Survey on Deep Learning Techniques for Wireless Communications

Theo Diamandis[1]

I. INTRODUCTION

A transmitter, channel, and receiver make up a typical wireless communication system. The channel model describes the underlying physical phenomenon and informs the design of the transmitter and receiver. Signaling schemes are devised according to system objectives, such as robustness to channel impairments, ease of implementation, and minimization of transmit power. Similarly, detection algorithms are designed to optimize system performance subject to chosen constraints.

Most deployed wireless communication systems use electromagnetic signals at frequencies below 10GHz. In these systems, the channels conform to tractable mathematical models that describe electromagnetic propagation with reasonable accuracy. However, in other systems, the channels do not have easily-described channel models. In this category, we highlight two categories of channels: channels describing the propagation of mmWave electromagnetic signals, and unknown channel models describing the propagation of non-electromagnetic signals. The former category, mmWave channels, offer large amounts of bandwidth to support high data rates in next-generation systems. In this category, channel models might be known, but they are too complex and/or change too fast to estimate with reasonable accuracy. This estimation of channel state information (CSI) is essential for many current detection algorithms. The latter category includes molecular channels (i.e. channels over which molecular signals are sent) [1], which offer a means to communicate when electromagnetic propagation is impossible or impractical. In this category, channel models are unknown or difficult to derive, so traditional detection algorithms cannot be used. These considerations prompted researchers to turn to estimation tools from other fields to deduce the channel or circumvent its explicit deduction.

Deep learning techniques have recently exhibited unprecedented success in classification problems for which no well-defined mathematical model exists [2]. For example, the mapping of pixels to object, or of spoken language to meaning. In this survey, we explore current research on deep-learning based channel estimation and detection. These topics are discussed in Sections III and IV respectively. We also overview other research in Section V, although we do not dive into detail. We aim to provide an overview of current methods being explored and to highlight promising areas of further exploration.

EE 359 Final Project

[1]T. Diamandis is with the Department of Electrical Engineering, Stanford University tdiamand at stanford.edu

This survey is organized as follows: In Section II we provide a brief background on deep learning techniques and define terms used. In Sections III and IV we survey literature that uses deep learning techniques for channel estimation and detection respectively, and in Section V we outline other work using deep learning in wireless communications. Using the surveyed work as a basis, we then propose metrics for comparison of detectors in Section IX and highlight promising applications of deep learning in Section X.

***Notation*:** We denote vectors by boldfaced lower case letters, e.g. $\mathbf{x}$ and the i^{th} element by x_i. Matrices are denoted by sans-serif font, e.g. G. Finally, $H(\mathbf{y},\hat{\mathbf{y}}) = H(\mathbf{y}) + D_{\mathrm{KL}}(\mathbf{y}||\hat{\mathbf{y}})$ denotes the cross-entropy between the correct and estimated PMFs of $\mathbf{y}$, and $D_{\mathrm{KL}}(\cdot||\cdot)$ is the Kullback-Leibler divergence [3].

II. DEEP LEARNING PRIMER

In deep learning, we represent the mapping from input to output via a graph. This graph, called a neural network, has several layers. The values of the nodes belonging to layer $n-1$ determine the values of the nodes at layer n through some mapping. The number of layers refers to the depth of the network.

Using a known data set (e.g. a pilot sequence in wireless communications), we train the neural network to best represent the mapping from input to output. In other words, we are looking to find the parameters that describe the mapping between the nodes of the network. In recent years, a combination of specialized hardware, open source deep learning libraries, and increased computing resources have fostered an explosion of growth in the field. We describe commonly used network architectures below. We will refer to these architectures in subsequent sections.

A. Network Architectures

1) Feedforward Neural Network: Feedforward neural networks, as defined in [4], describe a function

$$\mathbf{y} = f(\mathbf{x};\boldsymbol{\theta}) \tag{1}$$

mapping an input $\mathbf{x}$ to an output $\mathbf{y}$. We *train* this network on data to learn the parameters $\boldsymbol{\theta}$ that best describe the function by minimizing some *loss function* $L(\mathbf{y},\hat{\mathbf{y}})$ across the set of training data. These networks are feedforward in the sense that the values of nodes at any layer only depend on previous layers. Specifically, the values $\mathbf{x}_i$ at layer i are a function of $\mathbf{x}_{i-1}$. Layer i is *dense* if it is equal to

$$\sigma(\mathsf{W}_i\mathbf{x}_{i-1} + \mathbf{b}_i) \tag{2}$$

where $\sigma(\cdot)$ is a nonlinear *activation function* $\sigma : \mathbb{R}^n \to \mathbb{R}^n$. Common activation functions and loss functions are listed in Tables I and II respectively.

TABLE I
REFERENCED ACTIVATION FUNCTIONS

Name	Function $(\sigma(\mathbf{x}))_i$
ReLU	$\max(0, x_i)$
softmax	$\frac{e^{x_j}}{\sum_j e^{x_j}}$

TABLE II
REFERENCED LOSS FUNCTIONS

Name	Function $L(\mathbf{y}, \hat{\mathbf{y}})$
log-cosh	$\sum_i \log(\cosh(y_i - \hat{y}_i))$
Huber	$\sum_i \begin{cases} \frac{1}{2}(y_i - \hat{y}_i)^2 & \lvert y_i - \hat{y}_i \rvert < 1 \\ (y_i - \hat{y}_i) & \lvert y_i - \hat{y}_i \rvert \geq 1 \end{cases}$
cross-entropy	$H(\mathbf{y}, \hat{\mathbf{y}})$

B. Convolutional Neural Networks

Convolutional neural networks (CNNs) use a convolution in at least one of the layers. These networks are especially useful in processing grid-like data, such as time series (1d) or images (2d) [4]. In these types of data, only nearby inputs effect a particular output, and the parameters mappings at a particular layer will be the same (e.g. we decode pixels in block k in the same way we decoded block $k-1$). These considerations motivate the use of a convolution, which reduces computational complexity and parameter storage requirements.

C. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) tie the output of layers back into the neural network, effectively keeping track of an internal state (or memory). In other words, the output is conditioned on previous inputs.

III. CHANNEL ESTIMATION

In wireless detection, we often first must estimate parameters of the channel over which we are communicating. This estimate of the channel state information (CSI) is required for detection at the receiver. Traditional algorithms to estimate CSI, such as maximum a posteriori probability (MAP) or minimum mean square error (MMSE) estimation require an analytic model of the channel, but the combination of channel distortion and hardware imperfections can be difficult to model analytically.

[5] applies deep learning to estimate the carrier frequency offset (CFO) and timing estimates to enable detection of single carrier phase-shift keyed signals. We discuss the channel models, implementation, and results of this work below.

A. Channel model

[5] considered four channels in their work: additive white Gaussian noise (AWGN) with no fading, and Rayleigh fading (see discussion in [6, Ch. 3]) with mean delay spreads (in number of samples) of 0.5, 1, and 2 and SNRs of 0Db, 5Db, and 10Db. Data was generated as QPSK bursts with random iid symbols, shaped with a root raised cosine filter with roll-off $\beta = 0.25$ and filter span 6, and sampled at 400kHz. The symbol rate $R = 100$kHz. For the center frequency offset dataset, each burst has a center frequency from $U \sim (-\frac{R}{2}, \frac{R}{2})$ and a random phase offset from $U \sim (0, 2\pi)$. For the timing dataset, the authors prepended the burst with 64 know symbols and a random number of random noise symbols, where the number of noise samples is from from $U \sim (0, 125)$.

B. Implementation

A CNN was used for estimation of CFO and timing in [5]. The architecture used for both estimation consisted of several convolutional layers with the ReLU activation function (see Table I). The authors used the Huber and log-cosine hyperbolic loss functions (see Table II) for their robustness properties.

The authors also implemented the following more traditional methods for estimation. To estimate center frequency, the authors used a periodogram of the m^{th} power of the received signal, where m is the modulation order. The estimated CFO corresponds to the f around which the time average of the Fourier Transform of this received signal is maximized. To estimate timing offset, the authors used a matched filter.

C. Results and comparison

[5] presents evidence that neural network estimators of channel state information can outperform traditional estimators in harsh fading channels and are more efficient computationally.

1) Center Frequency Offset: In AWGN channels, the traditional estimator performs better at higher SNRs and block lengths. However, the neural network (NN) estimator performs better at lower block lengths and SNRs, as depicted in Figs. 1, 2, 3. In fading channels, the neural network approach always performs better in the simulations reported. The performance of the traditional estimator decreases with increasing mean delay spread of the fading channel. In fact, the neural network estimator's standard deviation is almost an order of magnitude less at block lengths on the order of 1024 for delay spreads of 1 and 2.

At block lengths around 256 or less, the neural network estimator uses a comparable number or fewer floating point operations (FLOPs). In the Rayleigh fading channels tested, they offered improved performance for less computational complexity–a win-win. However, at larger block lengths on the order of 1024, the neural network estimator requires over four times the number of FLOPs. Here, the trade off of computational complexity for performance must be considered. However, the authors did not optimize for estimator

complexity; they only optimized for minimization of the loss functions discussed previously.

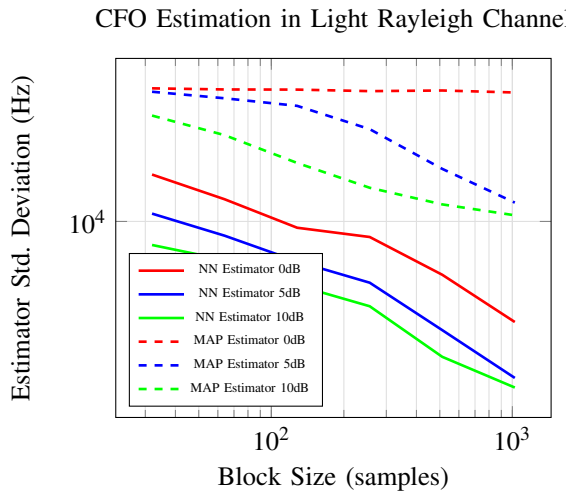

Fig. 1. Mean CFO Estimation Error (Fading σ=0.5) [5]

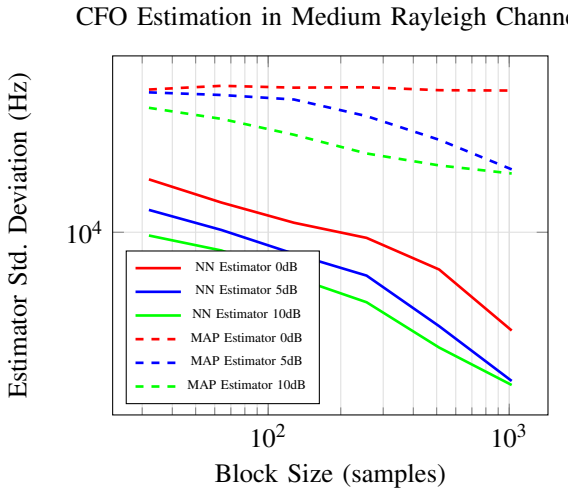

Fig. 2. Mean CFO Estimation Error (Fading σ=1) [5]

2) Timing: The neural network estimator of timing offset performs worse than the match filter in almost all cases, though the performance is on the same order of magnitude. However, the neural network estimator requires almost nine times the number of FLOPs. Based on these results, the authors suggest that there is no reason to use a neural network in practice for timing estimation unless labeled data is much easier to obtain than known reference signal information (e.g. modulation type, pulse shaping filter), as the neural network does not use this information [5].

IV. Detection

Neural networks used for detection are similarly motivated by channels that are difficult to characterize. This difficulty may result from conditions that prevent the acquisition of CSI at the receiver (CSIR) or the lack of a known

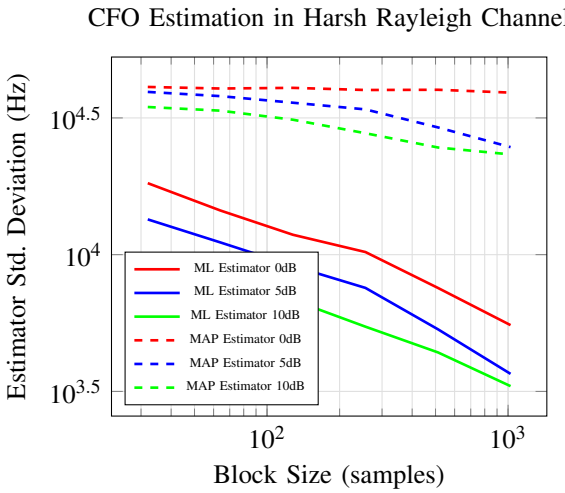

Fig. 3. Mean CFO Estimation Error (Fading σ=2) [5]

channel model. For example, in multiple-input-multiple-output (MIMO) systems with low-resolution analog to digital converters (ADC), reliable CSIR cannot be obtained due to the coarse quantization introduced by the ADC [7]. In molecular communication systems, the underlying channel models are unknown [1]. As a result, both of these systems lend themselves to neural network based detectors. [1] draws a parallel between speech recognition, a domain in which deep learning algorithms have performed remarkably well, and digital communication systems. Both begin with a signal (words or bits) which is send over a channel (acoustic, wireless, or chemical) to some receiver (microphone, cell phone, or chemical sensor) which attempts to detect the original signal. This comparison highlights the potential of deep learning algorithms in signal detection over unknown channels.

A. Implementation

Before a neural network based detector is deployed, a training dataset must be used off-line to train the detector. Afterwards, this detector can be deployed, and it does not have to be trained as part of the detection process.

1) Loss Function Selection: Let the channel be modeled by the conditional distribution

$$P_{\text{channel}}(\mathbf{y}_1, \mathbf{y}_2, \cdots | x_1, X_2, \cdots; \theta) \tag{3}$$

where each transmission symbol $x_k \in S = \{s_1, s_2, \cdots s_m\}$, $\mathbf{y}_k$ is the observed signal during the k^{th} transmission, and θ represents the model parameters. The MAP detector is given by

$$\hat{x_k} = \arg\max_{x \in S} P_{\text{channel}}(\mathbf{y}_k | x; \theta) \tag{4}$$

In [1], it is shown that minimizing the cross-entropy loss function (see Table II), maximizing the log-likelihoods, and maximizing (4) are all equivalent. Thus, this loss function is chosen for the implemented molecular channel detector.

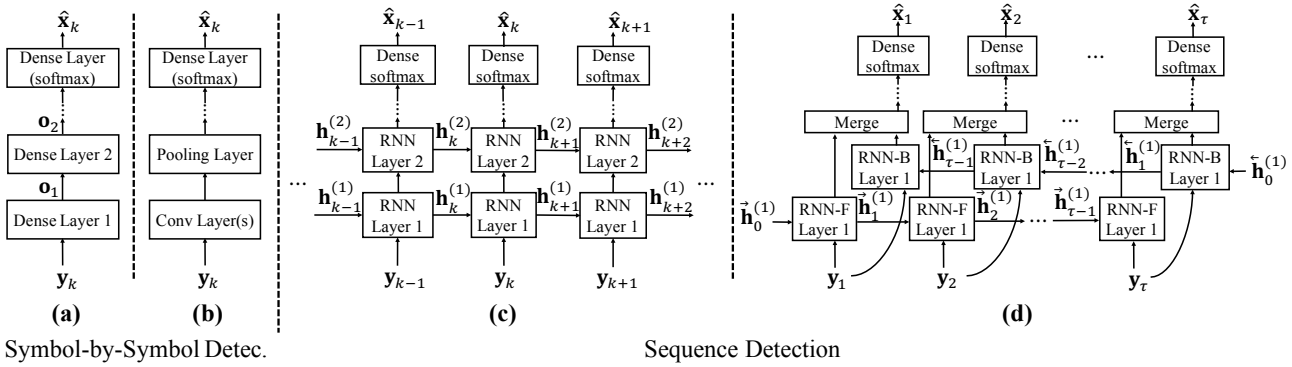

Fig. 4. Estimated directed information graph for the examined indices

2) Neural Network: Several different neural network architectures can be applied to the detection problem in wireless communications, described in [1]. These can be broken down into two primary categories: symbol-by-symbol detectors and sequence detectors.

Symbol-by-symbol detectors include a simple feedforward neural network with fully connected layers followed by a softmax layer (see Table I), or a CNN that ends with a softmax layer. These are depicted in Fig. 4(a) and 4(b) respectively. Note that a CNN is more robust to random delays introduced by the channel.

Sequence detectors consist of RNN detectors which estimate $\hat{x_k}$ not only based on $\mathbf{y}_k$, but also based on $\mathbf{y}_{k-1} \dots \mathbf{y}_1$. These detectors are shown in Fig 4(c). Note that $\mathbf{h}_k$ represents the state of the RNN. One limitation of an RNN is that the estimate $\hat{x}_k$ cannot take into account $\mathbf{y}_i$ for any $i > k$, even though subsequent observations may include information about the symbol at time k. Bidirectional RNNs (BRRNs) overcome this by feeding the observations $\mathbf{y}$ into two RNNs, one in the forward direction and one in the backwards direction. This architecture is depicted in Fig. 4(d).

The authors in [1] proposed to fix the length of the BRNN to L to limit computational complexity. They tune this L as a hyperparameter. This BRNN is slid one observation at a time to estimate each $\hat{x_k}$

3) Testing: The authors test this sliding BRNN detector using both a simulated Poisson channel an experimental molecular communications channel. We discuss their results in the following section.

B. Results and Comparison

In the Poisson channel, the neural network based detector can be compared to the optimal Viterbi detector. In [1], the sliding RNN architecture is shown to be better then the BRNN architecture and close to the optimal Viterbi detector. However, the neural network decoder exhibits two advantages over the Viterbi detector. First, for sequences with high memory order, the neural network detector is much more computationally efficient. Viterbi detector's computational complexity increases exponentially with the memory order, whereas the neural network detector's computational complexity increases linearly with the memory order, though it starts at a higher initially value. Second, the neural network detector works under changing channel conditions without CSI. Though the Viterbi detector exhibits consistently higher performance with CSI, sometimes CSI is difficult to estimate, or the channel changes too quickly to do so without significant overhead.

In the experimental setup, no good baseline of comparison exists; this molecular channel is unknown. The authors demonstrate that in this setup, the sliding BRNN outperforms the regular RNN, as predicted by simulations. The authors illustrate the promise of the neural network detectors by demonstrating reliable communications at data rates an order of magnitude faster than those previously demonstrated with other techniques. Finally, the authors show that this detector is robust to changing channel conditions in practice by training with a good sensor and testing with a degraded sensor.

V. Other Applications

Deep learning has been applied to other problems in wireless communications as well. We do not discuss these applications in detail, but we provide a brief overview in this section

VI. End-to-end System Design

Other techniques from deep learning allow for end-to-end system design. [8] represents a communications system as an autoencoder, a neural network typically used to learn a lower dimensional representation of data. This representation is learned at a middle layer, and the subsequent layers allow for reconstruction at the output. However, a key difference in this work is that the channel is assumed to be known. Specifically, [8] uses a noise layer to represent the channel in the autoencoder. [8] extends this concept to optimize an adversarial network with many transmitters and receivers competing for capacity. This optimization can be done with respect to individual metrics for each transmitter or receiver. Finally [8] points out that deep learning might be useful in augmenting performance in current signal processing algorithms used in wireless communications.

These ideas from [8] are implemented by [9] using software-defined radios (SDRs). [9] creates a communication system using only neural networks. These networks use dense feedforward layers with ReLU activations, and a final layer with the softmax activation (see Table I). They demonstrate results about 1dB worse than the default GNU Radio differential quadrature phase shift keying implementation. Still, [9] shows that this autoencoder technique can be implemented in practice.

VII. Decoding

Standard decoding uses belief propagation, however complexities can be introduced by correlated noise. [10] proposes a belief propagation decoder followed by a CNN to more accurately account for channel noise. The standard belief propagation decoder estimates the bits, and the CNN attempts to learn the noise to better correct noise estimation errors. Their loss function optimizes for both estimation error and the likelihood the estimation errors have a Gaussian distribution. Using simulation results, [10] claims that this decoder achieves better BER performance with lower complexity, can be implemented in parallel, demonstrates robustness to errors in training, and is not channel specific.

VIII. Resource Management

Wireless resource management, such as power allocation, is traditionally solved with numerical methods [6]. These methods can be computational complex, so [11] implements a neural network to approximate current power allocation algorithms. [11] demonstrates order of magnitude increases in speed while maintaining reasonable accuracy. These results present interesting future avenues of research.

IX. Suggested Metrics for Comparison

Unfortunately, implementation in different works is difficult to compare. To ameliorate this, we suggest the following metrics for quantitative comparison of different neural network detector implementations.

A. Bit Error Rate

First and foremost, the bit error rate (BER) is commonly used to differentiate between communications system receivers. We argue that this is the most important metric of comparison for two reasons. First, the bit error rate offers a clear comparison between communication systems using deep learning and those using the standard means of detection, which differ depending on channel type. Second, other factors (e.g. computational complexity) will become less important as computational power continues to increase. Deep learning itself was not feasible at a useful scale a decade ago due to computational limitations. Bit error rate offers a metric that will not be rendered irrelevant with increased technological capability.

B. Computational Complexity

In any implemented communications systems, the computational complexity of the underlying algorithms determines power consumption and delay time of detection. As a result, this metric should be evaluated for different neural network architectures and compared to a baseline of the standard detectors. Specifically, we propose comparison of the number of floating point operations of these detectors, as in [5], and comparison of the degree of potential parallelization. The technology for parallel processing of neural network computation continues to improve (as does NVIDIA's stock price at the time of writing), so one could reasonably expect these efficient, low power specialized architectures to be prevalent in future hand held wireless communications systems, including laptops, tablets, and cell phones.

C. Overhead

Neural networks in communication systems usually require considerable training data. Future work should evaluate the overhead associated with gathering this training data and then training the neural network to perform the desired function (e.g. detection or channel estimation). Both the frequency with which the model needs to be retrained and the amount of data needed to do so should be taken into account. These considerations likely depend in part on the time varying properties of the channel over which one is communicating.

D. Robustness

Wireless communications channels are not time invariant. Everything from movement of cars within a city small cell to ionosphere conditions for long distance communication links will affect the channel. Evaluation of BER across changing channel parameters and noise levels provides valuable results into the robustness of the system, especially for neural networks trained on rapidly varying channels.

X. Promising Applications

We expect the following applications of deep learning in wireless communications to present promising areas of future research based on the surveyed literature. Deep learning based communication systems are useful under certain circumstances such as when the channel's parameters are quickly changing or no known channel models exist. However, it is not a magic bullet for all problems facing future generation system designers. Promising applications are listed below.

A. mmWave Systems

Future wireless systems operating with carrier frequencies in the 10s of GHz (mmWave) can achieve high Gbps data rates due to the large amount of bandwidth available at these frequencies. However, increased system bandwidth necessitates increased sampling rate, leading to higher ADC power consumption. To ameliorate this [7] proposes the use of ADCs with lower precision, as power consumption is linear in and precision and sampling frequency. Low precision

ADCs introduce nonlinear quantization distortion, making CSIR difficult to estimate. [7] uses a machine learning based detector to ameliorate these effects. We propose that a neural network based detector, borrowing from methods discussed in Section IV, presents a promising application of deep learning, as these networks characterize unknown nonlinearities well.

B. MIMO

In MIMO systems, each of the receive antennas requires an ADC, so the ADC power consumption is multiplied by the number of receive antennas. As a result, these systems also would benefit from lower precision ADCs, for which neural network based detectors would likely outperform existing detectors, as discussed in Section X-A.

In addition, [12] shows that with a varying H that describes a MIMO channel, a neural network detector can achieve performance comparable to techniques used in practice with limited computational complexity and no knowledge of SNR. This neural network detector also is robust for ill-conditioned channels. These traits are due to the network's ability to optimize over the distribution of channels and present evidence for promising deep learning applications to MIMO.

C. Unknown or High Memory Channels

As discussed in Section IV, molecular communication systems do not have known channel models. Furthermore, these channels have memory, and the Viterbi Decoder's complexity increases exponentially with this memory length. As a result, neural network based detectors, which do not require any knowledge of the channel and do not have exponentially increasing complexity, have shown performance increases of an order of magnitude over previously used detectors [1]. Deep learning techniques are useful in any setup with an unknown channel or a channel that has a high memory order.

XI. Conclusion

Neural networks are an exciting tool for the field of wireless communication. While they do not outperform the standard detectors when channels are well characterized by known models, neural networks can provide robust performance in hard to characterize, high memory, or unknown channels. The works surveyed show improved performance for channel estimation in harsh fading channels and for molecular communication channels when compared to previous methods used. Furthermore, deep learning might yield improvements when used in tandem with existing techniques for decoding, or when used to approximate existing techniques for system resource optimization.

However, use of these neural networks presents several practical difficulties. First, generating a training dataset that accurately represents the final conditions of channel use is often difficult, and the networks require a large amount of data to train over. While they are robust to some changes in the channel, dramatic changes that require retraining would cause these methods to break down. In addition, the computational complexity of these networks might be significantly higher than the computational complexity of existing algorithms for channel estimation or detection, rendering them impractical because this complexity causes additional delay and power consumption.

Still, we highlight several promising application areas in wireless communications for neural networks based on the surveyed literature. Clearly, deep learning offers new tools to shape the way we think about communication system design.

References

[1] Nariman Farsad and Andrea J. Goldsmith. Neural network detectors for sequence detection in communication systems. 2017.

[2] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[3] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[5] Timothy J. O'Shea, Kiran Karra, and T. Charles Clancy. Learning approximate neural estimators for wireless channel state information. *CoRR*, abs/1707.06260, 2017.

[6] Andrea Goldsmith. *Wireless Communications*. Cambridge University Press, New York, NY, USA, 2005.

[7] Y. S. Jeon, S. N. Hong, and N. Lee. Blind detection for mimo systems with low-resolution adcs using supervised learning. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6, May 2017.

[8] Timothy J. O'Shea and Jakob Hoydis. An introduction to machine learning communications systems. *CoRR*, abs/1702.00832, 2017.

[9] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink. Deep Learning-Based Communication Over the Air. *ArXiv e-prints*, July 2017.

[10] F. Liang, C. Shen, and F. Wu. An Iterative BP-CNN Architecture for Channel Decoding. *ArXiv e-prints*, July 2017.

[11] Haoran Sun, Xiangyi Chen, Qingjiang Shi, Mingyi Hong, Xiao Fu, and Nikos D. Sidiropoulos. Learning to optimize: Training deep neural networks for wireless resource management. *CoRR*, abs/1705.09412, 2017.

[12] N. Samuel, T. Diskin, and A. Wiesel. Deep MIMO Detection. *ArXiv e-prints*, June 2017.

[13] Nariman Farsad and Andrea J. Goldsmith. Detection algorithms for communication systems using deep learning. *CoRR*, abs/1705.08044, 2017.

[14] Eliya Nachmani, Elad Marciano, Loren Lugosch, Warren J. Gross, David Burshtein, and Yair Be'ery. Deep learning methods for improved decoding of linear codes. *CoRR*, abs/1706.07043, 2017.