Dual-Attention GAN with ODE inspired architecture for Super Resolution COGS181 Final Project

Yuan Gao
University of California, San Diego
Computer Science
y1gao@ucsd.edu

Abstract

Convolutional Neural Network(CNN) based method proved to be very successful for Single Image Super-Resolution(SISR) problems, and Generative Adversarial models (GANs) are commonly used to produce perceptually satisfying image restoration. However, some GAN models like SRGAN[6] for SISR can be greatly improved by exploring the state-of-the-art neural network architecture. In this project, we have made improvement on both training stability and the quality of the recovered image, compared to the vanilla SRGAN. While keeping the same discriminator and loss functions, our model for the generator is based on the existing Residual Dense network (RDN)[1], on which we incorporates a dual-attention (channel-wise and spatial-wise) mechanism by jointly using the attention block inspired by the Squeeze-and-Excitation network[8] and the Non-Local neural network[9]. Furthermore, as neural networks can be viewed as a dynamic system modeled by Ordinary Differential Equations, we further improved our GAN using a ODE-inspired architecture that is consistent with the Runge-Kutta method. From experiments, our GAN produces a much lower generator loss and finer details on recovered images. Furthermore, based on our experiment on a simple vanilla model, we infer that the proposed dual-attention block, when used on a PSNR oriented model, gives better results than the existing single-attention models on our validation set.

1. Introduction

Since the invention of SRCNN[3], tons of deep-learning based model for SISR problems are being proposed every year, and thanks to more and more advanced architecture, the PSNR/SSIM results are gradually increasing on datasets like Set5/Set14. For instance, VDSR[4] took the advantage of residual learning and succesfully introduce a very deep network in solving SR problems. Later

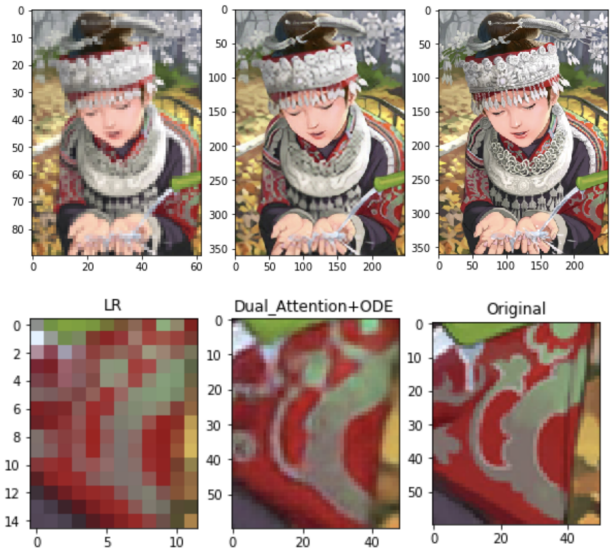

Figure 1. Temporary demo on 4x scaling. This is only from 105 epochs. Finer detailed can be achieved with better training

on, claiming that Batch Normalizatio(BN) layers are not ideal in SR problems due to sensitivity to the scaling, EDSR and MDSR[5] greatly improves the result by removing BN layers and adding a huge number of parameters (43 M). More recent works include the Residual Dense network (RDN)[1] which incorporates stoa Resnet and Densenet, and RCAN[2] which borrowed the idea of adding channel-wise attention from Squeeze-and-Excitation Network. These models above, however, mainly focuses on improving PSNR values for SR problem, which doesn't necessarily reflects human perception of a "high-resolution". By nature, PSNR oriented models always produce very smooth recovering and often leave some details blurred, while human are sensitive to the edges of details. In order to solve this problem, Christian Ledig proposed a new generative model SRGAN[6] which aims toward improving perceptual satisfaction by using the Perceptual Loss[7]. In their

method, they used a Resnet as their generator and a deep CNN as discriminator, and set the generative loss to be a combination of perceptual loss and L1 loss. This architecture proved to be quite successful in terms of recovering details, although often result in a lower PSNR.

The proposed SRGAN, however, is not perfect for several reasons. As mentioned above, Resnet with BN layers often performs worse than without BN layers, since SR models, compared to classification models, are much more sensitive to information like scaling. Furthermore, like for classification problems, vanilla residual network is gradually becoming a starting point and often time, architectures extended from Resnet, including ResNeXt, ResNeSt, SEnet, SKnet etc, gives better results. Also, several tricks for GAN training can be used to produce a more stable training process and faster convergence. Therefore, in this project we mainly tries to improve vanilla SRGAN by only improving the generator while keeping the discriminator and adversarial loss unchanged. We achieved this goal by 1) changing the global structure of generator to be the one from RDN[Fig 1], 2) proposing a dual-attention block as local structure, and 3) applying a 4th-order Runge Kutta inspired network structure.

2. Related work

2.1. Residual Dense network

RDN was originally proposed for generating a high PSNR recovery, which uses a global feature fusion in order to adaptively select features from each block, and a global residual learning in order to increase long-term connection and better recover high frequency image, and reduces computational complexity by learning only the residual image[4]. One thing worth noticing is that instead of directly upscale the input LR image at the first layer and then do image recovery, RDN(and many other networks from 2017) use a deconvolution layer, specifically a Pixel-Shuffle layer proposed by Shi et al[10] in 2016, which has a great impact on later models for SISR problems–the idea of upscaling in the last layers instead of directly on the input layers has been widely adopted and has proved to outperform it's counterpart. For simplification and better comparison, we choose our global structure to be same as Residual Dense net's(RDN)[1] structure. Our work is different from and potentially an improvement on RDN, **since instead of using the original Residual-Dense blocks, we added a dual-attention block and created several side connections in a residual block that essembles what Runge Kutta does**.

2.2. Runge Kutta structure for Residual models

According to work from Chang et al[12], a residual based network can be viewed as a dynamic system modeled by a simple ordinary differential equation. Specifically, if we let Y_j to be the output of jth layer of a Resnet, and $G(Y_j, W)$ represent a local operation inside a resblock, then the whole Resnet can be modeled as

$$Y_{j+1} = Y_j + G(Y_j, W)$$

Here if we implicitly add a step size $h = \delta t$, we get

$$Y_{j+1} = Y_j + h * F(Y_j, W)$$

where $G(Y_j, W) = h * F(Y_j, W)$. Therefore, by using a simple discretization, we get

$$\frac{\delta Y(t)}{\delta t} = F(Y(t), W)$$

Based on this idea, He et al proposed a ODE inspired Network design for super-resolution in their model named OISR[11]. The main idea is that since more complex numerical methods like Runge-Kutta(RK) gives better approximation than Euler's method, then it might be useful here as well in term learning a map $M(y_0, t)$ which satisfies the system above. Their work proved to be useful in terms of increasing precision and thus decreasing loss. However, OISR only considers RK method with order up to 3. Based on this inspiration and the fact that numerical methods based on higher-order taylor expansion gives a much more precise approximation, we infer that when the depth of the residual-based neural networks are increasingly large (meaning the time span t in the system gets larger, higher order methods are more needy. Thus, we proposed a network based on 4-th order RK.

2.3. Attention models

Since the success of attention-based models in NLP problems, they also turn out to be very useful in solving vision-related problems. For instance, the Squeeze-and - Excitation network gained its success in ImageNet classification competition by proposing a channel-wise attention model. Their finding is quite interesting: in image classification problems, they see a distinct difference between different category of images in excitation level in different channels. This means for different objects, the neural network should adaptively pay different attention to different channels. However, traditional convolution layers lacks the ability to do this and always give same attention to each channel. Here we adopt the idea of channel-wise attention for SR problems since we think during the learning process the network should be able to differentiate between textures/patterns, thus can learn to recover those details better. Unfortunately, our idea happens to coincide with a paper published a year ago on RCAN[12], but we are still happy about this since it supports our idea. Beside channel-wise attention, spatial attention should also be useful. Here we borrowed idea for the Non-local network proposed by Wang

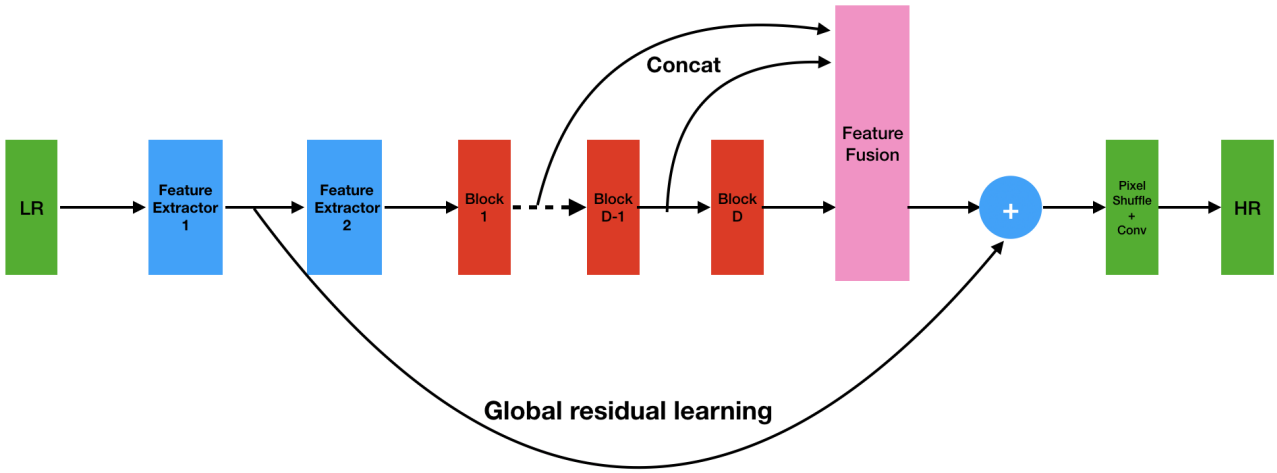

Figure 2. Our global architecture, which is same as RDN's global architecture

et al[13](who is coming to UCSD this fall), which was originally proposed for video-classifictaion, but is also useful for tasks related to images. In this project we apply adopt idea to low-level vision problems and choose to use Guassian instantiation or embedded Guassian instantiation. **Although there are already models on SR problems based on the attention models above, there are few that combines those two attention mechanisms.** In our project we combine these two attention blocks and when tested on a small architecture based on RCAN, it terms out to perform better than the single channel-wise attention model.

2.4. SRGAN

SRGAN[6] is the first successful GAN model in image super-resolution and is the first that successfully produces perceptually satisfying image. It uses an adversarial loss based on a perception loss and a content loss, which helps it to successfully produce much finer details. This successful work is chosen to be an oral presentaion in CVPR 2016.

3. Proposed method

3.1. Global architecture

As stated before, our global architecture is based on RDN's global architecture. The architecture is shown in figure 2, and here are some clarifications. Feature Extractor is consists of a single convolution layer; Blocks can essentially be any blocks like basic Resnet block, Densenet block, Residual Dense block as in RDN, or here we use our dual-attention block w/o RK design (Figure 3, 4). Feature fusion is used to select features from all previous block, and is consists of a concatenation block and a convolution layer for fusion after concat. Pixel-Shuffle is essentially a deconvolution layer which upscale the feature map for scaling. Mathematically, the process can be shown as

$$HR = F_{ps}(F_{fe1}(LR) + F_{ff}(F_{blocks}(F_{fe2}(F_{fe1}(LR)))))$$

3.2. Pure dual-attention block architecture

Our proposed dual-attention block can be viewed as an 'add-on' at the end of several non-linear opertaions inside a residual block. As shown in figure 3, the Basic Block can be any blocks that yield an arbitrary feature map, and we add the 'add-on' after that block for learning channel-wise and spatial-wise inter-dependencies. Empirically, spatial attention before channel attention performs better than the other way. The spatial attention is of form

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j) g(x_j)$$

and here we use the two simplist spatial-attention model from Non-local neural network, which corresponding to

$$f(x_i, x_j) = e^{x_i^T x_j}$$

or

$$f(x_i, x_j) = e^{\theta(x_i)^T \phi(x_j)}$$

3.3. Dual-attention block with high order RK design

Runge-Kutta 45 method is very precision in terms of solving ODEs numerically. The reason we choose 4th order specifically instead of the RK3 shown in OISR is that

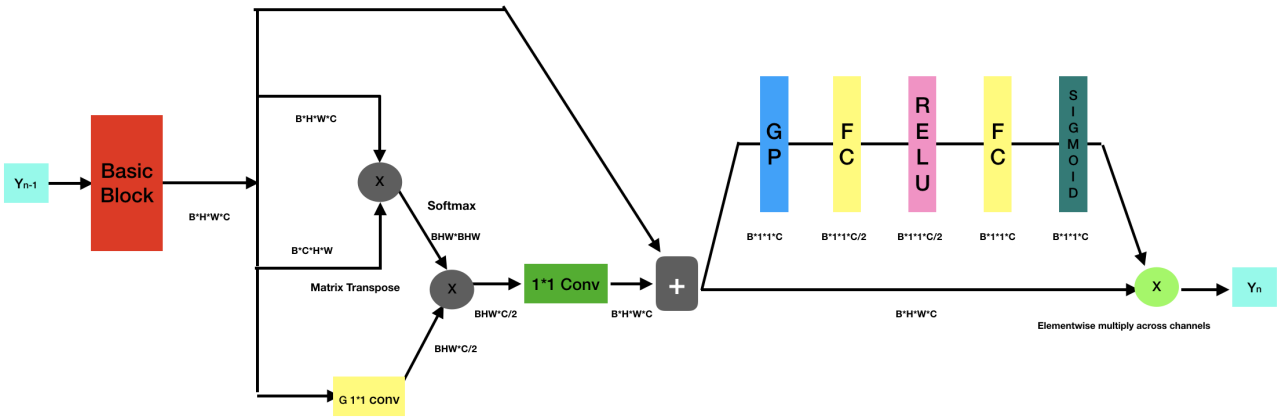

Figure 3. Dual Attention add-on. Here we apply Guassian instantiation.

RK4 is most widely used in computational science. It is derived from up to 4th-order derivative in taylor expansion and has an error of $\mathcal{O}(\Delta t^5)$, while traditional ResNet can be viewed as euler's method and is $\mathcal{O}(\Delta t^2)$. Thus we infer that when dealing with very deep neural networks, our RK4 method can be more precise in terms of learning in the dynamic system. Original RK4 method can be duduced by the following steps:

1) Unlike Euler's method which only uses $f(y_t, t)$ as the only slope, RK-based methods often propose several slopes within a time step to make smaller and preciser steps forward. Here let's propose 4 slopes namely k_1, k_2, k_3, k_4 with the following definition:

$$k_1 = f(y_t, t)$$

$$k_2 = f(y_t + \frac{h}{2}k_1, t + \frac{h}{2})$$

$$k_3 = f(y_t + \frac{h}{2}k_2, t + \frac{h}{2})$$

$$k_4 = f(y_t + hk_3, t + h)$$

2)We can then expand the equation above using second term from taylor expansion and express k_1, k_2, k_3, k_4 in terms of h and $\frac{d^n f}{dt^n}$:

$$k_1 = f(y_t, t)$$

$$k_2 = f(y_t, t) + \frac{h}{2}\frac{d}{dt}f(y_t, t)$$

$$k_3 = f(y_t, t) + \frac{h}{2}\frac{d}{dt}(f(y_t, t) + \frac{h}{2}\frac{d}{dt}f(y_t, t))$$

$$k_4 = f(y_t, t) + \frac{h}{2}\frac{d}{dt}(f(y_t, t) + \frac{h}{2}\frac{d}{dt}(f(y_t, t) + \frac{h}{2}\frac{d}{dt}f(y_t, t)))$$

3)With the combination of the proposed slopes above, we can get a much finer slope at $y(t)$ than just the $f(y_t, t)$. Thus, we can express $y(t + \Delta t)$ as following:

$$y(t + h) = y(t) + h(a * k_1 + b * k_2 + c * k_3 + d * k_4) \quad (1)$$

We can also express $y(t+\Delta t)$ in 5th order taylor expansion:

$$y(t+h) = y(t) + h\frac{dy}{dt} + \frac{1}{2}h^2\frac{d^2y}{dt^2} + \frac{1}{6}h^3\frac{d^3y}{dt^3} + \frac{1}{24}h^4\frac{d^4y}{dt^4} + \mathcal{O}(h^5) \quad (2)$$

By solving equation 1 and 2 we can get

$$y(t + h) = y(t) + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4)$$

Thus, with the expression above we can write our RK4 code as in figure 4. In our project we use structure from figure 3 to be our self.non_linear_block, but in general it can be any type of block. This design will not increase extra parameters except for 4 constants which can be neglected, and in our experiment it indeed boosts the performance. See section 4 for more comparison. However, without through experiment and analysis, we can't make sure this design can be generalized to other tasks like classification. Nonetheless, it is indeed an very interesting inspiration based on the dynamic system view of neural network, and provide an alternative view of deep learning architecture.

```
#We require grad for constants like self.one_sixth since they give better results.
yn = x
k1=self.non_linear_block(yn)
k2=self.non_linear_block(yn+self.one_half*k1)
k3=self.non_linear_block(yn+self.one_half*k2)
k4=self.non_linear_block(yn+k3)
yn_1 = yn+self.one_sixth*k1+self.one_third*k2+self.one_third*k3+self.one_sixth*k4
return yn_1
```

Figure 4. 4th order Runge-Kutta structure.

3.4. Other

4. Comparison of results

4.1. Demo

For 2x scaling:

Figure 5. Result based on 150 epochs while loss is still decreasing. Finer details can be achieved with more training

For 4x scaling:

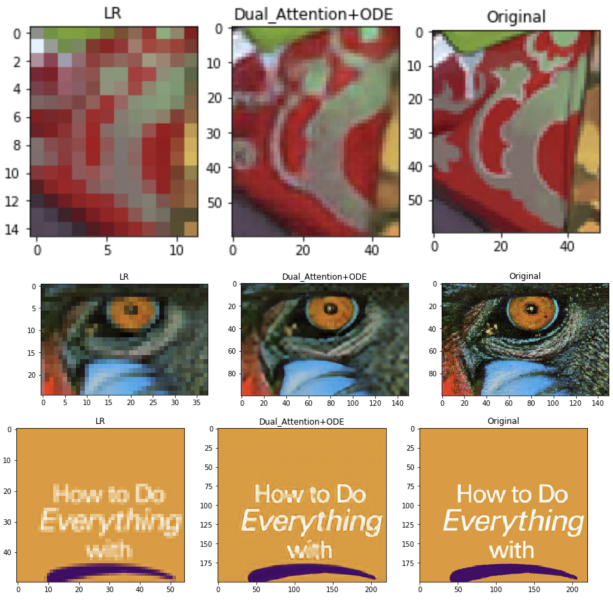

Figure 6. Result based on 105 epochs while loss is still decreasing. Finer details can be achieved with more training

4.2. Training curve comparision

As mentioned before, our dual-attention 'add on', combined with RK methods, produce a much nicer and stable training curve in terms of generator loss. For our 2x scaling model, there is a huge gap between original SRGAN and other methods, while the dual-attention block also outperforms models with no attention or single attention. Since we have limited computation resources, we only train on 100 epochs and we mostly use 2nd order RK method. However, without losing generality, it can also reflect effectiveness of 4th order RK method. The result can be seen in figure 7

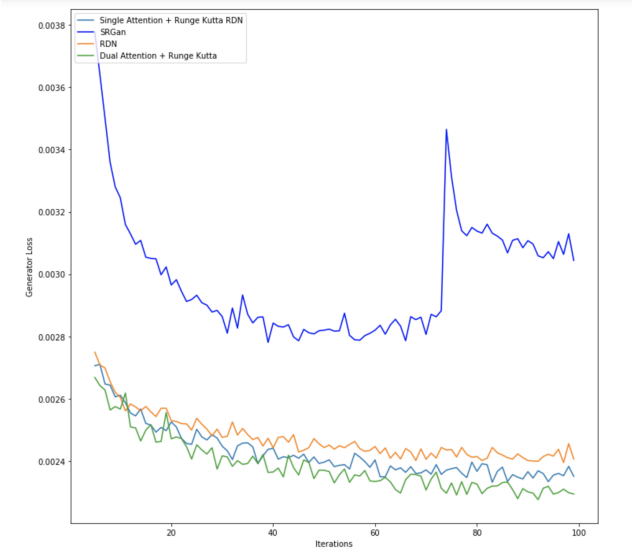

Figure 7. Training curves of different methods. The SRGAN result is from implementation of https://github.com/leftthomas/SRGAN

Also for the 4x scaling problem the dual-attention with ode outperforms no attention. Both model has comparable number of parameters.

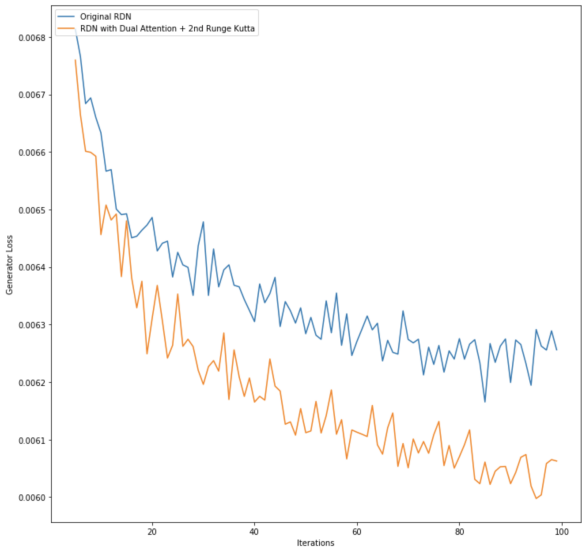

Figure 8. 4x taining loss for generator

4.3. Result on PSNR oriented models

We further demonstrated the effectiveness of the dual-attention 'add-on' by applying it to PSNR oriented models. Here we are mainly comparing PSNR results between a state-of-art single attention model(RCAN) and our dual-attention model. For controlled settings, our model simply change the single attention block from RCAN to the dual-attention block. **Since we have limited computation resources, both models are intentionally be a small model with roughly same number of parameters.** The result doesn't reflect true RCAN results since the RCAN model can be deeper and trained longer. **But we believe one can generalize our conclusion from small model to large model.** Here are Set5 results:

5. Conclusion and future work