Computer Vision project

Learning the Super Resolution Space

D'Urso Mattia

June 24, 2022

Contents

1 Introduction **3**

2 State-of-the-art architectures for SR **5**
- 2.1 Previous approaches 5
- 2.2 Flow models intuition 5
- 2.3 SRFlow: Learning the Super-Resolution Space with Normalizing Flow 5
- 2.4 SRFlow-DA: Super-Resolution Using Normalizing Flow with Deep Convolutional Block 7
- 2.5 Noise Conditional Flow Model for Learning the Super-Resolution Space 7
- 2.6 Results comparison 9

3 Comparison and variations **10**
- 3.1 Variations 10
- 3.2 Comparison 10

4 Conclusions **11**

1 Introduction

Within the Computer Science area, one of the topics of high interest has always been Artificial Intelligence (AI). We define AI as the simulation of human intelligence computed by machines. Specific applications of AI include natural language processing, speech recognition, and computer vision (CV). The latter is one of the most-studied topics, we can define it as a subset of AI that deals in teaching machines how to understand images.

The first problem CV tried to solve was to make a robot "see" the world by connecting a video camera to a computer, around the 1960s at MIT. Today the field made several steps further, and there are many more problems that the CV aims to solve. Among these, we find image classification, the task of deciding whether class images belong, object detection, which is finding and classifying objects within an image, and semantic segmentation, which consists of grouping the parts of an image that belong to the same class of objects. With the advancement of technology, new challenges presented themselves, like the problem of the upscaling image. It is about reconstructing a meaningful high-resolution image (HR) from a low-resolution image (LR). This task is called Image Super-Resolution (SR). SR is a fundamentally ill-posed problem. In fact, for a given LR image, there exist infinitely many compatible HR predictions. Nevertheless, SR has several real-world applications such as surveillance (e.g., face recognition in upscaled images) and media sharing (e.g., sending LR images and upscaling on the fly). This issue is of interest also to large companies like NVIDIA that developed the deep learning super sampling (DLSS) [2] to use it on its GPUs when rendering video game frames. The idea is to render the game at LR (e.g., 1920×1080) and then use DLSS to upscale images to HR (e.g., 3840×2160). In this way, it is possible to play HR games with less computational power.

Regarding the SR problem, there are also some public competitions, such as the "Learning the Super-Resolution Space" challenge in 2021, held as part of the $6thth$ edition of the New Trends in Image Restoration and Enhancement (NTIRE) workshop, in conjunction with the 2021 Conference on Computer Vision and Pattern Recognition (CVPR). The goal of the "Learning the Super-Resolution Space" challenge is to develop an SR method that can upscale downscaled images sampled from the space of plausible HR images. More formally, given an HR image IMG sampled from a space of images S_{IMG} and downscaled to an LR image img, the aim is to find a method f that

$$IMG' = f(img)$$

where IMG' is an HR image and $IMG' \approx IMG$. The challenge contains two tracks, targeting 4X and 8X super-resolution respectively. The running methods shall achieve the highest possible photo-realism and be consistent with the input low-resolution image. The methods are tested with the following metrics [5].

- **Diversity score**: This metric aims to measure how well the SR image of ground truth is represented in the predicted space. From the predicted image IMG' and the original image IMG, k random patches are sampled $y_k\ in \mathbb{R}^{N\times N\times 3}$ and the similarity is calculated with a distance metric d. To obtain the significant diversity that the samples represent, testers calculate by how much the minimum distance from the reference patch decreases when M samples are used,

$$S_M = \frac{1}{\overline{d}_M}(\overline{d}_M - \frac{1}{K}\sum_{k=1}^{K} min\{d(y_k, \hat{y}_k^i)\}_{i=1}^{M}) \tag{1}$$

where

$$\overline{d}_M = min\{\frac{1}{K}\sum_{k=1}^{K} d(y_k, \hat{y}_k^i)\}_{i=1}^{M} \tag{2}$$

Notice that $S_M \in [0,1]$ where $S_M = 0$ means no diversity (deterministic predictions) and $S_M = 1$ means that the ground-truth HR image was exactly captured by one of the generated samples.

- **LR-PSNR**: This metric aims to measure the amount of information retained in the SR image compared to the LR image measuring the LR-PSNR. It is calculated as the PSNR between the input LR image and the predicted sample resized with the given bicubic kernel. The goal of this challenge is to obtain an LR-PSNR of at least 45dB.
- **Mean Opinion Rank**: This metric aims to assess the degree of photorealism of an image. This task is performed by humans on three different 80×80 cutouts of the 100 images in the test set. Each task is performed by five different users. Finally, the user is asked to rank the cutouts according to their opinion.

The participating methods are ranked according to each metric. These ranks then are combined into a final score. The first method wins the challenge.

After AlexNet won the ImageNet Large Scale Visual Recognition Challenge in 2012, deep learning (DL) methods became increasingly popular in the CV field because they performed significantly better than methods used up to that point in a wide set of tasks. Following this trend, during NTIRE 2021 the most common approach was the use of DL methods. In particular, Flow models, Generative Adversarial Networks (GANs), Variational AutoEncoders (VAEs), and GANs trained with implicit maximum likelihood estimation (IMLE) were used. In this report we discuss the first methods.

In section 2 we describe the State-of-the-art architectures for SR. In section 3 we run some tests with the noise conditional Flow model. Finally, in section 4 se we draw the conclusions.

2 State-of-the-art architectures for SR

In this section we describe the benefits of the Flow models over the previous approaches, then we explain the Flow model and the noise conditional Flow model for SR architectures.

2.1 Previous approaches

Early DL approaches trained feed-forward architectures using L_1[1] or L_2[2] losses. The problem with these methods is they reduce high frequencies resulting in blurred images because of the average nature of the aforementioned loss functions. To address this problem more recent approaches introduced adversarial training and perceptual losses functions (work by summing all the squared errors between all the pixels and taking the mean). Adversarial methods obtain better results while they can compute a single prediction for an image. They also use multiple loss functions during the training phase with careful parameter tuning.

2.2 Flow models intuition

Flow models are particular architectures in which data flows in both directions. They aim to learn the distribution of HR images $p_{y|x}(y|x,\theta)$ (where x is the LR image, y is the HR image and θ are the parameters of the function f), and not only an injective correspondence $x \to y$. In this way, the model is able to sample multiple HR images from the learned plausible distribution given one LR image. In addition, Flow models are trained with a single loss function that is negative log-likelihood[3], which simplifies training. The core idea of normalizing flow is to parametrize the distribution $p_{y|x}$ using a neural network f_θ. The model is also required to be invertible w.r.t y, which means once computed the latent encoding $z = f_\theta(y;x)$ we can reconstruct y as $y = f_\theta^{-1}(z;x)$. ADD LOSS EXPLANATION

2.3 SRFlow: Learning the Super-Resolution Space with Normalizing Flow

The Super-Resolution using Normalizing Flow model (SRFlow) [8] consists of the invertible flow network f_θ and the LR encoder g_θ. The flow network is organized into L levels, each operating at a resolution of $\frac{H}{2^l} \times \frac{W}{2^l}$, where $l \in \{1,...,L\}$ is the level number and $H \times W$ is the HR resolution (e.g., let an input image be $3 \times 512 \times 512$, at level 2 will be $48 \times 128 \times 128$, at level 4 will be $768 \times 32 \times 32$, and so on.). Each level itself contains K number of flow steps. To achieve tractable expressions the neural network f_θ is decomposed into a sequence of N layers $h^{n+1} = f_\theta^n(h^n; g_\theta(x))$, where $h^0 = y$ and $h^N = z$. These layers can be easly inverted and derived.

The LR encoder g_θ is a Residual-in-Residual Dense Blocks (RRDB) [3] without any batch normalization layers and the final upscaling layers. To capture a richer representation of the LR image at multiple levels, the researchers additionally concatenate the activations after each RRDB block to form the final output of g_θ.

f_θ is organized as follows.

- The **squeeze** layer: this layer provides an invertible means to halving the resolution by reshaping each spatial 2×2 neighborhood into the channel dimension (e.g., $3 \times 64 \times 64$ into $12 \times 32 \times 32$). This operation is done to capture correlations and structures over larger distances. Figure 1 graphically shows the operation.

- The **Transition Step**: this block normalizes and permutes the input according to a learned method.

  - **Actnorm**: This provides a channel-wise normalization through a learned scaling and bias. This layer is ketp in its standard un-conditional form [1].

[1] $L_1(y,\hat{y}) = \sum_{i=1}^{n} |y - \hat{y}|$
[2] $L_2(y,\hat{y}) = \sum_{i=1}^{n} (y - \hat{y})^2$
[3] $\mathcal{L}(\theta; x, y) = -\log p_{y|x}(y|x,\theta)$

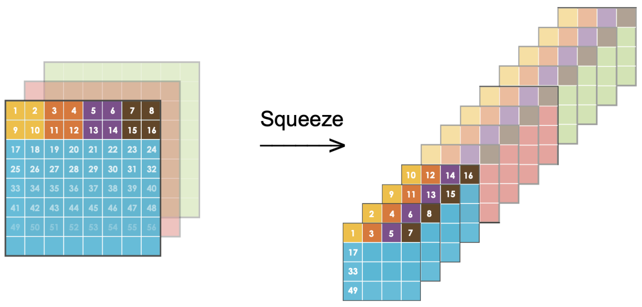


Figure 1: Visualization of the Squeeze operation.

- Invertible **1 × 1 Convolution**: this layer is a 1×1 convolution with equal number of input and output channels, which is a generalization of a permutation operation [1]. According to [6], the ordering of the dimensions can be critical to the training of the model.

- The **Conditional Flow Step**: this block passes the input in another Actnorm layer and 1×1 convolutional layer. Then, add the conditional information in two steps.

  - **Affine Injector**: this layer directly affects all channels and spatial locations in the activation map h^n. This is achieved by predicting an element-wise scaling and bias using only the conditional encoding $u = g_\theta(x)$,

  $$h^{n+1} = \exp\left(f_{\theta,s}(u)\right) \cdot h^n + f_{\theta,b}(u) \tag{3}$$

  where $f_{\theta,s}$ and $f_{\theta,b}$ can be any network, SRFlow uses a block that consists of 3×3 convolution, actnorm, ReLU, 1×1 convolution, actnorm, ReLU, and 3×3 conv. The exp power is used to have positive values in easy differentible way.

  - **Conditional Affine Coupling**: this layer allows applying complex unconstrained conditional learned functions that act on the normalizing flow, without harming its invertibility. The layer splits on half the input $\text{split}(h^n) = h_A^n, h_B^n$ along the channel dimension. Then the following transformation on the bypassed half h_A^n and conditional features u is computed

  $$h^{n+1} = \left\{ h_A^{n+1} = h_A^n,\ h_B^{n+1} = \exp\left(f_{\theta,s}(h_A^{n+1}; u)\right) \cdot h_B^n + f_{\theta,b}(h_A^{n+1}; u) \right. \tag{4}$$

  The network architectures of $f_{\theta,s}$ and $f_{\theta,b}$ are similar to those of the Affine Injector, described above. The only difference is that u is resized to the resolution of h_A^n, then it is concatenated to h_A^{n+1} along the channel dimension. Note that h^{n+1} is a couple.

- **Split** layer: this layer splits the given input in two halfs along the channel dimension.

Figure 2 graphically shows a representation of the model.

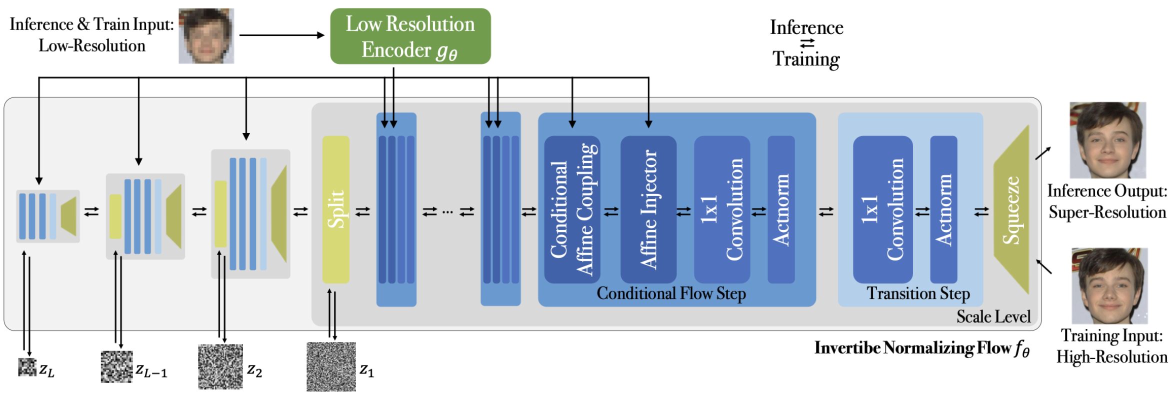
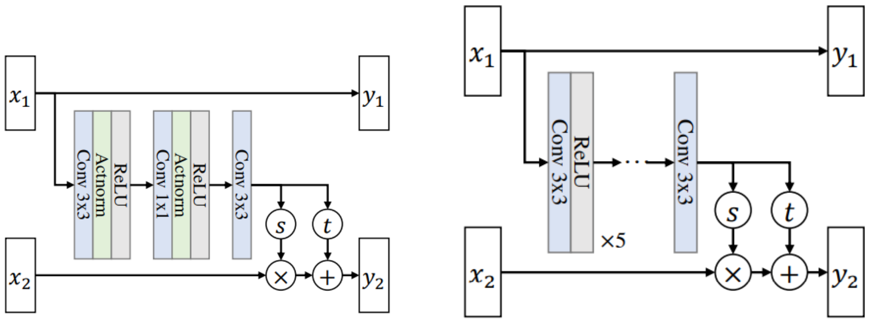

Inference & Train Input: Low-Resolution
Low Resolution Encoder g_θ
Inference ⇄ Training
Split
Conditional Affine Coupling
Affine Injector
1x1 Convolution
Actnorm
Conditional Flow Step
1x1 Convolution
Actnorm
Transition Step
Squeeze
Scale Level
Invertibe Normalizing Flow f_θ
Inference Output: Super-Resolution
Training Input: High-Resolution
z_L
z_{L-1}
z_2
z_1

Figure 2: Super-Resolution using Normalizing Flow's architecture.

2.4 SRFlow-DA: Super-Resolution Using Normalizing Flow with Deep Convolutional Block

Super-Resolution using Normalizing Flow with Deep convolutional block in the Affine couplings (SRFlow-DA) [9] model is an architecture based on SRFlow that increases the SR quality while reducing the model size and the training time. The idea is to improve the perfomance enlarging receptive field[4] size in the single flow step. The SRFlow model has an equivalent RF of 5×5 while the SRFlow-DA model has a RF of 13×13, plus the number of flow steps has been reduced from 16 to 6. These two changes reduced the number of parameters from 22.8M (SRFlow) to 8.7M (SRFlow-DA) improving training and inference time. To achive those results, the researchers stacked 3×3 convolutional layers followed by ReLU activation except for the last convolutional layer. s and t are emprically estimated coefficents that improves the perfomances. Figure 3 highligths the differences.



(a) Affine coupling in SRFlow. (b) Affine coupling in SRFlow-DA.

Figure 3: Comparison between the Affine Coupling layers of the two architectures.

2.5 Noise Conditional Flow Model for Learning the Super-Resolution Space

Noise Conditional Flow model for Super-Resolution (NCSR) [4] is an architecture based on SRFlow that improves the diversity of the predicted images by adding noise. In particular, the researchers add noise to the training data and through a new layer called Noise Conditional Layer. NCSR aims to solve also the dimension mismatch problem[5] typical of Flow models. The architectural

[4]A Receptive Field (RF) is defined as the size of the region in the input that produces the output feature. [10] shows applying a stack of smaller filters is equivalent and more efficent than a bigger one

[5]This problem concerns the inability to properly train the model if the size of the data distribution does not match that of the underlying target distribution. [7]

difference can be found in the Conditional Flow step. NCSR adds the Noise Conditional Layer in 4^{th} position obtaining a block with five elements, as shown in figure 4.

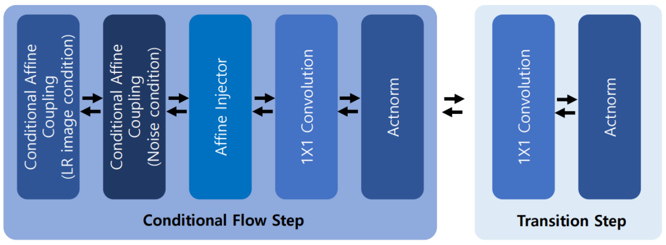


Figure 4: The Conditional Flow block.

Adding *noise* to the input data x results in a variation of the distribution of the latent data z during the inference process. To address that the *noise* used for the input x is resized and added to the LR image y as follows.

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

where, x is an HR image, f is the invertible model that maps z to x and y is a LR image.

The second improvement is the introduction of the Noise Conditional Layer, which is a Conditional Affine Coupling layer that uses *noise* instead of u. The idea is to inform noise to the model structure to avoid artifacts caused by the noise injection on x. The Noise Affine Conditional layer is designed as follows.

$$h^{n+1} = \Big\{ h_A^{n+1} = h_A^n,\ h_B^{n+1} = \exp\left(f_{\theta,s}(h_A^{n+1}; noise)\right) \cdot h_B^n + f_{\theta,b}(h_A^{n+1}; noise) \tag{5}$$

The third contribution is about the dimension mismatch problem. NCSR uses the same approach of [7] to overcome it by estimating a perturbed data distribution that is conditioned on noise parameters. The key is to add noise that is obtained from a randomly selected distribution and to use these distribution parameters as conditions. In particular random value c is obtained from uniform distribution $U(0, M)$, where M is 0.1, as [7] did. Next, set the to noise distribution $N(0, \Sigma)$, where $\Sigma = c^2 I$. Then, noise is sampled from $N(0, \Sigma)$.

Figure 2 graphically shows a representation of the model.

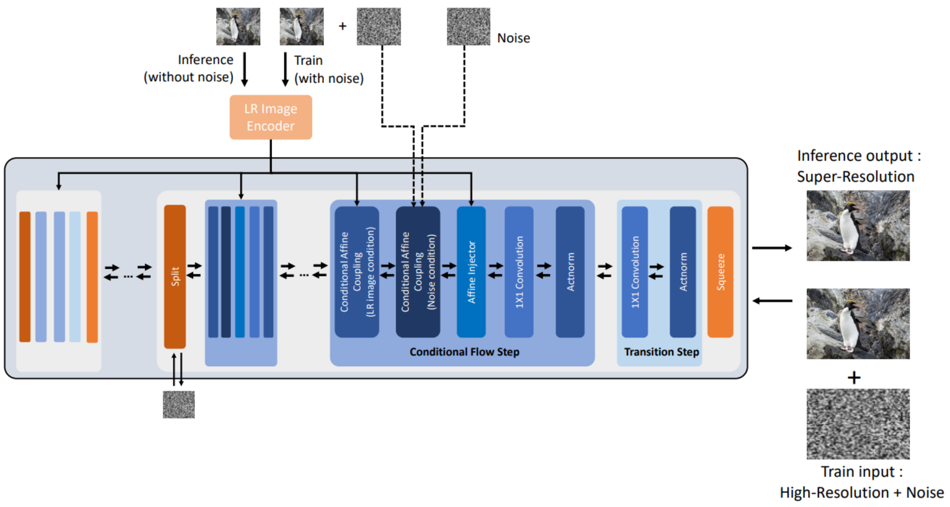

Noise
Inference (without noise)
Train (with noise)
LR Image Encoder
Inference output : Super-Resolution
Split
Conditional Affine Coupling (LR image condition)
Conditional Affine Coupling (Noise condition)
Affine Injector
1X1 Convolution
Actnorm
Conditional Flow Step
1X1 Convolution
Actnorm
Squeeze
Transition Step
Train input : High-Resolution + Noise

Figure 5: Noise Conditional Flow for Super-Resolution's architecture.

2.6 Results comparison

The table 1 and table 2 show the 4× and 8× results of the models on the 100 validation images of the DIV2K dataset. The arrows near the metric's name indicate the direction for a better result. Best results are highlighted in bold only if the value is reported for all the models. The * means the value is not reported in the original model's paper but in another paper for comparison purposes. Inference time is measured by [9] for generating an output image size of 1920×1080. Note that the training times and the inference times are measured on NVIDIA GeForce RTX 2080 TI.

Model	Diversity$\uparrow$	LPIPS$\downarrow$	LR-PSNR$\uparrow$	MOR	Params	Train/Infer. time
SRFlow	25.24*	0.120	50.64	22.8M*	120h*/1.98s*	
SRFlow-DA	23.55	0.121	**50.88**	8.7M	33h/1.18s	
NCSR	**26.72**	**0.119**	50.75	-	-	

Table 1: 4× results on DIV2K validation set.

Model	Diversity$\uparrow$	LPIPS$\downarrow$	LR-PSNR$\uparrow$	MOR	Params	Train/Infer. time
SRFlow	25.28*	0.272	50.09	34.1M*	120h* / 1.97s*	
SRFlow-DA	23.45	**0.261**	**50.91**	13.3M	47h / 1.01s	
NCSR	**26.80**	0.278	44.55	-	-	

Table 2: 8× results on DIV2K validation set.

ADD COMMENT

3 Comparison and variations

3.1 Variations

3.2 Comparison

4 Conclusions

References

[1] Kingma, D.P., Dhariwal, P. *Glow: Generative flow with invertible 1x1 convolutions*, Advances in Neural Information Processing Systems 2018.

[2] Deep Learning Super Sampling-Technology (DLSS) | NVIDIA
`https://www.nvidia.com/de-at/geforce/technologies/dlss/`,
last access 20 June 2022.

[3] Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Loy, C.C., Qiao, Y., Tang, X.: *Esrgan: Enhanced super-resolution generative adversarial networks*, ECCV 2018

[4] Younggeun Kim, Donghee Son
Noise Conditional Flow Model for Learning the Super-Resolution Space, NTIR Challenge 2021.

[5] Andreas Lugmayr, Martin Danelljan, Radu Timofte
NTIRE 2021 Learning the Super-Resolution Space Challenge,
Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2021.

[6] Oriol Vinyals, Samy Bengio, Manjunath Kudlur *Order Matters: Sequence to sequence for sets*, ICLR 2015.

[7] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, Nam Soo Kim
SoftFlow: Probabilistic Framework for Normalizing Flow on Manifolds,
Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2021.

[8] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, Radu Timofte
SRFlow: Learning the Super-Resolution Space with Normalizing Flow,
Spotlight at ECCV 2020.

[9] Younghyun Jo, Sejong Yang, Seon Joo Kim
SRFlow-DA: Super-Resolution Using Normalizing Flow with Deep Convolutional Block,
Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) 2021.

[10] Wenjie Luo, Yujia Li, Raquel Urtasun, Richard Zemel
Understanding the Effective Receptive Field in Deep Convolutional Neural Networks,
Conference on Neural Information Processing Systems (NIPS) 2016.