This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

Deep Learning based End-to-End Wireless Communication Systems without Pilots

Hao Ye, Geoffrey Ye Li, and Biing-Hwang Fred Juang

***Abstract*—The recent development in machine learning, especially in deep neural networks (DNN), has enabled learning-based end-to-end communication systems, where DNNs are employed to substitute all modules at the transmitter and receiver. In this article, two end-to-end frameworks for frequency-selective channels and multi-input and multi-output (MIMO) channels are developed, where the wireless channel effects are modeled with an untrainable stochastic convolutional layer. The end-to-end framework is trained with mini-batches of input data and channel samples. Instead of using pilot information to implicitly or explicitly estimate the unknown channel parameters as in current communication systems, the transmitter DNN learns to transform the input data in a way that is robust to various channel conditions. The receiver consists of two DNN modules used for channel information extraction and data recovery, respectively. A bilinear production operation is employed to combine the features extracted from the channel information extraction module and the received signals. The combined features are further utilized in the data recovery module to recover the transmitted data. Compared with the conventional communication systems, performance improvement has been shown for frequency-selective channels and MIMO channels. Furthermore, the end-to-end system can automatically leverage the correlation in the channels and in the source data to improve the overall performance.**

***Index Terms*—Wireless communications, pilot-free end-to-end communications, joint source channel coding.**

I. Introduction

The development of modern wireless communications has significantly improved our daily life. A full-fledged successful communication system takes advantage of various technical advances and realizes them in respective modules. As shown in Fig. 1(a), a chain of signal processing modules are designed in both the transmitter and the receiver to compensate for the channel effects and mitigate interference so that the data can be reliably transmitted from the transmitter to the receiver. However, these modules are often developed and optimized individually and the overall performance optimality may not be duly achieved. Moreover, the channel characteristics change with a variety of factors, *e.g.*, the operating frequency and the propagation environment. As a result, some of these modules should be designed to be adaptive to the varying environment in order to obtain the best performance, which is often a difficult task.

This work was supported in part by the National Science Foundation under Grants 1815637 and 1731017.

H. Ye and B-H. F. Juang are with the School of Electrical and Computer Engineering, Georgia Institute of Technology. (email: yehao@gatech.edu; juang@ece.gatech.edu)

G. Y. Li is with the Department of Electrical and Electronic Engineering, Imperial College London. (email: geoffrey.li@imperial.ac.uk)

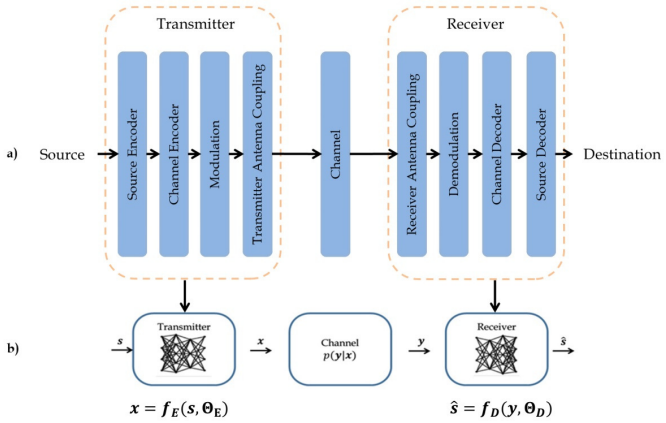

Fig. 1: Structures of (a) a conventional wireless communication system and (b) an end-to-end learning based communication system.

Recently, deep learning is reshaping the wireless communication system design, where the data-driven approaches can improve and complement the conventional model-based methods [1, 2, 3, 4, 5, 6]. End-to-end communication paradigms based on deep learning have been designed [2, 4, 7, 5] and have attracted increasing attention. For end-to-end communications, both the transmitter and the receiver are represented by deep neural networks (DNN) as shown in Fig. 1(b). The transmitter DNN learns to transform the transmitted data into an embedding vector and send over the wireless channel while the receiver DNN learns to recover the transmitted data from the received signals. The DNNs in the end-to-end pipeline are trained in a supervised manner to minimize an end-to-end loss, which measures the inconsistency between the recovered data and the transmitted data. Since it is a purely data-driven method with no pre-assumed channel models as a prerequisite, it can potentially provide a universal solution to various communication scenarios. Although the end-to-end paradigm has shown promising performance under additive white Gaussian noise (AWGN) channels [2], how to extend this framework to general wireless channels, such as frequency-selective channels and multi-input and multi-output (MIMO) channels, is still challenging.

Conventionally, a common practice to deal with the fading channels is to insert pilots in the data blocks, which are known to the receiver in advance. The receiver first estimates the current channel parameters with the help of the pilots and recovers the transmitted data via solving an inverse problem

with the estimated channel parameters. Nevertheless, this paradigm is not compatible with end-to-end communication systems. The usage of pilots complicates the design to a large degree since the receiver has to take two heterogeneous sources, *i.e.*, the received data and the estimated channel parameters, as input and solve an inverse problem without any prior knowledge. In the existing literature, the end-to-end learning system is designed under the conventional communication framework, such as orthogonal frequency-division multiplexing (OFDM) and singular value decomposition (SVD) based MIMO, which decomposes the channel into independent subchannels.

In this paper, we propose a deep learning based end-to-end communication system for general wireless channels, where the conventional modules, including source coding, channel coding, modulation, *etc.*, have been replaced with a DNN at the transmitter and one at the receiver. This problem is formulated as training a deep auto-encoder network with an untrainable stochastic convolutional layer, which represents the wireless channel. Convolutional neural networks (CNNs) are utilized in both the transmitter and the receiver and a training algorithm with mini-batches of input samples and channels is proposed. Instead of using pilots to implicitly or explicitly estimate the unknown channel parameters, the transmitter learns to encode the input data in a way that is robust to various channel conditions. The receiver consists of two DNN modules used for channel information extraction and data recovery, respectively. A bilinear production operation is employed to combine the features extracted from the channel information extraction module and the received signals. The combined features are further used in the data recovery module to recover the transmitted data. The proposed pilot-free end-to-end system is tested under two commonly-used wireless scenarios, *i.e.*, the frequency-selective channel and the flat-fading MIMO channel. From the experiments, the end-to-end system can successfully leverage a variety of correlation in a data-driven manner and achieve superior results. The main contributions of the paper are as follows,

- We develop a pilot-free end-to-end paradigm for general wireless channels, including frequency-selective channels and flat-fading MIMO channels, where the wireless channels are modeled as a stochastic convolutional layer.
- We design a channel feature extraction module, whose outputs are incorporated for data recovering via a bilinear production. In this way, the channel information can be efficiently utilized at each position for data reconstruction.
- We show the effectiveness of the pilot-free end-to-end communication system under the frequency-selective channels and flat-fading MIMO channels, showing especially its ability to save the pilot resources and leverage the correlation in wireless channels and source data.

Part of the work has been published in [8]. Compared with the previous work, we have extended the pilot-free system to structured data transmission, where the end-to-end framework outperforms the system with the source coding and the wireless transmission designed separately.

II. Related Works

In this section, we briefly review recent advances in deep learning for physical layer communications, deep learning for inverse problems, and adversarial layers in deep learning.

A. Deep Learning in Physical Layer Communication Systems

Deep learning has been used to enhance the traditional communication modules. Deep learning based approaches can improve the channel decoding performance [9, 10, 11, 12, 13]. Deep learning can also be utilized for channel estimation [14, 15] and signal detection [16, 17, 18]. More information on this topic can be found in [19, 20] and the references therein.

Apart from improving the traditional communication modules, deep learning based end-to-end communication systems have been developed recently, where both the transmitter and the receiver are represented by DNNs. First proposed in [2], the framework has similar performance as the traditional approaches with block structures under the AWGN channels. The end-to-end framework has also been adopted within the OFDM system [21] and SVD precoding based MIMO system [22], where the channels are viewed as a group of independent sub-channels.

Recently, how to learn an end-to-end communication system without prior knowledge of channel models has been investigated. In [23], a reinforcement learning based approach has been developed to optimize the transmitter DNN without knowing the channel transfer function or channel state information (CSI). In [24], a model-free end-to-end communication framework has been designed with the stochastic perturbation approach. Recently, we have developed a conditional generative adversarial net (GAN) based approach in [5] for building end-to-end communications, where the channel effects are modeled by a conditional GAN. The end-to-end system can be trained with the conditional GAN as a surrogate channel to allow the gradients to back-propagate from the receiver DNN to the transmitter DNN.

B. Deep Learning on Inverse Problems

Our proposed method in this paper is also related to solving the inverse problems and recover the original data with learning approaches. In fact, deep learning has shown its ability in solving inverse problems, especially in the image processing area, such as denoise and deblur [25, 26], where the degeneration can be expressed as the original images convoluted with a kernel. The prevalent approaches include learning the end-to-end mapping with CNNs [27, 28, 29] and learning the posteriors with GAN [30, 31, 32, 33].

For most of the inverse problems encountered in the image processing area, the noised and distorted image can be restored by learning the 'prior knowledge' in the original images, such as the shapes and textures [34]. In contrast, the input data in the end-to-end framework can be independent bitstreams without any useful prior knowledge and the transmitter DNN should learn to form the constellation of the transmitted signal with redundancy so that the receiver can infer the channel information implicitly, which is then leveraged for recovering the transmitted data.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

C. Joint Source-and-Channel Coding

Recently, the auto-encoder based system has been applied to data compression and has shown superior results to the conventional approaches. Therefore, the end-to-end communication system can be trained for transmission of structured data, such as images and texts, where the auto-encoder learns to compress the data and encode the data simultaneously. The auto-encoder has been used for joint source and channel coding in a binary symmetric channel (BSC) channel [7], an AWGN channel [35], and a slow Rayleigh fading channel[36].

In this paper, the proposed end-to-end communication system includes source encoding and decoding seamlessly when the system is employed to transmit structured data, such as images and videos. We demonstrate that the end-to-end system is better than the system with the source coding and the wireless transmission designed separately.

III. Problem Formulation

In this section, we will provide necessary background information about the learning based end-to-end communication system, with which the key problem in learning end-to-end communication systems for general wireless channels is formulated.

A. Auto-Encoder based End-to-End Communication Framework

As shown in Fig. 1, a general baseband wireless communication system includes a transmitter, a receiver, and a wireless channel, where the transmitter and the receiver can be optimally designed and the wireless channel is random and unknown. In operation, the information source provides the transmitter data, $\mathbf{s}$, for transmission, which can be either binary bit-streams or any structured data, such as images and videos. The transmitted data is encoded and transformed into a complex base-band signal $\mathbf{x}$ and after further modulation to a prescribed frequency, sent to the antenna for radio transmission. At the receiver, the wireless channel effects are compensated and the transmitted data is reconstructed from the received signal $\mathbf{y}$.

As shown in Fig. 1(b), the learning based end-to-end communication paradigm follows the structure of the auto-encoder, where the transmitter DNN and the receiver DNN correspond to the auto-encoder and the auto-decoder, respectively. The source data, $\mathbf{s} \in \mathbb{R}^p$, is encoded by the auto-encoder network at the transmitter into an embedded vector, $\mathbf{x} \in \mathbb{R}^q$, $\mathbf{x} = \mathbf{f_E}(\mathbf{s}; \mathbf{\Theta_E})$, where $\mathbf{f_E}$ represents the auto-encoder function and $\mathbf{\Theta_E}$ denotes the trainable parameters. The embedded vector is then transmitted via the wireless channel and the receiver gets the channel output, $\mathbf{y}$. The auto-decoder network learns to recover the transmitted data, $\hat{\mathbf{s}}$, based on the received signal, *i.e.*, $\hat{\mathbf{s}} = \mathbf{f_D}(\mathbf{y}; \mathbf{\Theta_D})$, where $\mathbf{f_D}$ and $\mathbf{\Theta_D}$ represent the auto-decoder function and the trainable parameters, respectively. In this way, the chain of conventional signal processing modules at the transmitter, such as encoding and modulation, are represented by the auto-encoder and the modules at the receiver, such as channel estimation and decoding, are represented by the auto-decoder. The parameters of auto-encoder and decoder in the end-to-end paradigm are learned in a supervised manner with measured or simulated data. An end-to-end loss function, $\mathcal{L}(\mathbf{s}, \hat{\mathbf{s}})$, which measures the data recovery accuracy, is employed as the training objective for the end-to-end framework.

B. Wireless Channels as an Untrainable Stochastic Convolutional Layer

The reason that the end-to-end communication system toward general wireless channels is difficult to learn is that the end-to-end data recovery loss depends not only on the trainable parameters of DNNs, $\mathbf{\Theta_E}$ and $\mathbf{\Theta_D}$, but also on the channel realization $\mathbf{h}$. In a wide range of scenarios, the wireless channels are modeled as time-varying linear systems with additive noise and discrete time-varying channel models can be attained by sampling the continuous channel [37]. The channel output is expressed as a weighted summation of the current input symbols and previous input symbols if there exists inter-symbol interference. The weights are considered to be consistent for symbols within the block but may vary across blocks.

Hence, the wireless channel can be formulated as a convolutional operation and realized by a convolutional layer, which has been developed in [38] to efficiently extract features from images. Convolutional layers differ from fully connected layers in the shared-weight architecture and translational invariance characteristics. In a convolutional layer, every neuron only connects locally to several neurons in the previous layer and the connection weights are shared by all neurons within the layer. The weight, $\mathbf{w}$, of a 1D convolutional layer is a tensor of $n_i \times n_o \times k$, where the n_i and n_o represent the numbers of feature maps in the input and output layers, respectively, and k denotes the kernel size. The output at position l of feature map j can be expressed as

$$z_l^j = \sigma(\sum_{i=1}^{n_i} \sum_{m=1}^{k} \mathbf{w}[i, j, m] x_{l-m}^i), \tag{1}$$

where σ is the activation function and x_{l-m}^i is the input at position $l - m$ of feature map i.

It is straightforward to represent the wireless channel with a 1D convolutional layer. For a channel with an impulse response denoted by a real vector $[h_0, h_1, ..., h_{L-1}]$, the numbers of feature maps n_i and n_o are set as one. The kernel size k is determined by the length of the channel memory while the weights are determined by the channel response vector. When a complex channel response is considered, the real part and the imaginary part of the channel input and output are both represented by two feature maps. The weight $\mathbf{w}$ is a tensor of $L \times 2 \times 2$. In addition, by further increasing feature maps, the convolutional layer can also be used to represent the MIMO channels. Given the channel response vector, $\mathbf{h} \in \mathbb{C}^{N_t \times N_r \times N_L}$, the weight $\mathbf{w}$ can be represented as

$$\mathbf{w} = \left[\begin{bmatrix} \Re(\mathbf{h}_0) & \Im(\mathbf{h}_0) \\ -\Im(\mathbf{h}_0) & \Re(\mathbf{h}_0) \end{bmatrix}, \ldots, \begin{bmatrix} \Re(\mathbf{h}_{L-1}) & \Im(\mathbf{h}_{L-1}) \\ -\Im(\mathbf{h}_{L-1}) & \Re(\mathbf{h}_{L-1}) \end{bmatrix} \right].$$

Although a wireless channel can be represented with a commonly used convolutional layer [38], the weights of the convolutional layer are untrainable and take different values for different blocks, which is different from the commonly used convolutional layer.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

C. Key Issue

In a traditional communication system, a set of prescribed pilot symbols known to the receiver in advance are inserted for channel estimation. The receiver first estimates the channel parameters according to the received pilot data and then recovers the transmitted data with the estimated channel. Nevertheless, the design of the learning end-to-end communication system becomes much more difficult with the pilots for estimating the channel since two heterogeneous sources need to be taken into consideration at the receiver, *i.e.*, the received data and the estimated channel parameters. Also, the optimal pilot design depends on the particular scenario and is a complicated issue.

Instead of using pilots explicitly in our framework, the transmitter DNN learns to encode the transmitted data into embedding vectors that are robust to the channel effects, which can be regarded as adding pilots implicitly through learning. Since the wireless channel is represented by an untrainable stochastic convolutional layer with addictive noise, the key problem is formulated as follows.

Given channels $\mathbf{h} \sim p_c(\mathbf{h})$ *and data* $\mathbf{s} \sim p_d(\mathbf{s})$*, find the encoding and decoding functions such that the end-to-end loss* L *is minimize, that is*

$$\min_{\Theta_E, \Theta_D} L = \mathbb{E}_{\mathbf{h}\sim p_c(\mathbf{h}), \mathbf{s}\sim p_d(\mathbf{s})} \mathcal{L}(\mathbf{f_D}(\mathbf{f_h}(\mathbf{f_E}(\mathbf{s}; \mathbf{\Theta_E})); \mathbf{\Theta_D}), \mathbf{s}), \tag{2}$$

where $\mathcal{L}(\cdot, \cdot)$ *is used to measure the distance between the recovered and the input data and* $\mathbf{f_h}$ *represents the wireless channel function, including a stochastic convolution layer and the additive noise layer.*

Loss function $\mathcal{L}$ can be customized under different scenarios. When source data vector $\mathbf{s}$ represents the bitstreams, the loss function can be binary cross-entropy. When source data vector $\mathbf{s}$ represents imagery data, the loss function can be the peak signal-to-noise ratio (PSNR) or the multi-scale structural similarity index measure (MS-SSIM), *etc.*

IV. Pilot-free End-to-End Communication Systems

Instead of using the pilot data explicitly, we propose a pilot-free end-to-end learning approach to address channel estimation issue. In this section, we are going to present the learning based end-to-end communication paradigm for general wireless channels in detail, where the overview of the training algorithm, the bilinear operations employed at the receiver DNN, and implementation details will be shown. A toy example is also provided to illustrate how the pilot-free end-to-end system works.

A. Overview

The proposed approach follows the structure of learning based end-to-end communication system from the previous works [2, 5, 6, 7], where DNNs are employed in both the transmitter and the receiver. 1D CNN is chosen in our framework for both the transmitter and the receiver DNNs for two reasons. First, for a wideband wireless channel, the channel output is the result of the linear convolution of the input with the impulse response of the channel [5]. Therefore, it is a natural choice to use convolutional layers in order to deal with the convolutional channel effects. Second, previous works have shown that using CNN is effective in realizing the coding gain with strong generalization ability to handle unseen codewords [6].

In order to train the end-to-end networks with the stochastic channel layer representing the channel, a simple but effective training algorithm is used with mini-batches of input data samples $\mathbf{s}$ in combination with the channel realization $\mathbf{h}$. Thus, the training set consists of two types of data: the input sample dataset, $\mathcal{S}$, and the channel sample dataset, $\mathcal{H}$. The training algorithm is illustrated in Algorithm 1. In the vanilla stochastic gradient descent (SGD) algorithm, the gradient is noisy for each updating since only a mini-batch of samples are considered. Therefore, the expected loss on the channel set can be estimated with a mini-batch of channels as well for each updating. For each mini-batch of the input samples, a mini-batch of channel samples are sampled from the channel set. The gradients for updating the weights of the neural networks are obtained by accumulating the gradients, $\mathbf{g}$, under each channel from the channel mini-batch,

$$\mathbf{g} = \sum_{\mathbf{h}\in\mathcal{H}_i} \bigtriangledown_{\mathbf{\Theta}} \mathcal{L}(\mathbf{f_D}(\mathbf{y_h}; \mathbf{\Theta_D}), \mathbf{s}), \tag{3}$$

where $\mathcal{H}_i$ is a minibatch of channel samples and $\mathbf{y_h}$ is the channel output under channel realization $\mathbf{h}$. With the attained $\mathbf{g}$, the parameters ($\mathbf{\Theta_E}$ and $\mathbf{\Theta_D}$) of the network are updated.

Algorithm 1 Training Pilot-free End-to-end Communication System

Require: Learning rate η
Require: Initial parameters $\mathbf{\Theta} = [\mathbf{\Theta_E}, \mathbf{\Theta_D}]$
$\mathbf{g} \leftarrow 0$
for $i = 0, 1, ...$ **do** ▷ Loops of samples
  Sample a minibatch $\mathbf{s}$ of samples from $\mathcal{S}$
  for $j = 0, 1, ...$ **do** ▷ Loops of channel realization
    Sample a channel $\mathbf{h}$ from $\mathcal{H}$
    $\mathbf{x} \leftarrow \mathbf{f_E}(\mathbf{s}, \mathbf{\Theta_E})$
    $\mathbf{y_h} = \mathbf{f_h}(\mathbf{x}, \mathbf{h})$
    $\mathbf{g} \leftarrow \mathbf{g} + \bigtriangledown_{\mathbf{\Theta}} \mathcal{L}(\mathbf{f_D}(\mathbf{y_h}; \mathbf{\Theta_D}), \mathbf{s})$
  $\mathbf{\Theta} \leftarrow \mathbf{\Theta} - \eta\mathbf{g}$

Remark 1 *Compared with end-to-end systems, where the channel effects are modeled as a black-box [5, 23], the proposed approach relies less on the accessibility of the real channel. Online training is necessary for the black-box end-to-end systems since a large number of input and output pairs of the channel should be recollected after each update of the transmitter DNN. In our approach, the wireless channel models are used and therefore the training can be offline once a training channel set* $\mathcal{H}$ *is collected or simulated.*

B. Why Pilot is Unnecessary: A Toy Example

Here we provide a toy example to show why DNN can effectively recover the transmitted data without requiring pilots for channel estimation.

Example: A CNN based pilot-free end-to-end system is build for Rayleigh fading channels with binary input of size

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

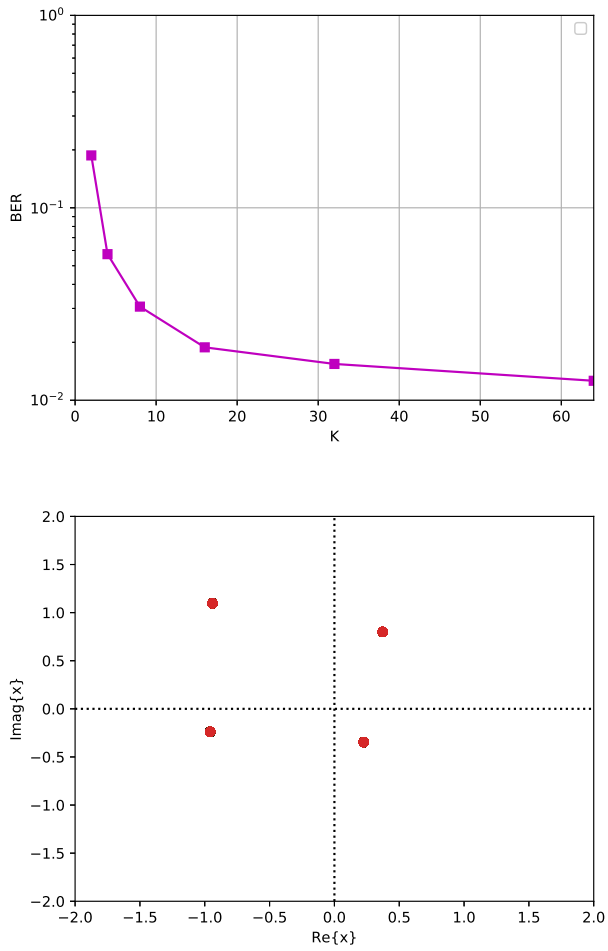

Fig. 2: A toy example on Rayleigh fading channel. (a) BER under different block length. (b) Constellation under Rayleigh fading channel.

$[K, 2]$, where K is the block size. In the transmitter, we use two convolutional layers with kernel sizes of one. Therefore, every two bits are encoded independently with others. Two fully connected layers are used at the receiver. Therefore, all received signals are involved in recovering each bit. The DNN structure used in this example is shown in Appendix A.

As shown in Fig. 2(a), with an increase of the block size K, the bit-error rate (BER) of the end-to-end system improves. Since every two bits are encoded independently, the constellation generated by the transmitter DNN can be easily drawn as in Fig. 2(b), which, unlike the conventional quadrature phase-shift keying (QPSK) modulation, is asymmetrical to the origin. With this asymmetrical constellation, the phase and gain of the channel can be inferred with multiple received data and the reliability of the inference increases with the block size, which leads to the performance shown in Fig. 2(a). This example shows the end-to-end system can generate the constellation of the transmitted signal to infer the channel information implicitly, which explains why pilots are unnecessary in our system.

C. Bilinear Receivers

Although our proposed end-to-end approach is pilot-free, it is still necessary to infer the current wireless channel because the end-to-end recovering loss relies on the current channel to a large degree. Without the pilots, the channel information can only be obtained from the received data $\mathbf{y}$. However, the output of each neuron in convolutional layers only depends on the input data within its receptive field. As a result, the received data, $\mathbf{y}$, cannot be fully leveraged for channel information extraction via the convolutional layers since the channel information contained outside the receptive field cannot be used for reconstruction of the transmitted data. To address this problem, two DNN modules are employed at the receiver for channel information extraction and data recovery, respectively. As shown in Fig. 3, the receiver DNN in the proposed end-to-end framework includes the channel information extraction module and data recovery module. To fully leverage the received data for channel information extraction, fully connected layers are exploited to obtain the global channel features, which are then combined with the received data via a bilinear production. The convolutional layers are employed in the data recovering module to further analyze the bilinear features and recover the transmitted data.

The bilinear operation of CNN is shown in Fig. 4. This structure is widely used in computer vision for combining features together [39]. With the bilinear operation, the received signal, $\mathbf{y} \in \mathbb{R}^{[K \times N_r]}$, is augmented at each position by the extracted channel feature, $\mathbf{z} \in \mathbb{R}^{l_z}$, from the channel information extraction module. In particular, the received signal is first reshaped into a vector and channel feature $\mathbf{z}$ is then multiplied to each item of the vector. As a result, an augment signal $\mathbf{Y} \in \mathbb{R}^{K \cdot N_r \times l_z}$ is obtained via $\mathbf{Y} = vec(\mathbf{y}) \otimes \mathbf{z}$, where $\otimes$ represents outer product operation. The augment signal $\mathbf{Y}$ is further reshaped into a matrix of size $[K, l_z \times N_r]$. Therefore, the bilinear operation increase the number of feature maps of the received signal by l_z times, as shown in Fig. 4. The gradients of loss L with respect to the bilinear operation can be expressed as

$$\frac{\partial L}{\mathbf{y}} = \mathbf{z}\frac{\partial L}{\mathbf{Y}}, \tag{4}$$

$$\frac{\partial L}{\mathbf{z}} = vec(\mathbf{y})\frac{\partial L}{\mathbf{Y}}. \tag{5}$$

D. Implementation Details

The details of the CNN structure used in each model of the proposed method are shown in Appendix B. The binary cross-entropy loss is utilized when the input are the bitstreams,

$$\mathcal{L}(\mathbf{s}, \hat{\mathbf{s}}) = \sum_n (s_n \log \hat{s}_n + (1 - s_n) \log(1 - \hat{s}_n)), \tag{6}$$

where s_n and $\hat{s}_n$ are the nth elements of $\mathbf{s}$ and $\hat{\mathbf{s}}$, respectively. Since $s_n \in \{0, 1\}$ and $\hat{s}_s$ is the output of the Sigmoid function, representing a probability value between 0 and 1, the cross-entropy loss increases as the predicted probability diverges from the actual label.

The MS-SSIM [40] is chosen as the loss function when the images are considered as the transmitted data. The SSIM

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

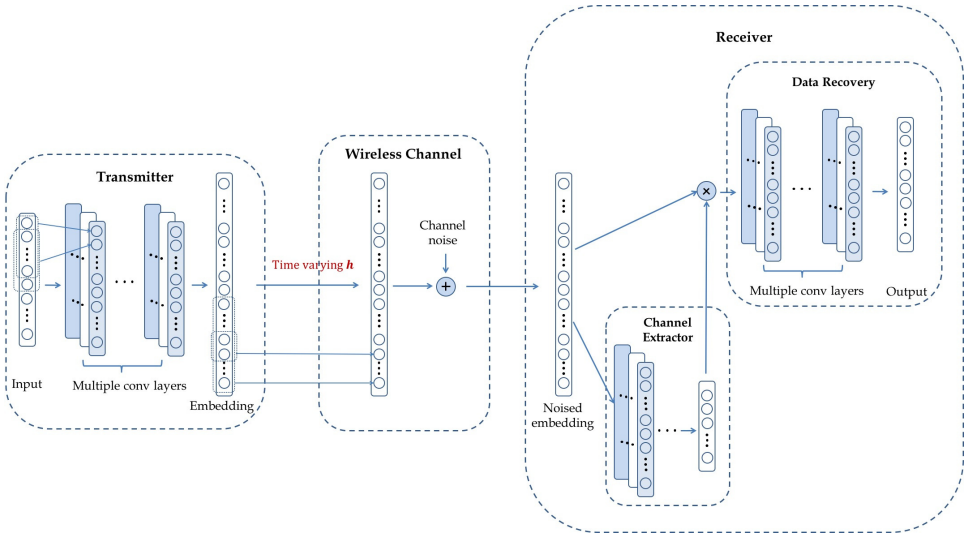

Fig. 3: Structures of proposed end-to-end communication system

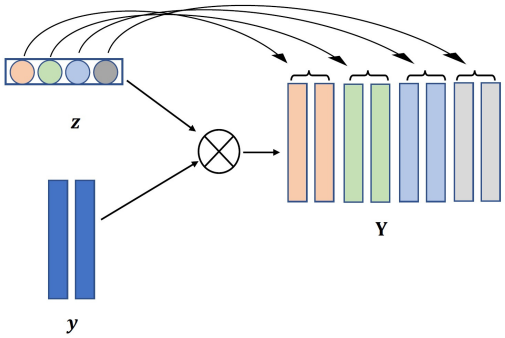

Fig. 4: Bilinear combination of channel extracted features and received signal.

metric compares various patches in two images on luminance, contrast, and structure and can be expressed as,

$$SSIM(\mathbf{s}, \hat{\mathbf{s}}) = \frac{(2\mu_{\mathbf{s}}\mu_{\hat{\mathbf{s}}} + c_1)(2\sigma_{\mathbf{s}\hat{\mathbf{s}}} + c_2)}{(\mu_{\mathbf{s}}^2 + \mu_{\hat{\mathbf{s}}}^2 + c_1)(\sigma_{\mathbf{s}}^2 + \sigma_{\hat{\mathbf{s}}}^2 + c_2)}, \quad (7)$$

where $\mu_{\mathbf{s}}$, $\mu_{\hat{\mathbf{s}}}$, $\sigma_{\mathbf{s}}$, and $\sigma_{\hat{\mathbf{s}}}$ denote mean pixel intensity and the standard deviations of pixel intensity in patches of $\mathbf{s}$ and $\hat{\mathbf{s}}$, respectively, the variable $\sigma_{\mathbf{s}\hat{\mathbf{s}}}$ denotes the sample correlation coefficient between corresponding pixels of patches in $\mathbf{s}$ and $\hat{\mathbf{s}}$. MS-SSIM extends this single-scale measure by conducting SSIM over multiple scales through a process of multiple stages of sub-sampling of size 2. The proposed end-to-end framework is trained under a fixed signal-to-noise ratio (SNR) and tested with different SNRs.

V. Experiments

In this section, the proposed pilot-free end-to-end communication system is tested under several different scenarios. We first show that the pilot-free end-to-end communication system can work effectively under the frequency-selective fading channels and flat-fading MIMO channels. We then show that it can also leverage the correlation in the source data to further improve the transmission performance by including the source coding in the end-to-end paradigm.

A. Frequency-Selective Fading Channel

Experimental Settings: When the delay spread of a channel is comparable with or larger than the symbol duration, inter-symbol interference exists and the corresponding channel is called the frequency-selective channel. The output of the frequency-selective channel is formulated as $\mathbf{y} = \mathbf{h} \circledast \mathbf{x} + \mathbf{n}$, where $\mathbf{h}$ and $\mathbf{x}$ are the channel impulse response and the channel input, respectively, $\circledast$ denotes the linear convolution operation, and $\mathbf{n}$ denotes the channel noise. Hence, the frequency-selective channel is modeled as multi-tap filtering. There have been some statistical models to characterize the distribution of $\mathbf{h}$. The complex Gaussian distribution assumption is prevalent for rich scattering environment. The strengths and delays of multiple taps are characterized by the power delay profile (PDP), $\mathbf{p} = \{p_i\}$. An eight-tap channel with equal average power is used in our experiments, that is $p_i = 1$, for $i = 1, \cdots, 8$.

Baselines: The proposed end-to-end system is compared with two benchmarks. The first one is using OFDM system to deal with the frequency-selectivity. QPSK is employed for modulation and the block length is set as 128 samples and the length of cycle prefix is 16 samples. In the OFDM system, additional overhead is necessary since the pilots are inserted for estimating the channel, which has the length of 16 in our experiment, *i.e.*, 12.5% overhead for channel estimation. For channel coding, we use the convolutional codes with rate 0.5 [41]. The proposed pilot-free end-to-end communication system is also compared with the conditional GAN based end-to-end approach in [6], where pilots are used for channel modeling and data recovering. Similar to the coherent OFDM system, additional pilots are require in the conditional GAN system.

End-to-End Model Training: The model structure is presented in detail in Appendix A. The training and test dataset are with 3×10^8 and 1×10^8 samples, respectively. The binary

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

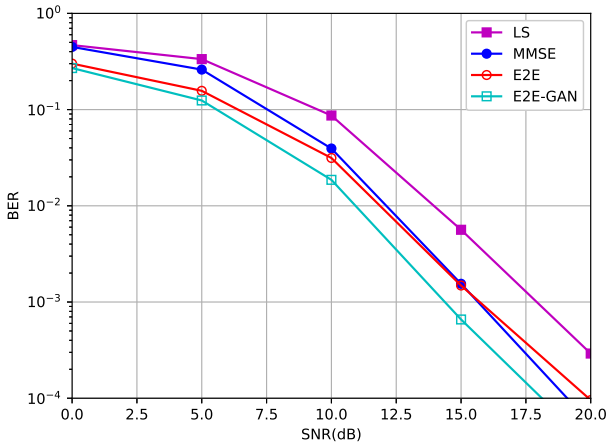

Fig. 5: BER curves on frequency-selective channel.

cross entropy loss and Adam optimizer are used. The training batch size is $1,000$ and the learning rate is 10^{-4}. The model is trained with SNR = 15 dB, and tested with SNR ranging from 0 dB to 20 dB.

Experimental Results: Fig. 5 demonstrates that the end-to-end communication system (labeled as 'E2E') can achieve better BER performance than the least square (LS) channel estimation approach and comparable performance with the minimum mean-square error (MMSE) approach. The conditional GAN based end-to-end approach performs better than the pilot-free approach because of the additional information provided by the pilots.

B. MIMO Channel

The MIMO is a critical technique in future communication systems. We use multiple filters to address multiple antennas in the pilot-free communication system.

Experimental Settings: A system with four transmitter antennas and four receiver antennas over a narrow-band channel is considered in this experiment. The channel matrix $\mathbf{H}$ is modeled as independent complex Gaussian distribution, where the channel output can be expressed as $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$, where $\mathbf{x}$ is the transmit signal vector, $\mathbf{n}$ is the noise vector, and $\mathbf{y}$ is the received signal vector. For correlated channels, the channel matrix can be expressed as $\mathbf{H_c} = \sqrt{\mathbf{R_t}}\mathbf{H}\sqrt{\mathbf{R_r}}$, where $\mathbf{H}$ is the independent channel used above and $\mathbf{R_t}$ and $\mathbf{R_r}$ represent the transmitter and the receiver channel correlation matrices, respectively. The correlation matrices can be expressed as

$$\mathbf{R_t}=\begin{bmatrix} 1 & \rho_t & \rho_t^4 & \dots \\ \rho_t & 1 & \rho_t & \dots \\ \vdots & & \ddots & \\ \rho_t^{(N_t-1)^2} & & \dots & 1 \end{bmatrix}, \mathbf{R_r}=\begin{bmatrix} 1 & \rho_r & \rho_r^4 & \dots \\ \rho_r & 1 & \rho_r & \dots \\ \vdots & & \ddots & \\ \rho_r^{(N_r-1)^2} & & \dots & 1 \end{bmatrix}, \tag{8}$$

where ρ_t and ρ_r denote the correlation coefficients in the transmitter and receiver while N_t and N_r denote the numbers of antennas in the transmitter and the receiver, respectively. $\rho_t = \rho_r = 0$ for independent channel.

Baselines: The pilot-free end-to-end system is compared with zero-forcing (ZF) and MMSE signal detection approaches. The block size is set as 256 in this experiment. Both approaches require additional overhead for channel estimation (12.5% each block in our experiment) and MMSE is used for channel estimation. QPSK and convolutional code with rate 0.5 are used as modulation and channel coding, respectively.

End-to-end Model Training: The detailed model of the MIMO end-to-end system is illustrated in Appendix A. The binary cross entropy loss in (6) is used and the training and testing sets are 3×10^8 and 1×10^8, respectively. The binary cross entropy loss and Adam optimizer are used. The training batch size is $1,000$ and the learning rate is 10^{-4}, as in Section V-A.

Experimental Results: The performance of the proposed method and the baseline are shown in Fig. 6(a) for independent channels. Maximum-likelihood (ML) detection [42] achieves the best performance. The end-to-end system outperforms the ZF detection approach by a large margin and is also better the MMSE detection approach when SNR is larger than 12.5 dB. If perfect CSI is available for MMSE, the performance can be improved about 5dB. In addition, both baseline approaches require additional overhead for channel estimation, which can be saved by the proposed approach.

The performance of our end-to-end method with different correlation coefficients is shown in Fig. 6(b), where $\rho = \rho_t = \rho_r$. The end-to-end approach and the baselines are evaluated with SNRs equal to 5dB and 10dB. With the end-to-end approach, the correlation of the channel can be leveraged automatically to improve the performance significantly while the correlation of the channel degenerates the performance in terms of BER in the baseline systems. As shown in Fig. 6(a) and (b), the MMSE approach with perfect CSI (labeled as 'MMSE-CSI') outperforms the end-to-end system when $\rho_t = \rho_r = 0$. But when $\rho_t = \rho_r = 0.5$, the pilot-free end-to-end system can achieve similar results with the MMSE approach with perfect CSI. In addition, when $\rho_t = \rho_r = 0.9$, the baseline approaches can hardly handle the channel correlation even with the perfect CSI while the pilot-free end-to-end approach remains unaffected.

C. End-to-End based Wireless Image Transmission

Besides the transmission of the binary data stream, the data-driven end-to-end communication system can also be exploited for structure data transmission. We take the wireless image transmission as an example. Conventionally the images need to be compressed first into binary data streams, which are further transmitted using a wireless communication system. While for end-to-end communication systems, image compression and recovery can be performed via the end-to-end transmitter and receiver DNNs, respectively.

Experimental Settings: The experiments are conducted on three image datasets, *i.e.*, the ImagenetVal [43], Kodak [1], and B100 [44]. The images are compressed and encoded

[1] http://r0k.us/graphics/kodak

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

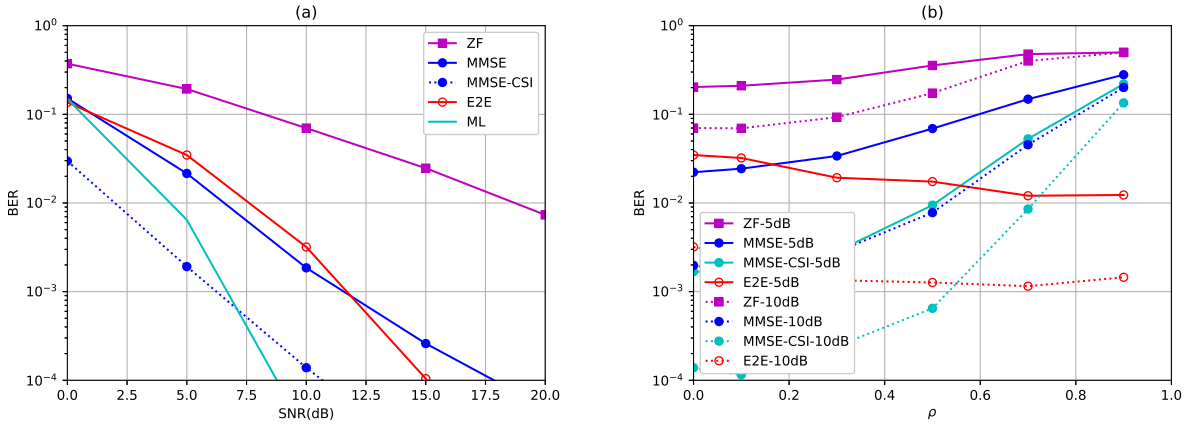

Fig. 6: (a) BER curves on MIMO channel. (b) BER with channel correlation ρ.

with either the baseline system or the end-to-end image transmission system. The encoded signal is then transmitted via the wireless channels to the receiver, where the original images are recovered. Although the images are with different sizes, the encoded data is grouped into blocks with a fixed length (128 used in our experiments) for wireless transmission. The multi-path channel, as in Section V-A, is considered in the experiments.

Baselines: The end-to-end communication approach is compared with the baseline systems, where the image compression system and the wireless communication system are designed separately. Several image compression benchmarks are selected for comparison, including a CNN based compression approach [45] and the two conventional image compression approaches, *i.e.*, BPG and JPEG2000. The details of the image compression baselines are described in Appendix B. In order to measure the improvement brought by the joint design of source coding and wireless transmission, the images are first compressed into binary compressed data with the baseline image compression approaches, the binary data is then grouped into blocks (128 used in our experiments) and transmitted with the bitstream end-to-end communication system used in V-A. The original images are recovered with the output of the bitstream end-to-end communication system at the receiver.

The conventional image compression and recovery methods are very sensitive to the errors introduced during the wireless transmission. For instance, the JPEG files often break down completely with only a few errors. The performance drops dramatically for the CNN based approach as well since the arithmetic coding, which is sensitive to the errors, is used to further compress the extracted feature. In order to make a fair comparison, we assume that a re-transmit communication protocol is used for the baseline systems, where the block will be re-transmitted if there exists any error in the current block during the transmission. Therefore, the source decoding is conducted without error and the expectation of the total blocks transmitted will increase by $1/(1-p_e)$ times, where p_e is the block-error rate. For simplicity, we assume that the errors in each block can be detected without any additional cost, which makes the comparison a little unfair against the proposed end-to-end approach.

End-to-End Model Training: Compared to the bitstream end-to-end communication systems used in Section V-A and V-B, the image end-to-end system incorporated 2D convolutional layer for image compression and recovery. The detailed model structure is illustrated in Appendix A. The model is trained with the training set of ImageNet dataset. The input images are cropped from the training set, with size 256×256 without any augmentation. The MS-SSIM is used as the training loss, where the implementation and weights follow [40]. The batch-size is 32 and the learning rate is 10^{-3}.

Experimental Results: Fig. 7 shows the MS-SSIM performance under different SNRs with only $\frac{J}{256}$ blocks transmitted, where J is the number of pixels. Since the length of the block is 128, the bit per-pixel (bpp) is 0.5 for the baseline systems if there is no transmission error. The source compression rates for the baseline systems increase with the increasing p_e. From the figure, the proposed approach (labeled as 'Joint E2E') outperforms the baselines by a large margin with the joint training of the source coding and communications. Among the three baselines, the CNN based compression approach (labeled as 'Separate E2E') achieves better performance than the other two conventional compression approaches. The baseline systems have an advantage by using the global compression approaches with context information. While in our system, the global compression is discarded and the encoded vectors for transmission are obtained locally and independently. However, this disadvantage is offset by the power of joint training. Therefore, even for the high SNR area, where the error of the communication is negligible, the joint training approach can still outperform the baselines. As the noise increases, the performance of the other systems drops quickly due to the increased error rate while the joint training approach is more robust to the channel noise.

D. Robustness

When deployed in the real environment, the channel distributions may differ from the training channel distribution, which may degrade the performance. If the distributional difference is large, the model should be fine-tuned or retrained with the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

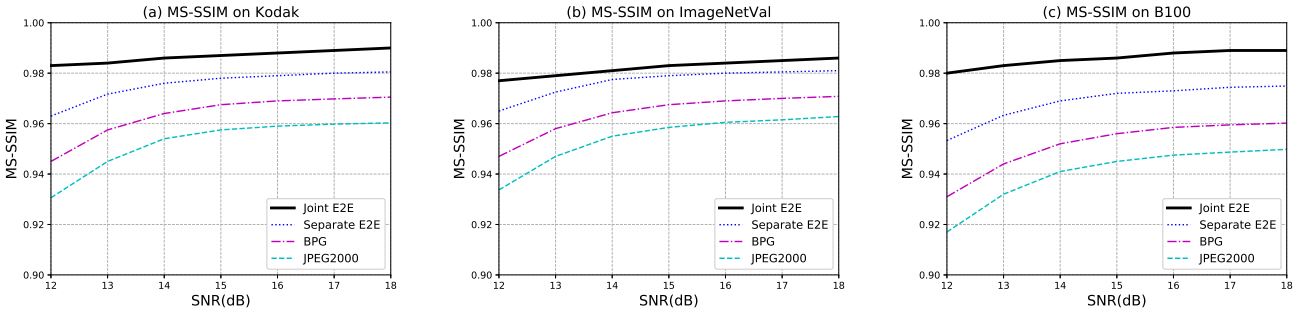

Fig. 7: Performance of our approach on Kodak, ImageNetVal, and B100.

new data. But we still wish the system to be robust for the small variance in the channels. The robustness of the proposed end-to-end system is tested in both types of channels, where there exists a discrepancy in the channel distributions used in the training and test stages.

The robustness to the PDP is tested in the frequency-selective channels. The equal average power is used during the training, *i.e.* $p_i = 1$, while the test channels are generated with new PDPs obtained by randomly adding or subtracting Δ in p_i used in the training. The BER performance of the end-to-end model (labeled as 'mismatch E2E') against Δ is shown in Fig. 8(a) with the SNR fixed at 15 dB. Although the end-to-end approach is trained with a different PDP, it shows robustness to the discrepancy in PDP. The performance stays nearly constant for $\Delta < 0.5$.

The robustness to the channel correlation coefficient ρ is tested in the MIMO channels. The model is trained with $\rho = 0.5$ and tested with ρ ranging from 0 to 0.9. As shown in Fig. 8(b), the end-to-end method also shows robustness to the discrepancy in ρ. The model trained with $\rho = 0.5$ achieves similar performance with models trained and tested in the corresponding correlation coefficient.

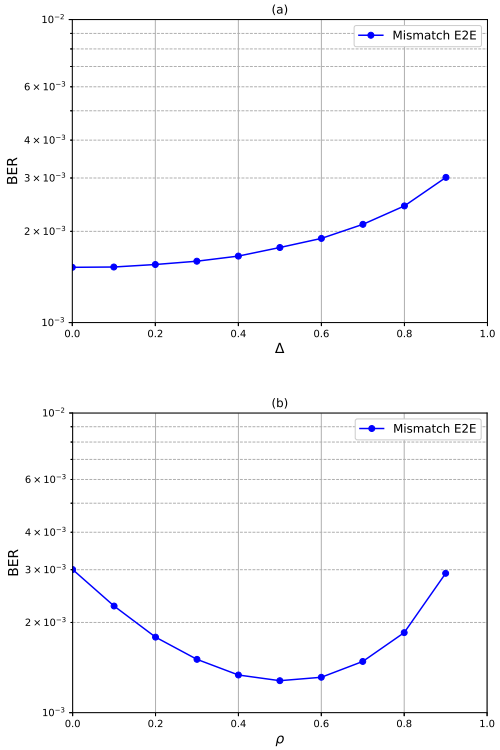

Fig. 8: Robustness on frequency-selective channel and MIMO channel. (a) Robustness to PDP, where a distortion Δ is considered. (b) Robustness to ρ for the model trained with $\rho = 0.5$.

E. Complexity Analysis

We compare the computation complexity of the proposed system and the baselines. The direct comparison between the end-to-end approach and the traditional communication system is difficult since the complexity is affected by many parameters of the particular architecture used. In Table I, the complexity of the proposed approach and each component in the baseline system is listed. The computational complexity of the CNN is $\mathcal{O}(\sum_{l=1:L} N k_l^2 F_{l-1} F_l)$, where the N is the block size, L is the number of the layers, k_l is the kernel size of the lth layer, and F_{l-1} and F_l are the number of filters in the $l-1$th layer and the lth layer, respectively.

For the OFDM system, the computational complexity of the fast Fourier transform (FFT) is $\mathcal{O}(N \log N)$ and the channel estimation complexity is $\mathcal{O}(N_p^3)$, where the N_p is the number of the pilot used. The complexity of Viterbi decoding is $\mathcal{O}(N 2^k)$, where k is the length of the memory. In general, the computation complexity of the end-to-end system grows linearly with the block length N but when the number of layer L and the number of feature map F_l are large, the computation complexity becomes large. When the multiple antennas are employed in the transmitter and the receiver, the feature maps of the output of the transmitter and the input of the receiver will be increased correspondingly. The computational complexity will grow linearly with the number of the antenna if the number of feature maps in the other layers remains constant. It will grow quadratically with the number. In the baseline system, the detection complexity is $\mathcal{O}(N_r^3)$ for both ZF and MMSE detection, where N_r is the number of receiver antenna.

To complete our discussion of computational complexity, we have measured the average run time of the proposed algorithm and the baseline approach on a Windows server with an Intel i7 CPU and an Nvidia 1080Ti GPU. We choose MMSE and OFDM as baseline used in frequency-selective channels and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

TABLE I: Model Complexity

Model	Number of Flops Operation
FFT	$\mathcal{O}(N\log(N))$
MIMO ZF	$\mathcal{O}(N_r^3 N)$
MIMO MMSE	$\mathcal{O}(N_r^3 N)$
MMSE Channel Estimation	$\mathcal{O}(N_p^3)$
Convolutional Code	$\mathcal{O}(N2^k)$
End-to-End	$\mathcal{O}(\sum_{l=1:L} Nk_l^2 F_{l-1} F_l)$

MMSE detection used in MIMO channels. The average running time is about 3.5×10^{-2} and 2.2×10^{-2} seconds for each block under MIMO and frequency-selective channel system, respectively, while the average running time for the end-to-end system is about 2.5×10^{-3} seconds.

VI. Conclusion

In this paper, we have proposed a pilot-free deep auto-encoder based end-to-end communication system for wireless channels. We show that the end-to-end communication system can be modeled as a deep auto-encoder system with a stochastic convolutional layer since linear channels can be model as a convolutional operation. We propose a training procedure with mini-batches of input samples and of the channels. In addition, an additional channel extraction model is employed in the receiver to make sure the channel information is extracted globally. For the frequency-selective channels and the flat-fading MIMO channels, the end-to-end models are effectively trained without pilot information and show better performance compared with the traditional baselines. Our proposed approach can successfully address the impact of unknown channels, which provides a new solution to build the end-to-end system for the wireless communications. In the future, there are many directions to be explored, including how to develop an end-to-end system that can be adaptive to different channels and SNRs.

Appendix A
Deep structures

Toy Example Model Structure: There are two 1D convolutional layers in the transmitter DNN with kernel size 1. The filter numbers are 256 and 2. Relu activation function is used in the first layer and a power normalization layer is used in the second layer for normalizing the power of the output. The receiver DNN consists of two fully connected layers. The number of neurons are 256 and $2 \times K$, respectively, where K is the block size. Relu activation function is used in the first layer and Sigmoid function is used in the second layer to map the output to the interval $[0, 1]$.

Frequency-selective and MIMO End-to-end Communication Model Structures: The structures of the DNNs used in the transmitter and the receiver are shown in Table II. There are four 1D convolutional layers in the transmitter and six convolutional layers in the receiver. The output of the transmitter is normalized to fit the power constraints. The differences between the models used in the frequency-selective channels and in the MIMO channels are the numbers of filters in the last convolutional layer of the transmitter DNN and receiver DNN, which change with the number of transmitter antennas N_t.

TABLE II: Model Parameters of Pilot-free End-to-End System

Type of layer	Parameters	Output size (Frequency-selective)	Output size (MIMO)
Transmitter			
Input	Input layer	128×1	64×4
Conv+Relu	kernel = 5, filter = 256	128×256	64×256
Conv+Relu	kernel = 3, filter = 128	128×128	64×128
Conv+Relu	kernel = 3, filter = 64	128×64	64×64
Conv	kernel = 3, filter = $2N_t$	128×2	64×8
Normalization	Power normalization	128×2	64×8
Wireless Channel			
Conv	kernel = channel memory, filter = $2N_r$	135×2	64×8
Gaussian Noise	NA	135×2	64×8
Receiver			
Conv+Relu	kernel = 5, filter = 256	135×256	64×256
Conv+Relu	kernel = 5, filter = 128	135×128	64×128
Conv+Relu	kernel = 5, filter = 128	135×128	64×128
Conv+Relu	kernel = 5, filter = 128	135×128	64×128
Conv+Relu	kernel = 5, filter = 64	135×64	64×64
Conv+Relu	kernel = 5, filter = 64	135×64	64×64
Conv+Relu	kernel = 5, filter = 64	135×64	64×64
Conv+Sigmoid	kernel = 3, filter = N_t	128×1	64×4
Channel Feature Extractor			
Conv+Relu	kernel = 5, filter = 256	135×256	64×256
Conv+Relu	kernel = 3, filter = 128	135×128	64×128
Conv+Relu	kernel = 3, filter = 64	135×64	64×64
Conv	kernel = 3, filter = 2	135×2	64×2
FC +Relu	100	100	100
FC	40	40	40

End-to-End Wireless Image Transmission Model Structure: The structure of the image transmission end-to-end system requires additional 2D convolutional and deconvolutional layers to compress and recover the image, respectively. The structure follows the architecture in [45] and is shown in Table III. The output of the 2D convolutional layers is of size $\frac{R}{8} \times \frac{C}{8} \times 32$, where R and C represent the height and width of the input image, respectively. Then the features of every four positions are grouped as a block and transmitted with a bitstream end-to-end system. Therefore, there are $\frac{J}{256}$ blocks in all. At the receiver, the output of bitstream end-to-end system of all blocks will form the feature map of size $\frac{R}{8} \times \frac{C}{8} \times 32$, which is considered as the input of the 2D deconvolutional layers to recover the original image.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

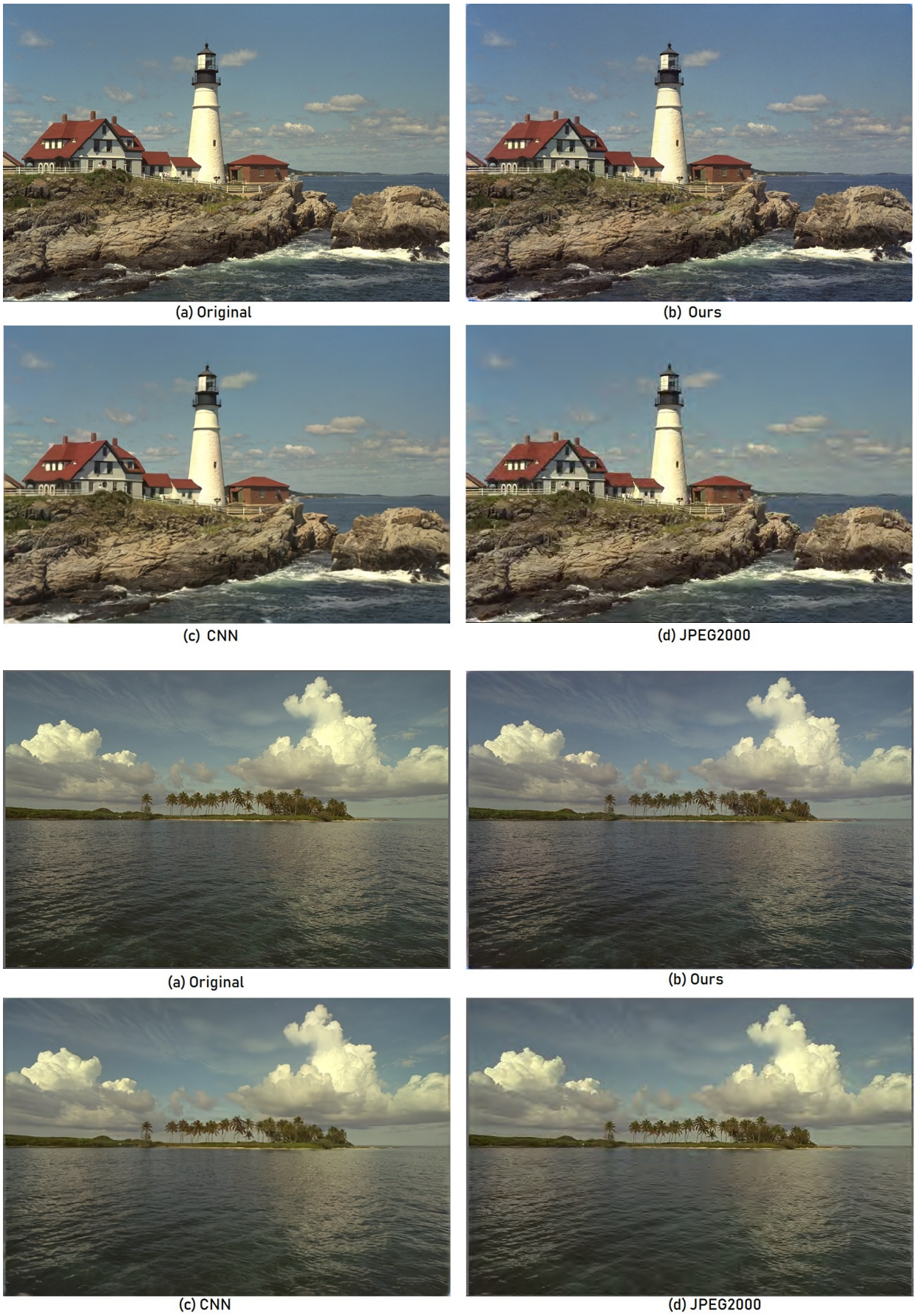

Fig. 9: Examples of recovered data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCCN.2021.3061464, IEEE Transactions on Cognitive Communications and Networking

TABLE III: 2D Convolutional and Deconvolutional Layers used in Wireless Image End-to-End System

Type of layer	Parameters	Output size
Transmitter		
Input	Input layer	$R \times C \times 3$
Conv+Relu	kernel = 5×5, stride = 2, filter = 64	$\frac{R}{2} \times \frac{C}{2} \times 64$
Conv+Relu	kernel =5×5, stride = 2, filter = 128	$\frac{R}{4} \times \frac{C}{4} \times 128$
$15\times$ Res Block	kernel =3×3, stride = 1, filter = 128	$\frac{R}{4} \times \frac{C}{4} \times 128$
	kernel =3×3, stride = 1, filter = 128	$\frac{R}{4} \times \frac{C}{4} \times 128$
	skip connection	$\frac{R}{4} \times \frac{C}{4} \times 128$
Conv	kernel =5×5, stride = 2, filter = 32	$\frac{R}{8} \times \frac{C}{8} \times 32$
Receiver		
Deconv+Relu	kernel = 3×3, stride = 2, filter = 128	$\frac{R}{4} \times \frac{C}{4} \times 128$
$15\times$ Res Block	kernel =3×3, stride = 1, filter = 128	$\frac{R}{4} \times \frac{C}{4} \times 128$
	kernel =3×3, stride = 1, filter = 128	$\frac{R}{4} \times \frac{C}{4} \times 128$
	skip connection	$\frac{R}{4} \times \frac{C}{4} \times 128$
Deconv+Relu	kernel = 5×5, stride = 2, filter = 64	$\frac{R}{2} \times \frac{C}{2} \times 64$
Deconv	kernel =5×5, stride = 2, filter = 3	$R \times C \times 3$

APPENDIX B
IMAGE COMPRESSION BASELINES

Conventional Image Compression Baselines: We use the Kakadu implementation[2] for JPEG2000. BPG[3] is based on HEVC, the state-of-the-art in video compression, and outperforms JPEG2000. We use BPG in the nondefault 4:4:4 chroma format.

CNN based Image Compression: The deep compression baseline follows [45]. The architecture consists of an encoder, a quantizer, and a decoder, where the encoder and the quantizer are used at the transmitter while the decoder is used at the receiver. The encoder E maps an input image into an embedding vector $\mathbf{z} = E(\mathbf{s})$. The quantizer Q discrete the coordinates of $\mathbf{z}$, obtaining $\hat{\mathbf{z}}$ with $\hat{\mathbf{z}} = Q(\mathbf{z})$, which is then encoded into a bitstream. With the decoder D, the reconstructed image $\hat{\mathbf{s}} = D(\hat{\mathbf{z}})$ can be obtained from the quantized vector $\hat{\mathbf{z}}$.

The training objective is to make the representation $\mathbf{z}$ compact while the reconstruction error $d(\cdot, \cdot)$, measured by MSE or MS-SSIM, small. The E and D are modeled with 2D CNNs and the loss function is expressed as,

$$L = d(\mathbf{s}, \hat{\mathbf{s}}) + \beta H(\mathbf{z}), \tag{9}$$

where $H(\mathbf{z})$ denotes the cost of encoding $\mathbf{z}$ to bits, *i.e.,* the entropy of $\mathbf{z}$ and β is the trade-off weight of the two components.

APPENDIX C
EXAMPLES OF RECOVERED IMAGES

In this section, we provide several examples of recovered images of different approaches, shown in Fig. 9, which are obtained when SNR is set as 12dB.

[2]http://kakadusoftware.com/

[3]https://bellard.org/bpg/

REFERENCES

[1] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Oct. 2019.

[2] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[3] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5438–5453, Oct. 2018.

[4] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep learning based communication over the air," *IEEE J. Select. Topics Signal Process.*, vol. 12, no. 1, pp. 132–143, Feb. 2018.

[5] H. Ye, G. Li, B.-H. Juang, and K. Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional GAN," in *Proc. IEEE Globecom Workshops*, Dec. 2018, pp. 1–5.

[6] H. Ye, L. Liang, G. Y. Li, and B.-H. F. Juang, "Deep learning based end-to-end wireless communication systems with conditional GAN as unknown channel," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, May 2019.

[7] N. Farsad, M. Rao, and A. Goldsmith, "Deep learning for joint source-channel coding of text," in *Proc. IEEE ICASSP*, Apr. 2018, pp. 2326–2330.

[8] H. Ye, G. Y. Li, and B.-H. F. Juang, "Bilinear convolutional auto-encoder based pilot-free end-to-end communication systems," in *Proc. IEEE ICC*, Jun. 2020, pp. 1–6.

[9] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *IEEE Annu. Allerton Conf. Commun. Control Comput. (Allerton)*, 2016, pp. 341–346.

[10] H. Kim, Y. Jiang, R. Rana, S. Kannan, S. Oh, and P. Viswanath, "Communication algorithms via deep learning," in *Proc. Int. Conf. Learn. Represent.(ICLR)*, 2018.

[11] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," in *Proc. 51st Annu. Conf. Inf. Sci. Syst (CISS)*, 2017, pp. 1–6.

[12] S. Cammerer, T. Gruber, J. Hoydis, and S. ten Brink, "Scaling deep learning-based decoding of polar codes via partitioning," in *Proc. IEEE Globalcom*, Dec. 2017, pp. 1–6.

[13] W. Xu, Z. Wu, Y.-L. Ueng, X. You, and C. Zhang, "Improved polar decoder based on deep learning," in *IEEE Int. Workshop on Signal Process. Syst*, 2017, pp. 1–6.

[14] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmwave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 852–855, Oct. 2018.

[15] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM

systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2017.

[16] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *Proc. IEEE SPAWC*, Jul. 2017, pp. 1–5.

[17] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Model-driven deep learning for joint MIMO channel estimation and signal detection," *IEEE Trans. Signal Process.*, vol. 68, pp. 1702–1715, MAR. 2020.

[18] X. Gao, S. Jin, C.-K. Wen, and G. Y. Li, "Comnet: Combination of deep learning and expert knowledge in OFDM receivers," *IEEE Commun. Lett.*, vol. 22, no. 12, pp. 2627–2630, 2018.

[19] Z. Qin, H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep learning in physical layer communications," *IEEE Wireless Commun.*, vol. 26, no. 2, Apr. 2019.

[20] H. He, S. Jin, C. Wen, F. Gao, G. Li, and Z. Xu, "Model-driven deep learning for physical layer communications," *IEEE Wireless Commun.*, vol. 26, no. 5, pp. 77–83, Oct. 2019.

[21] A. Felix, S. Cammerer, S. Dörner, J. Hoydis, and S. ten Brink, "OFDM-Autoencoder for end-to-end learning of communications systems," in *Proc. IEEE SPAWC*, Jul. 2018, pp. 1–5.

[22] T. J. O'Shea, T. Erpek, and T. C. Clancy, "Deep learning based MIMO communications," *arXiv preprint arXiv:1707.07980*, 2017.

[23] F. A. Aoudia and J. Hoydis, "Model-free training of end-to-end communication systems," *IEEE J. Sel. Areas Commun.*, pp. 2503–2516, Nov. 2019.

[24] V. Raj and S. Kalyani, "Backpropagating through the air: Deep learning at physical layer without channel models," *IEEE Commun. Lett.*, vol. 22, no. 11, pp. 2278–2281, Nov. 2018.

[25] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with BM3D?" in *Proc. IEEE CVPR*, Jul. 2012, pp. 2392–2399.

[26] C. J. Schuler, M. Hirsch, S. Harmeling, and B. Schölkopf, "Learning to deblur," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 38, no. 7, pp. 1439–1451, Jul. 2015.

[27] M. Hradiš, J. Kotera, P. Zemcık, and F. Šroubek, "Convolutional neural networks for direct text deblurring," in *Proc. BMVC*, vol. 10, Sept. 2015, pp. 1–13.

[28] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4509–4522, Sept. 2017.

[29] X.-J. Mao, C. Shen, and Y.-B. Yang, "Image denoising using very deep fully convolutional encoder-decoder networks with symmetric skip connections," *arXiv preprint arXiv:1603.09056*, vol. 2, 2016.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NIPS*, Dec 2014, pp. 2672–2680.

[31] M. S. Sajjadi, B. Scholkopf, and M. Hirsch, "Enhancenet: Single image super-resolution through automated texture synthesis," in *Proc. IEEE ICCV*, Oct. 2017, pp. 4491–4500.

[32] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE CVPR*, Jul. 2017, pp. 4681–4690.

[33] C. K. Sønderby, J. Caballero, L. Theis, W. Shi, and F. Huszár, "Amortised map inference for image super-resolution," *arXiv preprint arXiv:1610.04490*, 2016.

[34] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proc. IEEE CVPR*, Jul. 2018, pp. 9446–9454.

[35] K. Choi, K. Tatwawadi, T. Weissman, and S. Ermon, "Necst: Neural joint source-channel coding," *arXiv preprint arXiv:1811.07557*, 2018.

[36] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 3, Sept. 2019.

[37] T. S. Rappaport *et al.*, *Wireless Communications: Principles and Practice*. prentice hall PTR New Jersey, 1996, vol. 2.

[38] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[39] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear CNN models for fine-grained visual recognition," in *Proc. ICCV*, Dec. 2015, pp. 1449–1457.

[40] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2, 2003, pp. 1398–1402.

[41] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 260–269, Feb. 1967.

[42] S. Yang and L. Hanzo, "Fifty years of mimo detection: The road to large-scale MIMOs," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 1941–1988, Sept. 2015.

[43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[44] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. ICCV*, vol. 2, July 2001, pp. 416–423.

[45] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. Van Gool, "Conditional probability models for deep image compression," in *Proc. IEEE CVPR*, Jun. 2018, pp. 4394–4402.