Normalizing Flows(NF): Application on SR

Seobin Park

Contents

1. Analysis on NF losses

2. SRFlow
    1. Motivations
    2. Conditional NF for Super-Resolution
    3. Network Architectures
    4. Experimental Results

Analysis on NF losses

Analysis on NF losses

- We should recall that most of the losses that we try to optimize start with **Maximum Likelihood Estimation (MLE)**:

$$\mathcal{L}(\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} -\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$$

$$x \sim p_\theta(x)$$

Probability model of target data

Target data that we want to model (e.g. Natural image)

Analysis on NF losses

- We should recall that most of the losses that we try to optimize start with Maximum Likelihood Estimation (MLE):

$$\mathcal{L}(\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} -\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$$

- But in most situations, it is hard to **compute exact p**, and **sample x** from p.

Analysis on NF losses

- So generative models try to learn a mapping f from x to z (or z to x), where **z is a tractible latent variable** (e.g. Gaussian distribution)

$$\mathcal{L}(\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} -\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$$

$$z \sim p_\theta(z)$$

$$x = f_\theta(z)$$

Analysis on NF losses

- **NF** tries to learn a change of variable f, which maps z to x.

$$z \sim p_\theta(z)$$
$$x = f_\theta(z)$$

$$\mathcal{L}(\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} -\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$$

- Plugging this transformation (f) to the MLE above yields **two loss term**:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \underline{\log p_{\boldsymbol{\theta}}(\mathbf{z})} + \underline{\log \left| \det(d\mathbf{z}/d\mathbf{x}) \right|}$$

Analysis on NF losses

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log p_{\boldsymbol{\theta}}(\mathbf{z}) + \log \left| \det(d\mathbf{z}/d\mathbf{x}) \right|$$

$$= \log\left(\exp\left(-\frac{z^2}{\sigma^2}\right)\right) + C = -f_\theta(x)^2 + C$$

$\Rightarrow$ Forcing z to be in $N(0, \sigma^2)$

$\Rightarrow$ Forcing $f(x) = z$ and $f^{-1}(z) = x$

- But optimzing **only this term** will result $f(x) \approx 0$ for all x.

Analysis on NF losses

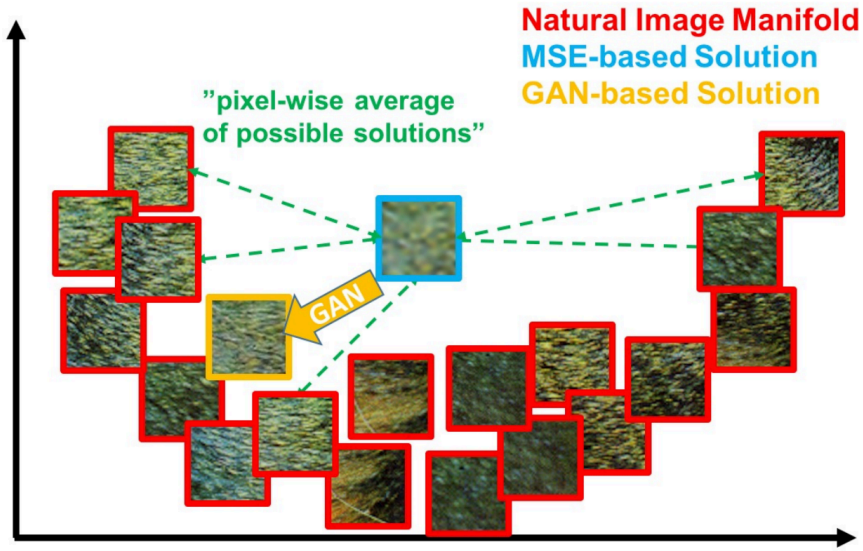
- Recall that the determinant term is a scaling factor

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log p_{\boldsymbol{\theta}}(\mathbf{z}) + \log \left| \det(d\mathbf{z}/d\mathbf{x}) \right|$$

$\Rightarrow$ **Prevents z to collapse to 0**

SRFlow

Motivations



SRGAN (Ledig et al)

Problem with SRGAN:

- Chooses **only one image** in the natural image manifold.
- **Unstable minimax** optimization process.

SRFlow:

⇒ Can sample **diverse images** given an LR image.

⇒ **No minimax** optimization, trained with a single loss.

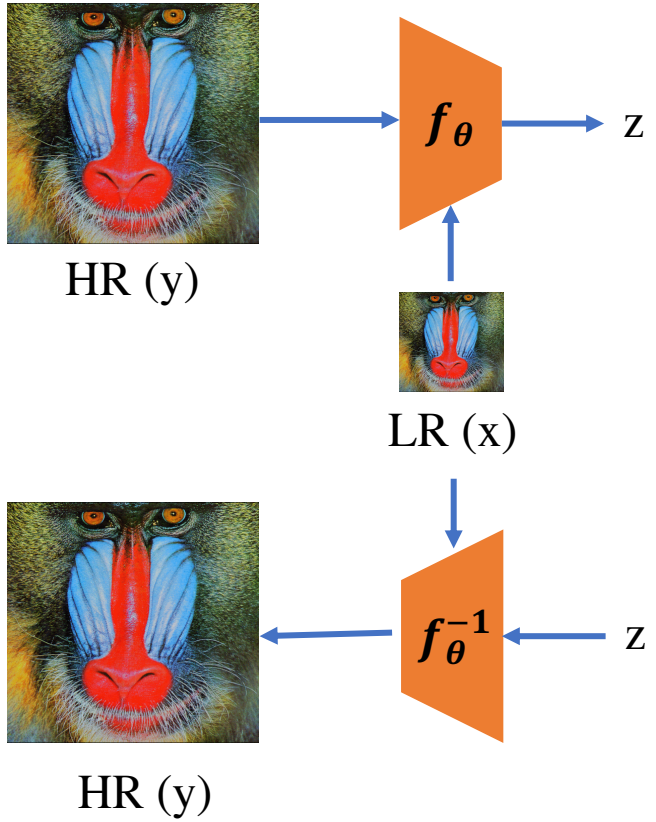
Conditional NF for Super-Resolution

- In conditional settings, we want to map HR, LR pair (y, x) to a tractible latent variable z with a function f:

$$z = f_\theta(y; x)$$

- f only needs to be invertible w.r.t. y and z.

$$y = f_\theta^{-1}(z; x)$$



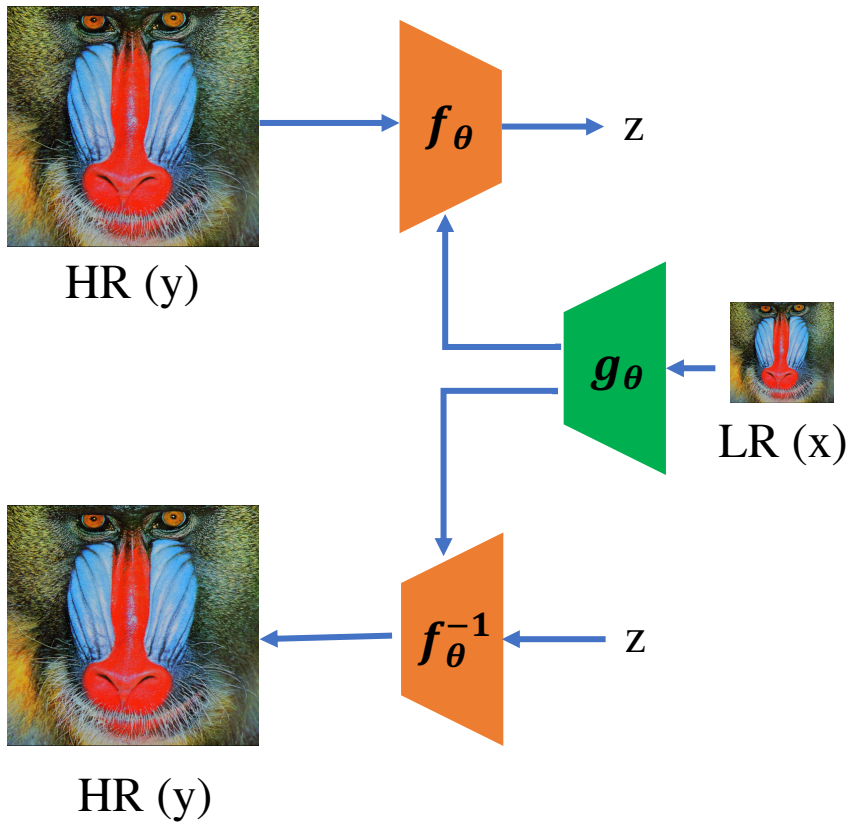
https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Conditional NF for Super-Resolution

- f cannot go deeper like conventional deep CNNs because of invertibility restriction. So SRFlow introduces an **LR encoder g**, which doesn't need to be invertible.

$$z = f_\theta\left(y; g_\theta(x)\right)$$

$$y = f_\theta^{-1}\left(z; g_\theta(x)\right)$$



https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Conditional NF for Super-Resolution

- Optimize the following following NLL loss w.r.t. θ :

$$\mathcal{L}(\theta; x, y) = -\log p_{y|x}(y|x, \theta)$$

$$= -\log p_z\Big(\underbrace{f_\theta\big(y; g_\theta(x)\big)}_{=z}\Big) - log\left|det\frac{\partial f_\theta}{\partial y}(y; g_\theta(x))\right|$$

https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Conditional NF for Super-Resolution

- This paper assumes $p_z \sim N(0, I)$ (variance=1) :

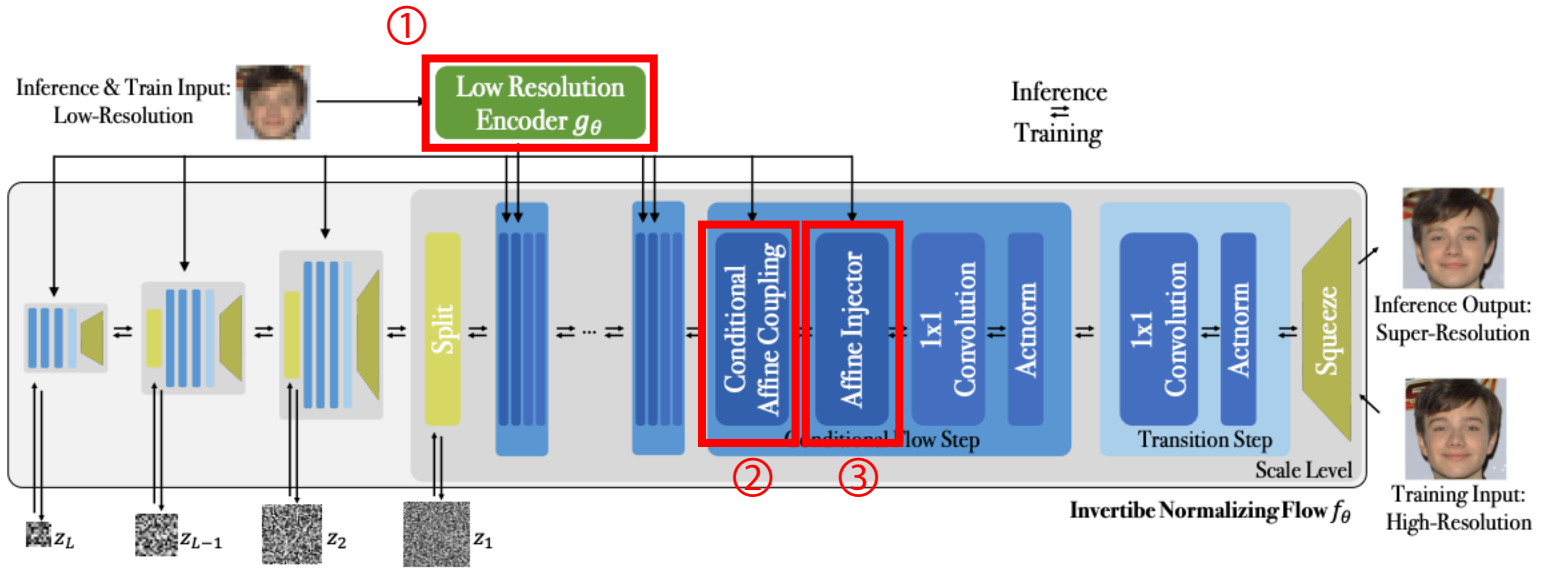
$$p_z\left(f_\theta\left(y; g_\theta(x)\right)\right) = C \cdot \exp(-f_\theta\left(y; g_\theta(x)\right)^2)$$

- So the final loss looks like:

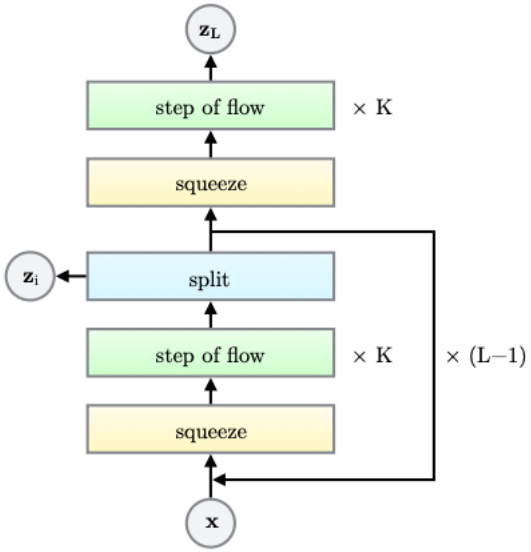
$$\mathcal{L}(\theta; x, y) = \underline{f_\theta\left(y; g_\theta(x)\right)^2} - log\left|det\frac{\partial f_\theta}{\partial y}(y; g_\theta(x))\right|$$

https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Network Architectures



SRFlow (Lugmayr et al.)



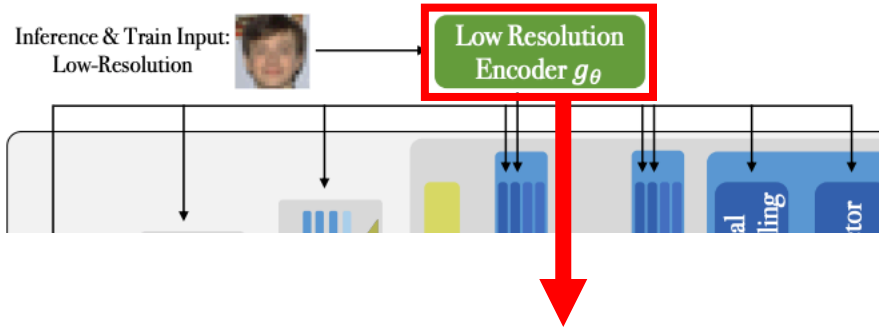
Glow (Kingma et al.)

- Same as Glow (Kingma et al.), except for the conditional terms (①, ②, ③)

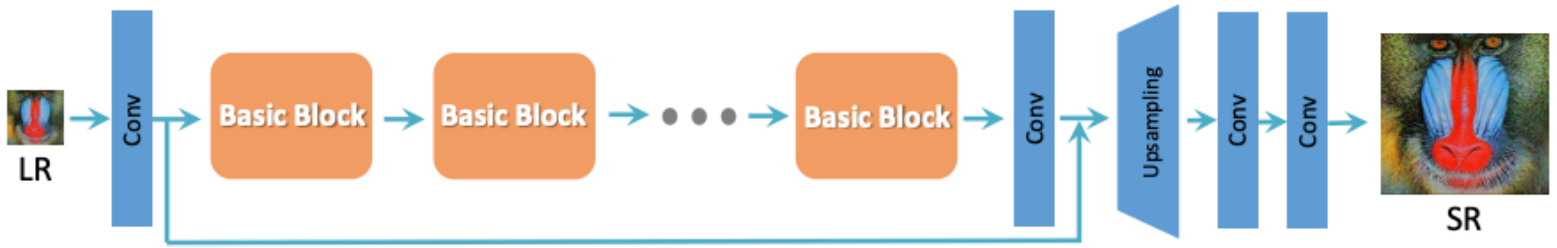
https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

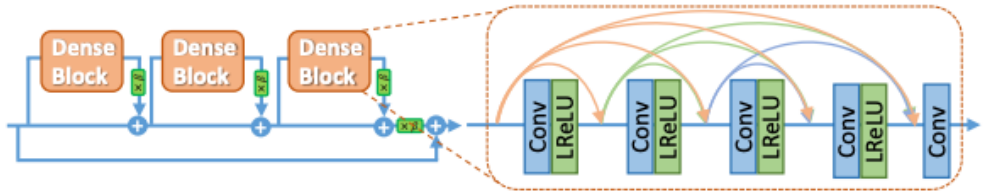
Network Architectures: g_θ

- ① RRDB



- Using deep residual network (RRDB) used in ESRGAN (Wang et al.)
- This network (g) is **pretrained** for conventional SR task!

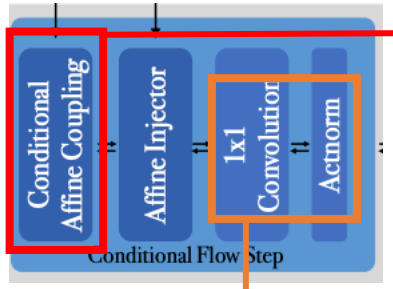




https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Network Architectures: f_θ

- ② Conditional Affine Coupling



Similar with Affine coupling layer, but including the conditional term $u(= g_\theta(x))$

$$\mathbf{h}_A^{n+1} = \mathbf{h}_A^n ,$$

$$\mathbf{h}_B^{n+1} = \exp\left(f_{\boldsymbol{\theta},\mathrm{s}}^n(\mathbf{h}_A^n; \mathbf{u})\right) \cdot \mathbf{h}_B^n + f_{\boldsymbol{\theta},\mathrm{b}}^n(\mathbf{h}_A^n; \mathbf{u})$$

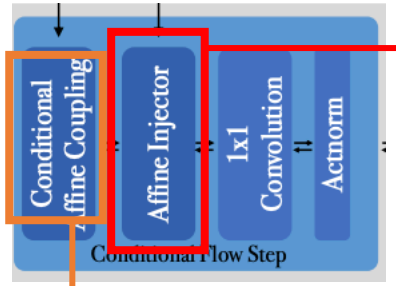
LR information

Same as Glow (Kingma et al.)

https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Network Architectures: f_θ

- ③ Affine Injector



Similar with Affine coupling layer, but including the conditional term $u(= g_\theta(x))$

$$\mathbf{h}^{n+1} = \exp\left(f^n_{\boldsymbol{\theta},\mathrm{s}}(\mathbf{u})\right) \cdot \mathbf{h}^n + f_{\boldsymbol{\theta},\mathrm{b}}(\mathbf{u})$$

LR information

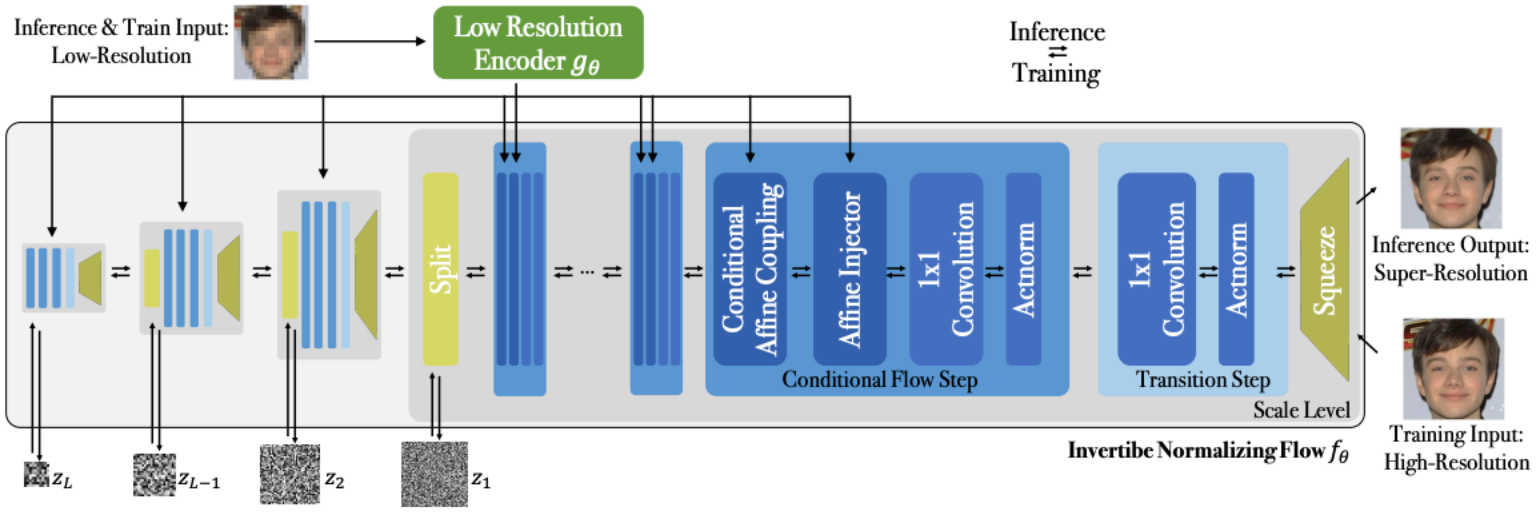
$$\mathbf{h}_A^{n+1} = \mathbf{h}_A^n \,,$$

$$\mathbf{h}_B^{n+1} = \exp\left(f^n_{\boldsymbol{\theta},\mathrm{s}}(\mathbf{h}_A^n; \mathbf{u})\right) \cdot \mathbf{h}_B^n + f^n_{\boldsymbol{\theta},\mathrm{b}}(\mathbf{h}_A^n; \mathbf{u})$$

https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Network Architectures: f_θ

- Details



- 16 Flow steps (K=16) for each Scale Level
- L=3 for SR factor x4, L=4 for SR factor x8
- 5 days of training with V100 GPU

https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Experimental Results

- Test time inference

- At test time, the HR image y is computed as follows given LR image x and a latent random variable z:

$$y = f_\theta^{-1}\left(z; g_\theta(x)\right)$$

$$z \sim N(0, \tau)$$

- It is commonly observed that best results are achieved with **slightly lower variance** $\tau(= 0.8)$.

https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf

Experimental Results

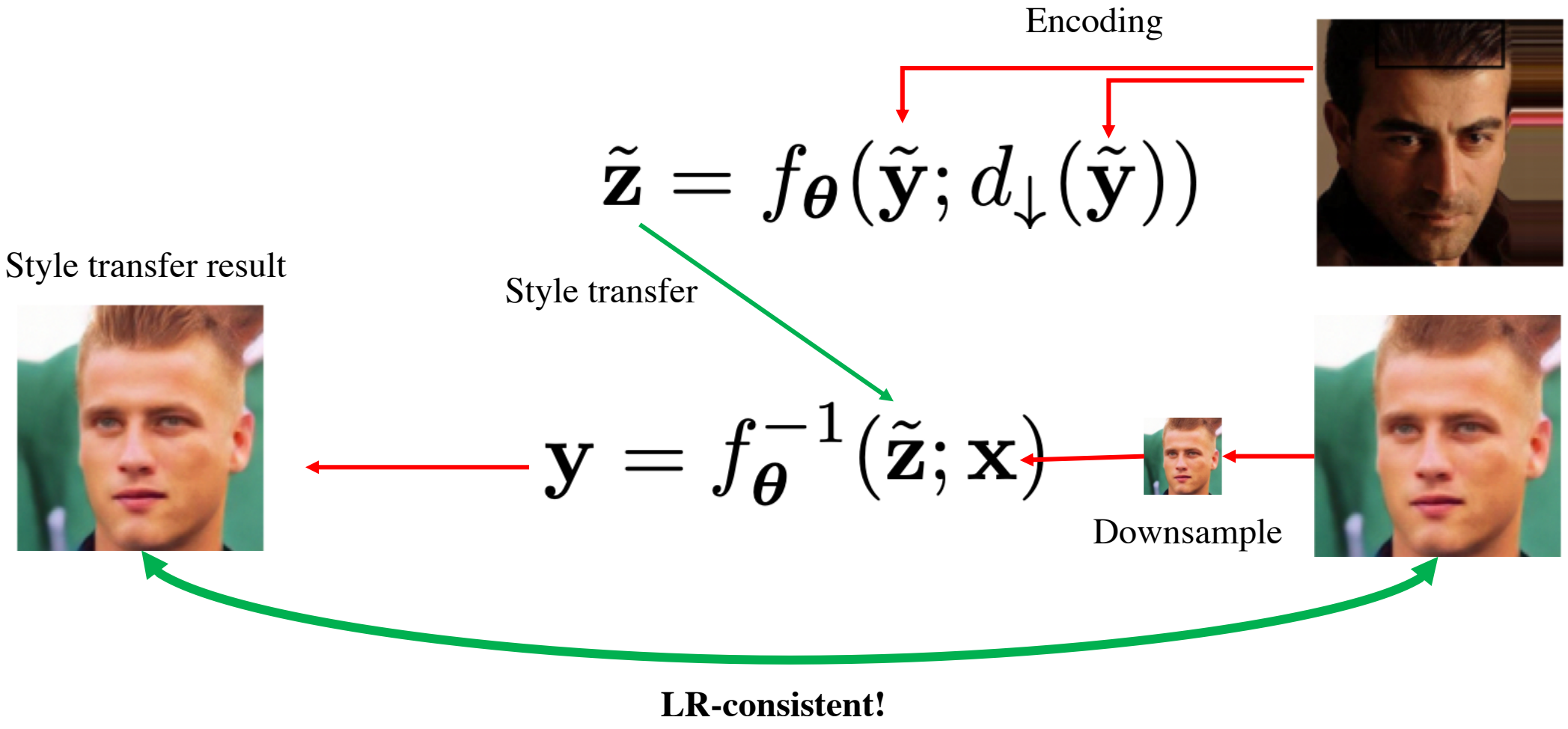
- Visual results x8

LR | RRDB [46] | ESRGAN [46] | ProgFSR [18] | **SRFlow** $\tau = 0.8$ | Ground Truth

	LR	↑PSNR	↑SSIM	↓LPIPS	↑LR-PSNR	↓NIQE	↓BRISQUE	↓PIQUE	↑Diversity σ
Bicubic	Bicubic	23.15	0.63	0.517	35.19	7.82	58.6	99.97	0
	RRDB [46]	26.59	0.77	0.230	48.22	6.02	49.7	86.5	0
	ESRGAN [46]	22.88	0.63	0.120	34.04	3.46	23.7	32.4	0
	SRFlow $\tau = 0.8$	25.24	0.71	0.110	50.85	4.20	23.2	24.0	5.21

Experimental Results

- LR-consistent style transfer



Experimental Results

- LR-consistent style transfer

Fig. 4. Latent space transfer from the region marked by the box to the target image. (8×)

Experimental Results

- Image content transfer

Fig. 5. Image content transfer for an existing HR image (top) and an SR prediction (bottom). Content from the source is applied directly to the target. By applying latent space normalization in our SRFlow, the content is integrated and harmonized.

Experimental Results

- Image content transfer

Target $\mathbf{y}$

f_θ

$\{z_i\}_{i=1\sim N}$

$$\tilde{\mu} = \frac{1}{N}\sum_{i=1}^{N} z_i\,, \qquad \tilde{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}(z_i - \tilde{\mu})^2$$

$$\hat{\mu} \sim N\left(0, \frac{\tau}{N}\right), \qquad \hat{\sigma}^2 \sim \Gamma\left(\frac{N-1}{2}, \frac{2\tau}{N-1}\right)$$

Input $\tilde{\mathbf{y}}$

f_θ

$\{\tilde{z}_i\}_{i=1\sim N}$

$$\hat{z} = \frac{\hat{\sigma}}{\tilde{\sigma}}(\tilde{z} - \tilde{\mu}) + \hat{\mu}$$

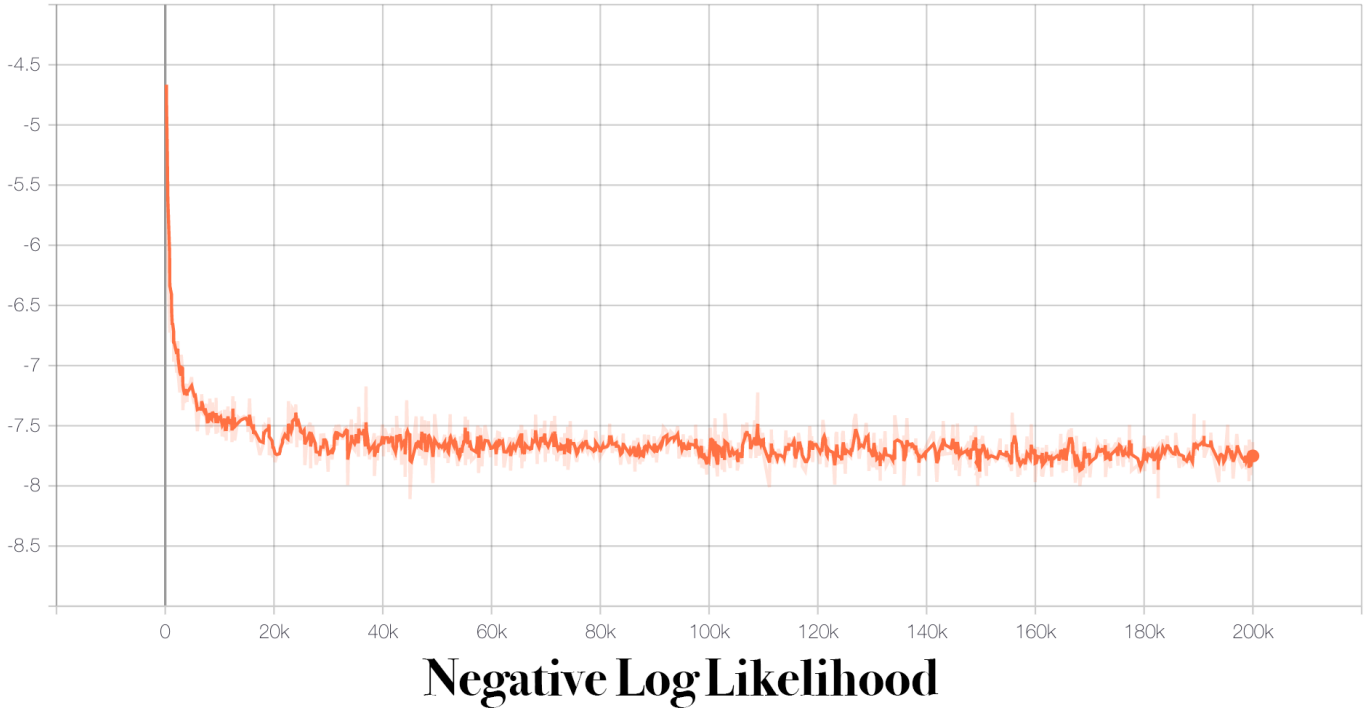
Latent space normalization

Transferred $\hat{\mathbf{y}}$

f_θ^{-1}

Experimental Results

- Loss graph



- SRFlow uses only one single loss, which is stably optimized.

References

- SRFlow: Learning the Super-Resolution Space with Normalizing Flow (Lugmayr et al. https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123500698.pdf)
- https://medium.com/swlh/why-i-stopped-using-gan-eccv2020-d2b20dcfe1d
- Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network (Ledig et al. https://openaccess.thecvf.com/content_cvpr_2017/papers/Ledig_Photo-Realistic_Single_Image_CVPR_2017_paper.pdf)
- ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks (Wang et al. https://openaccess.thecvf.com/content_ECCVW_2018/papers/11133/Wang_ESRGAN_Enhanced_Super-Resolution_Generative_Adversarial_Networks_ECCVW_2018_paper.pdf)
- Glow: Generative Flow with Invertible 1×1 Convolutions (Kingma et al. https://papers.nips.cc/paper/2018/file/d139db6a236200b21cc7f752979132d0-Paper.pdf)