Deep Learning for Radar

Eric Mason
Department of Electrical, Computer and Systems Engineering,
Rensselaer Polytechnic Institute
110 8th Street, Troy, NY 12180 USA
Email: masone2@rpi.edu

Bariscan Yonel
Department of Electrical, Computer and Systems Engineering
Rensselaer Polytechnic Institute
110 8th Street, Troy, NY 12180 USA
Email: yonelb@rpi.edu

Birsen Yazici
Department of Electrical, Computer and Systems Engineering
Rensselaer Polytechnic Institute
110 8th Street, Troy, NY 12180 USA
Email: yazici@ecse.rpi.edu

***Abstract*—Motivated by the recent advances in deep learning, we lay out a vision of how deep learning techniques can be used in radar. Specifically, our discussion focuses on the use of deep learning to advance the state-of-the-art in radar imaging. While deep learning can be directly applied to automatic target recognition (ATR), the relevance of these techniques in other radar problems is not obvious. We argue that deep learning can play a central role in advancing the state-of-the-art in a wide range of radar imaging problems, discuss the challenges associated with applying these methods, and the potential advancements that are expected. We lay out an approach to design a network architecture based on the specific structure of the synthetic aperture radar (SAR) imaging problem that augments learning with traditional SAR modelling. This framework allows for capture of the non-linearity of the SAR forward model. Furthermore, we demonstrate how this process can be used to learn and compensate for trajectory based phase error for the autofocus problem.**

I. Introduction

Deep learning has dramatically advanced the state-of-the-art for many challenging problems in different domains of science. These include record breaking performance in speech recognition, visual object recognition, natural language understanding and many others [1].

This technology is now adapted by Google's AI Brain, Microsoft's Deep Learning Technology Center and IBM's Watson project, with aspirations of developing revolutionary applications and products for healthcare, artificial intelligence, robotics, speech/language understanding and more [2]–[4]. The widespread popularity and success of deep learning is due to increase in computing power, availability of big data sets and its ability to process data with little upfront engineering.

Deep learning falls into a class of machine learning methods called representation learning. These methods attempt to create representations and models from large scale unprocessed data. With deep learning, features needed for detection or classification can be discovered automatically, bypassing the need for engineering domain expertise in hand crafting features [1], [5]. Some of the representations in deep learning are inspired by neuroscience and are based on information processing and communication patterns in the nervous systems.

Deep learning algorithms can be characterized by a cascade of many layers of non-linear processing units for feature extraction and transformation. Each successive layer extracts a representation of data at a different level of abstraction forming a hierarchy from low-level to high-level features. Varying numbers of layers and layer sizes are used to provide different degrees of abstraction [1].

There are various deep learning architectures such as deep feed-forward neural networks, convolutional deep neural networks, deep belief networks and recurrent neural networks [1], [6]. Deep feed-forward neural networks are the most general architecture, with all nodes in subsequent layers being interconnected to each other. These networks are generally explained in terms of the universal approximation theorem or probabilistic inference [7], [8].

Although deep learning has become an exciting trend over the last couple of years, its full potential as a signal processing tool for a broad range of applications has yet to be realized. In [9], [10] deep learning has been explored for sparse inverse problems. In [9], the iterated shrinkage thresholding algorithm is used as an architecture for a deep recurrent neural network. This idea is extended to the approximate message passing algorithm, showing improved performance in [10]. In this paper, we lay out a vision to explore the potential of deep learning in radar signal processing. Specifically, we consider challenging estimation-detection problems in radar imaging involving non-linearities. Estimation-detection problems are often cast as optimization problems in which practical considerations require adopting the use of linear forward models. Motivated by the capability of neural networks as universal approximators, we lay out a framework and propose a network architecture to address non-linear inverse problems in radar imaging. With a sufficient amount of training data and initializing its parameters with the linearized forward model, the neural network can be used to iteratively refine the measurement model to capture non-linearities of the underlying physical model. We identify several such problems in radar that can be posed and addressed within our framework.

One such class of problems are the ones that involves phase error or unknowns in the radar forward model. Phase errors occur for many reasons and lead to distortions in the reconstructed image. The most notable causes are moving targets and antenna trajectory location error [11]–[13].

The rest of the paper is organized as follows. In Section II, we provide an overview of the basic tenets of deep learning from a perspective relevant to our subsequent discussion. In Section III, we present radar problems that can be advanced by deep learning. In Section IV, we state the radar imaging

978-1-4673-8823-8/17/$31.00 ©2017 IEEE

problem and its solution in the deep learning framework. In section V we present numerical simulations with promising results motivating further investigation into deep learning for radar. Section VI concludes our paper.

II. Deep Learning

Deep feed-forward neural networks are the most general architecture for deep learning. Deep networks differ from standard Artificial Neural Networks (ANN) in terms of their "depth". Even though the universal approximation theorem states that a single hidden layer neural network is a universal approximator, the depth of the architecture introduces new strengths and capabilities to the network [1]. A deep architecture trades off the load of parallel computations with sequential computations [14]. As a result, deep networks gain more expressive power and can represent more complex functions between input and output spaces due to their nested non-linear mapping of the data space.

An ANN is a network of interconnected nodes called "neurons" as shown in figure 1. At each neuron, the inner product of the data with a weight vector, plus bias, is fed as the input. Hence, the output at each layer of the neural network can be characterized as

$$\tilde{\mathbf{x}}^{(k+1)} = \sigma(\tilde{W}^{(k)} \cdot \tilde{\mathbf{x}}^{(k)} + \mathbf{w_b^{(k)}}) \tag{1}$$

where $\tilde{\mathbf{x}}^{(k)} \in \mathbb{R}^n$ is the input vector of the k^{th} layer of the network, $\tilde{W}^{(k)} \in \mathbb{R}^{m \times n}$ is the weight matrix of k^{th} layer, $\mathbf{w_b^{(k)}} \in \mathbb{R}^m$ is the bias weight vector of k^{th} layer, $\sigma(.)$ is an element wise non-linear activation function of the neurons, and $\tilde{\mathbf{x}}^{(k+1)} \in \mathbb{R}^m$ is the output of the k^{th} layer, and subsequently the input of $(k+1)^{\text{th}}$ layer of the network. Augmenting the bias vector $\mathbf{w}^{(k)}$ to the weight matrix $\tilde{W}^{(k)}$ to form $W^{(k)}$, and $\tilde{\mathbf{x}}$ with 1 to form $\mathbf{x}$, the final output obtained for an N-layer network is given as:

$$\mathbf{y}^* = \sigma(W^{(N)} \cdot \sigma(W^{(N-1)} ... \cdot \sigma(W^{(1)} \cdot \sigma(W^{(0)} \cdot \mathbf{x}))\big), \tag{2}$$

which defines forward propagation, and provides an analytical representation of the ANN.

The network parameters $W^{(N)}, W^{(N-1)}, \cdots W^1, W^0$ are "learned" by optimizing some criteria defined in terms of a mismatch between the true and reconstructed signal. The optimization is carried out using first and second order iterative optimization methods. The analytic method of calculating the derivative of the objective functional given in equation (3) with respect to trainable parameters, W^k, is referred to as the *backpropagation* algorithm. A typical objective functional used for training ANN is the ℓ^2-norm of the mismatch between the network output $\mathbf{y}^*$ and the desired output $\mathbf{y}$. For a single training data sample $\mathbf{x}$ and desired value $\mathbf{y}$ the objective function using this error criteria is

$$\min_{W^N, W^{N-1}, ... W^0} ||\mathbf{y}^*(W^N, W^{N-1}, ..., W^0, \mathbf{x}) - \mathbf{y}||_2^2. \tag{3}$$

This optimization problem is typically a high-dimensional non-convex problem, often consisting of many saddle points and non-optimal local minima [1], [6], [15]. As a result, a critical aspect is the initialization of the network weights W^k. Therefore, prior knowledge of the input to output mapping should be utilized in initialization to increase the likelihood of obtaining a good solution.

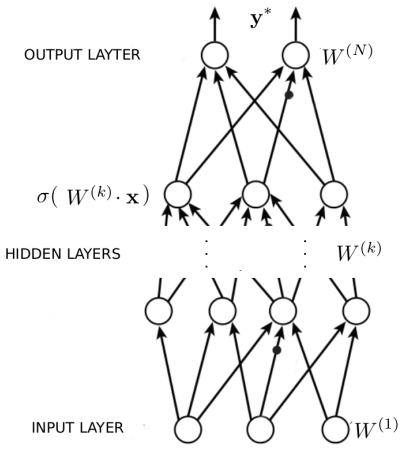

Fig. 1: A fully connected N-layer feed-forward neural network. Figure adapted from [6].

Several different deep network architectures have been investigated in the literature each performing well for different applications. Most notably, these architectures can be categorized as deep Feedforward Neural Networks (FFNNs), Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNN) [1], [5], [6]. In a multi layer FFNN architecture, all nodes in a lower level layer are connected to each node of the next layer. An example of this architecture is given in figure 1. In a CNN, the layers are more sparsely connected than the standard ANN architecture. The weighting between two layers is formulated as a convolution with a kernel much smaller than the number of nodes in the layer. This architecture significantly reduces the computation load in the network [1], [6]. CNN's are mostly preferred for computer vision tasks such as face recognition or image classification due to grid structure of the input data, as well as, the sparse and more localized nature of interactions between layers [6].

An RNN is formulated as a deep feedforward neural network by unfolding a recurring state update, where each layer of the network represents an iteration. This sharing of parameters amongst layers differentiates RNN's from other architectures [6]. Since the weights between every two layers of the network are identical, only a single weight matrix must be learned during training. As a result, an RNN is similar to an iterative process, where the latent parameters (weights) are learned directly from the training data.

III. Deep Learning and Radar Problems

The field of radar encompasses a broad range of problems. We group the radar problems that can utilize deep learning into three general categories: sensing, radar signal processing (RSP), and automatic target recognition (ATR) problems. This dichotomy is shown in figure 2 and some applications under

978-1-4673-8823-8/17/$31.00 ©2017 IEEE

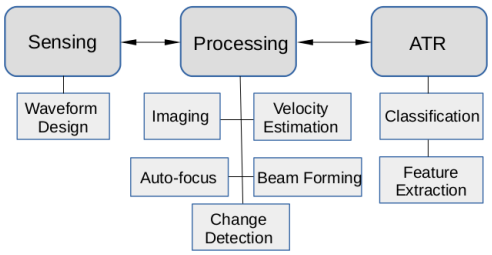

Fig. 2: Radar problems that can benefit from deep learning.

each category are listed. The most straightforward application of deep learning to radar is within the area of SAR ATR [16]–[18]. Different architectures have been widely studied for the purpose of SAR ATR feature extraction and classification. The CNN has proven to be one of the most successful architectures for classification. In [19] accuracy results averaging 99% were reported on the MSTAR data set. In [20], the auto-encoder architecture is used for feature extraction from SAR imagery.

While the relevance and applications of deep learning to SAR ATR tasks are straightforward, potential benefits and applications of deep learning to the problems of sensing and RSP are not immediately apparent. Here we argue that the deep learning framework can play an integral role in bridging sensing, radar signal processing and ATR problems, because all of these problems involve optimization methods with limiting assumptions made on the underlying models. In radar imaging the received data is a mapping of the scene reflectivity function (can include velocity) by a forward model $\mathcal{F}_\alpha$ to the received signal, given by

$$d = \mathcal{F}_\alpha[\rho + \rho_c] + n \tag{4}$$

where n and ρ_c denote noise and clutter, respectively. $\mathcal{F}_\alpha$ is typically a linear model based on the underlying physics of wave propagation and α denotes known or unknown parameters [21]–[24]. The radar problems listed in figure 2 are all related to measuring, estimating, or modelling different terms in (4). Sensing is the area related to obtaining desired measurements d. ATR research focuses on making decisions based on the estimated scene ρ. Modelling of $\mathcal{F}_\alpha$, its inversion, and estimation of α along with noise/clutter suppression are problems in RSP. Note that the problems in each category are coupled and effective solutions require addressing the problems in sensing, processing and decisions making jointly.

Data processing in radar largely relies on limiting assumptions and linear models. We posit that given a sufficient amount of training data the neural network can more accurately approximate the relationship between measurements and unknown quantities of interest.

As an example, consider the commonly used Born approximation which relies on single scattering assumption. This approximation is needed because the measured scattered field is an integral transform of the total field making the relationship between the reflectivity and measured signal non-linear. There are many applications in which capturing multiple scattering within target may lead to improved sensing, imaging and decision making. These include imaging with limited bandwidth in which classical techniques have to rely on a single scattering event within a "cell", foliage penetrating radar in which scattering phenomena is "volumetric", and imaging of large scenes using multiple scattering [25], [26], to mention a few.

Other radar problems that may benefit from the deep learning framework include those that can be posed as bilinear inverse problems such as waveform design [27]–[29] and other non-linear problems such as autofocus [11], [30], velocity and position estimation [12], [13], [31]–[35], three dimensional SAR imaging and adaptive radar problems. Our approach offers a framework and guides the use of deep learning for these applications.

IV. Deep Learning for Radar Imaging

A. Network architecture

Our approach exploits underlying optimization problems in radar to design network architectures, and uses unsupervised learning methods for ANNs to numerically learn parameters. To do so, we first design a network that imitates the image reconstruction process from measured data. For this purpose, RNNs provide the most suitable architecture to our vision for radar applications. Specifically, RNN's are obtained by unfolding iterative processes where each layer corresponds to an iteration step. This structure makes RNNs a natural choice for the ANN architecture.

By formulating a layer of the RNN as one step of an iterative reconstruction algorithm, such as the iterative shrinkage thresholding algorithm (ISTA), we can implement the image formation algorithm as forward propagation of the network. When the iterative scheme is unfolded for a fixed number of iterations, we obtain the network architecture provided in figure 3, where a gradient descent step followed by a point-wise shrinkage operation is applied at each layer. Moreover, in SAR imaging the ground truth is often not available to use in training. We overcome this limitation by designing a recurrent auto-encoder [36]. Auto-encoders are ANNs that learn a mapping of a space to itself [6]. They are constrained to learn the composition of two mappings so that the desired information is obtained midway through the network. We obtain this architecture by adding an additional layer that maps the estimated image back to data, allowing for unsupervised training. This final mapping is a linear stage at the output layer of our architecture, as displayed in figure 3.

Given the data model (4) an estimate of the scene reflectivity function can be obtained by minimizing the following functional with respect to ρ:

$$\mathcal{J}_1[\rho] = \frac{1}{2}\|\mathcal{F}_\alpha[\rho] - d_n\|_2^2 + \lambda_\rho f(\rho), \tag{5}$$

where λ_ρ is a regularization parameter. Minimization of (5) can be accomplished using the following iteration:

$$\rho_{k+1} = \mathcal{P}_f\left(\mathcal{K}[\rho_k] + b\right) \tag{6}$$

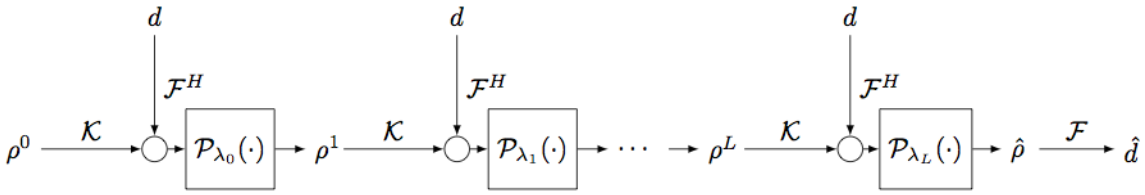

Fig. 3: The proposed network architecture. Linear stages $\mathcal{K}, \mathcal{F}$ are represented as arrows, whereas non-linear stages are represented as blocks that perform point-wise shrinkage operator with parameter λ_i.

where $\mathcal{K} = (\mathcal{I} - \beta\mathcal{F}_\alpha^\dagger\mathcal{F}_\alpha)$, $b = \beta\mathcal{F}_\alpha^\dagger[d]$, $\mathcal{I}$ is a bandlimited identity operator, and $\mathcal{P}_f$ is the proximity operator for f. The sequence formed by a fixed number of iterations given by (6), is then unfolded into an RNN, where the output is $\rho(\tilde{\theta})$ where $\tilde{\theta} = [\mathcal{K}, \beta\mathcal{F}_\alpha^\dagger, \lambda]$ are the trainable parameters of the network. The network can be trained using backpropagation-through-time (BPTT) to compute the gradient in conjunction with a stochastic gradient descent [6].

The regularization term depends on prior knowledge about the unknown function ρ. Under our framework it is necessary to use a regularization function with a closed form solution for its proximity operator, as it serves as the non-linear activation function.

B. Network Training

Given a training set $D = \{d_1, \ldots, d_N\}$, the training procedure can be viewed as minimizing the functional

$$\mathcal{J}_2[\theta, \rho] = \frac{1}{2N}\sum_{n=1}^{N} \|\mathcal{L}[\mathcal{F}_\alpha, \rho(\tilde{\theta})] - d_n\|_2^2 + \lambda_\theta g(\theta), \qquad (7)$$

with respect to the network parameters $[\mathcal{F}_\alpha, \tilde{\theta}]$. In (7) the network is $\mathcal{L}[\mathcal{F}_\alpha, \rho(\tilde{\theta})] = \mathcal{F}_\alpha[\rho(\tilde{\theta})]$. For notiational brevitity, we collect the trainable parameters in the vector $\theta = [\mathcal{F}_\alpha, \tilde{\theta}]$.

In the simplest case the minimization of (7) can be carried out using a stochastic gradient descent. Forward propagation is done by performing a fixed number of iterations given in (6), keeping θ fixed for each training sample d_n. This is equivalent to minimization of the objective functional (5). The parameter θ is updated using one step of a gradient descent for minimizing (7), where the gradients are calculated using BPTT. The regularization function g and the regularization parameter λ_θ are chosen based on prior knowledge or to achieve a desired structure in the learned parameters. For example, it may be desirable to impose sparsity in $\mathcal{K}$ to meet computational complexity and memory requirements of the trained network.

We summarize the training process as follows:

1) $\rho_L = \underset{\rho}{\text{Argmin}}\ \mathcal{J}_1[\rho]$
2) $\theta_{k+1} \leftarrow \theta_k - \eta \mathrm{E}_{\theta|D}[\nabla_\theta \mathcal{J}_2[\theta, \rho_L]]$

where η is typically referred to as the learning rate, $\nabla_\theta \mathcal{J}_2$ denotes the gradient of the discretized objective functional, L is the number of layers and $\mathrm{E}_{\theta|D}$ denotes statistical expectation over the data set. The first step of training is the forward propagation which amounts to solving (5) for a fixed number of iterations, determined by the depth of the network. The second iteration performs a step of an online stochastic gradient descent.

The training procedure could also be implemented by a "batch" gradient descent algorithm rather than a stochastic gradient descent. The main difference arises in the amount of data used to determine the step size at each epoch. Stochastic gradient descent offers computational advantages since only a random subset of data samples are used at each epoch, referred to as mini-batches. Whereas in batch gradient descent, the weight update is computed using the entire data set, which typically results in a smoother decrease in the objective value.

The training of RNNs requires extensive parameter tuning. Due to having the identical weight matrix at every layer of the network, a commonly encountered problem is exploding or diminishing gradients [37]. This phenomenon occurs in cases when the weight matrix is poorly conditioned. We address this issue by choosing a relatively small learning rate in our experiments, as is typically done in the literature.

V. Numerical Experiments

We consider the autofocus problem and apply the deep learning framework introduced in the previous sections to form focused SAR images from data formed with erroneous antenna trajectories. We present numerical simulations and display promising results.

We use a $15.5 \times 15.5\text{m}^2$ scene with a single stationary point target discretized into 31×31 pixels with the origin located at the center. A bistatic radar signal was generated using the forward model described in [11]. The two antennas traverse the same circular trajectory, defined as $\boldsymbol{\gamma}_R(s) = [11\cos(s), 11\sin(s), 6.5]\text{km}$ and $\boldsymbol{\gamma}_T(s) = \boldsymbol{\gamma}_R(s + \pi/2)$. The bandwidth and center frequency are 124MHz and 9.6GHz, respectively. We sampled in slow-time and fast-time uniformly to obtain 400 and 100 samples, respectively. We generated a training set of size 1000 by generating data for a single point target at a random position. Furthermore, different realizations of trajectory error were also added for each datum, sampled from a uniform distribution with variance 0.003m. A test set of 30 measurements was created in the same way.

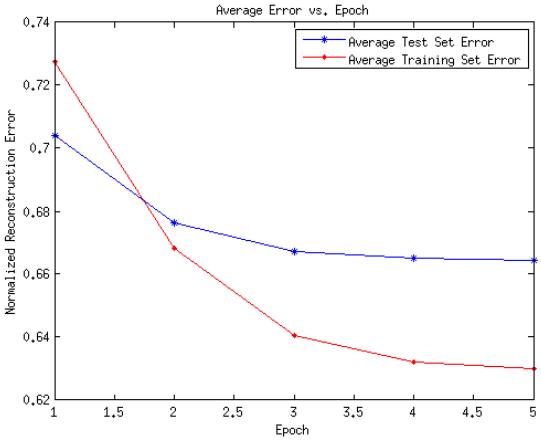

Fig. 4: The average error over the training (red) and test (blue) set at each epoch.

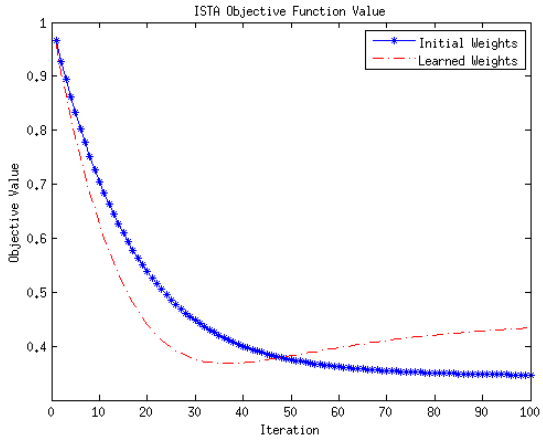

Fig. 5: The objective function values for 100 iterations of the iterative shrinkage algorithm. The algorithm is ran for a single point target in the test set. The algorithm with the unlearned parameters is shown in blue, and the learned parameters in red.

We constructed an 8 layer network using the architecture described in (6) in which the non-linear thresholding is the shrinkage operator commonly associated with ISTA [38]. We trained the network by solving (7) with no regularization, using a mini-batch (of 20) stochastic gradient descent. We initialized the learning rate $\eta = 1e^{-10}$ and decreased it at each mini-batch update. This was done at a rate of $\eta = \eta/(1+k)$ where $k = 0, 1, \ldots$ indexes the epoch.

In our experiments we trained for 5 epochs at which point the algorithm converged. In our experiments we considered the normalized ℓ^2-norm error, equivalent to the objective functional value. Figure 4 shows a plot of the average error for the training and test set at each epoch. At every epoch we observed that the average reconstruction error decreased, this can partially be attributed to decreasing the learning rate at each update.

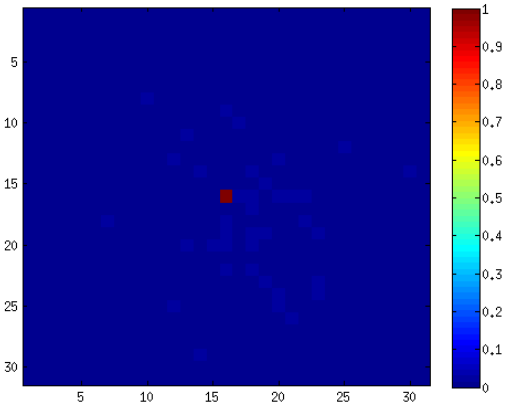

Fig. 6: The reconstructed image for 100 iterations of ISTA using the unlearned weights.

For visual demonstration of performance we consider a test image consisting of a single point target at pixel $(16, 16)$. Due to the fact that ISTA typically requires many hundreds of iterations to converge and a network of this depth is not practical, we investigate the performance of ISTA when used with the trained weights for 100 iterations. Figure 5 shows the reconstruction error per iteration. We see that the algorithm with the learned weights converges more rapidly and result in a smaller reconstruction error, at least for the depth of the network. Similar observations have been made for different architectures and training procedures in other inverse problems [10]. For iterations above 30 we observe that ISTA with the learned weights no longer monotonically decreases. While we cannot explain this behaviour we expect it relates to how the angle between columns of the weight matrix change during training. Figures 6 and 7 display the images formed at the final iteration of ISTA using the unlearned and learned weights, respectively. We see that the image formed using the learned weights display better suppression of spurious, non-zero, background components, despite having a larger objective value. Even more surprising is that the images produced by a hundred iterations of ISTA with the learned weights is visually better than those obtained at the iteration of minimum error (iteration 32).

VI. Conclusion

In this paper, we lay out our vision for addressing a wide range of radar problems in the deep learning framework. Specifically, we show that deep learning offers a unifying framework to address non-linear inverse problems and linear problems with unknown parameters/terms in the forward model. We propose an RNN architecture to solve such problems, and an accompanying method to train it. We argue that deep learning offers a unifying framework to integrate sensing, processing and decision making. We apply our approach to the autofocus problem in SAR imaging and present numerical simulations to demonstrate the potential deep learning offers.

978-1-4673-8823-8/17/$31.00 ©2017 IEEE

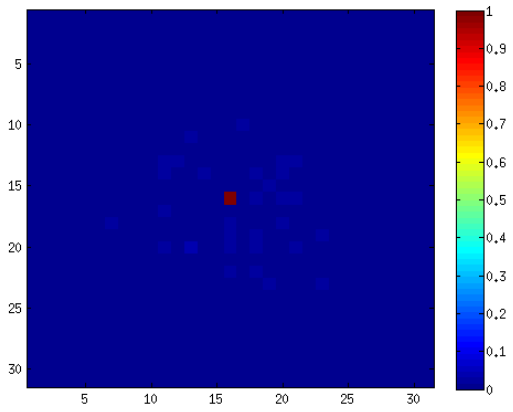

Fig. 7: The reconstructed image for 100 iterations of ISTA using the learned weights.

ACKNOWLEDGMENT

This work was supported by the Air Force Ofce of Scientic Research (AFOSR) under the agreements FA9550-16-1-0234, and by the National Science Foundation (NSF) under Grant Nos. CCF-1421496 and 60027860.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] "Microsoft's deep learning technology center," https://www.microsoft.com/en-us/research/group/dltc/, accessed: 11-30-2016.

[3] "Google's AI brain," http://research.google.com/teams/brain/, accessed: 11-30-2016.

[4] "Ibm's watson project," http://www.ibm.com/watson/, accessed: 11-30-2016.

[5] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug 2013.

[6] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," 2016, book in preparation for MIT Press. [Online]. Available: http://www.deeplearningbook.org

[7] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251 – 257, 1991.

[8] G. F. Cooper, "The computational complexity of probabilistic inference using bayesian belief networks," *Artificial intelligence*, vol. 42, no. 2-3, pp. 393–405, 1990.

[9] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," pp. 399–406, 2010.

[10] M. Borgerding and P. Schniter, "Onsager-corrected deep learning for sparse linear inverse problems," *arXiv preprint arXiv:1607.05966*, 2016.

[11] L. Wang, B. Yazici, and H. C. Yanik, "Antenna motion errors in bistatic sar imagery," *Inverse Problems*, vol. 31, no. 6, p. 065001, 2015.

[12] S. Wacks and B. Yazıcı, "Passive synthetic aperture hitchhiker imaging of ground moving targetspart 2: Performance analysis," *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 4126–4138, 2014.

[13] S. Wacks and B. Yazici, "Passive synthetic aperture hitchhiker imaging of ground moving targetspart 1: Image formation and velocity estimation," *IEEE Transactions on Image Processing*, vol. 23, no. 6, pp. 2487–2500, 2014.

[14] Y. Bengio and Y. LeCun, "Scaling learning algorithms towards AI," *Large-scale kernel machines*, vol. 34, no. 5, 2007.

[15] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *Advances in neural information processing systems*, 2014, pp. 2933–2941.

[16] H. Wang, S. Chen, F. Xu, and Y.-Q. Jin, "Application of deep-learning algorithms to MSTAR data," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2015, pp. 3743–3745.

[17] D. A. Morgan, "Deep convolutional neural networks for ATR from SAR imagery," in *SPIE Defense+ Security*. International Society for Optics and Photonics, 2015, pp. 94 750F–94 750F.

[18] M. Wilmanski, C. Kreucher, and J. Lauer, "Modern approaches in deep learning for SAR ATR," in *SPIE Defense+ Security*. International Society for Optics and Photonics, 2016, pp. 98 430N–98 430N.

[19] S. Chen, H. Wang, F. Xu, and Y. Q. Jin, "Target classification using the deep convolutional networks for SAR images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4806–4817, Aug 2016.

[20] Z. Sun, L. Xue, and Y. Xu, "Recognition of SAR target based on multilayer auto-encoder and SNN," *International Journal of Innovative Computing, Information and Control*, vol. 9, no. 11, pp. 4331–4341, 2013.

[21] L. Wang and B. Yazici, "Bistatic synthetic aperture radar imaging using narrowband continuous waveforms," *Image Processing, IEEE Transactions on*, vol. 21, no. 8, pp. 3673–3686, Aug 2012.

[22] C. Yarman and B. Yazici, "Synthetic aperture hitchhiker imaging," *IEEE Transactions on Imaging Processing*, pp. 2156–2173, 2008.

[23] V. Krishnan, J. Swoboda, C. Yarman, and B. Yazc, "Multistatic synthetic aperture radar image formation," *Image Processing, IEEE Transactions on*, vol. 19, no. 5, pp. 1290–1306, May 2010.

[24] L. Wang and B. Yazici, "Bistatic synthetic aperture radar imaging of moving targets using ultra-narrowband continuous waveforms," *SIAM Journal on Imaging Sciences*, vol. 7, no. 2, pp. 824–866, 2014.

[25] L. Wang, I.-Y. Son, and B. Yazici, "Passive imaging using distributed apertures in multiple-scattering environments," *Inverse Problems*, vol. 26, no. 6, p. 065002, 2010.

[26] X. Lv, B. Yazici, M. Zeghal, V. Bennett, and T. Abdoun, "Joint-scatterer processing for time-series InSAR," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 11, pp. 7205–7221, 2014.

[27] Y. Luo, Q. Zhang, W. Hong, and Y. Wu, "Waveform design and high-resolution imaging of cognitive radar based on compressive sensing," *Science China Information Sciences*, vol. 55, no. 11, pp. 2590–2603, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11432-011-4527-x

[28] M. C. Shastry, "Compressively sampled radar using random waveforms," Ph.D. dissertation, The Pennsylvania State University, 2013.

[29] B. Yazici and G. Xie, "Wideband extended range-Doppler imaging and waveform design in the presence of clutter and noise," *IEEE transactions on information theory*, vol. 52, no. 10, pp. 4563–4580, 2006.

[30] S. Uur and O. Arıkan, "Sar image reconstruction and autofocus by compressed sensing," *Digital Signal Processing*, vol. 22, no. 6, pp. 923–932, 2012.

[31] E. Mason, I. Y. Son, and B. Yazici, "Passive synthetic aperture radar imaging using low-rank matrix recovery methods," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 8, pp. 1570–1582, Dec 2015.

[32] E. Mason and B. Yazici, "Moving target imaging using sparse and low-rank structure," in *SPIE Defense+ Security*, vol. 9843. International Society for Optics and Photonics, 2016.

[33] L. Wang and B. Yazici, "Passive imaging of moving targets using sparse distributed apertures," *SIAM Journal on Imaging Sciences*, vol. 5, no. 3, pp. 769–808, 2012.

[34] ——, "Passive imaging of moving targets exploiting multiple scattering using sparse distributed apertures," *Inverse Problems*, vol. 28, no. 12, p. 125009, 2012.

[35] ——, "Bistatic synthetic aperture radar imaging of moving targets using ultra-narrowband continuous waveforms," *SIAM Journal on Imaging Sciences*, vol. 7, no. 2, pp. 824–866, 2014.

[36] J. T. Rolfe and Y. LeCun, "Discriminative recurrent sparse auto-encoders," *arXiv preprint arXiv:1301.3775*, 2013.

[37] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks." *ICML (3)*, vol. 28, pp. 1310–1318, 2013.

[38] P. Combettes and J. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, Eds. New York: Springer, 2011.