Unsupervised Representation Learning of Structured Radio Communication Signals

Timothy J. O'Shea
Bradley Department of Electrical and Computer Engineering
Virginia Tech
Arlington, VA
http://www.oshearesearch.com

Johnathan Corgan
Corgan Labs
San Jose, CA
http://corganlabs.com/

T. Charles Clancy
Bradley Department of Electrical and Computer Engineering
Virginia Tech
Arlington, VA
http://www.stochasticresearch.com/

***Abstract*—We explore unsupervised representation learning of radio communication signals in raw sampled time series representation. We demonstrate that we can learn modulation basis functions using convolutional autoencoders and visually recognize their relationship to the analytic bases used in digital communications. We also propose and evaluate quantitative metrics for quality of encoding using domain relevant performance metrics.**

Keywords*—*Radio communications, Software Radio, Cognitive Radio, Deep Learning, Convolutional Autoencoders, Neural Networks, Machine Learning

I. Introduction

Radio signals are all around us and serve as a key enabler for both communications and sensing as our world grows increasingly reliant on both in a heavily interconnected and automated world. Much effort has gone into expert system design and optimization for both radio and radar systems over the past 75 years considering exactly how to represent, shape, adapt, and recover these signals through a lossy, non-linear, distorted, and interference heavy channels. Meanwhile, in recent years, heavily expert-tuned basis functions such as Gabor filters in the vision domain have been largely discarded due to the speed at which they can be learned using end-to-end feature learning approaches in deep neural networks.

Here we explore a similar transition from using expert-centric representation and coding to emergent, learned encoding in the radio domain. We expect to better optimize for channel capacity, to be able to translate information to and from channel and compact representations, and to better reason about what kind of information is in the radio spectrum– enabling more lightly supervised radio systems for numerous applications.

This paper provides the first step towards that goal by demonstrating that common radio communications signal bases emerge readily using existing unsupervised learning methods. We outline several techniques which enable this to work to provide insight for continued investigation. This work extends prior supervised feature learning work in the domain in [12].

A. Basis Functions for Radio Data

Widely used single-carrier radio signal time series modulations schemes today use a relatively simple set of supporting basis functions to modulate information into the radio spectrum. Digital modulations typically use sine wave basis functions with pseudo-orthogonal properties in phase, amplitude, or frequency. Information bits are used to map a symbol value s_i to a location in this space $\phi_j, \phi_k, ...$. In figure 1 we show three common basis functions where ϕ_0 and ϕ_1 form phase-orthogonal bases used in Phase Shift Keying (PSK) and Quadrature Amplitude Modulation (QAM), while ϕ_0 and ϕ_2 show frequency-orthogonal bases used in Frequency Shift Keying (FSK). In the final figure of 1 we show a common mapping of constellation points into this space used in Quadrature Phase Shift Keying (QPSK) to encode two bits of information per symbol.

Digital modulation theory in communications is a rich subject explored in much greater depth in numerous great texts such as [3].

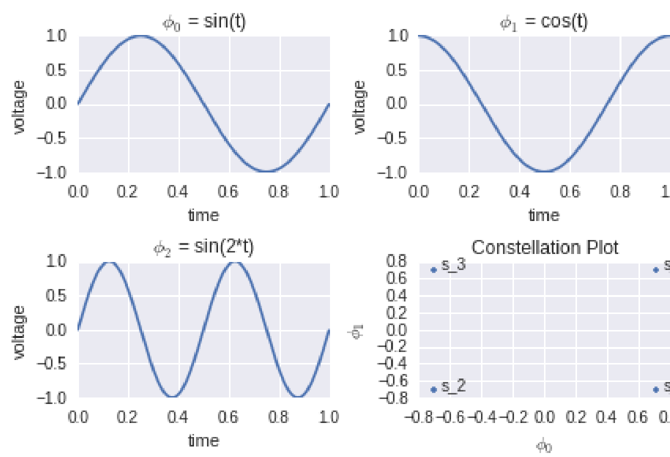

Figure 1. Example Radio Communications Basis Functions

B. Radio Signal Structure

Once basis functions are chosen, data bits are partitioned into symbols and each symbol period occupies a sequential time slot. To avoid creating wideband energy associated with rapid phase transitions on symbol boundaries, a pulse shaping envelope such as a root-raised cosine or sinc filter is used to provide smoothed transitions between discrete symbol values in adjacent time-slots [1]. Three such adjacent symbol time slots are shown in figure 2. Finally sequential shaped symbols are summed to form the transmit signal time-series, $s(t)$.

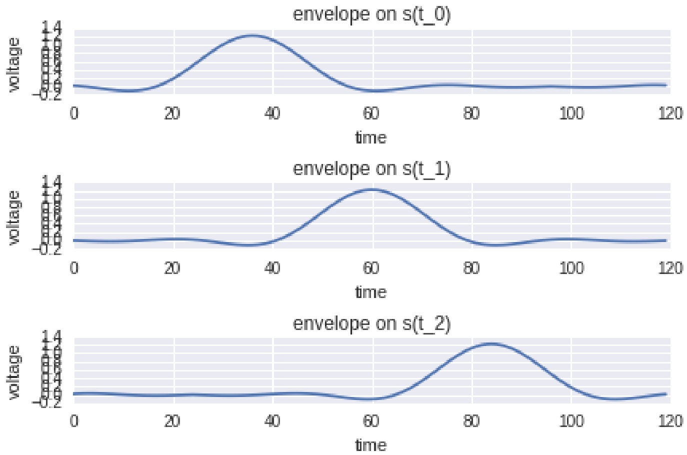

Figure 2. Discrete Symbols Envelopes in Time

C. Radio Channel Effects

The transmitted signal, $s(t)$, passes through a number of channel effects over the air before being received as $r(t)$ at the receiver. This includes time-delay, time-scaling, phase rotation, frequency offset, additive thermal noise, and channel impulse responses being convolved with the signal, all random time-varying processes. A closed form for all these effects can be approximated:

$$r(t) = e^{j*n_{Lo}(t)} \int_{\tau=0}^{\tau_0} s(n_{Clk}(t-\tau))h(\tau) + n_{Add}(t) \quad (1)$$

This complicates transmit data representation from its original when considering the effects of wireless channels as they exist in the real world.

II. Learning from Radio Signals

We focus initially on attempting to learn symbol basis functions from existing modulation schemes in wide use today. We use Quadrature Phase-Shift Keying (QPSK) and Gaussian Binary Frequency Shift Keying (GFSK) as modulations of interest and demonstrate learning the analytical basis functions for these.

A. Building a Dataset

We leverage the dataset from [12] and focus on learning only a single modulation type at a time in this work. This dataset includes the QPSK and GFSK modulations passed through realistic, but relatively benign wireless channels, sampled in 88 complex-valued sample times per training example.

B. Unsupervised Learning

Autoencoders [2] have become a widely used unsupervised learning tool. We review the autoencoder and several improvements within the application to this domain.

1) Autoencoder Architectures: Autoencoders (AEs) learn an intermediate, sparse encoding of an input by using reconstruction cost as their optimization criteria, typically minimize Mean Squared-Error (MSE). They consist of an encoder which encodes raw inputs to a sparse hidden representation, and a decoder which reconstructs an estimate for the input vector in the output.

A number of improvements have been made on autoencoders which we leverage below.

2) Denoising Autoencoders: By introducing noise into the input of an AE training, but evaluating its reconstruction of the unmodified input, Denoising Autoencoders [6] perform additional input noise regularization effect which is well suited in the communications domain where additive Gaussian noise is a modeled impairment.

3) Convolutional Autoencoders: Convolutional Autoencoders [7] are simply autoencoders leveraging convolutional weight configurations in their encoder and decoder stages. By leveraging convolutional layers rather than fully connected layers, we force time-shift invariance learning in features and reduce the parameter count required to fit. Since our channel model involves random time shifting of the input signal, this is an important property to the radio application domain which we feel is well suited.

4) Regularization: We leverage heavy $L_2 = \|\mathbf{W}\|_2$ weight regularization and $L_1 = \|\mathbf{h}\|_1$ activity regularization in our AE to attempt to force it to learn orthogonal basis functions with minimal energy. [4] Strong L_1 activation regularization is important in the narrow hidden layer representation between encoder and decoder where we would like to learn a sparse basis representation of the signal containing known symbols occurring at specific times. Dropout [10] is also used for regularization between intermediate layers.

C. Test Neural Network Architecture

Our goal in this effort was to obtain a minimum complexity network which allows us to convincingly reconstruct the signals of interest with a significant amount of information compression. By using convolutional layers with only one or two filters, we seek to achieve a maximally matched small set of time-basis filters with some equivalence to the expert features used to construct the signal. Dense layers with non-linear activations then sit in between these to provide some estimation of the logic for what the representation and reconstruction should be for those basis filters occurring at different times. The basic network architecture is shown below in figure 3.

D. Evaluation Methods for Reconstruction

For the scope of this work we use MSE as our reconstruction metric for optimization. We seek to evaluate reconstructed signals from BER and SNR, but plan to defer this for later work in the interest of space.

E. Visual Inspection of Learned Representations

Given a relatively informed view of what a smooth band-limited QPSK signal looks like in reality, visual inspection

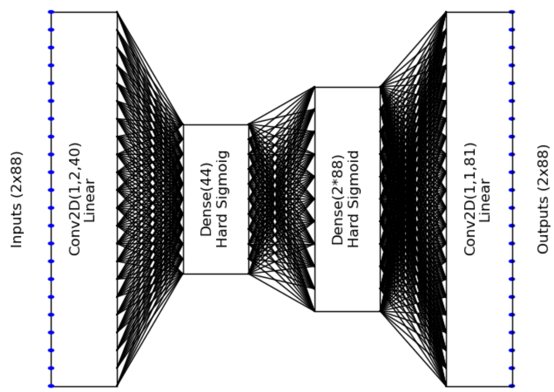

Figure 3. Convolutional Autoencoder Architecture Used

of the reconstruction vs the noisy input signal is an important way to consider the quality of the representation and reconstruction we have learned. The sparse representation is especially interesting as by selecting hard-sigmoid dense layer activations we have effectively forced the network to learn a binary representation of the continuous signal. Ideally there exists a direct GF(2) relationship between the encoded bits and the coded symbol bits of interest here. Figures 4 and 5 illustrate this reconstruction and sparse binary representation learned.

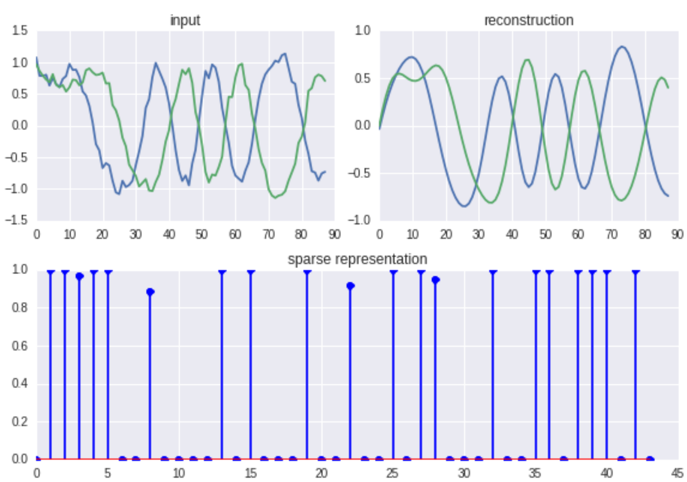

Figure 4. QPSK Reconstruction 1 through Conv-AE

For GFSK, we show reconstructions and sparse representations in figure 6. In this case, the AE architecture converges even faster to a low reconstruction error, but unfortunately the sparse representations are not saturated into discrete values as was the case for the constant modulus signal.

III. Results

We consider the significance of these results below in the context of the network complexity required for representation and the compression ratio obtained.

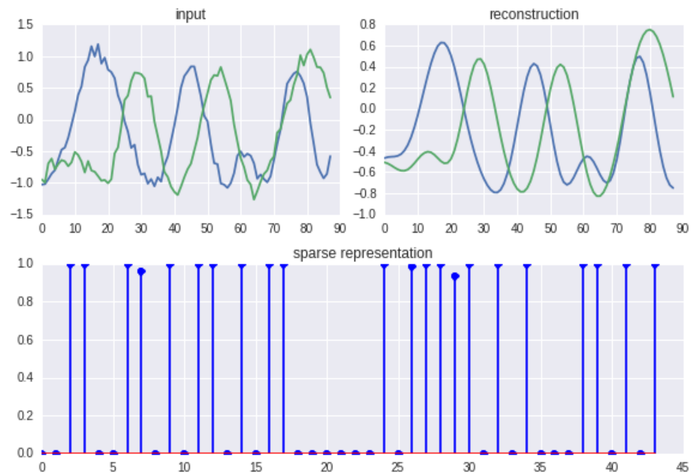

Figure 5. QPSK Reconstruction 2 through Conv-AE

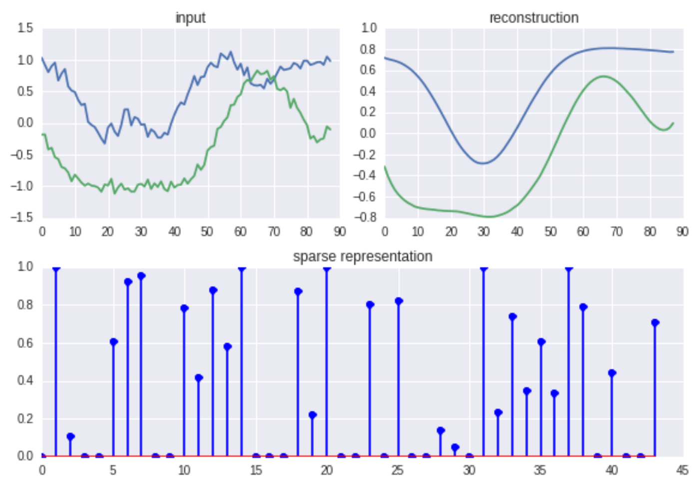

Figure 6. GFSK Reconstruction 1 through Conv-AE

A. Learned Network Parameters

We use Adam [9] (a momentum method of SGD) to train our network parameters in the Keras [11], however we the more widely used RMSprop [13] obtains similar results. Evaluating our weight complexity, we have two 2D convolutional layers, 2x1x1x40 and 1x1x1x81, making a total of only 161 parameters learned in these layers to fit the translation invariant filter features which form the primary input and output for our network. The Dense layers which provide mappings from occurrences of these filter weights to a sparse code and back to a wide representation, consist of weight matrices of 516x44 and 44x176 respectively, making a total of 30448 dense floating point weight values.

Training is relatively trivial with this size network and dataset, we converge on a solution after about 2 minutes of training, 25 epochs on 20,000 training examples using a Titan X GPU.

In figure 7 we show the learned convolutional weight vectors in the encoder first layer. We can clearly see a sinusoid occurs at varying time offests to form detections, and a second sinusoid at double the frequency, both with some minimal

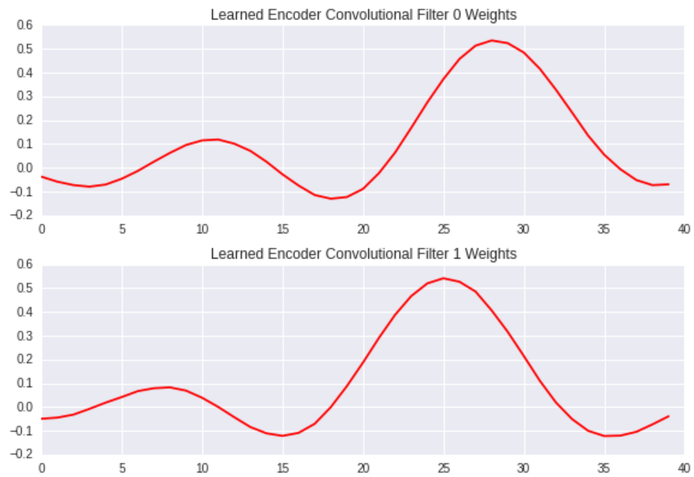

Figure 7. QPSK Encoder Convolutional Weights

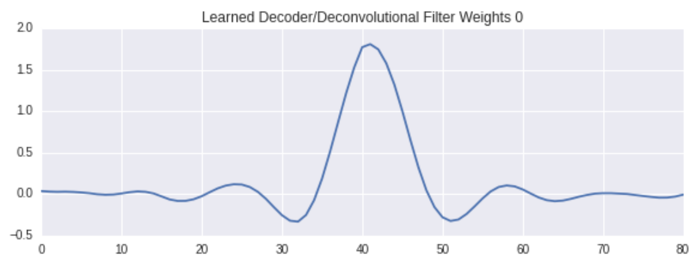

Figure 8. QPSK Decoder Convolutional Weights

pulse shaping apparent on them.

In the decoder convolutional weight vector in figure 8 we can clearly see the pulse shaping filter shape emerge in the form of a scaled sinc or root raise cosine function.

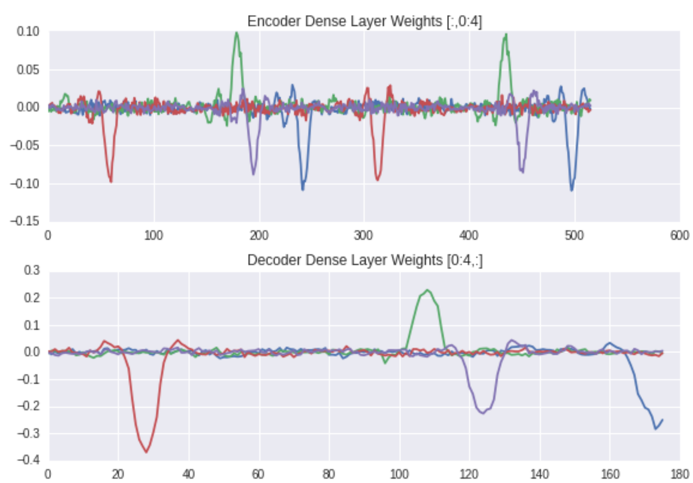

Figure 9. First Four Sparse Representation Dense Weights

In figure 9 we display the learned dense layer weights mappings of various symbol value and offset areas as represented by the convolutional filters. It is important to note that the 1x516 input is a linearized dimension of zero-padded I and Q inputs through two separate filters (2x2x129). We see that a single sparse hidden layer value equates to two pulses representing sinusoidal convolutional filter occurrences in time in the I and the Q channel, with roughly a sinc or root raised cosine window roll-off visibly represented in this time-scale.

B. Radio Signal Representation Complexity

To measure the compression we have achieved, we compare the effective number of bits required to represent the dynamic range in the input and output continuous signal domains with that of the number of bits required to store the signal in the hidden layer. [8]

If we consider that our input signal contains roughly 20dB of signal-to-noise ratio, we can approximate the number of bits required to represent each continuous value as follows.

$$N_{eff} = \lceil \frac{20dB - 1.76}{6.02} \rceil = 4 \text{ bits} \tag{2}$$

Given that we have 88*2 inputs of 4 bit resolution, compressed to 44 intermediate binary values, we get a compression ratio of **16x** = 88*2*4/44.

Given that we are learning around 8 symbols per example, this equates to roughly 16 bits being the most compact possible form of data representation. However, in the current encoder, we preserve timing offset information, phase error, and channel information needed to reconstruct symbols in their specific arrival modes. This is on the order of **4x** the most compact representation possible for the data symbols alone based on rotation, achieving 3x the capcoty bound at 44 bits is not bad. Also considering a compression of 88(time samples)*2(I/Q)*32(float32) to 44*1 bits of hidden representation, this is roughly a 128x compression rate significantly better than attempting to compress to 88*2*4 bits (8x compression), or other dynamic range limiting based methods used today.

IV. Conclusions

We obtain relatively good compression with autoencoders for radio communications signals, however these encode both the data bits and the channel state information which limits attainable compression.

Hard-sigmoid activations surrounding the hidden layer, for constant modulus modulations, seem effective in saturating representation into compact binary vectors, allowing compression of 88 x 64 bit complex values into 44 bits without significant degradation.

Convolutional autoencoders are well suited for reducing parameter space, forcing time-invariance features, and forming a compact front-end for radio data. We look forward to evaluating more quantitative metrics on reconstructed data, evaluating additional multi-level binary or hard-sigmoid representation for multi-level non-constant-modulus signals and investigating the use of attention models in removing channel variance to simplify representation.

Acknowledgments

The authors would like to thank the Bradley Department of Electrical and Computer Engineering at the Virginia Polytechnic Institute and State University, the Hume Center, and DARPA all for their generous support in this work.

This research was developed with funding from the Defense Advanced Research Projects Agency's (DARPA) MTO Office under grant HR0011-16-1-0002. The views, opinions, and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

[1] E. S. Sousa and S. Pasupathy, "Pulse shape design for teletext data transmission", *Communications, IEEE Transactions on*, vol. 31, no. 7, pp. 871–878, 1983.

[2] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length, and helmholtz free energy", *Advances in neural information processing systems*, pp. 3–3, 1994.

[3] B. Sklar, *Digital communications*. Prentice Hall NJ, 2001, vol. 2.

[4] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms", in *Advances in neural information processing systems*, 2006, pp. 801–808.

[5] C. Clancy, J. Hecker, E. Stuntebeck, and T. O'Shea, "Applications of machine learning to cognitive radio networks", *Wireless Communications, IEEE*, vol. 14, no. 4, pp. 47–52, 2007.

[6] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders", in *Proceedings of the 25th international conference on Machine learning*, ACM, 2008, pp. 1096–1103.

[7] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction", in *Artificial Neural Networks and Machine Learning–ICANN 2011*, Springer, 2011, pp. 52–59.

[8] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[9] D. Kingma and J. Ba, "Adam: A method for stochastic optimization", *ArXiv preprint arXiv:1412.6980*, 2014.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[11] F. Chollet, *Keras*, https://github.com/fchollet/keras, 2015.

[12] T. J. O'Shea and J. Corgan, "Convolutional radio modulation recognition networks", *CoRR*, vol. abs/1602.04105, 2016. [Online]. Available: http://arxiv.org/abs/1602.04105.

[13] Y. Dauphin, H de Vries, J Chung, and Y Bengio, "Rmsprop and equilibrated adaptive learning rates for non-convex optimization (2015). arxiv preprint", *ArXiv preprint arXiv:1502.04390*,