On the Efficient Design of Neural Networks in Communication Systems

Weihong Xu[1,2,3], Xiaosi Tan[1,2,3], Zaichen Zhang[2,3], Xiaohu You[2,3], Chuan Zhang[1,2,3], and Yair Be'ery[4]
[1]Lab of Efficient Architecture for Digital Communication and Signal Processing (LEADS)
[2]Purple Mountain Laboratories, Nanjing, China
[3]National Mobile Communications Research Laboratory, Southeast University, Nanjing, China
[4]School of Electrical Engineering, Tel-Aviv University, Israel
Email: [3]{wh.xu, chzhang}@seu.edu.cn, [3]ybeery@eng.tau.ac.il

***Abstract*—Recently, various types of neural networks (NNs) have shown promising performance in communication systems. However, the low-latency implementation of these tasks is currently impractical due to the high computational complexity and large model size of NNs. In this paper, we propose an iterative optimization framework with retraining process to adaptively find the quantization scheme for different NNs. Moreover, the efficient design of convolutional neural networks is presented to reduce the required parameters and computational complexity. Experiment results for modulation classification, channel decoder and equalizer are presented. Compared to the original full-precision models, the quantized NN models achieve comparable performance with only 4 to 5 weight bits and 8-bit activation. The size of optimized models is significantly compressed and the hardware complexity of the NN inference is also reduced.**

***Index Terms*—Neural networks, quantization, deep learning, communication, depthwise separable convolution.**

I. Introduction

Due to the great success of deep learning (DL) techniques, the variants of neural networks (NNs), including deep neural networks (DNNs) and convolutional neural networks (CNNs), have been used to improve the quality of communication systems. The applications of NNs [1–8] in communication systems have shown promising performance and flexibilities compared to conventional algorithms. The unified DNNs or CNNs model can be applied to carry out different tasks.

DNNs [1] can work as the decoder for polar codes [9] and achieve near-optimal error correction performance for short codes. Dorner *et al.* propose a deep learning-based end-to-end communication system [2] which adopts DNNs as the transmitter and receiver. Kim *et al.* [3] demonstrate that DNN-based sparse code multiple access (SCMA) detector can achieve similar performance to the conventional message-passing algorithms (MPA). Besides, the gain of DNNs is also observed on MIMO detection [4] and hybrid precoding [5].

As for CNNs, they can process the high-dimensional signals with features. CNNs can classify the modulation of received signals [6] without any expert knowledge. Xu *et al.* [7] use 1-D CNNs to cancel the inter-symbol interference (ISI) introduced by channel and the results show that CNNs outperform the conventional equalizer based on machine learning algorithms. Liang *et al.* [8] cascade a CNN denoiser after LDPC decoder to reduce the effects of correlated noise. The mentioned applications of NNs in communication systems are summarized in Table I.

TABLE I
Existing applications of NNs in communication systems

Networks	Applications	Year
DNN	Decoder [1]	2017
	Transceiver [2]	2018
	SCMA [3]	2018
	MIMO detection [4]	2019
	Precoding [5]	2019
CNN	Modulation classification [6]	2016
	Channel equalization [7]	2018
	Channel denoiser [8]	2018

However, the direct implementation of NNs on low-power and area-constraint embedded systems is impractical. There are several reasons that hinder the deployment of NNs. First, DNNs are a type of memory-intensive algorithms since the fully-connected (FC) layers require large amounts of weights. Second, CNNs are computation-intensive because the convolution operation requires substantial multiply-accumulate (MAC) operation. Moreover, the training and inference of existing NN models are usually performed on GPUs with floating-point arithmetic. The embedded devices generally adopt fixed-point arithmetic instead of floating-point arithmetic for the sake of power efficiency and area constraints.

There are several works [10, 11] implementing NN-based communication algorithms onto field-programmable gate arrays (FPGAs) platform. 32-bit fixed-point quantization is adopted in [10] while Kim *et al.* [11] use 16 bits for the DNN-based encoder and decoder. Furthermore, the low bit-width quantization of NN-based receiver is studied in [12], where 4-bit weights and 14-bit activations are used.

In this work, we present an iterative method to quantize and optimize low bit-width NN models for communication systems. Combined with the evaluation metric and quantization-aware retraining process, the proposed methods can effectively search the quantization parameters that result in minimum performance degradation. Moreover, the efficient design for 1-D CNNs is also presented to reduce the required parameters and computational complexity without performance loss. The experiments are conducted on various NN-based communica-

tion algorithms to show the feasibility of proposed methods.

The remainder of this paper is organized as follows. Background of DNNs and CNNs are briefly introduced in Section II. Section III first describes the quantization scheme and the iterative optimization methods. Then the efficient design for CNNs is described in detail. The experiment results on various NN models are shown in Section IV. Finally, Section V concludes this paper.

II. Preliminaries

A. Deep Neural Networks

In this paper, we consider the DNNs composed of several FC layers in the feed-forward pattern. Each FC layer computes the linear combination of given input vector $\mathbf{x}$ with the weights matrix $\mathbf{W}$. The computation of a single FC layer in DNNs can be shown as the following matrix-vector multiplication:

$$\mathbf{Y}_j = f(\sum_{i=1}^{I} \mathbf{W}_{j,i}\mathbf{x}_i + \mathbf{b}_j), \tag{1}$$

where $\mathbf{W}$ is the weights matrix corresponding to the input $\mathbf{x}$. $f(\cdot)$ denotes the activation function. The rectified linear unit (ReLU) $f(x) = \max(x, 0)$ or the sigmoid activation $f(x) = (1 + \exp^{-x})^{-1}$ can be adopted as the activation function depending on the specific task.

B. Convolutional Neural Networks

CNNs have similar feed-forward architectures with DNNs. The basic layer in CNNs is called convolution (CONV) layer. For an input feature map with multiple channels, the CONV layer performs two-dimensional (2-D) convolution on input feature map and several filter kernels with shape $K \times K$. The output feature map of a CONV layer can be computed as the following equation:

$$\mathbf{O}_{m,i,j} = f(\sum_{c=1}^{C}\sum_{x=1}^{K}\sum_{y=1}^{K} \mathbf{W}_{m,c,x,y}\mathbf{x}_{c,i+x,j+y}), \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{M\times C\times K\times K}$ denotes the weights consisting of M filters in a layer, each filter containing a C-channel kernel with shape $K \times K$. Same as DNNs, $f(\cdot)$ here denotes the activation function.

III. Proposed Framework for Efficient Neural Networks

A. Quantization Scheme

The quantization is key to the efficient realization of NN-based communication algorithms because the quantization not only saves the required storage space of NN models but also reduces the arithmetic complexity. Considering that the fixed-point quantization is widely used in hardware implementation, the uniform fixed-point quantization is used throughout this paper. We linearly quantize the weights and activations of each layer into fixed-point format with 1-bit sign flag.

The training and inference of most existing NN models are carried out in floating-point format. To simulate the quantized arithmetic operations, a quantization module is added before the computation as in [13]. Fig. 1 shows a computation graph of quantized inference for convolution layers. The input feature map, weights, and bias are first quantized to fixed-point format. Then the quantized input, weights and bias are used to compute the convolution results. The output of activation unit is also quantized and the results are passed to the next layer. Likewise, DNNs have similar structures to simulate the quantized inference. It is noted that only the feed-forward inference is quantized whereas the back-propagation is still operating in floating-point format during the training process.

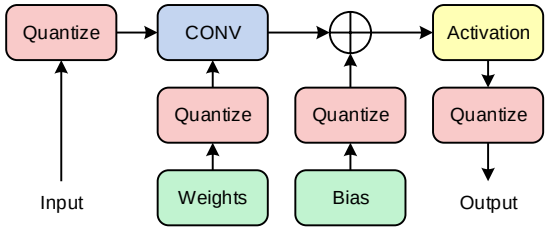

Fig. 1. Quantized computation of convolution layer for CNNs.

B. Iterative Optimization Method

After finishing the training process in floating point, the full-precision NN models can be directly quantized into fixed-point format. However, this will result in accuracy degradation when the quantization bit width is low [12]. We can train the quantized model from scratch to ensure the accuracy of NNs under low bit width. But it is computationally expensive to find a good quantization scheme since this process should be run for many times. To efficiently search the good quantization scheme, an iterative optimization method combined with quantization-aware retraining process is used. Fig. 2 illustrates the diagram of the iterative optimization method. The trained model that satisfies the desired accuracy is first quantized into fixed-point format that has relatively high precision, *e.g.* 16 bit. If the quantized model satisfies a specific metric, the quantization bits will be reduced. Otherwise, the quantized model will be retrained to restore accuracy. This procedure is iterating until the retraining process cannot provide a satisfactory model.

Similar to the *normalized validation error* (NVE) in [1], the following metric *normalized quantization error* (NQE) is adopted:

$$\text{NQE} = \frac{1}{S}\sum_{i=1}^{S}\frac{\text{BER}_{\text{quant}}(\rho_{v,s})}{\text{BER}_{\text{float}}(\rho_{v,s})}, \tag{3}$$

where $\text{BER}_{\text{float}}(\rho_{v,s})$ and $\text{BER}_{\text{quant}}(\rho_{v,s})$ represent the validation bit-error rate (BER) performance of floating-point model and quantized model at the s-th SNR denoted by $\rho_{v,s}$. S is the total point of tested SNRs. The NVE metric in [1] is to define how good a neural network trained at a specific SNR is compared to the optimal model. In comparison, the proposed NQE metric in Eq. (3) is used to evaluate the performance gap between the full-precision model and the quantized model within a SNR range. NQE ≤ 1 implies we obtain a quantized

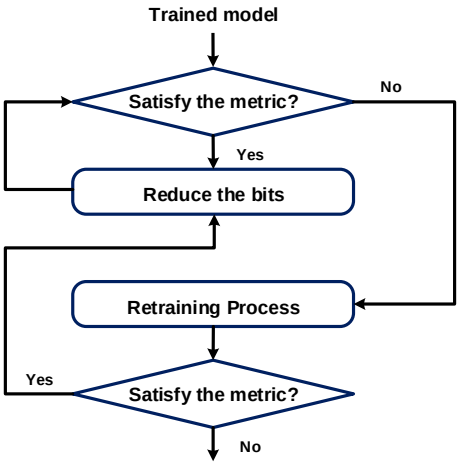

Fig. 2. Diagram of iterative quantization and optimization steps.

model with superior performance of full-precision counterpart. For the case where slight performance degradation is allowed, the NQE can be set to a value greater than 1.

C. Efficient Convolutional Neural Networks

The standard convolution of CNNs in Eq. (2) convolves the convolutional kernels with input features to generate new output features. The processed signals in communication systems are usually 1-D sequences. In this case, the filter kernels $\mathbf{W}$ are an $M \times C \times K$ tensor. The number of parameters in a layer is $M \cdot C \cdot K$ without the bias. Assuming the input sequence with length N has C channels and the output sequence has the same length with M channels, the computational complexity of MAC operation is $C \cdot M \cdot N \cdot K$.

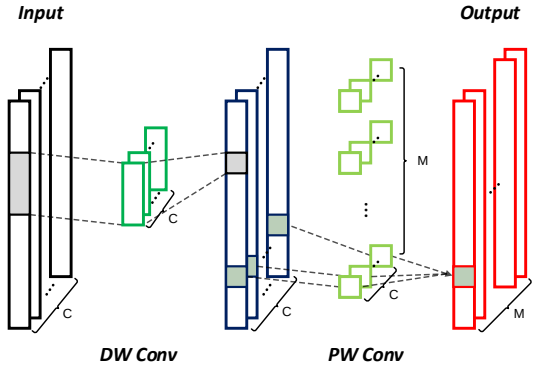

Fig. 3. Illustration of 1-D depthwise convolution and pointwise convolution.

As suggested in [14], the parameters and computational complexity of standard 2-D convolution can be greatly reduced by adopting depthwise separable convolution. We reformulate the 2-D depthwise separable convolution into the 1-D case. First, each channel of the input features convolves with a filter separately as follows:

$$\hat{\mathbf{O}}_{c,i} = f(\sum_{c=1}^{C}\sum_{x=1}^{K} \mathbf{W}_{c,x}\mathbf{x}_{c,i+x}), \tag{4}$$

where the intermediate results $\hat{\mathbf{O}}$ have the identical dimension with the input features. Then $\hat{\mathbf{O}}$ will convolve with M filters with C channels and 1×1 shape to produce the output features. The required number of parameters becomes $C\cdot(K+M)$ and the MAC complexity becomes $C \cdot N \cdot (K+M)$. Compared to the standard convolution, the reduction ratio of parameters and computational complexity is $\frac{M\cdot K}{K+M}$.

IV. Experiment Results

In this section, we study the effects of quantization in various NN models for communication systems. The efficient designs in Section III are also investigated.

A. Results on Modulation Classification Task

For the modulation classification task, the modulation dataset RadioML 2016.10a [15] generated by GNU Radio [16] is used. This dataset also includes a number of realistic channel imperfections such as channel frequency offset, sample rate offset, additive white Gaussian noise with multipath fading. It contains modulated signals with 4 samples/symbol and a sample length of 128 samples. RadioML 2016.10a dataset contains in total of $220,000$ signal samples. In the experiment, a 5-layer network with 3 CONV layers and 2 FC layers is used. The other experiments setups are summarized in Table II.

TABLE II
Experiments Setup for Modulation Classification

Parameters	Value
SNR Range	-20 dB to $+18$ dB
Optimizer	Mini-batch SGD
Learning Rate	0.001
Training mini-batch Size	1000
Training Set Size	$110,000$
Validation Set Size	$110,000$

Since the NQE metric is not suitable for this task, we train the model from scratch under different quantization schemes. The model under floating-point quantization is regarded as the baseline. The training loss for CNNs under different quantization schemes with 8-bit activation is shown in Fig. 4. It shows that 4-bit and 6-bit weights quantization can guarantee the loss close to the full-precision baseline while the losses of 2-bit and 3-bit weight quantization are much larger than the baseline. This suggests 2-bit and 3-bit quantization cannot provide enough arithmetic precision to ensure the accuracy.

We also test the classification accuracy for various quantization schemes. As shown in Fig. 5, 4-bit and 6-bit weight quantizations have negligible accuracy degradation compared to the baseline when SNR ≤ 0 dB. There are about 2% to 4% accuracy loss for SNR > 0 dB. Lower bit-width quantizations, such as 3 bits and 2 bits, introduce severe degradation.

B. Results on Neural Network Decoder

The NN-based polar decoder [1] is considered in the experiment. A 3-layer DNN with shape 128-64-32 is used. The models are trained at fixed $E_b/N_0 = 1$ dB and the mini-batch size is 256. For the fair comparison, other training

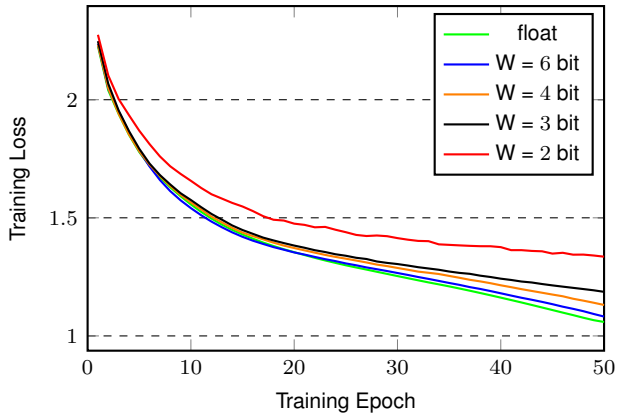

Fig. 4. Training loss under different quantizations with 8-bit activation.

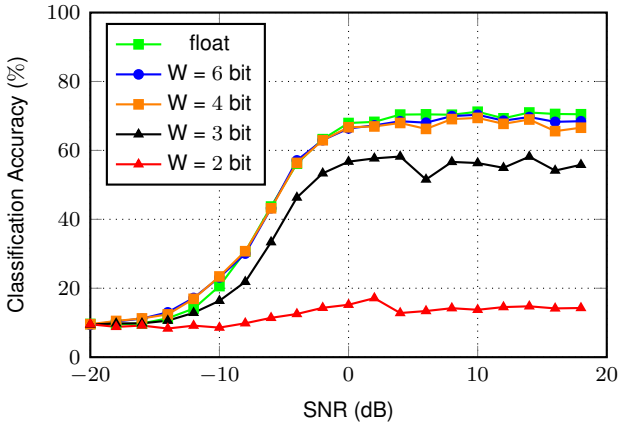

Fig. 5. Classification accuracy comparison for different quantization schemes.

configurations are the same as [1]. To reduce the search space of quantization bits, the initialized weight bits is set to 8 and the quantization bits for activation are fixed at 8.

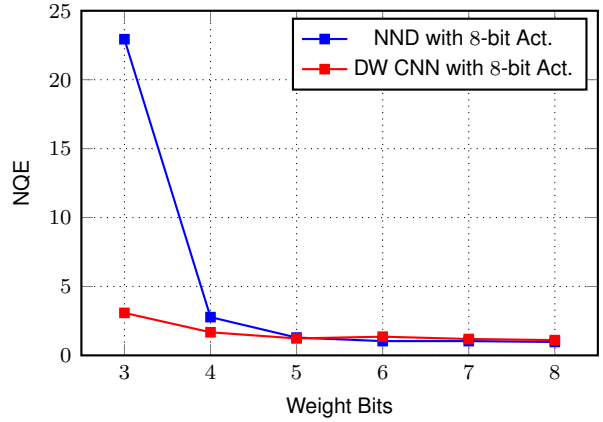

Fig. 6. NEQ for different networks and quantization configurations.

Fig. 6 illustrates the values of NQE metric under different quantization bits. The activation bits are set to 8 while the weight bits are searched by the iterative optimization methods in Section III. The NQE metric has a value of less than 2 when weight bits ≥ 5, which means the BER degradation caused by quantization is acceptable.

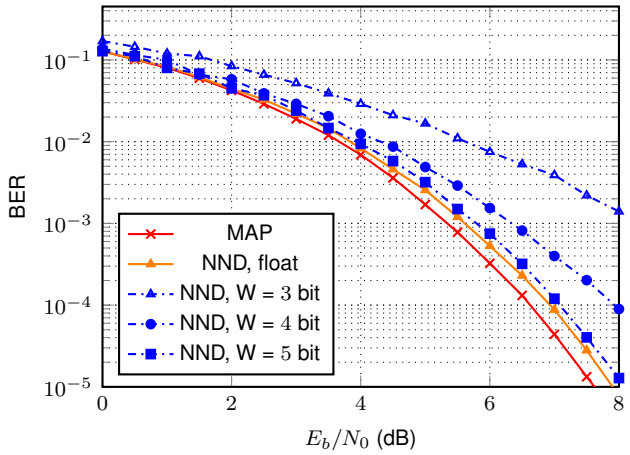

Fig. 7. BER comparison of NNDs with different quantization configurations for $(16, 8)$ polar codes. The activation bits are 8.

The BER comparison between quantized NND, full-precision NND, and the maximum a posteriori (MAP) decoding is shown in Fig. 7. It is observed that the NND with 5-bit weights and 8-bit activation quantization achieves comparable performance with the full-precision baseline.

C. Results on CNN Channel Equalizer

We also study the CNN equalizer in [7]. The dispersive channel with AWGN noise and ISI in [7, 17] is used. The channel impulse response in Eq. (5) is considered.

$$H(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}. \tag{5}$$

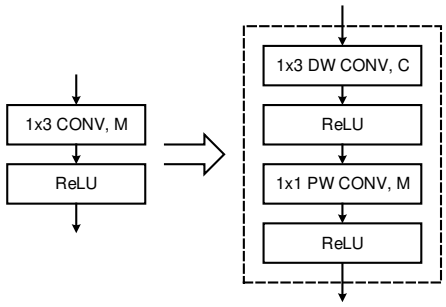

Fig. 8. 1-D mapping of standard convolution into depthwise separable convolution. The input features have C channels.

On the basis of the six-layer CNN of [7] with structure $\{6, 12, 24, 12, 6, 1\}$ (the value denotes 1×3 filter number for each layer), we convert each convolution layer into its equivalent depthwise separable convolution other than the first and last layers. Fig. 8 illustrates the mapping of 1-D standard convolution layer with C-channel input into the corresponding 1-D depthwise separable convolution. As shown in Fig. 9, the resulting DW CNN model only requires 46% parameters and 41% computational complexity compared to the original one. Most of 1×3 kernels and convolution operation are replaced by 1×1 kernels and point-wise multiplication.

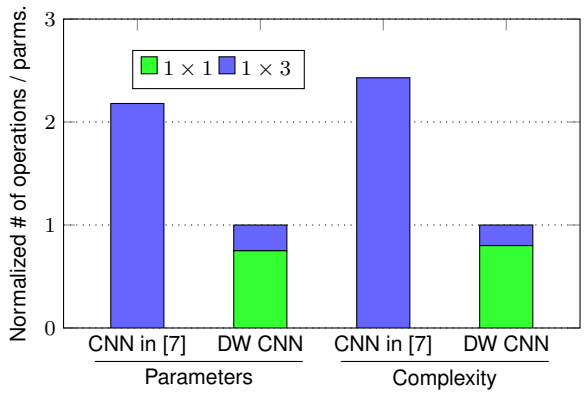

Fig. 9. Comparison of parameter and computational complexity between standard CNN [7] and optimized DW CNN.

To evaluate the performance of depthwise separable convolution, the DW CNN equalizer is trained with the same configurations in [7]. The resulting BER performance is depicted in Fig. (10), where the ML equalizer with perfect channel state information (CSI) and the ML-BCJR algorithms [17] are taken as the baselines. The DW CNN equalizer achieves almost the same performance with the CNN equalizer and outperforms the ML-BCJR algorithms. Under 8-bit activation, the NQE values for different weight bits are shown in Fig. (6). The performance of quantized DW CNN with 8-bit activation is also given in Fig. 10. It shows that 5-bit weights can ensure the BER performance near to the full-precision model.

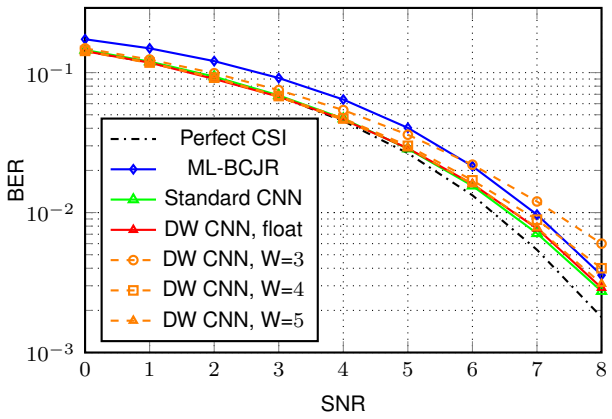

Fig. 10. BER comparison of various channel equalizers. The activation bits for quantized DW CNN are 8.

V. Conclusion

In this work, the quantization and efficient design of NNs in communication systems are studied. An iterative optimization method with quantization-aware retraining is proposed to efficiently search quantization parameters. Besides, the efficient design of 1-D standard CNNs is also discussed based on depthwise separable convolution. The results show that several existing DNNs and CNNs yield comparable performance with only 4 to 5 bits of weight and 8-bit activation. Moreover, the 1-D DW CNN for channel equalization achieves 46% parameters and 41% complexity reduction without performance loss. The future work will focus on further optimizations of NN models in communication systems.

Acknowledgement

This work is supported in part by NSFC under grants 61871115 and 61501116, Jiangsu Provincial NSF for Excellent Young Scholars under grant BK20180059, the Six Talent Peak Program of Jiangsu Province under grant 2018-DZXX-001, the Distinguished Perfection Professorship of Southeast University, the Fundamental Research Funds for the Central Universities, the SRTP of Southeast University, and the Project Sponsored by the SRF for the Returned Overseas Chinese Scholars of MoE.

References

[1] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," in *Annual Conference on Information Sciences and Systems (CISS)*, 2017, pp. 1–6.

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

[3] M. Kim, N.-I. Kim, W. Lee, and D.-H. Cho, "Deep learning-aided SCMA," *IEEE Communications Letters*, vol. 22, no. 4, pp. 720–723, 2018.

[4] N. Samuel, A. Wiesel, and T. Diskin, "Learning to detect," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2554–2564, 2019.

[5] H. Huang, Y. Song, J. Yang, G. Gui, and F. Adachi, "Deep-learning-based millimeter-wave massive mimo for hybrid precoding," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 3027–3032, 2019.

[6] T. J. O'shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *International Conference on Engineering Applications of Neural Networks*. Springer, 2016, pp. 213–226.

[7] W. Xu, Z. Zhong, Y. Be'ery, X. You, and C. Zhang, "Joint neural network equalizer and decoder," in *International Symposium on Wireless Communication Systems (ISWCS)*, 2018, pp. 1–5.

[8] F. Liang, C. Shen, and F. Wu, "An iterative bp-cnn architecture for channel decoding," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 144–159, 2018.

[9] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[10] Z.-L. Tang, S.-M. Li, and L.-J. Yu, "Implementation of deep learning-based automatic modulation classifier on fpga sdr platform," *Electronics*, vol. 7, no. 7, 2018.

[11] M. Kim, W. Lee, J. Yoon, and O. Jo, "Building encoder and decoder with deep neural networks: On the way to reality," *arXiv preprint arXiv:1808.02401*, 2018.

[12] F. A. Aoudia and J. Hoydis, "Towards hardware implementation of neural network-based communication algorithms," *arXiv preprint arXiv:1902.06939*, 2019.

[13] B. Jacob, S. Kligys, B. Chen, Zhu *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2704–2713.

[14] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[15] T. J. O'shea and N. West, "Radio machine learning dataset generation with gnu radio," in *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016.

[16] "GNU Radio," http://www.gnuradio.org.

[17] L. Salamanca, J. J. Murillo-Fuentes, and F. Pérez-Cruz, "Channel decoding with a bayesian equalizer," in *IEEE International Symposium on Information Theory*, 2010, pp. 1998–2002.