SUPERVISED DEEP LEARNING CLASSIFICATION FOR
MULTI-BAND SYNTHETIC APERTURE SONAR

MS Emigh Naval Surface Warfare Center, Panama City, USA
B Marchand Naval Surface Warfare Center, Panama City, USA
MG Cook Naval Surface Warfare Center, Panama City, USA
JL Prater Naval Surface Warfare Center, Panama City, USA

1. INTRODUCTION

In recent years, deep learning techniques have come to outperform traditional computer vision techniques in tasks such as classification, detection, and segmentation. Deep discriminative representations have been found to be not only suitable, but superior, for a growing number of visual tasks. Current conventional sonar image ATR algorithms work well when training and test sets are from similar environments. Unfortunately, changes in environmental conditions, high clutter density, or variance in target signature can degrade performance considerably. Deep learning promises to exploit data invariances yielding better generalization. Deep convolutional networks can be trained to be robust to variations in shape, translation, rotation, illumination, distortion, and noise. Although work has been previously published on supervised deep learning for high frequency synthetic aperture sonar (SAS) automatic target recognition (ATR)\(^1\)\(^2\), deep learning for multiband SAS has not yet been thoroughly examined. We define multiband SAS as SAS imagery containing multiple co-registered imaging bands.

As sonar imagery technology has progressed, there has been a movement away from lower resolution real aperture sonar toward higher resolution SAS. Ironically, the finer detail that SAS provides about targets can also serve to confuse ATR algorithms by producing finer detail about surrounding seafloor and clutter. Fortunately, lower frequency bands can be used to obtain additional information about targets. It has been shown empirically\(^3\) that two co-registered frequency bands separated by a few octaves, when coupled with a sophisticated ATR algorithm, can lower false alarm rates.

In this paper we share lessons learned in training deep networks using multiband SAS data. We discuss techniques for successfully training these networks, and the effects single and multiband data has on classification accuracy. We further investigate the effect of using natural image data, i.e., images generated from natural scenes, to pre-train these networks. As a comparison to other machine learning approaches, we evaluate the results on a shallow RVM classifier trained on expertly engineered features. Lastly, we investigate what information is added by the lower frequency band.

2. EXPERIMENTAL SETUP

Many sonar image data collection search trajectories, such as the “lawnmower” pattern, generate data such that each possible target may be seen multiple times at different ranges but at the same aspect angle. Thus, conventional train/test splits may not provide a validation or test set with adequately unique data. That is, training a network using this type of split will lead to overfitting due to the same targets being found in the training, validation, and test sets. To provide an appropriate split, it is necessary to create a validation set from data collection runs not included in the training set.

The amount of labeled multiband SAS data available to be used as training data is currently limited. To train a network, we desire to use as much of the data for training as possible. Thus, for the following experiments, we need be creative in generating validation and test sets. For the training set we used snippets generated by running an Reed-Xiaoli (RX)\(^4\) based anomaly detector over the entire SAS data set. Each snippet was resized to 224 \(\times\) 224 pixels. The detector produced hundreds of positive and tens of thousands negative examples. For the validation set, we used a small, very “difficult,” highly variable dataset consisting of tens of positive and thousands of negative examples. We created a test set by beam-forming, using a different image formation processor, and then running a different (KL-divergence based) detector on the validation data. In practice, different image formation processors often produce similar,
but rarely identical imagery. Using different image formation processors and detectors results in a test set that although is generated from the same raw data, is quite different from the training and validation sets. Both detector thresholds were set extremely low so as to maximize the number of positive examples found. The validation and test sets contain relatively similar target snippets, as the two beam-formers do not produce exceptionally disparate results. However, they do contain dramatically different non-target snippets, as the two detectors are built upon fundamentally different algorithms.

We use ResNet\textsuperscript{5,6} as the base architecture for its ease of training and for its property that increasing the number of layers does not decrease performance. Furthermore, it has somewhat become the de facto standard for supervised deep learning. For most of the experiments we use the 18 layer (Resnet-18), version 1\textsuperscript{5}, (Imagenet) variant. The Resnet-18 architecture consists of a 7 \times 7 input convolutional layer, followed by 16 3 \times 3 convolutional blocks. Each convolutional block consists of a convolutional layer, followed by a batch normalization layer\textsuperscript{7} and a rectified linear unit (ReLU) activation. We use the ADAM\textsuperscript{8} learning rule to train the network using an initial learning rate of .001. The learning rate is decreased when the learning curve plateaus. The inputs to all of the networks are 224 \times 224 images with one, two, or three channels, depending on the experiment. For experiments using SAS data, the first and second channels correspond to the SAS high frequency (HF) and low frequency (LF) channels respectively. For experiments using natural images, the three channels correspond to the red, green, and blue channels. Each network was trained by minimizing a cross-entropy cost function for 200 epochs. The learned network parameters were selected by minimizing cost on a validation set.

2.1 Class Imbalance

Classifier effectiveness is often reported as classification accuracy. However, overall accuracy is not very revealing due to the extreme class imbalance often found in the sonar ATR problem, i.e., the classification of non-targets overwhelms the accuracy calculation. A more useful alternative to classification accuracy is the confusion matrix, which gives the number of correctly and incorrectly classified targets for each class. For a binary classifier, a receiver operating characteristic (ROC) curve is even more descriptive.

Classification accuracy and confusion results can be obtained by selecting a threshold for the classifier using the ROC curve. For a deep learning classifier, which typically uses either a sigmoid or softmax output layer, classification rate is often taken to correspond to a threshold of 0.5 by default. However, there is no reason to avoid selecting another threshold.

For example, targets are typically sparse in a sonar dataset that was generated by following a given search pattern. That is, detectors commonly produce many snippets for each sonar image, while most images do not contain even a single target. It has long been known\textsuperscript{9,10,11} that class imbalance detrimentally affects the performance of not only neural networks but most machine learning algorithms. For deep networks, a simple fix is to weight the gradient update so that each sample gradient is weighted according to the class it belongs to. For example, if the number of negative examples outnumbers the number of positive examples by a factor of 10, each positive example would receive a weight of 10. This approach is a form of oversampling\textsuperscript{10} in which minority class samples are repeated so as to balance the dataset.

Figure 1a and Table 1 show (test set) ROC curves and confusion matrices (rows normalized by number of samples) for a four networks trained on varying class weights. The \times 0 ROC curve corresponds to not weighting any class, while \times 1 corresponds to a “balanced” weighting. The \times 2 and \times 4 correspond to weighting the positive examples at multiples even higher than what would be used to achieve balance. This may be desirable, since correctly classifying targets may be seen as more important than correctly rejecting false alarms. As can be seen by the confusion matrices in Table 1, the network trained using no class weighting achieved a sensitivity of only .461 while the \times 2 network achieved .824. However, for any given false positive rate (FPR) below 0.2 (Figure 1a), the sensitivity of the two networks remain close. Also note that the network trained using the very high weighting of \times 4 underperforms for FPRs below 0.2, but achieves a higher true positive rate (TPR) than the other networks at higher FPRs. This shows that appropriate class weight to train a network is highly dependent on the desired false positive rejection rate. Nonetheless, for the following experiments we use the \times 2 class weight for training, as it provides high sensitivity on all validation and test sets at the “default” threshold of 0.5.
Table 1: Confusion matrices for various class weighting approaches, normalized by row

<table>
<thead>
<tr>
<th></th>
<th>$\times 0$</th>
<th>$\times 1$</th>
<th>$\times 2$</th>
<th>$\times 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P'$</td>
<td>.461</td>
<td>.795</td>
<td>.824</td>
<td>.808</td>
</tr>
<tr>
<td>$n'$</td>
<td>.539</td>
<td>.204</td>
<td>.175</td>
<td>.191</td>
</tr>
<tr>
<td>$P$</td>
<td>.005</td>
<td>.034</td>
<td>.076</td>
<td>.102</td>
</tr>
<tr>
<td>$n$</td>
<td>.995</td>
<td>.966</td>
<td>.924</td>
<td>.897</td>
</tr>
</tbody>
</table>

(a) ROC curves using various class weights. A filled circle indicates the point on a curve corresponding to a threshold of 0.5. Although weighting greatly affects the accuracies and confusion matrices (Table 1), the corresponding ROC curves are very similar.

(b) ROC curves for undersampling. The “undersampled” curve contains 1/10 the number of false alarms as the “full” curve. Adding false alarms while increasing class imbalance by a factor of 10 actually improves the classifier ROC curve at most FPRs.

Figure 1: ROC curves for imbalance

2.2 Importance of false alarms

Another approach to dealing with class imbalance is to undersample. That is, to train a network using a subsampling of the majority (negative) examples. To test the effectiveness of this approach, we trained a network using only the example snippets found in images containing targets. This reduced the class imbalance from approximately 100:1 to 10:1. Figure 1b shows the ROC curves for the two classifiers. The difference between the two curves ranges between 1 and 15%. The sensitivities at the 0.5 threshold are .773 vs .824 for the undersampled and fully trained classifiers respectively. Thus, the additional negative examples clearly improve the accuracy of the trained classifier even while increasing the imbalance by a factor of 10.

2.3 Data augmentation

Given the relatively small number of positive examples, data augmentation is an important regularization tool to avoid overfitting. We dynamically augment the data such that a random set of transformations are applied to each image prior to computing the gradient for each mini-batch. These transformations include rotation of up to 5 degrees, translation of up to 10% in the range in cross-range directions, zooming in or out up to 20%, flipping the image in the cross-range direction, and performing a channel intensity shift of up to .5%. Table 2 shows the three augmentation approaches used here. Figure 2 compares the ROC curves for networks trained using the three settings. The results suggest that aggressive data augmentation is extremely beneficial for training a network with our dataset. This is likely due to the small number of available positive examples.
Table 2: Data augmentation approaches

<table>
<thead>
<tr>
<th>Augment</th>
<th>Rotate</th>
<th>Translate</th>
<th>Flip</th>
<th>Zoom</th>
<th>Channel Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>Augment 1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Augment 2</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>No Augment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: ROC curves for two dynamic augmentation approaches vs no augmentation. The augmentation methods are summarized in Table 2. Strong augmentation gives a substantial improvement at most FPRs.

2.4 Network Depth

It has been shown that deeper networks are able to learn better, more expressive representations. Figure 3 shows ROC curves for Resnet-18, Resnet-34, and Resnet-50, each trained using the same batch size, learning rate schedule, and learning rule. It appears that increasing network depth past 18 layers has little effect on classification rate. We speculate that a larger number of training examples, or further separating the examples by class, will be required to exploit the additional layers.

3. MULTIBAND CLASSIFICATION

Figure 3: ROC curves for three Resnet depths. The curves are all similar, indicating that, for this dataset, not much representational power is gained by adding layers beyond 18.
We train deep network classifiers with multiband data by treating the HF and LF bands in the same way conventional deep image classifiers treat the red, green, and blue channels of natural images. Clearly, differences between the HF/LF and color channels exist. Firstly, RGB channels are always perfectly aligned. Secondly, each of the color channels largely share the same statistical properties. The same cannot be guaranteed for the HF and LF SAS channels, as we will show. In the following sections we investigate these differences and their effect on performance.

3.1 Band alignment

SAS data undergoes geometric transformations during the process of image formation. Array positions are estimated and the data may undergo motion compensation prior to beamforming. Errors in the process will result in defocused imagery and localization errors. This may also result in misalignment between imagery from independent sonar bands if the different bands do not undergo precisely the same motion estimation and compensation processes. Even in cases where care is taken to ensure that the motion estimation and compensation is consistent across bands, misalignments can occur: cross-range misalignment can occur when the transmitters or receivers are not co-located, and range misalignment can be introduced during pulse or range compression.

Figure 4a shows a 2d histogram of misalignment errors measured in pixel shifts between co-registered HF and LF imagery. The centroid of the distribution appears to have a bias shifted by 3 pixels in cross-range. This shift is consistent with the expected shift related to the difference in location of the HF and LF transmitters and the corresponding shift in the along-track phase center locations. There is an approximately -1 pixel shift in range for the primary centroid, likely due to an unintended shift in the matched filters used for range compression of the data. There is also an outlier distribution that can be seen with similar along-track shifts but with an exaggerated range shift between the bands. This data was produced with a different transmitted signal/signal replica pair and the range compression step resulted in additional unintended error.

Misalignment error can lead to decreased classifier performance. The underlying assumption for multichannel convolutional filters is that certain features are found together on each of the channels. Misalignment error can lead to HF and LF snippets to be aligned “randomly,” which can result in learned low-level multichannel features to be independent across channels. Figure 4b shows the ROC curves for networks trained on aligned and unaligned datasets. Using aligned data results in a noticeable improvement at almost every FPR. Figure 4b also shows the ROC curve for a network trained on the HF band only. We urge practitioners to verify bands are aligned by cross-correlation before training, as misalignment may not be readily apparent by eye.
3.2 Comparison with RVM

A Relevance vector machine (RVM)\textsuperscript{12} is “shallow” classifier, closely related to the support vector machine (SVM). It produces a non-linear decision boundary between two classes by maximizing the margin or boundary distance between them. Instead of learning a representation, as with a deep network, we use expertly engineered features. More specifically, we use a combination of Fourier and Haar features. These features have been empirically validated on a large number of SAS datasets from varying environments. Figure 5 compares ROC curves of Resnet-18 and an RVM classifier. The improvement by the deep network is most significant at low FPR; at high FPR the difference narrows. The extremely poor performance of the RVM relative to the Resnet can be attributed to not generalizing well to a very difficult test set.

3.3 Transfer learning

It has been shown that the representations learned by deep neural networks trained on a large natural image dataset outperform traditional feature extraction methods when applied to arbitrary natural images\textsuperscript{13}. It is well known that first layer convolutional neural network filters tend to self-organize as V1-like Gabor filters when trained using natural image data (Figure 6a). In fact, checking that the first-layer filters have organized this way has become somewhat of a “sanity check” to see if a new architecture is learning. This same effect is also seen when applying unsupervised algorithms such as sparse filtering or independent component analysis (ICA) to natural images for learning overcomplete bases. It has long been argued that the primary visual cortex (V1) consists of a collection of oriented Gabor-like filters, used as edge-
4. CONCLUSION

In this paper, we discussed training deep networks for multiband SAS. We compared various hyperparameters and discussed how they affected classification performance. We discussed the effect of pre-training with natural image data. We showed that excellent results can be achieved by ensuring co-registered channels, weighting samples by class, and using aggressive data augmentation.

References


