Deep Convolutional Neural Networks with Integrated Quadratic Correlation Filters for Automatic Target Recognition

Brian Millikan, Hassan Foroosh
Department of Electrical Engineering and Computer Science
University of Central Florida
{brian.millikan@knights, foroosh@cs}.ucf.edu

Qiyu Sun
Department of Mathematics
University of Central Florida
qiyu.sun@ucf.edu

Abstract

Automatic target recognition involves detecting and recognizing potential targets automatically, which is widely used in civilian and military applications today. Quadratic correlation filters were introduced as two-class recognition classifiers for quickly detecting targets in cluttered scene environments. In this paper, we introduce two methods that integrate the discrimination capability of quadratic correlation filters with the multi-class recognition ability of multi-layer neural networks. For mid-wave infrared imagery, the proposed methods are demonstrated to be multi-class target recognition classifiers with very high accuracy.

1. Introduction

Automatic target recognition (ATR) involves detecting and recognizing potential targets automatically in digitized video data [22]. ATR was first introduced in the 1980s with aircraft-mounted targeting pods for military aircraft [21, 8]. Numerous ATR algorithms have been proposed during the last few decades [22, 21, 3, 2, 1], and it is still a crucial element to many military applications today [22, 8]. An ATR system can be divided into four parts [3, 9]: detection, clutter rejection, feature extraction and classification. In this paper, we consider the latter two items together by introducing a quadratic correlation filter (QCF) coupled with a convolutional neural network (CNN).

QCFs were introduced by Mahalanobis *el al.* in [13] to perform target detection. The original QCF is based on the Fukunaga-Koontz Transform (FKT), which keeps the eigenvectors corresponding to the largest and smallest eigenvalues. A single-layer perceptron QCF was introduced in [14] and shown to be effective for two-class target detection. In many military mid-wave infrared (MWIR) applications, multi-class target recognition is required. This suggests the modification of the two-class target detection methods to support multi-class target recognition.

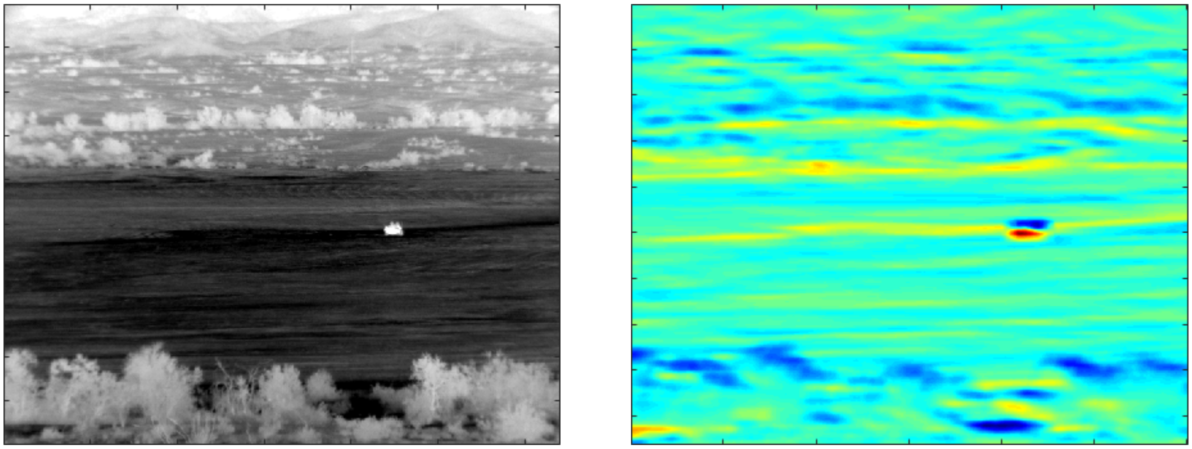
CNNs were introduced by LeCun *et al* [10, 11] to build networks that are invariant to certain transformations of the inputs. Rather than performing the feature extraction manually, it is built into the network and learned through the training process [4]. A typical CNN has correlation and pooling layers followed by fully-connected layers for performing the feature extraction and classification. Neural network approaches are broadly categorized as learning-based ATR [22]. In [9], one learning-based ATR approach uses K-means to cluster dense Histogram of Gradient (HOG) features for infrared (IR) targets, where similar visual words are grouped together in a Bag-of-Words (BoW) classification style. Another proposed method is the maximum margin correlation filter (MMCF), which uses a correlation filter coupled with a support vector machine (SVM) to classify targets [19, 20].

In this paper, we introduce a couple of learning-based ATR approaches by extending the two-class target detection method in [14] to a quadratic multi-layer perceptron and integrating a quadratic correlation filter (QCF) into a CNN for target recognition. We demonstrate the performance of our methods using the *ATR Algorithm Image Database* [23] mid-wave infrared (MWIR) dataset.

2. Quadratic Correlation Filters

The quadratic correlation filter (QCF) was developed in [13] to efficiently detect targets in a cluttered scene. In their paper, a couple of methods are proposed to find a QCF $\mathbf{H}$ to perform target detection. The filter $\mathbf{H} = \mathbf{F}\mathbf{F}^T - \mathbf{G}\mathbf{G}^T$ can be a composite matrix consisting of a filter $\mathbf{F}$ to detect targets and a filter $\mathbf{G}$ to detect the background [13, 14]. It can be applied efficiently to evaluate a scene at every location s in the image. If a region of the image $\mathbf{x}$ at location s is rearranged in vectorized form denoted as $\mathbf{x}_s$, then the application of the QCF $\mathbf{H}$ is $\varphi(\mathbf{x}_s) = \mathbf{x}_s^T \mathbf{H} \mathbf{x}_s$, which can be performed efficiently using a 2D cross-correlation:

$$\varphi(\mathbf{x}) = \sum_i |\mathbf{x} \otimes \mathbf{f}_i|^2 - \sum_j |\mathbf{x} \otimes \mathbf{g}_j|^2 \qquad (1)$$



(a)

(b)

Figure 1: Target detection using a quadratic correlation filter (QCF) learned from a single-layer perceptron neural network (SLPNN): (a) Original image with an armoured personnel carrier (BMP2), (b) correlation output. We observe that the BMP2 can be detected, but not recognized or identified, using this method.

where i is the index of column vectors $\mathbf{f}_i$ of $\mathbf{F}$ and j index of column vectors $\mathbf{g}_j$ of $\mathbf{G}$.

A design of QCFs $\mathbf{H}$ proposed in [14] minimizes the sum of the squared error between the output statistic φ and the true label d for all training samples. It is based on the back propagation training algorithm for kernel neurons to generate the filter [25]. Denote the statistic of an image patch $\mathbf{x}_n$ by

$$\varphi(\mathbf{x}_n) = \mathbf{x}_n^T \mathbf{H} \mathbf{x}_n, 1 \leq n \leq N_e,$$

where $N_e = 2N_S$ is the total number of training images. We expect that $\mathbf{x}_n$ is a target when $\varphi(\mathbf{x}_n)$ is positive and large, it is the background when $\varphi(\mathbf{x}_n)$ is negative and small.

Denote the output of the single-layer perception $d_n \in \{-1, 1\}$ by 1 when the training image is a target and by -1 when it is the background. To meet the expectation, we use a differentiable squashing function, $\sigma(\varphi(\mathbf{x}_n)) = \tanh(\varphi(\mathbf{x}_n))$, to force the output of $\varphi(\mathbf{x}_n)$ to be between -1 and $+1$, and minimize the objective function

$$J(\mathbf{H}) = \frac{1}{2}\sum_{n=1}^{N_e} |d_n - \sigma(\varphi(\mathbf{x}_n))|^2 = \frac{1}{2}\sum_{n=1}^{N_e} |d_n - \sigma(\mathbf{x}_n^T \mathbf{H} \mathbf{x}_n)|^2. \tag{2}$$

We can iteratively solve for $\mathbf{H}$ using the gradient descent algorithm

$$\mathbf{H}_{t+1} = \mathbf{H}_t - \eta \nabla J(\mathbf{H}_t), t \geq 0, \tag{3}$$

and $\mathbf{H}_0 = 0$, where the gradient $\nabla J(\mathbf{H})$ is given by

$$\nabla J(\mathbf{H}) = -\sum_{n=1}^{N_e} |d_n - \sigma(\varphi(\mathbf{x}_n))| \left(1 - \sigma^2(\varphi(\mathbf{x}_n))\right) \mathbf{x}_n \mathbf{x}_n^T. \tag{4}$$

We denote the solution to (3) as $\mathbf{H}^*$ which occurs after the convergence criterion has been met or T iterations have occurred. Given a test image patch $\mathbf{x}$, we can classify it as a target or the background by examining $\sigma(\varphi(\mathbf{x}))$ which will be between 0 and 1 for a target and between -1 and 0 for background, see Figure 1.

3. Proposed Methods

In Section 2, we recalled a single-layer perceptron QCF for two-class target detection. However, target recognition requires a multi-class discriminator. Therefore, in this section we propose a multi-layer perceptron neural network with a quadratic filter input layer for multi-class target recognition. Since multi-layer perceptron neural networks are fully-connected networks, the input size must match the training input patch size. To remedy this, we propose an all-convolutional CNN with a QCF layer for multi-class target recognition and take advantage of the invariance properties of the CNN [4]. In addition, an all-convolutional CNN offers the ability to perform pixel-wise image classification using only image patches for training [17]. This makes this type of CNN ideal for ATR applications.

3.1. Quadratic Multi-layer Perceptron Neural Network for Target Recognition

In this subsection, we introduce a quadratic multi-layer perceptron neural network (QMLPNN) for multi-class target recognition, see Figure 2. Given an input patch $\mathbf{x}_n, 1 \leq n \leq N$ of size m, the output of the first hidden layer is a quadratic function, which is learned during the network

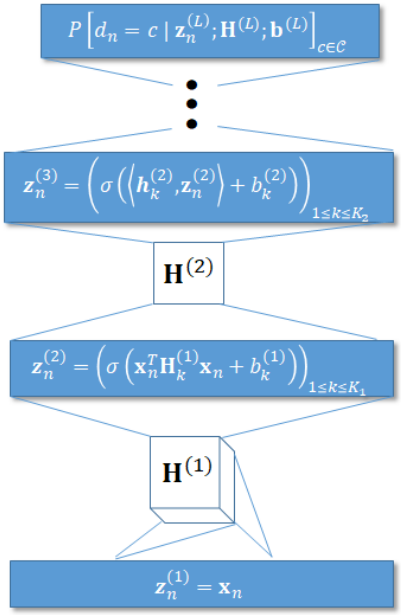

$P\left[d_n = c \mid \mathbf{z}_n^{(L)}; \mathbf{H}^{(L)}; \mathbf{b}^{(L)}\right]_{c\in\mathcal{C}}$
$\mathbf{z}_n^{(3)} = \left(\sigma\left(\left\langle \mathbf{h}_k^{(2)}, \mathbf{z}_n^{(2)}\right\rangle + b_k^{(2)}\right)\right)_{1\leq k\leq K_2}$
$\mathbf{H}^{(2)}$
$\mathbf{z}_n^{(2)} = \left(\sigma\left(\mathbf{x}_n^T\mathbf{H}_k^{(1)}\mathbf{x}_n + b_k^{(1)}\right)\right)_{1\leq k\leq K_1}$
$\mathbf{H}^{(1)}$
$\mathbf{z}_n^{(1)} = \mathbf{x}_n$

Figure 2: Quadratic multi-layer perceptron neural network (QMLPNN) for target recognition. The first layer is composed of a quadratic filter which requires a 3D weight tensor. The rest of the network forms a standard MLPNN.

training, of the form

$$\mathbf{z}_n^{(2)} = \left(\sigma(\mathbf{x}_n^T\mathbf{H}_k^{(1)}\mathbf{x}_n + b_k^{(1)})\right)_{1\leq k\leq K_1}, \tag{5}$$

where K_1 is the number of hidden nodes in the first layer, $\mathbf{H}^{(1)} = \left(\mathbf{H}_k^{(1)}\right)_{1\leq k\leq K_1}$ is a three-dimensional weight tensor of dimension $m \times m \times K_1$, $\mathbf{b}^{(1)} = \left(b_k^{(1)}\right)_{1\leq k\leq K_1}$ forms a bias vector and σ is a non-linear activation function. The other hidden layers are composed of conventional activation functions,

$$\mathbf{z}_n^{(\ell+1)} = \left(\sigma(\langle\mathbf{h}_k^{(\ell)}, \mathbf{z}_n^{(\ell)}\rangle + b_k^{(\ell)})\right)_{1\leq k\leq K_\ell}, 2 \leq \ell \leq L-1, \tag{6}$$

where K_ℓ is the number of hidden nodes, $\mathbf{H}^{(\ell)} = \left(\mathbf{h}_k^{(\ell)}\right)_{1\leq k\leq K_\ell}$ forms the weight matrix of dimension $K_\ell \times K_{\ell-1}$, $\mathbf{b}^\ell = \left(b_k^{(\ell)}\right)_{1\leq k\leq K_\ell}$ forms the bias vector for the ℓ-th layer, and L is the total number of layers [4, 15].

We denote the set of all target class types by $\mathscr{C}$ and label each input pattern by $d_n \in \mathscr{C}, 1 \leq n \leq N$. We note that the last layer has K_L hidden nodes which is equal to the number of classes in $\mathscr{C}$. Given this, we write $\mathbf{H}^{(L)} = \left(\mathbf{h}_c^{(L)}\right)_{c\in\mathscr{C}}$ for convenience. The objective function $E_n(\mathbf{H}, \mathbf{b})$ for the input patch $\mathbf{x}_n$ is defined as

$$E_n(\mathbf{H}, \mathbf{b}) = -\sum_{c\in\mathscr{C}} [d_n = c] \log\left(\frac{\exp\left(\langle\mathbf{h}_c^{(L)}, \mathbf{z}_n^{(L)}\rangle + b_c^{(L)}\right)}{\sum_{j\in\mathscr{C}} \exp\left(\langle\mathbf{h}_j^{(L)}, \mathbf{z}_n^{(L)}\rangle + b_j^{(L)}\right)}\right) \tag{7}$$

where $[i = j]$ is the Iverson bracket notation for the Kronecker delta function, $\mathbf{H} = \{\mathbf{H}^{(1)}, \mathbf{H}^{(2)}, ..., \mathbf{H}^{(L)}\}$ and $\mathbf{b} = \{\mathbf{b}^{(1)}, \mathbf{b}^{(2)}, ..., \mathbf{b}^{(L)}\}$.

Training is accomplished using gradient descent with gradients determined by the backpropagation algorithm. All layers will result in a weight matrix with the exception of the first layer, which is a three-dimensional tensor. The error term is

$$\delta^{(L)} = -\left([d_n = c] - P\left[d_n = c|\mathbf{z}_n^{(L)}; \mathbf{H}^{(L)}, \mathbf{b}^{(L)}\right]\right)_{c\in\mathscr{C}} \tag{8}$$

for the last layer, and

$$\delta^{(\ell)} = \left(\mathbf{H}^{(\ell)}\right)^T \delta^{(\ell+1)} \circ \left(\sigma'(\langle\mathbf{h}_k^{(\ell)}, \mathbf{z}_n^{(\ell)}\rangle + b_k^{(\ell)})\right)_{1\leq k\leq K_\ell} \tag{9}$$

for internal layers $2 \leq \ell \leq L-1$, where we use the notation $\circ$ to represent the element-wise Hadamard product [15]. The internal layers in the network have standard fully-connected nodes. As in [4, 15], the gradients associated with error term $\delta^{(\ell+1)}$ is

$$\nabla_{\mathbf{H}^{(\ell)}} E_n(\mathbf{H}, \mathbf{b}) = \delta^{(\ell+1)}(\mathbf{z}_n^{(\ell)})^T. \tag{10}$$

However, the framework in the first layer (5) is specifically designed for multi-target recognition networks. The gradient associated with the objective function $E_n(\mathbf{H}, \mathbf{b})$, is defined by

$$\nabla_{\mathbf{H}_k^{(1)}} E_n(\mathbf{H}, \mathbf{b}) = \delta_k^{(2)}\mathbf{x}_n\mathbf{x}_n^T, \quad 1 \leq k \leq K_1. \tag{11}$$

3.2. Quadratic Correlation Filter Convolutional Neural Network for Target Recognition

In this section, we introduce a quadratic correlation filter convolutional neural network (QCFCNN) for automatic target recognition. The weight matrix $\mathbf{H}^{(1)} = \left(\mathbf{H}_k^{(1)}\right)_{1\leq k\leq K_1}$ for the first layer, illustrated in Figure 3, contains multiple QCFs $\mathbf{H}^{(1)}$ of size $M_1 \times M_1$ that are applied to the input image. To apply the weight matrix in the first layer, we define

$$\mathbf{\Theta}_k^{(1)} = \mathbf{H}_k^{(1)} + \left(\mathbf{H}_k^{(1)}\right)^T, \tag{12}$$

and perform its eigendecomposition

$$\mathbf{\Theta}_k^{(1)}\mathbf{V}_k = \mathbf{V}_k\mathbf{\Lambda}_k \tag{13}$$

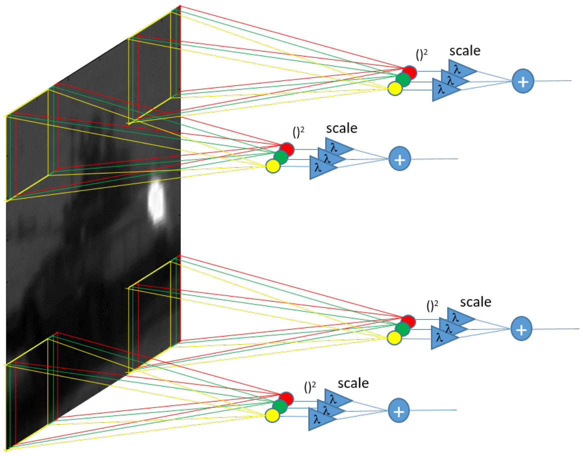


Figure 3: First layer of a quadratic correlation filter convolutional neural network (QCFCNN) for target recognition. The input is correlated with a learned weight eigenvector, squared, multiplied by the eigenvalue and summed together. The rest of the QCFCNN is a standard fully convolutional neural network.

where the diagonal matrix $\mathbf{\Lambda}_k$ has real eigenvalues $\lambda_{k,M_1} \geq \lambda_{k,M_1-1} \geq \cdots \geq \lambda_{k,1}$ as its diagonal entries and orthonormal matrix $\mathbf{V}_k = [\mathbf{v}_{k,1}, \cdots, \mathbf{v}_{k,M_1}]$ contains the corresponding eigenvectors. The first layer is

$$\mathbf{z}_n^{(2)} = \Big(\sigma\big(\sum_{i=1}^{M_1} \lambda_{k,i} |\mathbf{x}_n \otimes \mathbf{v}_{k,i}|^2 + b_k^{(1)}\big)\Big)_{1\leq k\leq K_1} \qquad (14)$$

for every location in the image $\mathbf{x}$ of height M and width K, where the symbol $\otimes$ indicates the 2D cross-correlation operation. The internal layers are standard convolutional layers,

$$\mathbf{z}_n^{(\ell+1)} = \Big(\sigma\big(\sum_{i=1}^{K_{\ell-1}} \mathbf{h}_{k,i}^{(\ell)} \otimes \mathbf{z}_{n,i}^{(\ell)} + b_k^{(\ell)}\big)\Big)_{1\leq k\leq K_\ell}, 2 \leq \ell \leq L-1, \qquad (15)$$

where L is the total number of layers in the CNN.

Training is accomplished by standard backpropagation for convolutional neural networks. The standard error function for CNN network internal layers is

$$\begin{aligned}\delta_k^{(\ell)} = \text{upsample}\Big(&\sum_{i=1}^{K_{\ell+1}} \mathbf{h}_{k,i}^{(\ell+1)} \otimes_f \delta_k^{(\ell+1)}\Big) \\ &\circ\sigma'\big(\sum_{i=1}^{K_{\ell-1}} \mathbf{h}_{k,i}^{(\ell)} \otimes \mathbf{z}_{n,i}^{(\ell)} + b_k^{(\ell)}\big) \qquad (16)\end{aligned}$$

which can be used to define the gradient, where $2 \leq \ell \leq L-1$, $1 \leq k \leq K_\ell$ and the notation $\otimes_f$ denotes a full 2D cross-correlation operation. However, for the gradient definition, we must first define the vectorization of a $d \times d$ subimage $\mathbf{X}_{l,m}^{(n)}$,

$$\begin{aligned}\text{vec}\left(\mathbf{X}_{l,m}^{(n)}\right) = [&x_{l,m}^{(n)}, \cdots, x_{l,m+d-1}^{(n)}, x_{l+1,m}^{(n)}, \cdots, \\ &x_{l+1,m+d-1}^{(n)}, \cdots, x_{l+d-1,m}^{(n)}, \cdots, x_{l+d-1,m+d-1}^{(n)}]^T. \qquad (17)\end{aligned}$$

Finally, we can evaluate the gradient for the first layer by

$$\begin{aligned}\nabla_{\mathbf{H}_k^{(1)}} E_n(\mathbf{H}, \mathbf{b}) = \frac{\partial E_n}{\partial \mathbf{H}_k^{(1)}} &= \frac{\partial E_n}{\partial \mathbf{\Theta}_k^{(1)}} \frac{\partial \mathbf{\Theta}_k^{(1)}}{\partial \mathbf{H}_k^{(1)}} \\ &= \sum_{l,m} \delta_k^{(2)} \text{vec}\left(\mathbf{X}_{l,m}^{(n)}\right) \text{vec}\left(\mathbf{X}_{l,m}^{(n)}\right)^T, \qquad (18)\end{aligned}$$

where $1 \leq k \leq K_1$.

4. Numerical Experiments

We conducted our simulations using the SENSIAC *ATR Algorithm Image Database* [23]. This database is a mid-wave infrared (MWIR) dataset from the U.S. Army Night Vision and Electronic Sensors Directorate (NVESD). It contains 207GB of MWIR data which includes 10 vehicle target types and 2 scenarios of humans. As shown in Figure 4, we considered all 10 vehicle target types, which include a Pickup Truck (PICKUP), Sport Utility Vehicle (SUV), Armored Personnel Carriers (BTR70 and BMP2), an Infantry Scout Vehicle (BRDM2), a Main Battle Tank (T72), an Anti-Aircraft Weapon (ZSU23-4), a Self-Propelled Howitzer (2S3), an Armoured Reconnaissance Vehicle (MTLB), and a Towed Howitzer (D20). A 40 × 80 bounding box is formed around each target using the ground truth data to generate a target image patch, $\mathbf{x}$, for training.

In our simulations, we select images that contain targets less than 3000 meters in range from the camera at any time of day. We train a single neural network (NN) using a different class of targets within each range of aspect angles. For instance, we may divide up a 360°aspect angle range into 4 separate classes for each target category. For a 10 target set, this will yield a total of 40 classes. All of these factors make this a difficult dataset to classify. In our simulations, a misclassification is only counted if a target is classified outside of its target category (i.e. aspect angle is ignored). We train and test the networks using image patches with the target centered in the patch. For the QCFCNN, we also train with example background image patches to segment target areas from background using the embedded QCF. Since this network is built using all convolutional layers, we can train with image patches and subsequently use it to perform classification on larger images giving a weakly-supervised network, see [17].

In our simulations, we apply the QCFCNN in Section 3.2 for target recognition. Shown in Figure 5 are some exam-

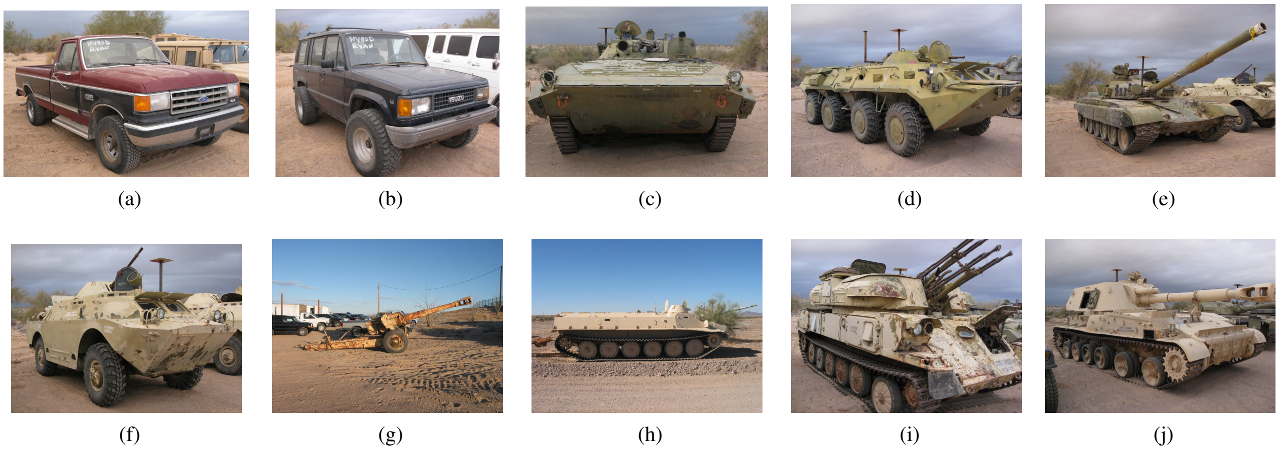


Figure 4: Target images in the visible spectrum: (a) civilian pickup truck (PICKUP), (b) civilian sport utility vehicle (SUV), (c) armoured personnel carrier (BMP2), (d) armoured personnel carrier (BTR70), (e) main battle tank (T72), (f) infantry scout vehicle (BRDM2), (g) towed howitzer (D20), (h) armoured reconnaissance vehicle (MTLB), (i) anti-aircraft weapon (ZSU23-4), and (j) self-propelled howitzer (2S3). We observe that many of the vehicles are very similar in appearance which makes the target recognition task more difficult.

ples of correctly classified targets containing labeled images and class label maps created from the output of the QCFCNN. These examples are created from targets that are less than 1500 meters from the camera. It is assumed that target detection has been accomplished and that a window has been created surrounding the detection. The target class is determined by taking the mode of a window around the centroid of the connected components in the QCFCNN output. Some misclassification can occur for the QCFCNN, see Figure 6 for some examples of misclassification. We observe that most of the misclassifications occur when there is similarity between the target types.

In our simulations, we train the QMLPNN with the quasi-Newton limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) optimization algorithm [16, 12], and the QCFCNN with stochastic gradient descent and an adaptive gradient algorithm (AdaGrad) [7, 18]. Their performance for target recognition could be evaluated by a confusion matrix, see Figure 7. As seen from their bottom right corners, there is a low probability to misclassify MTLB and D20 targets, (cf. Figure 6), while there is a very high accuracy to recognize other targets.

For our simulations, the MLPNN has an input layer, two hidden layers and an output layer. The QMLPNN is identical to the MLPNN, but with the first layer replaced with quadratic function. We used a nine layer CNN with convolution, ReLU and pooling layers followed by a softmax layer. The QCFCNN is identical to the CNN, but with the first convolution layer replaced with a QCF.

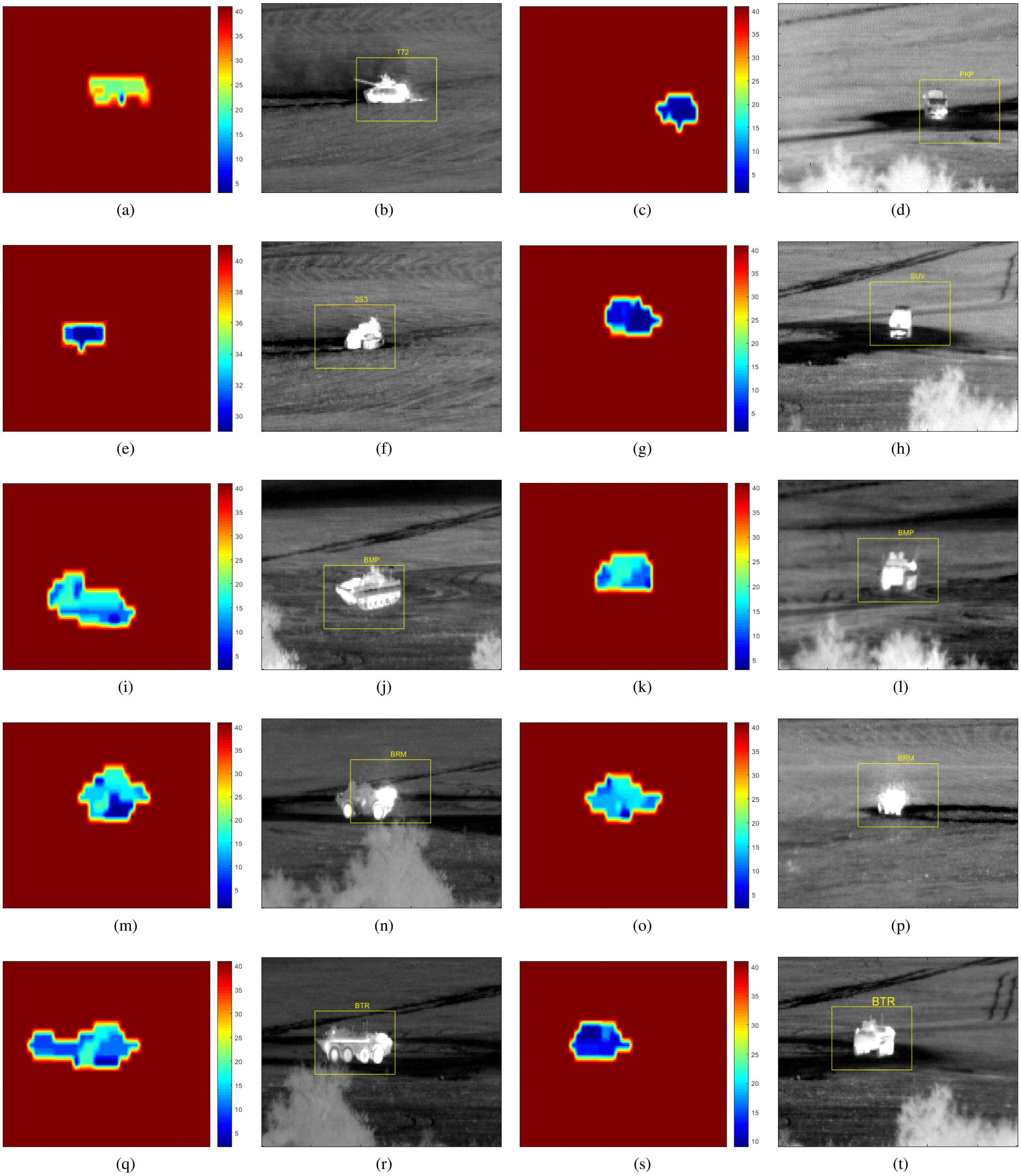
Table 1: Comparison of target recognition classifiers. The QMLPNN with quadratic layer has 7.3% increase in accuracy over the basic MLPNN. Similarly, the QCFCNN with quadratic correlation filter layer has an increase in accuracy of 2.32%. The QMLPNN has a 39.46% increase in accuracy and the QCFCNN has a 40.03% increase in accuracy over the baseline linear SVM (without retraining).

Method	Training	Iter/Tol	Accuracy
QMLPNN	L-BFGS	1000 iter	0.9813
MLPNN [24]	L-BFGS	1000 iter	0.9083
QCFCNN	AdaGrad	4680 iter	0.9870
CNN [4]	AdaGrad	4680 iter	0.9638
SVM [6]	C-SVC	0.001 tol	0.5867

We conducted experiments to compare our methods for target recognition with the standard convolutional neural network (CNN) [4], a conventional MLPNN [24] and a linear support vector machine (SVM) [6]. Table 1 shows the results from our simulations, where we use 1000 iterations for QMLPNN and MLPNN, 4680 iterations (or 20 epochs) for QCFCNN and CNN, and a tolerance of 0.001 in the libsvm MATLAB software package for the linear SVM [5]. From Table 1, we learn that substitution of a quadratic layer causes an increase in accuracy from 90.8% for the standard MLPNN to 98.1% for the QMLPNN, and a jump in accuracy from 96.38% for the standard CNN to 98.7% for the QCFCNN.

5. Conclusions

The single-layer perceptron QCF has satisfactory performance for two-target discrimination as demonstrated



(a) (b) (c) (d)

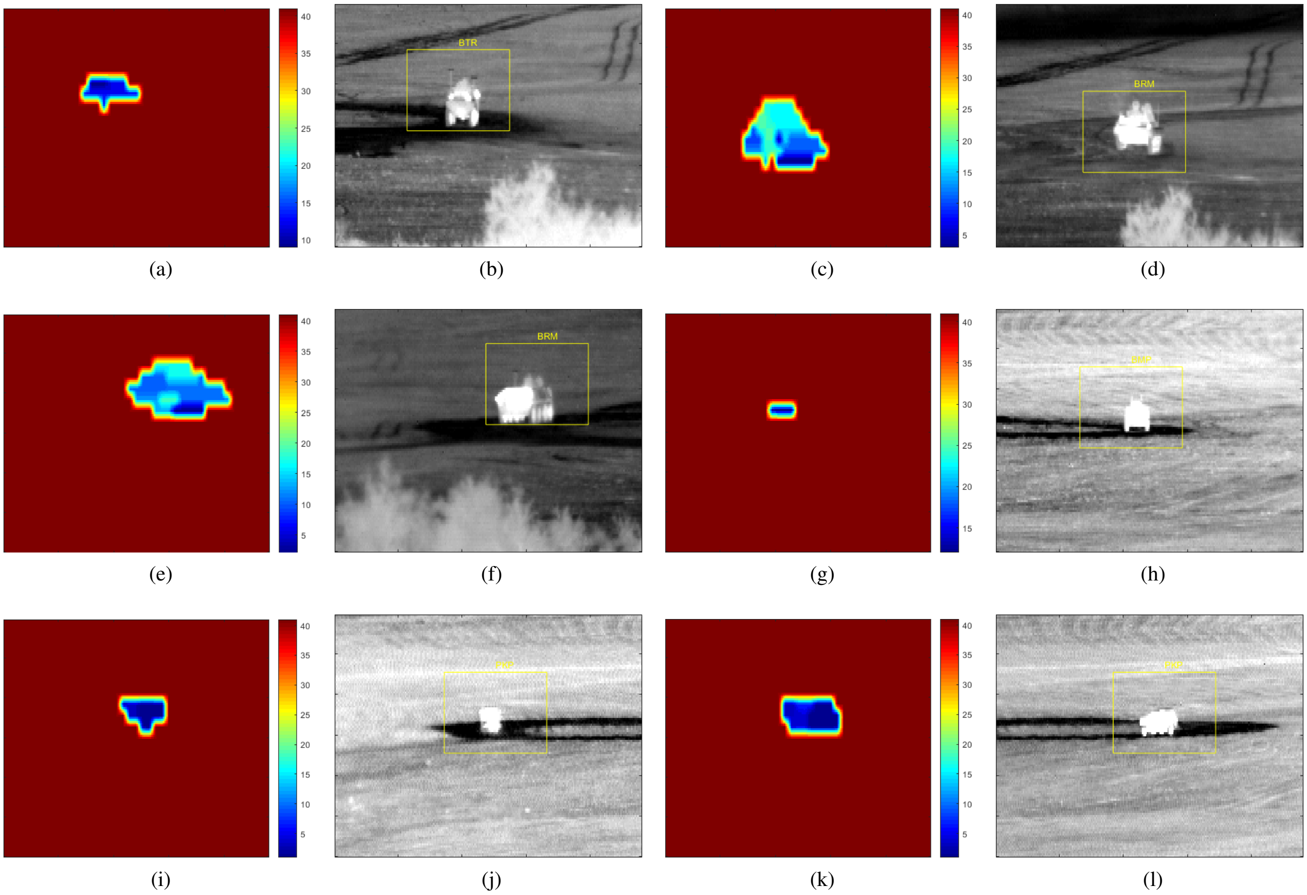
(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

(q) (r) (s) (t)

Figure 5: Some correctly classified examples: QCFCNN output class label map and labeled image for (a)-(b) main battle tank (T72); (c)-(d) pickup truck (PKP); (e)-(f) self-propelled howitzer (2S3); (g)-(h) sport utility vehicle (SUV); (i)-(l) armoured personnel carrier (BMP2); (m)-(p) infantry scout vehicle (BRDM2); (q)-(t) armoured personnel carrier (BTR70).

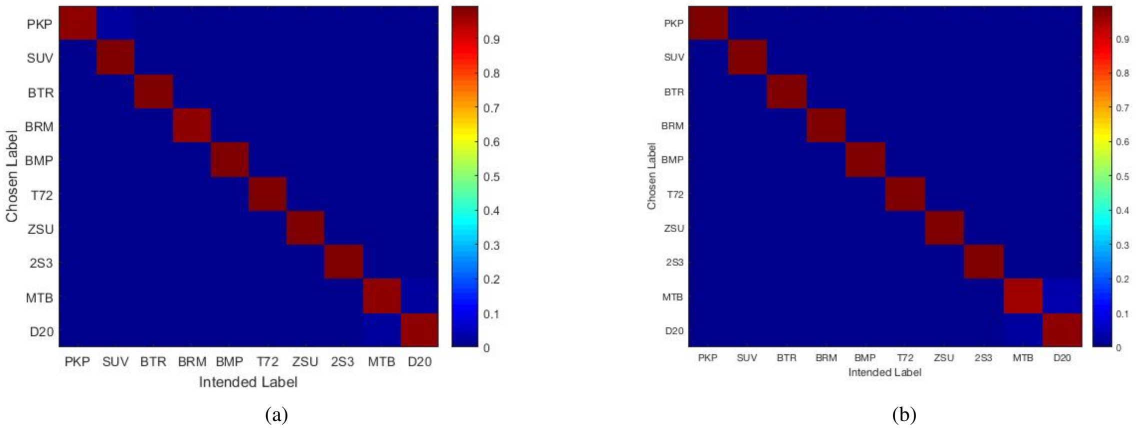


(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 6: Some mislabeled Examples: QCFCNN output class label map and labeled image for (a)-(b) BRDM2 labeled as BTR70; (c)-(d) BMP2 labeled as BRDM2; (e)-(f) BTR70 labeled as BRDM2; (g)-(h) BTR70 labeled as BMP2; (i)-(j) SUV labeled as PKP; Most misclassifications are due to the similarity between target classes.



(a) (b)

Figure 7: (a) Quadratic multi-layer perceptron neural network (QMLPNN) and (b) Quadratic correlation filter convolutional neural network (QCFCNN) confusion matrices for target recognition with the *ATR Algorithm Image Database* [23] dataset. As can be seen in both matrices, the target recognition accuracy is very high. However, in both matrices, there is some probability of misclassification of the D20 and MTB (MTLB) target types as seen in the lower right corner.

in [14]. The proposed QCFCNN and QMLPNN, that integrate QCF with multi-layer neural networks, have very high multi-target recognition rate for mid-wave infrared images.

References

[1] T. M. Bayik. Automatic target recognition in infrared imagery. Master's thesis, Middle East Technical University Press, Ankara, 2004.

[2] B. Bhanu. Automatic target recognition: State of the art survey. *IEEE Transactions on Aerospace and Electronic Systems*, AES-22(4):364–379, July 1986.

[3] B. Bhanu and T. L. Jones. Image understanding research for automatic target recognition. *IEEE Aerospace and Electronic Systems Magazine*, 8(10):15–23, October 1993.

[4] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[5] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011.

[6] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[7] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[8] D. E. Dudgeon and R. T. Lacoss. An overview of automatic target recognition. *The Lincoln Laboratory Journal*, 6(1):3–10, 1993.

[9] M. N. A. Khan, G. Fan, D. R. Heisterkamp, and L. Yu. Automatic target recognition in infrared imagery using dense hog features and relevance grouping of vocabulary. In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 293–298, June 2014.

[10] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[12] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

[13] A. Mahalanobis, R. R. Muise, S. R. Stanfill, and A. V. Nevel. Design and application of quadratic correlation filters for target detection. *IEEE Transactions on Aerospace and Electronic Systems*, 40(3):837–850, July 2004.

[14] B. Millikan, A. Dutta, Q. Sun, and H. Foroosh. Fast detection of compressively sensed ir targets using stochastically trained least squares and compressed quadratic correlation filters. *IEEE Transactions on Aerospace and Electronic Systems*, 53(5):2449–2461, October 2017.

[15] A. Ng et al. Deep learning tutorial. *URL http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders*, 2015.

[16] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[17] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Is object localization for free? - weakly-supervised learning with convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 685–694, June 2015.

[18] J. Perla. Notes on adagrad.

[19] A. Rodriguez, V. N. Boddeti, B. V. Kumar, and A. Mahalanobis. Maximum margin correlation filter: A new approach for localization and classification. *IEEE Transactions on Image Processing*, 22(2):631–643, February 2013.

[20] A. F. Rodriguez-Perez. *Maximum Margin Correlation Filters*. PhD thesis, Carnegie Mellon University, 2012.

[21] M. W. Roth. Survey of neural network technology for automatic target recognition. *IEEE Transactions on Neural Networks*, 1(1):28–43, March 1990.

[22] B. J. Schachter. *Automatic Target Recognition*. International Society for Optics and Photonics (SPIE), March 2016.

[23] SENSIAC. ATR Algorithm Development Image Database. `http://www.sensiac.org/external/products/list_databases.jsf`, 2011.

[24] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, Boston, fourth edition edition, 2009.

[25] J. Xu, X. Zhang, and Y. Li. Kernel neuron and its training algorithm. In *Proceedings of 8th International Conference on Neural Information Processing*, volume 2, pages 861–866, 2001.