**Abstract**

A Deep Neural Network (DNN) architecture for distributed image/source compression by a group of distributed cameras/sensors is presented. To this end, we construct a symmetric Encoder-Decoder Convolutional Recurrent Neural Network that outperforms JPEG in terms of rate-distortion performance. We also show that by joint training of distributed encoders and the (joint) decoder for correlated data sources, we significantly outperform separately training the codecs. For 8 distributed cameras/sensors, our distributed system remarkably performs within 1 dB peak signal-to-noise ratio (PSNR) of that of a single codec trained with all data sources. Our method is also shown to be robust to the lack of presence of encoded data from a number of distributed cameras/sensors. By presenting the outputs of a subset of distributed encoders, performance comparable to a distributed system optimized for the corresponding subset of cameras/sensors is achieved. Relationships between our results, and the Slepian-Wolf Theorem in distributed source coding of i.i.d. sources are also elucidated.

**Introduction**

Many researches show that DNN can achieve comparable results as traditional image compression techniques \cite{toderici2015variable,balle2016end,gregor2016towards,toderici2017full,theis2017lossy,johnston2017improved,liu2018cnn}. Most of these methods adopt basic autoencoder network and quantize the bottleneck representation. Many non-recurrent methods quantize the codes into 8-bit integers and minimize a L1 penalized loss to sparsify the codes. These models tend to rely on entropy codec to compress the sparsified codes. For different compression rates, we also need to train multiple models with different regularization parameter.

Among those different architectures, recurrent model is known to be the first model that outperforms traditional codecs \cite{toderici2015variable}. The model uses a recurrent autoencoder to encodes the residual between the original input and the reconstruction output. At each iteration, new binary coded information is extracted from the residual where the context from previous iterations is stored in the hidden state of the recurrent model. The compression rate, in this case, is varying according to the number of iterations of a single recurrent model. Since our work concentrates on DNN based DSC, we develop from the recurrent model introduced by \cite{johnston2017improved} and study a distributed coding architecture.

In information theory, DSC is an important problem regarding the compression of multiple correlated data sources. The Slepian-Wolf theorem shows that lossless coding of two or more correlated data sources with separate encoders and a joint decoder can compress data as efficiently as if coding them together \cite{slepian1973noiseless,cover1975proof}. The extension to lossy compression was proposed as Wyner-Ziv theorem \cite{wyner1976rate}. Although these theorems were published in 1970s, it was after about 30 years that practical applications like Distributed Source Coding Using Syndromes (DISCUS) emerged \cite{pradhan2003distributed}. One of the main advantages of DSC is that the computation complexity of the encoder is shifted to the decoder. Typically, low complexity encoders can be used in multi-view video coding and sensor networks \cite{girod2005distributed,xiong2004distributed}. Our DNN architecture consists of distributed encoders and a joint decoder. Through numerical experiments, we study the complexity of encoders and the number of distributed data sources. We show that distributed encoders can perform as well as a single encoder trained with all data sources together.

We review previous related work in Section 2 and describe our proposed method in details in Section 3. We describe our architecture for general image compression in Section 3.1. Then we elaborate the distributed framework in Section 3.2. Experimental results are shown in Section 4, followed by conclusions in Section 5.

**Related Work**

There exist many traditional codecs for lossy image compression. Although developed 30 years ago, JPEG \cite{wallace1992jpeg} is still the mostly widely used image compression method. Several extensions to JPEG like JPEG2000 \cite{skodras2001jpeg}, WebP \cite{google2010webp} and BPG \cite{bellard2014bpg}has been developed. Most of these traditional codecs rely on a quantization matrix applied on the coefficients of DCT or wavelet transform.

Autoencoders are widely used as a base module for neural network-based image compression. Bottleneck representations are quantized into 8-bit integers or binaries. Ball\'e et al. \yrcite{balle2016end} replaces non-differentiable quantization step with a continuous relaxation by adding uniform noise. Toderici \yrcite{toderici2015variable}, on the other hand, uses stochastic binarization during training and the gradient is replaced with 1. Non-recurrent autoencoders tend to rely on L1 penalty to sparsify the 8-bit integer codes while recurrent models add binary codes at each iteration. The compression rate of non-recurrent models is not scalable and their performance heavily rely on the sparsity that entropy codec can take advantage of. Recurrent model, on the other hand, has scalable compression rate but does not generate different bit rates according to the number of iterations across different images.

Basic information-theoretic results on DSC have been shown since 1970s. Slepian-Wolf \yrcite{slepian1973noiseless} and Wyner–Ziv \yrcite{wyner1976rate} theorem shows that two or more correlated data sources encoded separately and decoded jointly can perform better than decoded independently. The surprising result is that as long as the codes are jointly decoded, there can be no coding efficiency loss no matter the codes are separately encoded or jointly encoded. It is known that channel coding algorithms like Low-density Parity Check (LDPC) can be used with DSC \cite{asvadi2013joint}. Some reasearches have also shown the applicability of DSC on still images \cite{dikici2005distributed}. In practical applications, low complexity video encoding benefits from the DSC framework which can shift the complexity of encoder to decoder \cite{puri2002prism,aaron2002wyner}. Scalable Video Coding can also be incorporated with DSC \cite{xu2006layered}. In fact, the recurrent model we propose is naturally scalable. Codes can be decoded at more than one compression quality levels. This is very attractive in video streaming applications. In multi-view video coding and sensor networks, DSC allows efficient coding of correlated sources which are not co-located \cite{guillemot2007distributed,gehrig2008distributed}.

**Methods**

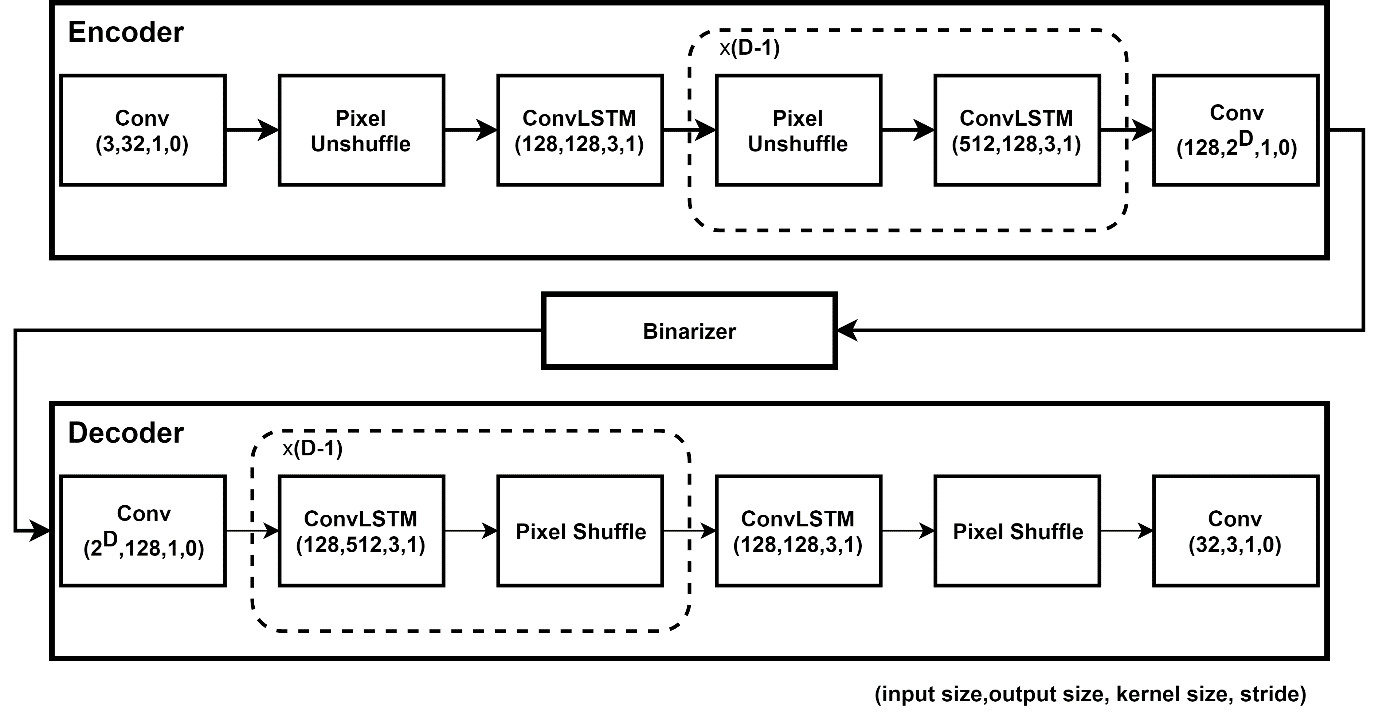
*Network Architecture*

Figure 1 shows a single iteration of the general compression architecture we propose to use in our distributed framework. Our compression network consists of an encoder, a quantizer, and a decoder. The input images are first encoded and transformed into [-1,1]. Binary codes are quantized from transformed bottleneck representations that can be stored or transmitted to the decoder. The decoder then outputs a reconstructed image based on the received binary codes. Finally, we compute the difference between the original input image and reconstructed output image. The procedure is repeated with the computed residual. The ultimate reconstructed images are the sum of output reconstructions across multiple iterations. The compression quality levels are scalable, because by decoding more iterations of codes, the compression quality is also increasing.

The first and last layers of the encoder and decoder are feed-forward convolutional neural network (CNN) with tanh activations. The recurrent layers are all convolutional Long short-term memory (LSTM) networks. We perform down-sampling and up-sampling with pixel (un)shuffle operation. The original pixel shuffle operation is a non-parametric up-sampling operation proposed to use in super-resolution \cite{shi2016real}. Although the method is widely used for up-sampling, it is invertible and can be used for down-sampling. Each down-sampling and up-sampling operation will reduce and increase the shape of feature from (H,W) to (H/2,W/2) and (H\*2,W\*2). The channel size will increase and reduce from C to 4C and C/4 respectively. During training, the quantizer stochastically rounds [-1,1] into {-1,+1}. The stochastic rounding operation is used to approximate the quantization error. The gradient in the backward pass is replaced with the derivative with respect to the expectation. We do not enforce any stochastic operation during testing \cite{theis2017lossy}.

Some formulation of convolutional LSTM and quantizer here.

To ensure the constant nominal compression rate (pre-entropy coding), the depth of model and the size of codes is controlled by D. For example, with D = 3, a 32x32 RGB image will generate (8,4,4) binary codes for a single iteration. Thus, we will add up 0.125 bit per pixel (bpp) for each iteration. We can also adjust the size of codes if more codes are desirable at each iteration

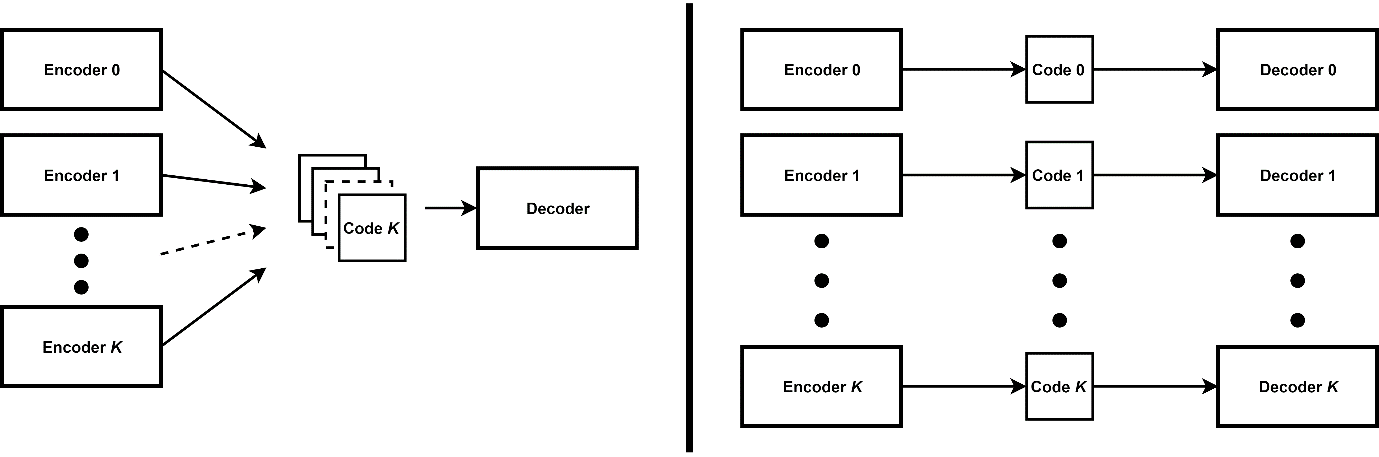


*Distributed Framework*

Figure 2 shows DSC and normal coding framework. In normal coding framework, each data source is encoded separately and decoded independently. In DSC, K encoders encodes correlated sources into K groups of codes. All codes are then transmitted to a single decoder, concatenated batch-wisely and jointly decoded by a single decoder.

In our neural-network based DSC, the whole encoding and decoding procedure are trained jointly. Correlated images are encoded and quantized into binary codes. K groups of codes are concatenated batch-wisely and decoded by a single decoder. We then compute the residual error and begin the next iteration. At each iteration, we minimize the loss between the sum of reconstructed outputs and the original images. Finally, the parameters of the model are updated through backward propagation. We find that the resulting distributed model can perform as well as encoding all sources by one encoder together. However, if we encode and decode each group of data source separately, the performance becomes significantly worse.

Some formulation of neural network DSC here



**Experiments**

1. General compression model compares with previous works on MNIST (ideal trained on ImageNet and test on Kodak), ignore special loss and training hack proposed by those works, only focus on model architecture.
2. Plot of DSC with different number of sources (2,4,8, 16?), including train on 8 and test on (2,4,8)
3. Plot of DSC with smaller encoder (low complexity encoding motivation of DSC)
4. Plot of DSC trained with sources from each classes, not the subset of dataset.

**Conclusion**