LMS TO DEEP LEARNING: HOW DSP ANALYSIS ADDS DEPTH TO LEARNING

Paul Gorday, Nurgün Erdöl, Hanqi Zhuang

Dept. Electrical & Computer Engineering & Computer Science, Florida Atlantic University
Email: pgorday@fau.edu, erdol@fau.edu, zhuang@fau.edu

ABSTRACT

Difficulty in analyzing deep learning systems is preventing association of its parameters and state outputs with elements that are derived from theoretic considerations. Medical and criminal justice communities are excited by the possibilities nevertheless reluctant to adopt machine learning for fear of errors and bias. There is also concern among educators that mystifying learning may have long-term adverse pedagogical implications on human learning. Deep learning techniques have not received enthusiastic attention in the realm of communication systems, either. This is in part due the effectiveness of traditional analytical solutions that render classification error rates larger that 10^{-3} unacceptable. Understanding of deep networks for communications, and ability to perform comparison tests can be useful in developing scalable methods to understand them in more complex scenarios. This paper contributes a perspective and analysis with focus on Nyquist and non-Nyquist pulse shapes for bandlimited channels. It is hoped that this fundamental presentation motivates wider consideration of neural networks and deep learning for demodulation.

Index Terms— LMS, DSP, demodulation, neural network, deep learning.

1. INTRODUCTION

Over the past few decades, adaptive signal processing has broadened to include learning systems [1-4]. Although the terminology can become blurred, one suggested distinction is that adaptive systems update their parameters while performing their intended function, while learning systems adjust their parameters in advance during a training phase. Regardless of the definitions, it is clear that more and more systems are becoming data-centric – using adaptable-parameter topologies that "learn" their functionality directly from data. Traditional adaptive linear combiners have expanded to include neural networks and deep learning topologies, which contain multiple layers of linear combining separated by nonlinear activation functions. While deep learning techniques have demonstrated significant improvements in performance for many regression and classification problems, they are not without challenges. Perhaps one of the biggest concerns is the difficulty of analyzing and explaining their behavior. The "magic box" quality of deep learning is preventing association of its parameters and state outputs with elements that are derived from theoretic considerations. Medical and criminal justice communities are excited by the possibilities nevertheless reluctant to adopt machine learning for fear of errors and bias. There is also concern among educators that mystifying learning may have long-term adverse pedagogical implications on human learning. Research is being focused on this topic as more attention is being brought to the field [5].

In terms of adaptation, many of the new learning systems are still based on gradient descent techniques that closely resemble the original LMS algorithm. The popular backpropagation technique is a method for estimating the sensitivity (gradient) of the system cost function with respect to each weight in a neural network. Research has produced many variations of this technique, along with new activation functions and deeper architectures for which gradient descent performs efficiently [4].

In the three decades since the backpropagation technique was introduced, neural networks and deep learning have been embraced in applications like speech and image classification, where performance gains over traditional methods are significant. In fields like communications, where traditional solutions have proven very effective, adoption has been slower. Literature search in this area reveals a number of point-studies, in which researchers evaluate neural networks as replacements or enhancements to existing demodulator or equalizer algorithms (e.g., [6-13]). A helpful addition would be an elementary study that examines learning approaches for simple traditional demodulators.

This paper contributes such a perspective that the author would like to have read when first exploring this area. It focuses on several elementary neural network structures and compares them with traditional textbook demodulators. Feature representation is a key element of learning algorithms, and understanding the feature representation in these shallow networks can provide insights and suggest research directions for deeper networks. Furthermore, exploring simple systems that are mathematically tractable will help us with the characterization of learning techniques that are increasingly being viewed as a magic box.

2. COMMUNICATION SYSTEM MODEL

This study considers a baseband communication technique known as pulse-amplitude-modulation (PAM), which is often the starting point for evaluating bandlimited communications in many textbooks [14, 15]. Using discrete-time notation, a message waveform $m(n)$ is produced using a series of time-shifted symbol pulses, $p(n)$.

$$m(n) = \sum_{k=1}^{L} d_k p(n - kD) \quad (1)$$

The variable n represents the discrete-time sample index, and the integer D defines the symbol period in samples. A binary antipodal system is considered here, with independent symbol values from the set $d_k \in \{1, -1\}$. The message passes through an additive white Gaussian noise (AWGN) channel and is input to a neural network demodulator (or simply neural demodulator) as shown in Fig. 1. The demodulator input signal $x(n)$ is sampled at a

rate of D samples per symbol, and only one symbol decision is made for every D samples shifted into the delay line ($D = 4$ is used in subsequent simulations). Ideal synchronization is assumed such that the symbol being demodulated is centered within the span of the tapped delay line. The demodulator output $y(n)$ is down-sampled by D as shown in Fig. 1 to produce soft symbol decisions. During neural network training, the soft symbol decisions are used to compute backpropagation error (as described below). For demodulator performance evaluation, the soft symbol decisions are converted to hard decisions using a threshold value of zero.

Digital communication signals are typically constructed using basis expansions such as (1). The time-shifted pulses are basis functions and the symbol values are expansion coefficients. This study considers two common pulse shapes for $p(n)$ [16]. The first is a root-raised-cosine (RRC) pulse with rolloff factor of 25% and duration of 33 samples (~8 symbols). This is a member of the Nyquist family, which has the useful property that all symbol-spaced copies are orthogonal to each other. Therefore, the basis functions in (1) are orthogonal. The second symbol shape is the Gaussian-filtered (GF) pulse, which is produced by passing a rectangular symbol pulse through a filter with Gaussian impulse response. The specific GF pulse used here has time-bandwidth parameter $BT = 0.5$ and duration 13 samples (~4 symbols). Symbol-spaced copies of the GF pulse are not orthogonal to each other.

3. NETWORK TOPOLOGIES

This section presents several elementary neural demodulators based on a time-delay neural network (TDNN) topology. They are elementary in the sense that each contains a minimum number of neurons for its type. Each uses time-varying features of the input signal over a small window of time to classify received data symbols. Their structures resemble simple traditional demodulators, but they include nonlinear activation functions and their weights are adapted using supervised learning techniques similar to the LMS algorithm. Simulations in the subsequent section examine behavior and performance of these neural demodulators compared to similar traditional topologies.

The feed-forward TDNN (FF-TDNN) shown in Fig. 2 is a single-layer network suitable for binary signal classification. The N samples of received signal stored in delay line are represented by an N-dimensional input vector $\boldsymbol{x}$. The single neuron computes a linear combination of the input samples followed by a nonlinear activation function, f, as expressed in (2). Except for the activation function, this simple neural network resembles the discrete-time correlation detector (matched filter) used in many communication systems.

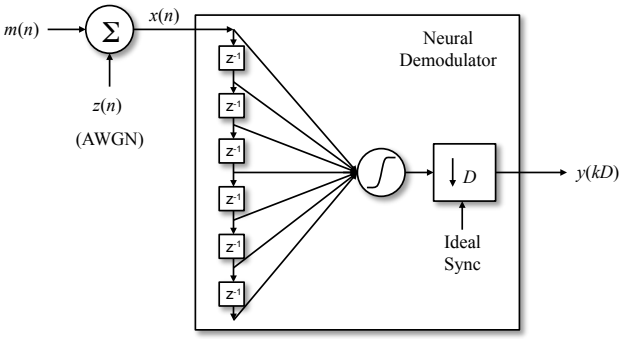

Fig. 1. System model.

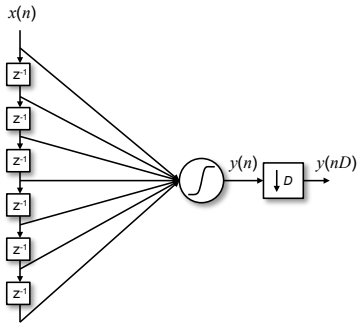

Fig. 2. Single-layer feed-forward TDNN demodulator.

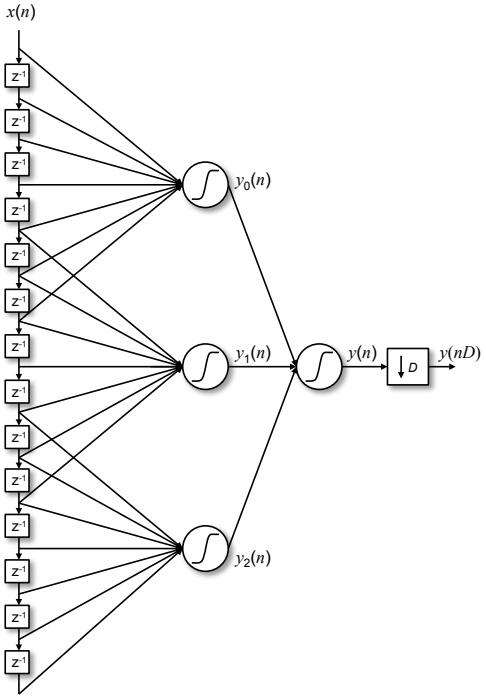

Fig. 3. Two-layer convolutional TDNN.

$$\boldsymbol{y}(n) = f(\boldsymbol{x}^T\boldsymbol{w}) \tag{2}$$

$$\boldsymbol{x} = \boldsymbol{x}(n) = \begin{bmatrix} x(n) \\ \vdots \\ x(n-N+1) \end{bmatrix}, \quad \boldsymbol{w} = \begin{bmatrix} w_0 \\ \vdots \\ w_{N-1} \end{bmatrix}$$

The activation function used throughout this study is the hyperbolic tangent, which is commonly used in shallow neural networks. We also omit neuron bias values due to the antipodal symmetry of the input data and the zero-mean additive noise considered here.

Fig. 3 shows a simple two-layer convolutional TDNN (C-TDNN) suitable for binary classification. The three neurons in the convolutional layer are not fully connected to the input delay line. Their N-tap receptive fields are offset by D samples, and all three neurons are constrained to use the same weights $\boldsymbol{w}_c$ and activation function, f. From a communications perspective, the convolutional network resembles a matched filter followed by a symbol-spaced equalizer.

The convolutional layer equations are expressed in (3), where the activation function operates element-wise on the product of the weight vector and input matrix $\boldsymbol{X}$.

$$\boldsymbol{y}_c = f(\boldsymbol{X}^T \mathbf{w}_c), \quad \mathbf{w}_c = \begin{bmatrix} \boldsymbol{w}_{c,0} \\ \vdots \\ \boldsymbol{w}_{c,N-1} \end{bmatrix} \tag{3}$$

$$\boldsymbol{X} = [\boldsymbol{x}(n) \quad \boldsymbol{x}(n-D) \quad \boldsymbol{x}(n-2D)]$$

$$\boldsymbol{x}(n) = \begin{bmatrix} x(n) \\ \vdots \\ x(n-N+1) \end{bmatrix}$$

The output neuron combines the results of the convolutional layer using weight vector $\boldsymbol{w}_o$ and activation function f.

$$\boldsymbol{y}(n) = f(\boldsymbol{y}_c^T \mathbf{w}_o), \quad \boldsymbol{w}_o = \begin{bmatrix} \boldsymbol{w}_{o,0} \\ \boldsymbol{w}_{o,1} \\ \boldsymbol{w}_{o,2} \end{bmatrix} \tag{4}$$

The final topology is the recurrent TDNN (R-TDNN) of Fig. 4. An augmented input vector $\boldsymbol{x}_r$ combines the input samples from the delay line with a feedback sample $y(n\text{-}D)$. The network equations are expressed in (5). The one-symbol delay gives this network the form of a matched filter with decision-feedback equalization.

$$\boldsymbol{y}(n) = f(\boldsymbol{x}_r^T \boldsymbol{w}) \tag{5}$$

$$\boldsymbol{x}_r = \begin{bmatrix} x(n) \\ \vdots \\ x(n-N+1) \\ y(n-D) \end{bmatrix}, \quad \boldsymbol{w} = \begin{bmatrix} w_0 \\ \vdots \\ w_{N-1} \\ w_N \end{bmatrix}$$

4. DEMODULATOR TRAINING

Training was performed using supervised learning with backpropagation. A modified version known as real-time recurrent learning (RTRL) was used for the R-TDNN [2, 3]. These are gradient descent optimization methods that minimize a given cost function at the demodulator output. The cost function selected here is mean squared error (MSE). Minimizing MSE is equivalent to maximizing signal-to-noise ratio (SNR), which is the optimization goal for the traditional matched filter. The demodulator output error is defined as follows.

$$e_k = d_k - y(kD) \tag{6}$$

The term $y(kD)$ is the sampled demodulator soft output and d_k is the known symbol value. The MSE cost function is estimated by averaging the squared demodulator output error over a training epoch (message) of L=1000 randomly generated symbols. Error gradients are computed for each weight, and all weights are updated at the end of each epoch. The basic backpropagation algorithm with learning rate of η = 0.1 provided acceptable convergence for a training set of 2000 epochs. Unit-energy pulse shapes and uniformly distributed initial weights in the range ±0.1 produce initial network output values in the linear region of the activation function where the local gradient gain is high.

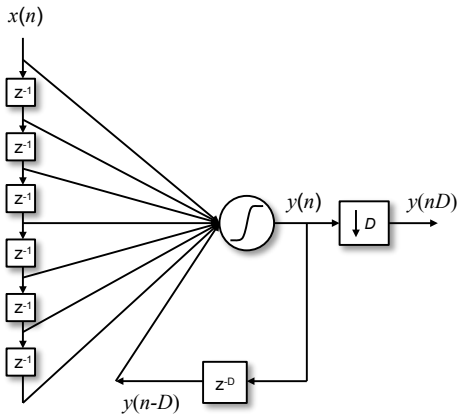

Fig. 4. Single-layer recurrent TDNN.

This ensures good convergence in the early training iterations. After training is completed, symbol error rate is evaluated for each demodulator using randomly generated messages and noise.

5. TRAINING AND TEST RESULTS

We begin with the FF-TDNN topology and a message synthesized using RRC pulse shapes. The number of network weights N was set equal to the length of modulation pulse shape (33 samples), and the signal-to-noise ratio (E_S/N_0) was set to 7 dB. The noise level affects training in two ways. When the noise power is too high, the weight update equation will be noisy and prevent convergence to a clean result. Batch-mode training helps this somewhat by averaging the gradient estimates over a training epoch prior to adjusting the weights. On the other hand, training with too little noise leads the network toward suboptimum solutions that do not provide noise filtering. Training at E_S/N_0 = 7 dB provided a good balance between these effects. As shown in Fig. 5, the learned weights are nearly identical to the RRC pulse shape and represent a matched filter. The neural network has learned the orthogonal series representation of the message, and the network weights are the analysis filter needed to recover the data with minimum noise and inter-symbol interference (ISI).

Next we retrain the 33-tap FF-TDNN using messages synthesized using GF pulse shapes. Training was again performed at E_S/N_0 = 7 dB. The results in Fig. 6 show that for this case the neural network learned a different feature detector (not a matched filter). Matched filtering with the GF pulse shape leads to ISI, which increases MSE at the network output. Instead, the neural network has learned an equalizing filter response that balances the minimization of noise and ISI. From a sampling perspective, the neural network has learned an approximation of the biorthogonal analysis filter needed to recover the data from the message expansion in (1) [17]. The exact biorthogonal analysis filter, $q(n)$, is shown in Fig. 6 for comparison.

Next we train the C-TDNN with messages comprised of GF pulse shapes. The input layer weights (33 per neuron) are randomly initialized as before, and the output layer weights are initialized to [0 1 0] to give preference to the center symbol. Training was again performed using an E_S/N_0 level of 7 dB.

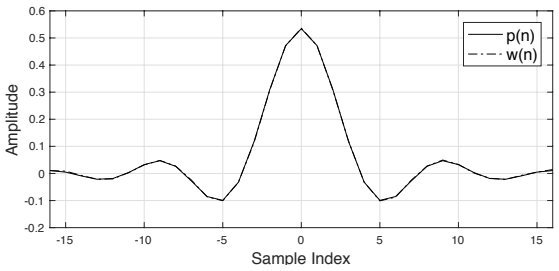

Fig. 5. RRC pulse shape and weights for FF-TDNN.

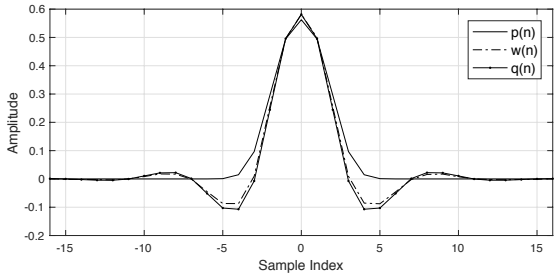

Fig. 6. GF pulse, weights for FF-TDNN, and biorthogonal pulse.

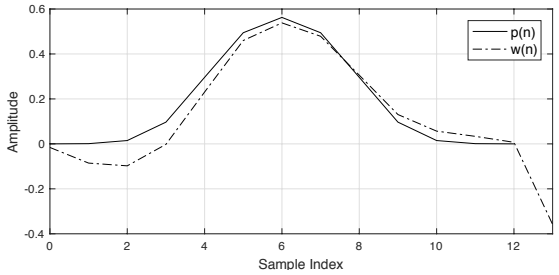

Fig. 7. GF matched filter and weights for R-TDNN.

The convolutional layer weights converged to the same equalizing filter as in Fig. 6, and the output layer weights converged to $\boldsymbol{w}_{\text{out}} = [-0.01\ 0.93\ -0.01]^{\text{T}}$. The output layer effectively ignored the outer two neurons in the convolutional layer. For comparison, a common traditional demodulation technique for GF pulse shaping would use a matched filter in the convolutional layer followed by a 3-tap zero-forcing equalizer (ZFE) in the output layer [14]. The ZFE equalizer weights for the GF pulse used here would be $\boldsymbol{w}_{\text{ZFE}} = [-0.2\ 1.0\ -0.2]^{\text{T}}$.

The final experiment applies the R-TDNN to the system with GF pulse shaping. As in the other cases, training was performed at $E_S/N_0 = 7$ dB, and initial weights and learning rate were similarly configured. The training results for a 14-weight R-TDNN are compared with the GF pulse shape in Fig. 7. Weights w_0-w_{12} are the feed-forward weights, while w_{13} is the feedback weight. It is interesting to note the asymmetric weight pattern. The higher weights (w_9-w_{12}), which overlap the prior symbol, converged closely to the matched filter, while the lower weights (w_0-w_3), which overlap the subsequent symbol, converged to an equalizer response. Each side of the feed-forward response converged in a way to make best use of the available information.

A summary of the symbol error rate curves for each of the neural demodulators is shown in Fig. 8. For RRC pulse shaping, the FF-TDNN performed essentially the same as the ideal matched filter (differences of 0.1 dB or less are attributed to simulation variance). This is consistent with the close agreement between the learned weights and the RRC pulse in Fig. 5. For GF pulse shaping, the traditional matched filter performs poorly due to ISI; however, performance improves significantly with the addition of a zero-forcing equalizer. The FF-TDNN and C-TDNN both perform slightly better than the matched filter plus ZFE, indicating that the minimum-MSE optimization of the neural demodulators found better balance between noise and ISI. The R-TDNN performed slightly better than the FF-TDNN and C-TDNN due to the additional information provided by the estimate of the previous symbol. For comparison, Fig. 9 includes performance of a maximum-likelihood sequence estimator (MLSE) for the GF pulse case. MLSE is known to provide optimum performance in channels with non-Nyquist pulse shapes [14], the results agree closely with ISI-free case of RRC pulse shapes.

6. CONCLUSIONS

From a communications perspective, the elementary neural demodulators considered here were able to learn feature detection equivalent to a matched filter or equalizing filter, depending on the modulation pulse shape. These results are intuitively satisfying because network training is driven by minimization of noise and interference at the symbol decision point. From a signal representation or sampling viewpoint, communication signals can be expressed using a basis expansion with time-shifted pulses serving as basis functions and the data values being the expansion coefficients. There is typically a dual or biorthogonal pulse shape that allows the data to be recovered directly from the message using linear projection. When trained with additive noise, the neural demodulators learned to approximate these biorthogonal pulse shapes with the added constraint of noise minimization.

Although deeper networks are not needed for the simple PAM system considered here, they can provide additional functionality or improve performance for more complex modulation formats or channel conditions. For bandpass communication systems, a more general approach would be to use additional layers at the input of the neural network that map complex (I/Q) samples into a new feature space suitable for the particular modulation format. The initial transformation could be learned as part of the training process as well, which is a key concept in deep learning techniques [18].

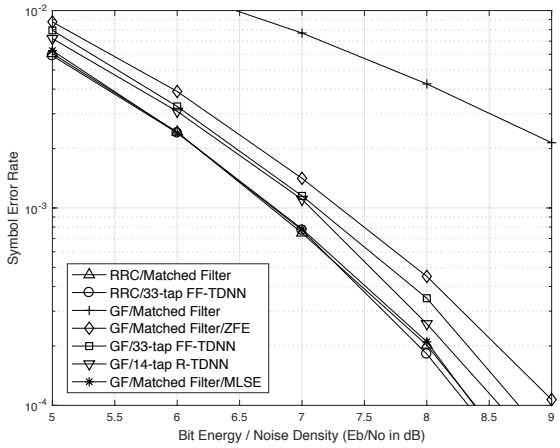

Fig. 8. Symbol error rate performance summary.

7. REFERENCES

[1] B. Widrow and S. Stearns, *Adaptive Signal Processing*, New Jersey: Prentice-Hall, 1985.

[2] J. Principe, et al., *Neural and Adaptive Systems: Fundamentals Through Simulations*, New York: Wiley, 2000.

[3] S. Haykin, *Neural Networks and Learning Machines, 3*rd ed., New York: Pearson, 2009.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, Cambridge, MA: MIT Press, 2016.

[5] S. Mallat, "Understanding Deep Convolutional Networks," *Phil. Trans. R. Soc.*, vol. 374, iss. 2065, Apr. 2016.

[6] R. Lippmann and P. Beckman, "Adaptive Neural Net Preprocessing for Signal Detection in non-Gaussian Noise," *Advances in Neural Information Processing Systems, pp. 124-132,* 1988.

[7] D. Bouras, P. Mathiopoulos, and D. Makrakis, "Neural-net Based Receiver Structures for Single- and Multi-Amplitude Signals in Interference Channels," *Proc. 1994 IEEE Workshop Neural Networks for Signal Processing,* Ermioni, Greece, 1994, pp. 535-544.

[8] T. Whitacre, *A Neural Network Receiver for EM-MWD Communication*, M.S. Thesis, Dept. Elect. Eng., California Polytechnic State Univ., San Luis Obispo, CA, 2011.

[9] S. Haykin, J. Nie, and B. Currie, "Neural Network-based Receiver for Wireless Communications", *IEEE Electron*. *Lett*., vol. 35, pp. 203-205, Feb. 1999.

[10] A. Aiello, D. Grimaldi, and S. Rapuano, "GMSK Neural Network Based Demodulator," *Int. Workshop Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, Crimea, Ukraine, 2001, pp. 2-6.

[11] E. Di Claudio, R. Parisi, and G. Orlandi, "Performance Comparison Among Neural Decision Feedback Equalizers," *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Networks*, Como, Italy, 2000, pp. 361-365.

[12] M. Li, H. Zhong, and M. Li, "Neural Network Demodulator for Frequency Shift Keying," *IEEE Int. Conf. Computer Science and Software Eng.*, Hubei, China, 2008, pp. 843-846.

[13] T. O'Shea and J. Hoydis (July 11, 2017). "An Introduction to Deep Learning for the Physical Layer." Available https://arxiv.org/abs/1702.00832.v2.

[14] J. Proakis, *Digital Communications, 3*rd ed., New York: McGraw Hill, 1995.

[15] L. Couch II, *Digital and Analog Communication Systems, 4*th ed., Macmillan, 1993.

[16] K. Feher, *Wireless Digital Communications: Modulation & Spread Spectrum Applications*, New Jersey: Prentice-Hall, 1995.

[17] Y. Eldar, *Sampling Theory – Beyond Bandlimited Systems*, Cambridge, UK, Cambridge University Press, 2015.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, pp. 436-444, May 2015.