Final Report for Super Resolution

Jianhua Huo, Shuo Zhuang, Xiaoyu Zhou

Abstract

One of the most challenging problems in super-resolution is finding the missing details in the upscaled image. We use convolutional neural networks to hallucinate details to a low-resolution image. We will be mainly implementing the algorithm based on the proposed paper "Image Super-Resolution Using Deep Convolutional Networks" by Chao Dong's team. We can train the neural network with a dataset that collects high-resolution images and downscaling them. The neural network can help us to upscale a low-resolution image to a higher resolution image by mapping between missing pixels in the input image and high-resolution images. Instead of using only MSE, we use a newly defined loss function to combine MSE with Sobel edge detection to preserve the edge without adding too much computation.

1 Introduction and Literature Review

Single image super-resolution is a process to recover a high-resolution image from a given low-resolution image. There are some traditional methods for super-resolution, such as weighted average neighborhoods and bilinear/bicubic interpolation methods. Patti and Altunbasak have also tried to optimize the traditional method by using higher interpolation and adding regularization in the inversion [3]. However, their performance is always poor since the missing data cannot be recovered by further processing.

With the rapid development of deep learning in the past decade, super-resolution models based on deep learning have been actively explored and the reconstructed images are significantly improved. Most of those models choose to learn mapping functions from pairs of low-resolution and high-resolution example images, and then implement the mapping function on the target low-resolution image [2].

In Chao Dong and his partners' work, they follow the algorithm of the sparse-coding-based method, which is one of the representative external example-based super-resolution methods. Instead, they use the deep convolutional neural network to achieve super-resolution and call the proposed model Super-Resolution Convolutional Neural Network (SRCNN) [2]. The convolutional neural network can directly learn an end-to-end mapping between

low-resolution and high-resolution images. Chao Dong et al. firstly convolve the image by n1 filters to extract patches from the low-resolution images. Then they transform the n1-dimensional vectors into n2-dimensional vectors through a nonlinear operation. In the last step, they convolve the n2 feature maps of high-resolution images into the final full image.

In the part of error evaluation, most of the work includes Chao Dong et al., use mean square error (MSE) as a loss function to evaluate the error between the true image and the reconstructed image produced by the algorithm. However, this may cause structural information about the content of the image to be ignored. Thus, the edges of the image may be blurred. Another super-resolution approach, Generative Adversarial Nets (GAN) (like SRGAN), adds texture details to the high-resolution output from the algorithm, aims to use the adversarial loss to enhance images [1].

Moreover, In Pandey and his partners' work, they firstly calculate the proposed mean squared Canny error (MSCE), which is used to evaluate the error between applying Canny edge detector on the reconstructed image and on the true high-resolution image [4]. And then they combine MSE and MSCE together through a certain weight.

In our project, we will mainly implement Chao Dong and his partners' algorithm to pre-process the data and reconstruct the low-resolution images. Then for error evaluation, we choose to use a loss function that is similar to Pandey and his partners' loss function, but we use Sobel edge detection instead of Canny edge detection. In our loss function, we calculate the MSE of the true image and the predicted image after detecting the Sobel edge to optimize the problem of blurred edges.

Our project is based on SRCNN and optimizes the loss function. We do some experiments on different parameters to find a better algorithm, such as the filter number in the convolution layers. By applying the algorithm with appropriate parameters, we can perform image super-resolution faster and more accurately.

2 Methodology, Results, and Experiments

2.1 Dataset

The models are trained and tested on a DIV2K dataset. Same as SRCNN, we randomly crop each image in the first T91 dataset into 32x32-pixel sub-images as our training dataset. And we use Set5 and Set14 as the testing sets to test our network.

2.2 Architecture

We are using three convolutional layers like the SRCNN model: patch

extraction and representation, non-linear mapping, and reconstruction, then apply the ReLU activation to the first two convolutional layers only [2].

2.2.1 Pre-process data:

First down-scaling the high-resolution (HR) image X (Ground truth) by a factor like 3 or 4, then up-scaling it by the same factor. Then, we will get a new low-resolution (LR) image Y, which has the same size as the true HR image X. For up-scaling and down-scaling processes, we use the OpenCV interpolation methods. There are many choices like linear interpolation, bicubic interpolation. We chose to use bicubic interpolation for this project, which has a better performance than linear interpolation.

Then, we want to find a mapping function F, such that F(Y) approaches to X.

2.2.2 Patch extraction and representation:

Use the LR image Y from the previous step as input, extracts (overlapping) patches from it and represents each patch as an n1 sized feature vector. It performs standard convolution with ReLU to get F1(Y), which uses max(0, W1*Y+B1) as the first layer. [2]

The size of the weight W1 is f1*f1*n1, which means applying n1 convolutions onto the image, each convolution with kernel size f1*f1, and size of bias B1 is n1 for each channel.

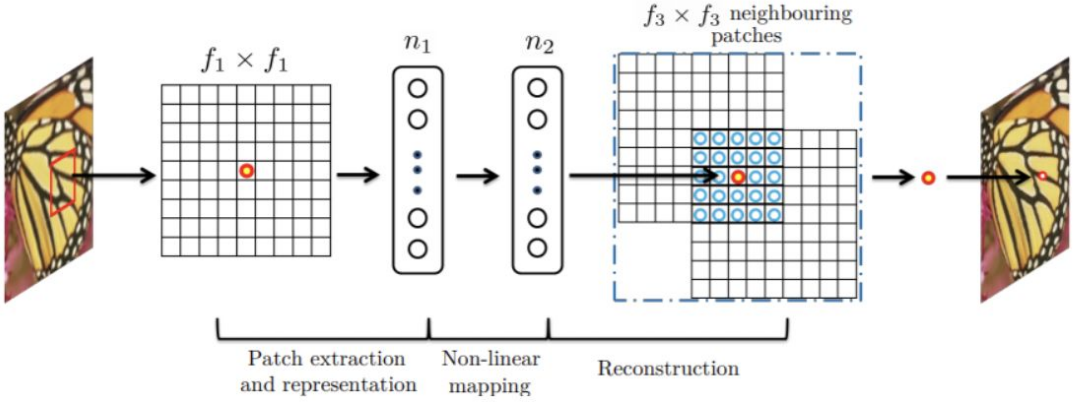
2.2.3 Non-linear mapping:

The non-linear mapping F2(F1(Y)) can be expressed as max(0, W2*F1(Y)+B2), the size of W2 is n1*n2, and the size of B2 is n2. We mapped the previous vector onto another feature vector of size n2 through this function. After mapping, we got an HR patch vector representation. [2]

This one-to-one convolution introduces more non-linearity to improve accuracy.

2.2.4 Reconstruction:

The last convolutional layer is used to produce the final HR image. In this layer, we reconstruct the HR image from aggregating the HR patches of size n2 in the previous step by the function: $F(Y) = W3 * F2(Y) + B3$, where W3 is c filters of size $n2 \times f3 \times f3$, and B3 is a c-dimensional vector (c is the number of channels) [2].



Here is the process of the three layers convolution.[2]

2.3 Loss Function

We need to refine our parameters {W1, W2, W3, B1, B2, B3}. We use a loss function to estimate how our algorithm performs. In addition to the MSE used in SRCNN, we define a loss function that is a combination of MSE and the edge preserve loss function, which is called mean square Sobel error (MSSE).[4]

$$Loss = \underbrace{\mu \times \frac{1}{N}\sum_{j=1}^{N} \| \Theta_\lambda(L_j) - H_j \|_F}_{MSE\ Loss\ (l_{mse})} + \underbrace{(1-\mu) \times \frac{1}{N}\sum_{j=1}^{N} \| C(\Theta_\lambda(L_j)) - C(H_j) \|_F}_{Edge\ preserving\ loss\ (l_{edge})}$$

Our loss function, combined MSE and MSCE together. [4]

Since SRCNN uses only MSE which is the pixel wised loss function, it may reduce the high-frequency content and blur the edges in the image.

We use the Sobel operator as our edge preserving loss which provides reliable edges. And compared to other models like GAN, it simplifies the computation.

2.4 Quality Metrics

SR reconstruction quality is usually measured in terms of PSNR (Peak Signal-to-Noise Ratio). Low MSE yields high PSNR.

2.5 Experiments

We did some experiments on the different parameters on both SRCNN algorithm and the loss function. We use the training dataset as described to compute our output F(Y) in terms of different parameters and compare it with the ground truth X over the set. Then, calculate PSNR to see which parameter fits better.

For the filter number in the convolution, the more filters we use in the convolution, the better results will be. However, too many filters will influence the speed of the algorithm. Thus, after trying a few experiments and learning from Dong and his partners, we set f1 = 9, f3 = 5, n1 = 64 and n2 = 32 in our

main evaluations. Since n2 < n1, it can be considered as a PCA model. And we use c=1 for convenience.

During the training process, we first apply the Canny edge detector to the reconstructed output and separately on the corresponding ground truth image to compute the proposed mean square Canny error (MSCE). However, in order to use the Canny operator, we need to convert the Y channel in Ycbcr into RGB. Our implementation involves a lot of computation which makes the training process much slower.

Instead, we use Sobel operator as edge preserving and apply on both the reconstructed output and the ground truth to calculate the loss. The result may be a little bit worse than the Canny operator, but it speeds up the training process.

2.6 Results

We performed a few experiments to determine the proper weight of MSCE. The models consistently give better results when μ is around 0.84+/-0.02. So, we take μ=0.85 for convenience [4].

$$\hat{\mu} = \underset{\mu}{\operatorname{argmin}}\{\mu \times l_{mse} + (1-\mu) \times l_{edge}\}$$

The function to find the best μ [4]

We have performed a few experiments comparing SRCNN (with original loss function) with our proposed mean square Sobel error loss function. And we found our MSSE loss leads to better results and is consistent across different scaling factors.

This is the perceptual quality comparison. The image is chosen from set14. After pre-processing, the leftmost image is the LR input, and it is fed into the SRCNN model and our modified MSCE model. The middle one is produced by the SRCNN model. More details are added to the original LR image. But the perceptual quality of the arm is not that good. The rightmost image is our model

with a modified loss function.

3 Conclusion

In this project, we mainly implemented the three convolutional layers by using the Keras framework based on the SRCNN algorithm, trying to reconstruct the low-resolution image into high-resolution. By using the idea of convolutional layers, we can do a better and faster super-resolution solution. But in SRCNN, there is a problem that the loss function ignores some edge information. To solve this problem, we improved the loss function by adding more edge details. Thus, instead of using only MSE, we also combined MSE with the Sobel edge detector to capture more edge features, so that the edges in the reconstructed high-resolution image are no longer blurred.

4 Authors' Contribution

Our team consists of three members: Shuo Zhuang, Xiaoyu Zhou, and Jianhua Hu. Jianhua is responsible for pre-processing image data and decides which framework we're going to work on. Xiaoyu is responsible for building the learning models. She's trying a lot of experiments to find out the proper parameters to get better test results. Shuo's job is to figure out a way to solve the problems of blurred edge perceptual quality images (especially for the blurred edge).

All our team members are committed to the success of the team and our shared goals for this project. We are aiming to provide a better solution for the SR problem.

5 References:

1. C. Ledig et al., "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 105-114, doi: 10.1109/CVPR.2017.19.
2. Dong, C., Loy, C., He, K., & Tang, X. Image Super-Resolution Using Deep Convolutional Networks. (2015, July 31). Retrieved November 04, 2020, from https://arxiv.org/abs/1501.00092
3. Patti, A. J., and Y. Altunbasak. "Artifact Reduction for Set Theoretic Super Resolution Image Reconstruction with Edge Adaptive Constraints and

Higher-Order Interpolants." *IEEE Transactions On Image Processing*, vol. 10, no. 1, Jan. 2001, pp. 179–186.

4. Pandey R.K., Saha N., Karmakar S., Ramakrishnan A.G. MSCE: An Edge-Preserving Robust Loss Function for Improving Super-Resolution Algorithms. In: Cheng L., Leung A., Ozawa S. (eds) Neural Information Processing. ICONIP 2018. Lecture Notes in Computer Science, vol 11306. Springer, Cham.