CS230

Smooth compression and reconstruction of human portrait image using convolutional auto-encoder

Sheng Tan
shengtan@stanford.edu

Abstract

Generative Adversarial Networks(GANs) have been delivering outstanding results in reconstructing high resolution images using lower resolution image inputs. Currently well studied deep neural networks structures for image super resolution, like SRGAN or SRResnet, requires the input data to follow image data structures and could be rather difficult to train. This project explores the possibility of using a convolutional auto-encoder to encode higher resolution images to data stream and reconstruct the image. The reconstructed images can not out perform the results produced by GANs, but the structure provides a lossy compression algorithm to human portrait images.

1 Introduction

Data transferring has gradually become a fundamental part of human life since the development of internet, yet even today, the limitation of bandwidth and network quality can still badly influence the performance of applications such as audio call, image uploading, and video streaming. For video streaming services, an common solution when facing poor network connections is to drop the resolution, which is not very desired by the users. Data compression, a fundamental and well-studied problem in engineering, have been producing algorithms that would allow robust data transferring without losing information, yet it relies heavily on knowledge of the probabilistic structure of the data and it is limited by the channel capacity. In this project, we limit the scope of the study to human face images, which are images commonly transferred for video conference application, and built a convolutional auto-encoder to encode human portrait images of 192 by 192 pixels and reconstruct them back to the same size. We compare the result to lower resolution images and studied the effect of different filter sizes used in layers.

2 Related work

The concept of training neural networks as auto-encoder was brought up by Geoffrey Hinton and Ruslan Salakhutdinov at as early as 2006[1]. In the paper they explored the possibility of training deep neural networks that flatten input data as auto-encoders on gray scale images of random curves, MNIST and low resolution human face images. Convolutional Neural Networks as auto-encoders were soon studied by Li Xu[7], Jonathan Masci[8] and Bao[9], and they were commonly used as feature extractors for GANs recently to further improve the performance of GANs.[10]

However, besides it's common application as feature extractors, convolutional atuo-encoders could be very powerful on its own. In the papers published Johannes Ballé, Valero Laparra, and Eero P. Simoncelli, a convolutional auto-encoder is trained over Kodak image set with unique activation functions generalized divisive normalization(GDN) function and the approximate of its inverse function. A distinctive cost function is also proposed to consider both the information lost during

the encoding process and the entropy of the encoded information. Such a network is the state-of-art auto-encoder for image compression as it has evidently outperformed JPEG-2000, the ISO standard image compression algorithm(ISO/IEC 15444) This project is motivated by their paper to train an auto encoder with simpler cost function and commoner activation functions, and specializes in the training of human portrait images. We adapted the general structure of the original neural network in terms of number of layers and filter sizes, and customized it for our use case.

3 Dataset and Features

For dataset, we use CelebA dataset which contains 202,599 RGB human portrait images. We used 180,000 images in training set, 10,000 for dev set and 10,000 for test set. The images are all reshaped to the same resolution of 192x192. The image data is normalized from [0,255] integers to [0,1] floats.

Figure 1: Sample Training images, without any processing, 178x128

In the state-of-art end-to-end image compression paper[2], the Kodak image set was used, which has higher resolution 768x512 images. The image set contains a variety of images including several human portrait images. However, the original paper mostly studied gray scale images, whereas this project focus on the encoding and decoding of RGB images.

4 Methods

Our model follows the guideline of the end-to-end compression paper[2] in general. We used a more common activation function ReLu in our neural network instead of GDN and IGDN. We use a different filter size for each layer and we are using a simpler cost function. Different filter size choices are explored and learning rate is tuned for faster learning process.

4.1 Loss function and activation function

In the original paper, a rather complicated cost function is proposed. The cost function is a measurement of entropy and the expectations are approximated using the average value over samples.

$$L[g_a, g_s, P_q] = -E[log_2 P_q] + \lambda E[d(z, \tilde{z})] \tag{1}$$

In this project, we will measure the loss using mean squared error

$$L = \frac{1}{n_C^2 n_W^2 n_H^2} \sum_{c=1}^{n_C} \sum_{w=1}^{n_W} \sum_{h=1}^{n_H} (X_{c,w,h} - Out_{c,w,h})^2 \tag{2}$$

The original paper also proposed two advanced activation functions to be used in the auto encoders: GDN for encoding phase and IGDN for the decoding layers.[2] In this project, we simplify the activation functions to ReLu. This is a legitimate simplification from a mathematical perspective. IGDN is engineered to be a approximate of inverse function of GDN, which allows it to gradually "decode" the original image. The ReLu function is also an approximation of it's own inverse function, as it maps the domain where input is negative to zero, and maps the non-negative region to itself.

4.2 Auto-encoder structure

The final structure of our auto-encoder is shown in figure 3

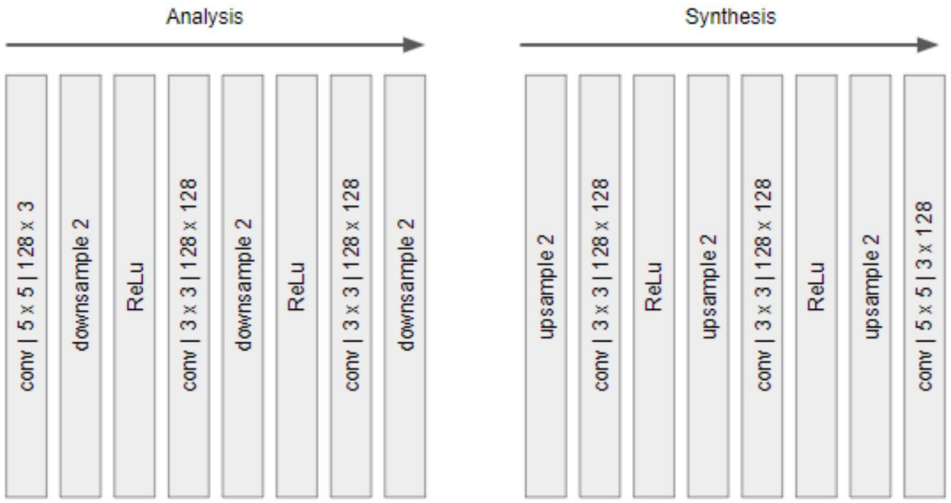

Figure 2: Convolutional Auto Encoder, filter support (x X y), number of channels (output X input)

Compare to the original paper, in which 9 by 9 and 5 by 5 filters are used, we eventually choose to use 5 by 5 and 3 by 3 filters to take into account of the relatively lower resolution training images used in our project.The inner most layer, which used to be an extra layer of GDN and IGDN for Analysis layer and Synthesis layer is omitted in this case, as passing the tensors through two ReLU functions consecutively have no real impact on the size of compressed information or make any mathematical sense other than setting all negative values to zero from the previous layer.

5 Experiments/Results/Discussion

5.1 Loss function and activation function

Hyperparameters of the network was tuned throughout the project. Initially, we set the learning rate to be 0.01 referring to an open source implementation of auto-encoder for MNIST handwritten digits, and soon we realized that for this project, having such a high learning rate would make the network quickly converge to a point where only black images, or all zero outputs are produced. During further tuning, we found that learning rates 1e-3, 1e-4, 1e-5 all give resonable results, but 0.0001 out performs the other two values as it decreases the cost after the fist epoch over 1,000 samples by 4e-4. Mini-batch size is set to 50 only because it is the largest batch size that could fit into the memory of our local machine with 16Gb memory and a GPU of 1070Ti(ec2 allows bigger batch size but we keep the code consistent).

Mean squared error is used to evaluate the performance of the neural network as well as comparing to bench marks, which are the original image, bicubic-interpolation after 2x downsample, 4x downsample, and 8x downsample.

5.2 Model Evaluation

The result of the auto-encoder is displayed on Figure 3 and Figure 4. After 3 epoch of training, the MSE of the original training image and the decoded image reached 2.1e-4, and MSE of the test image taken by hand is 3.1e-4. The MSE is higher than the image reconstructed by only performing a 2x downsampling(1.9e-4), but is way superior than that of 4x downsampling(4.5e-4) and 8x downsampling(9.3e-4). Is is though, hard to argue how much useful information that human can perceive is preserved during the encoding and decoding process due to the limitation of MSE as a loss function.

Figure 3: Training image id 30,000, from left to right: 192x192 original, 2x downsampling, 4x downsampling, 8x downsampling, reconstructed by Auto-encoder

Figure 4: Selfie of author taken by phone, from left to right: 192x192 original, 2x downsampling, 4x downsampling, 8x downsampling, reconstructed by Auto-encoder

5.3 Unexpected Results

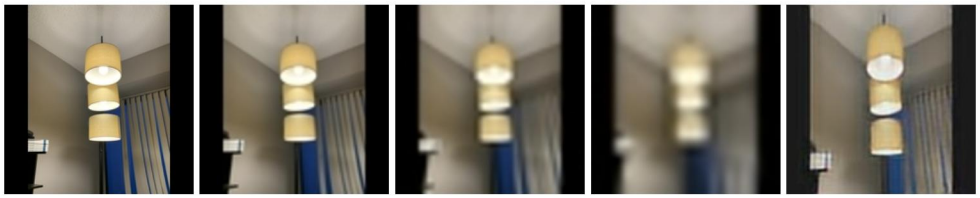

Figure 5: lamp image taken by phone, from left to right: 192x192 original, 2x downsampling, 4x downsampling, 8x downsampling, reconstructed by Auto-encoder

As a part of the experiment, we tried to pass in a non-human face image, which is a picture of lamp taken by our team member using their phone. Surprisingly, the auto-encoder, even only have been trained with images of human portraits, still delivers outstanding result as shown in Figure 5.The MSE of the original image and the reconstructed image is 3.5e-4, which is almost as good as the performance on test human face image.

We might be able to conclude that the filters trained in the auto-encoder have been capturing more common features rather than features specific to human faces. The result is certainly intriguing and could be further studied.

6 Conclusion/Future Work

We have built an auto-encoder that performs lossy compression of images while reconstructing them with smooth edges and capturing main features of the key object - human faces. It is unexpected and unexplained how the auto-encoder is also able to reconstruct non-human portrait images with such good performance yet, and it would certainly be interesting to learn why. Deeper networks and higher resolution training data set could be applied to this topic for future studies, and the potential of this auto-encoder as a feature extractor for GANs is yet to be explored.

7 Contributions

We really appreciate the help offered by TA Patrick Cho and would like to say thank you to all the teaching staff of CS230 for such a great quarter.

References

[1] Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks." science 313.5786 (2006): 504-507.

[2] Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. "End-to-end optimized image compression." arXiv preprint arXiv:1611.01704 (2016).

[3] Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. "End-to-end optimization of nonlinear transform codes for perceptual quality." 2016 Picture Coding Symposium (PCS). IEEE, 2016.

[4] Toderici, George, et al. "Variable rate image compression with recurrent neural networks." arXiv preprint arXiv:1511.06085 (2015).

[5] Chen, Min, et al. "Deep features learning for medical image analysis with convolutional autoencoder neural network." IEEE Transactions on Big Data (2017).

[6] Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. "End-to-end optimization of nonlinear transform codes for perceptual quality." 2016 Picture Coding Symposium (PCS). IEEE, 2016.

[7] Xu, Li, et al. "Deep convolutional neural network for image deconvolution." Advances in Neural Information Processing Systems. 2014.

[8] Masci, Jonathan, et al. "Stacked convolutional auto-encoders for hierarchical feature extraction." International Conference on Artificial Neural Networks. Springer, Berlin, Heidelberg, 2011.

[9] Bao, Jianmin, et al. "CVAE-GAN: fine-grained image generation through asymmetric training." Proceedings of the IEEE International Conference on Computer Vision. 2017.

[10] Klingler, Severin, et al. "Efficient Feature Embeddings for Student Classification with Variational Auto-encoders." EDM. 2017.