IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 3, 2019, accepted May 28, 2019, date of publication June 5, 2019, date of current version June 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2920929

Deep Learning-Constructed Joint Transmission-Recognition for Internet of Things

CHIA-HAN LEE, (Member, IEEE), JIA-WEI LIN, PO-HAO CHEN, AND YU-CHIEH CHANG
Institute of Communications Engineering, National Chiao Tung University, Hsinchu 30010, Taiwan

Corresponding author: Chia-Han Lee (chiahan@nctu.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST-107-2221-E-009-035-MY2.

ABSTRACT The widely deployed Internet-of-Things (IoT) devices provide intelligent services with its cognition capability. Since the IoT data are usually transmitted to the server for recognition (e.g., image classification) due to low computational capability and limited power supply, achieving recognition accuracy under limited bandwidth and noisy channel of wireless networks is a crucial but challenging task. In this paper, we propose a deep learning-constructed joint transmission-recognition scheme for the IoT devices to effectively transmit data wirelessly to the server for recognition, jointly considering transmission bandwidth, transmission reliability, complexity, and recognition accuracy. Compared with other schemes that may be deployed on the IoT devices, i.e., a scheme based on JPEG compression and two compressed sensing-based schemes, the proposed deep neural network-based scheme has much higher recognition accuracy under various transmission scenarios at all signal-to-noise ratios (SNRs). In particular, the proposed scheme maintains good performance at the very low SNR. Moreover, the complexity of the proposed scheme is low, making it suitable for IoT applications. Finally, a transfer learning-based training method is proposed to effectively mitigate the computing burden and reduce the overhead of online training.

INDEX TERMS Internet of things (IoT), recognition, transmission, joint source-channel coding, deep learning, deep neural networks, transfer learning, JPEG, compressed sensing.

I. INTRODUCTION

The need for connected device to provide seamless, intelligent services has aroused the prosperity of Internet of things (IoT) [1]–[3]. From industry, business, to consumer electronics, IoT has found applications that may profoundly change human life, improve manufacturing reliability, promote business sales, and make our daily life more convenient [4]–[8]. The intelligence of an IoT device relies on its ability to perform sensing of the outside world and connect to other devices. Therefore, recognition (e.g., image classification) and communication are the two most critical elements of IoT. As IoT devices have limited computational capability and are expected to consume minimum power (due to the limited power availability and the difficulty of frequent battery change), they may not be powerful enough to process some complex recognition tasks locally and timely. Instead, they have to transmit the data they sense to either a centralized server or an edge server, and rely on the server's powerful computational capability to accomplish recognition. After performing recognition, the recognition result may be sent back to the IoT device. Since the number of IoT devices is large and the IoT devices are diverse, the IoT devices are often connected wirelessly. Therefore, how to efficiently transmit data over wireless networks to the server for accurate recognition is crucial.

Tremendous research efforts have been made for efficient data transmission over wireless networks, and in the last few years we have witnessed significant improvement in recognition accuracy thanks to the development of deep learning [9]. However, little research has focused on how to design efficient data transmission schemes with recognition accuracy being the major design criterion. When designing IoT systems, not only do we have to minimize the transmission bandwidth and power consumption, but also the recognition accuracy has to be maximized. Transmitting the entire sensed

The associate editor coordinating the review of this manuscript and approving it for publication was Min Jia.

data would be ideal for recognition, but it requires large transmission bandwidth and consumes significant power. On the other hand, compressing data before transmission saves bandwidth and power, but the recognition accuracy may be sacrificed. In addition, extra care is needed for IoT devices due to their low power, low computational capability nature. Although strong channel codes (also called error correcting codes) allow efficient transmission over wireless networks, their complexity refrains them from being suitable for IoT devices. Similarly, advanced compression (i.e., source coding) techniques can save transmission bandwidth but the complexity may be high. Overall, the design for IoT recognition requires jointly considering the following four aspects: data compression to save bandwidth and transmission power, channel coding for wireless transmission, recognition accuracy, and IoT computational capability. Although compression and channel coding can be jointly designed, i.e., through joint source-channel coding, how to design wireless networks with the design metric of recognition accuracy (instead of data transmission rate) is largely unexplored.

In this paper, we propose an advanced design to optimize compression, communication, recognition, and computation for IoT systems. Since the state-of-the-art recognition is achieved by applying deep neural networks (DNNs) and it has recently been shown that channel codes and joint source-channel coding can also be realized by DNNs, our proposed design is constructed by a DNN architecture. We show in detail how to construct a DNN architecture that meets the design criterion of joint transmission and recognition. The performance of the proposed DNN-constructed joint transmission-recognition scheme is compared to the traditional approaches, i.e., a JPEG-compressed scheme, compressed sensing (CS) with reconstruction, and compressed sensing with direct recognition, showing superior performance. The proposed DNN-based architecture also has the advantage of low computational complexity once trained. Furthermore, a practical online training method, adopting transfer learning, is proposed to mitigate the online training burden and reduce overhead. We demonstrate that the performance of the proposed transfer learning method approaches the performance of the pure online training under the proposed DNN-constructed joint transmission-recognition architecture.

A. RELATED WORK

Although there have been a few works on joint transmission-recognition, those are mainly for automatic speech recognition over wireless networks [10]–[12]. Moreover, their approaches are based on feature extraction with separate channel coding, without considering joint design. On the other hand, there have been a lot of research on image/video transmission over wireless networks with joint design of compression and channel coding, but their target is to restore images/videos instead of performing recognition. Note that for recognition involving images or videos, researches have focused more on compression, without considering joint design with channel coding [13]–[15]. This is due to the difficulty of considering all design factors simultaneously.

Neural network-based channel decoders have been proposed in the 1990's [16]–[21]. However, the performance of these decoders was not competitive enough due to the difficulty of training DNNs. With the recent advance in deep learning, nowadays we can construct neural networks that are deep and have excellent performance. Therefore, in the last couple years a few deep learning-based channel decoders have been proposed [22]–[30]. However, we have not seen DNNs being applied to solving the joint transmission-recognition problem considered in this paper. Note that a deep learning-based joint source-channel coding scheme for image transmission was proposed [31], but their performance metric is image distortion, instead of recognition accuracy considered in this paper. Deep learning-based joint source-channel coding has also been proposed for transmitting texts [32], [33], but again the performance metric is different from this paper.

Deep learning has recently been applied to IoT systems [34]–[39], but none of them applies deep learning for solving the joint transmission-recognition problem as in this paper. Advanced communications schemes for IoT have also been discussed recently [40]–[43], but their approaches are different from ours. In particular, recognition accuracy is not their design criterion. In addition, computation and energy saving issues of IoT devices have recently been investigated [44]–[46], but their approaches are very different from ours.

B. ORGANIZATION

The rest of the paper is organized as follows. In Sec. II, system model is introduced, and in Sec. III, traditional approaches are discussed. Then, a DNN-constructed joint transmission-recognition architecture is proposed in Sec. IV and compared with various traditional methods in Sec. V. Furthermore, a practical transfer training method is proposed in Sec. VI. Finally, this paper is concluded in Sec. VII.

II. SYSTEM MODEL

We consider the scenario that an IoT device transmits data wirelessly to the server to perform recognition. Before transmission, the IoT device may perform data compression (i.e., source coding) and/or channel coding. Whether to adopt compression or channel coding and how to realize compression/channel coding depends on the transmitter architecture, as will be discussed later. The IoT device maintains a direct, point-to-point wireless link to the server. For the wireless channel, both the additive white Gaussian noise (AWGN) model and the Rayleigh fading channel are considered. At the receiver (i.e., the server), depending on the source coding and channel coding schemes adopted at the transmitter, corresponding decoding schemes are performed.

Note that although in this paper we only consider a single IoT device, the overall improvement in the transmission-recognition performance of an IoT network containing

multiple IoT devices is expected to be significant given the improvement in the transmission-recognition performance of individual IoT device.

III. TRADITIONAL APPROACH

In this section, we discuss traditional approaches that may serve as solutions to the considered problem.

An intuitive way of solving the joint transmission-recognition problem is to compress the data (in a lossy or lossless way), apply channel codes, and then transmit the coded data. The received data is decoded, decompressed, and then recognized (e.g., classified). The overall procedure is shown in Fig. 1(a). Note that other physical layer and data link layer functions may be implemented but ignored here since they are not the focus of this paper. The main concern about this approach is complexity, which refrains IoT devices from using the state-of-the-art techniques for compressing and transmitting data.

Another approach is using joint source-channel coding, which theoretically outperforms separate source coding and channel coding. In this approach, data goes through joint source-channel encoding, wireless channel, joint source-channel decoding, and then recognized, as shown in Fig. 1(b). This approach may be impractical due to not only the complexity concerns but also the difficulty of deriving optimal joint source-channel coding.

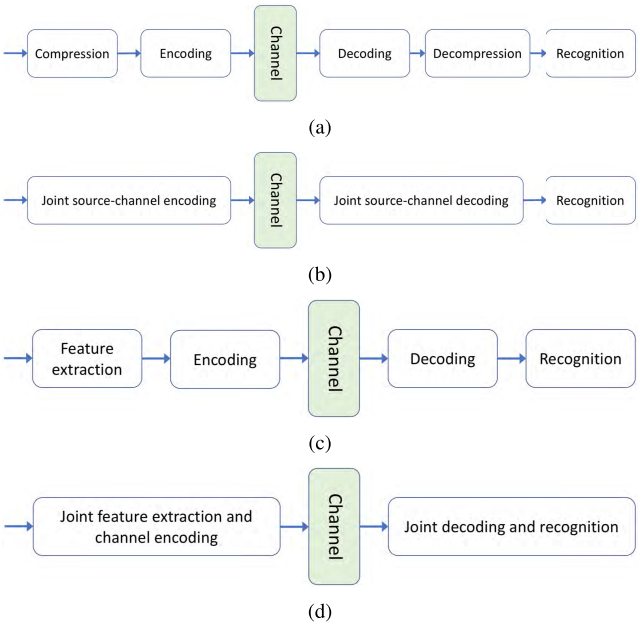

FIGURE 1. Different approaches of realizing recognition over wireless networks. (a) Separate source coding and channel coding. (b) Joint sourch-channel coding. (c) Feature extraction-based approach. (d) Proposed joint transmission-recognition.

Another concern is that our goal is to perform recognition, instead of simply transmitting data with minimum distortion. It is actually the most efficient if data can be compressed to the minimum level such that recognition is barely accomplished. In this way, minimum wireless resource is required for data transmission. However, it is generally difficult to mathematically or systematically derive the relationship between compression and recognition since the mechanism of the state-of-the-art recognition, which is usually done by deep learning, is largely unknown. A possible solution is to extract the features that may be useful for recognition and simply transmit the features for recognition. However, determining the minimum set of features that are useful for recognition is not an easy task at all; otherwise the feature-based recognition methods would have been successful and the DNN-based recognition approach is not required. In this approach, the features of the data are extracted, channel coded, transmitted, channel decoded, and finally recognized, as shown in Fig. 1(c). With the difficulty of deriving the relationship between feature extraction and recognition, further joint design with channel coding with the traditional method seems hopeless.

Note that image and video transmission over wireless systems has long been a hot research topic. However, the goal of these researches is to transmit images/videos wirelessly with minimum image/video distortion under specified rate (the so-called rate-distortion criterion). Instead, our goal is to accomplish the recognition task (i.e., achieving some recognition accuracy) using the minimum wireless resource. Also note that unequal error protection (UEP) is an effective approach but is usually more expensive to implement. Moreover, it is unknown how to optimally combine UEP with recognition. Thus, UEP is not considered in this paper.

IV. PROPOSED DNN-CONSTRUCTED JOINT TRANSMISSION-RECOGNITION

Different from the traditional approaches mentioned earlier, we propose a joint design of source coding, channel coding, and recognition, with the constraint of moderate complexity.

A. METHODOLOGY

In this paper, we focus on the pattern recognition tasks that are difficult to solve, such as image recognition, such that advanced pattern recognition methods are required to achieve good performance. The state-of-the-art recognition is achieved by DNNs [47]–[52]. One advantage of the DNN-constructed architecture over traditional machine learning methods is that the DNNs automatically extract features after training, while traditional machine learning methods usually rely on manual feature extraction. Thus, a well-trained DNN can be used as a feature extractor.

An efficient and high-performance transmission scheme with the target of recognition is more likely to be achieved with designing the transmission scheme and the recognition scheme jointly. Since the state-of-the-art schemes for recognition are implemented by DNNs, the joint transmission-recognition would be the most efficient with an all-DNN architecture. With the breakthrough of implementing channel codes by DNNs, the joint design of compression (feature extraction), channel coding, and recognition becomes possible with an architecture constructed entirely by DNNs.

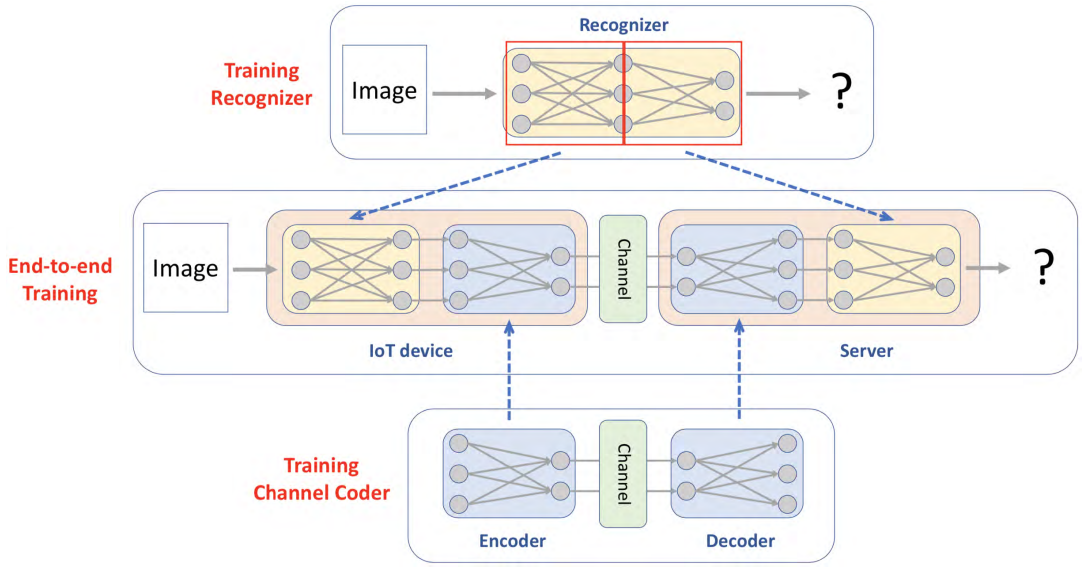

FIGURE 2. Proposed method for constructing a DNN for joint transmission-recognition.

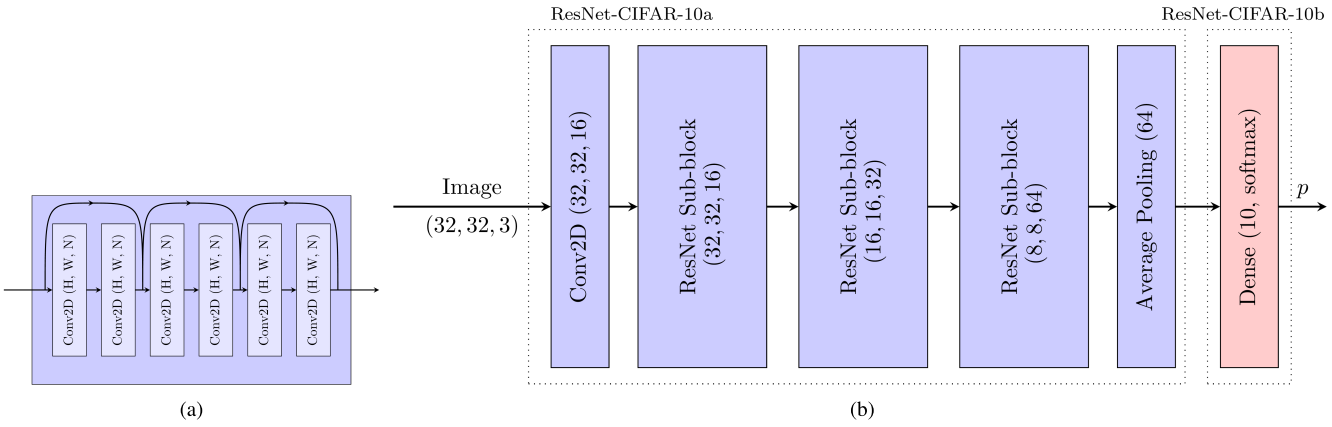

FIGURE 3. ResNet-CIFAR-10: Recognizer implemented by ResNet. (a) ResNet sub-block. (b) ResNet-CIFAR-10.

The DNN-constructed approach also offers other advantages such as low computational complexity [53], meeting the computing requirement of IoT devices.

The proposed DNN-constructed approach is illustrated in Fig. 1(d), showing that joint feature extraction and channel coding (using DNNs) is performed at the IoT device before data transmission, and the received data is jointly decoded and recognized (using DNNs) at the server. To our best knowledge, this is the first time that compression (feature extraction), channel coding, and recognition (e.g., classification) are jointly designed under the framework of deep learning.

The question is what DNN architecture should be used at the transmitter (the IoT device) and at the receiver (the server), respectively. We propose the following steps to construct a DNN for joint source-channel-recognition. The steps are described in the following and summarized in Fig. 2.

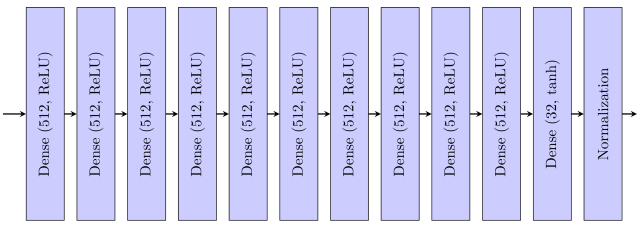

FIGURE 4. DNN-Encoder for a (32, 16) Polar code.

The first step is to train a DNN for recognition (the upper part of Fig. 2). The architecture, including width, depth, and type, of the DNN is determined according to the target recognition accuracy and affordable complexity. The trained DNN is then split into two parts. The first part, which functions as a feature extractor, becomes part of the source-channel encoder at the transmitter, and the second part, which functions as

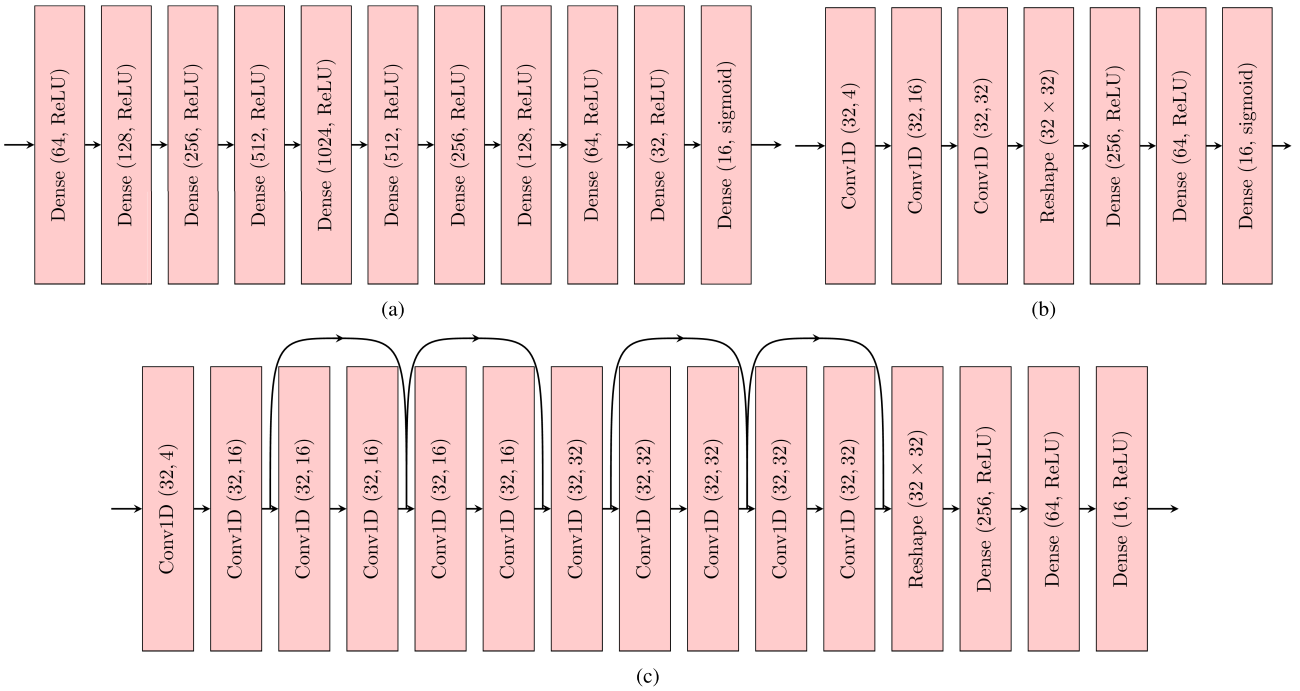

FIGURE 5. Various DNN architectures to implement the channel decoder for a (32, 16) Polar code. (a) Dense network-based decoder. (b) CNN-based decoder. (c) Res-Decoder.

a recognizer, becomes part of the decoder-recognizer at the receiver. The idea behind this is that a well-trained DNN should be able to extract features using the first few layers, and the rest of the layers are for performing recognition. The second step of the proposed method is to train an encoder and a decoder considering the channel, as shown in the lower part of Fig. 2. It is possible to directly, jointly train the encoder and decoder, but the training might be difficult. Therefore, we allow the encoder/decoder to mimic existing channel codes by training the encoder and the decoder separately. For example, the inputs and outputs of an existing channel encoder are used for training a DNN encoder such that the DNN encoder behaves like that channel encoder. Similarly, the received signals after wireless channel are used for training the DNN decoder such that the DNN decoder behaves like that channel decoder. After training, the encoder becomes part of the joint source-channel encoder at the transmitter, and the decoder becomes part of the joint decoder-recognizer at the receiver.

Once the DNNs as compressor (feature extractor), encoder, decoder, and recognizer are obtained, as shown in the middle part of Fig. 2, we perform joint re-training for better performance. Note that since the DNNs at the transmitter and the receiver are jointly trained, this is the approach shown in Fig. 1(d), different from the approach shown in Fig. 1(c).

B. DNN ARCHITECTURE

1) RECOGNIZER

Several DNN architectures have been proposed to achieve outstanding recognition performance. Among them are VGGNet [47], GoogLeNet [48], ResNet [49], FractalNet [50], DenseNet [51], and CapsuleNet [52]. In this paper we adopt the ResNet architecture [49] for recognition (i.e., image classification) because it is one of the state-of-the-art and is easy to set up and train. Moreover, the number of parameters in ResNet is low compared to other architectures, and thus the computation can be relatively low. The adoption of the ResNet architecture is thus suitable for IoT applications. Let us use the tuple (H, W, N) to represent the image height, the image width, and the number of feature maps, respectively, of the 2D-convolution (Conv2D) layer with kernel size 3. Firstly we build an (H, W, N) ResNet sub-block with two (H, W, N) Conv2D layers repeated three times, with skip connections, as shown in Fig. 3(a). Then, a DNN is constructed with the first layer being a (32, 32, 16) Conv2D layer, followed by a (32,32,16) ResNet sub-block, a (16,16,32) ResNet sub-block, and a (8,8,64) ResNet sub-block, as shown in Fig. 3(b). In all convolutional layers, ReLU is used as activation function, and after each convolution layer, batch normalization is adopted [49]. After the average pooling layer, the softmax output is implemented by a dense layer with 10 neurons.

The CIFAR-10 image dataset [54] is used for training. The CIFAR-10 dataset, one of the most widely used datasets for machine learning research, contains 60,000 32×32 color images in 10 different classes, with 6,000 images in each class. Since IoT devices usually deal with low resolution images and are usually instructed to recognize only a few classes of images due to relatively low computing power, the CIFAR-10 dataset is well suited for IoT applications.

Referencing the training setting in [49] with minor revision, we use the cross-entropy loss function and the stochastic gradient descent (SGD) with a mini-batch size of 128. The learning rate starts from 0.01 and is divided by 10 at 80 and 120 epochs, respectively, with the training terminated after 160 epochs. The weight decay coefficient is set to 0.0001 and the momentum parameter is chosen as 0.9. Batch normalization is only used in the feature extraction part and no dropout is used, following [49]. The data augmentation used for training is the same as [49]: we pad 4 pixels at each side of the image and then randomly crop 32×32 pixels from the padded image or its horizontal flip. The overall model is called ResNet-CIFAR-10 in this paper.

2) CHANNEL ENCODER

To construct a DNN architecture for the channel encoder, we use 10 dense layers with each layer having 512 neurons, as shown in Fig. 4. Dense networks are chosen to construct a DNN-based channel encoder because this is the most straightforward way to construct a DNN. ReLU is used as activation function, except at the output layer with 32 neurons where the tanh activation function is chosen. To ensure that the transmitted signal meets power constraint (i.e., $||\mathbf{x}||_2^2 \leq P$ with signal $\mathbf{x}$, maximum power P, and L^2 norm $||\cdot||_2$), a normalization layer is appended and the output from the previous layer, $\mathbf{r}$, is multiplied by $\frac{\sqrt{\mathbf{N}}}{||\mathbf{r}||_2}$, where $\mathbf{N}$ denotes the number of neurons.

To construct a DNN to mimic a channel encoder, we use the input-output map of a (32, 16) Polar code (with 16 information bits as input and 32-bit codewords as output) to train the DNN. Since the number of information bits is 16, there are $2^{16} = 65,536$ possible inputs. In each epoch, the model is trained by all possible inputs in a random order, and the training takes 1500 epochs. The Adam optimizer with learning rate 0.001 is used, and the mini-batch size is 1024. No dropout is adopted. During testing, the trained model is tested by all possible inputs and we use hard decision at the output to evaluate the accuracy. This model is able to reach 100% accuracy of the (32, 16) Polar code after training, meaning that the resulting DNN can be used to accurately generate all codewords of the (32, 16) Polar code.

3) CHANNEL DECODER

Since the channel decoder is usually more complicated to implement, we have tried various DNN architectures, including a dense network-based decoder (Fig. 5(a)), a convolutional neural network (CNN)-based decoder (Fig. 5(b)), and the Res-Decoder (Fig. 5(c)). Some previous works on the design of deep learning-based channel decoders utilize dense networks and CNNs [24]–[26], [30], while ResNet is also tried because it is built on top of CNNs and is a simple yet powerful DNN architecture. Note that although LSTM has been used in some papers to construct DNNs, we do not consider using LSTM since the LSTM model is complex and difficult to train (compared to ResNet), not suitable for

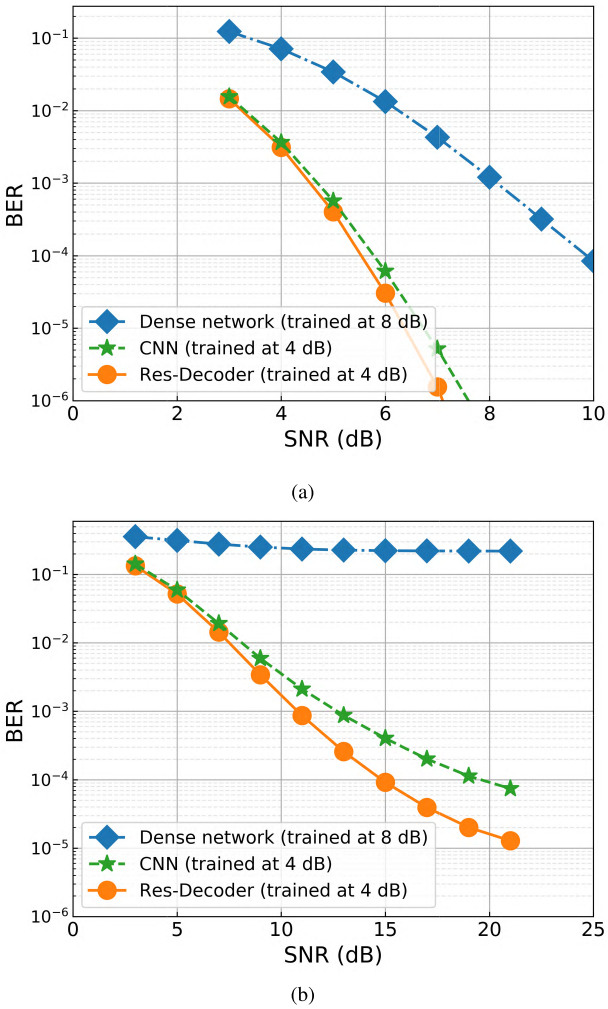

FIGURE 6. BER of various DNN implementations of channel decoders for a (32, 16) Polar code. (a) AWGN channel. (b) Rayleigh fading channel.

IoT applications. The detail of the Res-Decoder is the following. Let us use the tuple (W, N) to represent the image width and the number of feature maps, respectively, of the 1D-convolution (Conv1D) layer. The first two layers are a (32, 4) Conv1D layer and a (32, 16) Conv1D layer. Then, there are four (32, 16) Conv1D layers with skip connections, one (32, 32) Conv1D layer, and four (32, 32) Conv1D layers with skip connections. After reshaping, we use one dense layer with 256 neurons and one dense layer with 64 neurons. The output layer is a dense layer with 16 neurons. All dense layers use the ReLU activation function.

The proposed Res-Decoder is trained at 4 dB signal-to-noise ratio (SNR) and the cross-entropy loss function is adopted. The input for this model is codewords with additive noise and the label is the information bits corresponding to that input. The training and test datasets are random samples from the (32, 16) Polar code, and the numbers of samples in each dataset are 10^6 and 10^7, respectively. The training runs 1200 epochs with mini-batch size 1024. In the convolution layers, same padding is used to keep the block length the

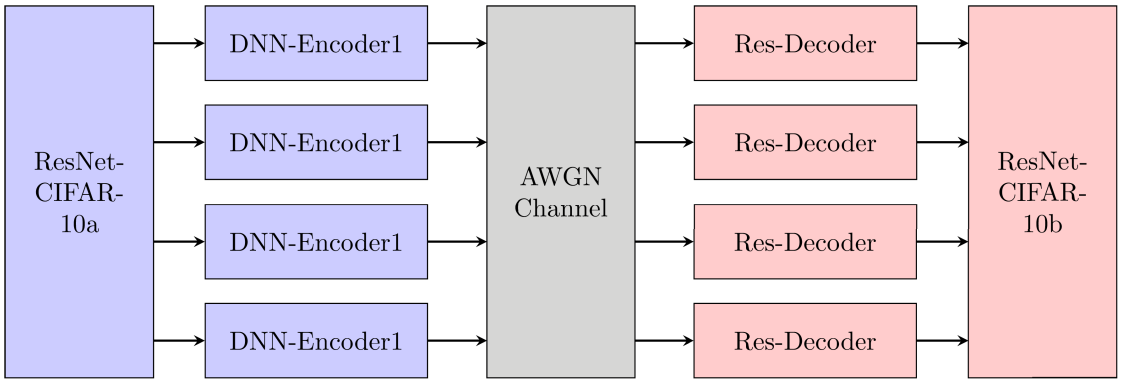

FIGURE 7. DNN-constructed joint transmission-recognition.

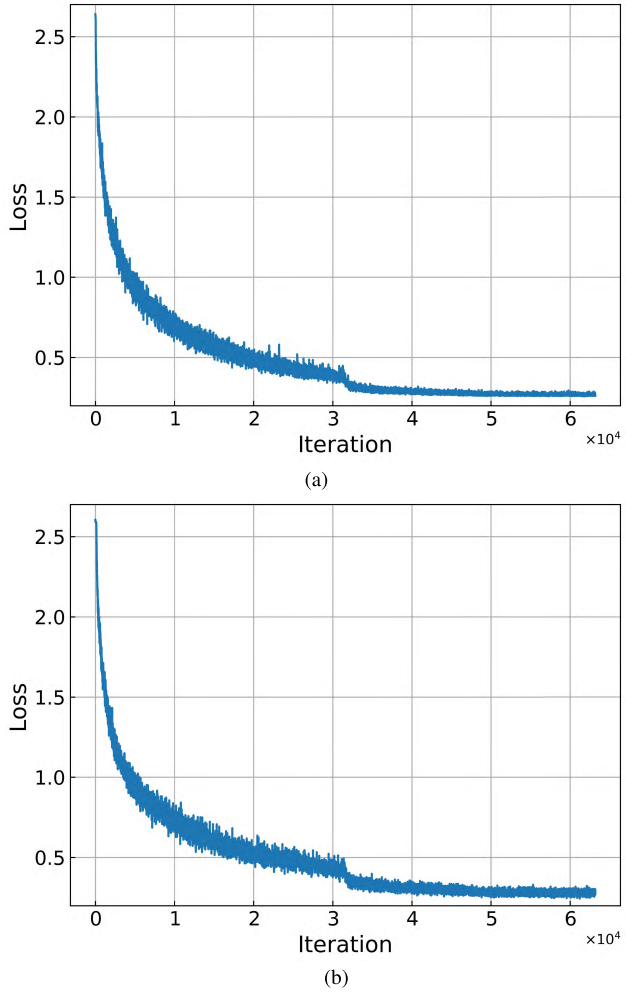

FIGURE 8. Loss vs. epoch during training the DNN-constructed joint transmission-recognition. (a) AWGN channel. (b) Rayleigh fading channel.

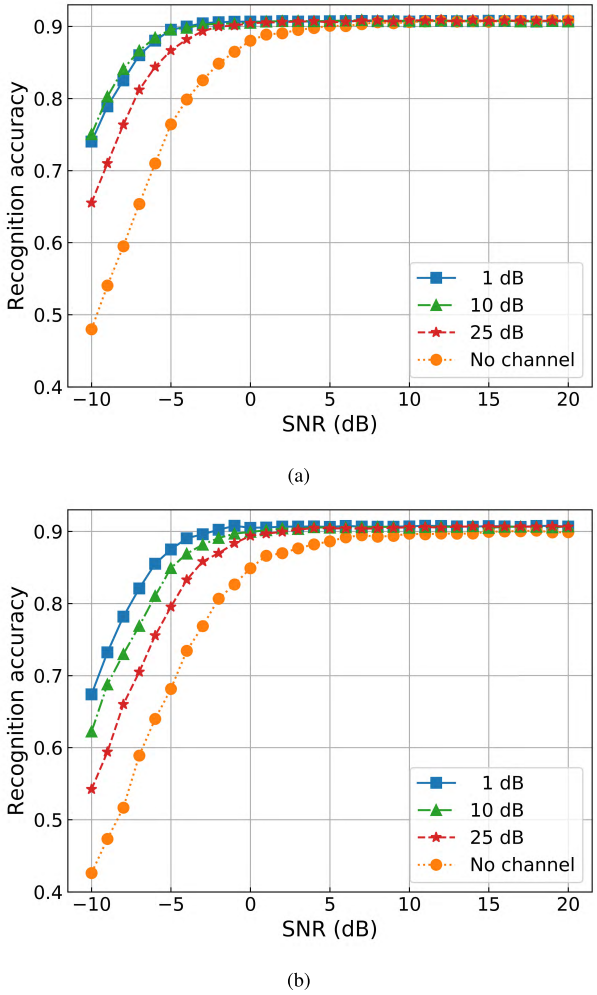

FIGURE 9. Recognition accuracy of the proposed DNN-constructed joint transmission-recognition trained at different SNR, i.e., 1 dB, 10 dB, and 25 dB. The case of training without considering channel (labeled "No channel") is also compared. (a) AWGN channel. (b) Rayleigh fading channel.

same, and the kernel size is 5. The Adam optimizer with learning rate of 0.001 is used, and no dropout is adopted. The dense network-based decoder is trained at 8 dB SNR with the training parameters the same as training the Res-Decoder. The CNN-based decoder is trained at 4 dB SNR, and all training parameters are the same as training the Res-Decoder. Note that trial and error is required when determining the best training SNR.

The bit error rate (BER) performances of the three considered decoders are compared in Fig. 6. We find that the Res-Decoder outperforms both the dense network-based decoder and the CNN-based decoder, and thus we utilize the Res-Decoder in the proposed DNN-constructed joint transmission-recognition architecture.

4) JOINT TRANSMISSION-RECOGNITION

To construct a DNN architecture for joint transmission-recognition, we follow the proposed methodology described in Sec. IV-A and Fig. 2. The ResNet-CIFAR-10 model in Fig. 3 is split into two parts: the convolutional layers and the average pooling layer (ResNet-CIFAR-10a in Fig. 3) form a feature extractor (i.e., a source encoder) while the dense layer (ResNet-CIFAR-10b in Fig. 3) becomes a recognizer. By concatenating the feature extractor and the channel encoder, a joint source-channel encoder is constructed. Since the number of neurons in the last layer of the feature extractor and the first layer of the channel encoder has to be matched, we connect the feature extractor (ResNet-CIFAR-10a in Fig. 3) to four DNN-Encoder blocks (Fig. 4), resulting in the joint source-channel-encoder architecture as shown in the left part of Fig. 7. Then, by concatenating the channel decoder and the recognizer, a joint channel decoder-recognizer is constructed. Again, to match the number of neurons, four Res-Decoder blocks (Fig. 5(c)) are connected to the recognizer (ResNet-CIFAR-10b in Fig. 3), resulting in the joint channel decoder-recognizer architecture as shown in the right part of Fig. 7. The overall DNN-constructed joint transmission-recognition architecture is shown in Fig. 7.

To boost performance, the joint source-channel encoder and the joint channel decoder-recognizer are jointly trained along with the wireless channel. The model is trained at various SNRs, including 1 dB, 10 dB, and 25 dB. The training environment is the same as when we train ResNet-CIFAR-10 (described in Sec. IV-B.1), except that the network parameters of ResNet-CIFAR-10 are used as initialization and then the model is trained for 50 epochs, with the learning rate started at 0.01 and then divided by 10 after half of the epochs. An example of the loss vs. epoch during the training process, for both the cases of AWGN channel and Rayleigh fading channel, is shown in Fig. 8. As shown in Fig. 9, training at 1 dB SNR yields the best performance for the Rayleigh fading channel and similar performance with training at 10 dB SNR for the AWGN channel, and training without considering the channel effect (labeled "No channel" in the figure) results in the lowest recognition accuracy. It is also shown that training along with channel is even more critical for the Rayleigh fading channel. As further shown in Fig. 10, compared to the model without incorporating channel coding (i.e., the model having only ResNet-CIFAR-10a and ResNet-CIFAR-10b, without DNN-Encoder1 and Res-Decoder), including channel coding in the model (i.e., with DNN-Encoder1 and Res-Decoder included) helps improving recognition accuracy for both the AWGN channel and the Rayleigh fading channel,

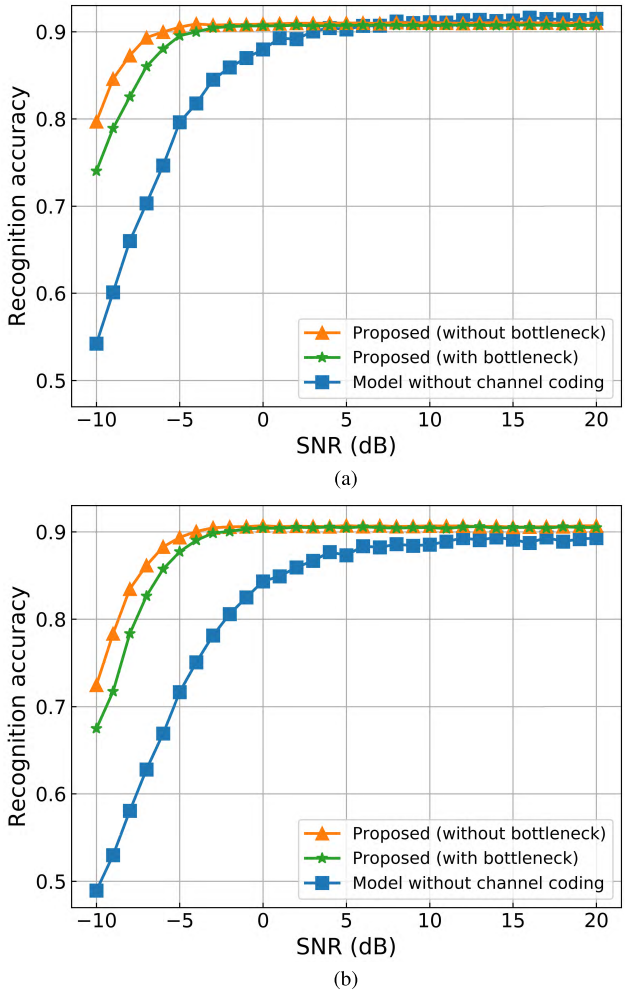

FIGURE 10. Recognition accuracy of the proposed DNN-constructed joint transmission-recognition, with and without bottleneck, and without channel coding. The models are trained at 1 dB. (a) AWGN channel. (b) Rayleigh fading channel.

with negligible accuracy loss at high SNR for the AWGN channel.

Directly concatenating the feature extractor (source encoder) with the channel encoders results in a DNN architecture that the width of the layer shrinks first and then gets larger later. To remove this "bottleneck", the architecture of the joint source-channel encoder is modified to the DNN architecture shown in Fig. 11. The training environment is the same as when we train the DNN architecture with bottleneck. As shown in Fig. 10, the model without bottleneck significantly outperforms both the one with bottleneck and the model without channel coding at moderate to low SNR regimes.

V. COMPARISON WITH VARIOUS TRANSMISSION-RECOGNITION APPROACHES

In this section, we compare the performance of the proposed DNN-constructed joint transmission-recognition architecture with three traditional schemes: a JPEG-compressed scheme and two compressed sensing-based schemes.

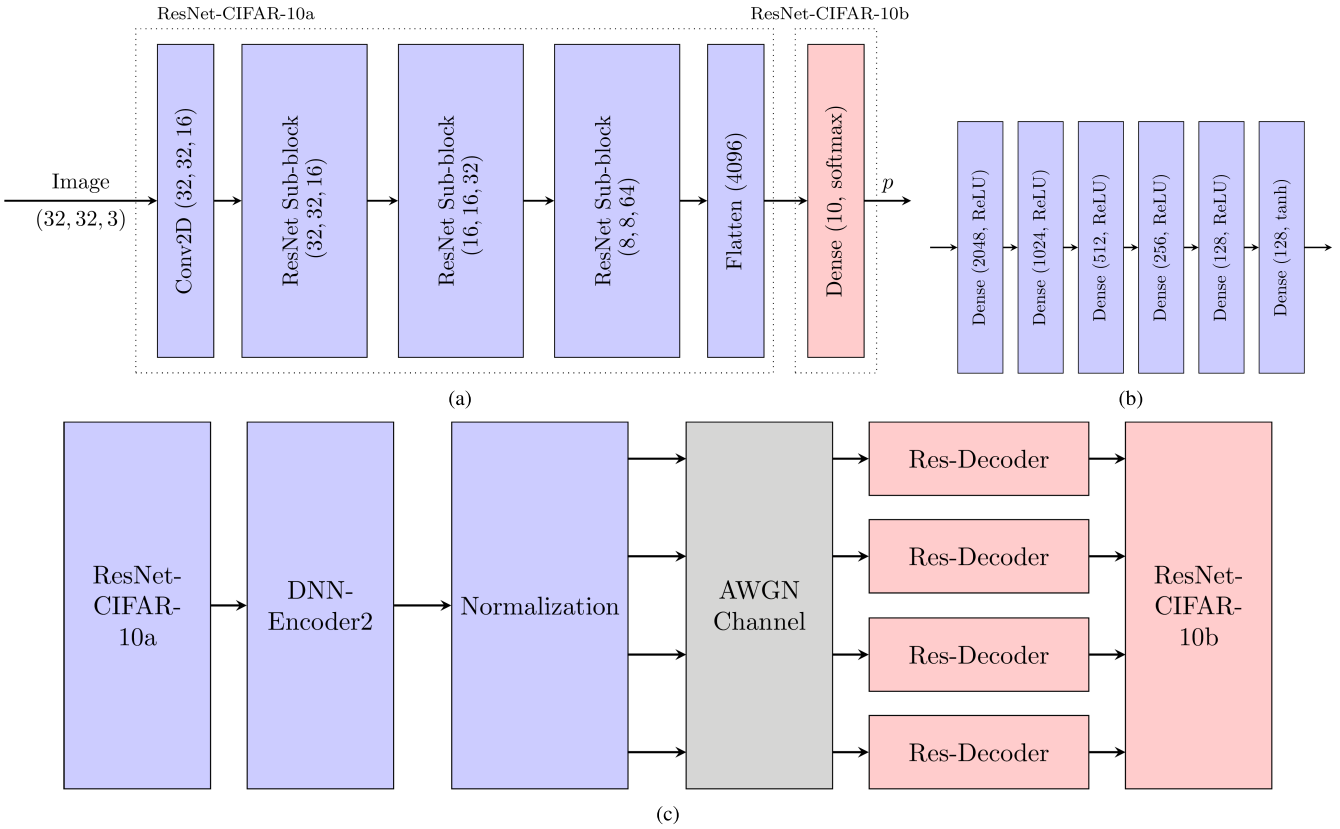

FIGURE 11. DNN-constructed joint transmission-recognition without bottleneck. (a) ResNet-CIFAR-10 without bottleneck. (b) DNN-Encoder2. (c) Overall architecture.

A. JPEG-COMPRESSED SCHEME

The first scheme we compare utilizes JPEG, a commonly used method of lossy compression for digital images, along with channel coding. In the encoding stage of JPEG, the image is converted from RGB to the $\mathrm{YC}_b\mathrm{C}_r$ color space and downsampled. Then, each component of $(\mathrm{Y}, \mathrm{C}_b, \mathrm{C}_r)$ is converted to frequency domain using the two-dimensional type-II discrete cosine transform (DCT). Since human eyes are insensitive to the variation of brightness in high frequency, a quantization matrix is used to reduce the amount of information in the high-frequency components. We encode the images with the downsampling ratios $4:2:0$ and medium quality quantization.

After JPEG compression, the compressed data is channel coded (if channel coding is applied) and then goes through the wireless channel. Note that to ensure the format of JPEG is maintained, the format markers are assumed to be intact from the channel and only the data bits are corrupted. This actually favors the JPEG-compressed scheme in performance comparisons. At the receiver, the received bits are firstly decoded by channel decoder (if channel coding is applied) and then decoded by the JEPG decoder. To have a fair comparison, we use the same recognizer, i.e., ResNet-CIFAR-10, for image recognition.

B. COMPRESSED SENSING WITH RECONSTRUCTION

The second scheme we compare is compressed sensing with reconstruction (CS-R). Here the compressed sensing is performed on digital data instead of at the time of sensing. Given the data (e.g., an image) $\mathbf{x}$ and the sensing matrix $\mathbf{A}$, the output of the compressed sensing is $\mathbf{y} = \mathbf{A}\mathbf{x}$. We adopt the sensing matrix $\mathbf{A} = \phi(\mathbf{D}_n \otimes \mathbf{D}_m)$, where ϕ is the sampling matrix, $\otimes$ is the Kronecker product, and $\mathbf{D}_n = \mathrm{IDCT}(\mathbf{I}_n)$ with IDCT being the inverse discrete cosine transform and $\mathbf{I}_n$ being the identity matrix of size n [55]. The compressed data is channel coded (if channel coding is applied) and then goes through the wireless channel. At the receiver, the data is channel decoded (if channel coding is applied) and then the convex optimization library CVXPY is used for recovering $\mathbf{x}$. After reconstructing the image, the ResNet-CIFAR-10 recognizer is used for image recognition.

C. COMPRESSED SENSING WITH DIRECT RECOGNITION

It is also possible to directly perform recognition with the compressively sensed data without needing reconstruction [56], called compressed sensing with direct recognition (CS-DR) in this paper. In this scheme, compressed sensing converts $\mathbf{x}$ to $\mathbf{y} = \Phi\mathbf{x}$, where $\mathbf{x}$ is a vector with length n and Φ is an $m \times n$ matrix with $n > m$. The compressed

data is channel coded (if channel coding is applied) and then transmitted through the wireless channel. At the receiver, channel decoding is performed (if channel coding is applied) and then Φ is applied to $\mathbf{y}$ to convert $\mathbf{y}$ to a proxy matrix having the same size with $\mathbf{x}$, where the proxy matrix $\bar{\mathbf{x}} = \Phi^T \mathbf{y}$ is obtained by linear projection [56]. Then, the output of the proxy matrix is sent to the recognizer. The ResNet, with batch normalization after each convolution layer with kernel size 1 and stride 2, is adopted as the recognizer. When training the ResNet, we use the following data augmentation: after performing per-pixel mean subtraction, 4 pixels are padded on each side of the image, and a 32×32 crop is randomly sampled from the padded image or its horizontal flip. The training is done at 10 dB SNR, with momentum parameter set as 0.9 and the weight decay coefficient as 0.0001. The cross-entropy loss function is used.

D. PERFORMANCE UNDER ANALOG TRANSMISSION

To transmit data, we may use analog transmission or digital transmission. Here the analog transmission means that the real-valued data is directly used to modulate the signal without going through the steps of quantization. On the other hand, in digital transmission, the data values need to be quantized and converted to bits before modulation and transmission.

In analog transmission, the amplitude of the transmitted signal is proportional to the data value, i.e., amplitude modulation (AM) is adopted, and channel is applied to the signal modulated by the real-valued data. Thus, SNR is defined as the average signal power to the noise variance.

To fairly compare various schemes, we have to make the compression level of different schemes the same. Let γ be the ratio of the number of output elements to the number of input elements, then the proposed DNN-constructed joint transmission-recognition architecture has $\gamma \approx 0.0417$. Due to the constraint of each scheme, not every value of γ is possible. Therefore, the sensing matrices of CS-R and CS-DR are chosen such that $\gamma = 0.04$, and the parameters of the JPEG-compressed scheme are chosen such that the closest γ is $\gamma \approx 0.043$.

Fig. 12 compares the performances of the proposed DNN-constructed joint transmission-recognition, the JPEG-compressed scheme, CS-R, and CS-DR under analog transmission. Both the AWGN channel and the Rayleigh fading channel are considered. The proposed DNN-constructed joint transmission-recognition yields much higher recognition accuracy than the JPEG-compressed scheme, CS-R, and CS-DR, at all SNR, for both the AWGN channel and the Rayleigh fading channel. Note that even at high SNR, CS-R has low recognition accuracy. This is because γ is very low (corresponding to high compression), refraining compressed sensing from reconstructing the original images for recognition. For CS-DR, the recognition accuracy is higher than that of CS-R because the images do not have to be reconstructed before performing recognition. The JPEG-compressed scheme also has poor performance

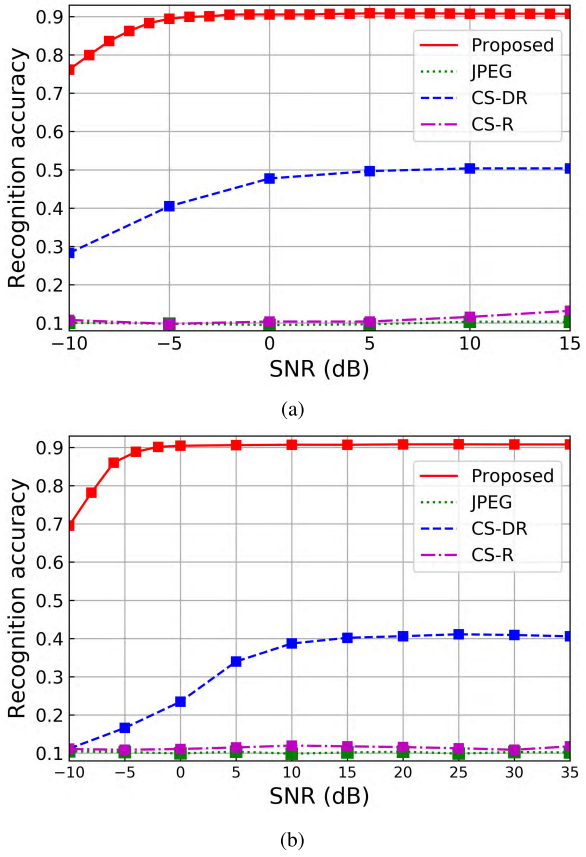

FIGURE 12. Recognition accuracy under analog transmission. "JPEG" is the JPEG-compressed scheme, "CS-DR" denotes compressed sensing with direct recognition, and "CS-R" denotes compressed sensing with reconstruction. Note the difference in the SNR range of the two figures. (a) AWGN channel. (b) Rayleigh fading channel.

since even small changes in JPEG values (due to analog transmission) may corrupt the JPEG structure, refraining it from reconstructing the original image for recognition. We also find that the advantage of the proposed scheme is more significant under the Rayleigh fading channel than under the AWGN channel.

E. PERFORMANCE UNDER DIGITAL TRANSMISSION

For digital transmission, real-valued data needs to be quantized and converted to bits for transmission. In such way, channel codes can be applied. Since IoT devices can only afford low-end processing, we assume 8-bit quantization. For the proposed DNN-constructed joint transmission-recognition, uniform 8-bit quantization is adopted. For the JPEG-compressed scheme and CS-R, since it happens that the data values are in the range of 0 to 255, 8-bit quantization can be directly applied. For CS-DR, as the values vary a lot, a nonlinear 8-bit quantization needs to be applied.

Assume that binary phase shift keying (BPSK) is adopted (note that extending to quadrature phase shift keying (QPSK) is straightforward), and the wireless channel is applied to the signals that are BPSK-modulated. Here we adopt a general approach of modeling the wireless channel that allows

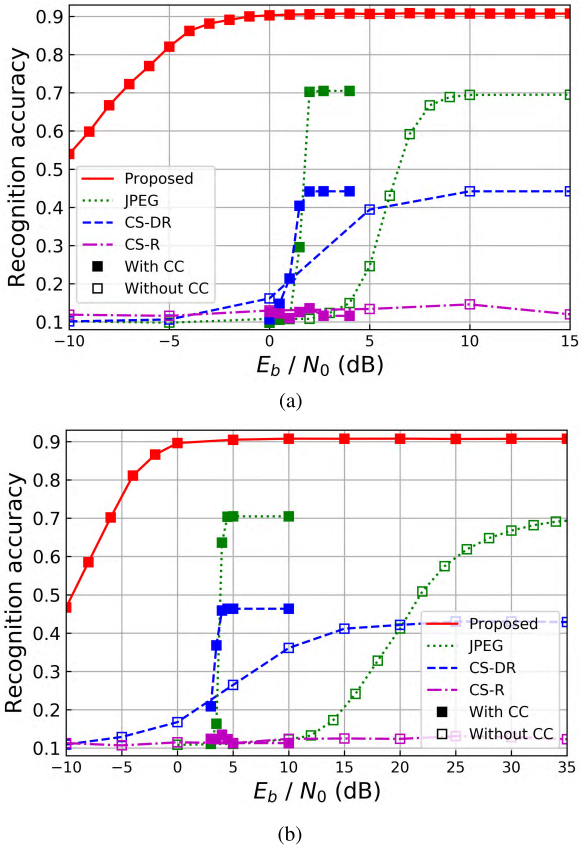

FIGURE 13. Recognition accuracy under digital transmission. "JPEG" is the JPEG-compressed scheme, "CS-DR" denotes compressed sensing with direct recognition, "CS-R" denotes compressed sensing with reconstruction, and "CC" means channel coding. Note the difference in the SNR range of the two figures. (a) AWGN channel. (b) Rayleigh fading channel.

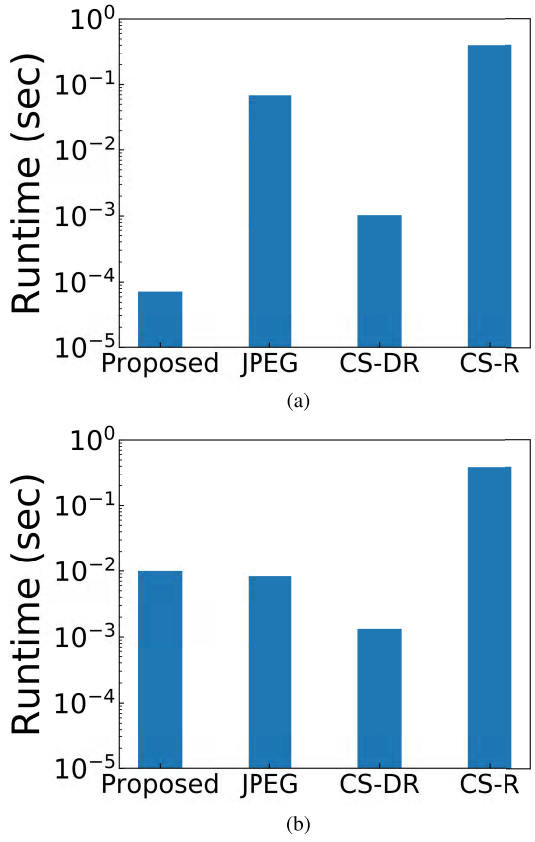

FIGURE 14. Comparison of complexity (in terms of runtime) of various schemes. "JPEG" is the JPEG-compressed scheme, "CS-DR" denotes compressed sensing with direct recognition, and "CS-R" denotes compressed sensing with reconstruction. Note that the *y*-axis is in log scale. (a) Analog transmission. (b) Digital transmission.

incorporating any channel codes under various channels. In this wireless model, the data bits are randomly flipped according to the specified BER given a channel code and an SNR. For example, if a channel code has BER 10^{-4} when the SNR is 2 dB, then 0.01% of the data bits are randomly flipped. Note that the SNR in digital transmission is defined as E_b/N_0, where E_b is energy per bit and N_0 is the noise power spectral density. On the other hand, if no channel coding is applied (i.e., uncoded), the BER mapping according to the uncoded BPSK is used. Since in practical transmission scrambling is usually applied, the random flipping of data bits reflects well the error that may be incurred during the transmission over the wireless channel. In this way, we can simulate the transmission with any channel codes. Since we choose the same (or similar) compression level and all the schemes use 8-bit quantization, they can be compared fairly. At the receiver, the received bits need to be converted to values for later processing. Note that for CS-DR, training does not include the quantization part.

Fig. 13 compares the performance of the proposed DNN-constructed joint transmission-recognition, the JPEG-compressed scheme, CS-R, and CS-DR under digital transmission. Both the AWGN channel and the Rayleigh fading channel are considered, and both the coded version and the uncoded version are compared. For the coded version, we consider an LDPC code, with BER 0.2, 0.1, 5×10^{-2}, 5×10^{-3}, and 1.5×10^{-5} at SNR 0 dB, 0.5 dB, 1 dB, 1.5 dB, and 2 dB, respectively, for the AWGN channel [57], and BER 5×10^{-2}, 1.2×10^{-2}, 5×10^{-4}, and 5×10^{-6} at SNR 3 dB, 3.5 dB, 4 dB, and 4.5 dB, respectively, for the Rayleigh fading channel [57]. Note that although channel codes for low SNR is possible, low-SNR channel codes usually have very small code rate, which is impractical for IoT applications, either due to delay or complexity concerns. Also note that the proposed scheme does not have a coded version since channel coding has already been embedded during the construction of the proposed DNN architecture. From Fig. 13 we see that the proposed DNN-constructed joint transmission-recognition outperforms, with large margin, the JPEG-compressed scheme, CS-R, and CS-DR, at all SNR, either coded or uncoded, for both the AWGN channel and the Rayleigh fading channel. CS-R performs poorly due to high compression, while CS-DR has relatively better performance. The JPEG-compressed scheme performs better than the compressed sensing-based schemes at higher SNR, especially when channel coding is applied. However, all those schemes are much

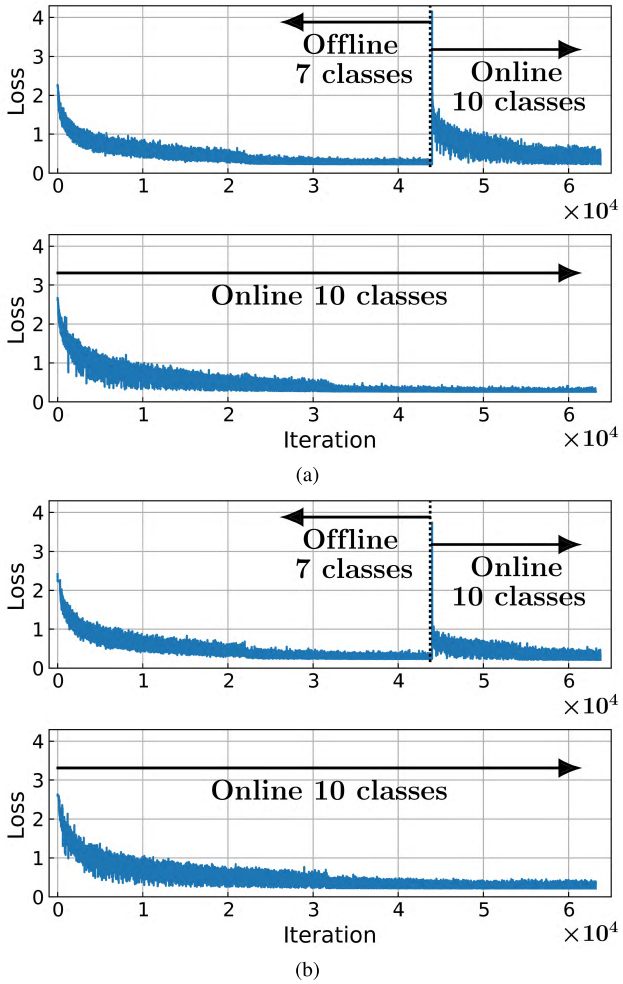

FIGURE 15. Comparison of iteration numbers with transfer learning (upper figure) and with pure online training (lower figure). For transfer learning, 7 classes are trained offline and then 10 classes are trained online. For pure online training, 10 classes are used. (a) AWGN channel. (b) Rayleigh fading channel.

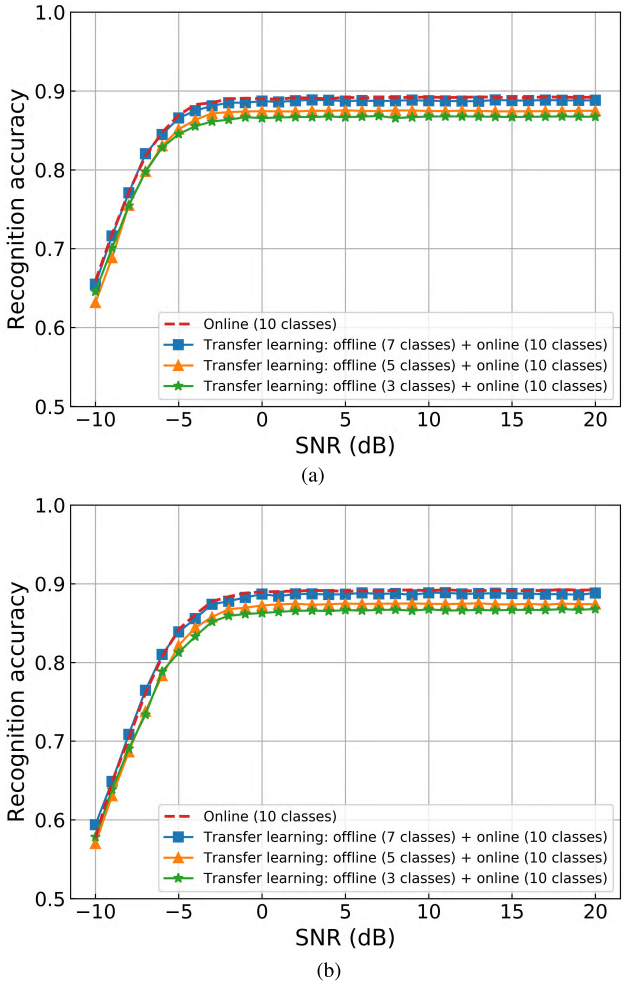

FIGURE 16. Recognition accuracy with transfer learning. (a) AWGN channel. (b) Rayleigh fading channel.

worse than the proposed DNN-based scheme at all SNR. Channel coding generally helps when SNR is large enough, while at low SNR, the uncoded version performs better. This is because at low SNR, the BER with channel coding and the BER without channel coding are similar, but channel coding requires transmitting more bits. Similar to the scenario of analog transmission, we find that the advantage of the proposed scheme is more significant under the Rayleigh fading channel than under the AWGN channel.

Note that since the SNRs of analog transmission and digital transmission are defined in different ways, the performances of the schemes using analog transmission and digital transmission should not be compared directly.

F. COMPLEXITY

The complexity of the proposed DNN-constructed joint transmission-recognition, the JPEG-compressed scheme, CS-R, and CS-DR are compared in Fig. 14 in terms of runtime. Note that the runtime is displayed in log scale in the figure. The desktop computer used for running the programs is equipped with Intel Core i7-8700 CPU@3.20GHz, 16G DRAM, and NVDIA GeForce GTX 1080Ti graphics card. Under analog transmission, the proposed DNN-constructed joint transmission-recognition scheme has much smaller runtime, while under digital transmission, the proposed scheme has moderate runtime. Note that since we use the bit-flipping wireless channel model, the complexity due to channel encoding and channel decoding, which usually take significant amount of time, is not included. Therefore, the runtime comparison actually significantly favors other schemes (because the proposed scheme does not use extra channel encoder and channel decoder). It should be noticed that we do not include the training time for comparison since for IoT applications, the training is likely to be executed only once.

VI. TRANSFER LEARNING FOR REDUCING TRAINING OVERHEAD

After an IoT device is installed, it starts the training process. The IoT device collects data and then uses the data

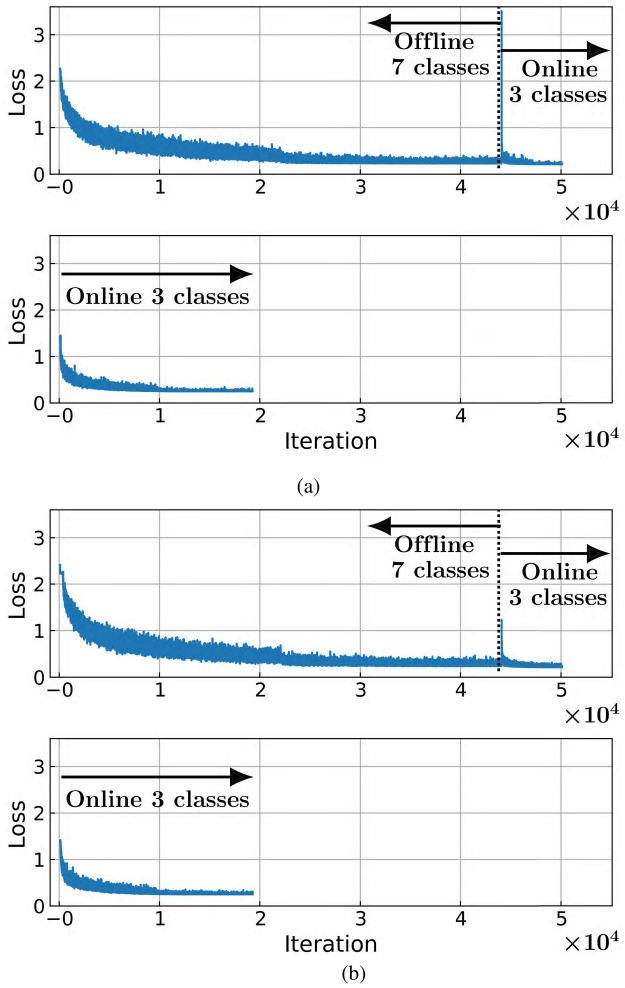

FIGURE 17. Comparison of iteration numbers with transfer learning (upper figure) and with pure online training (lower figure) when the scenes are different from the database. For transfer learning, 7 classes are trained offline, and then different 3 classes are trained online. For pure online training, 3 classes are used. (a) AWGN channel. (b) Rayleigh fading channel.

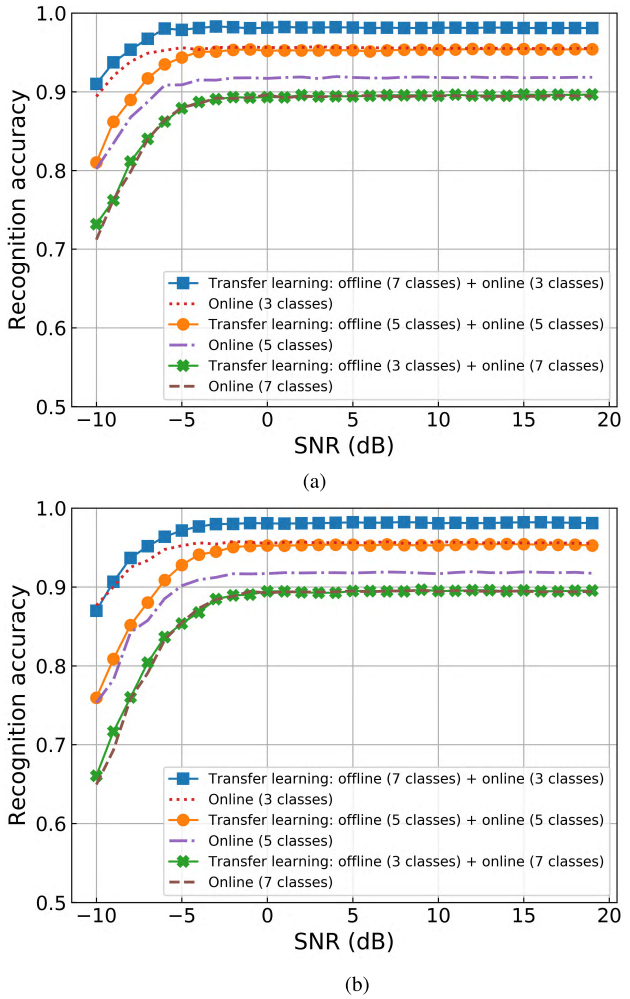

FIGURE 18. Recognition accuracy with transfer learning when the scenes are different from the database. (a) AWGN channel. (b) Rayleigh fading channel.

for training. However, the computing power of IoT devices is low and the training may consume significant power. To mitigate the training burden and reduce overhead, we propose using *transfer learning*, described as follows. Before setup, the IoT device is pre-trained offline (maybe in the factory), using database and predefined channel model. However, the scenario of the offline training can be different from the practical scenario, i.e., the recognition scene may be different from the database and/or the real channel can be different from the channel model. Therefore, after the IoT device is installed, new data is included for online training. With the proposed pre-training and transfer learning, the online learning can be done much faster, with marginal performance loss.

Fig. 15 shows that if we offline train 7 classes (out of the 10 classes) of images from the CIFAR-10 database and then online train all 10 classes (i.e., with transfer learning), the online training part needs much less iteration compared to the pure online training with 10 classes. The online training burden and the training overhead are thus reduced, and the saving in training time is over two-third. Fig. 16 shows that this transfer learning approach has almost the same recognition accuracy with the pure online training. Fig. 16 further shows that if we use fewer classes for offline training, the recognition accuracy decreases but the gap to the performance of the pure online training keeps small.

We also investigate the cases that the scenes are totally different from the database. Fig. 17 shows that even the scenes are different from the database (the 7 classes for offline training and the 3 classes for online training are different), the online training burden and overhead can be reduced to less than one-third. It is, however, surprising that sometimes transfer learning achieves higher accuracy than the pure online training, as shown in Fig. 18. This can be explained as that the offline training serves as pre-training and helps the online training with a better initialization, thus resulting in higher recognition accuracy.

VII. CONCLUSIONS

In this paper we have proposed a DNN-constructed joint transmission-recognition scheme for IoT devices to effectively transmit data to server for recognition. We have illustrated the idea of how to construct the DNN, and then compared the proposed DNN-constructed joint transmission-recognition with the JPEG-compressed scheme, compressed sensing with reconstruction, and compressed sensing with direct recognition. When testing on the CIFAR-10 image database, the proposed scheme significantly outperforms the JPEG-compressed scheme and the compressed sensing-based schemes under analog transmission and digital transmission at all SNR, maintaining high recognition accuracy around 90% when the SNR is larger than -1 dB under analog transmission and when E_b/N_0 is larger than 1 dB under digital transmission. Even at very low SNR, the recognition accuracy is good enough. The complexity of the proposed scheme is also shown to be low. Furthermore, a transfer learning-based training method is proposed and shown to effectively mitigate the computing burden and reduce overhead of online training for IoT devices.

REFERENCES

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.

[2] T. Qiu, N. Chen, K. Li, M. Atiquzzaman, and W. Zhao, "How can heterogeneous Internet of Things build our future: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2011–2027, 3rd Quart., 2018.

[3] L. Da Xu, W. He, and S. Li, "Internet of Things in industries: A survey," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.

[4] S. B. Baker, W. Xiang, and I. Atkinson, "Internet of Things for smart healthcare: Technologies, challenges, and opportunities," *IEEE Access*, vol. 5, pp. 26521–26544, 2017.

[5] B. Ahlgren, M. Hidell, and E. C.-H. Ngai, "Internet of Things for smart cities: Interoperability and open data," *IEEE Internet Comput.*, vol. 20, no. 6, pp. 52–56, Nov. 2016.

[6] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A survey on industrial Internet of Things: A cyber-physical systems perspective," *IEEE Access*, vol. 6, pp. 78238–78259, 2018.

[7] J. Santos, T. Vanhove, M. Sebrechts, T. Dupont, W. Kerckhove, B. Braem, G. V. Seghbroeck, T. Wauters, P. Leroux, S. Latre, B. Volckaert, and F. D. Turck, "City of things: Enabling resource provisioning in smart cities," *IEEE Commun. Mag.*, vol. 56, no. 7, pp. 177–183, Jul. 2018.

[8] S. M. Riazul Islam, D. Kwak, M. Humaun Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, Jun. 2015.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[10] V. Weerackody, W. Reichl, and A. Potamianos, "An error-protected speech recognition system for wireless communications," *IEEE Trans. Wireless Commun.*, vol. 1, no. 2, pp. 282–291, Apr. 2002.

[11] G. H. Lee, J. S. Yoon, Y. R. Oh, and H. K. Kim, "Design of a speech coder utilizing speech recognition parameters for server-based wireless speech recognition," in *Proc. Int. Symp. Intell. Signal Process. Commun. Syst. (ISPACS)*, Nov. 2004, pp. 159–163.

[12] Z.-H. Tan, P. Dalsgaard, and B. Lindberg, "Automatic speech recognition over error-prone wireless networks," *Speech Commun.*, vol. 47, no. 1, pp. 220–242, 2005.

[13] Y. Yan, R. Muraleedharan, X. Ye, and L. A. Osadciw, "Contourlet based image compression for wireless communication in face recognition system," in *Proc. IEEE Int. Conf. Commun.*, May 2008, pp. 505–509.

[14] L. Baroffio, A. Canclini, M. C. A. Redondi, M. Tagliasacchi, G. Dán, E. Eriksson, V. Fodor, J. Ascenso, and P. Monteiro, "Enabling visual analysis in wireless sensor networks," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 3408–3410.

[15] O. Atan, Y. Andreopoulos, C. Tekin, and M. van der Schaar, "Bandit framework for systematic learning in wireless video-based face recognition," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 1, pp. 180–194, Feb. 2015.

[16] J. Bruck and M. Blaum, "Neural networks, error-correcting codes, and polynomials over the binary n-cube," *IEEE Trans. Inf. Theory*, vol. 35, no. 5, pp. 976–987, Sep. 1989.

[17] W. R. Caid and R. W. Means, "Neural network error correcting decoders for block and convolutional codes," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, vol. 2, Dec. 1990, pp. 1028–1031.

[18] A. Di Stefano, O. Mirabella, G. Di Cataldo, and G. Palumbo, "On the use of neural networks for Hamming coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 3, Jun. 1991, pp. 1601–1604.

[19] I. Ortuno, M. Ortuno, and J. A. Delgado, "Error correcting neural networks for channels with Gaussian noise," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 4, Jun. 1992, pp. 295–300.

[20] X.-A. Wang and S. B. Wicker, "An artificial neural net Viterbi decoder," *IEEE Trans. Commun.*, vol. 44, no. 2, pp. 165–171, Feb. 1996.

[21] M. Ibnkahla, "Applications of neural networks to digital communications—A survey," *Signal Process.*, vol. 80, no. 7, pp. 1185–1215, 2000.

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

[23] E. Nachmani, E. Marciano, D. Burshtein, and Y. Be'ery, "RNN decoding of linear block codes," 2017, *arXiv:1702.07560*. [Online]. Available: http://arxiv.org/abs/1702.07560

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

[25] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 144–159, Feb. 2018.

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

[27] L. Lugosch and W. J. Gross, "Neural offset min-sum decoding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1361–1365.

[28] S. Cammerer, T. Gruber, J. Hoydis, and S. ten Brink, "Scaling deep learning-based decoding of polar codes via partitioning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Singapore, Dec. 2017, pp. 1–6.

[29] H. Kim, Y. Jiang, R. Rana, S. Kannan, S. Oh, and P. Viswanath, "Communication algorithms via deep learning," 2018, *arXiv:1805.09317*. [Online]. Available: https://arxiv.org/abs/1805.09317

[30] A. Bennatan, Y. Choukroun, and P. Kisilev, "Deep learning for decoding of linear codes—A syndrome-based approach," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1595–1599.

[31] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," 2018, *arXiv:1809.01733*. [Online]. Available: https://arxiv.org/abs/1809.01733

[32] N. Farsad, M. Rao, and A. J. Goldsmith, "Deep learning for joint source-channel coding of text," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 2018, pp. 2326–2330.

[33] M. Rao, N. Farsad, and A. J. Goldsmith, "Variable length joint source-channel coding of text using deep neural networks," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jun. 2018, pp. 1–5.

[34] J. Tang, D. Sun, S. Liu, and J.-L. Gaudiot, "Enabling deep learning on IoT devices," *Computer*, vol. 50, no. 10, pp. 92–96, 2017.

[35] S. Yao, Y. Zhao, A. Zhang, S. Hu, H. Shao, C. Zhang, L. Su, and T. Abdelzaher, "Deep learning for the Internet of Things," *Computer*, vol. 51, no. 5, pp. 32–41, May 2018.

[36] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan. 2018.

[37] H. Khelifi, S. Luo, B. Nour, A. Sellami, H. Moungla, S. H. Ahmed, and M. Guizani, "Bringing deep learning at the edge of information-centric Internet of Things," *IEEE Commun. Lett.*, vol. 23, no. 1, pp. 52–55, Jan. 2019.

[38] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2923–2960, 4th Quart., 2018.
[39] J. Zhou, Y. Wang, K. Ota, and M. Dong, "AAIoT: Accelerating artificial intelligence in IoT systems," *IEEE Wireless Commun. Lett.*, to be published.
[40] D. Wang, D. Chen, B. Song, N. Guizani, X. Yu, and X. Du, "From IoT to 5G I-IoT: The next generation IoT-based intelligent algorithms and 5G technologies," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 114–120, Oct. 2018.
[41] M. R. Palattella, M. Dohler, A. Grieco, G. Rizzo, J. Torsner, T. Engel, and L. Ladid, "Internet of Things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.
[42] G. A. Akpakwu, B. J. Silva, G. P. Hancke, and A. M. Abu-Mahfouz, "A survey on 5G networks for the Internet of Things: Communication technologies and challenges," *IEEE Access*, vol. 6, pp. 3619–3647, 2018.
[43] N. Javaid, A. Sher, H. Nasir, and N. Guizani, "Intelligence in IoT-based 5G networks: Opportunities and challenges," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 94–100, Oct. 2018.
[44] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for IoT devices with energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, Feb. 2019.
[45] S. Bi and Y. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.
[46] J. Xu, L. Chen, and S. Ren, "Online learning for offloading and autoscaling in energy harvesting mobile edge computing," *IEEE Trans. Cogn. Netw.*, vol. 3, no. 3, pp. 361–373, Sep. 2017.
[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556
[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[49] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[50] G. Larsson, M. Maire, and G. Shakhnarovich, "FractalNet: Ultra-deep neural networks without residuals," 2016, *arXiv:1605.07648*. [Online]. Available: https://arxiv.org/abs/1605.07648
[51] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2261–2269.
[52] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. NIPS*, 2017, pp. 3856–3866.
[53] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.
[54] *The CIFAR-10 Dataset*. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html
[55] *Compressed Sensing in Python*. [Online]. Available: http://www.pyrunner.com/weblog/2016/05/26/compressed-sensing-python/
[56] S. Lohit, K. Kulkarni, and P. Turaga, "Direct inference on compressive measurements using convolutional neural networks," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1913–1917.
[57] Y. Fang, G. Bi, Y. L. Guan, and F. C. M. Lau, "A survey on protograph LDPC codes and their applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 1989–2016, 4th Quart., 2015.

CHIA-HAN LEE received the B.S. degree from National Taiwan University, in 1999, the M.S. degree from the University of Michigan, Ann Arbor, in 2003, and the Ph.D. degree from Princeton University, in 2008, all in electrical engineering. From 1999 to 2001, he served in the ROC Army as a Missile Operation Officer. From 2008 to 2009, he was a Postdoctoral Research Associate with the University of Notre Dame, USA. From 2010 to 2016, he was with the Academia Sinica as an Assistant Research Fellow and then an Associate Research Fellow. Since 2016, he has been with National Chiao Tung University as an Associate Professor. His research interest includes machine learning-based wireless communications and networks. He received the Intel Labs Distinguished Collaborative Research Awards, in 2014, and was named the Intel Labs Distinguished Collaborator, in 2015. He serves as the Industry Presentations and Demonstrations Co-Chair for the IEEE GLOBECOM 2017 and the Symposium Co-Chair for the IEEE GLOBECOM 2019. He was an Editor of the IEEE COMMUNICATIONS LETTERS, from 2014 to 2018, and the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, from 2014 to 2019. He has been an Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS, since 2019.

JIA-WEI LIN received the B.S. degree from the Department of Electrical Engineering, National Tsing Hua University, Taiwan, in 2017. She is currently pursuing the M.S. degree with the School of Institute of Communication Engineering, National Chiao Tung University, Taiwan. Her main research interest includes applying deep learning in wireless communication systems.

PO-HAO CHEN was born in Taipei, Taiwan, in 1996. He received the B.S. degree from the Department of Electrical and Computer Engineering, National Chiao Tung University, Hsinchu, Taiwan, in 2018, where he is currently pursuing the M.S. degree with the School of Institute of Communication Engineering. His main research interest includes deep learning applications in communications.

YU-CHIEH CHANG was born in Kaohsiung, Taiwan, in 1996. He received the B.S. degree from the Department of Optoelectronics and Communication Engineering, National Kaohsiung Normal University, Kaohsiung, in 2018. He is currently pursuing the M.S. degree with the School of Institute of Communication Engineering, National Chiao Tung University. His research interest includes deep learning-based communications.