End-to-End ATR Leveraging Deep Learning

Matthew P. Masarik, Chris Kreucher, Kirk Weeks, and Kyle Simpson

1 Overview

Synthetic aperture radar (SAR) systems are widely used for intelligence, surveillance, and reconnaissance purposes. However, unlike electro-optical (EO) images, SAR images are not easily interpreted and therefore have historically required a trained analyst to extract useful information from images. At the same time, the number of high-resolution SAR systems and the amount of data they generate are rapidly increasing, which has resulted in a shortage of analysts available to interpret this vast amount of SAR data. Therefore, there is a significant need for efficient and reliable automatic target recognition (ATR) algorithms that can ingest a SAR image, find all the objects of interest in the image, classify these objects, and output properties of the objects (location, type, orientation, etc.). This chapter lays out the required steps in any approach for performing these functions and describes a suite of deep learning (DL) algorithms that perform this end-to-end SAR ATR. One novel feature of our method is that we rely on only synthetically generated training data, which avoids some of the main pitfalls of other DL approaches to this problem.

M. P. Masarik · C. Kreucher (✉)
KBR, Ann Arbor, MI, USA
e-mail: matthew.masarik@us.kbr.com; chris.kreucher@us.kbr.com; ckreuche@umich.edu

K. Weeks · K. Simpson
Signature Research, Inc., Navarre, FL, USA
e-mail: kweeks@signatureresearchinc.com; ksimpson@signatureresearchinc.com

M. Rysz et al. (eds.), *Synthetic Aperture Radar (SAR) Data Applications*, Springer Optimization and Its Applications 199, https://doi.org/10.1007/978-3-031-21225-3_1

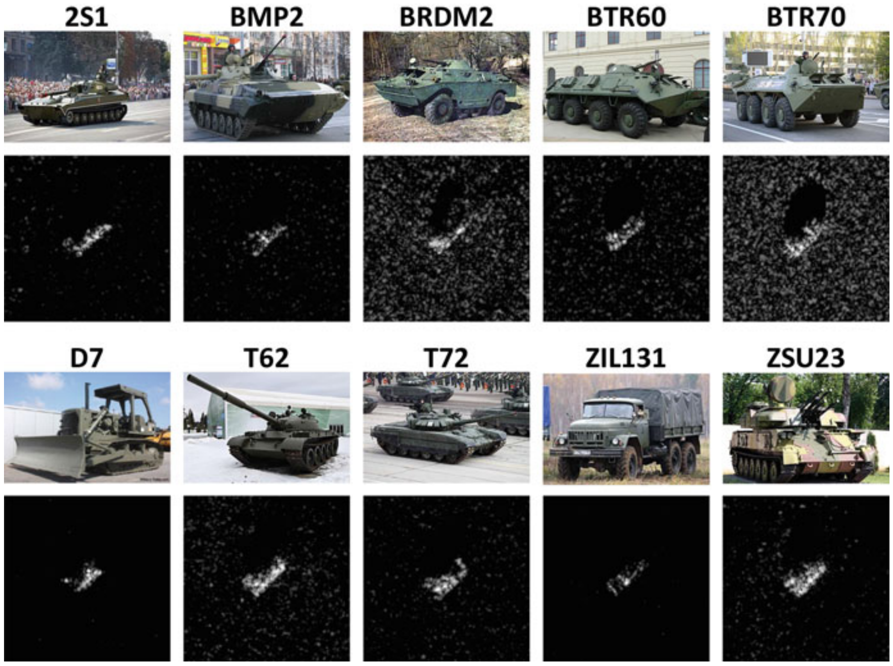


Fig. 1 Comparison of EO images and SAR images of the 10-class MSTAR target set. The significant differences between EO and SAR images make interpretation of SAR imagery non-trivial and traditionally reliant on trained analysts

2 Introduction

Synthetic aperture radar (SAR) is a powerful remote sensing technique that coherently processes a sequence of radar returns to form radar images [1, 2]. SAR is an active imaging mode, transmitting the microwave radio frequency (RF) energy it uses to make images; hence, a SAR sensor can operate day or night in all-weather conditions.

However, SAR images contain a significant amount of speckle noise, and the images have typically much lower resolution than electro-optical (EO) imagery. Moreover, SAR images are sensitive to the direction of illumination because of both shadowing and self-shadowing effects. These features mean that conventional image classification methods applied to EO imagery are not directly applicable to SAR imagery. Figure 1 shows some example EO images and SAR images of the publicly available 10-class Moving and Stationary Target Acquisition and Recognition (MSTAR) [3, 4] target set, providing one illustration of these differences.

Analogously, as shown in Fig. 2, full-scene SAR imagery includes a number of features not present in EO imagery that makes the tasks of target detection, orientation estimation, and classification more difficult. These include residual defocus, non-uniform illumination, and distinct background statistics that stem from

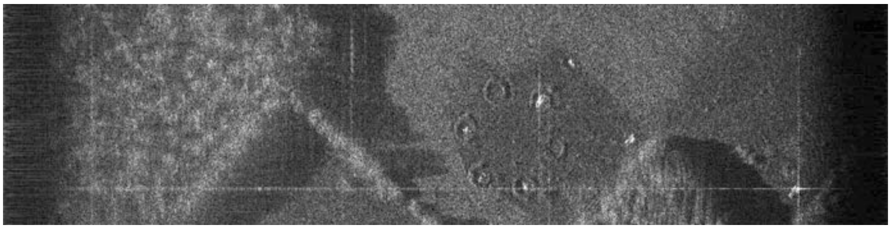


Fig. 2 A SAR image from Mission 78, Pass 1 collected by the Advanced Detection Technology Sensor (ADTS), an airborne SAR/RAR millimeter-wave sensor operated by MIT Lincoln Laboratory. The ADTS operates at Ka band (32.6 to 37 GHz) and was used to collect clutter and armor scenes using its stripmap mode

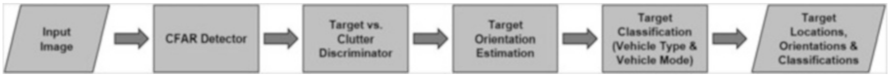

Input Image
CFAR Detector
Target vs. Clutter Discriminator
Target Orientation Estimation
Target Classification (Vehicle Type & Vehicle Mode)
Target Locations, Orientations & Classifications

Fig. 3 A flow diagram describing an example ATR processing chain

the physical nature of the ground cover (e.g., tree cover, water, dirt, road). Because of these factors, SAR imagery has been traditionally interpreted by human experts rather than machine automation.

The number of high-resolution SAR systems and the amount of data they generate are rapidly increasing, which has resulted in a shortage of analysts available to interpret this vast amount of SAR data. There is a significant need, therefore, for efficient and reliable algorithms that can ingest a SAR image, find all the objects of interest in the image, classify these objects, and output properties of the objects (type, size, orientation, etc.). We will refer to this suite of algorithms to as end-to-end SAR ATR (Automated Target Recognition). We note that some authors use the term ATR in reference to just the final step, that of classifying image chips that are known to contain objects of interest. In this chapter, we treat the entire end-to-end problem that contains all aspects starting from a full SAR scene and ending with a collection of classified image chips. A block diagram showing the steps involved in a representative end-to-end SAR ATR algorithm is shown in Fig. 3.

Note that while some ATR algorithms combine a constant false alarm rate (CFAR) detector and target–clutter discriminator into a single algorithm, the algorithm discussed in this chapter splits the detection task into these two stages. Similarly, some classification algorithms do not rely on knowledge of the target orientation, but the algorithm presented here exploits this estimate. Hence, different algorithms will result in slight modifications to the flow diagram in Fig. 3, but the basic structure of detect, characterize, classify, and output underpins all ATR algorithms.

Algorithms for SAR ATR have been studied for many years [5], receiving increased attention in the literature over the past 20 years in large part due to the public release of the high-resolution MSTAR dataset described earlier [3, 4]. The MSTAR dataset consists of SAR image chips of ten military vehicles collected with an airborne sensor using approximately 600 MHz of bandwidth at X-band with images formed to approximately 1 foot pixel spacing. Because the classification component of SAR ATR is widely regarded as challenging and the MSTAR dataset includes image chips known to contain vehicles, much of the community's focus has been on the classification component of SAR ATR rather than the front-end steps of detection and target characterization.

Many of the early SAR classification algorithms were template-matching approaches, which estimate target class by choosing the class corresponding to the template that best matches the data [6]. Some other early algorithmic approaches include the attributed scattering center model approach [7], support vector machines [8], and neural networks [9]. An excellent summary of the state of the art through approximately 2016 is given in [10], and a updated survey was recently given in [11]. Some of the most important recent efforts employ modern convolutional neural network (CNN) approaches [12–15].

While the MSTAR dataset has provided an excellent testbed for SAR ATR classification that is easily accessible, in recent years it has led to considerable misunderstanding and inflated performance predictions. The issue is that the typical MSTAR experiment presented in the literature uses a training set and a testing set (nominally collected at 17° and 15° elevation, respectively), that are so similar that nearly any reasonable technique should be capable of achieving very high (e.g., >99%) accuracy. The datasets are so similar for several reasons:

1. The targets in the training and testing sets were the exact same vehicle (i.e., the exact same T-72).
2. The targets were on the exact same patch of ground across the training and testing sets.
3. The training and testing data were collected on successive flight passes.
4. The sensors collecting the data and the image formation procedure were identical across the training and testing sets.

This level of similarity between the training and testing sets is unreasonable to expect in practice [11]; hence, the literature tends to exaggerate algorithm performance. Even more damning, however, is that due to the targets being on the same patch of ground across the training and testing sets, the authors have demonstrated that classification accuracies of >70% are achievable *using only the background clutter of the chips*.

Hence, even if an algorithm demonstrates excellent performance on the standard MSTAR experiment, it is unclear if the algorithm is performing target discrimination that will generalize to more realistic ATR scenarios or is just memorizing particulars of the vehicles and terrain. It is also unclear if the learned algorithms will perform well when applied to data collected by a different sensor.

To address these issues, as well as the relative dearth of data compared to optical images, this chapter describes an end-to-end SAR ATR approach that utilizes synthetically generated training data (e.g., asymptotic ray-tracing predictions based on target CAD models). In addition, our approach is novel as it is a hybrid method that combines the robustness of conventional algorithms (e.g., template matching and CFAR detectors) with the performance improvements possible using emerging deep learning (DL) techniques.

This chapter proceeds as follows: Sect. 3 describes an approach to the elements of the front-end ATR tasks (target detection and orientation estimation) that combines a conventional CFAR algorithm as a prescreener and a DL algorithm as a final target–clutter discriminator; next, Sect. 4 describes a hybrid conventional/DL approach to target classification that casts a template-matching algorithm into the DL framework and then allows learning to refine the templates (features) and also learns the interclass relationships; and finally, Sect. 5 concludes.

3 Front-End Algorithms

This section discusses the front-end SAR processing chain that begins with a collected SAR image and generates a series of target-centered image chips to be classified, as well as an orientation estimate of the object in the chip. These steps are highlighted in Fig. 4. In our approach, we perform this front-end function by first executing a CFAR detector to separate target-like areas from the background, then carrying out a DL-based target discrimination to separate true targets from target-like clutter, and finally carrying out an orientation estimation to predict the target angle relative to the image.

As is standard in the SAR community, we work with approximately 600 MHz bandwidth data with images formed with approximately 1 ft pixel spacing. For the MSTAR targets and graze angles, chips of size 128×128 are sufficient to capture both the target and its shadow. This chip size is also standard in the literature. All of our algorithms can be scaled appropriately if the chip sizes change either due to resolution or target size.

3.1 Target Detection

A comprehensive review of detection algorithms for SAR ATR can be found in [10]. The target detection algorithm described here employs a two-stage approach. The first stage is a prescreener that is implemented as a cell-averaged constant false alarm rate (CFAR) detector that models the clutter as Rayleigh-distributed. The second stage is a data-driven convolutional neural network (CNN) that predicts a probability that the image contains a target. This algorithm is illustrated in Fig. 5.

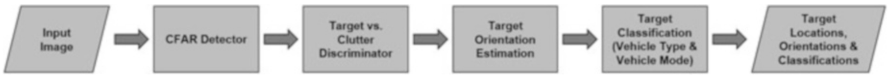

Input Image
CFAR Detector
Target vs. Clutter Discriminator
Target Orientation Estimation
Target Classification (Vehicle Type & Vehicle Mode)
Target Locations, Orientations & Classifications

Fig. 4 The end-to-end SAR ATR processing chain. This section describes algorithms for the operations outlined in red

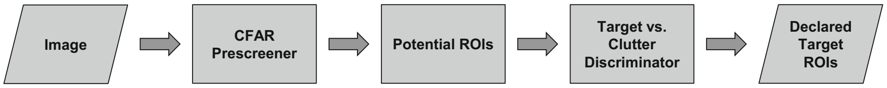


Fig. 5 Flow diagram describing the target detection algorithm

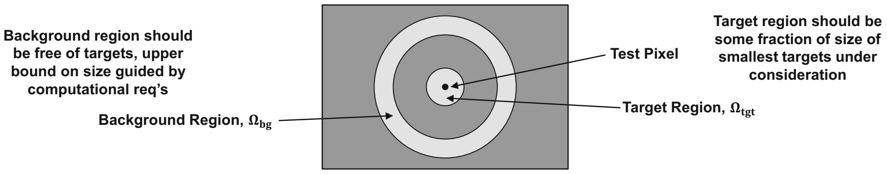

Background region should be free of targets, upper bound on size guided by computational req's
Background Region, Ω_{bg}
Test Pixel
Target region should be some fraction of size of smallest targets under consideration
Target Region, Ω_{tgt}

Fig. 6 Definition of target and background regions used in the CFAR

3.1.1 CFAR Prescreener

The prescreener is implemented as a CFAR detector. To derive the CFAR detector, consider Fig. 6, which defines the background region Ω_{bg} and the target region Ω_{tgt} for a test pixel. The regions are defined so that the inner diameter of Ω_{bg} is approximately as large as the largest target to be encountered, the thickness of Ω_{bg} is approximately the expected minimum spacing between targets (though the upper bound on the size of this region is dictated by computational requirements), and the diameter of Ω_{tgt} is approximately the size of the smallest object to be encountered.

The statistics of the non-target (clutter) regions are well studied [16, 17]. Here we have elected to model the clutter as Rayleigh-distributed. The maximum-likelihood estimates (MLEs) of the Rayleigh shape parameter in the background and target regions are given, respectively, in Eqs. (1) and (2):

$$\hat{\sigma}_{bg}^2 = \frac{1}{2|\Omega_{bg}|} \sum_{\Omega_{bg}} |I|^2 \tag{1}$$

$$\hat{\sigma}_{tgt}^2 = \frac{1}{2|\Omega_{tgt}|} \sum_{\Omega_{tgt}} |I|^2. \tag{2}$$

Detection of anomalies is thus reduced to the following hypothesis-testing problem:

$$\begin{aligned} H_0 &: \hat{\sigma}_{tgt}^2 \leq \hat{\sigma}_{bg}^2 \\ H_1 &: \hat{\sigma}_{tgt}^2 > \hat{\sigma}_{bg}^2. \end{aligned} \tag{3}$$

A reasonable test statistic for this problem is

$$T = \frac{\hat{\sigma}_{tgt}^2}{\hat{\sigma}_{bg}^2} = \frac{\frac{1}{2|\Omega_{tgt}|}\sum_{\Omega_{tgt}} |I|^2}{\frac{1}{2|\Omega_{bg}|}\sum_{\Omega_{bg}} |I|^2} = \frac{|\Omega_{bg}|\sum_{\Omega_{tgt}} |I|^2}{|\Omega_{tgt}|\sum_{\Omega_{bg}} |I|^2}. \tag{4}$$

The hypothesis test to declare a detection is then

$$T \underset{H_0}{\overset{H_1}{\gtrless}} \gamma, \tag{5}$$

where γ is some threshold.

It remains then to determine how to set the threshold γ in terms of a desired probability of false alarm (P_{FA}), which requires a model on the distribution of T. First, as commonly done [18], we model the SAR image pixels as 0-mean complex Gaussian with variance σ^2 in I and Q. This gives Rayleigh statistics for the detected pixels, and if the pixel values were independent, the sum over a region would be Gamma-distributed, i.e.,

$$\sum_{\Omega_{bg}} |I|^2 \sim \Gamma\left(|\Omega_{bg}|, \frac{1}{2\sigma^2}\right). \tag{6}$$

However, the SAR image formation process introduces a correlation between the pixels. The exact distribution of the sum of correlated Rayleigh variates is complicated [19, 20], but a useful approximation [21] is

$$\sum_{\Omega_{bg}} \frac{|I|^2}{\sigma^2/2} \sim \Gamma\left(\frac{|\Omega_{bg}|}{u}, 2u\right), \tag{7}$$

where $u = 1 + 2\rho(|\Omega_{bg}| - 1)$, with ρ capturing the average correlation in the region under sum. This is an empirical quantity which we estimate offline from a background dataset. With this approximation, the test statistic T is seen to be a ratio of independent Gamma-distributed variables, which is F-distributed as

$$T \sim F\left(\frac{2|\Omega_{tgt}|}{1 + 2\rho(|\Omega_{tgt}| - 1)}, \frac{2|\Omega_{bg}|}{1 + 2\rho(|\Omega_{bg}| - 1)}\right). \tag{8}$$

Thus, in terms of a desired P_{FA}, the threshold γ can be determined in terms of the inverse F-distribution, and the final hypothesis test is

$$T \underset{H_0}{\overset{H_1}{\gtrless}} F^{-1}\left(1 - P_{FA}, \frac{2|\Omega_{tgt}|}{1 + 2\rho(|\Omega_{tgt}| - 1)}, \frac{2|\Omega_{bg}|}{1 + 2\rho(|\Omega_{bg}| - 1)}\right). \tag{9}$$

Note that this test is performed for every pixel in the image but is still computationally efficient because the summation terms may be computed quickly

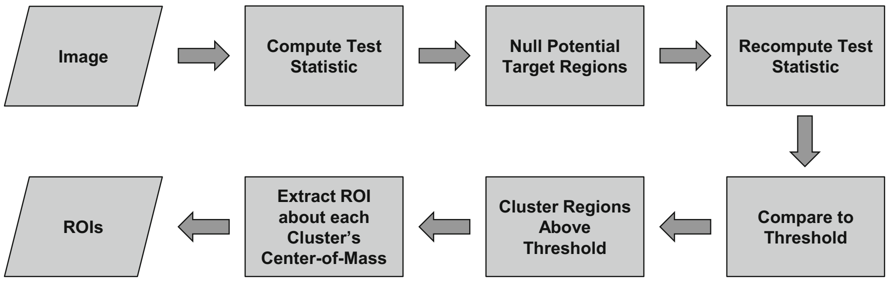


Image
Compute Test Statistic
Null Potential Target Regions
Recompute Test Statistic
Compare to Threshold
Cluster Regions Above Threshold
Extract ROI about each Cluster's Center-of-Mass
ROIs

Fig. 7 Flow diagram for the anomaly detector

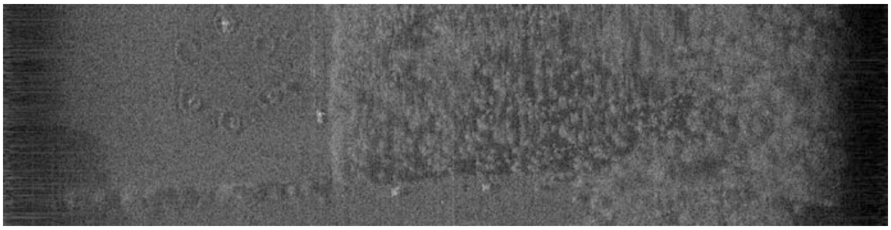


Fig. 8 An example input SAR image from the publicly available Advanced Detection Technology Sensor (ADTS) collection

via convolution. Additional gains in efficiency are possible by making the target and background regions rectangular and using integral images to compute the summations.

Since the background statistics are computed via averaging, they are susceptible to contamination by targets or other objects. This suggests that a censoring procedure should be employed wherein the prescreener is run once, and then it is run again while ignoring detections. Figure 7 shows the flow diagram for the anomaly detector.

Figure 8 shows an example SAR image containing an open field populated with four target vehicles (the image is the HH polarization image of frame 22, pass 7, mission 78 from the publicly available ADTS sample set images available on the AFRL Sensor Data Management System). In addition to the targets, the image contains a group of trees on the right side of the image.

Figure 9 shows the (transformed) CFAR test statistic as computed via Eq. (9). As expected, the bright areas correspond to the targets, but the regions around these targets are unnaturally low due to the corruption of the background statistics by the target.

Figure 10 shows the CFAR test statistic that has been recomputed by masking out detections. This results in a significantly cleaner CFAR image.

Finally, Fig. 11 shows detected anomaly regions outlined with a yellow box. Note that, apart from the five detected regions in the trees at the right part of the image, all the other detections are desirable anomaly detections as they correspond to targets.

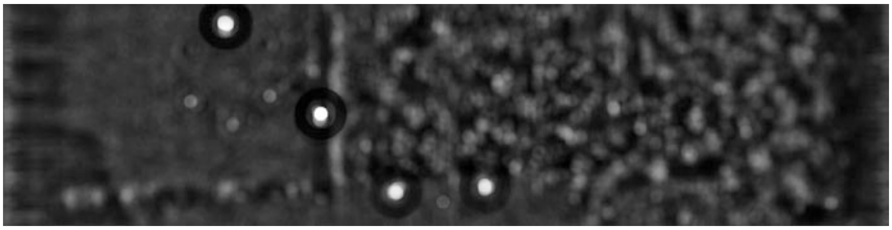


Fig. 9 The test statistic computed on the input SAR image without masking detections

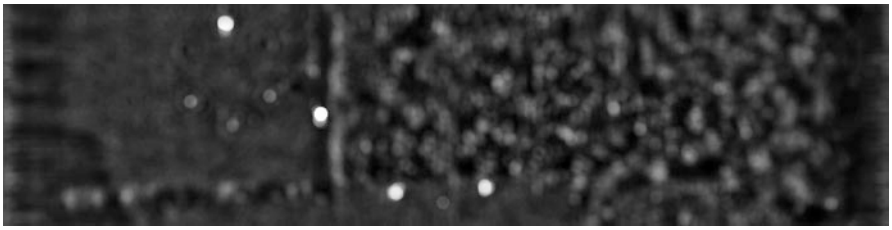


Fig. 10 The test statistic recomputed with initial detections masked

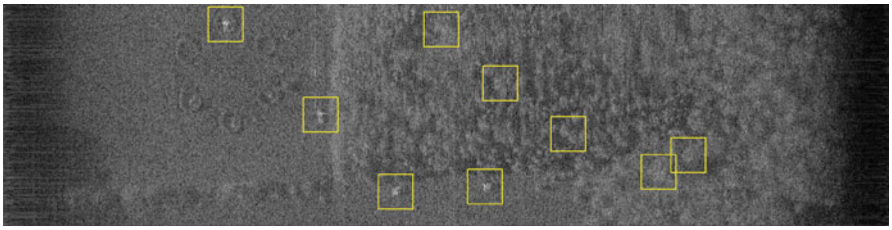


Fig. 11 Image with anomalous regions outlined with a yellow box

A method for dismissing the regions that do not contain targets will be discussed in Sect. 3.1.2.

3.1.2 Target vs. Clutter Discriminator

As seen in Fig. 11, the CFAR will detect some objects that, while they are statistically anomalous, are not targets and therefore should not be passed on for discrimination (e.g., the detections in the trees). This motivates the use of a second stage in the detection algorithm with the purpose of screening out detections that are clearly uninteresting from an ATR perspective. There are many approaches to designing such a screening algorithm (see [22] for a summary of previously applied techniques), but with the recent advances in computer vision via deep learning, it is natural to use deep learning techniques to build a target vs. clutter discrimination algorithm.

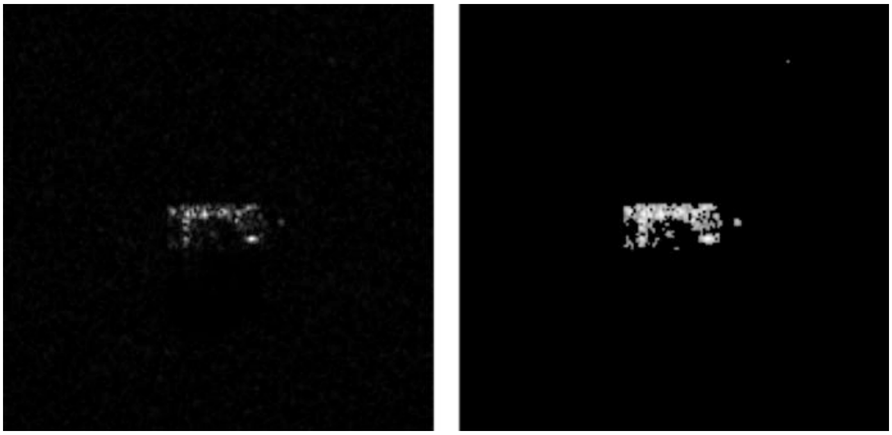


Fig. 12 Comparison of a SAR image chip (left) and preprocessed chip (right)

The target vs. clutter discriminator described here is constructed using a novel CNN architecture trained with synthetic SAR target data and historical measured SAR clutter data to classify a given region of interest (ROI) as containing a target or not. This discriminator is designed to be robust across sensors, terrain, and targets. In addition to training the discriminator on large amounts of diverse target and clutter data and applying data augmentation techniques, robustness is improved by feeding the network thresholded and remapped input images, where only the top-N magnitude pixels are kept and the values are remapped by linearly rescaling the dB-domain image to the interval [0, 1]. This preprocessing technique is effective because it maintains the shape of the anomaly detected by the CFAR (which allows for target vs. clutter discrimination), it removes absolute amplitude data (which improves discrimination robustness to sensor, target, and terrain variations), and it maps to a more visually relevant space for the pixel values (which facilitates the use of a CNN). An example target chip and its preprocessed counterpart are shown in Fig. 12. In this example, $N = 400$ and 40 dB of dynamic range was mapped to [0, 1].

The architecture of the target vs. clutter discriminator CNN is shown in Table 1. The CNN is comprised of standard layers ($2D$ convolutions, $2D$ max pooling, batch normalization, dropout, and fully connected layers) and contains $3{,}548{,}745$ trainable parameters.

The network is trained using synthetically generated target data and historical clutter data. The network is trained using the binary cross-entropy loss, i.e., for a predicted output y_p and a truth label y_t:

$$\ell(y_p, y_t) = -y_t \log(y_p) - (1 - y_t) \log(1 - y_p). \tag{10}$$

To account for targets that are not centered in the chip, the discriminator is trained using random translational augmentations for each training chip. The ADADELTA optimizer is used to train the discriminator [23].

Table 1 SAR target vs. clutter discriminator CNN architecture

Layer name	Layer parameters	Output size	No. of parameters
Input layer	128×128 input size	128×128	N/A
2D convolution	40 kernels of size 20×20	109×109×40	16,040
Batch normalization	N/A	109×109×40	160
Dropout	Dropout fraction = 0.2	109×109×40	0
2D Max pooling	Pool size = 2×2	54×54×40	0
2D convolution	80 kernels of size 15×15	40×40×80	720,080
Batch normalization	N/A	40×40×80	320
Dropout	Dropout fraction = 0.2	40×40×80	0
2D Max pooling	Pool size = 2×2	20×20×80	0
2D convolution	160 kernels of size 10×10	11×11×160	1,280,160
Batch normalization	N/A	11×11×160	640
Dropout	Dropout fraction = 0.2	11×11×160	0
2D convolution	320 kernels of size 5×5	7×7×320	1,280,320
Batch normalization	N/A	7×7×320	1280
Dropout	Dropout fraction = 0.2	7×7×320	0
Flatten	N/A	15,680×1	0
Fully connected	16 output nodes	16×1	250,896
Batch normalization	N/A	16×1	64
Dropout	Dropout fraction = 0.2	16×1	0
Fully connected	1 output node	1	17

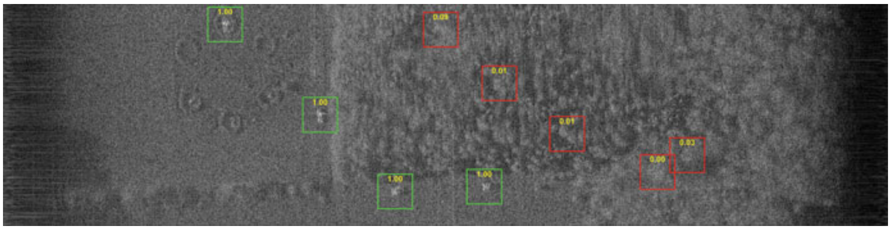


Fig. 13 Example SAR image with CFAR anomaly detections

The final result of our two-stage detector (CFAR plus target–clutter discriminator) is shown in Fig. 13. Boxes in green have passed both stages, while boxes in red have been flagged by the CFAR stage but rejected at the second stage. In this example, we find that after the second stage, all of the true targets have been detected, while no false targets are detected.

3.2 Target Orientation Estimation

After detection, we next characterize the vehicle by performing target orientation estimation. The goal of the target orientation estimation algorithm is to estimate the aspect angle of the target within the SAR image (c.f., Fig. 14). Robust, accurate estimation of this parameter is important because it can be used to improve target classification performance. In template-matching methods, knowledge of the aspect angle reduces the number of comparisons required to declare a target, which improves efficiency but also reduces the number of potential false matches. In the classification algorithm discussed here, the aspect angle is used to give preference to those nodes of the CNN that correspond to angles close to the input chip's aspect angle.

There are several traditional image-processing-based algorithms for determining this angle. One such example is based on the Radon transform that preprocesses the image, computes the discrete Radon transform of the image [24], and then selects the angle corresponding to the maximum of the Radon transform of the image. While these algorithms perform well in some cases, their performance leaves much to be desired, especially for target images without a prominent edge. To address this deficiency, we have developed a CNN approach to aspect angle estimation. A high-level description of the algorithm is shown in Fig. 15.

The appropriate loss function for the CNN is not obvious due to angle wrapping. Moreover, the authors have found that achieving accurate angle estimation better than modulo 180° is unrealistic due to vehicle symmetry and a lack of detail in SAR images. Hence, the loss function that was used for the angle estimation algorithm is

$$\ell(\theta_{est}, \theta_{true}) = |\sin(\theta_{est} - \theta_{true})|. \tag{11}$$

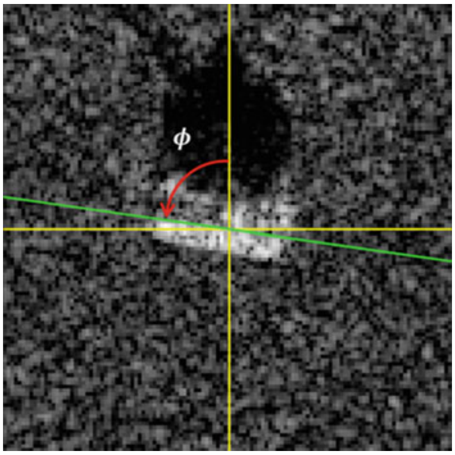


Fig. 14 The orientation angle ϕ is useful for the classification algorithm that is the final stage of the end-to-end algorithm

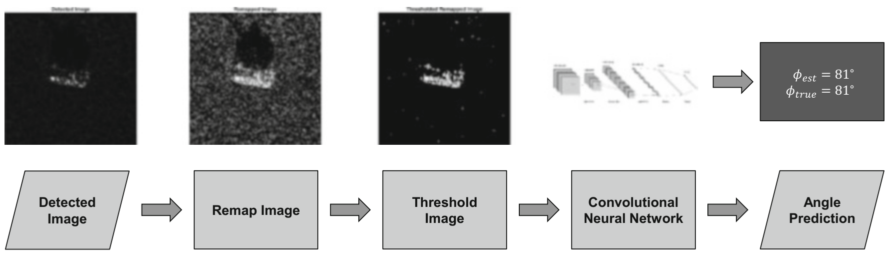


Fig. 15 High-level description of the orientation angle estimation algorithm

Table 2 Architecture of the aspect angle estimation CNN

Layer name	Layer parameters	Output size	No. of parameters
Input layer	128×128 input size	128×128	N/A
2D convolution	10 kernels of size 40×40	89×89×10	16,010
Batch normalization	N/A	89×89×10	40
Dropout	Dropout fraction = 0.2	89×89×10	0
2D convolution	20 kernels of size 40×40	70×70×20	80,020
2D Max pooling	Pool size = 2×2	35×35×20	0
Batch normalization	N/A	35×35×20	80
Dropout	Dropout fraction = 0.2	35×35×20	0
2D convolution	30 kernels of size 10×10	26×26×30	60,030
Batch normalization	N/A	26×26×30	120
Dropout	Dropout fraction = 0.2	26×26×30	0
2D convolution	40 kernels of size 5×5	22×22×40	30,040
Batch normalization	N/A	22×22×40	60
Dropout	Dropout fraction = 0.2	22×22×40	0
Flatten	N/A	19,360×1	0
Fully connected	32 output nodes	32×1	619,552
Batch normalization	N/A	32×1	128
Dropout	Dropout fraction = 0.2	32×1	0
Fully connected	1 output node	1	33

The loss is ambivalent to $\pm 180°$ estimation errors and is maximized for $\pm 90°$ errors. Note the actual implementation is a softened version of this function to overcome the discontinuity in the derivative at 0° and multiples of $\pm 180°$.

The architecture of the orientation estimation CNN is given in Table 2. The CNN consists of $805,949$ trainable parameters and is trained using only synthetically generated training data.

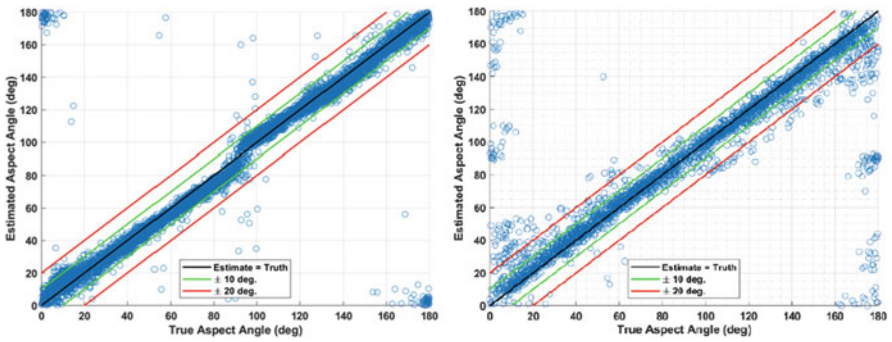


Fig. 16 Left: Deep learning aspect angle estimation performance. Right: Radon transform aspect angle estimation performance

Table 3 Aspect angle estimation algorithm performance summary on the MSTAR dataset

Threshold	Deep learning % within threshold	Radon % within threshold
±10°	96.3%	84.4%
±20°	99.4%	92.1%

The deep learning algorithm and a Radon transform algorithm performance on the MSTAR dataset are shown in Fig. 16. The performance of the algorithms is summarized in Table 3. The figures and the table show that the deep learning algorithm performs excellently and significantly outperforms the Radon transform algorithm.

4 Classification Algorithm

This section describes the final stage of the end-to-end ATR algorithm, which is the classification (i.e., declaration of object type and class) of a chip nominated by the front-end processing. The classification algorithm presented here is a novel hybrid of a classical template-matching algorithm and a deep learning CNN image classification algorithm. The motivation for this approach is the desire for an algorithm that would:

1. Maintain the robustness of template-matching algorithms to variations in sensor, sensor geometry, clutter, and minor target variations.
2. Improve performance of the template-matching algorithm by using more complex information from each input image as well as information about how an input relates to each of the different target classes.

The new algorithm builds on standard template-matching algorithms [10], which nominally consist of the following steps:

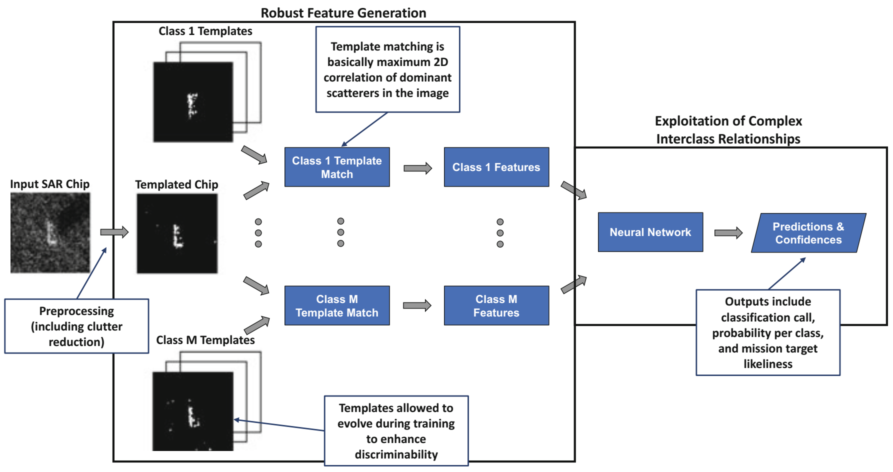

Robust Feature Generation
Class 1 Templates
Template matching is basically maximum 2D correlation of dominant scatterers in the image
Class 1 Template Match
Class 1 Features
Exploitation of Complex Interclass Relationships
Input SAR Chip
Templated Chip
Neural Network
Predictions & Confidences
Class M Template Match
Class M Features
Preprocessing (including clutter reduction)
Class M Templates
Outputs include classification call, probability per class, and mission target likeliness
Templates allowed to evolve during training to enhance discriminability

Fig. 17 A high-level overview of the proposed algorithm. The first stage learns features, and the second stage learns interclass relationships

1. Estimate the target orientation.
2. Preprocess the chip by keeping only the top N-valued pixels and then binning the pixels into discrete bins between zero and one.
3. Compute the maximum 2D correlation between the preprocessed test chip and each processed training set template chip near the estimated orientation.
4. Declare the predicted target class as the class of the training chip that achieves the maximum correlation.

Our hybrid deep learning/template-matching algorithm first recasts template matching as a CNN with large kernels (with size on the order of the target size in pixels) and then allows network weights to evolve (i.e., the templates to deform) to improve discrimination. The final step of the algorithm is a fully connected layer that maps feature-match scores to a score indicating the likelihood of belonging to each target class. Note that this is different from a classic template-matching step where the declared target class is the class of the best template match. Instead, each training class contributes to the classification call of the input test chip, allowing both "positive" and "negative" information to play a part in the decision.

With this as background, the classification CNN can be viewed as splitting classification into two interconnected stages: (1) robust feature generation and (2) exploitation of complex interclass relationships. This is illustrated in Fig. 17.

Our CNN implementation incorporates the four steps in the conventional template-matching algorithm described above. Step 1 (orientation estimation) uses the front-end orientation estimate described earlier; Step 2 (preprocessing) is implemented by preprocessing all chips; Step 3 (correlation estimation) is implemented with a convolution layer followed by a max-pooling operation; and Step 4 (maximization) is softened from a maximum operation to a weighted-sum operation via a fully connected layer.

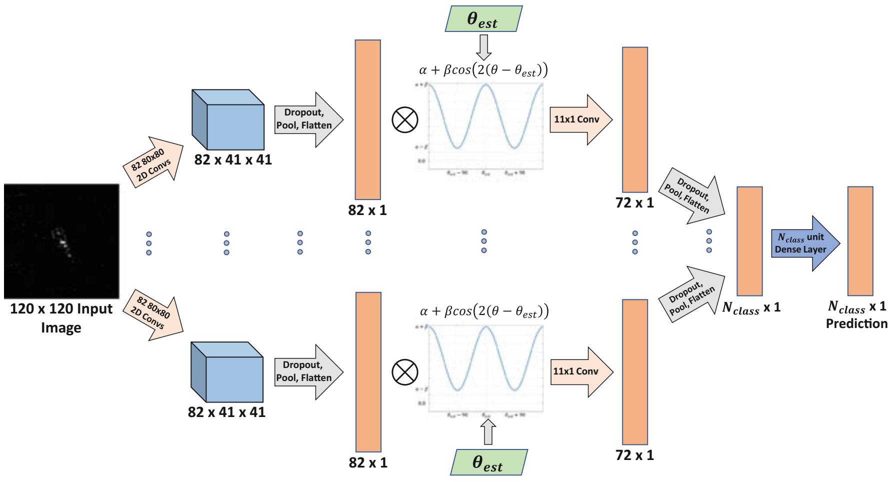


Fig. 18 Detailed architecture of the proposed algorithm

In the absence of training (and with proper initialization on the dense layer), this configuration literally performs conventional template matching. This gives the "epoch 0" (untrained) network a unique advantage over other DL techniques in that it performs fairly well without any training. This can also be viewed as a good initial estimate in the optimization process carried out by training. When the network is trained, both the templates and weights in the final dense layer are updated, resulting in an improvement over template matching that stems from both (i) improved features and (ii) exploitation of interclass relationships. Figure 18 shows a more detailed illustration of the algorithm.

The convolutional kernels in the first stage of the CNN are initialized before training as preprocessed chips at 5° spacing. The estimated azimuth orientation angle is included by applying a weighting function to the template-match scores, which lowers the scores from azimuths far from the estimated azimuth and elevates scores near the correct azimuth (modulo 180°). In addition, to further improve performance: (1) dropout layers have been included after the initial template-match score generation stage as well as just before the final fully connected layer, and (2) a convolution across template-match scores in azimuth with a nominally Gaussian kernel is applied to induce robustness to spurious template matches and errors in the orientation estimation algorithm.

4.1 Algorithm Training

To overcome issues arising from the similarity of the training and testing sets, as well as to demonstrate ATR in the important scenario where collected training data is either scarce or unavailable, the algorithm is trained using only synthetic data. Synthetic data was generated using asymptotic ray-tracing techniques from 3D CAD models of the MSTAR targets. The data was simulated at X-band, with bandwidth and aperture chosen to achieve 1 foot resolution. The image chips were then formed by backprojection of the synthetic data at 1° increments for each of the 10 targets using HH, HV, VH, and VV polarizations, yielding 1440 synthetic chips per target class. Due to reciprocity, we elected to only use HV polarizations.

The algorithm is trained using the ADADELTA optimizer [23] for 50 epochs. Each training chip undergoes a series of data augmentation transformations, including:

1. A random translation of no more than 10% of image size in each dimension.
2. Addition of zero-mean Gaussian noise ($\sigma = 5^\circ$) to the chip orientation angle.
3. Addition of Rayleigh noise to achieve a target-to-clutter ratio of $\sim$ 10 dB. More complicated clutter models can be used at the cost of computational efficiency.

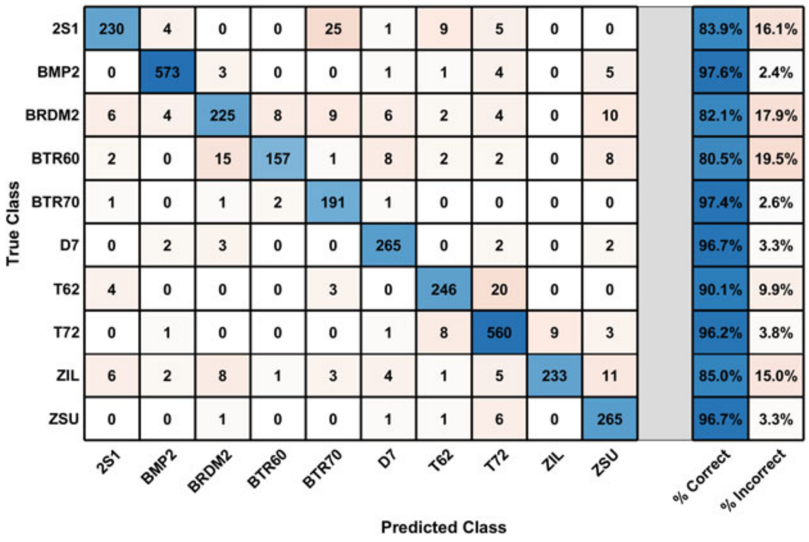
4.2 Validation Experiment

The classification algorithm was validated using the synthetically generated target chips described above for training, and the trained model was tested on the 10-class MSTAR flight-collected dataset at 15° elevation. The algorithm achieved an overall classification accuracy of $\approx$ 92% using fully synthetic training and collected testing data. The detailed classification results of the experiment are summarized by the confusion matrix shown in Fig. 19. It can be seen that the classification performance is 80% or better for each of the ten targets. Moreover, six of the ten targets performed above 90%. Performance on the lower-performing classes could likely be improved by improvements to the CAD model and/or signature generation.

4.2.1 The Learned Network

Examples of the learned features ("templates") of the network are shown in Fig. 20. It is interesting to compare these learned templates with the example chips in Fig. 21. Clearly, there is a strong correspondence between the field-collected targets and the learned templates even though the network was trained on only synthetic data. This shows that the feature extraction stage of the proposed algorithm remains similar to a template-matching algorithm, which is a promising indicator for robustness.

The off-diagonal weights of the fully connected layer mapping from the maximum template-match score for each class to the pre-softmax outputs are shown in Fig. 22. This shows that the dense network has learned a weighted average that is



True Class \ Predicted Class	2S1	BMP2	BRDM2	BTR60	BTR70	D7	T62	T72	ZIL	ZSU	% Correct	% Incorrect
2S1	230	4	0	0	25	1	9	5	0	0	83.9%	16.1%
BMP2	0	573	3	0	0	1	1	4	0	5	97.6%	2.4%
BRDM2	6	4	225	8	9	6	2	4	0	10	82.1%	17.9%
BTR60	2	0	15	157	1	8	2	2	0	8	80.5%	19.5%
BTR70	1	0	1	2	191	1	0	0	0	0	97.4%	2.6%
D7	0	2	3	0	0	265	0	2	0	2	96.7%	3.3%
T62	4	0	0	0	3	0	246	20	0	0	90.1%	9.9%
T72	0	1	0	0	0	1	8	560	9	3	96.2%	3.8%
ZIL	6	2	8	1	3	4	1	5	233	11	85.0%	15.0%
ZSU	0	0	1	0	0	1	1	6	0	265	96.7%	3.3%

Fig. 19 Confusion matrix for MSTAR targets with the algorithm trained on purely synthetic data

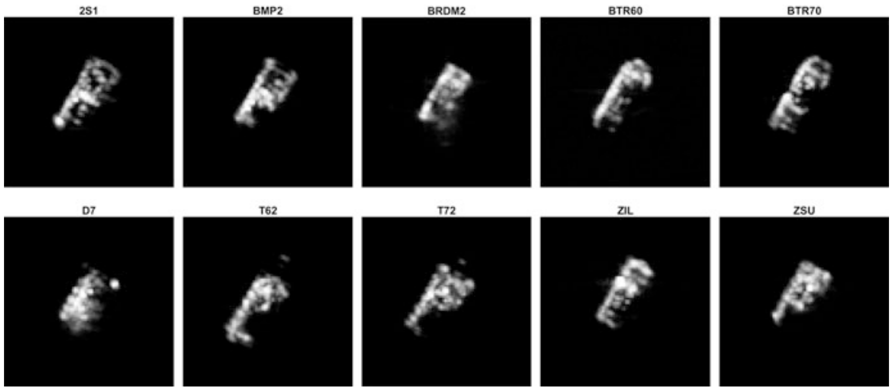


Fig. 20 Examples of the convolutional kernels of the trained network

dependent on the target class, which implies that the network has learned detailed relationships between the target classes. The following observations can be made:

1. The T62 and T72 are strongly attracted to each other.
2. The T72 is attracted to tracked vehicles but is agnostic to or strongly repelled from the wheeled vehicles (BRDM2, BTR60, BTR70, and ZIL131).
3. The 2S1 is strongly attracted to the other tank-like vehicles (BMP2, T62).
4. The BTR70 is strongly attracted to other wheeled vehicles (BRDM2, ZIL131).

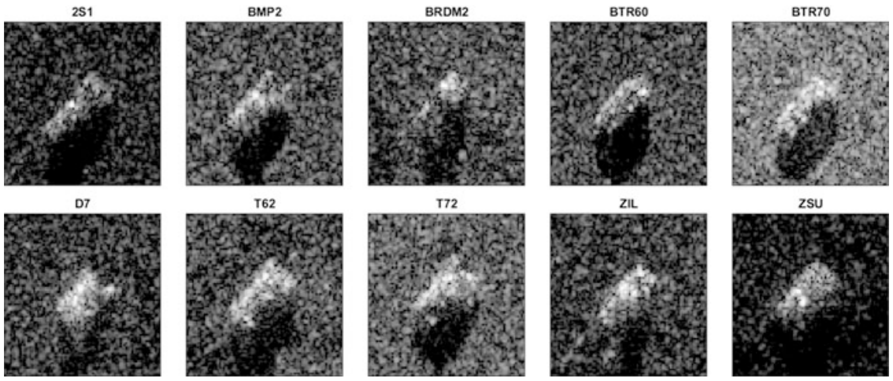
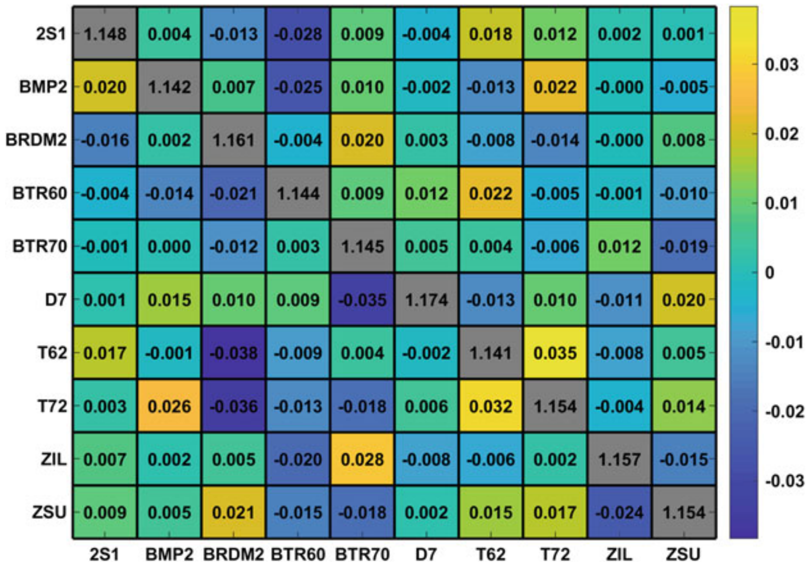


Fig. 21 Examples of the collected MSTAR data

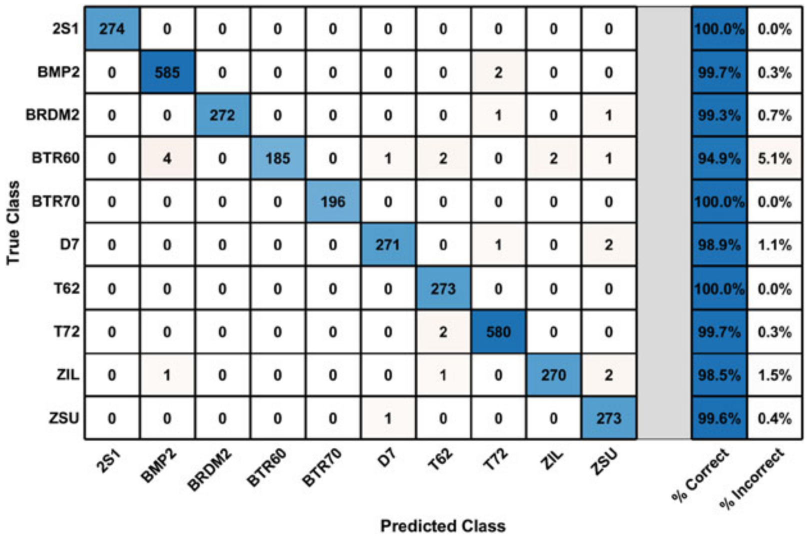


	2S1	BMP2	BRDM2	BTR60	BTR70	D7	T62	T72	ZIL	ZSU
2S1	1.148	0.004	-0.013	-0.028	0.009	-0.004	0.018	0.012	0.002	0.001
BMP2	0.020	1.142	0.007	-0.025	0.010	-0.002	-0.013	0.022	-0.000	-0.005
BRDM2	-0.016	0.002	1.161	-0.004	0.020	0.003	-0.008	-0.014	-0.000	0.008
BTR60	-0.004	-0.014	-0.021	1.144	0.009	0.012	0.022	-0.005	-0.001	-0.010
BTR70	-0.001	0.000	-0.012	0.003	1.145	0.005	0.004	-0.006	0.012	-0.019
D7	0.001	0.015	0.010	0.009	-0.035	1.174	-0.013	0.010	-0.011	0.020
T62	0.017	-0.001	-0.038	-0.009	0.004	-0.002	1.141	0.035	-0.008	0.005
T72	0.003	0.026	-0.036	-0.013	-0.018	0.006	0.032	1.154	-0.004	0.014
ZIL	0.007	0.002	0.005	-0.020	0.028	-0.008	-0.006	0.002	1.157	-0.015
ZSU	0.009	0.005	0.021	-0.015	-0.018	0.002	0.015	0.017	-0.024	1.154

Fig. 22 Off-diagonal weights of the final dense layer

4.2.2 Comparison to the Literature

As discussed previously, the typical MSTAR experiment in the literature [3, 8, 12, 25, 26] is to train on the data at 17° elevation and test on the data at 15° elevation. To baseline our algorithm against the algorithms in the literature, we applied our approach to the standard MSTAR experiment. The algorithm achieved an overall accuracy of 99.3% in 50 training epochs. The confusion matrix is shown in Fig. 23. The results show that the algorithm achieves nearly perfect performance



True Class \ Predicted Class	2S1	BMP2	BRDM2	BTR60	BTR70	D7	T62	T72	ZIL	ZSU		% Correct	% Incorrect
2S1	274	0	0	0	0	0	0	0	0	0		100.0%	0.0%
BMP2	0	585	0	0	0	0	0	2	0	0		99.7%	0.3%
BRDM2	0	0	272	0	0	0	0	1	0	1		99.3%	0.7%
BTR60	0	4	0	185	0	1	2	0	2	1		94.9%	5.1%
BTR70	0	0	0	0	196	0	0	0	0	0		100.0%	0.0%
D7	0	0	0	0	0	271	0	1	0	2		98.9%	1.1%
T62	0	0	0	0	0	0	273	0	0	0		100.0%	0.0%
T72	0	0	0	0	0	0	2	580	0	0		99.7%	0.3%
ZIL	0	1	0	0	0	0	1	0	270	2		98.5%	1.5%
ZSU	0	0	0	0	0	1	0	0	0	273		99.6%	0.4%

Fig. 23 Confusion matrix for our algorithm in the standard MSTAR classification setup

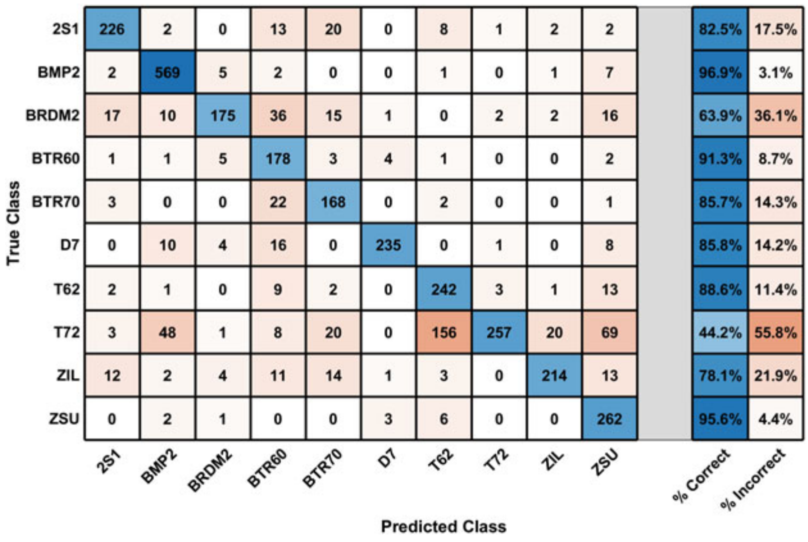
in this experiment; however, as discussed previously, this result is not indicative of algorithm performance so much as the similarity between the training and testing data.

4.2.3 Comparison to Template Matching

As the proposed algorithm is supposed to be an improvement over template matching, it is important to compare the proposed algorithm performance to that of a template-matching algorithm. To this end, the primary experiment (train on synthetic data, test on the MSTAR publicly available data) was repeated using a template-matching algorithm to perform the predictions. The template-matching algorithm used is to:

1. Process a given test chip and all template chips (keeping top N pixels and then binning the values into discrete bins).
2. Find the maximum correlation between the processed test chip and all processed template chips.
3. Report the predicted class as the class of the template that produced the best match.

Using the same training data as described above, the template-matching algorithm achieved only $\approx 79\%$ overall classification accuracy. The confusion matrix is shown in Fig. 24. Hence, the proposed algorithm has enabled an overall classification performance improvement of about 13%.



True Class \ Predicted Class	2S1	BMP2	BRDM2	BTR60	BTR70	D7	T62	T72	ZIL	ZSU	% Correct	% Incorrect
2S1	226	2	0	13	20	0	8	1	2	2	82.5%	17.5%
BMP2	2	569	5	2	0	0	1	0	1	7	96.9%	3.1%
BRDM2	17	10	175	36	15	1	0	2	2	16	63.9%	36.1%
BTR60	1	1	5	178	3	4	1	0	0	2	91.3%	8.7%
BTR70	3	0	0	22	168	0	2	0	0	1	85.7%	14.3%
D7	0	10	4	16	0	235	0	1	0	8	85.8%	14.2%
T62	2	1	0	9	2	0	242	3	1	13	88.6%	11.4%
T72	3	48	1	8	20	0	156	257	20	69	44.2%	55.8%
ZIL	12	2	4	11	14	1	3	0	214	13	78.1%	21.9%
ZSU	0	2	1	0	0	3	6	0	0	262	95.6%	4.4%

Fig. 24 Confusion matrix for the synthetic MSTAR experiment using template matching

5 Conclusion

This chapter provided a framework for end-to-end SAR ATR and discussed a suite of deep learning algorithms in that framework. We focus on approaches that use synthetic data for training to sidestep some of the issues present in deep learning approaches that use training and testing data that are very similar. In addition, our deep learning algorithms are novel as they build on the success and approaches of legacy algorithms, producing hybrid conventional/deep learning approaches. The algorithms were applied to the publicly available MSTAR dataset and demonstrated excellent performance, even when training only on the synthetically generated data. This promising suite of algorithms is a significant step forward in the state of the art for SAR ATR.

Acknowledgments This material is based upon work supported by the Air Force Research Laboratory (AFRL) under Contract No. FA8651-20-C-0046. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the AFRL.

References

1. Carrara, W.G., Goodman, R.S., Majewski, R.M.: Spotlight Synthetic Aperture Radar: Signal Processing Algorithms. Artech House, Boston (1995)
2. Jakowatz, C.V., Wahl, D.E., Eichel, P.H., Ghiglia, D.C., Thompson, P.A.: Spotlight-Mode Synthetic Aperture Radar: A Signal Processing Approach. Springer (2011). https://doi.org/10.1007/978-1-4613-1333-5

3. Ross, T., Worrell, S., Velten, V., Mossing, J., Bryant, M.: Standard SAR ATR Evaluation Experiments using the MSTAR Public Release Data Set. SPIE Conference on Algorithms for Synthetic Aperture Radar Imagery V **3370**(April 1998), 566–573 (1198). https://doi.org/10.1117/12.321859
4. Keydel, E.R., Lee, S.W., Moore, J.T.: MSTAR Extended Operating Conditions: A Tutorial. In: E.G. Zelnio, R.J. Douglass (eds.) Proceedings of SPIE Algorithms for Synthetic Aperture Radar Imagery, vol. 2757, pp. 228–242. International Society for Optics and Photonics (1996). https://doi.org/10.1117/12.242059. http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=1018994
5. Brown, W.M., Swonger, C.W.: A prospectus for automatic target recognition. IEEE Transactions on Aerospace and Electronic Systems **25**(3), 401–410 (1989). https://doi.org/10.1109/7.30795
6. Novak, L.M., Owirka, G.J., Brower, W.S.: Performance of 10- and 20-target MSE classifiers. IEEE Transactions on Aerospace and Electronic Systems **36**(4), 1279–1289 (2000). https://doi.org/10.1109/7.892675
7. Potter, L.C., Moses, R.L.: Attributed scattering centers for SAR ATR. IEEE Transactions on Image Processing **6**(1), 79–91 (1997). https://doi.org/10.1109/83.552098
8. Zhao, Q., Principe, J.C.: Support vector machines for SAR automatic target recognition. IEEE Transactions on Aerospace and Electronic Systems **37**(2), 643–654 (2001). https://doi.org/10.1109/7.937475
9. Roth, M.W.: Survey of neural network technology for automatic target recognition. IEEE Transactions on Neural Networks **1**(1), 28–43 (1990). https://doi.org/10.1109/72.80203
10. El-Darymli, K., Gill, E.W., Mcguire, P., Power, D., Moloney, C.: Automatic Target Recognition in Synthetic Aperture Radar Imagery: A State-of-the-Art Review. IEEE Access **4**, 6014–6058 (2016). https://doi.org/10.1109/ACCESS.2016.2611492
11. Kechagias-Stamatis, O., Aouf, N.: Automatic target recognition on synthetic aperture radar imagery: A survey. IEEE Aerospace and Electronic Systems Magazine **36**(3), 56–81 (2021). https://doi.org/10.1109/MAES.2021.3049857
12. Morgan, D.A.E.: Deep convolutional neural networks for ATR from SAR imagery. In: Proc. SPIE 9475, Algorithms for Synthetic Aperture Radar Imagery XXII, 94750F, vol. 9475 (2015). https://doi.org/10.1117/12.2176558
13. Chen, S., Wang, H., Xu, F., Jin, Y.: Target Classification Using the Deep Convolutional Networks for SAR Images. IEEE Transactions on Geoscience and Remote Sensing **54**(8), 4806–4817 (2016). https://doi.org/10.1109/TGRS.2016.2551720
14. Zhou F.and Wang, L., Bai, X., Hui, Y.: SAR ATR of Ground Vehicles Based on LM-BN-CNN. IEEE Transactions on Geoscience and Remote Sensing **56**(12), 7282–7293 (2018). https://doi.org/10.1109/TGRS.2018.2849967
15. Soldin, R.J., MacDonald, D.N., Reisman, M., Konz, L.R., Rouse, R., Overman, T.L.: HySARNet: a hybrid machine learning approach to synthetic aperture radar automatic target recognition. In: Proc. SPIE 10988, Automatic Target Recognition XXIX, vol. 10988 (2019). https://doi.org/10.1117/12.2518155
16. Billingsley, J.B., Farina, A., Gini, F., Greco, M.V., Verrazzani, L.: Statistical analyses of measured radar ground clutter data. IEEE Transactions on Aerospace and Electronic Systems **35**(2), 579–593 (1999). https://doi.org/10.1109/7.766939
17. Greco, M.S., Gini, F.: Statistical analysis of high-resolution SAR ground clutter data. IEEE Transactions on Geoscience and Remote Sensing **45**(3), 566–575 (2007). https://doi.org/10.1109/tgrs.2006.888141
18. Skonik, M.I.: Radar Handbook, Third Edition. McGraw Hill (2008)
19. Alexandropoulos, G.C., Berberidis, K., Sagias, N.C., Lazarakis, F.I., Datsikas, C.K., Alexandridis, A.A., Dangakis, K.P.: On the sum of squared correlated Rayleigh variates and applications to maximal-ratio diversity. In: 2007 IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications, pp. 1–5 (2007). https://doi.org/10.1109/PIMRC.2007.4394161

20. Joarder, A.H., Omar, M.H., Gupta, A.K.: The distribution of a linear combination of two correlated chi-squared variables. Rev. Colombiana Estadíst **36**(2), 209–219 (2013)
21. Ferrari, A.: A note on sum and difference of correlated chi-squared variables (2019). https://doi.org/10.48550/arXiv.1906.09982
22. El-Darymli, K., McGuire, P., Power, D., Moloney, C.R.: Target detection in synthetic aperture radar imagery: a state-of-the-art survey. J. of Applied Remote Sensing **7**, 71535–71598 (2013). https://doi.org/10.1117/1.JRS.7.071598
23. Zeiler, M.D.: ADADELTA: An Adaptive Learning Rate Method. arXiv p. 6 (2012). http://arxiv.org/abs/1212.5701
24. Beylkin, G.: Discrete radon transform. Acoustics, Speech and Signal Processing, IEEE Transactions on **35**(2), 162–172 (1987). https://doi.org/10.1109/TASSP.1987.1165108
25. Chen, S., Lu, F., Wang, J., Liu, M.: Target aspect angle estimation for synthetic aperture radar automatic target recognition using sparse representation. In: 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), pp. 1–4 (2016). https://doi.org/10.1109/ICSPCC.2016.7753656
26. Wilmanski, M., Kreucher, C., Lauer, J.: Modern approaches in deep learning for SAR ATR. In: Proceedings of the 2016 SPIE Defense, Security, and Sensing Symposium, vol. 9843, pp. 98430N–98430N–10 (2016). http://dx.doi.org/10.1117/12.2220290