Deep Belief Networks for False Alarm Rejection in Forward-Looking Ground-Penetrating Radar

John Becker[a], Timothy C. Havens[a,b], Anthony Pinar[a], and Timothy J. Schulz[a]

[a]Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI USA

[b]Department of Computer Science, Michigan Technological University, Houghton, MI USA

ABSTRACT

Explosive hazards are one of the most deadly threats in modern conflicts. The U.S. Army is interested in a reliable way to detect these hazards at range. A promising way of accomplishing this task is using a *forward-looking ground-penetrating radar* (FLGPR) system. Recently, the Army has been testing a system that utilizes both L-band and X-band radar arrays on a vehicle mounted platform. Using data from this system, we sought to improve the performance of a *constant false-alarm-rate* (CFAR) prescreener through the use of a *deep belief network* (DBN). DBNs have also been shown to perform exceptionally well at generalized anomaly detection. They combine unsupervised pre-training with supervised fine-tuning to generate low-dimensional representations of high-dimensional input data. We seek to take advantage of these two properties by training a DBN on the features of the CFAR prescreener's *false alarms* (FAs) and then use that DBN to separate FAs from true positives. Our analysis shows that this method improves the detection statistics significantly. By training the DBN on a combination of image features, we were able to significantly increase the probability of detection while maintaining a nominal number of false alarms per square meter. Our research shows that DBNs are a good candidate for improving detection rates in FLGPR systems.

Keywords: e xplosive hazards, ground penetrating radar, deep belief networks, unsupervised learning, signal processing

1. INTRODUCTION

Buried explosive hazards represent one of the greatest threats to human life in modern combat as well as a danger to civilian populations residing in former war zones. Every month, explosive devices kill an average of 310 people and wound 833 others.[1] Several systems have been investigated as a means of detecting these hazards. Among them are *ground-penetrating-radar* (GPR), *infrared* (IR) and visible-spectrum cameras, and acoustic technologies.[2–4] A particularly attractive system that has been researched and has shown good progress is the *forward-looking* GPR (FLGPR). These have been applied to not only buried threat detection, but also to side attack and surface threat detection as well.[5–7] The feature that makes FLGPR a particularly attractive system is the increased stand-off distance, or the distance away from the platform at which the threat is detected. This increased stand-off distance does not come for free though. In addition to explosive hazards, FLGPRs are also sensitive to other objects, both buried and on the surface. It has been shown that detection performance can be improved in FLGPRs by fusing multiple sub-bands and spectral features.[8–10]

Recently, the U.S. Army has been investigating an FLGPR system that integrates both L- and X-band radar arrays. Our focus was the development of computer-aided classification methods for this system. We have investigated a deep learning method called a *deep belief network* (DBN), in addition to more common methods such as *support vector machines* (SVMs) and *multiple kernel learning* (MKL).[11] Deep learning algorithms are a relatively new approach to the explosive hazard detection problem,[12,13] which have already shown great results. In this work, we seek to expand the understanding of what DBNs can achieve when applied to the forward-looking explosive hazards detection problem.

2. RADAR SYSTEM AND IMAGING

A dual-band, stepped-frequency radar was used for this experiment. The systems consists of a *multiple-input, multiple-output* (MIMO) L-Band radar array and an X-band radar array. The L-band radar collects HH and VV *transmit/recieve* (T/R) polarizations when operating in the *dual-polarization* (dualpol) mode. The L-band may also operate in an *all-polarization* (allpol) mode, in which it collects HH, VV, VH, and HV Tx/Rx polarizations. The X-band may only collect VV Tx/Rx polarization. The specific parameters of the FLGPR system are shown in Table 2.

Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XX,
edited by Steven S. Bishop, Jason C. Isaacs, Proc. of SPIE Vol. 9454, 94540W
 · doi: 10.1117/12.2176855

Table 1. Symbols Used in This Paper

Symbol	Description
$\mathbf{x}(t_{GPS})$	Position of vehicle in UTM at time t_{GPS}
$\mathbf{v}(t_{GPS})$	Velocity (m/s) of vehicle at time t_{GPS}
$\mathbf{x}_j(t_{GPS})$	Position of jth antenna element at time t_{GPS}
$\mathbf{w}_{jk}(f, t_{GPS})$	I/Q signal of jkth T/R pair at time t_{GPS} and frequency f
$a_w(f)$	frequency-domain window (Hamming)
$a_r(j,k)$	aperture window
c	speed of light, 2.998×10^8 (m/s)

Table 2. FLGPR Parameters[11]

	L-band	X-band
Waveform	Stepped frequency	Stepped frequency
Transmitters	8	32
Receivers	8	4
Bandwidth	0.5–3.4 GHz	8.4–10.4 GHz
# Frequencies	2702	1024
Sweep rate	12 Hz	50 Hz
Polarizations*	HH, VV, HV, VH	VV

*Note that we only have HH and VV polarizations for the L-band data sets used in this paper.

The govenment-furnished data is represented as I/Q pairs for each frequency, Tx/Rx pair, GPS location, roll, pitch, and yaw of the array. Using these data, we transform the location of the Tx/Rx pairs to 3D UTM coordinates. This allows for full motion-compensated imaging using the backpropagation method illustrated in Figure 1.[11] Here we denote the radar images as $I_p(u,v)$, where p is the polarization and (u,v) are the horizontal and vertical UTM coordinates. The steps for our backpropagation process are:

1. Remove self-interference by subtracting a windowed time-average of $\mathbf{w}_{jk}(f, t_{GPS})$ over the variable t_{GPS} at each frequency f and for each T/R pair.
2. For each frame (as indicated by t_{GPS}) and polarization:
   (a) Inverse Fourier transform the zero-padded (up-sampled), windowed signals $a_w(f)\mathbf{w}_{jk}(f, t_{GPS})$, where $a(f)$ is a Hamming window, producing the range/time signals $\mathbf{r}_{jk}(t, t_{GPS})$.
   (b) Interpolate and coherently integrate the windowed range signals $a_r(j,k)r_{jk}(t, t_{GPS})$ onto the predetermined grid (u,v), and apply the amplitude and phase correction $r^2 \exp\{-i4\pi f_1 t\}$, where f_1 is the lowest frequency in the stepped frequency transmission. Note that the windowed range signals are only interpolated and integrated onto grid points that are between r_{min} and r_{max} range in front of the array (we use 5 and 25 meters, respectively).

Examples of the images produced by this method may be seen in Figure 2. Note that we use a grid spacing of 2.5cm for both cross-range and down-range, with an upsampling factor on $\mathbf{w}$ of 16 (to the nearest power of 2). This is the most basic form of synthetic aperture radar imaging.

3. CFAR PRESCREENER

The backpropagation process returns a coherently integrated image in UTM coordinates for each polarization of the L-band and X-band FLGPRs. These images are then put through a prescreening algorithm to generate a list of detections. We consider two prescreening methods, both of which are considered *constant false alarm rate* (CFAR) detectors. A block diagram of these prescreeners is shown in Fig. 3. First, we consider the FLGPR image $I(u,v)$. We let u represent the cross-range coordinate and v the down-range coordinate. From here, four images are calculated, denoted as $I_{\mu_c}(u,v)$,

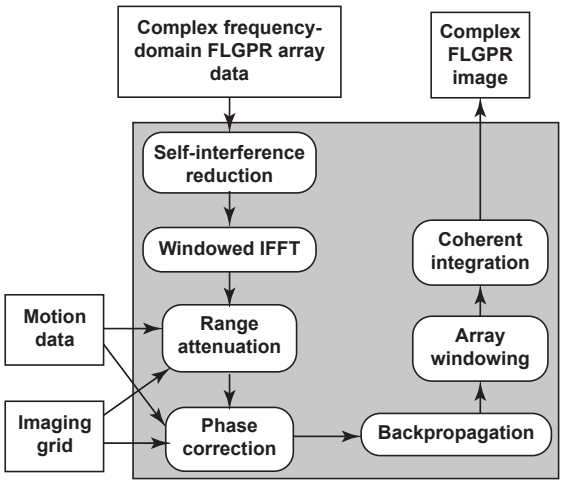
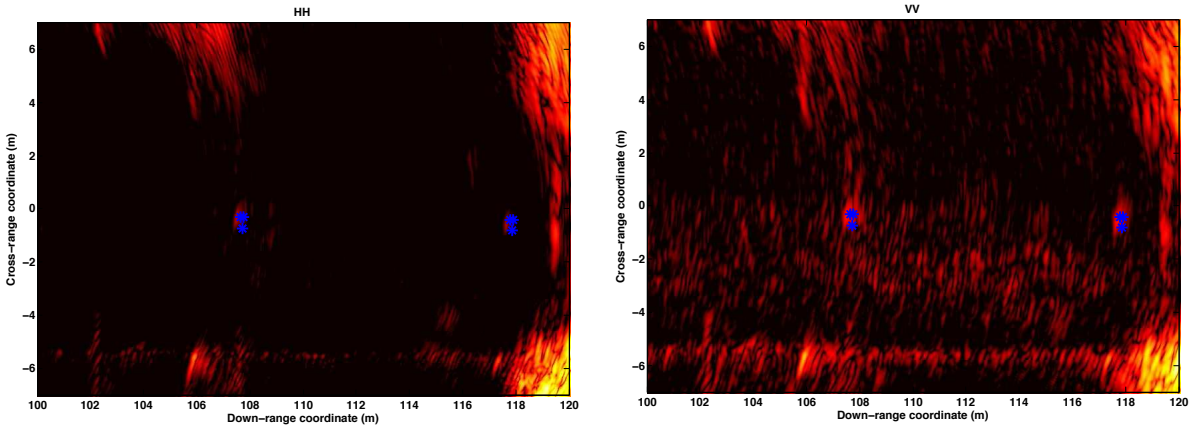
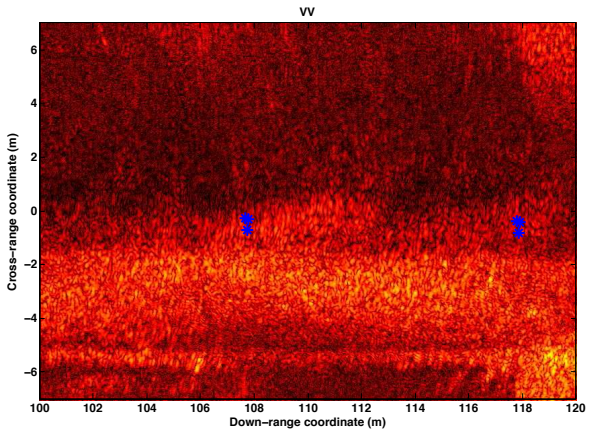


Figure 1. Block diagram of FLGPR backpropagation-based imaging algorithm.[11]



(a) L-band HH

(b) L-band VV



(c) X-band VV

Figure 2. Example FLGPR images for X- and L-band radars—blue markers indicate ground-truth locations.

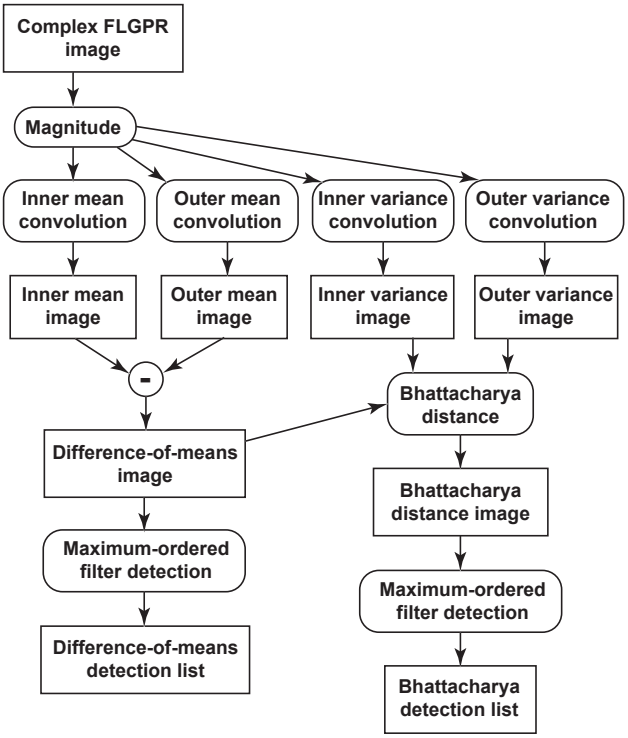


Figure 3. Block diagram of prescreener detection algorithm.[11]

$I_{\mu_h}(u,v)$, $I_{\sigma_c^2}(u,v)$, $I_{\sigma_h^2}(u,v)$, calculated as

$$I_{\mu_c}(u,v) = \frac{\{I * H_c\}(u,v)}{\sum H_c}; \tag{1a}$$

$$I_{\mu_h}(u,v) = \frac{\{I * H_h\}(u,v)}{\sum H_h}; \tag{1b}$$

$$I_{\sigma_c^2}(u,v) = \{I^2 * H_c\}(u,v) - \{I * H_c\}^2(u,v); \tag{1c}$$

$$I_{\sigma_h^2}(u,v) = \{I^2 * H_h\}(u,v) - \{I * H_h\}^2(u,v); \tag{1d}$$

where I^2 indicates the image with each element squared, $*$ indicates convolution, and H_c and H_h are elliptical convolution kernels as shown in Figure 4. This means that I_{μ_c} and I_{μ_h} are mean values of the pixels in the center and halo, respectively, surrounding each pixel and thus $I_{\sigma_c^2}$ and $I_{\sigma_h^2}$ are the corresponding variances. We make use of two methods for indicating detections, one is the *difference of means* (DOM) between the center and halo (also know as a size-contrast filter). The other is a modified Bhattacharya distance, which is signed such that a positive distance indicates that the center mean is greater than the halo mean. This modified Bhattacharya distance is calculated as

$$I_{sc}(u,v) = I_{\mu_c}(u,v) - I_{\mu_h}(u,v); \tag{2a}$$

$$I_B(u,v) = \operatorname{sgn}\{I_{sc}(u,v)\} \cdot \left[\log\left(\frac{1}{4}\left[\frac{I_{\sigma_c^2}(u,v)}{I_{\sigma_h^2}(u,v)} + \frac{I_{\sigma_h^2}(u,v)}{I_{\sigma_c^2}(u,v)} + 2 \right] \right) + \frac{\left(I_{\mu_c}(u,v) - I_{\mu_h}(u,v)\right)^2}{I_{\sigma_c^2}(u,v) + I_{\sigma_h^2}(u,v)} \right]. \tag{2b}$$

From here, we calculate a maximum order-filtered image, denoted $I_o(u,v)$, using a 3m by 1m (cross-range, down-range) rectangular kernel. We use

$$A = \arg_{(u,v)}\{I_*(u,v) = I_o(u,v)\}, \tag{3}$$

to indicate detection locations. In this equation, I_* can be either I_{sc} or I_B and A is the list detection locations given in cross-range, down-range coordinates. Next, we extract a variety of image and texture features from each of these hit locations.

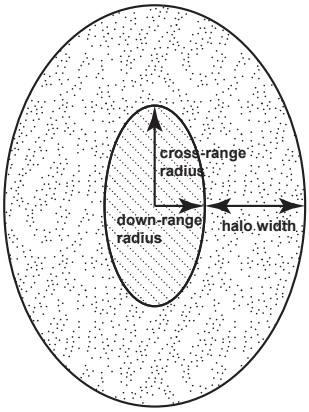
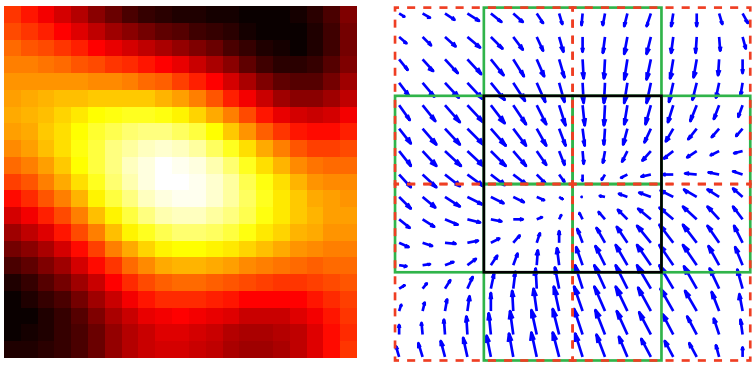


Figure 4. Elliptical convolution kernels used in prescreener. Detection is indicated by comparing distribution of pixel intensities in inner ellipse to distribution of pixel intensities in outer halo.[11]



(a) Sub-image at hit location

(b) Gradient calculation and 3×3 cell arrangement with 50% overlap

(c) Cell-based 9-bin histogram of gradients feature

Figure 5. Example of histogram of ordered gradients (HOG) with 3×3 cell arrangement, 50% overlap of cells, 8×8 pixels per cell. Feature is a $3 \times 3 \times 9 = 81$-length vector of histogram components.

4. FEATURE EXTRACTION

Once the hit list is generated, image features are extracted from the hit locations. The primary image features that were used were the *Histogram of Ordered Gradients* (HOG), *Local Binary Pattern* (LBP) and the *Local Statistics* (LSTAT). Additionally, the imagelet and *Fast Finite Shearlet Transform* (FFST) of each hit were also extracted. The descriptions of these features provided in this section use a 3×3 cell pattern with each cell containing 8×8 pixels. In addition to this cell arrangement, structures of 4×4 cells with each cell containing 16×16 pixels as well as 5×5 cells with each cell containing 24×24 pixels were also tested.

4.1 Histogram of Ordered Gradients

The image-based texture feature, *histogram of ordered gradients* (HOG), was the first feature extracted.[14] The HOG feature works by first calculating the local gradients of each cell surrounding the hit location. From this, an angle histogram is computed for each cell. Each histogram has 9 bin centers, which leads to a total of 9 features per cell. In the 3×3 cell pattern, this yields a total of 81 features to describe each detection. The HOG calculation for an example hit is illustrated in Figure 5.

4.2 Local Binary Patterns

The second feature we use is called *local binary pattern* (LBP). Monochromatic or gray-scale variations in an image can be used to capture the texture of the objects in the image. A very popular and effective method for capturing texture

information is the LBP feature. We use a rotation-invariant uniform LBP that was developed by Ojala et al.[15] The first step of the rotation-invariant uniform LBP is to capture a binary pattern for each pixel in the cell. Although the neighborhood of the LBP can be defined generally, we use an 8 pixel neighborhood with a radius of 1. For each neighborhood we calculate

$$LBP_{8,1} = \sum_{p=1}^{8} s(t_p - t_c)2^p, \tag{4}$$

where

$$s(x) = \begin{cases} 1 & x \geq 0, \\ 0 & x < 0. \end{cases}$$

This is where the feature gets its name, because each value of the summation in (4) contributes a unique bit to the binary representation of the LBP feature. The LBP operator in (4) is calculated for each pixel in the cell. Then each binary string is rotated and uniformed to produce 10 unique labels for each pixel in the cell; this is accomplished by a look-up table. For a detailed description of this process, see Ojala et al.[15] The final step of the LBP feature extraction is the calculation of the histogram for each cell,

$$h_{LBP}(m) = \sum_{u,v \in \text{cell}} S\{LBP_{8,1}(u,v) = m\}, \ m = 1, \ldots, 10, \tag{5a}$$

where $S\{H\}$ is a Boolean function that takes the value of 1 if the argument H is true and 0 else. Since there are 10 unique labels, the histogram contains 10 bins; each bin contains the count of the pixels in the cell with the corresponding uniform rotation-invariant LBP pattern. The histogram is then normalized by

$$\tilde{h}_{LBP}(m) = \frac{h_{LBP}(m)}{\sum_{i=1}^{10} h_{LBP}(i)}. \tag{5b}$$

The normalized histogram values comprise the LBP feature. The LBP feature is calculated for each cell; hence, a feature set for a (3×3) cell is composed of 90 LBP feature values.

4.3 Local Statistics

The third feature extracted is the *local statistics* (LSTAT). These are simply the first four central moments of the imagelet. For reference, the equations for each are given here. In these equations, $x_{m,n}$ refers to the *m-th* row and *n-th* column of the imagelet and $\mathbf{x_n}$ refers to the column vector *n* in the imagelet.

$$\mathbf{U_1} = \left[\left| \frac{\sum_{i=1}^{m} x_{i,1}}{m} \right|, \left| \frac{\sum_{i=1}^{m} x_{i,2}}{m} \right|, \ldots, \left| \frac{\sum_{i=1}^{m} x_{i,n}}{m} \right| \right] \tag{6a}$$

$$\mathbf{U_2} = \left[\left| \sum_{i=1}^{m} (x_{i,1} - \mu_1)^2 \right|, \left| \sum_{i=1}^{m} (x_{i,2} - \mu_2)^2 \right|, \ldots, \left| \sum_{i=1}^{m} (x_{i,n} - \mu_n)^2 \right| \right] \tag{6b}$$

$$\mathbf{U_3} = \left[\left| \frac{\mathbf{E}(\mathbf{x_1} - \mu_1)^3}{\sigma_1^3} \right|, \left| \frac{\mathbf{E}(\mathbf{x_2} - \mu_2)^3}{\sigma_2^3} \right|, \ldots, \left| \frac{\mathbf{E}(\mathbf{x_n} - \mu_n)^3}{\sigma_n^3} \right| \right] \tag{6c}$$

$$\mathbf{U_4} = \left[\left| \frac{\mathbf{E}(\mathbf{x_1} - \mu_1)^4}{\sigma_1^4} \right|, \left| \frac{\mathbf{E}(\mathbf{x_2} - \mu_2)^4}{\sigma_2^4} \right|, \ldots, \left| \frac{\mathbf{E}(\mathbf{x_n} - \mu_n)^4}{\sigma_n^4} \right| \right] \tag{6d}$$

These features are then normalized and strung together as

$$LSTAT(i) = [\mathbf{U_{1,i}}, \mathbf{U_{2,i}}, \mathbf{U_{3,i}}, \mathbf{U_{4,i}}] \tag{7}$$

The reasoning behind using the absolute value of each statistic is to condition the data for use with the deep learning algorithms, which require inputs of $\mathbf{x} \in [0, 1]$ for the use of the sigmoid function. Due to this requirement, the LSTAT feature is somewhat limited in the information it presents and thus is not expected to perform as well as the other two features. It may, however, be useful in conjuction with other features.

4.4 Fast Finite Shearlet Transform

The final feature pulled from the imagelets was the *Fast Finite Shearlet Transform* (FFST) coefficients. This feature was computed using Häuser's FFST algorithm.[16,17] This algorithm is freely available and regularly updated. The FFST is a discrete version of the Shearlet Transform, computed as

$$\|f - f_N\|_2^2 \leq CN^{-2}(logN)^3, \quad N \to \infty. \tag{8}$$

In this equation, f_N is a nonlinear Shearlet approximation of the function f and N is the largest Shearlet coefficient in absolute value. In order to discretize this, finite Shearlets must be introduced. Finite Shearlets rely on three factors: dilation, shear, and translation. Assume an image size of $M \times N$, then let $j_0 = \left\lfloor \frac{1}{2} \log_2 \max\{M, N\} \right\rfloor$ which is equal to the number of considered scales. Also assume that this image is on a grid of $\mathscr{G} = \{(m_1, m_2) : m_1 = 0, ..., M-1, m_2 = 0, ..., N-1\}$. Given this, the dilation, shear, and translation can be defined as

$$\text{Dilation: } a_j = \frac{1}{4^j}, \quad j = 0, ...j_0 - 1; \tag{9a}$$

$$\text{Shear: } s_{j,k} = k2^{-j}, \quad -2^j \leq k \leq 2^j; \tag{9b}$$

$$\text{Translation: } t_m = \left(\frac{m_1}{M}, \frac{m_2}{N}\right), \quad m \in \mathscr{G}. \tag{9c}$$

With these properties, the Shearlets can be written in the time domain as

$$\Psi_{j,k,m}(x) = \Psi_{a_j, s_{j,k}, t_m}(x) = \Psi\left(A^{-1}_{a_j, \frac{1}{2}} S^{-1}_{s_{j,k}} (x - t_m)\right), \tag{10}$$

where $A_a = \begin{pmatrix} a & 0 \\ 0 & \sqrt{a} \end{pmatrix}$ is the dilation matrix and $S_s = \begin{pmatrix} 1 & s \\ 0 & 1 \end{pmatrix}$ is the shear matrix. Now we can take the shearlets into the Fourier domain to get

$$\hat{\Psi}_{j,k,m}(\omega) = \hat{\Psi}(A^T_{a_j} S^T_{s_{j,k}} \omega) e^{-2\pi \mathbf{i} \langle \omega, t_m \rangle} \tag{11a}$$

$$= \hat{\Psi}_1(4^{-j}\omega_1) \hat{\Psi}_2(2^j \frac{\omega_2}{\omega_1} + k) e^{-2\pi \mathbf{i} \left\langle \omega, \binom{m_1/M}{m_2/N} \right\rangle}. \tag{11b}$$

From here, the discrete shearlet transform is simply a matter of multiplying the function of interest by the shearlets and applying the inverse FFT. Parsevel's formula is used for first step to obtain (12), then the 2D IFFT is applied to obtain

$$SH(f)(h, j, k, m) = \frac{1}{MN} \sum_{\omega \in \Omega} \hat{\Psi}(4^{-j}\omega_1, 4^{-j}k\omega_1 + 2^{-j}\omega_2) \hat{f}(\omega_1, \omega_2) e^{2\pi \mathbf{i} \left\langle \omega, \binom{m_1/M}{m_2/N} \right\rangle} \tag{12}$$

This equation can be notationally simplified by defining $\hat{g}_{j,k}(\omega) := \hat{\Psi}(4^{-j}\omega_1, 4^{-j}k\omega_1 + 2^{-j}\omega_2)\hat{f}(\omega_1, \omega_2)$. Using this and applying the 2D IFFT, we arrive at

$$SH(f)(h, j, k, m) = \mathcal{F}^{-1}(\hat{g}_{j,k}), \tag{13}$$

which is the discrete shearlet transfom. A more in-depth derivation of the fast finite shearlet transform may be found in Hauser's paper.[17] Using the algorithm freely available at Häuser's website,* we are able to compute the shearlet coefficients of each hit's imagelet.

5. DEEP BELIEF NETWORKS

DBNs are a type of deep learning network formed by stacking *restricted Boltzmann machines* (RBMs) in successive layers to reduce dimensionality by making a compressed representation of the input. DBNs are trained layer by layer using greedy algorithms and information from the previous layer. In this section, we will first discuss RBMs and how to train them, then move on to training DBNs. To perform the training and testing of both the RBMs and DBNs, we use the freely available Deep Learning Toolbox developed by Rasmus Palm.[18]

*http://www.mathematik.uni-kl.de/imagepro/members/haeuser/ffst/

5.1 Restricted Boltzmann Machines

In this section, we will denote $\sigma(x)$ as the sigmoid activation function,

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{14}$$

Restricted Boltzmann Machines are simple binary learners that generate stochastic representations of the input data. They consist of two layers, one visible and one hidden. The visible layer is the input layer and typically consists of a $1 \times N$ vector of normalized, grayscale pixel values. The hidden layer can then be thought of as a feature representation layer. The defining equation of the RBMs is the energy equation,

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{bv} - \mathbf{ch} - \mathbf{vhW}, \tag{15}$$

where **v** is the input vector, **h** is the hidden feature vector, **b** and **c** are the visible and hidden layer biases, respectively, and **W** is the weight matrix that connects the layers. It should be noted that weights only exist between the hidden and visible layers, that is to say, that the nodes in either layer are not interconnected. **v** is the input and used to train hidden layer **h** as

$$\mathbf{h} = \sigma(\mathbf{c} + \mathbf{vW}). \tag{16}$$

The hidden layer is then used to reconstruct the visible layer in the same manner,

$$\mathbf{v}_{recon} = \sigma(\mathbf{b} + \mathbf{hW}^T). \tag{17}$$

The reconstruction of the visible layer is put back through (16) to form $\mathbf{h}_{recon}$ and then the weight updates are given by

$$\Delta\mathbf{W} = \epsilon(\langle\mathbf{vh}\rangle_{data} - \langle\mathbf{vh}\rangle_{recon}), \tag{18}$$

where ϵ is the learning rate. Iterated over several epochs, this weight update performs a type of gradient descent called Contrastive Divergence.[18]

5.2 Training DBNs

Training a DBN is done layer by layer, where each layer is an RBM. Once the first RBM is trained, its reconstructed hidden layer h_{recon} is used to create the visible layer of the next RBM by

$$\mathbf{v_{n+1}} = \sigma(\mathbf{c}_n^T + \mathbf{h}_n\mathbf{W}_n^T), \tag{19}$$

where n denotes the layer number. Once the visible layer has been created, the layer is trained as an RBM. This cycle is repeated for the number of layers desired. After all layers have been trained, the DBN is typically mirrored to make an encoder-decoder as shown in the *Unrolling* column of Figure 6.[19] Once this has been done, passing an input through the encoder-decoder will produce a reconstruction of the same size as the input. This pass through is done simply using

$$x_{recon,n+1} = \mathbf{W}_n\mathbf{x}_{recon,n}, \tag{20a}$$

$$x_{recon,n-1} = \mathbf{W}_n^T\mathbf{x}_{recon,n}, \tag{20b}$$

where $x_{recon,1} = x_{data}^T$ for the encoding side and $x_{recon,1} = x_{recon}$ on the decoding side. This reconstruction along with the input can then be passed though a cost function and fine-tuning can be performed as shown in the rightmost column of Figure 6. This fine-tuning is often done using stochastic gradient descent.

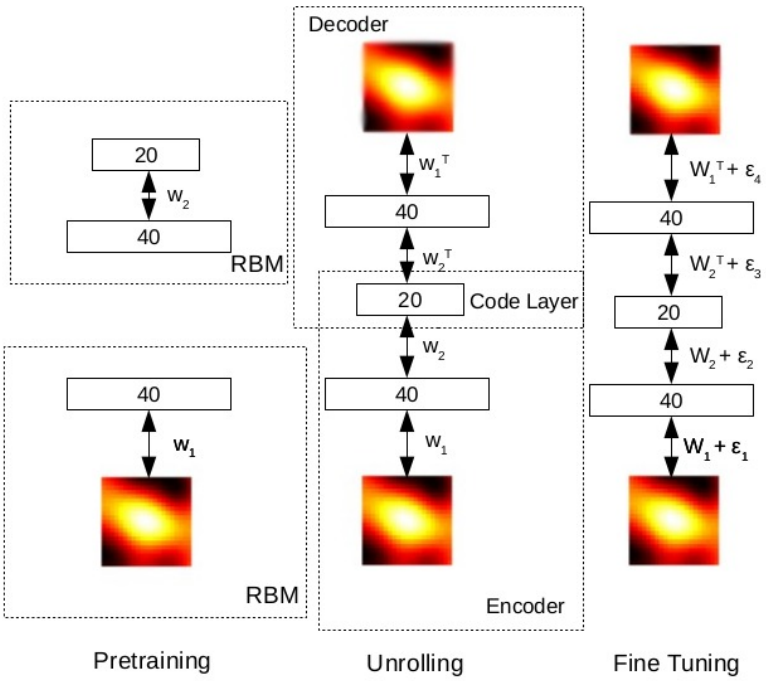


Figure 6. Training of a deep belief network

6. RESULTS

Four lanes of the government-furnished data were used to test our DBN method. We denote these lanes as A, B, C, and D. We tested each lane individually, using the other three lanes for training. In this way, no information from the testing lane is used for training in order for the results to be more representative of real-world operation. We use the *normalized area under the ROC curve* (NAUC) as our performance metric. The NAUC is calculated by

$$NAUC = \frac{1}{0.1}\int_0^{0.1} p_d(FAR)d(FAR), \tag{21}$$

where $p_d(FAR)$ is the probability of detection at a certain *false alarm per square-meter rate* (FAR).[11] We choose to limit the NAUC calculation to FAR < 0.1 for this experiment as an FAR > 0.1 roughly translates to one FA per meter of down range vehicle travel. The NAUC may take values ranging from 0 to 1; where 0 indicates no true-positives are found and 1 indicates that every hit is a true-positive. The data in the tables presented represent the percentage NAUC improvement over the CFAR prescreener. *Reciever operating characteristic* (ROC) curves will also be shown for the most improved results.

Deep belief networks are very useful for learning representations and patterns in large data sets. Because of this, they can be trained to generate accurate representations and reconstructions of data and images, and thus could be useful for learning to detect and reject false alarms in FLGPR data. One way to do this is to use a DBN to learn the notion of a false alarm and then use the reconstruction *root mean-square error* (RMSE) to determine if the object is a target or not. Our process is as follows. First, we use the CFAR Prescreener to determine where the candidate hits are in the training lane and then extract the image features as described in the Section 5. Second, we remove all the true hits so that only the false alarms remain. Third, we train three DBNs, one for each FLGPR channel, on the image features of the false alarms. In doing this, we hope that the DBNs will learn to very accurately model false alarms and, by contrast, do a very poor job of reconstructing true positives. Once the DBNs are trained, we load the testing lane. We once again use the CFAR Prescreener to find hits in the testing lane and then extract their features. We then push each of the prescreener hits through the DBN and take the RMSE between the reconstruction and the feature input. This RMSE is then used as the hits' confidences in the ROC curve. The main idea is that since the DBN is well trained on false alarms, it will model targets very poorly, thus leading to a high RMSE and therefore a high confidence for true positives.

Table 3. Percent NAUC Improvements using Single Image Features with DBN and Phase Adjustment

	Train B, D, A; Test C			**Train C, B, A; Test D**			**Train C, B, D; Test A**			**Train C,B,D; Test B**		
Cellsize:	3x3	4x4	5x5	3x3	4x4	5x5	3x3	4x4	5x5	3x3	4x4	5x5
HOG												
L-band HH:	23	**28**	28	**9.1**	2.0	4.7	-4.8	0.042	**0.57**	1.8	**6.8**	1.2
L-band VV:	31	**52**	38	15	**26**	22	-7.7	-1.7	**2.9**	0.38	5.1	**6.2**
X-band VV:	-9.3	51	**53**	-4.0	**7.3**	5.0	1.2	**2.9**	-0.57	-1.2	0.63	**1.6**
LBP												
L-band HH:	23	**26**	10	-8.0	-1.2	**0.29**	**-6.6**	-7.0	-7.0	-8.0	**-7.9**	-9.5
L-band VV:	0.75	-9.2	**11**	**-4.9**	-6.8	-9.7	-6.2	-7.2	**-4.8**	-8.1	**-2.4**	-6.8
X-band VV:	6.0	**17**	14	**-3.0**	-6.2	-10.0	-7.2	-4.8	**-3.4**	-7.8	-8.2	**-3.0**
LSTAT												
L-band HH:	**9.0**	-6.8	-1.2	**-1.7**	-8.7	-8.9	-7.6	-9.2	**-6.8**	**-5.4**	-8.3	-6.9
L-band VV:	**20**		4.8	-7.2	-9.7	**-5.5**	**-4.8**	-7.9	-7.3	-6.1	-7.4	**-4.6**
X-band VV:	**20**	-6.8	11	**-5.8**	-7.4	-7.1	**-4.7**	-4.9	-8.0	**-4.3**	-4.3	-7.3
FFST												
L-band HH:	**5.2**	4.7	3.4	**3.5**	-4.6	-0.65	**-5.9**	-6.3	-6.8	-5.1	**-2.3**	-5.5
L-band VV:	10	2.7	**16**	-0.68	-1.7	**3.8**	-7.1	**-6.1**	-7.2	-5.3	-8.4	**-4.7**
X-band VV:	**33**	21	22	-6.8	-9.5	**-3.7**	-5.4	-5.6	**-4.0**	-7.6	**-4.4**	-7.0

Table 4. Percent NAUC Improvements using Combinations of Image Features with DBN and Phase Adjustment

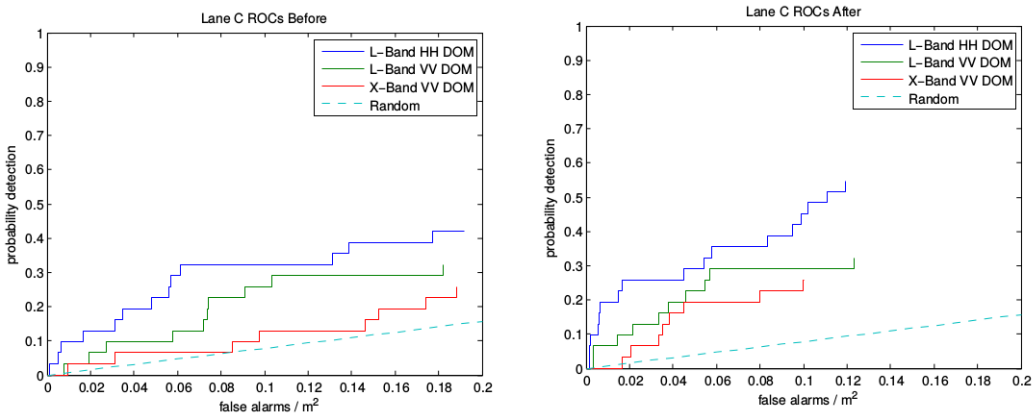
	Train B, D, A; Test C			**Train C, B, A; Test D**			**Train C, B, D; Test A**			**Train A, C, D; Test B**		
Cellsize:	3x3	4x4	5x5	3x3	4x4	5x5	3x3	4x4	5x5	3x3	4x4	5x5
HOG & LBP												
L-band HH:	9.5	13	**23**	8.8	**13**	13	-9.4	-5.3	**1.2**	0.2	-3.6	**6.3**
L-band VV:	36	**52**	39	21	33	**34**	-5.6	-2.8	**2.1**	0.2	2.2	**7.9**
X-band VV:	31	50	**85**	0.19	8.3	**8.9**	-0.15	4.1	**5.4**	-2.9	**0.9**	0.022
HOG & LSTAT												
L-band HH:	**22**	13	3.9	**4.0**	-9.8	-9.9	**-3.1**	-4.8	-6.4	**-0.81**	-2.1	-1.8
L-band VV:	**30**	11	9.1	**2.2**	1.3	-5.5	**-0.67**	-5.6	-4.5	**-3.3**	-3.8	-3.9
X-band VV:	**29**	10	16	**-4.9**	-5.1	-5.3	-0.61	**0.13**	-1.1	**-2.5**	-4.4	-5.5
LBP & LSTAT												
L-band HH:	**11**	4.4	-1.7	**-4.8**	-3.6	-6.3	**-5.6**	-7.6	-8.4	-4.0	**-4.0**	-7.2
L-band VV:	**14**	-3.0	-8.1	-4.8	**3.2**	0.53	**-3.4**	-6.6	-7.6	**-5.0**	-7.2	-6.0
X-band VV:	17	**20**	9.9	-7.5	-6.2	**-5.5**	-0.75	**-0.43**	-0.27	**-6.0**	-6.8	-6.6
HOG, LBP & LSTAT												
L-band HH:	**9.4**	7.9	2.8	**-0.27**	-2.9	-6.1	**-5.7**	-7.3	-7.4	-2.7	**-1.6**	-3.6
L-band VV:	**17**	-1.2	-3.8	-4.3	**5.6**	3.4	**-4.5**	-6.5	-6.5	-4.7	-4.7	**-4.0**
X-band VV:	8.1	**26**	16	-5.6	**-4.8**	-4.8	-1.1	-0.18	**0.62**	-6.1	-5.5	**-5.3**

Since DBNs have many parameters to adjust, many different architectures, learning rates, and epoch amounts were tested. The best combination we found is an architecture using two hidden layers of sizes 40 and 20, giving a full encode-decode stack architecture of [$\mathbf{x}$ 40 20 40 $\hat{\mathbf{x}}$], where $\mathbf{x}$ is the $1 \times N$ feature vector input and $\hat{\mathbf{x}}$ is the $1 \times N$ reconstruction. This architecture is used for all three channels, as is the learning rate of 0.9 and 30 epochs of contrastive divergence for the RBM training. This process was repeated for every combination of image features over three cell-sizes. In each trial, the NAUC of the prescreener alone was compared to the NAUC of the DBN FA rejector and the improvement percentage was calculated. The NAUC improvements of all of these trials can be seen in Tables 3 and 4. The number in each cell is the percentage improvement of the classifier NAUC over the prescreener NAUC. The maximum improvement of each feature set for each polarization and each run is bolded for clarity.

7. CONCLUSIONS AND FUTURE WORK

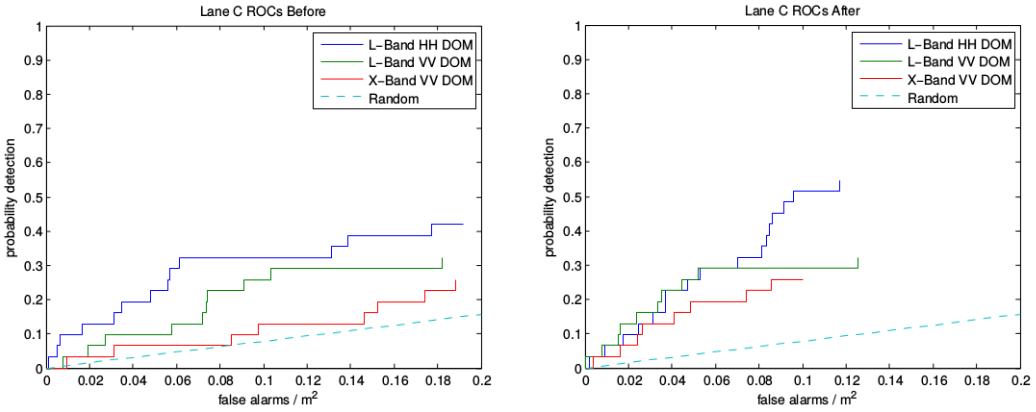
Tables 3 and 4 clearly shows that the DBN is a viable choice for the explosive hazard detection problem using this FLGPR system. We see that the HOG feature is the best single feature, showing NAUC improvements of over 50% in both the L-band and X-band. The {HOG, LBP} combination is the best combination of features, with a maximum of 85% improvement in the X-Band and 52% in the L-band.

We can also see that some lanes are better for testing than others. In particular, the CFAR prescreener performed poorly on lanes C and D, thus the DBN was able to significantly improve the probability of detection. Conversely, the CFAR prescreener performed very well on lanes A and B, which made them better to use as training lanes but left the DBN little room for improvement. Figures 7 and 8 further illustrate the effectiveness of the DBN classifier.



(a) Lane C ROCs: CFAR Detector (b) Lane C ROCs: DBN Classification

Figure 7. Best test lane performance of DBN using single image features



(a) Lane C ROCs: CFAR Detector (b) Lane C ROCs: DBN Classification

Figure 8. Best test lane performance of DBN using combinations of image features

While these figures show great improvements, there is still plenty of room to go further, especially in the overall probability of detection (a max probability of about 50% is unacceptable). Moving ahead, we are investigating the possibility of using a neural network for fine tuning of the DBN. We are also investigating the possibility of fusing data from both the L- and X-band arrays to further improve detection rates. Lastly, we are developing a temporal method for computer-aided detection and classification that we believe we show great improvements by leveraging the (ideally) disparate temporal fluctuations of target and clutter.

ACKNOWLEDGMENTS

This work has been funded in part by U.S. Army contracts W909MY-13-C-0013 and W909MY-13-C-0029. We would like to acknowledge U.S. Army RDECOM CERDEC NVESD for vital support in this project.

Superior, a high performance computing cluster at Michigan Technological University, was used in obtaining results presented in this publication.

REFERENCES

[1] JIEDDO COIC MID, "Global IED monthly summary report," (2012).

[2] Cremer, F., Chavemaker, J. G., deJong, W., and Schutte, K., "Comparison of vehicle-mounted forward-looking polarimetric infrared and downward-looking infrared sensors for landmine detection," in [*Proc. SPIE*], **5089**, 517–526 (2003).

[3] Playle, N., Port, D. M., Rutherford, R., Burch, I. A., and Almond, R., "Infrared polarization sensor for forward-looking mine detection," in [*Proc. SPIE*], **4742**, 11–18 (2002).

[4] Costley, R., Sabatier, J. M., and Xiang, N., "Forward-looking acoustic mine detection system," in [*Proc. SPIE*], **4394**, 617–626 (2001).

[5] Bradley, M. R., Witten, T. R., Duncan, M., and McCummins, R., "Anti-tank and side-attach mine detection with forward-looking gpr," in [*Proc. SPIE*], **5415**, 421–432 (2004).

[6] Cosgrove, R. B., Milanfar, P., and Kositsky, J., "Trained detection of buried mines in sar images via the deflection-optimal criterion," *IEEE Trans. Geoscience and Remote Sensing* **42**(11), 2569–2575 (2004).

[7] Sun, Y. and Li, J., "Plastic landmine detection using time-frequency analysis for forward-looking ground-penetrating radar," in [*Proc. SPIE*], **5089**, 851–862 (2003).

[8] Havens, T. C., Keller, J. M., Ho, K. C., Ton, T. T., Wong, D. C., and Soumekh, M., "Narrow band processing and fusion approach for explosive hazard detection in flgpr," in [*Proc. SPIE*], **8017**, 80171F (2011).

[9] Farrell, J., Havens, T. C., Ho, K. C., Keller, J. M., Ton, T. T., Wong, D. C., and Soumekh, M., "Detection of explosive hazards using spectrum features from forward-looking ground penetrating radar imagery," in [*Proc. SPIE*], **8017**, 80171F (2011).

[10] Farrell, J., Havens, T. C., Ho, K. C., Keller, J. M., Ton, T. T., Wong, D. C., and Soumekh, M., "Evaluation and improvement of spectral features for the detection of buried explosive hazards using forward-looking ground-penetrating radar," in [*Proc. SPIE*], **8357**, 8357C (2012).

[11] Havens, T. C., Becker, J., Pinar, A., and Schulz, T. J., "Multi-band sensor-fused explosive hazards detection in forward-looking ground-penetrating radar," in [*Proc. SPIE*], **9072**, 90720T (2014).

[12] Stone, K. and Keller, J. M., "Convolutional neural network approach for buried target recognition in fl-lwir imagery," in [*Proc. SPIE*], **9072**, 907219 (2014).

[13] Besaw, L. E. and Stimac, P. J., "Deep learning algorithms for detecting explosive hazards in ground penetrating radar data," in [*Proc. SPIE*], **9072**, 90720Y (2014).

[14] Dalal, N. and Triggs, B., "Histograms of oriented gradients for human detection," in [*Proc. IEEE CVPR*], 886–893 (2005).

[15] Ojala, T., Pietikainen, M., and Maenpaa, T., "A generalized local binary pattern operator for multiresolution gray scale and rotation invariant texture classification," in [*Proc. Int. Conf. Advances in Pattern Recognition*], 397–406, Springer Berlin Heidelberg (2001).

[16] Hauser, S. and Steidl, G., "Convex multiclass segmentation with shearlet regularization," *International Journal of Computer Mathematics* **90**, 62–81 (2013).

[17] Hauser, S. and Steidl, G., "Fast finite shearlet transform: a tutorial," *ArXiv* (1202.1773v2) (2014).

[18] Palm, R. B., *Prediction as a candidate for learning deep hierarchical models of data*, PhD thesis, Technical University of Denmark (2012).

[19] Hinton, G. E. and Salakhutdinov, R. R., "Reducing the dimensionality of data with neural networks," *Science* **313**(5786), 504–507 (2006).