Wireless Image Transmission Using Deep Source Channel Coding With Attention Modules

Jialong Xu, *Student Member, IEEE,* Bo Ai, *Senior Member, IEEE,* Wei Chen, *Senior Member, IEEE,* Ang Yang, Peng Sun

***Abstract*—Recent research on joint source channel coding (JSCC) for wireless communications has achieved great success owing to the employment of deep learning (DL). However, the existing work on DL based JSCC usually trains the designed network in a specific signal-to-noise ratio (SNR) and applies the network for the scenario with the target SNR. A number of networks are required to cover the scenario with a broad range of SNRs, which is computational inefficiency (in the training stage) and requires large storage. These shortages hinder the use of DL based JSCC for real wireless scenarios. We propose a novel method called Attention DL based JSCC (ADJSCC) that can deal with different SNRs with a single neural network. This design is inspired by the resource assignment strategy in traditional JSCC, which dynamically adjusts the compression ratio in source coding and the channel coding rates according to the SNR. As a resource allocation scheme, Attention Mechanism allocates computing resources to more critical tasks, which naturally fits for the resource assignment strategy. Instead of applying the resource allocation strategy in traditional JSCC, the ADJSCC uses the channel-wise soft attention to scale features according to the information of SNR. We compare the ADJSCC method with the state-of-the-art DL based JSCC method through extensive experiments to demonstrate its adaptability, robustness and versatility. Compared with the existing methods, the proposed method takes less storage and is more robust in the burst channel.**

***Index Terms*—Joint source channel coding, deep learning, deep neural network, attention mechanism.**

I. Introduction

FROM the first generation to the fifth generation of mobile communication systems, the design criteria of modularity are to separate a complicated communication system into several simple modules (e.g., the transmitter is divided into source coder, channel coder and modulation module). Shannon's separation theorem [1] promises the rationality of modularity from the theoretical perspective, while the conditions in the separation theorem are too ideal. The theorem assumes source and channel are stationary, and the latency, complexity and code length are unlimited, which is unrealistic in the practical wireless communication system. Path loss, shadowing effect, multi-path fading, interference and noise affect channel's statistic characteristics; Autonomous driving, smart grid, body area network and smart manufacturing have strict demands on latency; Massive end devices in Internet of Things cannot support high computational complexity due to the limited energy supply and the consideration of hardware cost. When coming back to Shannon's groundbreaking work in 1948, the fundamental problem of communication is that of reproducing at sink either exactly or approximately a message selected at source, which does not suggest the separate source channel coding (SSCC) is the best solution. Shannon states that if natural redundancy of the source is matched with the statistical characteristics of the channel input, to combat channel noise, there is no need to remove the redundancy in the source [2]. From then on, joint source channel coding (JSCC) had become a research hotspot. Gallager gave a mathematical description of the lower limit of the lossless joint source channel coding [3]. Csiszár adopted random coding method for the discrete memoryless system containing discrete memoryless source and discrete memoryless channel [4]. Error exponent represents the exponential decay of block error rate of a code as the code length increases. Zhong et al. studied the error exponent of a point-to-point communication system [5] and a two-user asymmetric communication system [6], and systematically compared the error exponent between SSCC and JSCC. The above results indicate the potential advantages of JSCC over SSCC. In addition to the theoretical work, there are various designs in practical coding schemes. One strategy of JSCC is to reserve the structure of SSCC and apply resource assignment [7], information interaction [8] and unequal error protection [9] to adapt to different signal-to-noise ratios (SNRs). The other strategy of JSCC is to integrate source coding and channel coding as a single process to optimize the communication system [10].

The impressive performance gain brought by deep learning (DL) in computer vision [11] and natural language processing [12] in recent years have been widely concerned by researchers to deal with various wireless communication problems such as routing control [13], channel decoding [14], channel state information feedback [15], channel identification [16]. In the image coding domain, A series of work from Google Research concentrating on image compression show noteworthy results. Toderici et al. [17] and Ballé et al. [18] demonstrated the autoencoder [19] based on DL can achieve better compression than JPEG and JPEG2000, respectively. Minnen et al. further improved the method of [18] that completely surpassed the hand-engineered image codec BPG in [20] and [21]. In the channel coding domain, O'Shea et al. were the first to construct an end-to-end autoencoder with performance close to the Hamming code [22]. Considering high-speed mobile channel and channel mismatch, Xu et al. evaluated the autoencoder to reveal the robustness of the autoencoder [23]. Jiang et al.

Jialong Xu, Bo Ai and Wei Chen are with the State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing 100044, China (e-mail: jialongxu@bjtu.edu.cn; boai@bjtu.edu.cn; weich@bjtu.edu.cn).

Ang Yang and Peng Sun are with vivo Communication Research Institute, Beijing 100015, China (e-mail: ang.yang@vivo.com; sunpeng@vivo.com).

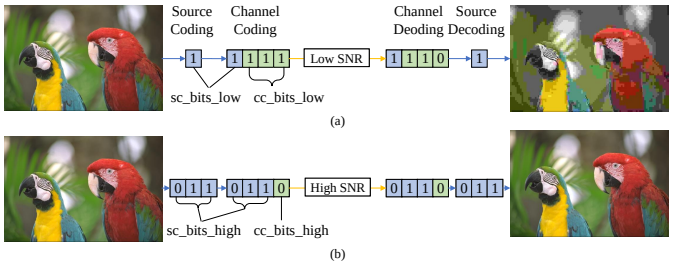


Fig. 1. Illustration of traditional JSCC. (a) In the low SNR, more bits (sc_bits_low) are allocated for channel coding and fewer bits (cc_bits_low) are allocated for source coding, (b) In the high SNR, fewer bits (cc_bits_high) are allocated for channel coding and more bits (sc_bits_high) are allocated for source coding.

proposed Low-latency Efficient Adaptive Robust Neural in [24] and TurboAE code in [25] to demonstrate the superiority of channel coding based on DL under the condition of the medium block length. Considering that the accurate differentiable channel model is hard to obtain in wireless communications, [26] and [27] adopted Reinforcement Learning and Generative Adversarial Networks to circumvent this problem, respectively.

The successive advancement of image coding and channel coding with DL has boosted deep research on JSCC with DL. From the perspective of DL based JSCC design, [28] introduced DL to JSCC domain to process text source for the first time. [29] designed a joint source channel encoder and a joint source channel decoder. This DL based JSCC scheme outperforms the SSCC scheme combining JPEG or JPEG2000 with capacity-achieving channel codes. [30] proposed three hierarchical DL based JSCC schemes leveraging the joint source channel encoder and joint source channel decoder of [29] as basic structures for successive refinement of images. [31] proposed a DL based JSCC scheme that exploits channel output for images, which further improved the performance of DL based JSCC. Considering a discrete channel, [32] proposed a new discrete variational autoencoder model named Neural Error Correcting and Source Trimming based on the maximization of the mutual information between the source and noisy received codeword. Variational inference for Monte Carlo objectives [33] is used to circumvent the non-differential network introduced by the discrete channel. From the perspective of DL based JSCC application, [34] extracted the person's features by employing ResNet-50 network [35] pre-trained on ImageNet dataset [36], then used the joint source channel encoder for dimension reduction and the joint source channel decoder for feature recovery. [37] proposed a joint transmission-recognition scheme that builds the encoder combining the feature extraction with the joint source channel encoder and the decoder combining recognition with the joint source channel decoder to transmit the encoded feature wirelessly to the server for recognition.

All of the existing methods in DL based JSCC imply that the transmitter and the receiver know the SNR to achieve the best performance. That is when the transmitter and the receiver know the specific SNR, the trained network at the specific SNR will be loaded. If the SNR varies due to the fluctuation of the wireless channel, another trained network should be loaded to ensure the optimal performance of the system. Otherwise, the degradation of the performance occurs. Even though DL based JSCC has graceful degradation (robustness) compared to the cliff effect brought by SSCC under deterioration of channel conditions, there is still non-negligible performance loss when the real SNR is inconsistent with the training SNR. This phenomenon can be inferred from Fig. 4. and Fig. 6. of [29] and Fig. 4. (a) of [38]. This disadvantage limits the use of DL based JSCC methods in the real wireless scene. Although we could train multiple networks for different SNRs to alleviate this disadvantage, the time-consuming in the training stage and storage-consuming in the test stage impede it put into practice. Another disadvantage of the existing DL based JSCC methods is the lack of theoretical guidance in the design, even if the performance of existing DL based JSCC methods is better than that of traditional JSCC methods. In this paper, we combine the performance advantage of DL based JSCC with the theoretical advantages of JSCC and propose a novel DL based JSCC method that takes the SNR as input to construct a single network for a range of SNR. The Inspiration of our method originates from the resource assignment strategy adopted by traditional JSCC illustrated in Fig. 1. Specifically, when the channel is in bad condition, for the same image, more bits are allocated to the channel coder and fewer bits are allocated to the source coder. The increased channel bits improve the redundancy to combat the intense channel noise. When the channel is in good condition, for the same image, fewer bits are given to the channel coder and more bits are given to the source coder. The increased source bits are used to improve image quality. In our proposed method, channel-wise soft attention network is used to replace the artificially designed resource allocation strategy to dynamically adjust the compression ratio in source coding and the channel coding rate according to the range of SNR. Compared with [29], we do not need to store the weights of the networks for each SNR. Additionally compared with [31], we no longer need to consume large feedback bandwidth for channel output feedback. Another advantage of our proposed method is that when the burst noise interferes with the channel, our proposed method is more robust than the method proposed in [29].

The rest of this work is organized as follows. In Section II, we introduce the system model based on channel SNR. Then, the proposed method is presented in Section III. In Section IV, we provide the experiment details and simulation results of our proposed method. Section V is dedicated to the evaluation of the performance and the storage of the proposed method compared with the DL based JSCC schemes trained at the specific SNR. Finally, the paper is concluded in Section VI.

II. System Model

Consider a point-to-point image transmission system with SNR feedback as shown in Fig. 2. An input image of size H(height)×W(weight)×C(channel) is represented by a vector $\boldsymbol{x} \in \mathbb{R}^n$, where n = H×W×C. The JSCE encodes $\boldsymbol{x}$ and the feedback SNR μ by the encoding function f_θ to form a vector

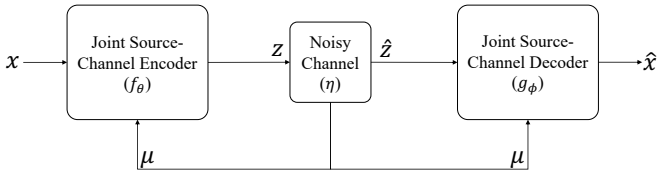


Fig. 2. The model of the point-to-point image transmission system with SNR feedback.

of complex-valued channel input symbols $\boldsymbol{z}$. The encoding process can be expressed as:

$$\boldsymbol{z} = f_\theta(\boldsymbol{x}, \mu) \in \mathbb{C}^k, \tag{1}$$

where k is the size of channel input symbols, θ is the parameter set of JSCE and $\mu \in \mathbb{R}^+$ is the channel SNR that can be perceived by the encoder and the decoder through measurement or feedback. To satisfy the average power constraint, $\frac{1}{k}\mathbb{E}(\boldsymbol{z}\boldsymbol{z}^*) \leq 1$ is imposed on the end of JSCE, where $\boldsymbol{z}^*$ denotes the complex conjugate of $\boldsymbol{z}$. The encoded symbols $\boldsymbol{z}$ are transmitted over a noisy channel represented by the function $\eta : \mathbb{C}^k \to \mathbb{C}^k$. The AWGN channel is considered in our work. The channel output symbols $\hat{\boldsymbol{z}} \in \mathbb{C}^k$ received by JSCD are expressed as:

$$\hat{\boldsymbol{z}} = \eta(\boldsymbol{z}) = \boldsymbol{z} + \boldsymbol{n}, \tag{2}$$

where the vector $\boldsymbol{n} \in \mathbb{R}^k$ consists of independent and identically distributed (i.i.d) samples with the distribution $\mathbb{CN}(0, \sigma^2\boldsymbol{I})$. σ^2 is the noise power. JSCD uses a decoding function g_ϕ to decode the symbols $\hat{\boldsymbol{z}}$ with the assistance of μ, which is expressed as:

$$\hat{\boldsymbol{x}} = g_\phi(\hat{\boldsymbol{z}}, \mu) = g_\phi(\eta(f_\theta(\boldsymbol{x}, \mu)), \mu), \tag{3}$$

where $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ is an estimation of the original image $\boldsymbol{x}$, ϕ is the parameter set of JSCD. The distortion between the original image $\boldsymbol{x}$ and the reconstructed image $\hat{\boldsymbol{x}}$ is expressed as:

$$d(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2, \tag{4}$$

where x_i and $\hat{x}_i$ represents the intensity of the color component of each pixel corresponding to $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$, respectively.

Akin to [29]–[31], we call the image size n, the channel input size k and $R = k/n$ as the source bandwidth, the channel bandwidth and bandwidth ratio, respectively. Under a certain R, the goal is to minimize the expected value of distortion with SNR:

$$(\theta^*, \phi^*) = \arg\min_{\theta,\phi} \mathbb{E}_{p(\mu)}\mathbb{E}_{p(\boldsymbol{x},\hat{\boldsymbol{x}})}[d(\boldsymbol{x}, \hat{\boldsymbol{x}})], \tag{5}$$

where θ^* is the optimal parameter set of θ, ϕ^* is the optimal parameter set of ϕ, $p(\boldsymbol{x}, \hat{\boldsymbol{x}})$ represents the joint probability distribution of the original image $\boldsymbol{x}$ and the reconstructed image $\hat{\boldsymbol{x}}$, $p(\mu)$ represents the probability distribution of the SNR.

III. THE PROPOSED METHOD

The existing methods in DL based JSCC pursue the optimality under a specific SNR. In order to ensure optimality under a range of SNRs, they need to train a number of networks for different SNRs and then choose an appropriate network in the test. In these methods, the information of SNR is only used to decide which network should be chosen and is not used to adjust the coding parameters of neural networks. The existing methods in DL based JSCC lead to multiple computational consumption in the training stage and the requirement of large storage in the test stage.

The goal of the proposed method is to design a neural network for joint source channel encoding and decoding, which could be applied for a range of SNRs. The proposed method is motivated by the resource assignment strategy in the traditional concatenated source channel coders [39] which concatenate the source encoder and the channel encoder, and adjust the compression ratio and the channel coding rate according to the SNR to achieve the optimal quality of reconstructed images under the limited bandwidth. To transmit an image with some fixed bandwidth resource in the low SNR regime, it is desired to compress the source with a high ratio and use more resource to increase the redundancy in channel coding. On the other hand, in the high SNR regime, it would be better to compress the source with a low ratio and waste less resource in channel coding. However, the existing DL based JSCC methods do not support a flexible network structure that can automatically change and adapt to the channel state. Furthermore, it is not trivial how to design a neural network that can provide such flexibility for JSCC.

To conquer this challenge, we employ the attention mechanism, which is a technique of DL widely used in natural language processing [40]–[42] and computer vision [43]–[46]. It refers to an additional neural network that can rigidly select certain features or assign different weights to different features in the original neural network. Based on the attention mechanism, relevant features corresponding to the specific task can be filtered out from a large amount of features. We introduce the attention mechanism to our proposed method Attention DL based JSCC (ADJSCC) to implement the strategy of concatenated source channel coders.

The architecture of ADJSCC has two parts, i.e., the neural encoder at the transmitter and the neural decoder at the receiver. The neural encoder in ADJSCC corresponds to the source encoder and the channel encoder in concatenated source channel coders. It usually consists of multiple non-linear layers. An ingenious analogy is that the first few layers of the neural encoder are regarded as the source encoder, and the remaining layers of the neural encoder are regarded as the channel encoder. The similar analogy is adopted in the decoder. To mimic the strategy of concatenated source channel coders, the size of the output after source encoding should be adjusted according to the SNR input. In addition, for different images and different channel conditions, it is still unknown about the appropriate number of layers to execute source coding and channel coding. The proposed method is able to dynamically adjust the neural network via the attention

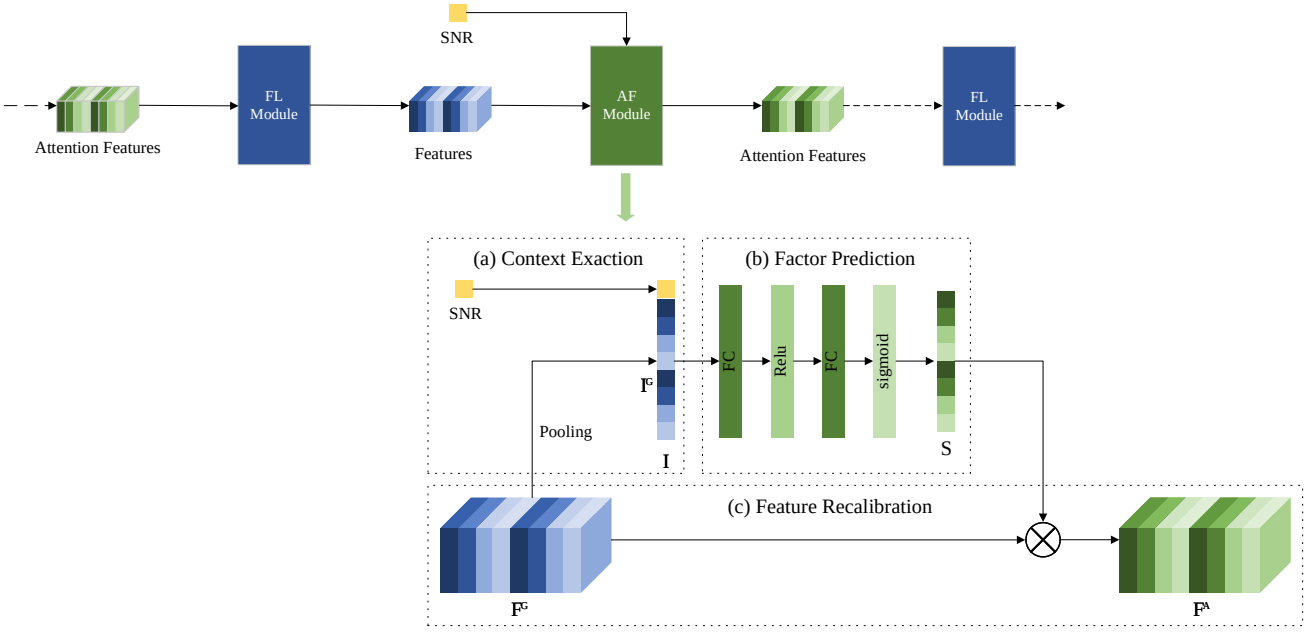


Fig. 3. The relationship between FL modules and AF modules in ADJSCC. FL Module based on convolution layers has been studied in several works [29]–[32]. AF Module contains three parts: (a) Context Extraction, (b) Factor Prediction, (c) Feature Recalibration.

mechanism, which is akin to the adaptation of the source compression ratio and the channel coding rate according to the SNR.

Now we provide the design of the proposed ADJSCC method. Both of the proposed encoder and decoder are constructed by two types of modules, i.e., feature learning (FL) module and attention feature (AF) module. FL modules and AF modules are connected alternately as shown in the upper part of Fig. 3. FL module learns features from the input of the FL module represented, and AF module takes the SNR and the output of the FL Module as the input and produces a sequence of scaling parameters. The product of the outputs of the FL module and the AF module can be seen as a filtered result of the FL module. The design of the FL Module based on convolution layers has been studied in several works [29]–[32]. Therefore we focus on the design of AF Module.

The ideal hard attention generates a mask with elements of either 0 or 1 changes the size of the effective (i.e., nonzero) features. However, the non-differentiable property of the loss in hard attention hinders the execution of the backpropagation algorithm in the training stage. Soft attention is generally adopted instead of hard attention to facilitate backpropagation. Similar to the transform coding in data compression which converts the signal in the time domain to the code in the transformed domain, the features extracted by FL module can be regarded as the signal components on the convolution kernel. These features have different contributions to the features of the last layer of the encoder to be transmitted in the wireless channel. Adding scale parameter to each of these features reveals the correlation between each of the features and the features of the last layer. The greater the scale parameter, the higher the correlation, that is, the more we need to pay attention to this feature. The aforementioned mechanism is channel-wise soft attention. Different from the meaning of the channel in the wireless channel, here the channel in the channel-wise soft attention represents the feature channel.

The architecture of AF Module based on channel-wise soft attention is shown in the lower part of Fig. 3. Let $F^G = [F_1^G, F_2^G, \cdots, F_c^G] \in \mathbb{R}^{h\times w\times c}$ denote the features extracted by the FL Module. c is the number of the features and $h \times w$ is the size of each feature. $F_A = [F_1^A, F_2^A, \cdots F_c^A] \in \mathbb{R}^{h\times w\times c}$ denotes the scaled features by the AF Module. F_i^A is the scaled feature corresponding to $F_i^G, i = 1, 2, \cdots, c$. For F_i^A, channel-wise soft attention is employed to dynamically recalibrate F_i^G according to the context information I, which contains the wireless channel information and the feature information. The AF module contains three parts, i.e., Context Extraction, Factor Prediction and Feature Recalibration. Context Extraction generates a low dimensional representation of the context information. Factor Prediction calculates the scale factors of the features for each feature by using a neural network. At last, the Feature Recalibration scales F^G to F^A.

1) Context Extraction: Context information includes channel SNR μ and feature information I^G. Features of images are usually extracted by convolution kernels limited in a local receptive field. Hence these features usually cannot perceive the information out of this region especially in the lower feature extraction with small kernel size. Global average pooling $G(\cdot)$ is to extract the global information by averaging the values of elements u_{ij} in the feature F_i^G:

$$I_i^G = G(F^G) = \frac{1}{h \times w} \sum_{i=1}^{h} \sum_{j=1}^{w} u_{ij} \in \mathbb{R}. \tag{6}$$

Channel information I^C and global information of feature I_i^G are concatenated to form the context information I:

$$I = (\mu, I_1^G, I_2^G, \cdots, I_c^G) \in \mathbb{R}^{c+1}. \tag{7}$$

2) Factor Prediction: We apply a factor prediction neural network $P_\omega(\cdot)$ to predict the scale factor S. In order not to excessively increase the complexity, $P_\omega(\cdot)$ is a simple neural network consisting of two fully connected (FC) layers. The first FC layer with ReLU is to increase non-linearity and the last FC layer with sigmoid is to limit the output range:

$$S = P_\omega(I) = \sigma(W_2\delta(W_1 I + b_1) + b_2) \in \mathbb{R}^c, \tag{8}$$

where W_1 and b_1 refer to the weights and the biases of the first FC layer, W_2 and b_2 refer to the weights and the biases of the second FC layer, δ and σ represent the activation function ReLu and sigmoid, respectively. ω = (W_1, b_1, W_2, b_2) is the parameter set of the factor prediction neural network.

3) Feature Recalibration: To express more clearly, we rewrite the scale factor $S = [S_1, S_2, \cdots, S_c] \in \mathbb{R}^c$. Feature Recalibration is a channel-wise multiplication as:

$$F_i^A = R(F_i^G, S_i) = S_i \cdot F_i^G, i = 1, 2, \cdots, c. \tag{9}$$

Our AF Module is shown in Algorithm 1. Different from the attention mechanism used in computer vision, we use SNR as the wireless channel information combined with the image inherent features to compute channel-wise attention. Additionally, our motivation originates from the thought that the number of bits to allocate to source coding and channel coding is affected by different SNRs, that caused an adaptive architecture for channel SNR, while the attention mechanism used in computer vision is devoted to the performance improvements for deep architectures.

Algorithm 1 AF module

Input: the features F^G, the SNR information μ
Output: the attention features F^A

calculate the size of the features: $(height, width, channel) = \text{size}(F^G)$
calculate the global information of features: $I^G = \text{GlobalAveragePooling}(F^G)$
calculate the context information $I = \text{concatenate}(\mu, I^G)$
calculate the scale factor $S = P_\omega(I)$
convert the scale factor S to the channel-wise scale factor $S_i, i = 1, 2, \cdots, c$
convert the features F^G to the channel-wise feature $F_i^G, i = 1, 2, \cdots, c$
for i = 0 : 1 : c **do**
    the channel-wise attention feature: $F_i^A = S_i \cdot F_i^G$
end for
convert the channel-wise attention feature $F_i^A, i = 1, 2, \cdots, c$ to the attention features F^A
return F^A

IV. Experiments And Simulation

The first DL based JSCC architecture proposed by [29] consists of tandem modules for image source. Each module consists of a convolution layer followed by a parametric ReLU (PReLU) in the non-last layers or a sigmoid activation function in the last layer. It has shown comparable performance to the standard SSCC scheme (JPEG/JPEG2000 + LDPC). [31] further improved the performance by introducing the generalized normalization transformations (GDN) as a normalization method and widening the channel of the convolution layer for each module. To prove the efficiency of our proposed ADJSCC scheme, we adopt the state-of-the-art DL based JSCC architecture proposed by [31] as the basic DL based JSCC (BDJSCC) architecture shown as Fig. 4. The layers of the BDJSCC Encoder except the normalization layer, the reshape layer and the power normalization layer are divided into five modules. Each of the first four modules consists of a convolution layer, a GDN layer [47] and a PReLU layer [48]. The fifth module only consists of a convolution layer and a GDN layer. Similarly, The layers of the BDJSCC Decoder except for the normalization layer and the reshape layer are divided into five modules. The first four modules have the same construction consisting of a transposed convolution layer, a GDN layer and a PReLU layer. The only difference between the last module and the first four modules is that the sigmoid layer replaces the PReLU layer. The notation $F \times F \times K|S$ in a convolution/transpose convolution layer denotes that it has K filters with the size F and stride down/up S.

The corresponding ADJSCC architecture[1] are shown in Fig. 5. Five modules in BDJSCC Encoder are considered as five FL Modules, each of which is followed by an AF Module except the last FL Module. The output of the previous FL Module is one input of the present AF Module and the output of the present AF Module is the input of the next FL Module. The other input of the AF Module is the SNR come from the wireless channel. The same way is adopted to construct the ADJSCC Decoder. By changing the output channel number in the last convolution layer of the encoder, different bandwidth ratio can be obtained. To compare with the existing method proposed in [31], we comply with the loss function and the metric of [31]. The loss function is the average mean squared error (MSE) between the original image $\boldsymbol{x} \in \mathbb{R}^n$ and the reconstruction image $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ over N transmitted images:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} d(\boldsymbol{x}^{(i)}, \hat{\boldsymbol{x}}^{(i)}), \tag{10}$$

where $\boldsymbol{x}^{(i)}$ and $\hat{\boldsymbol{x}}^{(i)}$ represent the original image sample and the reconstructed image sample, respectively. N is the number of image samples. The PSNR is defined as follows:

$$\text{PSNR} = 10\log_{10}\frac{\text{MAX}^2}{\text{MSE}}(\text{dB}). \tag{11}$$

To calculate the average PSNR under certain channel condition and dataset, the MSE in Eq. (11) is usually replaced by the average MSE.

[1]Source codes for constructing the ADJSCC architecture and the BDJSCC architecture are available at: https://github.com/alexxu1988/ADJSCC.

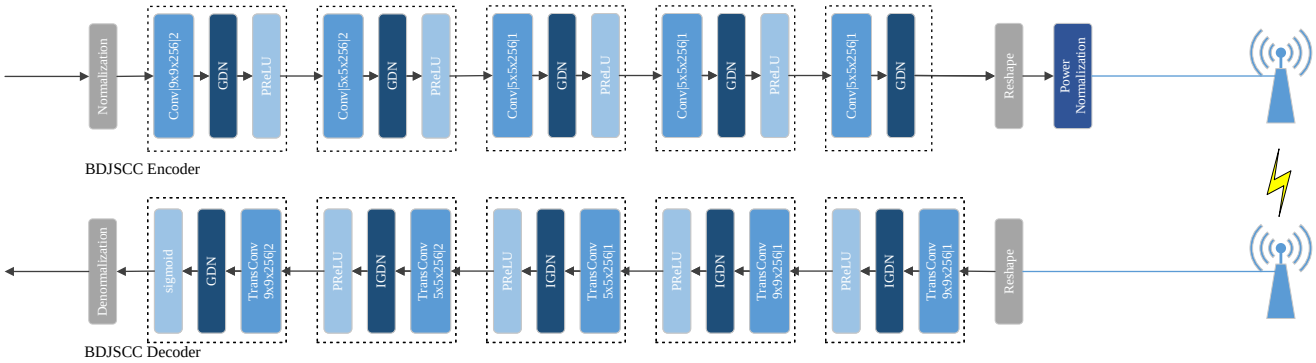


Fig. 4. The architecture of BDJSCC proposed in [31] with hyperparameters. Convolution and transposed convolution are parameterized by $F \times F \times K|S$, where F and K are the filter size and the number of filters, respectively. In the convolution layer, S represents downsampling strides. In the transposed convolution layer, S represents downsampling strides.

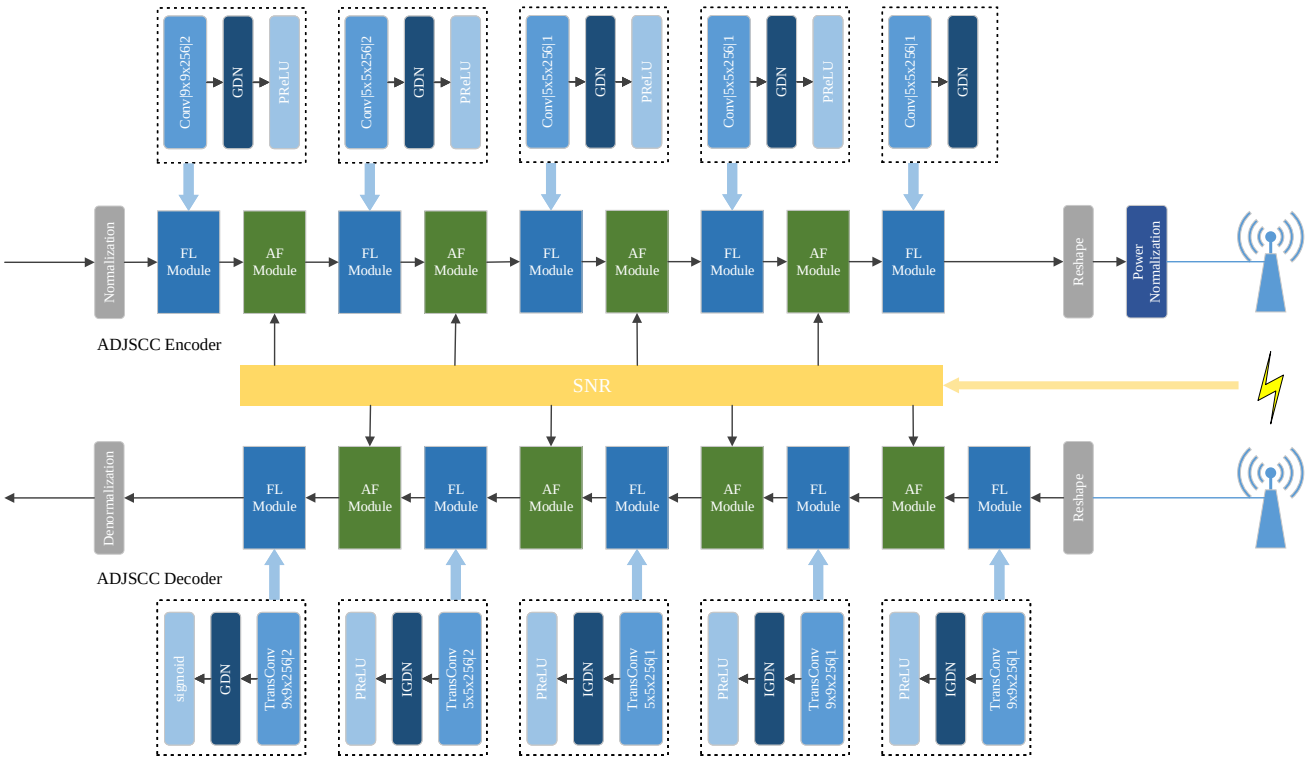
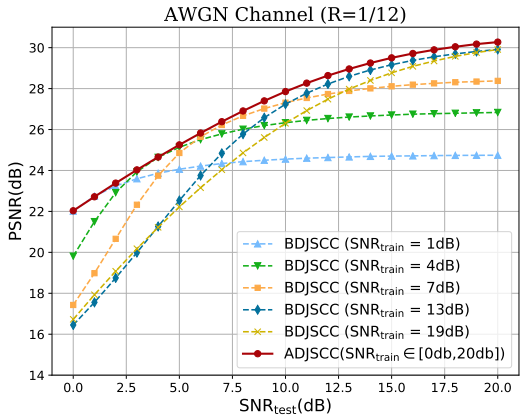


Fig. 5. The architecture of ADJSCC proposed in this paper. The FL Modules of ADJSCC consists of a convolution layer, a GDN layer and a PReLU (or sigmoid) layer. Each FL Module is followed by an AF Module except the last FL Module in the encoder and the decoder. The SNR coming from channel feedback is the other input of the AF Module.

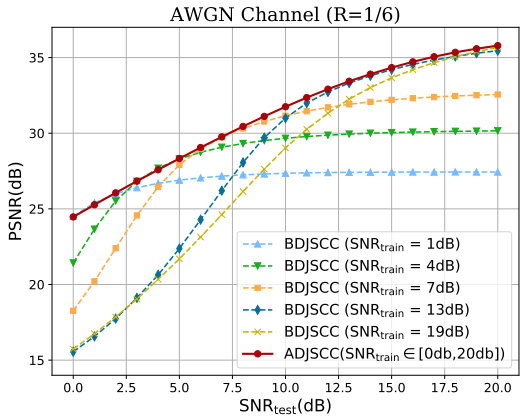
We use Tensorflow [49] and its high-level API Keras to implement the BDJSCC and ADJSCC models. Consistent with the work [31], Adam optimizer with a learning rate of 10^{-4} and the batch size with 128 is chosen to optimize the models. To measure the training efficiency, we fix the training epoch to 1280 and pick up the model with the minimum loss value on a test dataset as the best model. Unless stated otherwise, CIFAR10 [50] is used for training and evaluating the BDJSCC and ADJSCC models. CIFAR10 dataset consists of 60000 $32 \times 32 \times 3$ color images in 10 classes and each class has 6000 images. Training dataset and test dataset contain 50000 images and 10000 images, respectively. Our mission is to reconstruct the transmitted images with minimum distortion on the receiver. Thus we only focus on the relationship between image pixels. The image labels are useless for our task. To evaluate the performance under the specific SNR, each image in the test dataset is transmitted 10 times to alleviate the effect caused by the randomness of the channel noise. All of our experiments are performed on a Linux server with twelve octa-core Intel(R) Xeon(R) Silver 4110 CPUs and sixteen GTX 1080Ti GPU. Each experiment runs on six CPU cores and a GPU.

Since no previous approach paid attention to the arbitrary SNR with a single model, we compare our approach with the BDJSCC method trained at special SNR to prove the superiority of the ADJSCC method.

A. The adaption of ADJSCC



(a)

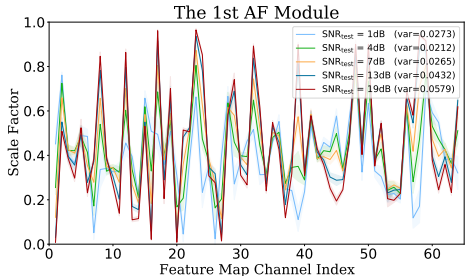


(b)

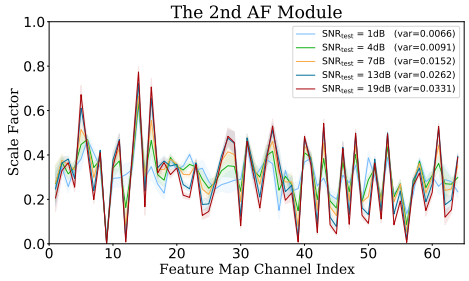
Fig. 6. Performance of the ADJSCC and BDJSCC on CIFAR-10 test images. (a) R =1/12 and (b) R=1/6. Each curve of ADJSCC is trained under a uniform distribution of SNR from 0 dB to 20 dB. Each curve of BDJSCC is trained at a specific SNR.

We first consider the performance of our proposed ADJSCC on the AWGN channel expressed as Eq. (2). The proposed ADJSCC method is an adaptive SNR method, it needs to be trained under a range of SNR to express its adaptivity. However, BDJSCC should be trained at a fixed SNR point and evaluated at the same SNR point to get the optimal performance. In the following experiment, the ADJSCC model is trained under a uniform distribution of $\mathrm{SNR}_{\mathrm{train}}$ from 0 dB to 20 dB, and all of the BDJSCC models are trained at specific $\mathrm{SNR}_{\mathrm{train}}$ = 1 dB, 4 dB, 7 dB, 13 dB, 19 dB, respectively, which are adopted in [29]. We evaluate the performance of both ADJSCC and BDJSCC models at specific integer $\mathrm{SNR}_{\mathrm{test}} \in [0, 20]\mathrm{dB}$. Fig. 6 compares the ADJSCC method with the BDJSCC method at the bandwidth ratio R=1/12, 1/6, respectively. In Fig. 6(a) with R=1/12, the performance of the ADJSCC model is better than the performance of any BJSCC model trained at the specific $\mathrm{SNR}_{\mathrm{train}}$. With the increase of the $\mathrm{SNR}_{\mathrm{test}}$, the ADJSCC model brings a gradually increased performance, outperforming the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$= 1 dB) by a margin of at most 6 dB. With the decrease of the $\mathrm{SNR}_{\mathrm{test}}$, the ADJSCC model brings a gradually slower performance degradation than the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$= 13 dB or 19 dB). The ADJSCC model is more efficient in the low SNR regime than the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$= 13dB or 19dB). It is worth noting that, despite the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$= 13 dB or 19dB) trained at a specific $\mathrm{SNR}_{\mathrm{train}}$, the performance of the ADJSCC Model still outperforms the BDJSCC model($\mathrm{SNR}_{\mathrm{train}}$= 13 dB or 19 dB) at least 0.3 dB when $\mathrm{SNR}_{\mathrm{train}} = \mathrm{SNR}_{\mathrm{test}}$. The performance of the BJSCC model ($\mathrm{SNR}_{\mathrm{train}}$= 4 dB or 7 dB) is comparable to the performance of ADJSCC when $\mathrm{SNR}_{\mathrm{test}} = \mathrm{SNR}_{\mathrm{train}}$. With the $\mathrm{SNR}_{\mathrm{test}}$ stays away from the $\mathrm{SNR}_{\mathrm{train}}$ of the BJSCC model, the ADJSCC model shows better performance than the BDJSCC model($\mathrm{SNR}_{\mathrm{train}}$= 4dB or 7 dB). The above comparison coincides with the practical condition that the transmitter and the receiver are storage sensitive and the system only has limited storage to store one model of ADJSCC or BDJSCC. No matter which BDJSCC model is selected, the overall optimal performance cannot be achieved when the $\mathrm{SNR}_{\mathrm{test}}$ fluctuates between 0dB and 20dB. While the ADJSCC model can achieve and even better than the overall optimal performance of the BDJSCC models. Fig. 6(b) with $R = 1/6$ reveals similar results to Fig. 6(a). With the increase of the bandwidth ratio, the gap between the ADJSCC model and the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$= 13 dB or 19dB) at high $\mathrm{SNR}_{\mathrm{test}}$ regime almost disappeared. It shows the ADJSCC model has more advantages comparing with the whole BDJSCC model at low bandwidth ratio than that at high bandwidth ratio.

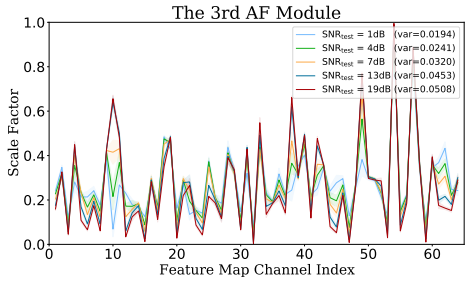
The aforementioned performance evaluation of the ADJSCC model demonstrates that the ADJSCC model can carry a range of SNR by using Attention Mechanism. We would like to understand how the AF Module effects the features in practice. To provide a thoughtful understanding of the AF Module, we study the scale factors generated by the AF Module for the features. Specifically, we transmit the test dataset of CIFAR10 at specific $\mathrm{SNR}_{\mathrm{test}}$ and then compute the mean and standard derivation of the scale factors for each AF Module. We plot the mean and the standard derivation of the scale factors in Fig. 7. The solid line represents the mean values of the scale factors and the translucent area around the solid line with the same color represents the standard derivation of the scale factors. In order to show the scale factors of each GF Model more clearly, we select the scale factors of the first 64 channels in the encoder. From top to bottom, the rows correspond to the order from the first AF Module to the forth AF Module in the encoder. Two observations are inferred from Fig. 7. First, the scale factors across the channels fluctuate more drastically with the increase of the $\mathrm{SNR}_{\mathrm{test}}$. This suggests that the scale factors are more sensitive to high $\mathrm{SNR}_{\mathrm{test}}$ than low $\mathrm{SNR}_{\mathrm{test}}$, which coincides with the intuition. When the $\mathrm{SNR}_{\mathrm{test}}$ is low, the channel noise is severe for each of the features. When the $\mathrm{SNR}_{\mathrm{test}}$ is high, some of the features have a better contribution to the performance than others. The scale factors of good features should be turned up to improve the performance and the scale factors of bad features should be turned down to avoid resource occupation. Second, the difference between the



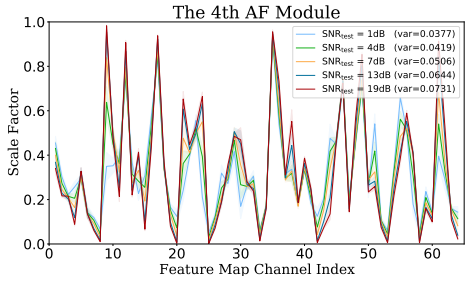
(a)



(b)



(c)



(d)

Fig. 7. The scale factors of the first 64 channels in the encoder of ADJSCC on AWGN channel (R=1/6). (a) the scale factors of the 1st AF Module, (b)the scale factors of the 2nd AF Module, (c) the scale factors of the 3rd AF Module and (d) the scale factors of the 4th AF Module. Each curve of the scale factors is evaluated at a specific SNR on CIFAR10 test dataset. The solid line represents the mean values of the scale factors and the translucent area around the solid line with the same color represents the standard derivation of the scale factors.The var represents the variance of the mean values of the scale factors on specific channel SNR.

curves at different $\mathrm{SNR}_{\mathrm{test}}$ gets smaller as the AF Module gets deeper. This observation shows that the channel noise has a more significant impact on low-level features than high-level features. Low-level features concentrate on the pixel relationship of an image. In contrast, high-level features concentrate more on the linguistic representation implied in an image than the pixel relationship. Compared with the low-level features, high-level features are more robust to the channel noise. Therefore in the fourth AF module shown in Fig. 7(d), the scale factors at different $\mathrm{SNR}_{\mathrm{test}}$ are similar.

We visual the 23rd feature of the first AM Module in the encoder at $\mathrm{SNR}_{\mathrm{test}}$=1 dB shown in Fig. 8(b) and $\mathrm{SNR}_{\mathrm{test}}$=1 dB shown in Fig. 8(c). The 32×32 images of CIFAR10 dataset are too small to be recognized by human eyes. Thus we use a 512×768 image from Kodak dataset instead of the image from CIFAR10 dataset. The detail of Kodak will be introduced in IV-C. Fig. 8(a) shows the original image. The comparison of Fig. 8(b) and Fig. 8(c) shows that when the channel is in good condition, the information about caps should be enhanced to contribute more for the quality of the reconstructed image. It seems reasonable that the detailed information about the caps is easier to be disturbed as the channel gets worse. So when the channel is in bad condition, the detailed information about the caps should be decreased and save the transmit power for more robust information.

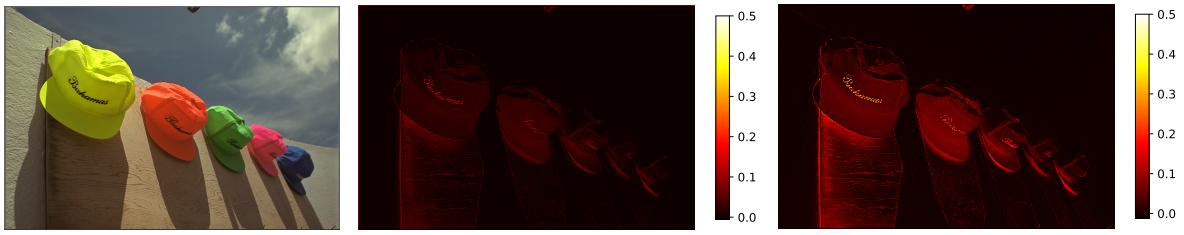
B. The Robustness of ADJSCC

In this part, we study the robustness of the proposed AJSCC scheme to variations in channel conditions. Burst channel [38] is a kind of channel caused by burst interference. Radar Channel [38] due to the sharing spectral resource between mobile communication and radar at 3.5GHz is considered as a typical burst channel [51]. Burst noise is expressed as a random interference modeled as the product of a Bernoulli distribution a high variance Gaussian noise. Following the AWGN channel expression of the Eq. (2), burst channel can be expressed as:

$$\boldsymbol{z_b} = \hat{\boldsymbol{z}} + B(p)\boldsymbol{\omega} = \boldsymbol{z} + \boldsymbol{n} + B(p)\boldsymbol{\omega}, \tag{12}$$

where $B(p)$ is the binomial distribution, p is the probability when $B(p) = 1$ and $\boldsymbol{\omega} \sim \mathbb{CN}(0, \sigma_b^2 \boldsymbol{I})$. We consider a low-power ($\sigma_b = 0.5$), a mid-power($\sigma_b = 2$) and a high-power ($\sigma_b = 5$) burst.

Fig. 9 shows the performance of the ADJSCC model and the BDJSCC model under the burst channel with $R = 1/6$. Fig. 9(a) compares the performance between the ADJSCC model and the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$ = 7dB) at $\mathrm{SNR}_{\mathrm{test}}$ = 7dB under different standard derivation σ_b and the probability of burst noise p. Fig. 9(b) and Fig. 9(c) compare the performance between the ADJSCC model and the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$ = 13dB or 19dB) at $\mathrm{SNR}_{\mathrm{test}}$ = 13dB or 19dB, respectively. The performance of the ADJCC model and the BDJSCC models shown in Fig. 9 degrades as the probability of burst noise p increases, but the ADJSCC model is more robust to the burst noise. The robustness of the ADJSCC model may due to the ADJSCC model traversing the channel noise at different SNR in a batch during the training process, that learns weights for global robustness. However, the BJSCC model trained at a specific $\mathrm{SNR}_{\mathrm{train}}$ learns weights for local robustness, which is effective only around the $\mathrm{SNR}_{\mathrm{train}}$.



(a) (b) (c)

Fig. 8. The original image and the heatmaps of the 23rd recalibrated feature in the first AM Module of the encoder. (a) original image, (b) the heatmap at $\mathrm{SNR}_{\mathrm{test}}$ = 1 dB, (c) the heatmap at $\mathrm{SNR}_{\mathrm{test}}$ = 19 dB.

C. The versatility of ADJSCC

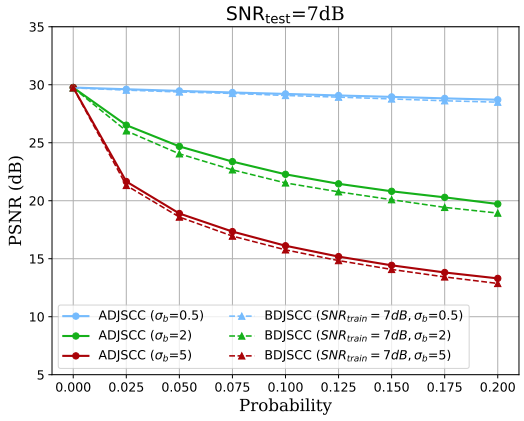
In order to further verify the generality of the proposed ADJSCC method on large-scale dataset, we train the ADJSCC on ImageNet dataset [36]. Until now, ImageNet dataset contains more than 14 million images with more than 21 thousand classes. Various sizes of the images exist in ImageNet. To train our ADJSCC with batch data, we choose the images which size are larger than 128×128 and than randomly crop them to the size of 128×128 to generate the training dataset containing about 5.8 million images that satisfy the condition. Two epochs with batch size of 16 and learning rate of 0.0001 is enough to make the ADJSCC model converge. The model is saved every 200 training batches. Note that ADJSCC is a full convolutional architecture, which can be fed by the inputs with different sizes. The ADJSCC model is evaluated by the Kodak dataset consisting of 24 768×512 images. To average the random of the channel, each images is transmitted 100 times. Fig. 10 shows the comparison of the ADJSCC model and the BDJSCC models on ImageNet dataset. If we consider the BDJSCC models as a whole, the performance of the ADJSCC model approaches to that of the whole BJSCC when $\mathrm{SNR}_{\mathrm{test}} \leq 17\mathrm{dB}$ with neglectable difference. The performance of the whole BJSCC is better than that of B by up to 0.3dB when $\mathrm{SNR}_{\mathrm{test}} > 17\mathrm{dB}$. It is hardly distinguish the difference of two images by human eyes when the PSNR is greater than $35\mathrm{dB}$. The comparison between the performance of ADJSCC and that of the whole DJSCC under $\mathrm{SNR}_{\mathrm{test}} > 17\mathrm{dB}$ seems meaningless. The comparison of Fig. 10 reveals that our proposed ADJSCC method is also applicable to large dataset. We present a visual comparison between the ADJSCC model and the BDJSCC models for the sample image of the Kodak dataset in Fig. 11. Note that even if the SNR is the same, the reconstructed image is different at each time due to the randomness of the noise.

V. Storage Overhead and Computational Complexity

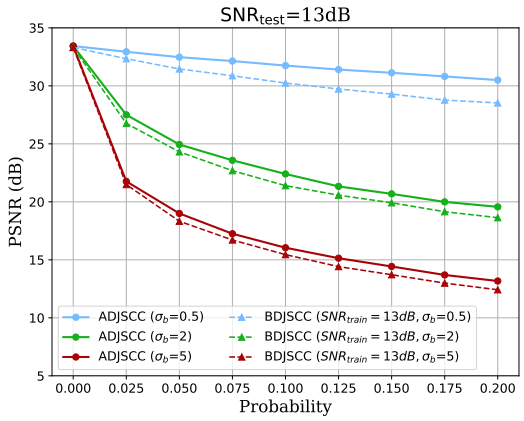
In order to demonstrate the advantages of our proposed ADJSCC, first, we calculate the storage overhead required by the BDJSCC model and the ADJSCC model with R=1/6. There are 10,690,351 total parameters of the BDJSCC model. The type of Each parameter is float32, which requires 4 bytes to save. Hence the storage overhead of the BDJSCC model is 10,690,351×4≈40.78 MB. The ADJSCC model has more parameters than the BDJSCC model because of the existing AF Modules. The amount of the total parameters of the BDJSCC model is 10,758,191. The storage overhead of the BDJSCC model is almost 41.04 MB. The ADJSCC model has 0.6% more parameters than the BDJSCC model, which needs a little additional storage.

If we want to put the BDJSCC method into practice, the strategy should be designed to cover a range of $\mathrm{SNR}_{\mathrm{test}}$. We assume the distribution of $\mathrm{SNR}_{\mathrm{test}}$ follows the uniform distribution of [0, 20] dB. The strategy BDJSCC-1 represents one BJSCC model trained at SNR_{train} = 10 dB that is evaluated by the CIFAR10 dataset with the assumed $\mathrm{SNR}_{\mathrm{test}}$. The strategy BDJSCC-2 represents two BJSCC models trained at SNR_{train} = 5 dB, 15 dB that is evaluated by the CIFAR10 dataset with the assumed $\mathrm{SNR}_{\mathrm{test}}$. The principle of selecting the model is to judge the distance between $\mathrm{SNR}_{\mathrm{train}}$ and $\mathrm{SNR}_{\mathrm{test}}$, and the model with the closest distance is selected as the evaluation model. The strategy BDJSCC-5 contains five models trained at $\mathrm{SNR}_{\mathrm{train}}$ = 2 dB, 6 dB, 10 dB, 14 dB, 18 dB, respectively, and the strategy BDJSCC-10 contains ten models trained at $\mathrm{SNR}_{\mathrm{train}}$ = 1 dB, 3 dB, 5 dB, 7 dB, 9 dB, 11 dB, 13 dB, 15 dB, 17 dB, 19 dB, respectively. The evaluation results are showed in Table I. Although the BDJSCC-1 strategy costs the smallest storage overhead, the PSNR of the BDJSCC-1 is 5.3 dB lower than that of the ADJSCC. The BDJSCC-10 has similar performance with the ADJSCC; however, the storage overhead is 10 times that of the ADJSCC.

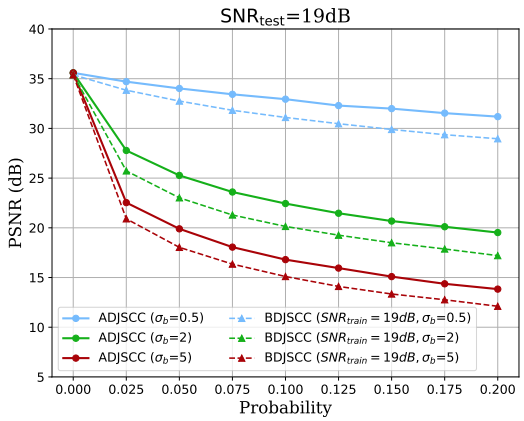
Then we evaluate the computational complexity of the ADJSCC model and the BDJSCC model on the Linux Server we mentioned in Section IV. With a training mini-batch of 128 images from CIFAR10 training dataset, the training time of the ADJSCC model for a batch is almost 114 ms by mean and the training time of the ADJSCC model for a batch is almost 110 ms by mean. The inference experiment based on CIFAR10 test dataset. The inference time for the ADJSCC model takes 53 ms, compared to 49 ms for the BDJSCC model. The training time of the ADJSCC model is 3.6% more than that of the BDJSCC model and the inference time of theADJSCC model is 8.1% more than that of the BDJSCC model. While considering the strategies employed in the practical scenario, the BDJSCC method needs to train multiple models so that it can achieve similar performance to the ADJSCC model. This process spends many times longer than that of training a single ADJSCC model.



(a)



(b)



(c)

Fig. 9. Performance of ADJSCC and BDJSCC under the burst channel (R=1/6). (a) The ADJSCC model and the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$ = 1dB) at $\mathrm{SNR}_{\mathrm{test}}$ = 1dB, (b) The ADJSCC model and the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$ = 7dB) at $\mathrm{SNR}_{\mathrm{test}}$ = 7dB and (c) The ADJSCC model and the BDJSCC model ($\mathrm{SNR}_{\mathrm{train}}$ = 19dB) at $\mathrm{SNR}_{\mathrm{test}}$ = 19dB.

VI. Conclusion

In this work, we have proposed a novel ADJSCC method for channel SNR based on Attention Mechanism, which significantly reduce the storage and the training time of applying the DL based JSCC to the practical wireless communication scenarios.

We propose the ADJSCC method built upon two types of modules, which are the FL modules and the AF modules. The FL module is a general module that can utilize the existing module designed in the existing DJSCC work. The AF module takes the context information to generate the scale factors by using the channel-wise soft attention and recalibrates the channel-wise features. The motivation of our ADJSCC method originates from the resource assignment strategy in the traditional concatenated source channel coders. We evaluate the proposed ADJSCC method on AWGN channel, burst channel and large dataset to demonstrate the adaptability, the robustness and the versatility of the ADJSCC method. Lastly, we compare the storage overhead and computational complexity of the ADJSCC method with that of the BDJSCC method. To achieve the same performance (PSNR), ADJSCC only needs 10.06% storage of the BDJSCC-10 and 10.36% training time of the BDJSCC-10.

Although the ADJSCC methods have shown promising performance, the flexibility of a single network such as the adaptivity and the scalability is worth exploring further in the future. This will play an important part in promoting the DL based JSCC technology to the practical wireless communications. With appropriate modifications, our proposed architecture can be potentially applied to other sources.

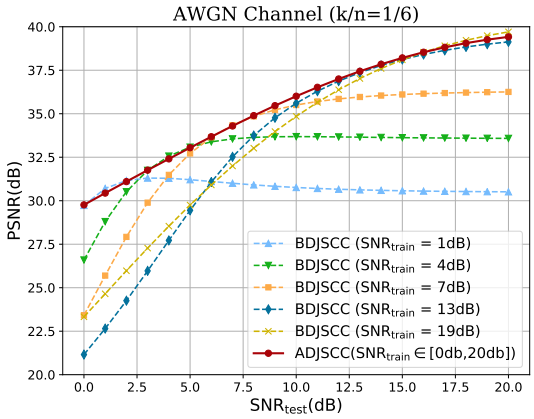


Fig. 10. Performance of the ADJSCC and BDJSCC comparison. The models is trained on ImageNet dataset and tested on Kodak dataset. The bandwidth ratio R is 1/6. Each curve of ADJSCC is trained under a uniform distribution of SNR from 0 dB to 20 dB. Each curve of BDJSCC is trained at a specific SNR.

TABLE I
EVALUATION OF THE ADJSCC MODEL AND BDJSCC MODEL STRATEGIES

Strategy Name	Storage	PSNR
ADJSCC	41.04 MB	**29.831 dB**
BDJSCC-1	**40.78 MB**	24.474 dB
BDJSCC-2	81.56 MB	28.495 dB
BDJSCC-5	203.9 MB	29.694 dB
BDJSCC-10	407.8 MB	29.826 dB

Fig. 11. Visual comparison between the ADJSCC model and the BDJSCC models ($\mathrm{SNR}_{\mathrm{train}} = 7\mathrm{dB}, 13\mathrm{dB}$) for the sample image of the Kodak dataset.

REFERENCES

[1] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.

[2] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[3] R. G. Gallager, *Information theory and reliable communication*. Springer, 1968, vol. 2.

[4] I. Csiszár, "Joint source-channel error exponent," *Problems of Control & Information Theory*, vol. 9, pp. 315–328, 1980.

[5] Y. Zhong, F. Alajaji, and L. L. Campbell, "Joint source–channel coding error exponent for discrete communication systems with markovian memory," *IEEE transactions on information theory*, vol. 53, no. 12, pp. 4457–4472, 2007.

[6] ——, "Error exponents for asymmetric two-user discrete memoryless source-channel systems," in *2007 IEEE International Symposium on Information Theory*. IEEE, 2007, pp. 1736–1740.

[7] B. Belzer, J. D. Villasenor, and B. Girod, "Joint source channel coding of images with trellis coded quantization and convolutional codes," in *Proceedings., International Conference on Image Processing*, vol. 2. IEEE, 1995, pp. 85–88.

[8] S. Heinen and P. Vary, "Transactions papers source-optimized channel coding for digital transmission channels," *IEEE transactions on communications*, vol. 53, no. 4, pp. 592–600, 2005.

[9] A. Vosoughi, P. C. Cosman, and L. B. Milstein, "Joint source-channel coding and unequal error protection for video plus depth," *IEEE Signal Processing Letters*, vol. 22, no. 1, pp. 31–34, 2014.

[10] T. Guionnet and C. Guillemot, "Joint source-channel decoding of quasiarithmetic codes," in *Data Compression Conference, 2004. Proceedings. DCC 2004*. IEEE, 2004, pp. 272–281.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[13] Y. Lee, "Classification of node degree based on deep learning and routing method applied for virtual route assignment," *Ad Hoc Networks*, vol. 58, pp. 70–85, 2017.

[14] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," in *2017 51st Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2017, pp. 1–6.

[15] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep learning for massive mimo csi feedback," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 748–751, 2018.

[16] M. Yang, B. Ai, R. He, C. Huang, J. Li, Z. Ma, L. Chen, X. Li, and Z. Zhong, "Identification of vehicle obstruction scenario based on machine learning in vehicle-to-vehicle communications," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*. IEEE, 2020, pp. 1–5.

[17] G. Toderici, S. M. O'Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, and R. Sukthankar, "Variable rate image compression with recurrent neural networks," *arXiv preprint arXiv:1511.06085*, 2015.

[18] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," *arXiv preprint arXiv:1611.01704*, 2016.

[19] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[20] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 771–10 780.

[21] D. Minnen and S. Singh, "Channel-wise autoregressive entropy models for learned image compression," *arXiv preprint arXiv:2007.08739*, 2020.

[22] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[23] J. Xu, W. Chen, B. Ai, R. He, Y. Li, J. Wang, T. Juhana, and A. Kurniawan, "Performance evaluation of autoencoder for coding and modulation in wireless communications," in *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2019, pp. 1–6.

[24] Y. Jiang, H. Kim, H. Asnani, S. Kannan, S. Oh, and P. Viswanath, "Learn codes: Inventing low-latency codes via recurrent neural networks," *IEEE Journal on Selected Areas in Information Theory*, 2020.

[25] ——, "Turbo autoencoder: Deep learning based channel codes for point-to-point communication channels," in *Advances in Neural Information Processing Systems*, 2019, pp. 2758–2768.

[26] F. A. Aoudia and J. Hoydis, "End-to-end learning of communications systems without a channel model," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2018, pp. 298–303.

[27] H. Ye, G. Y. Li, B.-H. F. Juang, and K. Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional gan," in *2018 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2018, pp. 1–5.

[28] N. Farsad, M. Rao, and A. Goldsmith, "Deep learning for joint source-channel coding of text," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2326–2330.

[29] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 3, pp. 567–579, 2019.

[30] D. B. Kurka and D. Gündüz, "Successive refinement of images with deep joint source-channel coding," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2019, pp. 1–5.

[31] ——, "Deepjscc-f: Deep joint source-channel coding of images with feedback," *IEEE Journal on Selected Areas in Information Theory*, 2020.

[32] K. Choi, K. Tatwawadi, A. Grover, T. Weissman, and S. Ermon, "Neural joint source-channel coding," in *International Conference on Machine Learning*, 2019, pp. 1182–1192.

[33] A. Mnih and D. J. Rezende, "Variational inference for monte carlo objectives," *arXiv preprint arXiv:1602.06725*, 2016.

[34] M. Jankowski, D. Gündüz, and K. Mikolajczyk, "Deep joint source-channel coding for wireless image retrieval," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 5070–5074.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[36] 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," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[37] C.-H. Lee, J.-W. Lin, P.-H. Chen, and Y.-C. Chang, "Deep learning-constructed joint transmission-recognition for internet of things," *IEEE Access*, vol. 7, pp. 76 547–76 561, 2019.

[38] D. Burth Kurka and D. Gündüz, "Joint source-channel coding of images with (not very) deep learning," in *International Zurich Seminar on Information and Communication (IZS 2020). Proceedings*. ETH Zurich, 2020, pp. 90–94.

[39] K. Sayood, H. H. Otu, and N. Demir, "Joint source/channel coding for variable length codes," *IEEE Transactions on Communications*, vol. 48, no. 5, pp. 787–794, 2000.

[40] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[41] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[43] V. Mnih, N. Heess, A. Graves *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, 2014, pp. 2204–2212.

[44] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3156–3164.

[45] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[46] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[47] J. Ballé, V. Laparra, and E. P. Simoncelli, "Density modeling of images using a generalized normalization transformation," *arXiv preprint arXiv:1511.06281*, 2015.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[49] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[50] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[51] H.-A. Safavi-Naeini, C. Ghosh, E. Visotsky, R. Ratasuk, and S. Roy, "Impact and mitigation of narrow-band radar interference in down-link lte," in *2015 IEEE international conference on communications (ICC)*. IEEE, 2015, pp. 2644–2649.