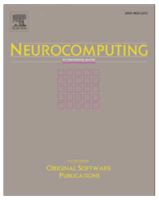
Neurocomputing 479 (2022) 47–59

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



SRDiff: Single image super-resolution with diffusion probabilistic models

Haoying Li [a], Yifan Yang [a], Meng Chang [a], Shiqi Chen [a], Huajun Feng [a,*], Zhihai Xu [a], Qi Li [a], Yueting Chen [a]

[a] State Key Laboratory of Modern Optical Instrumentation, Zhejiang University, Hangzhou, Zhejiang, China

ARTICLE INFO

Article history:
Received 18 May 2021
Revised 15 November 2021
Accepted 14 January 2022
Available online 20 January 2022
Communicated by Zidong Wang

Keywords:
Single image super-resolution
Diffusion probabilistic model
Diverse results
Deep learning

ABSTRACT

Single image super-resolution (SISR) aims to reconstruct high-resolution (HR) images from given low-resolution (LR) images. It is an ill-posed problem because one LR image corresponds to multiple HR images. Recently, learning-based SISR methods have greatly outperformed traditional methods. However, PSNR-oriented, GAN-driven and flow-based methods suffer from over-smoothing, mode collapse and large model footprint issues, respectively. To solve these problems, we propose a novel SISR diffusion probabilistic model (SRDiff), which is the first diffusion-based model for SISR. SRDiff is optimized with a variant of the variational bound on the data likelihood. Through a Markov chain, it can provide diverse and realistic super-resolution (SR) predictions by gradually transforming Gaussian noise into a super-resolution image conditioned on an LR input. In addition, we introduce residual prediction to the whole framework to speed up model convergence. Our extensive experiments on facial and general benchmarks (CelebA and DIV2K datasets) show that (1) SRDiff can generate diverse SR results with rich details and achieve competitive performance against other state-of-the-art methods, when given only one LR input; (2) SRDiff is easy to train with a small footprint(The word “footprint” in this paper represents “model size” (number of model parameters).); (3) SRDiff can perform flexible image manipulation operations, including latent space interpolation and content fusion.

1. Introduction

Over the years, single image super-resolution (SISR) has drawn active attention due to its wide applications in computer vision, such as object recognition [1,2], remote sensing [3], and surveillance monitoring tasks [4,5]. SISR aims to recover high-resolution (HR) images from low-resolution (LR) images, which is an ill-posed problem because multiple HR images may be degenerated to one LR image, as shown in Fig. 1.

To establish the mapping between HR and LR images, many deep learning-based methods have emerged and can be categorized into three main types: PSNR-oriented, GAN-driven and flow-based methods. PSNR-oriented methods [6–13] are trained with simple distributions and assumption-based losses (e.g., Laplacian for L_1 and Gaussian for L_2) and achieve excellent PSNRs. However, these losses tend to drive the SR result to an average of several possible SR predictions, causing **over-smoothed** images with high-frequency information loss. One ground-breaking solution to address the over-smoothing problem is the class of GAN-driven methods [14–17], which combine content losses (e.g., L_1 and L_2) and adversarial losses to obtain sharper SR images with better perceptual quality. However, GAN-driven methods easily fall into **mode collapse**[18,19], which leads to a single generated SR sample without diversity. Additionally, the GAN-based training process does not easily converge and requires an extra discriminator which is not used in inference. Flow-based methods [18] directly settle the ill-posed problem with an invertible encoder, which maps HR images to the flow-space latent variables conditioned on LR inputs. Trained with the negative log-likelihood loss, flow-based methods avoid training instability but suffer from **extremely large footprints and high training costs** due to the strong architectural constraints to maintain the bijection between the latent variables and data.

Recently, the successful adoption of diffusion probabilistic models (diffusion models for short) in image synthesis tasks [20], speech synthesis tasks [21,22], music generation tasks [23] and 3D cloud tasks [24] has witnessed the power of diffusion models in generative tasks. A diffusion model uses a Markov chain to convert the data x_0 in a complex distribution to latent variable x_T in a simple distribution (e.g., Gaussian) by gradually adding noise ϵ during the diffusion process, and the noise ϵ in each diffusion step is predicted to recover the data x_0 through a learned reverse pro-

* Corresponding author.

https://doi.org/10.1016/j.neucom.2022.01.029
0925-2312/© 2022 Elsevier B.V. All rights reserved.

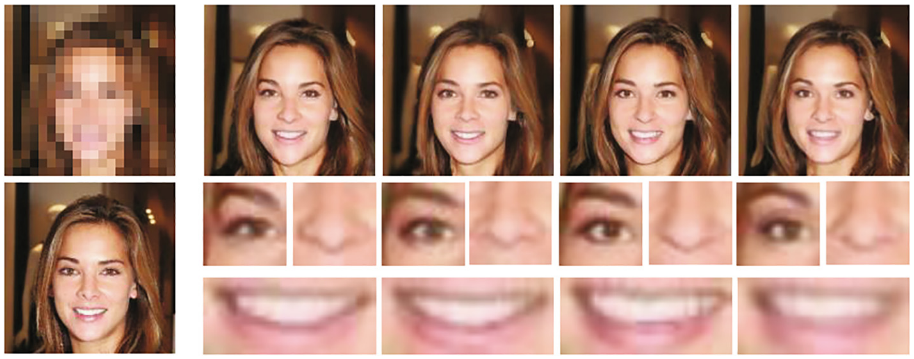


Fig. 1. Random SR (8x) predictions generated by our method. The LR/HR images are shown on the top/bottom left respectively. The right columns contain diverse SR predictions and their facial regions, which are different from each other in terms of expressions and attributes. For example, the first nose looks flat, while the third nose looks straight. The SR predictions generated by our methods all correspond to the given LR image and obtain high image quality.

cess. Diffusion models are trained by optimizing a variant of the variational lower bound, which is efficient and avoids the mode collapse issue encountered by GANs.

In this paper, we propose a novel SISR diffusion probabilistic model (SRDiff) to tackle the over-smoothing, mode collapse and large footprint problems in previous SISR models. Specifically, 1) to extract the image information from an LR image, SRDiff exploits a pre-trained LR encoder to convert the LR image into hidden conditions; 2) to generate the HR image conditioned on the LR image, SRDiff employs a conditional noise predictor to recover x_0 iteratively; 3) to speed up convergence and stabilize training, SRDiff introduces residual prediction by taking the difference between the HR and LR images as the input x_0 in the first diffusion step, making SRDiff focus on restoring high-frequency details. To the best of our knowledge, SRDiff is the first diffusion-based SR model and has several advantages:

- **Diverse and high-quality outputs**: SRDiff converts Gaussian white noise into an SR prediction through a Markov chain, which does not suffer from mode collapse and over-smoothing issue, and can generate diverse and high-quality SR results.
- **Stable and efficient training with small footprints**: Although the data distribution of HR images is hard to estimate, SRDiff utilizes a variant of the variational bound maximization and applies residual prediction. Compared with GAN-driven methods, SRDiff is stably trained with a single loss and does not need any extra modules (e.g., a discriminator, which is only used during training). Compared with flow-based methods, SRDiff has no architectural constraints and thus enjoys benefits such as small footprints and fast training.
- **Flexible image manipulation**: SRDiff can perform flexible image manipulation, including latent space interpolation and content fusion, using both the diffusion process and the reverse process, which shows broad application prospects of the proposed method.

Our extensive experiments on the CelebA [25] and DIV2K [26] datasets show that 1) SRDiff can reconstruct multiple SR results given one LR input and achieve promising results; 2) compared with GAN-based methods, SRDiff is easy and stable to train with only one loss term and no extra discriminator module; 3) compared with SRFlow, SRDiff is fast to train (approximately 28 h on 1 GPU until convergence) and has only 1/4 of the number of parameters; 4) we can manipulate the generated SR images in the latent space to obtain more diverse outputs.

2. Related Works

2.1. Single Image Super-Resolution

Recently, deep learning methods are widely adopted for SISR, and we categorize them into three types: PSNR-oriented, GAN-driven and flow-based methods.

PSNR-oriented methods. The training goal of PSNR-oriented methods is to minimize the L_1 or L_2 loss between the ground truths and the SR images recovered from the input LR images. SRCNN [6] set a precedent of end-to-end mapping between the LR and HR images. [27,28] then applied residual neural network techniques to SISR tasks and found that increasing recursion depth can improve SISR performance. In order to solve the convergence problem brought by Stochastic Gradient Descent (SGD), they introduced recursive-supervision and skip-connection. [7] removed redundant modules in the residual neural network and expanded the model size. [29] proposed a multi-scale residual block (MSRB), which was capable of adaptively detecting and fusing image features at different scales and obtained accurate SR images efficiently. [9] used different modules to reconstruct different information of frequencies, since shallow network structures take the advantage of recovering low-frequency information and deep network structures restore high-frequency information well. [30] proposed the first Cross-Scale Non-Local (CS-NL) attention module for SISR task. They proved that the cross-scale non-local attention is able to obtain better reconstruction by making full use of the abundant repeated structures in the images, and integrating cross-scale feature similarities with local and in-scale non-local priors improves the SISR performance. To improve super-resolution performance, [31] utilized feature distillation with both dense concatenation and skip connection to extract deep and shallow features. They also used attention mechanism to adjust channel-wise features and restore high-frequency feature information.

GAN-driven methods. The pioneering work of GAN-driven methods was SRGAN [16], which used SRResNet and perceptual loss, as well as adversarial loss, to improve the naturalness of the recovered image. ESRGAN [17] further enhanced SRGAN with structure and loss function adjustments. It replaced the Residual Block with the Residual-in-Residual Dense Block (RRDB) as the basic unit of the whole network, and abandoned batch normalization. It also proved that calculating the perceptual loss before the activation layer is beneficial to restoring more detailed information. [14,32] solved the over-smoothing problem through perceptual restrictions [33,34]. [35] proposed a general framework called Generative

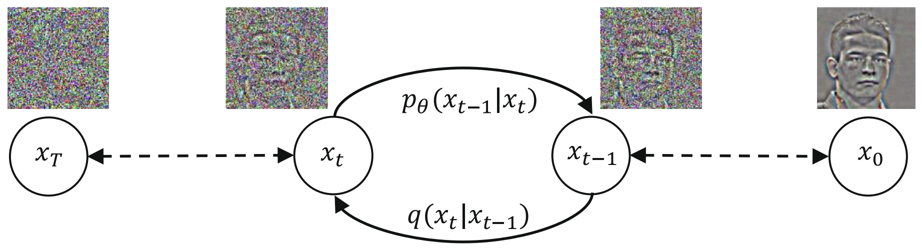


Fig. 2. Overview of the two processes in SRDiff. The diffusion process goes from right to left, and the reverse process goes from left to right. θ in p_θ denotes the learnable components, including the conditional noise predictor and low-resolution encoder in SRDiff.

Collaborative Networks (GCN), which optimized the generator (i.e., the mapping of interest) in the feature space of a network. [15] explored GAN-driven super-resolution framework to face SR task. They addressed constraints to each step of the SR network and proposed a facial attention loss to restore the attributes of the adjacent area to the facial landmarks. Later, DeepSEE [36] leveraged semantic maps to solve the ill-posed SR problem and allowed for controlling over both shape and appearance for face SR tasks. However, GAN-driven methods can easily suffer from the mode collapse problem and only produce samples from a single or few modes of the distribution but ignore other modes, according to the papers [37,38]. Thus, GANs may lead the output samples without diversity. Mode collapse problem can be more severe in conditional generation task [18,19,39]. Because they easily learn to ignore the stochastic input signal when the conditional inputs (e.g., low-resolution image in super-resolution task) exist. Although some techniques are proposed to alleviate mode collapse in GAN (e.g., bayesian inference [40] and explicit regularization loss [41]), they have not been verified in the super-resolution task to achieve state-of-the-art and diverse results.

Flow-based methods. Flow-based methods [42–44] are a kind of generative model that can map the training data to a space where its distribution is factorized, through a series of learned transformations that are invertible. Compared with PSNR-oriented methods and GAN-driven methods, flow-based methods have received relatively less attention in SISR tasks. The first flow-based SISR method was SRFlow [18], which built an invertible neural network to transform a Gaussian distribution into an HR image space instead of modeling a single output. In this way, SRFlow explicitly addresses the “one-to-many” mapping problem (also called ill-posed problem) in super-resolution task[1].

2.2. Diffusion Models

Diffusion models, first presented in [45], are a kind of generative model using a Markov chain to gradually add noise to a data point and convert it to a latent variable through an iterative diffusion process. And through a learned reverse process, the diffusion models gradually transform a latent variable in a simple distribution (e.g., Gaussian) to a data point in the complex distribution. The diffusion model is flexible and suitable for tackling “one-to-many” mapping problem and can generate high-quality results in many tasks, including image synthesis tasks [20], speech synthesis [21,22], music generation tasks [23] and 3D cloud tasks [24]. In image synthesis tasks, [20] proved the capability of generating high-quality samples for the diffusion model. It exploited straightforward log-likelihood evaluation, and the training process used variational inference to explicitly train the Langevin dynamic sampler. In speech synthesis tasks, DiffWave [21] utilized a versatile diffusion model for waveform generation and could generate high-fidelity audios. DiffSinger [22] applied the diffusion model to synthesize high-quality and expressive singing voice to tackle the over-smoothing and unstable training issues in the previous methods. It also introduced a shallow diffusion mechanism to make better use of the prior knowledge learned by the simple loss. In music generation tasks, [23] introduced a technique for training diffusion models on sequential data and applied it to model symbolic music. For 3D point cloud generation, [24] viewed points in point clouds as particles in a thermodynamic system in contact with a heat bath and then introduced a diffusion model to estimate the original point clouds distribution. However, to the best of our knowledge, diffusion models have not yet been used in SISR tasks. In this paper, we propose our impressive work, SRDiff, which is built on a diffusion model to generate diverse SR images with a single LR input and solves the over-smoothing, mode collapse and large footprint issues simultaneously. Our work proves the potential of the diffusion model for SISR.

3. Diffusion Model

In this section, to provide a basic understanding of a diffusion model [20], we briefly review its formulation.

A diffusion model is a kind of generative model that adopts a parameterized Markov chain trained using variational inference to gradually generate data x_0 in a complex distribution from a latent variable x_T in a simple distribution, where T is the total number of diffusion steps. We set $x_t \in \mathbb{R}^d$ to be the result of each diffusion timestep $t \in \{1, 2, \ldots, T\}$, and x_t possesses the same dimensionality d as that of x_0. As shown in Fig. 2, a diffusion model is composed of two processes: the **diffusion process** and the **reverse process**.

The posterior $q(x_1, \cdots, x_T|x_0)$, named the **diffusion process**, converts the data distribution $q(x_0)$ to a latent variable distribution $q(x_T)$ and is fixed to a Markov chain that gradually adds Gaussian noise ϵ to the data according to a variance schedule $\beta_1, \cdots, \beta_T$:

$$\begin{aligned} q(x_1, \cdots, x_T|x_0) &:= \prod_{t=1}^{T} q(x_t|x_{t-1}), \\ q(x_t|x_{t-1}) &:= \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbf{I}\right), \end{aligned} \tag{1}$$

where $\mathbf{I}$ is an all-one matrix, $\mathbf{N}$ represents the Gaussian distribution, and β_t is a small positive constant that can be regarded as a constant hyperparameter. We explain how β_t is defined in A. Setting $\alpha_t := 1 - \beta_t, \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$, the diffusion process allows for sampling x_t at an arbitrary timestep t in closed form:

$$q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\right), \tag{2}$$

[1] The “one-to-many” mapping problem in super-resolution task means that one LR input has multiple corresponding SR solutions.

which can be further reparameterized as

$$x_t(x_0,\epsilon)=\sqrt{\bar{\alpha}_t}x_0+\sqrt{1-\bar{\alpha}_t}\epsilon,\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I}), \tag{3}$$

where $\mathbf{0}$ is an all-zero matrix.

The **reverse process** transforms the latent variable distribution $p_\theta(x_T)$ into the data distribution $p_\theta(x_0)$ parameterized by θ. It is defined by a Markov chain containing learned Gaussian transitions beginning with $p(x_T)=\mathcal{N}(x_T;\mathbf{0},\mathbf{I})$:

$$\begin{aligned}p_\theta(x_0,\cdots,x_{T-1}|x_T)&:=\prod_{t=1}^{T}p_\theta(x_{t-1}|x_t),\\ p_\theta(x_{t-1}|x_t)&:=\mathcal{N}\left(x_{t-1};\mu_\theta(x_t,t),\sigma_\theta(x_t,t)^2\mathbf{I}\right),\end{aligned} \tag{4}$$

where $\mu_\theta(x_t,t)$ is the mean of the Gaussian distribution of the t reverse step, and $\sigma_\theta(x_t,t)^2$ is the variance of the Gaussian distribution of the t reverse step.

In the training phase, we maximize the variational lower bound (ELBO) on the log likelihood $\log p_\theta(x_0)$ and introduce KL divergence and variance reduction [20]:

$$\begin{aligned}\mathbb{E}[\log p_\theta(x_0)]&\geqslant\mathbb{E}q\left[\log\frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right]\\ &=\mathbb{E}_q\left[\log p(x_T)+\sum_{t\geqslant1}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right]\\ &=\mathbb{E}_q\left[\log\frac{p(x_T)}{q(x_T|x_0)}+\sum_{t>1}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)}+\log p_\theta(x_0|x_1)\right]\\ &=\mathbb{E}_q\Bigg[\underbrace{-D_{KL}\left(q(x_T|x_0)||p(x_T)\right)}_{L_T}\cdot\\ &\quad-\sum_{t>1}\underbrace{D_{KL}(q(x_{t-1}|x_t,x_0)||p_\theta(x_{t-1}|x_t))}_{L_{t-1}}\underbrace{+\log p_\theta(x_0|x_1)}_{L_0}\Bigg].\end{aligned} \tag{5}$$

This transformation requires a direct comparison between $p_\theta(x_{t-1}|x_t)$ and its corresponding diffusion process posteriors. Setting

$$\tilde{\mu}_t(x_t,x_0):=\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0+\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t, \tag{6}$$

we have

$$q(x_{t-1}|x_t,x_0)=\mathcal{N}\left(x_{t-1};\tilde{\mu}_t(x_t,x_0),\tilde{\beta}_t\mathbf{I}\right), \tag{7}$$

where $\tilde{\mu}_t(x_t,x_0)$ is the mean of the Gaussian distribution in the t diffusion step, and $\tilde{\beta}_t$ is the variance of the Gaussian distribution in the t diffusion step.

Eqs. (2), (4) and (7) ensure that all KL divergences in Eq. (5) are comparisons between Gaussians. With $\sigma_t^2=\tilde{\beta}_t=\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$ for $t>1,\tilde{\beta}_1=\beta_1$, and a constant C, we have

$$L_{t-1}=\mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\mu}_t(x_t,x_0)-\mu_\theta(x_t,t)\|^2\right]+C. \tag{8}$$

For simplicity, the training procedure minimizes the variant of the ELBO with x_0 and t as inputs:

$$\min_\theta L_{t-1}(\theta)=\mathbb{E}_{x_0,\epsilon,t}\left[\|\epsilon-\epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0+\sqrt{1-\bar{\alpha}_t}\epsilon,t\right)\|^2\right], \tag{9}$$

where ϵ_θ is a noise predictor. The deduction of Eq. (9) is added in C. During the inference process, we first sample $x_T\sim\mathcal{N}(x_T;\mathbf{0},\mathbf{I})$ and then sample $x_{t-1}\sim p_\theta(x_{t-1}|x_t)$ according to Eq. (4), where

$$\begin{aligned}\mu_\theta(x_t,t)&:=\frac{1}{\sqrt{\alpha_t}}\left(x_t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\right),\\ \sigma_\theta(x_t,t)&:=\tilde{\beta}_t^{\frac{1}{2}},t\in\{T,T-1,\ldots,1\}.\end{aligned} \tag{10}$$

Then, x_{t-1} could be paramized as:

$$\begin{aligned}x_{t-1}(x_t,t)&=\mu_\theta(x_t,t)+\sigma_\theta(x_t,t)^2z\\ &=\frac{1}{\sqrt{\alpha_t}}\left(x_t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\right)+\tilde{\beta}_tz,z\sim\mathcal{N}(\mathbf{0},\mathbf{I}).\end{aligned} \tag{11}$$

4. SRDiff

As depicted in Fig. 2, SRDiff is built on a T-step diffusion model that contains two processes: a diffusion process and a reverse process. Instead of predicting the HR image directly, we apply residual prediction to predict the difference between the HR image x_H and the upsampled LR image $up(x_L)$ and denote the difference as the input residual image x_0.

The diffusion process converts x_0 into a latent variable x_T with a Gaussian distribution by gradually adding Gaussian noise ϵ, as implied in Eq. (3). According to Eqs. (4), (10) and (11). The reverse process is determined by ϵ_θ, which is a conditional noise predictor with an RRDB-based [17] low-resolution encoder (LR encoder for short) $\mathscr{D}$. The reverse process converts a latent variable x_T into a residual image x_r via iterative denoising with a finite number of steps T using the conditional noise predictor ϵ_θ, which is conditioned on the hidden states encoded from the LR image by the LR encoder $\mathscr{D}$. The SR image is reconstructed by adding the generated residual image $\hat{x_r}$ to the upsampled LR image $up(x_L)$. The top part of Fig. 3 shows the reverse process (also the inference procedure) and the bottom part of Fig. 3 shows the network structure of the conditional noise predictor.

SRDiff addresses the problems of previous SISR methods in the following ways:

- *Over-smoothing*: Compared with PSNR-oriented methods which have unimodal assumption on the target data distribution[2], SRDiff does not impose any distribution assumption on the target data x and directly maximize the variational lower bound of $p_\theta(x)$. Thus, SRDiff inherently avoids the over-smoothing problem.
- *Mode collapse*: GAN-driven methods tend to mode collapse since the generator can find a type of data that is easily able to fool the discriminator and thus make training unbanlanced. So they do not cover the full support of the training data [47]. However, SRDiff is a likelihood-based model. According to Eq. (5), it is trained to put probability mass on all samples in the training set and tends to cover all the modes. Therefore, SRDiff avoids mode collapse.
- *Large footprint*: Compared with flow-based methods that needs large number of model parameters to keep bijection between latent and data, SRDiff does not impose any architectural constraints and thus saves the model footprint.

[2] L1 loss is derived from the Laplace distribution and MSE from the Gaussian distribution [46]

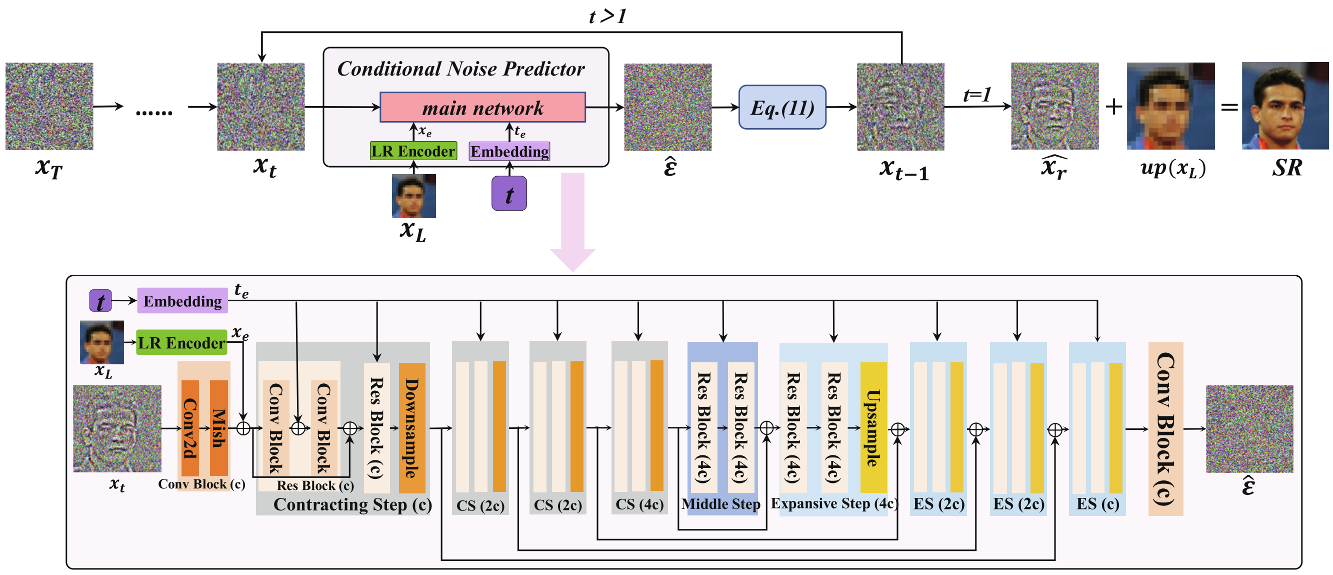


Fig. 3. The inference procedure and the architecture of the conditional noise predictor in SRDiff. The top part is the inference procedure. We extract the conditional noise predictor and depict its structure in details in the bottom part of the diagram. The content in parentheses (c, 2c and 4c) after the block name indicates the channel size of each block. “Conv2d”, “Mish”, “Conv Block”, “Res Block”, “Downsample” and “Upsample” denote a 2D convolution layer, a Mish activation, a 2D convolution block, a residual block, a downsampling layer and an upsampling layer, respectively; “CS” and “ES” denote “Contracting Step” and “Expansive Step” respectively. Due to the space limitation, this diagram is compressed. We enlarge this diagram in B.

In the following subsections, we introduce the architectures of the conditional noise predictor, LR encoder, training procedure and inference procedure.

4.1. Conditional Noise Predictor

The goal of conditional noise predictor ϵ_θ is to predict the noise ϵ added at each diffusion timestep[3]conditioned on the LR image information, according to Eqs. (9) and (10). As shown in Fig. 3, the bottom part is the architecture of the conditional noise predictor. We use U-Net as its main body, taking the 3-channel x_t, the diffusion timestep $t \in \{1, 2, \ldots, T-1, T\}$ and the output of the LR encoder as inputs. Firstly, x_t is transformed into hidden state through a 2D convolution block that consists of one 2D convolutional layer and Mish activation [49]. Then, the LR information is fused with the hidden state output by the next 2D convolution block. Following Ho et al. [20], we transform the timestep t into a timestep embedding t_e using sin positional encoding proposed in Transformer [50]. After these operations, the last output hidden state of the 2D convolution block and t_e are fed into the residual block of the first contracting step. Through the following contracting path, middle step, expansive path and the last 2D convolution block successively, noise added in the t diffusion step is predicted. Specifically, the contracting path and expansive path both consist of four steps, each of which successively applies two residual blocks and one downsampling/upsampling layer. To reduce the model size, we double the channel size only in the second and fourth contracting steps and halve the spatial size of the feature map in each contracting step. The downsampling layer in the contracting path is a two-stride 2D convolution, and the upsampling layer in the expansive path is a 2D transposed convolution. The middle step is inserted between the contracting path and the expansive path and consists of two residual blocks. In addition, the inputs of each expansive step concatenate the corresponding feature map from the contracting path. Finally, a 2D convolution block is applied to generate $\hat{\epsilon}$ in the timestep t as the predicted noise, which is then used to recover x_{t-1} according to Eqs. (4), (10) and (11). Our conditional noise predictor is easy and stable to train due to its multi-scale skip connection. Moreover, it combines local and global information through contracting and expansive paths.

4.2. LR Encoder

An LR encoder encodes the LR information x_e, which is added to each reverse step to guide the generation to the corresponding HR space. In this paper, following SRFlow [18], we choose the RRDB architecture as the LR encoder, which employs a residual-in-residual structure and multiple dense skip connections without batch normalization layers. In particular, we abandon the last convolution layer of the RRDB architecture because we do not aim to acquire concrete SR results but rather the hidden LR image information.

4.3. Training and Inference

In this subsection, we describe the training and inference procedures of SRDiff.

Training. During the training phase, as illustrated in Algorithm 1, the input LR-HR image pairs in the training set are used to train SRDiff with T total diffusion steps (Line 1). We randomly initialize the conditional noise predictor ϵ_θ, and the RRDB-based LR encoder $\mathscr{D}$ is pre-trained by the L_1 loss function (Line 2). We then sample a minibatch of LR-HR image pairs from the training set (Line 4) and compute the residual image x_r (Line 5). The LR images are encoded by the LR encoder as x_e (Line 6), which is fed into the noise predictor ϵ_θ together with t and x_T. Then, we sample ϵ from the standard Gaussian distribution and t from the integer set $\{1, \cdots, T\}$ (Line 7). We optimize the noise predictor by performing the gradient step of Eq.(9) (Line 8).

[3] Instead of the original L_2 in Eq. (9), following Chen et al. [48], we use L_1 for better training stability.

Algorithm 1: Training

```
1: Input: LR and HR image pairs $P = \{(x_L^k, x_H^k)\}_{k=1}^K$, the total number of diffusion steps $T$.
2: Initialize: randomly initialize the conditional noise predictor $\epsilon_\theta$ and pre-trained LR encoder $\mathscr{D}$.
3: repeat
4:     Sample $(x_L, x_H) \sim P$
5:     Upsample $x_L$ as $up(x_L)$, compute $x_r = x_H - up(x_L)$
6:     Encode the LR image $x_L$ as $x_e = \mathscr{D}(x_L)$
7:     Sample $\epsilon \sim \mathscr{N}(\mathbf{0}, \mathbf{I})$, and $t \sim \text{Uniform}(\{1, \cdots, T\})$
8:     Perform the gradient step on
       $\nabla_\theta \|\epsilon - \epsilon_\theta(x_t, x_e, t)\|, x_t = \sqrt{\bar{\alpha}_t} x_r + \sqrt{1 - \bar{\alpha}_t}\epsilon$
9: until converged
```

Inference. We show the inference procedure in the top part of Fig. 3 and Algorithm2. A T-step SRDiff inference procedure takes an LR image x_L as input (Line 1), as illustrated in Algorithm2. We sample a latent variable x_T from the standard Gaussian distribution (Line 3) and upsample x_L with a bicubic kernel (Line 4). Different from the training procedure, we encode the LR image x_L to x_e via the LR encoder only once before the iterative procedure begins (Line 5) and apply it in every iteration, which speeds up the inference process. The iterations start from $t = T$ (Line 6). Each iteration predicts a noise $\hat{\epsilon}$ and outputs a residual image with a different noise level, which gradually declines as t decreases. When $t > 1$, we sample z from a standard Gaussian distribution (Line 7) and compute x_{t-1} using the noise predictor ϵ_θ with x_t, x_e and t as the inputs (Line 8). When $t = 1$, we set $z = 0$ (Line 7), and x_0 is the final residual prediction. x_0 also means the predicted residual image $\hat{x_r}$. An SR image is recovered by adding the residual image x_0 to the upsampled LR image $up(x_L)$ (line 10).

Algorithm 2: Inference

```
1: Input: An LR image $x_L$, the total number of diffusion steps $T$
2: Load: conditional noise predictor $\epsilon_\theta$ and LR encoder $\mathscr{D}$
3: Sample $x_T \sim \mathscr{N}(\mathbf{0}, \mathbf{I})$
4: Upsample $x_L$ to $up(x_L)$
5: Encode the LR image $x_L$ as $x_e = \mathscr{D}(x_L)$
6: for $t = T, T-1, \cdots, 1$ do
7: Sample $z \sim \mathscr{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $z = 0$
8: Compute $x_{t-1}$ according to Eq. (11):
   $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, x_e, t)\right) + \sigma_t(x_t, t)z$
   $= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\epsilon}\right) + \sigma_t(x_t, t)z$
9: end for
10: return $x_0 + up(x_L) = \hat{x_r} + up(x_L)$ as SR prediction
```

5. Experiments

In this section, we firstly describe the experimental settings, including the utilized datasets and model configurations, as well as the details in training and evaluation. Then, we report the experimental results and conduct some analyses.

5.1. Experimental Settings

Datasets. SRDiff is trained and evaluated on face SR ($8\times$) and general SR ($4\times$) tasks. For face SR, we use the CelebFaces Attributes Dataset (CelebA) [51], which is a large-scale face attributes dataset with more than 200 k celebrity images. The images in this dataset cover large pose variations and background clutter. In this paper, we use the whole training set, which consists of 162,770 images for training, and evaluate on 5000 images from the test split, following SRFlow [18]. We centrally crop the aligned patches[4] and resize them to 160×160 as the HR ground truths using the standard MATLAB bicubic kernel. To obtain the corresponding LR images, we downsample the HR images with a bicubic kernel. For ProgFSR [15], we use the progressive bilinear kernel introduced in its original paper for a fair comparison.

For general SR, we use the DIV2K [26] and Flickr2K [26] datasets. These datasets consist of HR RGB images with high content diversity. During training, we use the whole training datasets (800 images) in DIV2K and the whole training datasets (2650 images) in Flickr2K. We crop each image into patches with sizes of 160×160 as the HR ground truths following SRFlow [18]. To obtain the corresponding LR images, we downsample the HR images with a bicubic kernel. During evaluation, we use the whole validation dataset (100 images) in DIV2K. We downsample the HR images with a bicubic kernel to obtain the LR images and directly apply SISR methods to the LR images to obtain the SR predictions without cropping.

Model Configuration. Our SRDiff model consists of a 4-step conditional noise predictor and an LR encoder with multiple RRDB blocks. The number of channels c in the first contracting step is set to 64. The numbers of RRDB blocks in the LR encoder are set to 8 and 15 for CelebA and DIV2K, respectively, and the channel size is set to 32. For the diffusion process and reverse process, we set the number of diffusion steps T to 100, and our noise schedule $\beta_1, \ldots, \beta_T$ follows Nichol et al. [52] and is described in detail in A, which has been proven beneficial for training. We also explore the model performance under different values of T and c in Section 5.3.

Training and Evaluation. Firstly, we pre-train the LR encoder $\mathscr{D}$ using an L_1 loss for 100 k iterations for the sake of efficiency. The training process of the conditional noise predictor uses Eq. (9) as the loss term and Adam [53] as the optimizer, with a batch size of 16 and a learning rate of 2×10^{-4}, which is halved every 100 k steps. The entire SRDiff procedure takes approximately 28 h (300 k steps) for training on CelebA, and 36 h (400 k steps) for training on DIV2K, on 1 GeForce RTX 3090Ti with 24 GB of memory. The evaluation is also conducted on 1 GeForce RTX 3090Ti.

In addition to the well-known PSNR and SSIM evaluation metrics [54], we also evaluate our SRDiff on the LPIPS [55], LR-PSNR [18] and pixel standard deviation σ. The LPIPS was recently introduced as a reference-based image quality evaluation metric that computes the perceptual similarity between the ground truth and the output SR image. The LR-PSNR is computed as the PSNR between a downsampled SR image and an LR image, indicating the consistency with the LR image. The pixel standard deviation σ indicates the diversity of the SR output, which is defined as $\sigma = \frac{\sum_{i=0,j=0}^{i<N,j<M} Std\left(\left\{I_{i,j}^0, \ldots, I_{i,j}^S\right\}\right)}{N*M}$, where N and M is the height and weight of the image; S is the number of SR samples generated by one method (we set S to 100 in our experiments); $Std(\cdot)$ calculates the standard deviation; and $I_{i,j}^k$ denotes the pixel value (0 to 255) located in (i,j) from the image sample I^k.

[4] https://drive.google.com/drive/folders/0B7EVK8r0v71pWEZsZE9oNnFzTm8

Table 1
Results for the 8x SR of faces on CelebA. The 1st column indicates how the LR images degenerate from the HR images and *Prog.* denotes the progressive linear kernel from ProgFSR. The 3rd column is the classification of each method. σ means the pixel standard deviation. *Speed* is measured in terms of images/s (ips). The *Parameters* column shows the number of parameters each model uses.

	Methods	Classification	↑PSNR	↑SSIM	↓LPIPS	↑LR-PSNR	↑ σ	Speed	Parameters
Bicubic	Bicubic	/	23.38	0.65	0.484	34.66	0.00	/	/
	RRDB	PSNR-oriented	26.89	0.78	0.220	48.01	0.00	28.63	12.4 M
	OISR	PSNR-oriented	26.93	0.80	0.184	54.15	0.00	67.07	46.6 M
	ESRGAN	GAN-driven	23.24	0.66	0.115	39.91	0.00	28.12	12.4 M
	SRFlow	Flow-based	25.32	0.72	0.108	50.73	5.21	4.10	40.0 M
	SRDiff	Diffusion-based	25.38	0.74	0.106	52.34	6.13	1.23	12.0 M
Prog.	ProgFSR	GAN-driven	24.21	0.69	0.126	42.19	0.00	180.11	9.0 M
	SRFlow	Flow-based	25.28	0.72	0.109	51.15	5.32	4.12	40.0 M
	SRDiff	Diffusion-based	25.32	0.73	0.106	51.41	6.19	1.20	12.0 M

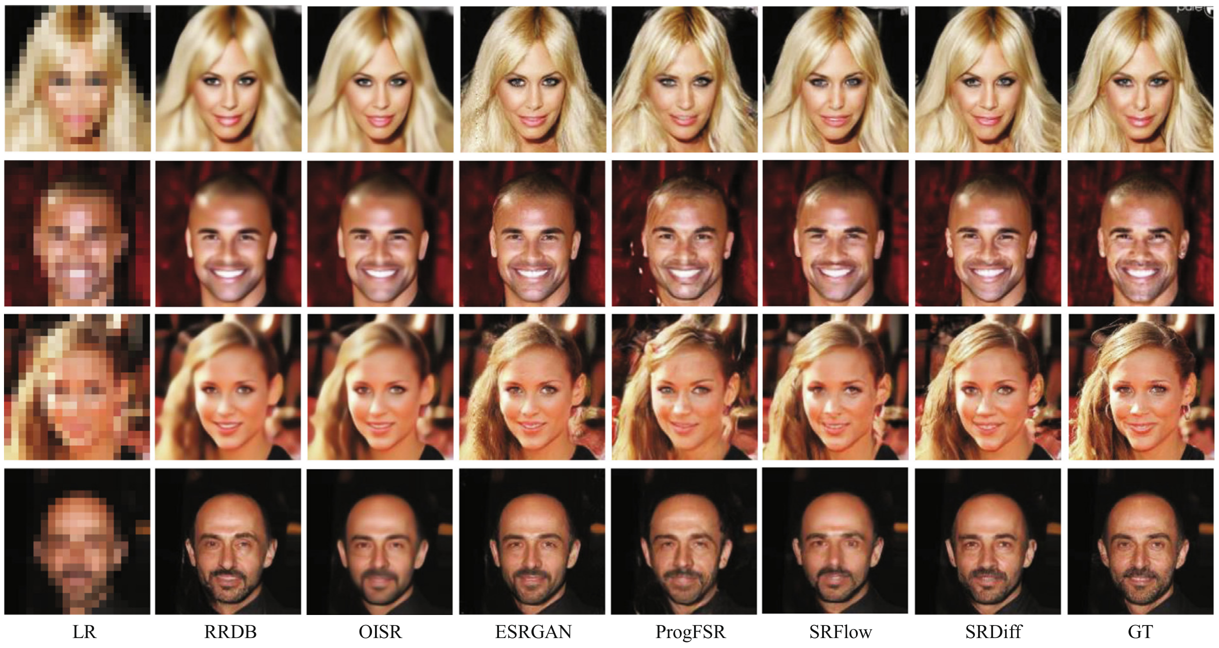


Fig. 4. Face SR (8×) visual results. SRDiff generates sharp images with richer details than RRDB, OISR and SRFlow, avoids the artifacts (e.g., grids on the women's hair and stripes on the first man's head) encountered by ESRGAN and ProgFSR and maintains consistency with the ground truth.

5.2. Performance

In this subsection, we evaluate SRDiff by comparing it with several state-of-the-art SR methods on face SR (8×) and general SR (4×) tasks. The detailed configurations of the baseline models can be found in their original papers.

Face SR. We compare SRDiff with RRDB [17], OISR [56], ESRGAN [17], ProgFSR [15] and SRFlow ($\tau = 0.8$) [18][5]. RRDB is trained with only the L_1 loss and can be regarded as a PSNR-oriented method. The evaluation results are shown in Table 1, which reveals that SRDiff outperforms previous works in terms of most of the evaluation metrics and can generate high-quality and diverse SR images with strong LR consistency. Specifically, 1) as shown in Fig. 1, SRDiff provides diverse and realistic SR predictions given only one LR input. Every SR prediction is a complete portrait of a human face with rich details, and it maintains consistency with the input LR image. 2) As shown in Fig. 4, compared with the PSNR-oriented methods, SRDiff reconstructs clearer textures. Compared with the GAN-driven methods, SRDiff avoids artifacts, and the results look more natural. Moreover, SRDiff uses fewer model parameters (12 M) than SRFlow (40 M) and a comparable number of parameters to that of the other methods listed in Table 1. SRDiff only takes 28 h until convergence, as described in Section 5.1, while SRFlow needs about 4 days and ESRGAN needs about 3 days, which demonstrates that SRDiff is training-efficient and can achieve comparable performance with a small model footprint. This is because SRDiff does not impose any architectural constraints to guarantee bijection. Compared with GAN-driven methods, SRDiff does not need any extra modules (e.g., discriminator) for training. For inference speed, our method is slower than other methods since we use a large $T = 100$ to achieve good performance. But when T decreases, the test speed increases (also means the test time reduces). For example, when $T = 25$, the testing speed is 4.43 (images/s) on CelebA dataset, which outperforms SRFlow. We illustrate this more specifically in Table 3 in Section 5.3. Though the diffusion model is relatively slow in inference, it has great potential to generate images much faster, according to recent works [57–59], which focus on boosting the inference speed of diffusion models. We will try to overcome the shortcomings of inference speed of SRDiff in our future work.

[5] Due to patch size inconsistencies, we retrain all these baseline models from scratch on our preprocessed CelebA dataset with released codes. SRFlow uses the same patch size as that of our model, but we cannot obtain the same patch with its released image example, and therefore, we also must retrain it.

Table 3
Ablations of SRDiff for face SR (8×) on CelebA. *T*, *c* and *Res.* denote the total number of diffusion steps, the channel size of the conditional noise predictor, and the residual prediction, respectively. *Speed* is measured in terms of images/s (ips).

No.	*T*	*c*	*Res.*	↑PSNR	↑SSIM	↓LPIPS	↑LR-PSNR	Speed	↓*Steps*
1	**100**	**64**	√	**25.38**	**0.74**	**0.106**	**52.34**	**1.20**	**300 k**
2	25	64	√	25.12	0.71	0.109	52.17	4.43	300 k
3	200	64	√	25.41	0.74	0.106	52.31	0.56	300 k
4	1000	64	√	25.43	0.75	0.105	52.35	0.12	300 k
5	100	32	√	25.15	0.72	0.108	52.20	1.33	300 k
6	100	128	√	25.40	0.74	0.106	52.37	1.12	300 k
7	100	64	×	24.88	0.70	0.111	51.90	1.22	600 k

Table 2
Results for 4x SR of general images on DIV2K. The 2nd column is the classification of each method. σ means the pixel standard deviation. *Speed* is measured in terms of images/s (ips). The *Parameters* column shows the number of parameters each model uses.

Methods	Classifications	↑PSNR	↑SSIM	↓LPIPS	↑LR-PSNR	↑ σ	Speed	Parameters
Bicubic	/	26.70	0.77	0.409	38.70	0.00	/	/
EDSR	PSNR-oriented	28.98	0.83	0.270	54.89	0.00	0.52	43.1 M
OISR	PSNR-oriented	28.95	0.84	0.252	55.72	0.00	0.42	44.3 M
RRDB	PSNR-oriented	29.44	0.84	0.253	49.20	0.00	1.32	16.7 M
RankSRGAN	GAN-driven	26.55	0.75	0.128	42.33	0.00	4.58	1.6 M
ESRGAN	GAN-driven	26.22	0.75	0.124	39.03	0.00	1.29	16.7 M
SRFlow	Flow-based	27.09	0.76	0.120	49.96	5.14	0.30	39.5 M
SRDiff	Diffusion-based	27.41	0.79	0.136	55.21	6.09	0.04	12.4 M

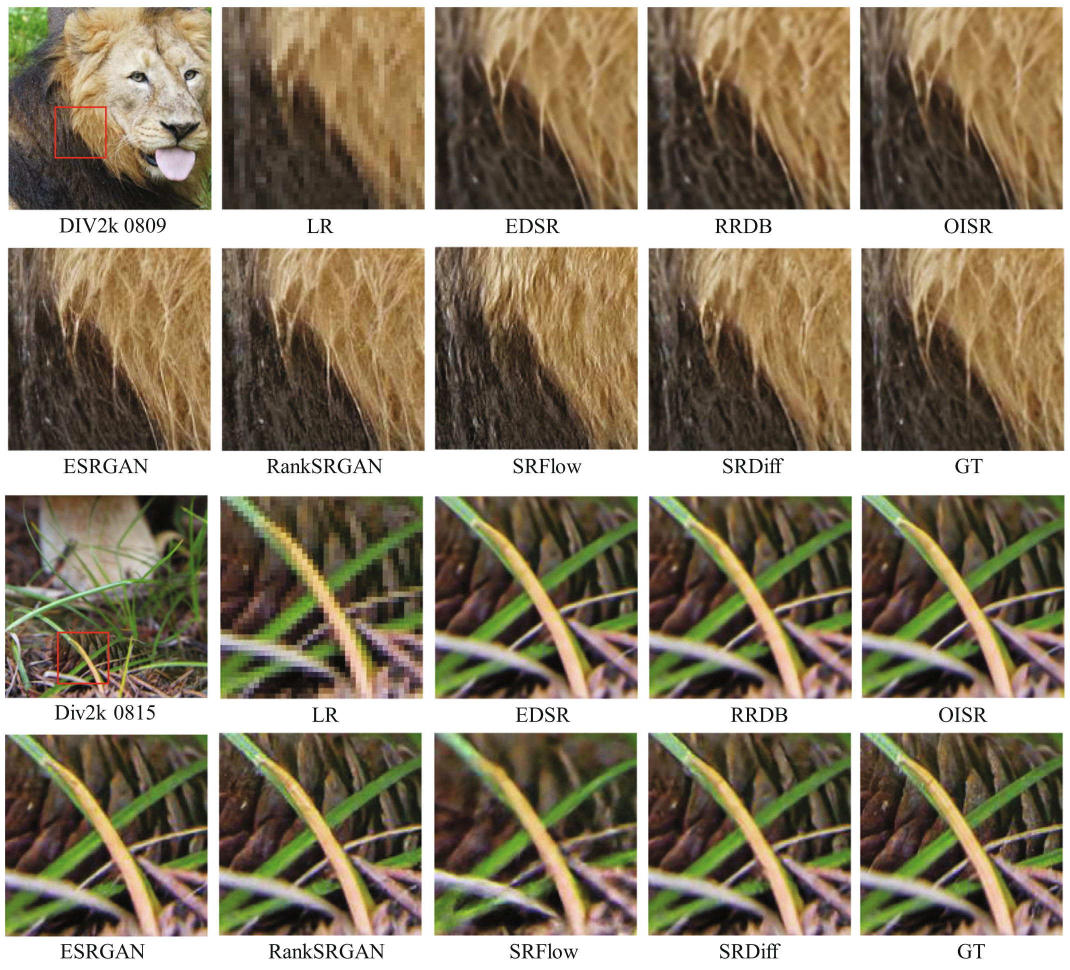


Fig. 5. General SR (4×) visual results. SRDiff maintains more natural textures in rich details, e.g., animal fur and the background pine cone, than EDSR, RRDB, OISR and SRFlow. SRDiff produces fewer unpleasant artifacts, e.g., speckles on the yellow grass, than ESRGAN and RankSRGAN. Only SRDiff maintains the brown streak on the yellow grass, which is consistent with the ground truth.

General SR. We also evaluate SRDiff on general SR ($4\times$) tasks in comparison with EDSR [7], RRDB [17], OISR [56], ESRGAN [17], RankSRGAN [34] and SRFlow ($\tau = 0.9$) [18] with their officially released pre-trained models[6]. As shown in Table 2, SRDiff achieves better quantitative results than those of the previous methods in terms of most evaluation metrics (PSNR, SSIM and LR-PSNR) and the comparable LPIPS, which reveals the effectiveness and great potential of our method. Fig. 5 shows that SRDiff balances sharpness and naturalness well and produces strong consistency with the LR image. In contrast, the PSNR-oriented methods (EDSR, OISR and RRDB) and SRFlow smear the edges of the objects, and the GAN-driven methods (ESRGAN and RankSRGAN) introduce more artifacts. In terms of the model footprints, SRDiff uses fewer model parameters than most of the methods listed in Table 2.

The performances of our methods prove that SRDiff could solve the over-smoothing problem in PSNR-oriented methods, mode collapse problem in GAN-driven methods and large footprint issue in flow-based methods:

- **over-smoothing problem**: We consider LPIPS as an evaluation metric, since it has been shown to correlate much better with human opinions [18]. Compared with methods listed in Table 1 and Table 2, our method obtains the lowest LPIPS on Face SR tasks and relatively low LPIPS in General SR task. Consequently, our method solves the over-smoothing problem with improved perceptual quality.
- **mode collapse problem**: From pixel standard deviation σ in Table 1 (8th column) and Table 2 (7th column), we can see that the diversity of each pixel in SRDiff is higher than that in GAN-driven methods. This indicates that SRDiff could generate diverse SR results and avoid the mode collapse problem.
- **large footprint problem**: In Table 1 and Table 2, the last column shows that our method uses less parameters than flow-based methods and therefore solves the large footprint problem.

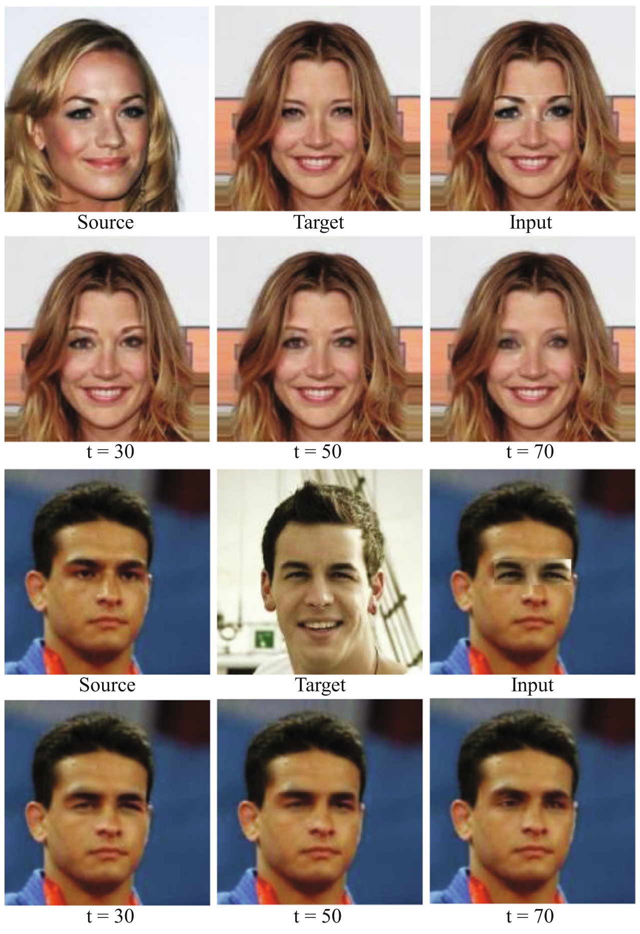
5.3. Ablation Study

To probe the influences of the total number of diffusion steps T, the channel size c of the conditional noise predictor and the effectiveness of residual prediction, we conduct ablation studies, as illustrated in Table 3. From rows 1, 2, 3 and 4, we can see that the image quality is improved as the total number of diffusion steps T increases, whil the testing speed decreases (also means the testing time grows) as T increases. From rows 1, 5 and 6, we can see that a larger model width results in better performance. However, more total diffusion steps and a larger model width both lead to slower inference. Therefore, we choose $T = 100$ and $c = 64$ as the default settings to achieve a trade-off. Rows 1 and 7 indicate that residual prediction not only greatly improves the image quality but also speeds up the training process, which demonstrates the effectiveness of residual prediction.

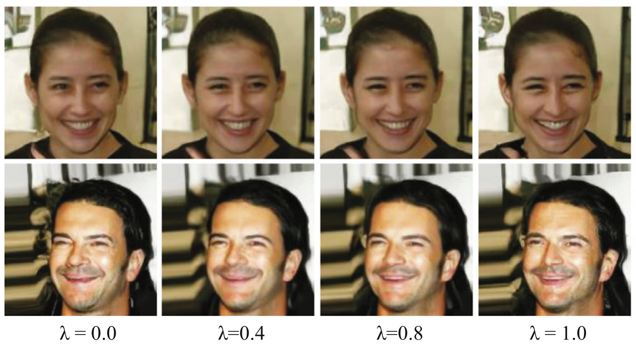
5.4. Extensions

In this subsection, we explore some extended applications, including content fusion and latent space interpolation.

Content Fusion. SRDiff is applicable in content fusion tasks, which aim to generate an image by fusing contents from two source images, e.g., a source eye image and a source face image that provide the eye and face contents, respectively. In this paragraph, we use SRDiff to conduct face content fusion by a demonstration of fusing one's eyes with another person's face. The procedure of content fusion is shown in Algorithm 3. Firstly, we directly fuse a face image x_f by replacing the eye region of the source face image x_{face} with that of the source eye image x_{eye} and compute the differences between x_f and the upsampled LR face source image $up(x_L)$ to obtain the residual x_r. Secondly, x_r goes through a $\bar{t}$-step diffusion process, which outputs $x_{\bar{t}}$ in latent space. Then, $x_{\bar{t}}$ is denoised to an HR residual using the conditional noise predictor iterated from $\bar{t}$ to 0 with the LR source face information encoded by the LR encoder, which ensures the compatibility of the two contents. After this, we obtain a fused SR image by adding the SR residual to $up(x_L)$. Finally, we replace the eye region of the



(a) Applying SRDiff on a content fusion task.



(b) Applying SRDiff on a latent interpolation task.

Fig. 6. Extended applications of SRDiff.

[6] except RRDB, which is trained from scratch with the L_1 loss used for the face SR tasks.

source face image with that of the SR face image and preserve the non-manipulated face. As shown in Fig. 6a, we set different time-steps $t \in \{30, 50, 70\}$ and find that the eye (including the eye-brows) region of the fusion result is more similar to the source eye image when t is small and is closer to the source face image as t becomes larger.

Algorithm 3: Content Fusion

```
1: Input: source eye image $x_{eye}$, source face image $x_{face}$,
   number of diffusion steps $\bar{t}$.
2: Load: conditional noise predictor $\epsilon_\theta$ and LR encoder $\mathscr{D}$.
3: Replace the eye region of $x_{face}$ with the that of $x_{eye}$ to form
   the initial fused image $x_f$
4: Upsample the LR source face image $x_L$ as $up(x_L)$
5: Compute $x_r = x_f - up(x_L)$
6: Put $x_r$ into the diffusion process and compute
   $x_{\bar{t}}(x_r, \epsilon) = \sqrt{\bar{\alpha}_{\bar{t}}} x_r + \sqrt{1 - \bar{\alpha}_{\bar{t}}} \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
7: Encode $x_L$ as $x_e = \mathscr{D}(x_L)$
8: for $t = \bar{t}, \cdots, 1$ do
9:     Sample $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $z = 0$
10:    Compute $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, x_e, t) \right) + \sigma_t(x_t, t) z$
11: end for
12: Compute the SR face prediction $x_H = x_0 + up(x_L)$
13: Crop the eye region of $x_H$ and insert it into the
    corresponding eye region of $x_{face}$ to generate $x_{fused}$
14: return $x_{fused}$ as the content fusion result
```

Latent Space Interpolation. Given an LR image, SRDiff can manipulate its prediction by latent space interpolation, which linearly interpolates the latent variables of two SR predictions and generates a new prediction. Let $x_{\bar{t}}, x'_{\bar{t}} \sim q(x_{\bar{t}}|x_0)$; we decode the latent variable $\bar{x}_{\bar{t}} = \lambda x_{\bar{t}} + (1 - \lambda) x'_{\bar{t}}$ by the reverse process, which feeds $\bar{x}_{\bar{t}}$ into the noise predictor with the LR information iteratively encoded by the LR encoder. Then, we add the output residual result to $up(x_L)$ to obtain the interpolated SR prediction. We set $\bar{t} = 50$ and $\lambda \in \{0.0, 0.4, 0.8, 1.0\}$. Fig. 6b shows that as λ approaches 1.0, the woman's expression becomes closer to x_t, which is the top right image exhibiting a large laugh. In the same way, the man's mouth becomes wider as $\lambda = 0.0$ shifts to $\lambda = 1.0$. The trend of the interpolated images show the effectiveness of SRDiff in latent space interpolation. The detailed algorithm of latent space interpolation is shown in Algorithm 4.

Algorithm 4: Latent Space Interpolation

```
1: Input: LR image $x_L$, number of diffusion steps $\bar{t}$, $\lambda \in [0, 1]$
2: Load: conditional noise predictor $\epsilon_\theta$ and LR encoder $\mathscr{D}$
3: Sample $x_{\bar{t}}, x_{\bar{t}'} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
4: Compute $\bar{x}_{\bar{t}} = \lambda x_{\bar{t}} + (1 - \lambda) x_{\bar{t}'}$
5: Upsample $x_L$ as $up(x_L)$
6: Encode $x_L$ as $x_e = \mathscr{D}(x_L)$
7: for $t = \bar{t}, \cdots, 1$ do
8:     Sample $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $z = 0$
9:     Compute $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\bar{x}_t, x_e, t) \right) + \sigma_t(x_t, t) z$
10: end for
11: return the interpolated SR face prediction
    $x_H = x_0 + up(x_L)$ as the latent interpolation results
```

6. Conclusion

In this paper, we proposed SRDiff, which is the first diffusion-based model for SISR to the best of our knowledge.Our work firstly exploited a Markov chain to convert HR images into latent variables in simple distributions. And then we conducted the reverse process that generates the SR predictions by iteratively denoising the latent variables using a noise predictor conditioned on LR information. To speed up the convergence process and stabilize the training procedure, SRDiff introduces residual prediction. Our extensive experiments on both face and general datasets demonstrated that SRDiff can generate diverse and realistic SR images. Moreover, both the theory and the performances show that our method is capable of solving the over-smoothing, mode collapse and large footprint issues that occur in PSNR-oriented methods, GAN-driven methods and flow-based methods, respectively. In addition, SRDiff allows for flexible image manipulation, including latent space interpolation and content fusion. The diverse SR results generated by our method can provide more references for

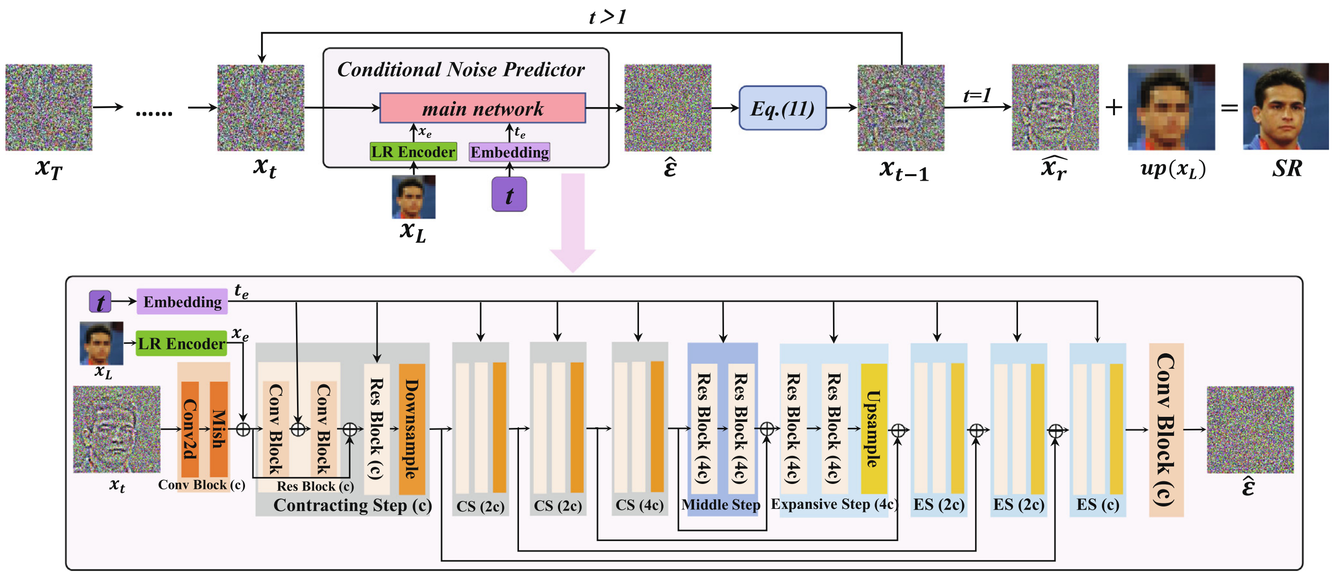


Fig. 7. Enlarged version of Fig. 3.

the user to select according to their demands. Our method sheds much light on potential new research directions.

In the future, we will further improve the performance of the developed diffusion-based SISR model and speed up the inference process. We will also extend our work to more image restoration tasks (e.g., image denoising, deblurring and dehazing) to verify the potential of diffusion models in the image restoration domain.

CRediT authorship contribution statement

Haoying Li: Conceptualization, Methodology, Validation, Formal analysis, Investigation, Resources, Data curation, Writing - original draft, Writing - review & editing, Visualization. **Yifan Yang:** Data curation, Writing - review & editing. **Meng Chang:** Investigation. **Shiqi Chen:** Funding acquisition. **Huajun Feng:** Supervision, Funding acquisition. **Qi Li:** Funding acquisition. **Yueting Chen:** Project administration, Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China: 61975175.

Appendix A. The Definition of β_t

To define the variable β_t, following [52], we define

$$\beta_t = 1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, \bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T+s}{1+s}\cdot\frac{\pi}{2}\right)^2, \quad s = 0.008.$$

In practice, we clip β_t to be smaller than 0.99 to prevent singularities at the end of the diffusion process near $t = T$.

Appendix B. The Enlarged Fig. 3

We enlarge Fig. 3 and place it in Fig. 7.

Appendix C. The Deduction of Eq. (9)

For the deduction of Eq. (9), we further reparameterize 2 as $x_t(x_0,\epsilon) = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ for $\epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ and applying the forward process posterior formula:

$$\begin{aligned} L_{t-1} - C &= \mathbb{E}_{x_0,\epsilon}\left[\frac{1}{2\sigma_t^2}\left\|\tilde{\mu}_t\left(x_t(x_0,\epsilon), \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t(x_0,\epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon\right)\right)\right.\right. \\ &\quad \left.\left. - \mu_\theta(x_t(x_0,\epsilon),t)\right\|^2\right] \\ &= \mathbb{E}_{x_0,\epsilon}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(x_t(x_0,\epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right) - \mu_\theta(x_t(x_0,\epsilon),t)\right\|^2\right], \end{aligned}$$

Given x_t, μ_σ could predict $\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$. Because x_t is the input of the model, we could parameterize μ_σ as

$$\begin{aligned} \mu_\theta(x_t,t) &= \tilde{\mu}_t\left(x_t, \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t)\right)\right) \\ &= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\right), \end{aligned}$$

This equation could be rewritten as:

$$\mathbb{E}_{x_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right)\right\|^2\right].$$

For easier training of the variational bound, we minimize the variant of the ELBO with x_0 and t as inputs:

$$\min_\theta L_{t-1}(\theta) = \mathbb{E}_{x_0,\epsilon,t}\left[\|\epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right)\|^2\right].$$

References

[1] C. Fookes, F. Lin, V. Chandran, S. Sridharan, Evaluation of image resolution and super-resolution on face recognition performance, Journal of Visual Communication and Image Representation 23 (1) (2012) 75–93.
[2] M.S. Sajjadi, B. Scholkopf, M. Hirsch, Enhancenet: Single image super-resolution through automated texture synthesis, in: ICCV, 2017, pp. 4491–4500..
[3] F. Li, X. Jia, D. Fraser, A. Lambert, Super resolution for remote sensing images based on a universal hidden markov tree model, IEEE Transactions on Geoscience and Remote Sensing 48 (3) (2009) 1270–1278.
[4] H. Fang, S. Yao, Q. Chen, C. Liu, Y. Cai, S. Geng, Y. Bai, Z. Tian, A.L. Zacharias, T. Takebe, et al., De novo-designed near-infrared nanoaggregates for super-resolution monitoring of lysosomes in cells, in whole organoids, and in vivo, ACS nano 13 (12) (2019) 14426–14436.
[5] S. Park, Y. Kang, J. Park, J. Kim, Self-controllable super-resolution deep learning framework for surveillance drones in security applications, EAI Endorsed Transactions on Security and Safety 7 (23) (2020) e5.
[6] C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks, IEEE transactions on pattern analysis and machine intelligence 38 (2) (2015) 295–307.
[7] B. Lim, S. Son, H. Kim, S. Nah, K. Mu Lee, Enhanced deep residual networks for single image super-resolution, in, CVPRW (2017) 136–144.
[8] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, Y. Fu, Image super-resolution using very deep residual channel attention networks, in, ECCV (2018) 286–301.
[9] Y. Qiu, R. Wang, D. Tao, J. Cheng, Embedded block residual network: A recursive restoration model for single-image super-resolution, in: ICCV, 2019, pp. 4180–4189..
[10] Y. Guo, J. Chen, J. Wang, Q. Chen, J. Cao, Z. Deng, Y. Xu, M. Tan, Closed-loop matters: Dual regression networks for single image super-resolution, in: CVPRW, 2020, pp. 5407–5416..
[11] G. Lin, Q. Wu, L. Qiu, X. Huang, Image super-resolution using a dilated convolutional neural network, Neurocomputing 275 (2018) 1219–1230.
[12] X. Jin, Q. Xiong, C. Xiong, Z. Li, Z. Gao, Single image super-resolution with multi-level feature fusion recursive network, Neurocomputing 370 (2019) 166–173.
[13] S. Li, R. Fan, G. Lei, G. Yue, C. Hou, A two-channel convolutional neural network for image super-resolution, Neurocomputing 275 (2018) 267–277.
[14] M. Cheon, J.-H. Kim, J.-H. Choi, J.-S. Lee, Generative adversarial network-based image super-resolution using perceptual content losses, in, Proceedings of ECCV (2018).
[15] D. Kim, M. Kim, G. Kwon, D.-S. Kim, Progressive face super-resolution via attention to facial landmark, arXiv preprint arXiv:1908.08239..
[16] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A.P. Aitken, A. Tejani, J. Totz, Z. Wang, et al., Photo-realistic single image super-resolution using a generative adversarial network, CVPR..
[17] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, C.C. Loy, Y. Qiao, X. Tang, Esrgan: Enhanced super-resolution generative adversarial networks, ECCV..
[18] A. Lugmayr, M. Danelljan, L. Van Gool, R. Timofte, Srflow: Learning the super-resolution space with normalizing flow, in: ECCV, Springer, 2020, pp. 715–732..
[19] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks..
[20] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, NeurIPS..
[21] Z. Kong, W. Ping, J. Huang, K. Zhao, B. Catanzaro, Diffwave: A versatile diffusion model for audio synthesis, arXiv preprint arXiv:2009.09761..
[22] J. Liu, C. Li, Y. Ren, F. Chen, P. Liu, Z. Zhao, Diffsinger: Diffusion acoustic model for singing voice synthesis, arXiv preprint arXiv:2105.02446..
[23] G. Mittal, J. Engel, C. Hawthorne, I. Simon, Symbolic music generation with diffusion models, arXiv preprint arXiv:2103.16091..
[24] S. Luo, W. Hu, Diffusion probabilistic models for 3d point cloud generation, arXiv preprint arXiv:2103.01458..
[25] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, in: ICCV, 2015, pp. 3730–3738..
[26] R. Timofte, S. Gu, J. Wu, L. Van Gool, Ntire 2018 challenge on single image super-resolution: methods and results, in: CVPR Workshops, 2018..
[27] J. Kim, J. Kwon Lee, K. Mu Lee, Deeply-recursive convolutional network for image super-resolution, in: CVPR, 2016, pp. 1637–1645..
[28] J. Kim, J. Kwon Lee, K. Mu Lee, Accurate image super-resolution using very deep convolutional networks, in: CVPR, 2016..
[29] J. Li, F. Fang, K. Mei, G. Zhang, Multi-scale residual network for image super-resolution, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 517–532.

[30] Y. Mei, Y. Fan, Y. Zhou, L. Huang, T.S. Huang, H. Shi, Image super-resolution with cross-scale non-local attention and exhaustive self-exemplars mining, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5690–5699.

[31] Y. Dun, Z. Da, S. Yang, X. Qian, Image super-resolution based on residually dense distilled attention network, Neurocomputing 443 (2021) 47–57.

[32] X. Chen, X. Wang, Y. Lu, W. Li, Z. Wang, Z. Huang, Rbpnet: An asymptotic residual back-projection network for super-resolution of very low-resolution face image, Neurocomputing 376 (2020) 119–127.

[33] M.S. Rad, B. Bozorgtabar, U.-V. Marti, M. Basler, H.K. Ekenel, J.-P. Thiran, Srobb: Targeted perceptual loss for single image super-resolution, in: ICCV, 2019, pp. 2710–2719..

[34] W. Zhang, Y. Liu, C. Dong, Y. Qiao, Ranksrgan: Generative adversarial networks with ranker for image super-resolution, in: ICCV, 2019..

[35] M.E.A. Seddik, M. Tamaazousti, J. Lin, Generative collaborative networks for single image super-resolution, Neurocomputing 398 (2020) 293–303.

[36] M.C. Bühler, A. Romero, R. Timofte, Deepsee: Deep disentangled semantic explorative extreme super-resolution, arXiv preprint arXiv:2004.04433..

[37] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, M.-H. Yang, Mode seeking generative adversarial networks for diverse image synthesis, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 1429–1437.

[38] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training gans, Advances in neural information processing systems 29 (2016) 2234–2242.

[39] M. Mathieu, C. Couprie, Y. LeCun, Deep multi-scale video prediction beyond mean square error, arXiv preprint arXiv:1511.05440..

[40] D. Patel, A.A. Oberai, Bayesian inference with generative adversarial network priors, arXiv preprint arXiv:1907.09987..

[41] D. Yang, S. Hong, Y. Jang, T. Zhao, H. Lee, Diversity-sensitive conditional generative adversarial networks, arXiv preprint arXiv:1901.09024..

[42] L. Dinh, D. Krueger, Y. Bengio, Nice: Non-linear independent components estimation, arXiv preprint arXiv:1410.8516..

[43] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real nvp, arXiv preprint arXiv:1605.08803..

[44] D.P. Kingma, P. Dhariwal, Glow: Generative flow with invertible 1x1 convolutions, in: Advances in neural information processing systems, 2018, pp. 10215–10224..

[45] J. Sohl-Dickstein, E.A. Weiss, N. Maheswaranathan, S. Ganguli, Deep unsupervised learning using nonequilibrium thermodynamics, arXiv preprint arXiv:1503.03585..

[46] T. Chai, R.R. Draxler, Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature, Geoscientific model development 7 (3) (2014) 1247–1250.

[47] T. Lucas, K. Shmelkov, K. Alahari, C. Schmid, J. Verbeek, Adaptive density estimation for generative models, Advances in Neural Information Processing Systems 32 (2019) 12016–12026.

[48] N. Chen, Y. Zhang, H. Zen, R.J. Weiss, M. Norouzi, W. Chan, Wavegrad: Estimating gradients for waveform generation, in: ICLR, 2021. URL:https://openreview.net/forum?id=NsMLjcFaO8O.

[49] D. Misra, Mish: A self regularized non-monotonic neural activation function, arXiv preprint arXiv:1908.08681..

[50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: NeurIPS, 2017, pp. 5998–6008..

[51] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, in: ICCV, 2015..

[52] A.Q. Nichol, P. Dhariwal, Improved denoising diffusion probabilistic models (2021). URL:https://openreview.net/forum?id=-NEXDKk8gZ..

[53] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980..

[54] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE transactions on image processing 13 (4) (2004) 600–612.

[55] R. Zhang, P. Isola, A.A. Efros, E. Shechtman, O. Wang, The unreasonable effectiveness of deep features as a perceptual metric, CVPR..

[56] X. He, Z. Mo, P. Wang, Y. Liu, M. Yang, J. Cheng, Ode-inspired network design for single image super-resolution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 1732–1741.

[57] Z. Kong, W. Ping, On fast sampling of diffusion probabilistic models, arXiv preprint arXiv:2106.00132..

[58] D. Watson, J. Ho, M. Norouzi, W. Chan, Learning to efficiently sample from diffusion probabilistic models, arXiv preprint arXiv:2106.03802..

[59] A. Jolicoeur-Martineau, K. Li, R. Piché-Taillefer, T. Kachman, I. Mitliagkas, Gotta go fast when generating data with score-based models, arXiv preprint arXiv:2105.14080..

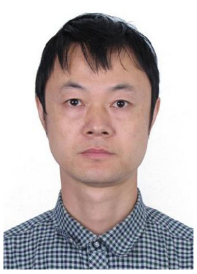
Haoying Li received her B.S. degree from the College of Optical Science and Engineering in Zhejiang University in 2019. Recently, she is a Ph.D. candidate at State Key Laboratory of Modern Optical Instrumentation in Zhejiang University. Her research interests include super-resolution and deblurring.

Yifan Yang received the B.E. degree from School of Informatics in Xiamen University in 2016. Recently, he is a Ph.D. candidate at State Key Laboratory of Modern Optical Instrumentation in Zhejiang University. His research interests include computer vision and optical engineering.

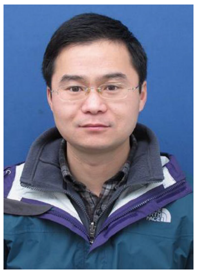
Meng Chang received his Ph.D. degree from the College of Optical Science and Engineering in Zhejiang University in 2021.His research lies in the field of image signal processing.

Huajun Feng received his B.S. and M.S. degrees from Zhejiang University in 1983. Currently, he is a professor at the State Key Laboratory of Modern Optical Instrumentation in Zhejiang University. His research lies in the field of imaging techniques, imaging processing and precision testing.

Zhihai Xu received his bachelor and master degrees from Zhejiang University in 1986 and 1989, respectively. He received his Ph.D. degree from Zhejiang University in 1996. Currently, he is a professor at the State Key Laboratory of Modern Optical Instrumentation in Zhejiang University.



Qi Li received his Ph.D. degree at Zhejiang University with a major of optical engineering in 2004. Currently, he is a professor at State Key Laboratory of Modern Optical Instrumentation in Zhejiang University. His research lies in the field of optical system design.at



Yueting Chen received his bachelor degree andoctor'sor degree in Zhejiang University in 2004 and 2009, respectively. Currently he is an associate professor at State Key Laboratory of Modern Optical Instrumentation in Zhejiang University. His research is in the field of computational imaging.

Shiqi Chen received the B.S degree at Huazhong University of Science and Technology in 2018. Recently, he is a Ph.D candidate at State Key Laboratory of Modern Optical Instrumentation in Zhejiang University. His research interests include image processing and computer vision.