Deep Learning for Channel Decoding Under Correlated Noise

Xiangxiang Zhang[1], Tao Luo[1]
[1]*Beijing Key Laboratory of Network System Architecture and Convergence*
Beijing University of Posts and Telecommunications

Keywords: DHDV decoder, DnCNN, Convolutional code, Correlated noise.

Abstract: Traditional Viterbi decoding algorithm degrades performance under correlated noise. Inspired by the powerful learning from data of deep neural networks, we apply denoising convolutional neural network (DnCNN) to channel decoding under correlated noise. In this paper, we propose a DnCNN hard decision Viterbi decoder (short for DHDV decoder) architecture to enhance the performance of convolutional codes under correlated noise. The architecture applies DnCNN to denoise and improves the decoding SNR (signal to noise ratio) through the correlation of noise correlation. Then, the Viterbi Algorithm decoder decodes from the denoised data. DHDV decoder obtains greater BER performance improvement under stronger noise correlation, and it is robust under different convolutional codes and correlated noise models. Comparing the complexity of DnCNN network and matrix multiplication whitening method, DnCNN network complexity is lower when the code length is longer.

1 INTRODUCTION

The convolutional code is a type of error-correcting code that generates parity symbols through the sliding application of a boolean polynomial function to source bits and is widely used in WLAN and TD-LTE communication system. The Viterbi algorithm based on Maximum Likelihood Decoding Method performs well under AWGN channels. However, the practical channel is not ideal like AWGN channel. The correlation in channels and noise exists in the real communication system. Noise correlation is caused by filtering, oversampling and device noise (S. K. Sharma et al, 2013). Colored noise is from common buildings, residential electronic in power line communication (L. Di Bert et al, 2011) and exists in 10GBASE-t Ethernet (B. Karanov, 2018).

The channel decoder is always designed for the realistic channel environment, but not performing well under correlated noise (J. D. Wang and H. Y. Chung, 1900). The popular method to solve the noise correlation is to transform correlated noise to white noise. However, it is hard to get expert knowledge about the correlated noise, which increases the difficulty of estimating noise. In addition, correlated noise presents diverse, so the specific structure is desired for each type of correlated noise. Moreover, the problem of high complexity exists in noise whitening (T. Ishihara and S. Sugiura, 2016), which needs matrix multiplication.

Nowadays, Deep learning (DL) provides new ideas for solving complex problems and it shines in computer vision (K. He et al, 2016), natural language processing (I. Sutskever et al, 2014), pattern recognition and many other areas for his strong ability to learn from the training data. The potential of DL to the physical layer has increasingly attracted our attention and the DL-based methods provide new ideas to tackle communication problems (T. Wang et al, 2017). However, no work is mentioned about the channel decoder for convolutional code under correlated noise.

In this paper, we use DnCNN to remove the interference caused by correlated noise before hard decision Viterbi decoder decoding for the convolutional codes. The DnCNN hard decision Viterbi decoder (short for DHDV decoder) receives the bits from the channel. Then, we divide the received bits into several fixed-size blocks and input them in the trained DnCNN model. The data from the channel can be regarded as a 2D image and

Zhang, X. and Luo, T.
Deep Learning for Channel Decoding Under Correlated Noise.
DOI: 10.5220/0008098301350141
In *Proceedings of the International Conference on Advances in Computer Technology, Information Science and Communications (CTISC 2019)*, pages 135-141
ISBN: 978-989-758-357-5

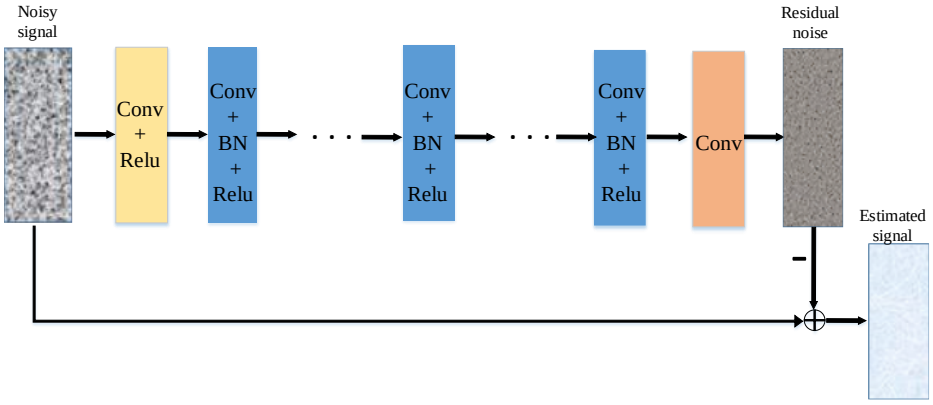


Figure 1: Network architecture of the DnCNN denoise.

DnCNN can extract feature for denoising. After the DnCNN model, the hard decision Viterbi algorithm follows to decode for the source bits. The DnCNN model can improve the decoding SNR and hence the DHDV decoder achieves better decoding performance.

The DHDV decoder has many advantages. Firstly, it performs better than the hard decision Viterbi decoder, even better than the soft decision Viterbi decoder under the strongly correlated noise. Besides, the DnCNN model mainly consists of CNN structure, which requires a few parameters to generate the neural network, so it runs fast in the calculation. Secondly, the DnCNN model learns the network directly from the training data, so it does not require knowledge from the channel and additional ways. Thirdly, the DHDV decoder structure is flexible. As for different situations, we can obtain the new training data and retrain the DnCNN model. Unlike the difficulty of data acquisition in images and texts, we can generate as many training samples as we like through man-made signals.

2 RELATED WORK

2.1 DnCNN Network

Combined with Residual learning and batch normalization techniques based on CNN, DnCNN gets the better performance in Gaussian denoising, image resolution and JPEG image deblocking (K. Zhang et al, 2017). Figure 1 illustrates the DnCNN architecture.It consists of 6 convolutional layers. For the first Conv+ReLU layer, 20 filters of size 2×3×1 are used to generate 20 feature maps and rectified linear unit is respectively defined as

$$g_{relu}(x) = \max\{0, z\} \quad (1)$$

As for Conv+BN+ReLU layer, 20 filters of size 2×3×20 are used, and batch normalization is added between convolution and ReLU, which can speed up the convergence of training. For the last Conv layer, 1 filter of size 2×3×20 is used to reconstruct the output. Besides, output matrix size should keep the same as the input matrix size, we directly pad zeros before convolution to make sure that each feature map of the middle layers has the same size as the input matrix.

In general, the training efficiency of mini-batch stochastic gradient descent (SGD) is largely reduced by internal covariate shift. Batch normalization is proposed to alleviate the internal covariate shift by incorporating a normalization step and a scale and shift step before the nonlinearity in each layer (K. Zhang et al, 2017). For batch normalization, only two parameters per activation are added, and they can be updated with back-propagation. Batch normalization performs well at fast training, better performance, and low sensitivity to initialization. The DnCNN adopts a residual learning strategy, and the hidden CNN layers aim to remove the clean signals implicitly. The motivation for doing this is to use residuals to learn described entity mapping or approximate identity mapping.

2.2 Deep Learning for Communicaition

O'Shea et al (2017) use auto-encoder to design the communication system through the machine learning. The DnCNN based on CNN is robust and low-complexity in image denoising (K. Zhang et al, 2017). Learned Denoising-based Approximate Message Passing (LDAMP) network based on DnCNN has been applied to the channel

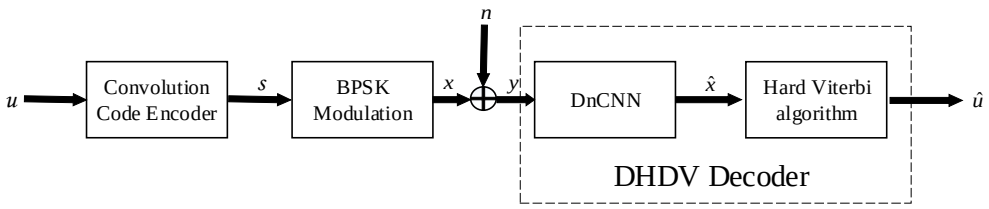


Figure 2: The channel decoding structure.

estimation for beamspace mmWave massive MIMO system and thus presents excellent performance (H. He et al, 2018). The deep learning-based decoding of polar codes shows a competitive performance (T. Gruber et al, 2017; S. Cammerer et al, 2017). Fei Liang et al (2018) design an iterative BP-CNN architecture for decoding LDPC code under correlated noise and use CNN network to estimate the noise efficiently. In addition, deep learning is applied to communication over the air (S. Dörner et al, 2018) and enables the optimization of the transceiver in a single end-to-end process (B. Karanov et al, 2018).

The early studies do not provide an effective solution to the performance degradation problem of traditional convolutional code decoding algorithms under correlated noise. In this paper, we design the DHDV decoder to improve the decoding performance.

3 SYSTEM MODEL

In this section, we introduce the DHDV decoder in detail. In order to show our system clearly, we will present our system framework firstly. Besides, the noise correlation model is explained in order to demonstrate our system model comprehensively. Then, we will explain why DnCNN can work in DHDV decoder. Finally, we will introduce how to train the network.

3.1 System Framework

Figure 2 illustrates our proposed architecture. At the transmitter, the convolution code is defined as (n,k,K) code with k input bits, n output bits, and K constraint length. The uniformly distributed source bits u of length k is encoded into target bits s of length n through the boolean polynomial function. Then, the target bits s are mapped to a symbol vector x through BPSK modulation. The symbols will pass through the Gaussian additive noise channel.

The convolutional codes such as (2, 1, 7) and (3, 1, 7) convolutional codes have small receive dimensions, so we apply the temporal dimension to the correlated noise in our model. The correlated noise is denoted as

$$n_c = n_w C^{1/2} \tag{2}$$

where n_w is an $n \times M$ matrix of independent identically distributed, standard Gaussian random variables. To keep the power of noise constant, $(1/M)trace\{C\}=1$ is required. We use the standard noise correlation model mentioned in the literature (S. K. Sharma et al, 2013). The correlation matrix C is given by:

$$C_{ij} = \begin{cases} c^{j-i}, & i \le j \\ (c^{i-j})^*, & i \ge j \end{cases} \tag{3}$$

where C_{ij} is the (i, j) th element of C and c is the correlation coefficient with $|c| \le 1$.

At the receiver, the symbols are obtained by

$$y = x + n_c \tag{4}$$

Then, the DnCNN model filters the noise from y according to the noise correlation. The DnCNN estimate the noise, and y subtract it to obtain the estimation of x, which is denoted as $\hat{x}$. The symbol $\hat{x}$ will be judged as $\hat{s}$ just consist of 0 or 1 through the hard decision. At last, we can use the Viterbi algorithm to decode for the source code.

We regard the data from the channel as a 2D image, and not only the correlation between the convolutional code symbols but also the correlation between the noise signals is well suited for the feature extraction.

3.2 Training

In our design for the DnCNN, the input matrix size keeps same as the length of the decoding window of the Viterbi decoder, designed as $n \times 50$ matrix. We

train the DnCNN model to perform well before putting it into the DHDV architecture.

Many parameters influence the performance of the DnCNN. The parameters optimization dealing with the optimization of neural work is called hyper-parameter optimization (J. S. Bergstra et al, 2011). In this paper, we do not further consider the optimization and choose general parameters to achieve the performance of the DHDV decoder. The network is initialized using random initialization. We choose mean squared error (MSE) as the loss function to train the DnCNN. The goal of the training process is to minimize the loss.

$$f_\ell(u,v) = |u-v|_2^2 \tag{5}$$

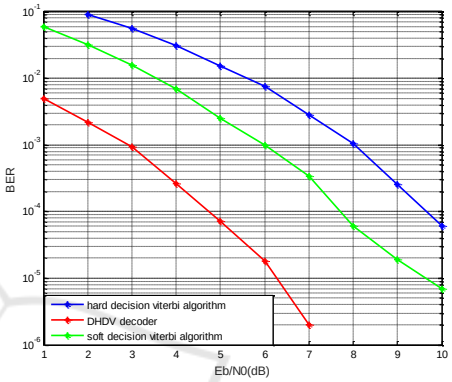
The training data needs to be generated before training the DnCNN network. In the practical use for network training, the validation data is generated to test the capability of network and void overfitting. We use single SNR (measured as E_b / N_0) training data to train the network. Our training of DnCNN network is implemented in Keras (Manaswi and Kumar N, 2018). The neural network runs on fast concurrent GPU architecture, which speeds up learning. After choosing $\{1,2,3,4,5,6\}dB$ training data to train the network individually, we found training model based on $3dB$ training data presents slightly better performance comparing the loss value on the uniform validation data containing different SNRs. We should emphasize that this is a reasonable but not best way to select a training model. We just provide a simple basis for selecting model.

Table 1: DnCNN SETTING FOR EXPERIMENTS

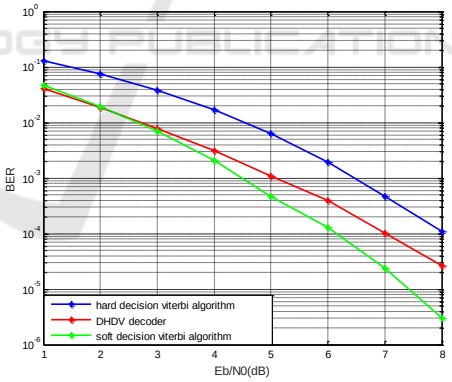
DnCNN input data size	$n \times 50$
Kernel size	2×3
Total parameter number	15261
Trainable parameter number	15021
Mini-batch size	500
Num of the training data	1000000
Num of the validation data	100000
SNR for generating the training data	3dB
Initialization method	Random initialization
Optimization method	Adam optimization

We use the mini-batch gradient descent method to train the network. Each mini-batch contains 500 blocks of training data and the network updates the parameters one time based on one mini-batch. We adopt Adam optimization (D. Kingma and J. Ba, 2015) method to adjust the learning rate adaptively for training. We check the loss value for each epoch, in which the gradient of the loss function is calculated over the entire training set. In addition, the early stopping (training will stop if the loss of validation does not drop for a consecutive period of time) is used to against the overfitting. The DnCNN setting is summarized in Table 1.

4 SIMULATION RESULTS



(a) $c = 0.8$



(b) $c = 0.5$

Figure 3: Performance of DHDV decoder for (2, 1, 3) convolutional code.

In this section, we present simulated and analytical results of the DHDV decoder based on the trained DnCNN network. Firstly, we get the DHDV decoder performance measured by the BER and the BER performance of traditional Viterbi algorithm is given in the simulation results. Then, we compare the complexity of DnCNN network and matrix

multiplication whitening method. The four parts following present the results in detail.

4.1 Performance Based On (2,1,3) Convolutional Code

We first compare the performances of the DHDV decoder and the Viterbi Algorithm decoder based on general (2, 1, 3) convolutional code. Our simulation results are based on two correlation parameters: $c = 0.8$ means a relatively strong correlation model and $c = 0.5$ means a moderate correlation model. In addition, we test the results under an AWGN channel without any correlation ($c = 0$).

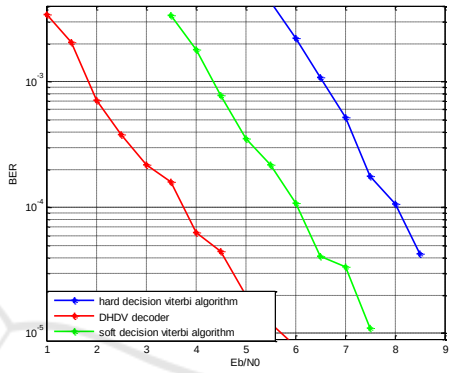
The experiments results are shown in Figure 3. We can see that the DHVD decoder totally achieves enhanced performance in different parameters *c*. In case where $c = 0.8$, the DHVD decoder can improve the decoding performance by approximately $4.5dB$ at $BER = 10^{-4}$ comparing hard decision Viterbi algorithm decoder. Comparing the performance with the soft decision Viterbi algorithm decoder, the DHVD decoder can also improve the decoding performance by approximately $3dB$ at $BER = 10^{-4}$. In the case where $c = 0.5$, at moderate correlation model, the performance improvement becomes smaller than that where $c = 0.8$, because the noise correlation is weaker and DnCNN model has degraded performance for denoising.

4.2 Performance under Different Convolutional Codes

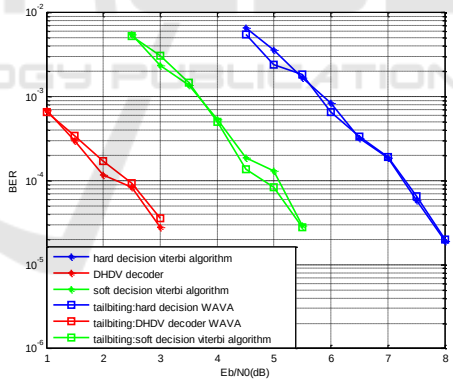
Then, we choose (2, 1, 7), (3, 1, 7) convolutional codes and (3, 1, 7) tail-biting convolutional code to test the performance of DHDV decoder. The (2, 1, 7) convolutional code has longer constraint length. The (3, 1, 7) convolutional code has different code rate. As for (3, 1, 7) tail-biting convolutional code, we choose the Wrap Around Viterbi Algorithm (WAVA) as the Viterbi algorithm decoder to present the performance.

The simulation results of different convolutional codes illustrate the robustness of the DHDV decoder under strong correlation $c = 0.8$ in Figure 4. The DHVD decoder improves the decoding performance by more than $4dB$ at $BER = 10^{-4}$ comparing hard decision Viterbi algorithm decoder and more than 2 dB at $BER = 10^{-4}$ comparing soft decision Viterbi algorithm for (2, 1, 7) convolutional code, which presents slightly lower improvement extent than (2,1,3) convolutional codes.

Moreover, as for (3, 1, 7) convolutional code, the BER capability achieves greater improvement comparing (2, 1, 3) and (2, 1, 7) convolutional code, improved by approximately $5dB$ at $BER = 10^{-4}$ comparing hard decision Viterbi algorithm decoder and approximately $3dB$ at $BER = 10^{-4}$ comparing soft decision Viterbi algorithm decoder. As for WAVA decoder, we choose 3 as the number of the max iterations. The (3, 1, 7) tail-biting convolutional code gets the almost the same performance improvement as general (3, 1, 7) convolutional code.



(a) (2, 1, 7) convolutional code



(b) tailbiting and non-tailbiting (3, 1, 7) convolutional code

Figure 4: Performance of DHDV decoder under different convolutional codes where $c = 0.8$.

The BER capability improvement of DHDV decoder depends on the benefits from the DnCNN denoising. From the results above, we can conclude that the longer constraint length of the convolutional code suppresses the potential of performance enhancement for DHDV decoder. However, the smaller code rate has the opposite effect in this

respect. As for the smaller code rate with the same length of source bits, it has longer target bits, so the DnCNN can get a better effect on denoising.

4.3 Robustness under Different Correlation Noise Models

So far, our simulation results are based on the noise correlation model defined in (2) and (3). In this section, we will test the robustness of DHDV decoder under another noise correlation model, which is defined as $n_c = n_w \Phi$. We apply Symbol interference model with weighted values to the noise model, and consider the number of adjacent interference symbols as 10, so the correlation matrix is given by:

$$\Phi_{ij} = \begin{cases} c^{|i-j|}/p, & if\ |i-j| \le 10 \\ 0, & else \end{cases} \quad (6)$$

In (6), c is the correlation coefficient with $|c| \le 1$ and normalization of noise power is done through dividing p.

$$p = \sqrt{\sum_{i,j}^{|i-j| \le 10} c^{|i-j|_2^2}} \quad (7)$$

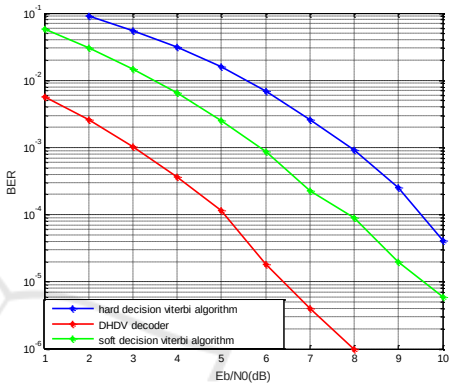
As for the new correlation model, the methods still work by retraining the DnCNN network. Figure 5 presents the performance of (2, 1, 3) convolutional code under the new noise correlation model. The BER capability of DHDV decoder is close to it under standard noise correlation model in (2) and (3). It should be emphasized that DHDV decoder can extract features efficiently for different situations, so it achieves great BER capability under different noise models.

4.4 Computational Complexity

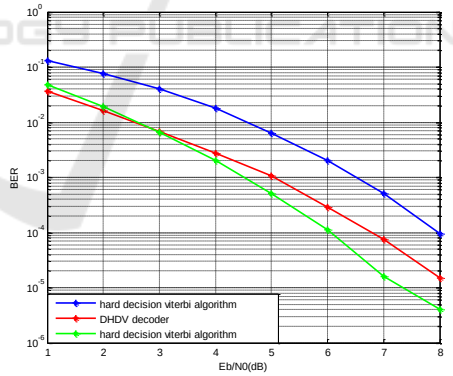
In this section, we will compare the computational complexity of DnCNN and traditional whitening methods. In general, matrix multiplication is used for whitening noise. We denote the receive data as an $n \times M$ matrix. Then in the whitening process, the received data needs to be multiplied by a left matrix of $n \times n$, and the matrix of right multiplication by $M \times M$. It is easy to get the computational complexity as $O(n^2 M + M^2 n)$.

In addition to the size of the input data, the computational complexity of DnCNN depends on the number of convolution kernels k (20 in our experiment), the convolution kernel size $a \times b$ (2×3 in our experiment), and the number of convolutional layers v (6 in our experiment). The computational complexity of DnCNN is $O(n * M * k^2 * a * b * v)$.

In general, the number of convolution kernels k, the convolution kernel size $a \times b$, and the number of convolutional layers v are fixed and relatively small. Comparing the computational complexity above, we can conclude that when the input data dimension is small, the model complexity of DnCNN is greater than the whitening computational method. Conversely, when the input data dimension is large, the whitening computational complexity will be greater than the DnCNN computational method.



(a) $c = 0.8$



(b) $c = 0.5$

Figure 5: Performance of DHDV decoder under another correlation model for general (2, 1, 3) convolutional code.

5 CONCLUSIONS

In this paper, we apply DnCNN network to the decoder of the convolutional codes and propose a DHDV decoder for correlated noise. The DHDV

decoder presents an excellent performance for different kinds of convolutional codes and correlated noise models. From the simulation results, we also found the constraint length and code rate of convolutional codes will affect the BER capability. Comparing the complexity of DnCNN network and matrix multiplication whitening method, DnCNN network complexity is lower when the code length is longer; DnCNN network complexity is higher when the code length is shorter.

Furthermore, future investigations will be based on the hyperparameter optimization and DHDV decoder complexity optimization.

ACKNOWLEDGEMENTS

This work is supported in part by the National Natural Science Foundation of China under Grant No. 61571065 and the Beijing Natural Science Foundation (L161005).

REFERENCES

S. K. Sharma, S. Chatzinotas and B. Ottersten., 2013. SNR Estimation for Multi-dimensional Cognitive Receiver under Correlated Channel/Noise. IEEE Transactions on Wireless Communications, 12(12), pp. 6392-6405.

L. Di Bert , P. Caldera, D. Schwingshackl, and A. M. Tonello., 2011. On noise modeling for power line communications. Proc. IEEE Int. Symp. Power Line Commun. Appl, pp. 283–288.

B. Karanov et al., 2018. End-to-End Deep Learning of Optical Fiber Communications. Journal of Lightwave Technology, 36(20): 4843-4855.

J. D. Wang and H. Y. Chung., 1990. Trellis coded communication systemscolored noise and the swapping technique. IEEE Trans.Commun, 38(9), pp. 1549–1556.

T. Ishihara and S. Sugiura., 2016. Frequency-domain equalization aided iterative detection of faster-than-Nyquist signaling with noise whitening. 2016 IEEE International Conference on Communications (ICC), pp. 1-6.

K. He, X. Zhang, S. Ren, and J. Sun., 2016. Deep residual learning for image recognition. Proc. IEEE Conf. Comput. Vis. Pattern Recognit, pp. 770–778.

I. Sutskever, O. Vinyals, and Q. V. Le., 2014. Sequence to sequence learning with neural networks. Proc. Adv. Neural Inf. Process. Syst, pp. 3104–3112.

T. Wang, C. K. Wen, H. Wang, F. Gao, T. Jiang and S. Jin., 2017. Deep learning for wireless physical layer: Opportunities and challenges. China Communications, 14(11), pp. 92-111.

T. O'Shea and J. Hoydis., 2017. An Introduction to Deep Learning for the Physical Layer. IEEE Transactions on Cognitive Communications and Networking, 3(4), pp. 563-575.

K. Zhang, W. Zuo, Y. Chen, D. Meng and L. Zhang., 2017. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. IEEE Transactions on Image Processing, 26(7), pp. 3142-3155.

H. He, C. K. Wen, S. Jin and G. Y. Li., 2018. Deep Learning-based Channel Estimation for Beamspace mmWave Massive MIMO Systems. IEEE Wireless Communications Letters, 7(5), pp. 852-855.

T. Gruber, S. Cammerer, J. Hoydis and S. t. Brink., 2017. On deep learning-based channel decoding. 2017 51st Annual Conference on Information Sciences and Systems (CISS), pp. 1-6.

S. Cammerer, T. Gruber, J. Hoydis and S. ten Brink., 2017. Scaling Deep Learning-Based Decoding of Polar Codes via Partitioning. 2017 IEEE Global Communications Conference, pp. 1-6.

F. Liang, C. Shen and F. Wu., 2018. An Iterative BP-CNN Architecture for Channel Decoding. IEEE Journal of Selected Topics in Signal Processing, 12(1), pp. 144-159.

S. Dörner, S. Cammerer, J. Hoydis and S. t. Brink., 2018. Deep Learning Based Communication Over the Air. IEEE Journal of Selected Topics in Signal Processing, 12(1), pp. 132-143.

B. Karanov et al., 2018. End-to-End Deep Learning of Optical Fiber Communications. Journal of Lightwave Technology, 36(20), pp. 4843-4855.

J. S. Bergstra, R. Bardenet, Y. Bengio, and B. K´egl., 2011. Algorithms for hyper-parameter optimization. Advances in Neural Information Processing Systems, pp. 2546–2554.

Manaswi, Kumar N., 2018. Deep learning with applications using python ‖ understanding and working with keras. Springer, 10.1007/978-1-4842-3516-4(Chapter 2), 31-43. Avaliable at: onacademic.com

D. Kingma and J. Ba., 2015. Adam: A method for stochastic optimization. Proc. Int. Conf. Learn. Represen.