Analog Joint Source-Channel Coding for Gaussian Sources over AWGN Channels with Deep Learning

Ziwei Xuan and Krishna Narayanan
Dept. of Electrical & Computer Eng., Texas A&M University, College Station, 77843, USA
Email: xuan64@tamu.edu, krn@tamu.edu

***Abstract*—We consider the design of neural network based joint source channel coding (JSCC) schemes for transmitting an independent and identically distributed (i.i.d.) Gaussian source over additive white Gaussian noise (AWGN) channels with bandwidth mismatch when the source dimension is small. Unlike existing deep learning based works on this topic, we do not resort to domain expertise to constrain the model; rather, we propose to employ fine tuning techniques to optimize the model. We show that our proposed techniques can provide performance that is comparable to that of the state-of-the-art when the source dimension is small. Furthermore, the proposed model can spontaneously learn encoding functions that are similar to those designed by conventional schemes. Finally, we empirically show that the learned JSCC scheme is robust to mismatch between the assumed and actual channel signal to noise ratios.**

I. Introduction

We consider the classical joint source-channel coding (JSCC) problem of transmitting an independent and identically distributed (i.i.d.) memoryless Gaussian source, $\mathbf{u} \in \mathbb{R}^k$ in $n = \beta k$-uses of a power-constrained additive white Gaussian noise (AWGN) channel. While separately optimizing the source encoder and the channel encoder is optimal for asymptotic lengths and in the absence of any complexity constraints, it is well known that JSCC can outperform separation-based coding schemes for small block lengths, and in the presence of complexity constraints. Further, JSCC schemes can also provide a more graceful degradation in the presence of channel signal-to-noise ratio (CSNR) mismatch. Hence, they may be more robust than separation-based coding schemes [1], [2].

For $\beta = 1$, Goblick proved that uncoded analog transmission (simply scaling the source) is optimal in terms of mean-squared-error (MSE) distortion [3]. However, in many practical applications, it is common that the bandwidth (BW) of source and the BW of the channel are mismatched, i.e., $k \neq n$ (or, $\beta \neq 1$). Past decades have seen many works attempting to design analog joint source-channel coding schemes for mismatched BW transmission. Several works have designed of JSCC schemes in the asymptotic (in length) regime [4]. Our focus in this paper is when k and n are very small (delay-sensitive regime). For small values of k and n, in [5], Hekland *et al.* have designed JSCC schemes using Archimedes' spiral which is based on Shannon-Kotel'nikov mappings. In [1], Hu *et al.* proposed the construction of another group of space-filling curves which are suitable for transmitting both i.i.d. Gaussian and Laplacian sources with mismatched BW over the AWGN channel, and designed the corresponding maximum-likelihood (ML) and minimum-mean-squared-error (MMSE) decoding algorithms. In [6], a necessary condition for the optimality of the encoder and decoder in the presence of BW mismatch was derived, and a steepest gradient descent based algorithm was provided for designing a JSCC scheme for BW compression. JSCC schemes based on chaotic dynamical systems were designed for transmitting uniform sources with BW expansion over noisy channel in [7], [8]. Tent map code and mirrored Baker's code were proposed in [7] and [8], respectively, for the uniform source for the BW expansion case. In [8], Liu *et al.* further extended the results to i.i.d. Gaussian sources.

Motivated by the success of deep learning in image processing, natural language processing and other areas, machine learning based JSCC scheme have been designed in [9], [10], [11], [12]. In [9], convolutional network-based JSCC schemes are designed for transmitting images over AWGN and slow fading channels and it is shown that this scheme is more robust to channel mismatch than advanced separation-based schemes. Most closely related to our work are the works in [10], [11], [12]. In [10], a neural network is constructed with the hyperspherical coordinates of the source taken as the input. The encoder network has two branches which is inspired by the fact that a spiral curve based encoder for $k = 2, n = 1$ splits the source space into two parts. By assuming the general Gaussianity of the received signal and normality of the reconstructed signal in [11], proof of a variational upper bound for the regularized MSE cost was given, and comparable performance with a traditional scheme was attained. In [12], the result was further improved with revised hyperspherical coordinates, and by using a more flexible Lagrangian regularizer.

In this work, we design JSCC schemes for both the BW compression ($\beta < 1$) and BW expansion ($\beta > 1$) for i.i.d. Gaussian sources using deep learning methods. In contrast to [10], [11], [12], we resort to very little prior information about the source or channel, or human expertise in designing the JSCC scheme. Based on recurrent neural networks (RNN), we show that a simple channel autoencoder (AE) structure can achieve performance comparable to the state-of-the-art (SOTA) at low to moderately high CSNR for both BW compression and expansion. In addition, the spontaneously learned encoder transformation resembles those that have been traditionally used. For $k = 1, n = 2$ (also called 1:2) BW expansion task,

978-1-7281-8895-9/20/$31.00

the corresponding transformation shares a high resemblance with a spiral curve, which corresponds to a mapping with a dichotomy in source space. For 2:4 BW expansion, the network is able to learn an encoding function that is similar to that of a chaotic dynamical system. The main contributions of this work are three-fold:

- We propose a simple RNN based channel AE, and rely on fine tuning techniques to achieve performance comparable to the SOTA.
- We provide an interpretation for why the learned network is successful for BW compression for $2:1$ and $3:1$.
- We demonstrate the similarity between the encoding function learned by the neural encoder and that of a chaotic dynamical system based encoding function, and we empirically demonstrate the robustness of our proposed scheme to CSNR mismatch.

II. Problem Formulation

In this section, we focus on the basic settings of the problem we explore, as shown in Fig. 1. We assume that the source to be transmitted is a k-dimensional random variable $\mathbf{u} \in \mathbb{R}^k$, following an i.i.d. Gaussian distribution, i.e., $\mathbf{u} \in \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. The source is encoded by an encoder function $f_\phi(\cdot) : \mathbb{R}^k \to \mathbb{R}^n$, parameterized by ϕ and the transmitted channel codeword is $\mathbf{x} = f_\phi(\mathbf{u})$. The channel codeword satisfies a power constraint given by $\frac{1}{n}\mathbb{E}[||\mathbf{x}||]^2] \leq P_T$, and without loss generality it is assumed $P_T = 1$. The user observes the noisy signal $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where the channel noise is assumed to be additive white Gaussian noise i.e., $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma_\mathbf{n}^2\mathbf{I})$. The observation is processed by the decoder $g_\psi(\cdot) : \mathbb{R}^n \to \mathbb{R}^k$ parameterized by ψ, and the output is denoted as $\hat{\mathbf{u}} = g_\psi(\mathbf{y})$. The goal is to minimize the MSE distortion between $\mathbf{u}$ and $\hat{\mathbf{u}}$ given by

$$D(\mathbf{u}, \hat{\mathbf{u}}) = \frac{1}{k}\mathbb{E}[||\mathbf{u} - \hat{\mathbf{u}}||_2^2].$$

The Shannon lower bound on the distortion as a function of n, k is given by

$$D(\mathbf{u}, \hat{\mathbf{u}}) \geq D_{opt} = \frac{\sigma^2}{(1 + \frac{P_T}{\sigma_\mathbf{n}^2})^{n/k}}.$$

We define the channel signal to noise ratio (CSNR) as $\mathrm{CSNR} := 10\log_{10}\left(\frac{P_T}{\sigma_\mathbf{n}^2}\right)$ (in dB), and signal-to-distortion ratio (SDR) as $\mathrm{SDR} := 10\log_{10}\left(\frac{\sigma^2}{D}\right)$ (in dB).

III. Model and training techniques

In this section, we introduce our deep learning based model and specific training techniques, without resorting to generative models or restriction on the transmitted signals. In later sections, we show that our model can induce comparable performance to the SOTA.

A. Model structure

1) Autoencoder: An Autoencoder (AE) is a popular type of deep learning model for the task of dimension reduction, feature extraction, image restoration, and neural machine translation. The vanilla AE consists of an encoder and a decoder, and the output size of the AE is the same as that of the input. It is straightforward for us to relate the encoder and decoder in an AE as the encoder and decoder in a JSCC scheme. The only difference is that the latter includes a noisy channel between the encoder and decoder. Hence, this model is referred to as channel AE. For the AWGN channel, we can simply set the channel as a non-trainable layer, and thus it will not influence the training of the model. To satisfy the power constraint, we add a non-trainable normalization layer at the end, i.e., we set $\mathbf{x_{L,i}} = \frac{\mathbf{x_{L-1,i}} - \bar{\mathbf{x}}_{\mathbf{L-1,i}}}{\sigma_{x_{L-1,i}}}$, where L, i denote the last layer of the encoder and ith symbol respectively. Then the objective function to be minimized is

$$\frac{1}{k}\mathbb{E}_{\mathbf{u},\mathbf{n}}\left[||\mathbf{u} - g_\psi(f_\phi(\mathbf{u}) + \mathbf{n})||_2^2\right].$$

2) Recurrent Neural Network: We find that long-short-term-memory (LSTM) cells [13] are effective for short block lengths, and also have the potential for being expandable to a longer sequence. In the rest of work, we build stacked bidirectional LSTM for both the encoder and decoder so as to increase the depth of the network. We denote h_e and h_d as the numbers of hidden units for LSTMs in the encoder and decoder. It is then followed by a feed-forward (FF) dense layer to obtain values for each component in the sequence.

Task	source	reshape	RNN output	FF output	reshape
compress	k	$(n, 1/\beta)$	(n, h_e)	$(n, 1)$	n
expand	k	$(k, 1)$	(k, h_e)	(k, β)	n

TABLE I: Size of the output of each part in the neural encoder.

We design RNN-based neural encoders for BW compression and expansion tasks using the parameters shown in Table I if β is an integer. Otherwise, we can first pass the source through an additional simple dense layer to get h_d feature vectors of length k, where h_d is least common multiple of k and n. The neural decoder has a structure that mirrors the encoder.

3) Loss Function: MSE is a common measure for restoration evaluation, and also is widely used as the cost function for regression. Hence we directly set the MSE as the cost function without any regularizer.

4) Network Architecture: For RNN-based channel AE, we use 2 layers of stacked bidirectional LSTM with $h_e = 16$ hidden units in the encoder, and 2 layers of stacked bidirectional LSTM with $h_d = 48$ hidden units in the decoder for small k and n. And we use batch normalization between layers.

B. Training Techniques

In our experiments, we found that the batch size, the learning rate schedule, and weight decay play an important role.

First, according to [14] [15] [16], batch size, jointly with learning rate, determines the 'noise scale' in gradient descent. Within reasonable range, larger batch size together with increased learning rate is beneficial for decreasing the number of necessary parameter updates to achieve the same performance, without deteriorating the generalization capability. Notice that when the channel is noiseless, the reconstruction is not perfect. The suboptimal training process could contribute to the distortion which becomes a limiting factor at high CSNRs.

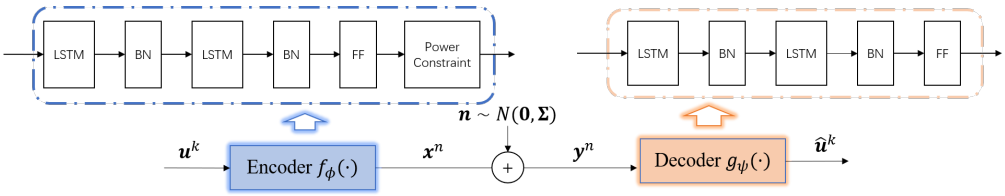


Fig. 1: System model for single transmitter and single receiver over AWGN channel.

It is important to minimize this distortion by choosing large batch sizes. In addition, in our problem, the codeword sequence in a small batch usually has a smaller size than the number of parameters in the neural network, and thus it is inadequate to generalize the model well. Finally, the power constraint in the encoder relies on the statistics of the whole batch of data, which means a more accurate approximation of the second moment can be obtained with a larger batch size. In experiments, we randomly generate mini batches of 5×10^4 samples every update for low to medium CSNRs.

Secondly, warm restart schedule [17] prevents deeper layers from creating training instability [18], by letting the learning rate bounce back to a larger value periodically. Cosine annealing is a learning rate decay schedule, where the learning rate attenuates from a large initial value to any designated small value following a cosine curve. Combining them together, we schedule the learning rate as warm restart with cosine annealing as,

$$\eta_t = \eta_{min}^{(i)} + 0.5(\eta_{max}^{(i)} - \eta_{min}^{(i)})(1 + \cos \pi T_{cur}/T_o^{(i)})$$

for any epoch t satisfying $\sum_{j=0}^{i-1} T_o^{(j)} \leq t < \sum_{j=0}^{i} T_o^{(j)}$. Here, $\eta_{min}^{(i)}$ and $\eta_{max}^{(i)}$ are the minimal and maximal learning rate during the i-th restart cycle. $T_o^{(i)}$ and $T_{cur} = t - \sum_{j=0}^{i-1} T_o^{(j)}$ stand for the length of this restart cycle, and the number of epochs that have lapsed since the recent restart cycle, respectively. Such a schedule has shown to be effective in many tasks, achieving almost better anytime performance [17]. Furthermore, we set the values of $\eta_{max}^{(i)}$ to decay exponentially every restart cycle, and we let $T_o^{(i)}$ grow exponentially every cycle, so that it can explore the minima with fine resolution in the later stages. We set $\eta_{max}^{(0)} = 0.01$ with decay factor 0.9, and $T_o^{(0)} = 4$ with growing factor 1.5 for compression task at CSNR = 20 dB.

Third, weight decay is advantageous to improving generalization during training. In [19], a method is provided to decouple the learning rate and weight decay so as to search these hyperparameters more efficiently. In experiments, we set weight decay $w_o = 5 \times 10^{-4}$ for high CSNRs and a marginal performance improvement can be obtained by this choice.

In the following experiments, we use Adam optimizer [20], and for every 100 updates (as one epoch) we evaluate the validation loss, until the loss converges or up to 2.5×10^4 epochs, depending on which comes ealier. We keep track of the set of weights with best validation performance, and use them for testing.

IV. Experimental Results

A. Bandwidth Compression

In Fig. 2, we plot the SDR of different schemes as a function of CSNR. It can be seen that our model performs as well as the SOTA at low CSNR [6]. Also its performance is similar to that of the spiral-curve based encoder and MMSE decoder in [1], and the results in [12] for mid-to-high CSNRs. The performance of our scheme is slightly worse than other schemes at high CSNRs.

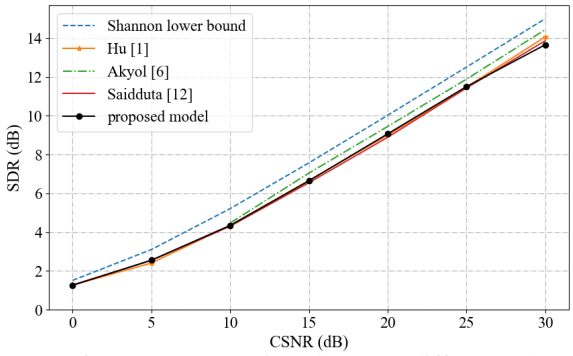


Fig. 2: Performance comparison between different schemes for 2:1 BW compression.

We plot the learned transformation in Fig. 3 for the 2:1 BW compression task. The two source samples u_0 and u_1 are plotted along the X and Y axes, respectively, and the color represents the value of the transmitted scalar x. We can see that even with the simple channel AE structure and without explicitly using any additional information, the model can spontaneously learn an encoding function that is close to the spiral curve of [1]. Specifically, the learned encoder exhibits a dichotomy similar to those in earlier schemes designed by hand [1] [5], iterative learning [6], or deep learning [12]. This demonstrates that even without explicitly using additional prior information about the source or explicitly using information about the distribution of $\mathbf{y}$, our model is capable of learning a good encoding scheme. We can also interpret the neural encoder from the point of view of vector quantization. If we draw a line passing through the origin in the figure in the left panel of Fig. 3, we can see that the neural network partitions the line into several sections and picks a set of close values to represent the source values in each section, similar to what can be expected in a good quantization scheme. It can also be seen that lines close in angle share a high similarity in the quantized values. Accordingly, the learned encoder for 3:1 BW compression can be interpreted from a similar perspective. As

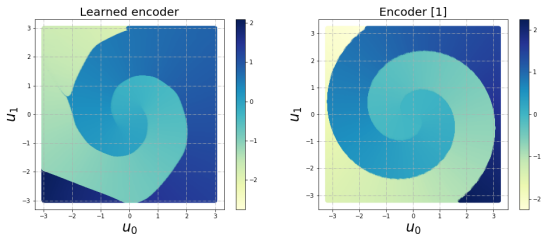


Fig. 3: 2:1 BW compression at CSNR=20 dB. Left: the learned encoder; Right: the encoder in [1].

shown in Fig. 4, if we take planar slices of input space, it can be seen that the network partitions each sliced plane into several areas, and within each area source vectors are mapped to a set of close values. The success of the neural network model lies in correctly learning such a scheme that mimics the performance of a good vector quantizer.

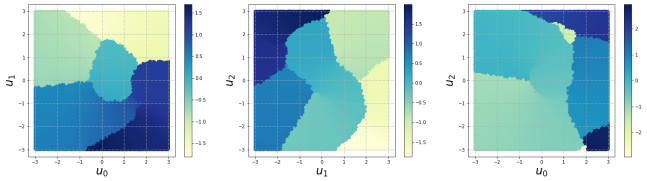


Fig. 4: Compression of $\mathbf{u} \in \mathbb{R}^3$ into a scalar. The left, middle and right figures represent $X-Y$, $Y-Z$, and $X-Z$ plane of the learned encoder at CSNR=20 dB.

B. Bandwidth Expansion

Conventional schemes for JSCC of an i.i.d. Gaussian source with BW expansion include designing encoders based on the inverse transformation of Archimedes' spiral and spiral curve [5], [1]. Based on chaotic dynamical system, mirrored Baker's codes have been designed in [8]. For our proposed model, the encoder structure remains the same as in the BW compression case, but the sizes of the layers are chosen as shown in Table I. In Fig. 5, we compare our model with the above works, as well as with Shannon lower bound. Our model outperforms mirrored Baker's codes in the entire range of CSNRs considered. It also outperforms Archimedes' spiral curve for CSNRs between 0-25 dB, and performs similar to the spiral curve at CSNR of 30 dB. The performance of the proposed scheme is also similar to that in [1] at low and mid CSNR and is within 2 dB SDR at high CSNR. Note that in [8], to apply mirrored Baker's codes to Gaussian sources, inputs are truncated to be bounded to $[-1, 1]$ but the truncation error is not considered. Thus, the mirrored Baker's code appears to slightly outperform the Shannon lower bound in the low CSNR region.

In Fig. 6, we compare the robustness performance of our proposed scheme to the Archimedes' spiral optimized for one specific CSNR in [5]. We train the model at a fixed CSNR, and test it over a wide range of CSNRs. It can be seen that our proposed scheme has better performance over almost the whole CSNR range, and the performance degrades gracefully at low CSNRs in particular.

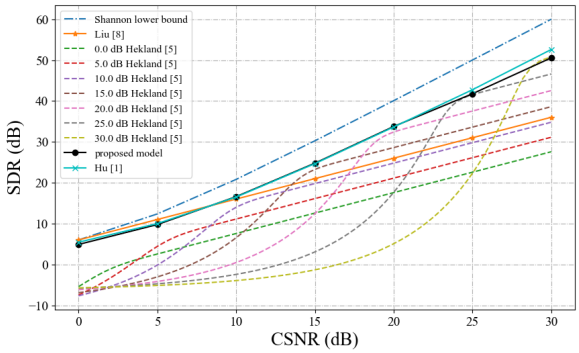


Fig. 5: Performance comparison of different schemes for BW expansion 1:2.

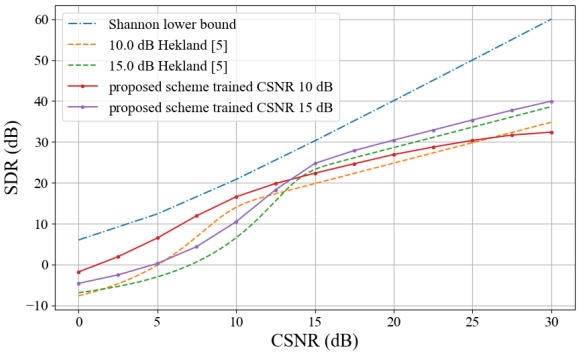
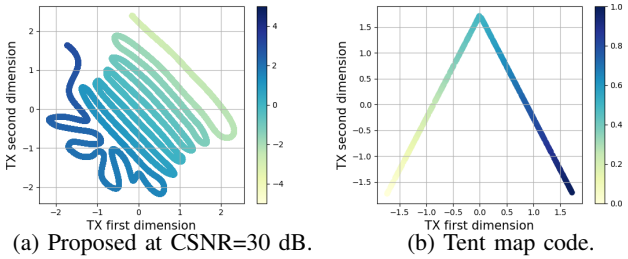


Fig. 6: Robustness comparison for 1:2 BW expansion.

We show the learned transformation trained at CSNR=30 dB in Fig. 7a for $k = 1, n = 2$. It can be seen that the network learns a different space-filling curve than the inverse transformation of the spiral curve. The close SDR performance to the SOTA suggests that there are multiple schemes that perform close to optimality for the BW expansion case.

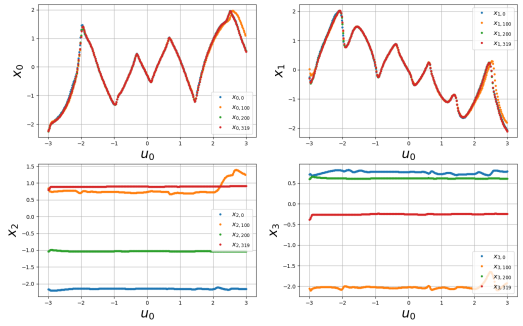
Chaotic dynamical system based codes use a linear transformation of the initial bits to derive later bits in the code sequence. An example of length-2 tent map code is shown in Fig. 7b. The first bit, which is exactly the original message, is mapped to the second bit piece-wise linearly, looking like a single saw. The number of 'saws' of the mapping between the last bit and the initial bit grows exponentially as the code length increases, and thus a tiny change in the original message will be amplified at later code bits. This property facilitates effective error-protection. A similar scheme is learned by the network for BW expansion for larger k. For example, when $k = 2$, the source is denoted as $[u_0, u_1]$ and learned channel codeword is denoted by $[x_0, x_1, x_2, x_3]$. Fig. 8a shows how $[x_0, x_1, x_2, x_3]$ change with u_0 for 4 fixed values of u_1, and Fig. 8b shows how $[x_0, x_1, x_2, x_3]$ change with u_1 for 4 fixed values of u_0. The X-axis represents the value of each bit in the source sequence, and the Y-axis represents the value of each bit in the channel codeword. The top row in Fig. 8a indicates that x_0 and x_1 are mainly determined by u_0 through approximately piece-wise linear transformations. However, in top row in Fig. 8b, the mappings from u_1 to x_0 and x_1 are approximately horizontal

lines, which means u_1 only has slight influence on them. The mappings from the bits in source sequence to x_2 and x_3 have similar characteristics. Furthermore, there is high resemblance between top row in Fig. 8a and bottom row in Fig. 8b. This suggests that the network learns a 2-stage encoding scheme. The 4 bits in the analog channel code are first divided into two sets and the bits in each set are predominantly controlled by the same source bit according to piece-wise linear transformations, and such transformations are similar across the sets.

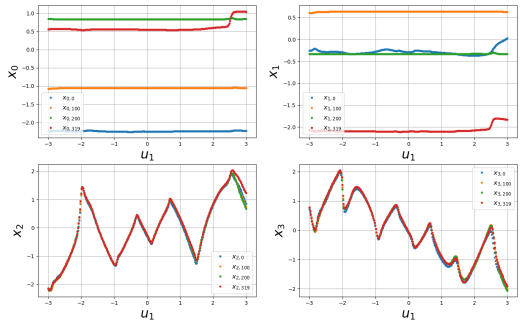


(a) Proposed at CSNR=30 dB. (b) Tent map code.

Fig. 7: 1:2 BW expansion.



(a) Relation between u_0 and x_i.



(b) Relation between u_1 and x_i.

Fig. 8: Relation between source and channel code bits of the learned encoder for 2:4 BW expansion.

V. Conclusion

We introduced a simple channel AE model to design JSCC schemes for transmission of i.i.d. Gaussian sources over AWGN channels with bandwidth mismatch when the source dimension is small. We show that proper fine tuning techniques can improve the effectiveness of the model to achieve results comparable with the SOTA without explicitly using prior information about the source or channel. The learned encoding function resembles that of some conventional SOTA schemes.

VI. Acknowledgement

This work was funded by the National Science Foundation under grant CCF-1718886.

References

[1] Y. Hu, J. Garcia-Frias, and M. Lamarca, "Analog joint source-channel coding using non-linear curves and MMSE decoding," *IEEE Transactions on Communications*, vol. 59, no. 11, pp. 3016–3026, November 2011.

[2] O. Bursalioglu, M. Fresia, G. Caire, and H. V. Poor, "Lossy joint source-channel coding using Raptor codes," *Int. J. Digital Multimedia Broadcasting*, vol. 2008, 06 2008.

[3] T. Goblick, "Theoretical limitations on the transmission of data from analog sources," *IEEE Transactions on Information Theory*, vol. 11, no. 4, pp. 558–567, October 1965.

[4] U. Mittal and N. Phamdo, "Hybrid digital-analog (HDA) joint source-channel codes for broadcasting and robust communications," *IEEE Transactions on Information Theory*, vol. 48, no. 5, pp. 1082–1102, May 2002.

[5] F. Hekland, P. A. Floor, and T. A. Ramstad, "Shannon-Kotel'nikov mappings in joint source-channel coding," *IEEE Transactions on Communications*, vol. 57, no. 1, pp. 94–105, January 2009.

[6] E. Akyol, K. B. Viswanatha, K. Rose, and T. A. Ramstad, "On zero-delay source-channel coding," *IEEE Transactions on Information Theory*, vol. 60, no. 12, pp. 7473–7489, Dec 2014.

[7] B. Chen and G. W. Wornell, "Analog error-correcting codes based on chaotic dynamical systems," *IEEE Transactions on Communications*, vol. 46, no. 7, pp. 881–890, July 1998.

[8] Y. Liu, "Reliable and efficient transmission of signals: Coding design, beamforming optimization and multi-point cooperation," Ph.D. dissertation, Lehigh University, 2016, theses and Dissertations. 2691.

[9] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 4774–4778.

[10] Y. M. Saidutta, A. Abdi, and F. Fekri, "M to 1 joint source-channel coding of Gaussian sources via dichotomy of the input space based on deep learning," in *2019 Data Compression Conference (DCC)*, March 2019, pp. 488–497.

[11] ——, "Joint source-channel coding for Gaussian sources over AWGN channels using variational autoencoders," in *IEEE International Symposium on Information Theory (ISIT)*, July 2019, pp. 1327–1331.

[12] ——, "Joint source-channel coding of gaussian sources over awgn channels via manifold variational autoencoders," in *57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Sep. 2019, pp. 514–520.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] S. L. Smith, P. Kindermans, and Q. V. Le, "Don't decay the learning rate, increase the batch size," *CoRR*, vol. abs/1711.00489, 2017. [Online]. Available: http://arxiv.org/abs/1711.00489

[15] P. Goyal, P. Dollár, R. B. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch SGD: training Imagenet in 1 hour," *CoRR*, vol. abs/1706.02677, 2017. [Online]. Available: http://arxiv.org/abs/1706.02677

[16] S. Jastrzkebski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. J. Storkey, "Three factors influencing minima in SGD," *CoRR*, vol. abs/1711.04623, 2017. [Online]. Available: http://arxiv.org/abs/1711.04623

[17] I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with restarts," *CoRR*, vol. abs/1608.03983, 2016. [Online]. Available: http://arxiv.org/abs/1608.03983

[18] A. Gotmare, N. S. Keskar, C. Xiong, and R. Socher, "A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=r14EOsCqKX

[19] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in Adam," *CoRR*, vol. abs/1711.05101, 2017. [Online]. Available: http://arxiv.org/abs/1711.05101

[20] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.