Performance of Deep Learning LDPC Coded Communications in Large Scale MIMO Channels

Viet Q. Pham
Ho Chi Minh City University of Technology - Ho Chi Minh City University of Technology
Hung N. Dang
Truòng Dai hoc Bách Khoa Hànôi -Hànôi University of Technology
Thuy V. Nguyen
Truòng Dai hoc Bách Khoa Hànôi - Hànôi University of Technology
Hieu T. Nguyen
Institutt for realfag og industrisystemer - Universitetet i Sørøst-Norge

Accepted version of article in:
2019 6th NAFOSTED Conference on Information and Computer Science (NICS)

Publisher's version:
Pham, V. Q., Dang, H. N., Nguyen, T. V., & Nguyen, H. T. (2019, 12.-13. December). Performance of Deep Learning LDPC Coded Communications in Large Scale MIMO Channels. In V. N. Q. Bao, P. M. Quang & H. V. Hoa (Eds.), 2019 6th NAFOSTED Conference on Information and Computer Science (NICS) (pp. 214-218). IEEE.
https://doi.org/10.1109/NICS48868.2019.9023820

Performance of Deep Learning LDPC Coded Communications in Large Scale MIMO Channels

Viet Q. Pham
Faculty of Electrical and Electronics Engineering
Ho Chi Minh City University of Technology
Ho Chi Minh City, Vietnam
Email: vietpq09@gmail.com

Hung N. Dang
Faculty of Information Tech.
Posts and Telecommunications Institute of Technologies
Hanoi, Vietnam
Email: hungdn@ptit.edu.vn

Thuy V. Nguyen
Faculty of Information Tech.
Posts and Telecommunications Institute of Technologies
Hanoi, Vietnam
Email: thuynv@ptit.edu.vn

Hieu T. Nguyen
Faculty of Technology
Natural Sciences, and Maritime Sciences
USN, Norway
Email: hieu.nguyen@usn.no

***Abstract*—In this paper, we investigate the performance of a large-scale multiple-input multiple-output (LS-MIMO) receiver, which deploys a deep neural network and a low-density parity-check (LDPC) code for detecting and decoding disturbed signals. The structure of the low-complexity receiver is also proposed. The proposed receiver was tested with different LS-MIMO configurations to reveal the performance and complexity tradeoff. Besides, our investigation shows that the performance gap of the proposed receiver and the conventional one decreases as the number of transmitting and receive antennas increase. In particular, our experiment results show that the proposed low-complexity receiver has performance loss of about 1.8 dB and 1.5 dB in 10×10 and 32×32 LS-MIMO configurations, respectively.**

***Index Terms*—Deep-learning, large-scale multiple-input multiple-output detector, low-density-parity-check code, soft output, low complexity.**

I. Introduction

The maximum likelihood detector has been known as the optimal solution for not only detecting all symbols simultaneously but also providing minimum joint probability error. The detector can be conducted by some searching algorithms, such as the sphere encoder in [1]. However, since this approach is high complexity to some degree, it is impractical in many applications. Low-complexity algorithms are still an active topic in the research community [2], [3]. Since sub-optimal detection algorithms can provide acceptable results at much lower complexity than the optimal one, they have been received a great deal of attention from researchers. For example, linear receivers, such as the Minimum Mean Squared Error (MMSE) detector, the Matched Filter (MF), and the decorrelator or Zero-Forcing (ZF) detector, are the popular suboptimal detectors.

Some more cutting-edge detectors are Semidefinite Relaxation (SDR), [4], [5], Decision Feedback Equalization (DFE) and Approximate Message Passing (AMP) [6]. Although both AMP and SDR can approximate optimal accuracy under different practical circumstances, they have both advantages and disadvantages. On the one hand, the AMP method is simple and cheap to implement in practice. But, this algorithm could diverge in some problematic settings as it is an iterative method. On the other hand, the SDR is more robust and has polynomial complexity. It takes longer computing time than the AMP.

In recent years, machine learning algorithms, especially deep neural networks, have been increasingly applied to the study of wireless communications. The authors in [7] suggested that unfolding an existing iterative algorithm is a promising approach to design deep architectures. In this regard, each iteration will be considered as a layer, and the algorithm could be viewed as a network. For example, in [8], [9], a variant of AMP and Iterative Shrinkage and Thresholding have been enhanced by unfolding iterations into a network then learning to find the optimal parameters. Also, in [10], the authors discussed various applications of machine learning methods in wireless communications. The work provided a new way of thinking about communications and pointed out a lot of unsolved problems, encouraging the academics to dig deeper into this promising aspect.

In [11], the authors investigated the application of machine learning for channel estimation. An end-to-end detection with continuous signals is studied in [12]. There are also a significant number of new contributions in the context of error-correcting codes, such as [13], [14] and [15]. In [16], the authors used deep learning to train the edges of a Tanner graph, hence improving the belief propagation algorithm. The authors in [17] used a machine learning approach to decode over molecular communication systems. In these systems, chemical signals are utilized for transferring information, and an accurate model of channels is impossible to gain. The authors then developed the approach of decoding without CSI further in [18].

In [19], the task of MIMO detection using an end-to-end approach is solved by deep neural networks. In this work, the learning process is deployed on both sides, i.e., at the transmitter and the receiver. At the former side, deep neural networks are deployed to encode the transmitted signal. In [20], the authors considered the use of deep neural networks for MIMO detection. The work is then developed further in [21]. In this work, the authors proposed two different deep neural networks applied in MIMO detection: a standard fully connected multi-layer network and a Detection Network (DetNet). Deep neural networks are also utilized for detecting real multilevel modulation symbols, which are shown in [22]. In another works, [23], [24], [25], sub-optimal message-passing iterative MIMO decoders are enhanced by utilizing the approach in [7], [16].

The paper will focus on investigating the performance of deep-learning protograph LDPC coded communications in LS-MIMO channels. Besides, we propose an approach to connect channel coding technique from [26] and a deep MIMO detector from [20] together. In particular, we use the *atanh*-function to produce the soft output for the deep-learning detector. The soft information at the output of the deep-learning detector is coupled to the input of the conventional message-passing LDPC decoder to form a low-complexity receiver. Then, we carry out experiments to reveal the performance and complexity tradeoff of the proposed receiver.

II. System Model

The proposed system will utilize a deep neural network for detecting received signals at the receiver. The network was mostly proposed in from [20]. Besides, the Low-Density Parity Check (LDPC) code technique from [26] is deployed at both the transmitter and the receiver to combat the noisy received signals. All signals will be assumed to be baseband, e.g., digital, for the simplification purpose.

Consider the transmission between a transmitter and a receiver where M and N are the numbers of transmitting and receiving antennas, respectively. Let $\boldsymbol{s}$ be a message vector at the transmitter. $\boldsymbol{u}$ is a coded message. $\boldsymbol{x} = [x_1, x_2, \ldots, x_M]^T \in \mathbb{R}^{M\times 1}$ denotes the transmitted signal.

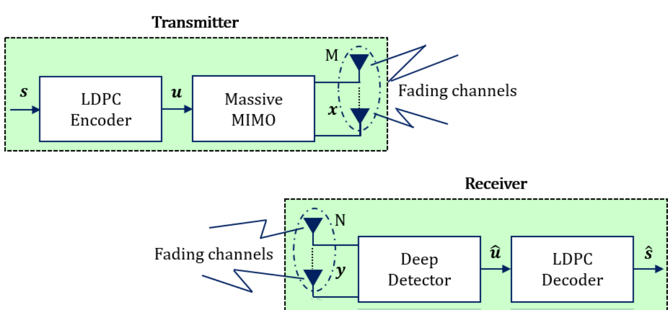

Fig. 1. Communication model.

The elements of $\boldsymbol{x}$ are assumed to be uniformly distributed between the two values, one and minus one, which means that $x_i \in \{-1, 1\}, \forall i = 1, \ldots, M$. The received signal is then:

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}, \tag{1}$$

where $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^T \in \mathbb{R}^{N\times 1}$. $\boldsymbol{H} \in \mathbb{R}^{N\times M}$ denotes a channel matrix whose elements follow independent and identically distributed real Gaussian with zero mean and unit variance, $\mathcal{N}_{\mathbb{R}}(0,1)$. $\boldsymbol{w} \in \mathbb{R}^{N\times 1}$ is real additive white Gaussian noise with zero mean and variance N_0, $\boldsymbol{w}_{\mathbb{R}} \sim \mathcal{N}(0, N_0)$. The received signal after being detected, $\hat{\boldsymbol{u}}$, will be passed through the channel decoder to regain the estimate of the original message, $\hat{\boldsymbol{s}}$. The system model is illustrated in Fig. 1.

III. Deep-Learning-Based Receiver

In this section, the architecture of the detector and how it will be combined with the channel decoder will be described. Given that the network is fundamentally promoted from [20], there is a modification in its structure in order to be matched with the decoder.

A. The architecture of the detector

Before going into the architecture of the network, a rectifier and a soft sign operator which are two essential components of the deep neural network will be discussed first.

A rectifier is an activation function which returns its input if the value is positive. Otherwise, the rectifier will return zero. Mathematically, the rectifier is defined as:

$$r(x) = x^{+} = max\{0, x\}. \tag{2}$$

In 2000, the function was first proposed by Hahnloser et al. in [27], [28] with strong biological motivations and plenty of mathematical justifications. However, the function was first proved its potential a decade later. In 2011, the authors in [29] demonstrated the outstanding performance of the rectifier compared to the conventional activation functions, such as the sigmoid function, when it comes to training a considerable network. Nowadays, the rectifier is the most widely-used activation function in the field of deep neural network [30], [31]. A rectified linear unit (ReLU) is a unit employing the rectifier.

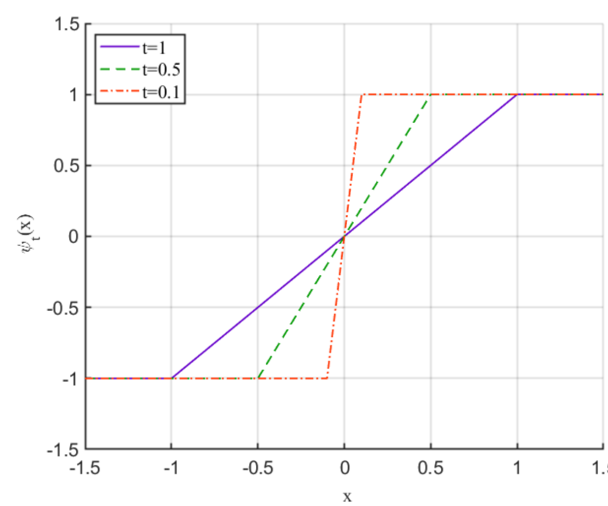

Fig. 2. The piecewise linear soft sign operator with different values of t.

Another fundamental component in the detection network is the soft sign operator. In deep learning, since outputs of an arbitrary layer could be amplified by the later layers and kept increasing excessively, the soft sign operator is taken into account for mapping outputs of a layer into a specific range, especially from -1 to 1. In this detection network, instead of choosing a hard sign operator, such as

$$\psi(x) = \begin{cases} 1 & 0 \le x \\ -1 & x < 0 \end{cases}, \tag{3}$$

the operator is chosen as in [20]:

$$\psi_t(x) = -1 + \frac{r(x+t)}{|t|} - \frac{r(x-t)}{|t|}, \quad (4)$$

The function in Eq. 4 is called a piecewise linear soft sign function. It is a soft sign function as it does not return binary outputs but the values ranging from -1 to 1. In addition, the function has an adjustable parameter, t, which allows the operator to vary its gradient from layer to layer. This would result in increasing flexibility of the model, hence avoiding underfitting. The functions with different parameters are illustrated in Fig. 2. The operator is a piecewise function as if the input is a vector or a matrix, the operator will operate on every entry of the input. This is illustrate by the following equation

$$\psi_t(\boldsymbol{x}) = \begin{bmatrix} \psi_t(x_1) \\ \psi_t(x_2) \\ \vdots \\ \psi_t(x_M) \end{bmatrix}. \quad (5)$$

The rectifier is a piecewise function also.

The architecture of each layer in the detection network is motivated from [20] and shown in Fig. 3.

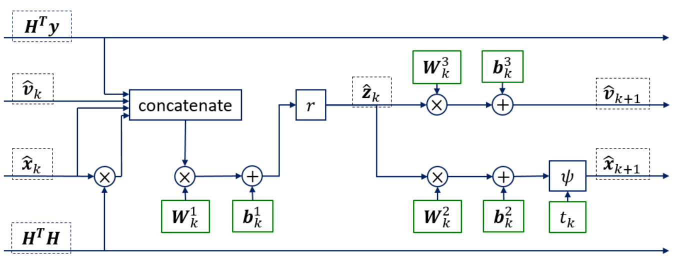

Fig. 3. The architecture of a layer.

As shown in figure 3, there are three essential operators in each layer: the concatenator, the rectifier, and the picewise linear soft sign function. Each layer will have four inputs and two outputs. The inputs are $\boldsymbol{H}^T\boldsymbol{y} \in \mathbb{R}^{M\times 1}$, $\hat{\boldsymbol{v}}_k \in \mathbb{R}^{M\times 1}$, $\hat{\boldsymbol{x}}_k \in \mathbb{R}^{M\times 1}$, and $\boldsymbol{H}^T\boldsymbol{H} \in \mathbb{R}^{M\times M}$. $\hat{\boldsymbol{x}}_k$ and $\hat{\boldsymbol{v}}_k$ are the outputs from the previous layer, the $(k-1)^{th}$ layer. $\hat{\boldsymbol{x}}_k$ is the estimate of the received signals of the $(k-1)^{th}$ layer. $\hat{\boldsymbol{v}}_k$ is a lifting vector which is deployed to lift $\hat{\boldsymbol{x}}_k$ to a higher dimension. The computing process in each layer could be summarized in the following equations:

$$\hat{\boldsymbol{z}}_k = r\left(\boldsymbol{W}_k^1 \begin{bmatrix} \boldsymbol{H}^T\boldsymbol{y} \\ \hat{\boldsymbol{x}}_k \\ \boldsymbol{H}^T\boldsymbol{H}\hat{\boldsymbol{x}}_k \\ \hat{\boldsymbol{v}}_k \end{bmatrix} + \boldsymbol{b}_k^1\right), \quad (6a)$$

$$\hat{\boldsymbol{x}}_{k+1} = \psi\left(\boldsymbol{W}_k^2\hat{\boldsymbol{z}}_k + \boldsymbol{b}_k^2, t_k\right), \quad (6b)$$

$$\hat{\boldsymbol{v}}_{k+1} = \boldsymbol{W}_k^3\hat{\boldsymbol{z}}_k + \boldsymbol{b}_k^3, \quad (6c)$$

$$\hat{\boldsymbol{x}}_0 = \boldsymbol{0}, \quad (6d)$$

$$\hat{\boldsymbol{v}}_0 = \boldsymbol{0}, \quad (6e)$$

where $k = 1, \ldots, K-1$. $\boldsymbol{W}_k^1 \in \mathbb{R}^{8M\times 5M}$, $\boldsymbol{W}_k^2 \in \mathbb{R}^{M\times 8M}$, $\boldsymbol{W}_k^3 \in \mathbb{R}^{2M\times 8M}$ are weights in each layer. $\boldsymbol{b}_k^1 \in \mathbb{R}^{8M\times 1}$, $\boldsymbol{b}_k^2 \in \mathbb{R}^{M\times 1}$, $\boldsymbol{b}_k^3 \in \mathbb{R}^{2M\times 1}$ are biases. t_k is the coefficient of the piecewise linear soft sign operator of the k^{th} layer.

B. Training Process

The loss function is chosen similarly as in [20] to combat difficulties in training deep networks, such as sensitivity to initialization, the saturation of the activation functions, and vanishing gradients. An extra feature from ResNet in [32] is also utilized to enhance the performance of the training process. In particular, the output of each layer will be weighted with the output of the previous ones. The Adam optimizer is also utilized for training the deep neural network.

Besides, since the detector tends to be converged when it gets close to the optimal point, the learning rate with exponential decay will be utilized in the training processes. The detector will be trained through 50000 iterations with 5000 random samples for each batch. In this work, since the channel model is known, the training samples will be randomly created for each batch, which results in a significant number of training samples, hence preventing overfitting.

C. The proposed receiver's architecture

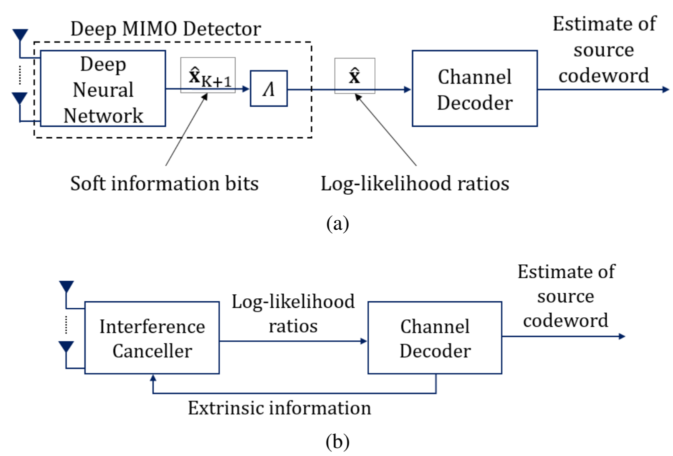

Fig. 4. Block diagrams of: (a) the proposed system; and (b) the conventional one.

Given that the detector is fundamentally introduced from [20], there is a modification in its structure to be fit with the message-passing decoder. The DetNet detector has a total of K layers, but only the first $(K-1)$ layers have trainable coefficients as the last layer is an estimator which has nothing to do with the training process. In the original detector proposed in [20], since the neural network outputs values which are ranged from -1 to 1, the authors had chosen the sign function as the estimator to convert these values into binary bits of -1 and 1. However, in this work, the estimator will be chosen as the inverse hyperbolic tangent function as given in (7).

$$\hat{\boldsymbol{x}} = 2.atanh\left(\hat{\boldsymbol{x}}_{K+1}\right), \quad (7)$$

where $\hat{\boldsymbol{x}}$ is the outputs of the deep neural detector. This parameter plays the role of channel log-likelihood (LLR) ratios of the received information signal in the conventional belief propagation decoder, as shown in Fig. 4(b). $\hat{\boldsymbol{x}}_{K+1}$ is the output of the $(K-1)$ layer of the deep neural network, which is considered as soft information bits. The inverse function in the equation above was given in [33] in which outputs of the canceller would be treated as log-likelihood ratios of the transmitted symbols. The combination of the detector and the channel decoder is plotted in Fig. 4(a).

By deploying the deep detector, the proposed receiver is not required to feed back the extrinsic information as in the conventional receiver. This results in a system with lower complexity. It is even more important to note that the proposed receiver does not require the knowledge of the signal to noise ratio (SNR) to detect the received signals. This advantage of the proposed receiver is interpreted as a means to lower the receiver's complexity [21].

In contrast, there are feeding forward and backward information between the canceller and the decoder in the conventional system, as shown in Fig. 4(b). This Turbo-like architecture helps to increase the accuracy of the decoded bits at each iteration.

IV. Numerical Results

A. Set-up

The simulations are implemented on a Core-i5 laptop with a 2.5-GHz Central Processing Unit (CPU) and a 4-GB Random-Access Memory (RAM). The training process is implemented in Python programming language with supporting from the Tensorflow [34] framework running on Python 3.6 platform. The optimal coefficients are saved in text files and treated as parameters for the proposed system. All simulations were performed in the C++ programming language.

The number of layers is chosen three times of the number of transmit antennas (i.e., $K = 3M$). The transmitter and the receiver will have the same number of antennas in all simulations, $M = N$. The protograph LDPC code with protomatrix in (8) is used. This code is one of our optimized codes for LS-MIMO channels. The number of bits per frames for the LDPC code is 4800 bits. At each data point in the simulations, the proposed system is tested until reaching 100 frame errors.

$$\mathbf{B}_{1/2} = \begin{pmatrix} 2 & 2 & 1 & 0 & 0 & 1 \\ 2 & 2 & 1 & 1 & 0 & 0 \\ 2 & 1 & 2 & 1 & 2 & 0 \end{pmatrix} \tag{8}$$

B. Numerical Results

Fig. IV-B illustrates the bit error rates (BERs) of the deep-learning protograph LDPC coded communications systems with different numbers of antennas. The graph reveals some insights for comparing the performance between the conventional and the proposed receivers. The $BER = 10^{-4}$ line will be utilized as a reference when it comes to comparing the average performance.

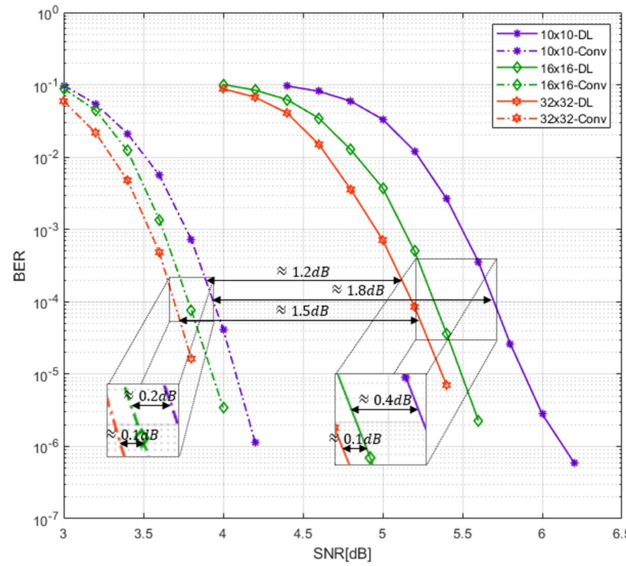

Fig. 5. The BER performance.

It can be inferred from the graph that the BERs of both systems tend to decrease either when raising the SNR or when increasing the number of antennas. For example, the BER of the 10×10 LS-MIMO system is a little bit below 10^{-6} at $SNR = 6.2$ dB. In the 16×16 LS-MIMO configuration, the BER approximating 2.10^{-6} at $SNR = 5.6$ dB, and the 32×32 LS-MIMO configuration provides the BER level slightly below 10^{-5} at $SNR = 5.4$ dB. The 32×32 LS-MIMO configuration posses the lowest BER when comparing at the same SNR level.

It is observed from Fig. IV-B, it is evident that the conventional receiver outperforms the proposed one. In particular, the coding gain of the traditional receiver over the proposed receiver is 1.8 dB in the 10×10 LS-MIMO configuration. Nevertheless, the coding gain decreases to 1.5 dB when the number of antennas increases to 32. The coding gain of the conventional receivers comes at the penalty for the system complexity. Before closing this section, we should note that the proposed receiver possesses not only complexity but also the robustness. The later attribute comes from the fact the deep-learning LS-MIMO detection does not require the estimated SNR.

V. Conclusions

In this paper, we propose an approach to produce the soft output information for the deep-learning LS-MIMO detector. The proposed approach facilitates the usage of the deep-learning LS-MIMO detector together with the message-passing LDPC decoder to have low complexity. We performed the experiments to reveal the performance and complexity tradeoff of the proposed receiver. To reduce the complexity and increase the robustness, the proposed receiver has coding loss about 1.8 dB and 1.5 dB for 10×10 and 32×32 LS-MIMO configurations, respectively.

REFERENCES

[1] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, Aug 2002.

[2] Z. Guo and P. Nilsson, "Algorithm and implementation of the k-best sphere decoding for MIMO detection," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 3, pp. 491–503, March 2006.

[3] S. Suh and J. R. Barry, "Reduced-complexity MIMO detection via a slicing breadth-first tree search," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1782–1790, March 2017.

[4] J. Jalden and B. Ottersten, "The diversity order of the semidefinite relaxation detector," *IEEE Trans. Inf. Theory*, vol. 54, no. 4, pp. 1406–1422, April 2008.

[5] Z. Luo, W. Ma, A. M. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 20–34, May 2010.

[6] C. Jeon, R. Ghods, A. Maleki, and C. Studer, "Optimality of large MIMO detection via approximate message passing," in *2015 IEEE International Symposium on Information Theory (ISIT)*, June 2015, pp. 1227–1231.

[7] J. L. R. J. R. Hershey and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," *arXiv:1409.2574*, 2014.

[8] K. Gregor and Y. Lecun, "Learning fast approximations of sparse coding," 08 2010.

[9] M. Borgerding and P. Schniter, "Onsager-corrected deep learning for sparse linear inverse problems," 12 2016, pp. 227–231.

[10] T. O'Shea and J. Hoydis, "An introduction to machine learning communications systems," 02 2017.

[11] H. Ye, G. Y. Li, and B. Juang, "Power of deep learning for channel estimation and signal detection in ofdm systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, Feb 2018.

[12] S. Dörner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep learning based communication over the air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, Feb 2018.

[13] E. Nachmani, E. Marciano, D. Burshtein, and Y. Be'ery, "Rnn decoding of linear block codes," 02 2017.

[14] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 119–131, Feb 2018.

[15] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," 03 2017, pp. 1–6.

[16] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Sep. 2016, pp. 341–346.

[17] N. Farsad and A. Goldsmith, "Detection algorithms for communication systems using deep learning," 05 2017.

[18] ——, "Neural network detection of data sequences in communication systems," *IEEE Trans. Signal Process.*, vol. PP, 01 2018.

[19] T. O'Shea, T. Erpek, and T. Clancy, "Deep learning based MIMO communications," 07 2017.

[20] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, July 2017, pp. 1–5.

[21] N. Samuel, A. Wiesel, and T. Diskin, "Learning to detect," *IEEE Transactions on Signal Processing*, pp. 1–1, 2019.

[22] V. Corlay, J. J. Boutros, P. Ciblat, and L. Brunel, "Multilevel MIMO detection with deep learning," *CoRR*, vol. abs/1812.01571, 2018. [Online]. Available: http://arxiv.org/abs/1812.01571

[23] X. Tan, W. Xu, Y. Be'ery, Z. Zhang, X. You, and C. Zhang, "Improving massive MIMO belief propagation detector with deep neural network," *CoRR*, vol. abs/1804.01002, 2018.

[24] M. Imanishi, S. Takabe, and T. Wadayama, "Deep learning-aided iterative detector for massive overloaded MIMO channels," *CoRR*, vol. abs/1806.10827, 2018. [Online]. Available: http://arxiv.org/abs/1806.10827

[25] H. He, C. Wen, S. Jin, and G. Y. Li, "A model-driven deep learning network for MIMO detection," *CoRR*, vol. abs/1809.09336, 2018. [Online]. Available: http://arxiv.org/abs/1809.09336

[26] T. V. Nguyen, C. Pham, and H. T. Nguyen, "Delay-limited protograph low density parity codes for space-time block codes," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2018, pp. 1–6.

[27] R. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. Douglas, and H. Sebastian Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, pp. 947–51, 07 2000.

[28] R. Hahnloser, H. Seung, and J.-J. Slotine, "Permitted and forbidden sets in symmetric threshold-linear networks," *Neural computation*, vol. 15, pp. 621–38, 04 2003.

[29] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudík, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 315–323. [Online]. Available: http://proceedings.mlr.press/v15/glorot11a.html

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, 05 2015.

[31] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *CoRR*, vol. abs/1710.05941, 2017. [Online]. Available: http://arxiv.org/abs/1710.05941

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[33] W. Fukuda, T. Abiko, T. Nishimura, T. Ohgane, Y. Ogawa, Y. Ohwatari, and Y. Kishiyama, "Low-complexity detection based on belief propagation in a massive MIMO system," in *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, June 2013, pp. 1–5.

[34] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. J. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Józefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. G. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. A. Tucker, V. Vanhoucke, V. Vasudevan, F. B. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, vol. abs/1603.04467, 2016. [Online]. Available: http://arxiv.org/abs/1603.04467