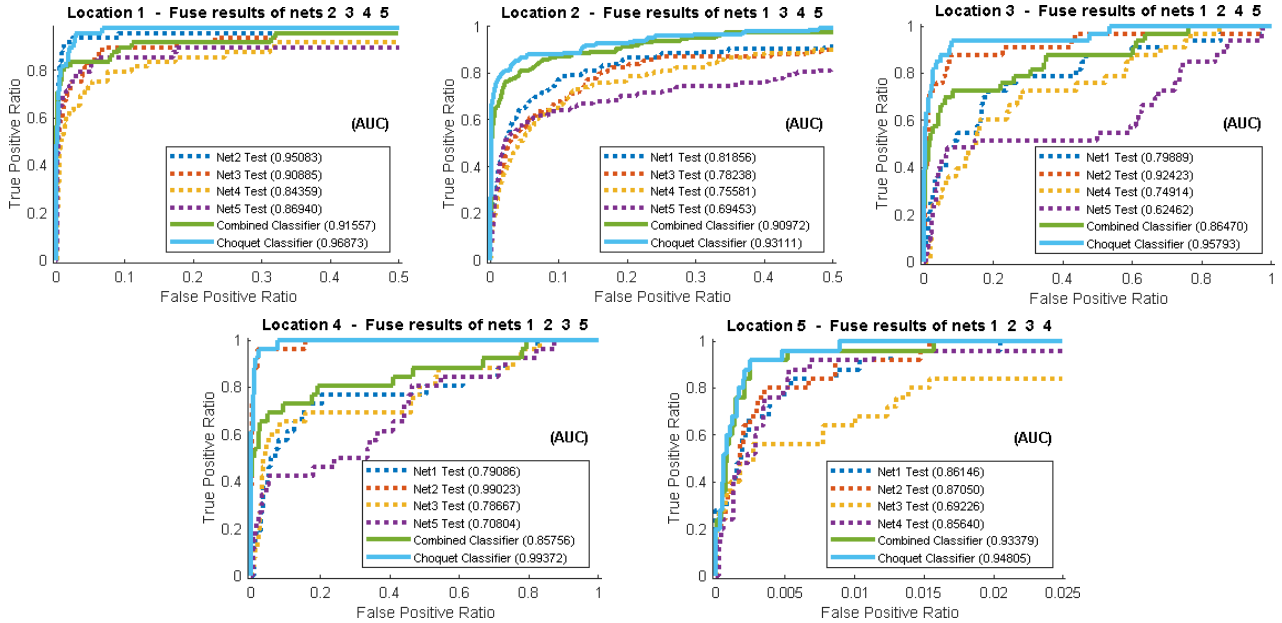

Figure 9: Augmented models trained on single location and tested on other four, combined and Choquet fuse opposing four location model predictions, the six ROC curves are plotted in correlation to each location's test data.

Merging all test predictions together produces the curves below where similar results occur. The Choquet continues to produce the highest accuracy with network two close behind, followed by the Combined classifier. It is important to note that the curves for networks one through five are made from smaller data sets than the fused curves here, since each only contains predictions from their respective four test sets. See how the augmented data networks over the whole data set perform compared to the Choquet in Section 3.4.

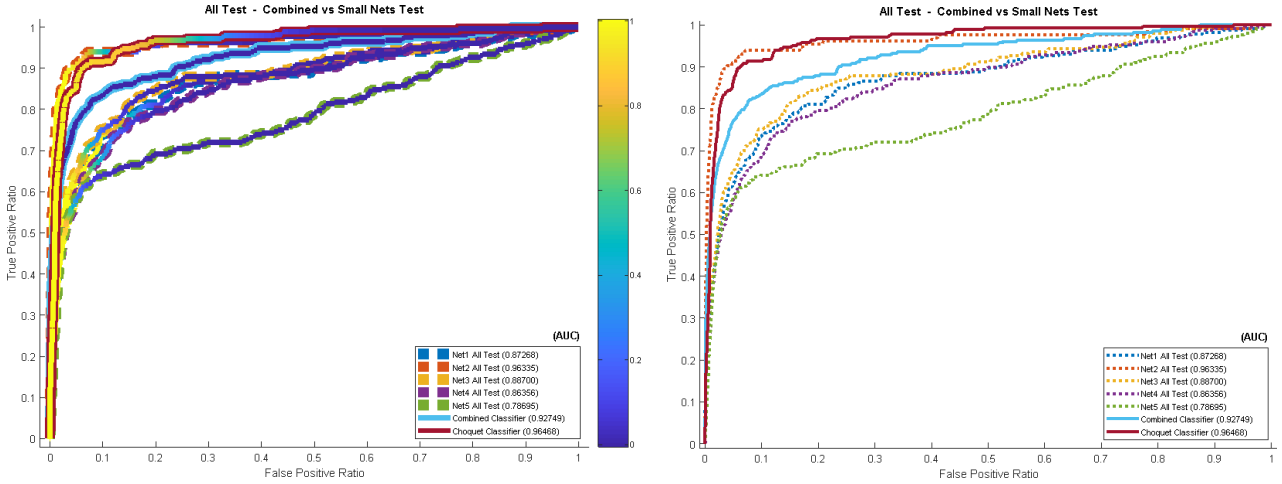

Figure 10: Curves are the aggregate of test prediction data over the 5 locations, yields test ROC curves over the whole data set, color within the curve designates prediction values between zero and one.

3.4 Classifiers and Detector Comparison

Plotting the best classifiers with the detector lets us observe clear improvement across all curves prior to a two false alarms per image false alarm rate. After this point, the detector stays consistent while combined and regular augmented networks drop off in accuracy. Conversely, we see the Choquet classifier continue outperforming the detector throughout all false alarm rates.

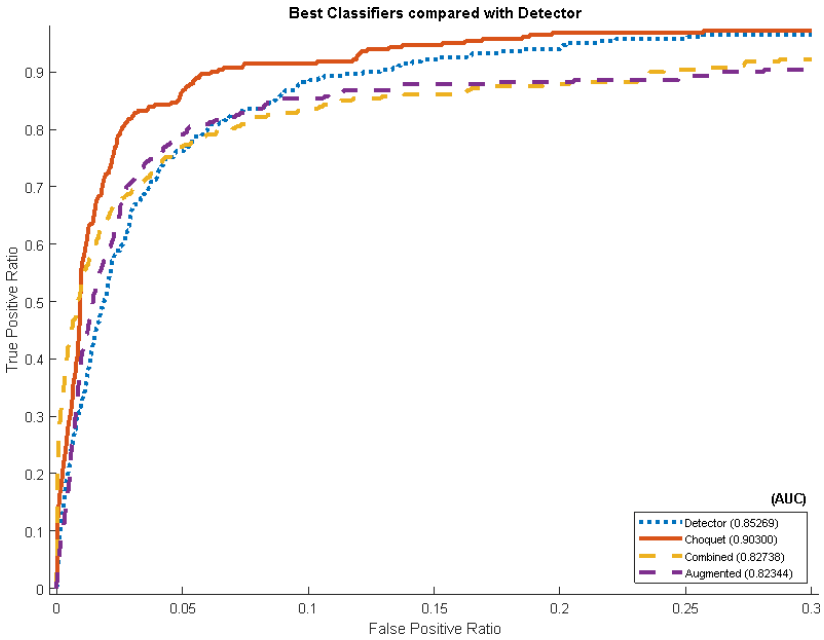

Figure 11: Aggregate test predictions from the three best classifiers over the five locations yield ROC curves over the whole data set.

4. CONCLUSION

In this paper, we have shown that deep CNNs are a viable binary target classifier for SAS image tiles. After extensive testing using location-based cross-validation, it has been shown that both data source and size have large hold on a classifier's prediction accuracy. Here there is also clear evidence that augmenting the target set made significant improvement over its raw data counterpart. From the augmented dataset, fusion between network predictions that were trained on small folds gave a compelling increase in classification performance over detector accuracy.

While advancements here are clear, there is room for improvement into lower false alarm rates. Further research will involve transfer learning of larger networks and fusing detector confidence values with classifier predictions through sigmoid and hyperbolic tangent translation.

ACKNOWLEDGEMENTS

This work was funded by the Office of Naval Research grant number N00014-161-2323 to support the U.S. Naval Surface Warfare Center.

REFERENCES

[1] P. Galusha, Aquila & M. Keller, James & Zare, Alina & Galusha, Gable. (2018). A fast target detection algorithm for underwater synthetic aperture sonar imagery. 19. 10.1117/12.2304976.

[2] Heesung Kwon, Nasser M. Nasrabadi, "Kernel RX-Algorithm: A Nonlinear Anomaly Detector for Hyperspectral Imager," IEEE transactions on Geoscience and Remote Sensing, (2005).

[3] Viola, Paul & Jones, Michael. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE Conf Comput Vis Pattern Recognit. 1. I-511. 10.1109/CVPR.2001.990517.

[4] Philippe Courmontagne, "A new approach for mine detection in SAS imagery," OCEANS 2008 – MTS/IEEE Kobe Techno-Ocean, (28 May 2008).

[5] James M. Keller, Derong Liu, David B. Fogel, [Fundamentals of Computational Intelligence], John Wiley & Sons, Inc., Hoboken, New Jersey (2016).

[6] Williams, David P.. "Underwater target classification in synthetic aperture sonar imagery using deep convolutional neural networks." *2016 23rd International Conference on Pattern Recognition (ICPR)* (2016): 2497-2502.

[7] Dale, Jeffrey, Galusha, Aquila, Keller, James, and Zare, Aina, "Evaluation of image features for discriminating targets from false positives in synthetic aperture sonar imagery," in [*Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXIV*], (2019).

[8] L. Chen, Johnny & Summers, Jason. (2017). Deep convolutional neural networks for semi-supervised learning from synthetic aperture sonar (SAS) images. The Journal of the Acoustical Society of America. 141. 3963-3963. 10.1121/1.4989020.

[9] Auephanwiriyakul, Sansanee & M Keller, James & Gader, Paul. (2002). Generalized Choquet fuzzy integral fusion. Information Fusion. 3. 69-85. 10.1016/S1566-2535(01)00054-9.

[10] Chiang, Jung-Hsien. "Aggregating membership values by a Choquet-fuzzy-integral based operator." Fuzzy Sets and Systems 114 (2000): 367-375.

[11] Krizhevsky, Alex & Sutskever, Ilya & E. Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.

[12] Lu, Yi, Hong Guo and Lee A. Feldkamp. "Robust neural learning from unbalanced data samples." (1998).

[13] Perez, Luis & Wang, Jason. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning.