Hello everyone. My name is Xinghao Chen, and I am along with my teammate Meng Zhang. We are both from Georgia Tech Shenzhen Campus, and we are delighted to share our project on image super-resolution. Firstly, I would like to apologize for giving such a dull aca’demic style presentation instead of the selected brisk videos on our professor’s website. I am sorry but we do not really have time to work on an appealing video.

So let’s get started.

We will first introduce our research topic. That is, what is super-resolution. Then we will show the approaches and outcomes of some state-of-the-art super-resolution methods. // Our own project begins from the third part. We will demonstrate how we reached our goal, and we will include our referenced papers and introduce our network which challenged SotA performance.

Let’s first discuss what super-resolution is. From my perspective, we can view super-resolution as an upscaling interpolation problem. But it is different from ordinary bicubic or bilinear interpolation, because we should improve the image quality. Our output should be visually appealing, and we must infer the information hidden in the small, low-resolution image. Usually in academic researches, we execute super-resolution on a degraded dataset, and compare our results with the groundtruth, and we calculate the average PSNR and SSIM of our output to measure our performance. And I also think that we should consider subjective quality assessments, because super-resolution finally serves our eyes instead of a groundtruth.

Now we are watching the landscape of super-resolution methods from academic papers. The best super-resolution methods are all based on very deep convolutional networks, but non-deep-learning methods are still active because they can be computed quickly. So they are good for real-time applications. The paper selected by our professor is marked RDN in the chart, and we are expected to improve its performance. We can see from the bottom of the chart that these networks have tens of millions of parameters. That is really a heavy burden for our common hardware.

Then we are showing the operations in the SotA networks. I have given up running an RDN network when I saw it has trillions of operations. So when I wrote the initial project proposal, I had expected that I must find something lighter than that.

Now I am showing the performance of RDN, and we are trying to break the record achieved by it. I would say that we only had time to run an experiment on Set14 2 times upscaling. We will show our results later.

OK, now I am introducing my initial goal when I wrote the proposal. I think it would be very difficult to break the PSNR record, because it’s truly high even compared with the best networks. And meanwhile we do not have powerful GPUs. So our target is to break the SSIM record without a severe loss in PSNR. And I think our subjective output quality should be good enough.

Here are our referenced papers. We collected some best networks in recent years and we combined some methods used by them. We majorly referred to RDN and ZSSR. And we also studied the basic network structures like DenseNet and ResNet. We also managed to select a nice loss function after reading researches on losses.

First we are to look at our network architecture. We applied residual learning, which became a trend after ResNet was published. Residual learning is simply to add the input to the output. In this way we can ask the model to output only the difference between the input and output. And compared to a model which has to output the true values, this technique saves the model complexity.

Secondly we used dense connection. This is adding the output of every layer to every subsequent layers. This technique is similar to residual learning to some extent. That is like we are using residual learning almost everywhere. But here the combination of input from different layers is not element-wise addition. So all the features are fed into the convolutional layer as discrete data. This method is advertised by DenseNet, and is good for data propagation, and it reduces the model complexity.

Finally we have RDN which uses both residual learning and dense connection. In our network we also used this strategy.

Then we are to see some operations that we called “back-projection”. Now we are going to upscale an input high-resolution image, but we firstly degrade it into half the size to have a low-resolution image. We do so just using bicubic interpolation. And then we upscale the LR image again using bicubic interpolation. Now we have got two images of the same size, and can be studied by our convolutional network. We would majorly base our training on the pair of HR images.

Now we are going to train our model, We first generated out back-projected HR image, and compare it with the original input. And we feed the model with 128-pixel blocks. Our model is built with 8 layers of convolutional filters. This is much lighter than RDN. After one epoch of training we will generate another back-projected image. And we will use some random data augmentation when we generate the HR image. When the training is finished, we just upscale the original HR image with bicubic upscaling, and we feed the super-resolution sized image to the network. This network will amend the quality of our image with residual learning.

Then I am going to introduce our interesting feature of the network. Our network is unsupervised. So in other words, we do not rely on extra training set when we upscale the image, and the training process is done just after the input is given. We must train another specific network only for that new input image. This is going to generate some benefits for us. First we know that when we process a public dataset, we cannot assume we know the degradation kernel. But supervised learning tries to extract the kernel from a lot of pairs of images. So the kernels for different images are potentially not the same. And we actually do not know what is the good kernel for any given image. But with unsupervised learning we can estimate a good kernel only for that image. And we may avoid some generalization problems when we only have to deal with our input image. And the third advantage is that we do not need any training dataset at all. We do not need a bunch of GPUs to train or use an extremely heavy model.

But in supervised learning we usually need some data augmentation techniques to remedy some problems caused by insufficient data. So we just used some common methods in image augmentation. We have affine transforms. That can mirror the image or shear it and rotate it. We have shifting which moves the boundary of the image to another side. We may try some different kernels in back-projection, and the upscaling and downscaling kernel do not have to be the same. We could add some random noise, but this is not implemented by us. And we can try to upscale the image for 1.5 times, and gradually make it to 2 times. Usually lower times of magnification gives better signal-to-noise ratio, and our neural network can also have more time to learn the features better.

Then we are talking about our loss metrics. This is playing an important role in our work, but it just required some very simple adjustments. We know that our goal is to improve the SSIM, but at first we did not come up with very good ideas to actually enhance it. Then we considered adding it directly to the loss. But as far as I know at that time, the most common losses are L1 and L2 losses. That is the mean absolute error and mean squared error. Some people tried some combination of L1 and L2 loss and found it to be effective. Finally we chose our loss as minus 40 SSIM minus PSNR. This is usually a minus value, but when the loss get smaller, we do have better image quality. This is actually using loga’rithmic element-wise error. That is quite different from any combination of L1 and L2, because when the PSNR is large enough, our network will pay much less attention to PSNR or element-wise error. Instead it will pay most efforts on optimizing SSIM. This design ensures that our network is much likely to achieve good score in SSIM. And certainly this loss focuses more on our perceptual visual quality.

This is our results. We have really defeated RDN plus substantially in SSIM, and our PSNR is just OK, but we had a quite large gap in PSNR when compared to SotA methods. However its lucky that our output images are still of good visual quality. And with our subjective assessments, we think that our network does preserve the low-frequency features, but the high-frequency details are usually lost. This is because we have no information about the SR groundtruth in the training set, and it can be much harder for us to recover the details from a degraded input. In the future we may introduce some weak supervision to improve the ability to recover delicate details.

Then I am going to introduce an example of high-frequency loss. We can see the roof of the palace is quite blurred after processing. This is not processed by our final network, and I am just showing the example to help you observe our results better.

Here are our final results. We can see that the boat gets blurred on the edges, and the SIEMENS text get unrecognizable after super-resolution. These are true problems of our network, and we can still do more work to improve its subjective output quality.

And then we are going to raise a hypothesis on the effects of SSIM loss function. We think that the SSIM loss may generate very good SSIM when the global contrast of the input image is very high. We can see that in the first row, the input image consists of all kind of colors and detailed features that can be clearly seen by our eyes. And finally our output SSIM is truly higher even if the PSNR is low. And in the second row, we have the hairs of the boy almost unrecognizable. We cannot really tell the detailed features with our eyes, and the output SSIM is not good enough even if the PSNR is relatively higher. In this hypothesis we should notice that we are training the model for each image, and the model may have different behaviors and choices on each of the images.

So these are all of my presentation. Thank you for watching.