Noise Conditional Flow Model for Learning the Super-Resolution Space

Younggeun Kim[1, 3] Donghee Son[2, 3]
[1] Seoul National University, Seoul, Republic of Korea
[2]Lomin Inc., Seoul, Republic of Korea
[3]Deepest, Seoul, Republic of Korea
eyfydsyd97@snu.ac.kr dh.son@lomin.ai

Abstract

Fundamentally, super-resolution is ill-posed problem because a low-resolution image can be obtained from many high-resolution images. Recent studies for super-resolution cannot create diverse super-resolution images. Although SRFlow tried to account for ill-posed nature of the super-resolution by predicting multiple high-resolution images given a low-resolution image, there is room to improve the diversity and visual quality. In this paper, we propose Noise Conditional flow model for Super-Resolution, NCSR, which increases the visual quality and diversity of images through noise conditional layer. To learn more diverse data distribution, we add noise to training data. However, low-quality images are resulted from adding noise. We propose the noise conditional layer to overcome this phenomenon. The noise conditional layer makes our model generate more diverse images with higher visual quality than other works. Furthermore, we show that this layer can overcome data distribution mismatch, a problem that arises in normalizing flow models. With these benefits, NCSR outperforms baseline in diversity and visual quality and achieves better visual quality than traditional GAN-based models. We also get outperformed scores at NTIRE 2021 challenge [21].

1. Introduction

Single image super-resolution is a computer vision task to reconstruct a high-resolution image from its low-resolution image. Super-resolution is an important problem in computer vision due to its various applications including surveillance [34], medical imaging [3, 28], astronomical imaging [25, 19] and object detection [23].

With the development of deep learning in computer vision [7, 9, 10, 11, 26], super-resolution methods based on deep learning [15, 18, 20, 29, 32, 33] improve performance significantly. Models [15, 20, 32, 33] trained with L_1 or L_2 loss achieve high PSNR performance. Likewise, models [13, 18, 29] trained with adversarial loss or perceptual loss accomplish high visual quality performance.

Figure 1: $\times 4$ Super-Resolution result of our model on "0831" from DIV2K validation set compared with other baseline models.

Most super-resolution methods based on deep learning take a low-resolution image as an input, then output a high-resolution image. However, super-resolution is an ill-posed problem. That is to say, one low-resolution image can be mapped from multiple high-resolution images. SRFlow [22] proposed the method that can predict multiple high-resolution images for a given low-resolution image by learning the super-resolution space. [22] utilize normalizing flow to learn super-resolution space.

SRFlow[22] produces a variety of results other than a deterministic super-resolution output, but there are possibili-

This work was done as a project of Deepest

ties to improve performance. Their diversity comes from training with negative log-likelihood, not L_1 or L_2 loss. SRFlow [22] outputs more diverse super-resolution images with better visual quality than not only models that give the deterministic result such as GAN based but also a model that simply creates diversity through noise [27]. However, in addition to training with negative log-likelihood, there are more ways to increase diversity.

In this paper, we propose a model that produces results with more advanced diversity and better visual quality than SRFlow. Our method increases diversity by adding noise. However, simply adding noise to input images generates low-quality images with artifacts. To deal with this problem, we propose a structure called noise conditional layer, which results in superior results in both metrics over SRFlow [22]. We also analyze that these improvements come from resolving data distribution mismatch that exists in other flow models such as [17]. Existing flow-based models aim to map complex data x from simple data z, but when the two manifold dimensions are not the same, flow models are not trained smoothly. Therefore, these distribution dimension mismatches should be addressed. This was addressed in SoftFlow[14] and we applied similar ideas to super-resolution tasks. Our contribution is as follows.

1. We propose a method that can improve performance on learning the super-resolution space using flow model through adding noise and noise conditional layer.

2. Our method improve the performance of the diversity by adding noise to the training data to expand the data distribution

3. Our method solves the performance degradation caused by the mismatch of the data distribution of SR model using normalizing flow.

2. Related Works

2.1. Single Image Super-Resolution

As deep learning-based methods [7, 9, 10, 11, 26] provide significant performance improvement, many single image super-resolution methods [15, 20, 32, 33] based on deep learning are proposed. Dong *et al*. [6] proposed the first super-resolution model based on deep learning. Dong *et al*. [6] propose the model, which use three convolution layers, trained with L_2 loss. After that, many methods [15, 20, 32, 33] which optimized with L_1 or L_2 loss are proposed. Although these models show performance improvement in terms of PSNR, some of their predictions are blurry. To deal with this problem, super-resolution models [18, 29] which use adversarial loss or perceptual loss are proposed. However, these works predict a reconstructed high-resolution image for a given low-resolution image.

2.2. Normalizing Flow

Flow-based model, originally introduced in [4], proposed deep learning framework for modeling complex high-dimensional density. Flow-based models have made many advances to map accurate complex distributions from simple distribution. Several approaches such as [4, 5, 17] use invertible networks to map complex distributions from simple distributions (ex. Gaussian). [8] uses a continuous-time invertible generative model with unbiased density estimation and one-pass sampling. Besides, [14] aims to estimate the conditional distribution of perturbed input data instead of learning the data distribution directly to solve the discrepancy problem of a dimension of data distribution. Recently flow-based models are gaining popularity in the field of image generation [5, 17]. Moreover, [2, 30] present a conditional image generation method based on the Glow architecture. [30] deals with SR tasks but does not produce influential results compared to GAN-based models. For the first time, SRFlow [22] propose a flow-based super-resolution model which outperforms GAN-based models. SRFlow [22] is trained with the negative log-likelihood loss only. By using negative log-likelihood loss, SRFlow [22] solves the deterministic output problem posed by the previous super-resolution works and learns to generate diverse photo-realistic super-resolution images. Our method following Glow architecture [17] along SRFlow, solves the data distribution problem like [14] and generates super-resolution images of better visual quality and more diverse outputs.

3. Method

Our main goal is to learn super-resolution space. In other words, we aim to generate diverse super-resolution images with high visual quality for a given low-resolution image. In this section, we introduce our proposed method. Firstly, we will briefly address the background to understand our model. Next, we will discuss how to improve diversity. Finally, we explain the noise conditional layer which improves image visual quality and diversity by solving mismatch in the distribution of data.

3.1. Background

Flow-based model is one of the effective methods for predicting the complex distribution of real data. These models aim to convert from a simple (ex. Gaussian) distribution to a complex (ex. Real-world) using a series of invertible functions. These properties make complex data x can be always reconstructed from z, which is the latent vector. These flow-based generative models are defined as:

$$z \sim p_z(z)$$

$$x = g(z),\ z = f(x)$$

$$f(x) = f_n \circ f_{n-1} \circ \cdots \circ f_1(z)$$

In this case, z is the latent variable, f and g are invertible to each other, $z = f(x) = g^{-1}(x)$. Moreover, the flow model f consists of invertible transformation, which maps dataset x to Gaussian latent variable z and each f_i has a tractable inverse and a tractable Jacobian determinant. The series of invertible transformations is called a normalizing flow, and the advantage of this normalizing flow is that probability density p_x can be written as follows by applying the change of variable formula:

$$p_x(x|\theta) = p_z(f_\theta(x))|\det \frac{\partial f_\theta}{\partial x}|$$

It allows network to be trained through the following objective functions.

$$-\log p_x(x|\theta) = -\log p_z(f_\theta(x)) - \log|\det \frac{\partial f_\theta}{\partial x}|$$

Model f is trained by directly minimizing negative log-likelihood. These training methods prevent the output of the model from being deterministic. This guarantees the diversity of the output.

SRFlow [22] is a normalizing flow-based super-resolution method that, given a low-resolution image, can learn a super resolution conditional distribution for that image. They utilized the basic Glow architecture and modified the existing flow step to create the conditional flow step. The conditional flow step consists of Actnorm, 1×1 convolution, Affine injector, and Conditional affine coupling. The parts that learn low-resolution images conditionally are affine injector and conditional affine coupling.

Affine injector is as follow:

$$h^{n+1} = exp(f^n_{\theta,s}(u)) \cdot h^n + f^n_{\theta,b}(u)$$

which, $f^n_{\theta,s}$ and $f^n_{\theta,b}$ can be any network and $u = g_\theta(x)$ is low resolution image encoding , where g_θ is encoder network.

Conditional affine coupling is as follow:

$$h^{n+1}_A = h^n_A$$

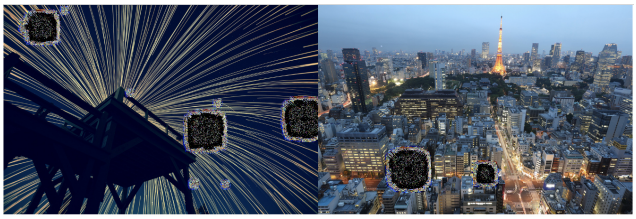
$$h^{n+1}_B = exp(f^n_{\theta,s}(h^n_A; u)) \cdot h^n_B + f^n_{\theta,b}(h^n_A; u)$$

Similar to affine injector, $f^n_{\theta,s}$ and $f^n_{\theta,b}$ can be any network and $u = g_\theta(x)$ is low-resolution image encoding. Moreover, $h^n = (h^n_A , h^n_B)$ is a partition in the channel dimension.

3.2. How to improve Diversity

Learning the super resolution space was already addressed in [22]. [22] was trained with negative log-likelihood, which does not use the loss as L_1 loss. We use [22] which obtained a meaningful diversity as baseline.

To achieve a higher diversity score, we started from the fact that flow-based models aim to match the distribution of simple data z with complex data x.



DIV2k 828
LR-PSNR: 21.53

DIV2k 873
LR-PSNR: 24.11

Figure 2: These images are samples of super-resolution results of DIV2k validation set from a model trained only with injecting noise, without noise conditional layer.

1. By varying the distribution of complex data x, i.e. high-resolution images, the distribution of super-resolution outputs mapped from data z will also vary during the inference process.

2. Among the methods to vary the distribution of this data x, we use noise injection.

$$x^+ = x + noise$$

$$f^{-1}(x^+|y) = z$$

where, x is high resolution image, f is invertible model that maps z to x and y is low resolution image. However, while the method of varying the distribution by adding noise to the high-resolution image has increased the diversity significantly, this is not the exact LR image conditional training. These training methods generate super-resolution images with severe noise when generating images from simple distribution z.

3. Therefore, for exact LR image conditional training, noise injected into the high-resolution image was resized to the size of LR image and added to the LR image for training.

$$x^+ = x + noise$$

$$y^+ = y + noise^-$$

$$f^{-1}(x^+|y^+) = z$$

where, $noise^-$ is a vector whose noise is resized in the same size as y

This method leads to an improvement in diversity. However, we address that this noise injection causes artifact. This can be seen in Figure 2. Therefore, we propose the structure of the model to remove the noise-induced artifact.

3.3. Noise Conditional Layer

It is our motivation to inform the model of information about noise for removing the artifacts caused by adding noise. By training the model with noise information, noise

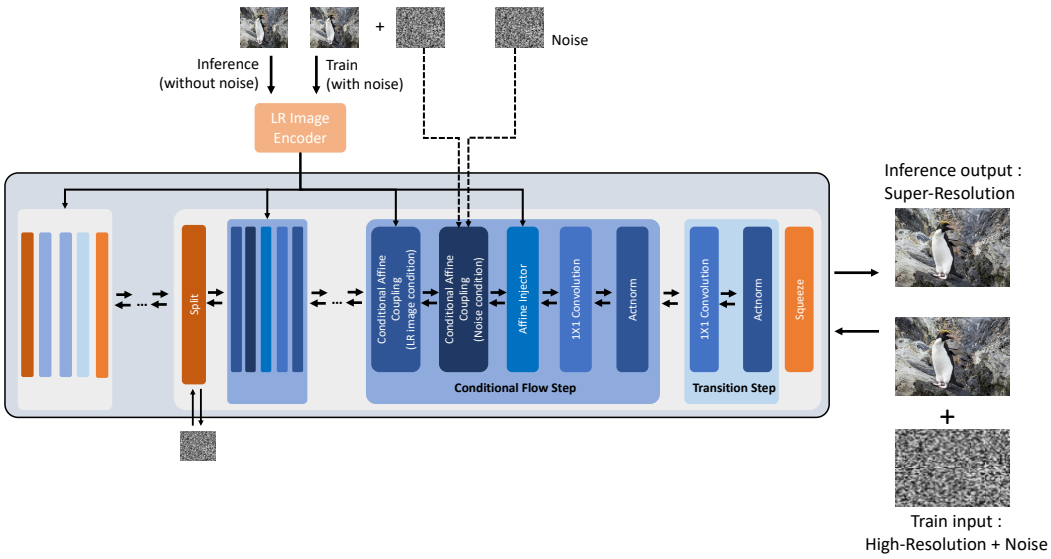


Figure 3: Our method proposes a flow model with noise conditional layer. Our method adds the same noise in LR images and HR images and proceeds with noise conditional training according to these noise distributions.

will be reflected for generating images. Due to reflected noise information, images with reduced artifacts can be generated. Therefore, we add layers that inform noise to the model structure. We propose a noise conditional layer.

Our method is as follows. Initially, random value c is obtained from uniform distribution $U(0, M)$ as [14] did. Next, set noise distribution $N(0, \Sigma)$, where $\Sigma = c^2 I$. Then, we sample noise vector v from $N(0, \Sigma)$ and add noise to the original high resolution image x to obtain perturbed data x^+. Finally, resize these vector v to get noise vector w for low-resolution images and obtain y^+ by adding w to the original low resolution image y.

$$x^+ = x + v$$

$$y^+ = y + w$$

$$f^{-1}(x^+|y^+, v) = z$$

where, x is high resolution image, y is low resolution image, v is noise vector and w is a noise vector which is resized in the same size as y. At this point, model f is trained with LR image and noise information. Thus, the goal is to obtain a model conditioned on the noise and LR image that converts latent variable z to x^+ given the vector v and y^+. ($i.e.$ $f(z|y^+, v) = x^+$)

Moreover, we conduct noise conditional training in two ways, one for noise itself and one for standard deviation for noise distribution. We proceed both methods in a similar way to the conditional affine coupling of [22]. Although standard deviation conditional training, such as those used in [14], improves diversity and LPIPS[31], it tends to create artifact from the generated images. In contrast, with noise conditional training, the numerical performance is slightly lower, but the frequency of occurring artifacts in the generated images is reduced and we finally adopt noise conditional training.

We also deal with the following problem:

- The flow-based model aims to create a complex distribution, x, from a simple distribution z. However, the manifold data dimension between data x and data z is not always the same. This makes it difficult to predict the distribution of complex data.

Our method solves the mismatch of data distribution. The idea that solves this problem exists in SoftFlow[14], which adds noise to improve the performance of the flow model. [14] proposes to estimate the conditional distribution of perturbed data to diminish the dimension difference between these data and the target latent variable. The key here is to add noise that is obtained from randomly selected distribution and to use these distribution parameters as conditions. [14] has shown that these methods can experimentally succeed in capturing the innate structure of manifold data. In the same principle, we increase performance for learning the super-resolution space and image visual quality using normalizing flow through adding noise and noise (distribution parameters) conditional training.

Our model goes through the same process as [22]: squeeze, flowstep, split. Similar to [22], the LR image is encoded through the low-resolution encoder, which is used for conditional training. Also, flowstep consists of transition step and conditional flow step, which is equivalent to [22]. The difference between [22] and our model lies at the core

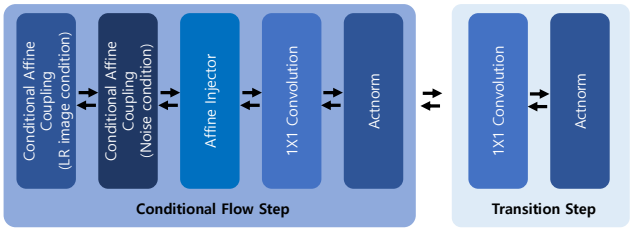


Figure 4: The key to our model is the Flowstep block in the picture above.

of SRFlow, conditional flow step. The noise conditional layer is added to the existing four configuration steps. In other words, it consists of five steps: actnorm, 1×1 convolution, affine injector, and two conditional affine couplings (Noise conditional layer, LR conditional layer). There is the structure of five steps in Figure 4. Moreover, only negative log-likelihood was used for loss, like [22]. The network is trained with the aim of minimizing the following negative log-likelihood.

$$-\log p_{x|y,v}(x|y,v,\theta) =$$

$$-\log p_z(f_\theta(x;y,v)) - \log|\det\frac{\partial f_\theta}{\partial x}(x;y,v)|$$

where x is high resolution image, y is low resolution image and v is noise vector. During inference, we add a zero vector instead of noise. In inference, because we add zero vector, we need dequantization the same as SRFlow. This architecture enables the model to perform noise conditional training, resulting in the reduced artifact.

Noise conditional layer performs better than existing models because our method not only recovers image visual quality dropped by noise injection, but also solves the problem with data distribution mismatch. Noise conditional layer solves the problem of data distribution mismatch and increase the diversity of the data distribution to add noise to the underlying HR image, resulting in performance improvements in both visual quality and diversity.

4. Experiments

4.1. Dataset and Metric

We use DF2K dataset, which is a merged dataset with DIV2K [1] and Flick2K[1], for training the proposed model. DIV2K dataset [1] is composed of 800 train images, 100 validation images, and 100 test images. Similarly, Flick2K contains 2560 training images. Additionally, we use crawled 498 high-resolution images, which are 2K resolution, from the Unsplash website[2] to increase the amount of train data for NTIRE 2021 challenge [21]. We refer to the

[1] https://github.com/limbee/NTIRE2017

[2] https://unsplash.com

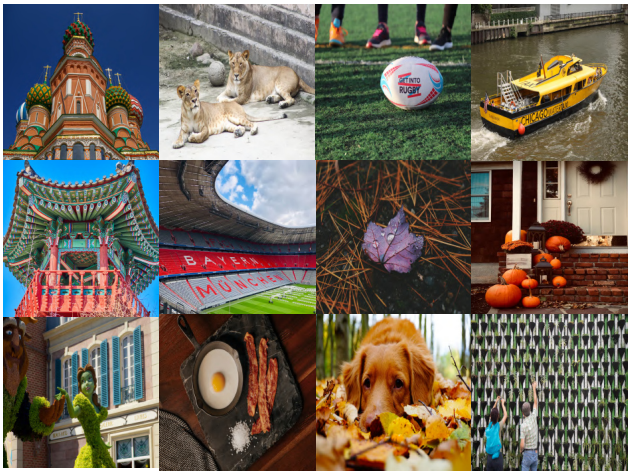


Figure 5: The sample images of Unsplash 2K

crawled dataset as Unspalsh2K. Figure 5 shows sample images of Unsplash2K dataset. Unsplash2K dataset is publicly available[3]. For testing, we use the DIV2K validation images because the ground truth images of the DIV2K testset are not publicly available.

We use the following metrics for comparing performance.

- To compare the visual quality, we use LPIPS [31] as several works [12, 22] did. LPIPS [31] is computed by measuring distance in feature space between two images.
- NTIRE 2021 challenge [21] uses a diversity score to measure the spanning of the super-resolution space. We also use the same metric to evaluate diversity. They calculated the diversity in the following way. Sample 10 images and calculate global best and local best between the samples and the ground truth. The local best is the full image's average of the best score for each pixel out of 10 samples. The global best is the best average of the whole image's score. The diversity formula is as follows:

$$diversity = \frac{global\_best - local\_best}{global\_best} * 100$$

- To measure the low-resolution consistency, we use the average LR-PSNR of 10 samples. The LR-PSNR is calculated by computing the difference between downsampled prediction image and a low-resolution image.
- We use the LR-PSNR worst, which is the minimum value, of 10 samples to check if artifacts exist. If LR-PSNR worst is low, some of the generated samples have terrible artifacts.

[3] https://github.com/dongheehand/unsplash2K

Model	Diversity	LPIPS	LR PSNR
RRDB [29]	0	0.253	49.20
ESRGAN [29]	0	0.124	39.03
ESRGAN+ [27]	22.13	0.279	35.45
SRFlow [22]	25.26	0.120	49.97
NCSR (Ours)	26.72	0.119	50.75
NCSR* (Ours)	26.79	0.118	50.88

Table 1: General image SR $\times 4$ results on the 100 validation images of the DIV2K dataset

Model	Diversity	LPIPS	LR PSNR
RRDB [29]	0	0.419	45.43
ESRGAN [29]	0	0.277	31.35
SRFlow [22]	25.31	0.272	50.00
NCSR (Ours)	26.8	0.278	44.55
NCSR* (Ours)	25.7	0.253	49.97

Table 2: General image SR $\times 8$results on the 100 validation images of the DIV2K dataset

4.2. Implementation Details

We describe the training details and model hyperparameters in this section. For each training step, 18 high-resolution patches are extracted. The size of the extracted patch is 160×160 and extracted patches are used as ground-truth. The low-resolution images downsampled from high-resolution patches via bicubic downsampling are used as input. Both input images and ground-truth images are normalized to $[0, 1]$.

RRDB [29] is used for low-resolution encoder for our proposed model. Noise conditional layers are added at the first and the second block based on the order of inference. We use ADAM optimizer [16] by setting $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 10^{-8}$. To augment data, we randomly rotate patches 90, 180, 270 degrees and randomly flip horizontally. The learning rate is initialized to 2×10^{-4}. The learning rate is halved at 110K and 165K updates. The other settings are the same as SRFlow [22]. PyTorch [24] is used to implement our model. The code and pretrained models are publicly available[4].

4.3. Comparision with other models

To show superiority of our model, we compare our method with other super-resolution methods. We compare performance with RRDB [29], ESRGAN [29], ESRGAN+ [27] and SRFlow [22]. RRDB [29] is PSNR oriented model which is trained with L_1 loss. ESRGAN [29] and ESRGAN+ [27] are GAN based model. SRFlow [22] is Flow-based model.

We evaluated the performance of these models with three metrics: LPIPS, diversity score, LR-PSNR. For $\times 4$ SR model, our proposed model outperformed the state-of-the-art model SRFlow on all metrics as shown in table 1. Furthermore, our model achieves superior result than GAN based model in terms of LPIPS, which is a perceptual measure.

Similarly, we measured the performance of our model for $\times 8$ SR. We do not compare with ESRGAN+, because there is no model for $\times 8$ SR. In $\times 8$ task, our method shows a better diversity score than SRFlow. Comparable results are also shown in terms of LPIPS and LR-PSNR.

We also do an experiment by adding Unsplash2K, which is an extra training dataset. When we add Unsplash2K to train data, all metrics are slightly better for both $\times 4$ SR and $\times 8$ SR. In table 1 and 2, NCSR means the model trained with DF2K only, and NCSR* means the model trained with DF2K and Unsplash2K.

Qualitative results are shown in Figure 6, 7. Figure 6 shows that our proposed model reconstruct textures and details compared to other works. Random samples generated by our model are shown in Figure 7.

For the flow-based SR model, we set the temperature as 0.9. However, the temperature is 0.85 for our $\times 8$ SR model trained with DF2K only.

4.4. Ablation Study

In this section, we analyze the performance of the proposed model according to three factors.

Std conditional layer vs. noise conditional layer We compare the performance of which value is used for conditional layer. While the noise conditional layer uses sampled noise, the standard deviation conditional layer uses the standard deviation of sampled noise, which is injected into high-resolution images. As you can see in the table 6, the standard deviation conditional layer makes LR-PSNR worst value lower than our noise conditional layer. This means that the standard deviation conditional layer creates many artifacts. It is because the standard deviation conditional layer did not provide sufficient information to the model. Therefore, the standard deviation is not enough for noise-aware training when noise-injected inputs are given.

With or without noise conditional layer We investigate the effect of the noise conditional layer. Without noise conditional layer, noises are injected in input images. As you can see in Table 5, LPIPS is high without a noise conditional layer. In other words, the image visual quality is not good. Furthermore, LR-PSNR is low, which means that there are lots of artifacts. Images with artifacts can be seen in Figure 2. Table 5 shows the detailed performance comparison with the presence of noise conditional layer.

Where to add noise conditional layer We also experi-

[4] https://github.com/younggeun-kim/NCSR

Figure 6: Qualitative comparisons with other methods for $\times 8$ SR model.

Figure 7: Random samples generated by NCSR. Upper : $\times 4$ SR model, Lower : $\times 8$ SR model

Team	LPIPS	LR PSNR	Div. Score
svnit_ntnu	0.355	27.52	1.871
SYSU-FVL	0.244	49.33	8.735
nanbeihuishi	0.161	50.46	12.447
SSS	0.110	44.70	13.285
FudanZmic21	0.273	47.20	16.450
FutureReference	0.165	37.51	19.636
SR_DL	0.234	39.80	20.508
CIPLAB	0.121	50.70	23.091
BeWater	0.137	49.59	23.948
njtech&seu	0.149	46.74	26.924
Deepest(ours)	**0.117**	**50.54**	**26.041**

Table 3: **Quantitative results for NTIRE 2021 Challenge on Learning Super Resolution Space on $\times 4$ track**

Team	LPIPS	LR PSNR	Div. Score
svnit_ntnu	0.481	25.55	4.516
SYSU-FVL	0.415	47.27	8.778
SSS	0.237	37.43	13.548
FudanZmic21	0.496	46.78	14.287
SR_DL	0.311	42.28	14.817
FutureReference	0.291	36.51	17.985
CIPLAB	0.266	50.86	23.320
BeWater	0.297	49.63	23.700
njtech&seu	0.366	29.65	28.193
Deepest(ours)	**0.259**	**48.64**	**26.941**

Table 4: **Quantitative results for NTIRE 2021 Challenge on Learning Super Resolution Space on $\times 8$ track**

Model	w/o NCL	with NCL
Diversity	25.38	26.72
LPIPS	0.1228	0.1193
LR PSNR	50.08	50.75
LR PSNR-worst	47.32	49.14

Table 5: Performance comparison between model with noise conditional layer and without noise conditional layer

mentally show that it is recommended that the noise conditional layer is only included in the first block and the second block based on the order of inference. The model with a noise conditional layer in all blocks generates images with artifact. They show slightly better scores in terms of diversity, but due to the occurrence of these artifacts, we adopt to add our noise conditional layer in the first and second block. When generating super-resolution output, we need an interval at the end of network to generate the image without such noise because of noise added in the input and the noise conditional layer.

That is to say, we find that there should be noise-free block at the end of network. The noise-free block is the block that there is no noise conditional layer. Therefore, if there is no **noise-free block**, this could be the factor that make the artifact. You can show the results described above in the following table 6.

5. NTIRE2021 challenge

Our method, NCSR, scored high in both tracks of NTIRE 2021 Learning Super Resolution Space Challenge [21]. To measure how much information is preserved in the super-resolution image from the low-resolution image, the competition measured the LR-PSNR. In this competition, a team with high perceptual image visual quality and diversity scores becomes the winner in the case that LR-PSNR exceeds only 45. Among teams with LR-PSNR over 45, we take the first place in LPIPS and the second place in diversity score for $\times 4$ track. Moreover, in the $\times 8$ track, our model takes the first place in both LPIPS and diversity scores. The quantitative results of NTIRE 2021 Super-Resolution Challenge are shown in Table 3, Table 4.

6. Conclusion

We propose noise conditioned flow model for learning super-resolution space. Our proposed model uses a noise conditional layer to generate more diverse super-resolution images with high visual quality.

To learn more diverse data distribution, we add a random noise to images. Although data distribution is broader, adding noise cause artifacts with terrible quality in super-resolution images. Therefore, a noise conditional layer is

Noise Conditional Layer	✗	✗	✓	✓	✓
Std Conditional Layer	✗	✓	✗	✗	✗
Noise-free block	✗	✓	✗	✓	✓
Add extra data	✗	✗	✗	✗	✓
LR-PSNR worst	47.32	45.78	49.01	49.14	50.13

Table 6: LR-PSNR worst comparison for Ablation Study

proposed for stable training when noise is added to images. By using the noise conditional layer, we can obtain more diverse super-resolution images without visual degradation. We show the superiority of our proposed model on the DIV2K dataset under several settings. Furthermore, our proposed model achieves high quantitative results on NTIRE 2021 Super-Resolution Challenge [21].

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017.

[2] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*, 2019.

[3] Yuhua Chen, Feng Shi, Anthony G Christodoulou, Yibin Xie, Zhengwei Zhou, and Debiao Li. Efficient and accurate mri super-resolution using a generative adversarial network and 3d multi-level densely connected network. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 91–99. Springer, 2018.

[4] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[6] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.

[7] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[8] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[12] Xiaozhong Ji, Yun Cao, Ying Tai, Chengjie Wang, Jilin Li, and Feiyue Huang. Real-world super-resolution via kernel estimation and noise injection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 466–467, 2020.

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[14] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. Softflow: Probabilistic framework for normalizing flow on manifolds. *arXiv preprint arXiv:2006.04604*, 2020.

[15] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

[18] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[19] Zhan Li, Qingyu Peng, Bir Bhanu, Qingfeng Zhang, and Haifeng He. Super resolution for astronomical observations. *Astrophysics and Space Science*, 363(5):1–15, 2018.

[20] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[21] Andreas Lugmayr, Martin Danelljan, Radu Timofte, et al. Ntire 2021 learning the super-resolution space challenge: Methods and results. *CVPR Workshops*, 2021.

[22] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflow: Learning the super-resolution space with normalizing flow. In *European Conference on Computer Vision*, pages 715–732. Springer, 2020.

[23] Junhyug Noh, Wonho Bae, Wonhee Lee, Jinhwan Seo, and Gunhee Kim. Better to follow, follow to be better: Towards precise supervision of feature super-resolution for small object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9725–9734, 2019.

[24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[25] Klaus G Puschmann and Franz Kneer. On super-resolution in astronomical imaging. *Astronomy & Astrophysics*, 436(1):373–378, 2005.

[26] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[27] Nathanael Carraz Rakotonirina and Andry Rasoanaivo. Esrgan+: Further improving enhanced super-resolution generative adversarial network. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020.

[28] Wenzhe Shi, Jose Caballero, Christian Ledig, Xiahai Zhuang, Wenjia Bai, Kanwal Bhatia, Antonio M Simoes Monteiro de Marvao, Tim Dawes, Declan O'Regan, and Daniel Rueckert. Cardiac image super-resolution with global correspondence using multi-atlas patchmatch. In *International conference on medical image computing and computer-assisted intervention*, pages 9–16. Springer, 2013.

[29] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[30] Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *arXiv preprint arXiv:1912.00042*, 2019.

[31] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018.

[32] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 286–301, 2018.

[33] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018.

[34] Wilman WW Zou and Pong C Yuen. Very low resolution face recognition problem. *IEEE Transactions on image processing*, 21(1):327–340, 2011.