Deep convolutional neural networks for ATR from SAR imagery

David A. E. Morgan[a]

[a] BAE Systems, UK

ABSTRACT

Deep architectures for classification and representation learning have recently attracted significant attention within academia and industry, with many impressive results across a diverse collection of problem sets. In this work we consider the specific application of Automatic Target Recognition (ATR) using Synthetic Aperture Radar (SAR) data from the MSTAR public release data set. The classification performance achieved using a Deep Convolutional Neural Network (CNN) on this data set was found to be competitive with existing methods considered to be state-of-the-art. Unlike most existing algorithms, this approach can learn discriminative feature sets directly from training data instead of requiring pre-specification or pre-selection by a human designer. We show how this property can be exploited to efficiently adapt an existing classifier to recognise a previously unseen target and discuss potential practical applications.

Keywords: Deep Learning, Convolutional Neural Networks, ATR, SAR, Machine Learning, Image Processing

1. INTRODUCTION

It has long been an ambition of artificial intelligence researchers to use deep, distributed computational architectures to solve complex recognition problems by learning abstraction hierarchies that are robust to noisy data. Many of these ambitions are biologically inspired, with a commonly held view being that that the human brain derives much of its sophisticated processing ability from operating in this way.[1] Recent advances in the understanding of computation with such architectures have opened up new possibilities for developing sophisticated algorithms for classification, representation learning and feature extraction. 'Deep learning' is a methodology involving the learning of layered models in which different layers relate to concepts of increasing complexity. In many of the most common implementations such as artificial neural networks and Deep Belief Networks, the desired goal is for higher-level concepts to emerge naturally through training as combinations of simpler ones.

Despite their many desirable properties and theoretical advantages, deep architectures have until relatively recently been largely outperformed in practical classification scenarios by more traditional black-box techniques such as Support Vector Machines (SVMs) and decision trees with hand-crafted feature sets. However, sparked by a research breakthrough by Hinton in 2006[2] and advances in computer hardware, this maxim no longer holds for many applications. A number of results published in the past 5 years suggest that classification with deep architectures may represent a step improvement in solving difficult image classification problems, particularly in situations where expansive training data sets are available and there may be many class labels.[3] Furthermore, the traditional view of a machine learning problem being either a purely supervised or unsupervised task has become more flexible in recent times. Even in situations where the quantity of unlabelled data vastly exceeds the available labelled data, approaches such as stacked autoencoders allow progress to be made in respect of identifying a feature hierarchy.

The benefits of capable ATR systems in battlefield environments are significant and well understood. However, retaining the ability to classify increasingly diverse sets of small, configurable objects captured using high-fidelity sensor systems which generate large quantities of data may require the development of algorithmic approaches that generalise better and are easier to train. Deep learning algorithms may be particularly well suited to this task because of their proven ability to discover discriminating feature sets through training, particularly in data rich environments. Traditional approaches that rely on hand specified feature sets can be very effective, but

Further author information: (Send correspondence to David Morgan)
David Morgan: E-mail: david.morgan15@baesystems.com

Algorithms for Synthetic Aperture Radar Imagery XXII, edited by Edmund Zelnio, Frederick D. Garber, Proc. of SPIE Vol. 9475, 94750F · © 2015 SPIE
CCC code: 0277-786X/15/$18 · doi: 10.1117/12.2176558

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

may also be time-consuming to develop and may not respond well to new or unforeseen target types because the optimal encoding of the discriminating characteristics may change in a non-trivial way.

In this study we explore the performance of a Deep CNN approach to the classification of SAR imagery using the MSTAR public release data set. Much of the existing literature and benchmarking for Deep CNNs focusses on the classification of optical imagery in the visible band; here we seek instead to understand the extent to which this machine learning technique can capture the complexities of SAR imagery. For example, unlike typical optical imagery, small variations in perspective can lead to substantial differences in the intensity of energy scattered into the receptive field of the sensor e.g. due to irregularly shaped surfaces. When compared with typical data sets cited in much of the current Deep learning research, raw SAR data can be characterised as generally dark, specular and having relatively poorly defined edges.

2. DEEP LEARNING WITH NEURAL NETWORKS

Deep learning is an umbrella term for an assortment of machine learning techniques that typically aim to learn a compact representation of complex data.[4] One of the most commonly used approaches is to use artificial neural networks, which have in various forms been used for many years to solve classification and regression problems.[5] The key difference between traditional 'shallow' neural networks and more recent 'deep' network designs relates to the number of hidden layers that encode the patterns to which the network responds; with the former there may be a single hidden layer whereas the latter may have 5-10 or more. The additional expressive power afforded by such a construction can allow for complicated and highly non-linear relationships to be identified through training alone, in many cases against raw data that may have had little or no significant pre-processing. In this study we work with raw pixel-wise SAR magnitudes and therefore do not require a pre-processing pipeline of the type used in other leading work on this problem, often captured in a directed graph format such as those given by Jingjing,[7] Pham[8] and Schumacher.[9]

Deep CNNs are neural networks whose topology encodes a spatial relationship between nodes in successive layers that is analogous to the convolution operation. There are many examples of such architectures performing well on image classification problems, such as the performance reported by Lecun[6] for the MNIST handwritten digit dataset. A multitude of extensions to the basic architecture and training mechanism have been proposed, for example supplementing the data set with noisy, rotated or scaled versions of the images and randomly setting to zero a fraction of the pixels in a given layer ('dropout'). Reducing reliance on bespoke or generic feature sets without precluding their inclusion is a desirable property of a CNN based approach that promotes robust classification and may facilitate the integration of existing pre-processing techniques where these are known to be beneficial.

Adopting this type of data-driven approach opens up some exciting possibilities, such as modifying and adapting the parameters of a pre-trained classifier to respond to previously unseen objects types. In this scenario, it is envisaged that the lower level features encoded into the existing representation would remain largely unchanged whilst additional high level structures could be learned with little effort. In practical terms, this might mean running a training algorithm in an online fashion against data being labelled in real time by a human. This could improve ATR performance against specific targets by incorporating any distinguishing customisations or the prevailing weather conditions. Deep CNN implementations increasingly exploit GPU capability because the operations of forward and backward propagation are highly parallelisable. As the power needed per unit of computation decreases, the possible applications of this type of online learning become increasingly broad.

When training a deep neural network, the main challenges are in setting up the correct topology, choosing appropriate parametric initialisations, defining an objective function to quantify misclassification error and an appropriate algorithm for training. We now address each of these in the context of the MSTAR public release data set.

2.1 Topology & Initialisation

For this study the topology detailed in Table 1 is used, which corresponds to a total of 88,162 parameters. Through successive convolution operations, the input image is passed through a series of non-linear transforms to create a collection of feature maps. The initial layers contain relatively few, larger feature maps, whilst latter

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

Table 1. Deep CNN topology used in this study. The input layer is a direct mapping to the SAR image and the output layer denotes the class marginal probabilities. The final fully connected layer behaves like a traditional neural network and the operation of the convolution and max pooling operators are described in appendices A.1 and A.2 respectively.

Layer Type	Image Size	Feature Maps	Kernel Size
Input Layer	128×128	1	-
Convolutional	120×120	18	9×9
Max Pooling	20×20	18	6×6
Convolutional	16×16	36	5×5
Max Pooling	4×4	36	4×4
Convolutional	1×1	120	4×4
Fully Connected	-	1	120×10
Output	10	-	-

layers contain a greater number of smaller feature maps. The final convolutional layer is a set of $1{\times}1$ feature maps which have no specific spatial relationship and can be represented as numbers within a fully connected layer. By stacking multiple convolutional layers, the effective size of the receptive field (with respect to the input image) grows with each successive layer. Therefore, whilst the pixels in each layer are locally connected to the previous layer, pixels in the latter layers are globally coupled to the input.

By modern standards this is a modestly sized CNN, but it is arguably a reasonable choice given that the quantity of training data is also relatively small. The network parameter values are initialised using a fan-in scheme, meaning that each weight is set to a uniformly random value in the range $[-\frac{1}{fan\,in}, \frac{1}{fan\,in}]$ where $fan\,in$ refers to the number of nodes connecting as an input to a given node.

From a theoretical perspective, the 2D convolution operation itself is arguably too simplistic a representation for this application because it is only implicitly invariant to translations, not to rotations, scaling or any of the scattering processes involved in the complex physics behind a SAR image. This is an important consideration because, for example, it is well-known that the inclusion of logic to estimate the pose of a target can improve classification performance. Any quantity that a CNN is not structurally invariant to must therefore be captured by its parameters, which increases the risk of over-fitting and makes the classification algorithm more sensitive to the specific sensor used to capture the imagery. Despite these potential deficiencies, the approach has proven to be unexpectedly effective in a variety of scenarios, particularly where there is a suitable quantity of labelled data and a sufficiently large number of nodes in the network.

2.2 Error Function

To train the parameters θ of a neural network it is necessary to first define an error function $E(\theta, D)$ that measures the quality of fit between the class label predictions made using θ and the ground truth labels in the data set D. For this purpose the cross entropy error function (1) is used, which is a common choice for multi-way classification problems. The data set consists of the tuples $\left\{\mathbf{x}^{(j)}, T^{(j)}\right\}$, i.e. the vectorised input image $\mathbf{x}$ and the associated class label T from the j$^{\text{th}}$ training instance. The output layer of the CNN is denoted $\mathbf{y}$, which is a function of the network input $\mathbf{x}$ and the current parameters θ. It has N entries, where N is the number of class labels, and is therefore the same length as the one hot encoding of the target class label T_i.

$$E(\theta, D) = \frac{1}{|D|} \sum_{j=0}^{|D|} \sum_{i=0}^{N} T_i^{(j)} \log \mathbf{y}_i^{(j)} \left(\mathbf{x}^{(j)}, \theta\right) \tag{1}$$

To use the cross entropy error in this form, the output layer of the CNN is chosen to be a softmax layer. The entries in $\mathbf{y}$ are therefore set by passing the output from the previous layer $\mathbf{x}'$ through the softmax function (2). This function has the necessary property for the cross entropy error that each element $\mathbf{y}_i \in [0, 1]$.

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

$$\mathbf{y}_i = \frac{\exp \mathbf{x'}_i}{\sum_{k=0}^{K} \exp \mathbf{x'}_k} \tag{2}$$

2.3 Training

Whilst it has long been known that parametric gradients with respect to error functions in neural networks can be calculated efficiently using back propagation, the most effective techniques for solving the associated optimisation problem remain an active area of research. Training deep CNNs is similarly accepted as being difficult, because despite the weight sharing implicit in their convolutional structure a very large number of free parameters typically remain. The fitness landscape associated with this error function usually has a challenging topology with many local minima into which standard optimisation algorithms can easily become trapped. Furthermore, the computational complexity of calculating the error function $E(\theta, D)$ and its exact Jacobian $\frac{\partial E}{\partial \theta}$ scales linearly with the size of the data set which may be problematic for large data sets. A popular approach that addresses both of these issues is to use mini-batch Stochastic Gradient Descent (SGD - see A.4); we use this method in conjunction with momentum and weight decay, following an approach similar to Krizhevsky.[3]

$$p_{i+1} = \alpha p_i - \beta \varepsilon \theta_i - \varepsilon \left\langle \left. \frac{\partial E}{\partial \theta} \right|_{\theta_i} \right\rangle_{D_i} \tag{3}$$

$$\theta_{i+1} = \theta_i + p_{i+1} \tag{4}$$

The parameters in the equations are the learning rate ε, the momentum coefficient α and the weight decay coefficient β, where the average of the derivatives with respect to θ over the i$^{\text{th}}$ batch of data D_i is given by $\langle \frac{\partial E}{\partial \theta}\big|_{\theta_i} \rangle_{D_i}$. The coefficient values given in Table 2 were used for the initial part of the optimisation, with a reduced learning rate used during the latter phase of the training process as the parameters converge towards a solution.

Table 2. Parameters for Stochastic Gradient Descent training.

Parameter	Value
α	0.9
β	5×10^{-4}
ε	0.0125

3. EXPERIMENTAL APPROACH

The deep CNN outlined in Section 2 is now applied to the problem of ATR for the SAR imagery in the MSTAR public release data set. This data set consists of airborne SAR images of various vehicle types taken at depression angles of 15, 17, 30 and 45 degrees, with the majority of these at 15 and 17 degrees. These 128×128 image chips are taken from many different longitudinal angles and there are therefore a broad range of perspectives of the same vehicle contained within the data. In general the underlying terrain is flat and there are no occlusions, although some vehicles are present in multiple configurations (i.e. serial numbers).

We consider the ten-way classification problem which involves the vehicles listed in Table 3 and the associated number of training and test images. Detailed in Ross[10] are widely referenced guidelines for evaluating performance. The standard approach is to fill the training set with images taken at a 17 degree depression angle and the test set with images taken at a 15 degree depression angle. We adhere to this methodology when training and testing the deep CNN for the benefit of comparison with other work. Whilst this ten-way classification problem has been addressed by other literature, there are inevitably variations in the methodology and presentation of performance figures that make a direct comparison difficult. For example, some studies distinguish between different vehicle serial numbers and do not consider rejected samples as a misclassification. Here we do neither and simply take the highest scoring class label to be the result. Furthermore, we have made no attempt to fine-tune parameters such as kernel sizes, the neural network topology or the parameters for the SGD optimisation.

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

Table 3. Test and training image sets.

Vehicle	Training Images (17 Degrees)	Test Images (15 Degrees)
2S1	299	274
BMP2	698	587
BRDM2	298	274
BTR60	256	195
BTR70	233	196
D7	299	274
T62	299	273
T72	691	582
ZIL131	299	274
ZSU23	299	274

4. RESULTS

The history of the cross entropy error function throughout the SGD optimisation is shown in Figure 1. There are three distinct stages of the optimisation. The first shows only very marginal improvements in performance and runs to ∼800 iterations. With the parameters having been initialised randomly, the initial steps tend to move parameters in directions such that successive iterations have an oscillatory cancelling effect. The second phase, between ∼800 and ∼3,000 iterations demonstrates learning at an effective rate; there is an approximately linear decrease in the error function during this period. The learning rate parameter ε was reduced by a factor of 10 at ∼3,000 iterations, which is necessary to damp some of the wild oscillations in the cross entropy error of successive mini-batches. With this reduced learning rate parameter the latter stage of the optimisation process exhibits stable convergence towards a good minimum of the objective function. After 15,000 iterations the optimisation is terminated and the trained CNN is evaluated against the test data.

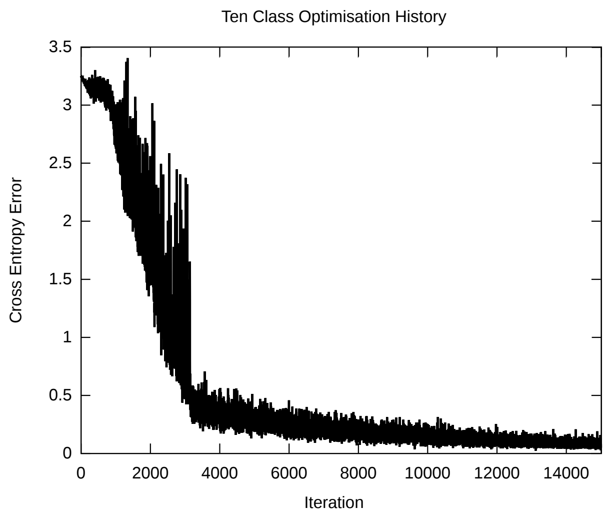

Figure 1. History of the cross entropy error value for each mini-batch throughout the training procedure.

Shown in Appendix B are the nodal activations of the first two convolutional layers in the network for a successfully classified image of a BTR60 target using the CNN. The input layer is the raw SAR image. Figure 7 depicts how each of the 18 feature maps in the first convolutional layer respond to stimulation by this input image and Figure 8 shows how the second convolutional layer responds to the output of the first max pooling layer. It is evident that around a third of the feature maps in the second convolutional layer are within a few

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

Table 4. Confusion matrix for the MSTAR 10-way ATR classification problem.

	2S1	BMP2	BRDM2	BTR60	BTR70	D7	T62	T72	ZIL131	ZSU23	Overall (%)
2S1	241	1	2	0	3	0	1	15	11	0	88.0
BMP2	5	551	1	5	2	1	1	21	0	0	93.9
BRDM2	6	10	235	6	6	0	0	4	6	1	85.8
BTR60	3	3	3	175	6	0	0	3	1	1	89.7
BTR70	9	4	3	2	177	0	0	1	0	0	90.3
D7	0	0	0	0	0	267	2	0	2	3	97.4
T62	2	1	0	0	0	2	236	10	11	11	86.4
T72	3	7	0	1	0	0	19	548	4	0	94.2
ZIL131	4	1	2	2	0	0	3	0	262	0	95.6
ZSU23	0	0	0	0	0	7	1	1	2	263	96.0

pixels of being fully deactivated, suggesting the emergence of a learned feature set by this mid-point in the neural network structure. The output layer is a 10 element vector representing the softmax probability for class membership; a confident and accurate response is given for this test image.

4.1 Performance

Shown in Table 4 is the confusion matrix for the trained CNN. Whilst variations in the testing methodologies used by different authors make it difficult to compare results directly with other studies, in broad terms the overall performance of 92.3% correct classifications achieved on this problem appears generally in line with what has been achieved in comparable studies. For example O'Sullivan[11] reports an overall 95.05% successful classification rate for the same 128×128 problem under similar conditions using the Conditionally Gaussian model technique. Experiments conducted by Srinivas[12] suggest a similar but slightly lower classification rate 92.8% for this method, compared with 90.1% for the SVM derived method developed by Zhao[13] and 92.7% for the Adaptive Boosting technique proposed by Sun.[14]

4.2 Adaptive Training

We now explore the extent to which a pre-trained deep CNN classifier for SAR imagery can be retrained to respond to previously unseen objects. This hypothesis is tested in the following way:

1. Training the CNN in the usual way against the standard training data set with all of the BTR60 vehicles excluded. This is a 9-way classification problem, the mini-batch error function history for which is shown in Figure 2.
2. Re-adding the BTR60 vehicles and measuring how long the algorithm takes to re-converge to an acceptable solution. Shown in Figure 3 are the error function values over 2500 iterations of this task.

This experiment demonstrates that this deep CNN can be reconfigured to give good accuracy on the 10-way problem using relatively few additional iterations in comparison to the initial training process. It is approximately 10-20 times quicker to adapt the 9-way classifier than it is to train a new 10 way classifier from a random initialisation. This ability to fine tune the feature set in response to new objects with little additional effort is potentially a significant advantage in using a CNN approach to ATR over a traditional approach.

5. CONCLUSIONS

In this study we have explored the performance of a deep CNN for the problem of ATR using SAR imagery. On the well-studied MSTAR ATR benchmark problem, this method was found to offer performance broadly similar to that available using other state-of-the-art methods across all vehicle types. Through introspection of

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

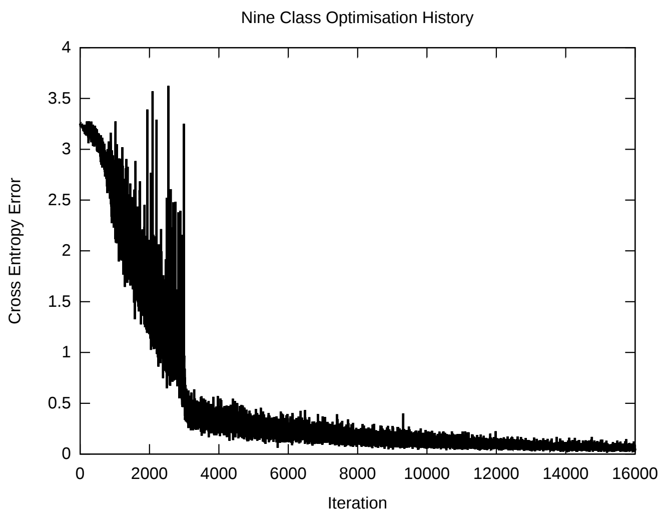

Figure 2. Cross entropy error training history for the 9-way classification problem with BTR60 vehicles excluded.

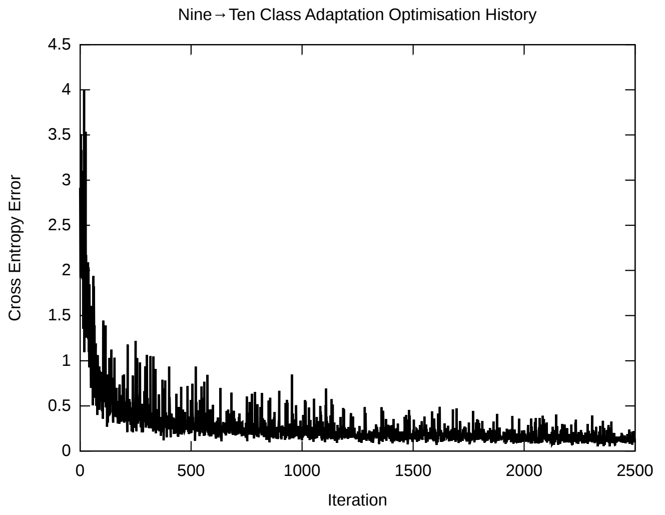

Figure 3. Cross entropy error training history for the 10-way classification problem, with parameters initialised to the values obtained at the converged solution demonstrated in Figure 2.

the parameters in the trained neural network, it is possible to understand the type of features learned by the CNN and by extension understand what the discriminating features of the imagery in the data set are.

The CNN approach was able to offer competitive performance whilst learning all of its features directly from the data rather than needing these pre-specified, which could be of significant benefit in some circumstances. These results were obtained without any specific effort to fine tune the network topology, optimisation parameters or selectively choose the most successful instance of the trained model on this data set. Furthermore, this study suggests that a trained classifier using the CNN approach could be adapted relatively quickly to incorporate new, previously unseen targets. We conclude that a CNN and more broadly the 'Deep learning' approach to ATR may hold significant promise for image analysis and recognition problems.

ACKNOWLEDGMENTS

This research was supported under the ASUR programme by DSTL.

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

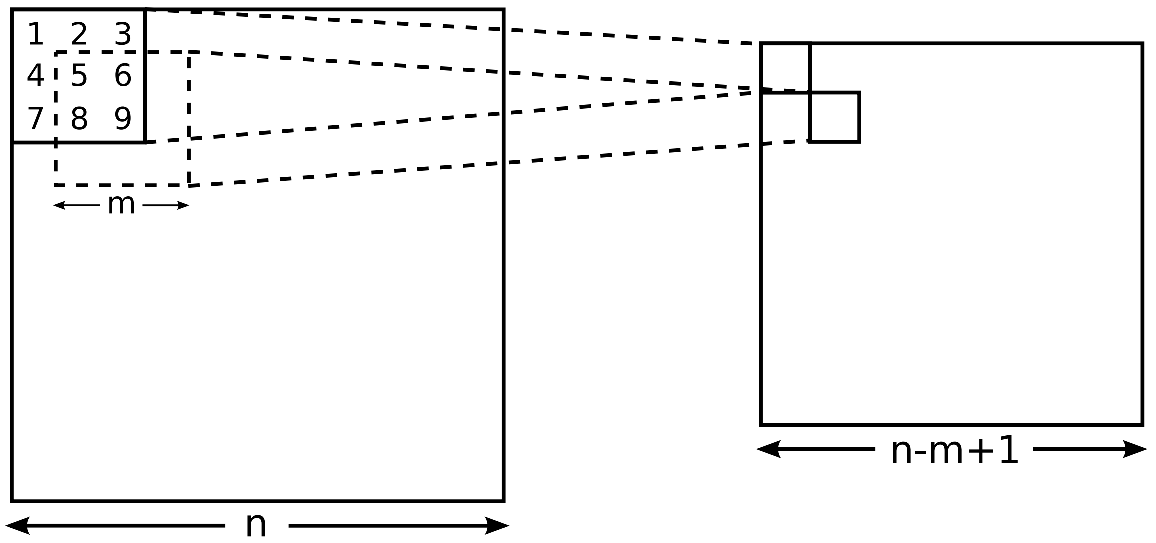

Figure 4. The valid convolution operation.

APPENDIX A. CONVOLUTIONAL NEURAL NETWORK ARCHITECTURE

A.1 Convolution Operation

The fundamental operation underpinning a CNN is the 2D convolution. Figure 4 depicts the 'valid' convolution operation that constitutes the basis of the CNN implemented in this work. Following notation similar to Masci[15] the ij$^{\text{th}}$ pixel of the k$^{\text{th}}$ output feature map, h^k_{ij}, is expressed in Equation 5 in terms of a sum of 2D convolutions between one or more input images x^l and the entry from the CNN weight tensor W^{kl}_{ij} associated with the correspondingly indexed entries.

$$h^k_{ij} = \sigma\left(\left(\sum_l x^l * W^{kl}_{ij}\right)_{ij} + b^k\right) \tag{5}$$

Other terms in this expression are the biases b^k associated with each output feature map and the activation function σ, which is taken as being constant for each layer. Note that the sum over 2D convolutions is sometimes interpreted as a 3D convolution, which can be a helpful observation when trying to implement this operation efficiently. Formally, the convolution operator implies a stride of size 1, i.e. the change in offset of the window function in each dimension of the input image between neighbouring pixels in the output image. This is depicted in Figure 4 and is how the CNN is implemented in this work. However, it is noted that larger strides are frequently used in literature, which reduces the dimensionality of the output feature maps and can therefore be particularly useful when working with large images.

A.2 Max Pooling Operation

It has been established that the topology of a CNN requires a gradual decrease in image size between successive layers. One way to achieve this is to intersperse convolutional layers with 'pooling' layers, which typically reduce the resolution of the output by performing operations over groups of pixels in the input feature map. Unlike convolutional layers, such a layer may not have any tunable parameters. We assume that each feature map in a convolutional layer is connected to only a single map in the associated pooling layer.

Following the notation in Scherer,[16] a general pooling operation transforms an $N \times N$ image patch from an input into a smaller $n \times n$ patch at the output. One commonly used pooling operation is the sub sampling function, which is a form of weighted averaging over all of the $n \times n$ image patches within the larger $N \times N$ patch.

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

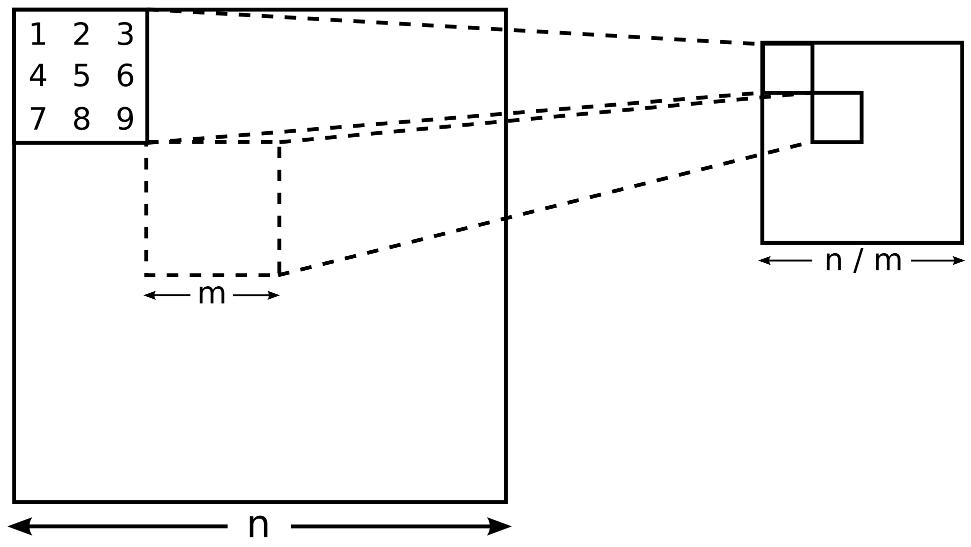

Figure 5. The max pooling operation.

$$a_{kl}^{n\times n} = \sigma\left(\beta \sum_{(i,j)\in N\times N} a_{ij}^{n\times n} + b\right) \tag{6}$$

One possible sub sampling pooling function includes the trainable scalar β, a bias parameter b and an activation function σ. Setting $\beta = 1/\#$image patches in $N \times N$, b = 0 and σ as the identity function corresponds to simple averaging. An alternative is the max pooling function, which selects the patch with the maximum value in the pooling region, subject to a shaped windowing function $u(x, y)$.

$$a_{kl}^{n\times n} = \max_{(i,j)\in N\times N}\left(a_{ij}^{n\times n} u\left(n,n\right)\right) \tag{7}$$

The relative performance of these pooling operations was compared by Scherer,[16] including the effect of choosing different pooling region shapes. That study found that the max pooling operation significantly outperformed sub sampling operations and that overlapping pooling windows offered no significant improvement over non-overlapping windows.

$$a_{kl} = \max_{\substack{i\in[m\times k, m\times(k+1)) \\ j\in[m\times l, m\times(l+1))}}\left(a_{ij}^{1\times 1}\right) \tag{8}$$

Shown in Figure 5 is the pooling operation described by the above equation, which uses a non-overlapping square region. This operation is not associated with any parameters, but instead selects the largest value within the input square. This operation was experimentally determined to yield good results and we therefore use it in this study.

A.3 Rectified Linear Units

The most commonly used non-linearities in neural networks have typically been *tanh* or sigmoidal (frequently inspired by a biological interpretation), which return a result in a fixed range such as $[0, 1]$ that is useful, for example, when a probabilistic interpretation is required. However, in deeper networks these are known to cause problems during training, saturating quickly away from their inflexion point and causing the network as a whole to suffer from the 'vanishing gradients' problem. This refers to the nullifying effect a saturated node has on the

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

propagation of derivatives through to the preceding nodes, i.e. successful gradient based training cannot occur in the initial layers.

Recent research has suggested that the use of Rectified Linear Units (ReLU), which are defined by the non-linearity $f(x) = \max(0, x)$, may be more effective. This is not a saturating non-linearity so the associated network does not suffer from the vanishing gradients problem because the gradient is binary. Furthermore, this choice can improve general computation performance because the max function is simpler to evaluate than the exponential function. We use ReLU units throughout this study apart from where otherwise mentioned.

A.4 Optimisation Strategies

Training a deep CNN involves solving a non-convex, non-linear and degenerate optimisation problem. In general, the computational cost of calculating its parametric Hessian for a large data set is prohibitive and therefore Newton based solvers are rarely used. More tractable alternatives include the Conjugate Gradient method and the Quasi-Newton family of algorithms such as (L-)BFGS; these algorithms require only a Jacobian, which can be calculated efficiently via error back propagation. However, the prevailing optimisation technique for training large neural networks is Stochastic Gradient Descent (SGD). This is an efficient approach that takes sequential steps in the approximate gradient direction of the fitness function, with a step size determined by the learning rate parameter ε. Its performance is derived from approximating the gradient at each step using only a small subset of the data points known as a mini-batch. This approximation becomes increasingly accurate for larger mini-batch sizes and SGD can therefore be scaled to trade speed off against a precise search direction, which is particularly important for large data sets. The additional noise introduced by using this stochastic approach to the gradient calculation is generally considered to be helpful for preventing the algorithm getting trapped in local minima. One further consideration when training a large neural network is the need for regularisation to mitigate the tendency to over fit. This can be done in several ways. For example, an explicit L_1 or L_2 regularisation term can be added to the objective function or 'dropout' can be used, which involves randomly setting some of the activations in each layer to zero. In this study we implicitly add an L_2 regulariser to the objective function by appending the associated weight decay term into the Jacobian (Equation 3).

APPENDIX B. TRAINED ACTIVATIONS ON A SAR IMAGE

For a BTR60 vehicle the nodal activations in the network are visualised in Figures (6-8), giving an indication of the structure to which the feature maps in the CNN are sensitive.

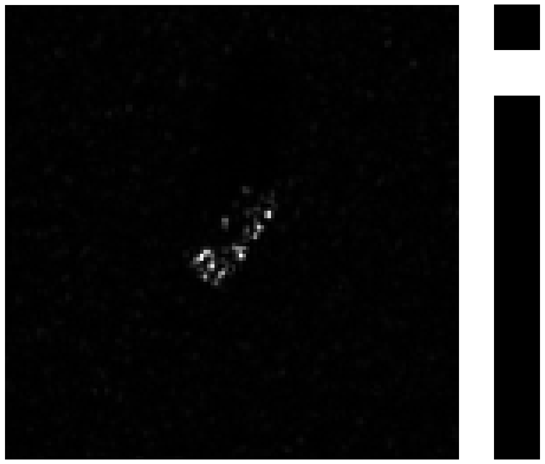

Figure 6. Input (left) and output (right) layer activations for a BTR60 SAR image. The 128×128 input layer activations are set by the raw magnitudes whilst the output layer activations are those of the 10×1 softmax layer.

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

REFERENCES

[1] Arel, I., Rose, D. C., and Karnowski, T. P., "Research frontier: Deep machine learning–a new frontier in artificial intelligence research," *Comp. Intell. Mag.* **5**, 13–18 (Nov. 2010).

[2] Hinton, G. E., Osindero, S., and Teh, Y.-W., "A fast learning algorithm for deep belief nets," *Neural Comput.* **18**, 1527–1554 (July 2006).

[3] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," in [*Advances in Neural Information Processing Systems 25*], Pereira, F., Burges, C., Bottou, L., and Weinberger, K., eds., 1097–1105, Curran Associates, Inc. (2012).

[4] Bengio, Y., Courville, A., and Vincent, P., "Representation learning: A review and new perspectives," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **35**, 1798–1828 (Aug 2013).

[5] Bishop, C. M., [*Neural Networks for Pattern Recognition*], Oxford University Press, Inc., New York, NY, USA (1995).

[6] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE* **86**, 2278–2324 (Nov 1998).

[7] Cui, J., Gudnason, J., and Brookes, M., "Automatic recognition of mstar targets using radar shadow and superresolution features," in [*Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*], **5**, v/589–v/592 Vol. 5 (March 2005).

[8] Pham, Q. H., Ezekiel, A., Campbell, M. T., and Smith, M. J. T., "New end-to-end sar atr system," *Proc. SPIE* **3721**, 292–301 (1999).

[9] Schumacher, R. and Schiller, J., "Non-cooperative target identification of battlefield targets - classification results based on sar images," in [*Radar Conference, 2005 IEEE International*], 167–172 (May 2005).

[10] Ross, T. D., Worrell, S. W., Velten, V. J., Mossing, J. C., and Bryant, M. L., "Standard sar atr evaluation experiments using the mstar public release data set," *Proc. SPIE* **3370**, 566–573 (1998).

[11] O'Sullivan, J., DeVore, M., Kedia, V., and Miller, M., "Sar atr performance using a conditionally gaussian model," *Aerospace and Electronic Systems, IEEE Transactions on* **37**, 91–108 (Jan 2001).

[12] Srinivas, U., Monga, V., and Raj, R., "Sar automatic target recognition using discriminative graphical models," *Aerospace and Electronic Systems, IEEE Transactions on* **50**, 591–606 (January 2014).

[13] Zhao, Q. and Principe, J., "Support vector machines for sar automatic target recognition," *Aerospace and Electronic Systems, IEEE Transactions on* **37**, 643–654 (Apr 2001).

[14] Sun, Y., Liu, Z., Todorovic, S., and Li, J., "Adaptive boosting for sar automatic target recognition," *Aerospace and Electronic Systems, IEEE Transactions on* **43**, 112–125 (January 2007).

[15] Masci, J., Meier, U., Cirean, D., and Schmidhuber, J., "Stacked convolutional auto-encoders for hierarchical feature extraction," in [*Artificial Neural Networks and Machine Learning ICANN 2011*], Honkela, T., Duch, W., Girolami, M., and Kaski, S., eds., *Lecture Notes in Computer Science* **6791**, 52–59, Springer Berlin Heidelberg (2011).

[16] Scherer, D., Mller, A., and Behnke, S., "Evaluation of pooling operations in convolutional architectures for object recognition," in [*Artificial Neural Networks ICANN 2010*], Diamantaras, K., Duch, W., and Iliadis, L., eds., *Lecture Notes in Computer Science* **6354**, 92–101, Springer Berlin Heidelberg (2010).

[17] Patnaik, R. and Casasent, D., "Sar classification and confuser and clutter rejection tests on mstar ten-class data using minace filters," *Proc. SPIE* **6574**, 657402–657402–15 (2007).

[18] Singh, R. and Vijaya Kumar, B., "Performance of the extended maximum average correlation height (emach) filter and the polynomial distance classifier correlation filter (pdccf) for multiclass sar detection and classification," *Proc. SPIE* **4727**, 265–276 (2002).

[19] Zhao, Q., Principe, J. C., Brennan, V. L., Xu, D., and Wang, Z., "Synthetic aperture radar automatic target recognition with three strategies of learning and representation," *Optical Engineering* **39**(5), 1230–1244 (2000).

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

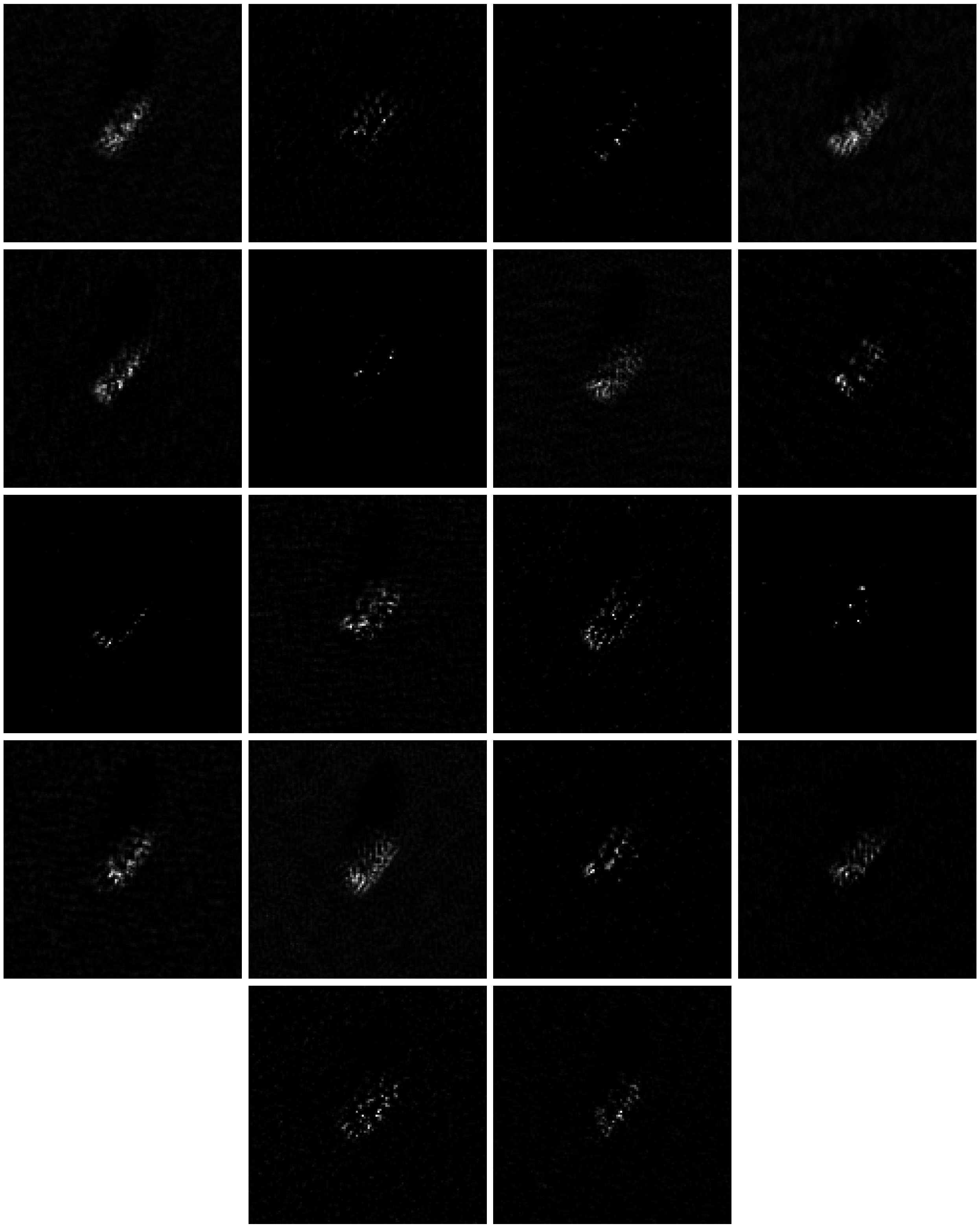

Figure 7. Feature map activations in the first convolutional layer for the trained CNN given the input from Figure 6.

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx

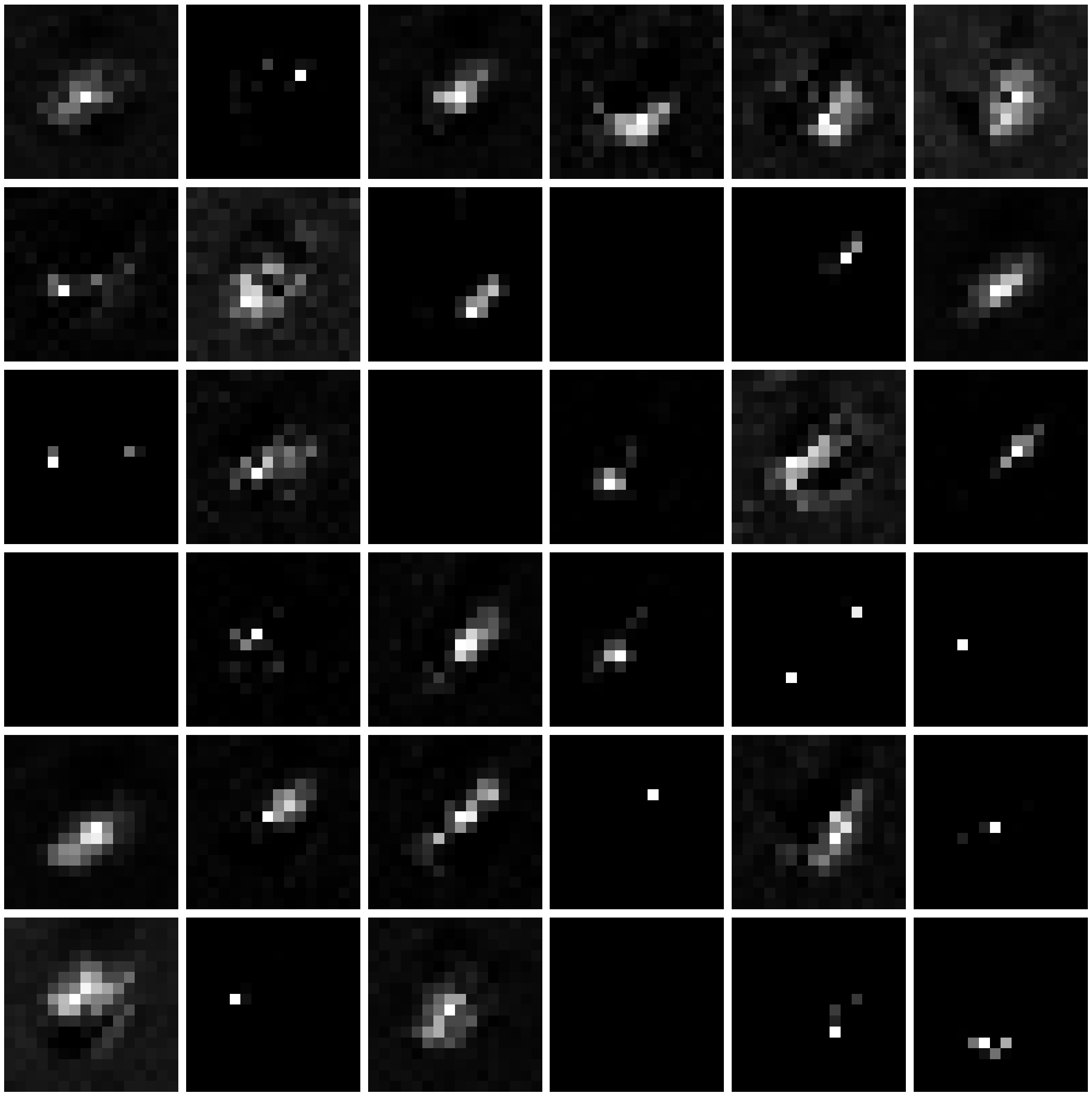

Figure 8. Feature map activations in the second convolutional layer for the trained CNN given the input from Figure 6.

Downloaded From: http://proceedings.spiedigitallibrary.org/ on 12/21/2016 Terms of Use: http://spiedigitallibrary.org/ss/termsofuse.aspx