Group B: Image Compression and Restoration

Shayla Luong
Aditi Agrawal
Arushi Singh

Background

Images and videos are compressed when sent over internet.

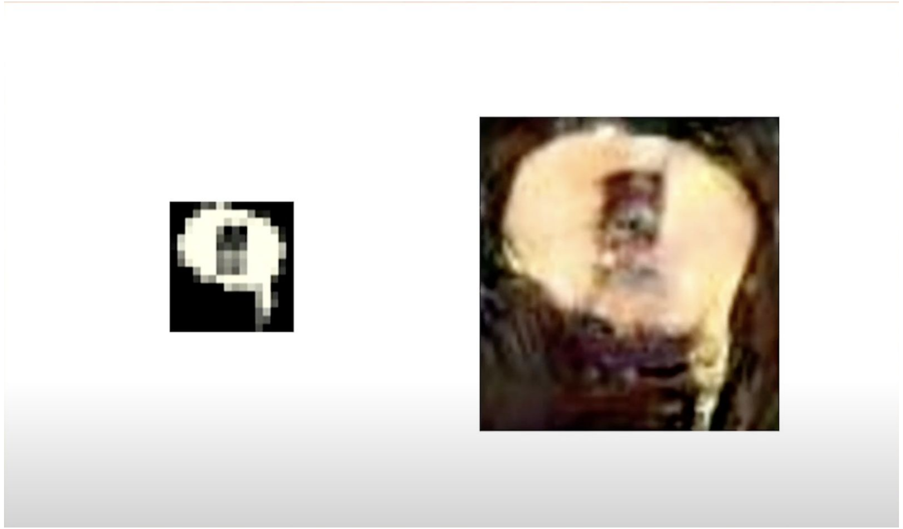
WhatsApp converts (& Compress) most images to jpeg format and it puts Lossy Compression capabilities to good use in reducing image size and to make it light.

Here are some sizes of the images sent over internet via various social media platforms

- **Facebook Profile Image Size** – 180 x 180 px
- **Facebook Shared Image Size** – 1200 x 630 px
- **Instagram Reel Image Size** – 1080 x 1920 px
- **Instagram Feed Image Size** – 1080 x 1350 px
- **Twitter Profile image** – 400 x 400 px
- **Twitter Share image** – 1200 x 675 px
- **WhatsApp Square Post to Send Image Size** – 800 x 800 px

Dataset

- Image enhancement dataset focus on specific problems like low lighting, underwater enhancement
- VGGFace2 dataset has 3.31 million images of faces
- Improve the quality of any image
- Super-Resolution dataset
    - BSDS200
    - DIV2K
    - Flicker2K
    - Set5
    - Set14
    - Urban10
- Apply different degradation



Classical Approaches

Image Interpolation

- Nearest-neighbor, bilinear, bicubic

Translational motion in frequency domain

Regularization

Statistics-based

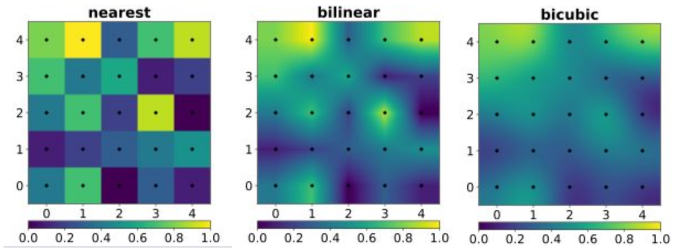


Image Source: https://en.wikipedia.org/wiki/Bicubic_interpolation

State-of-the-Art

- Convolutional Neural Network
- Residual Networks
- Generative Adversarial Network
- Transformers

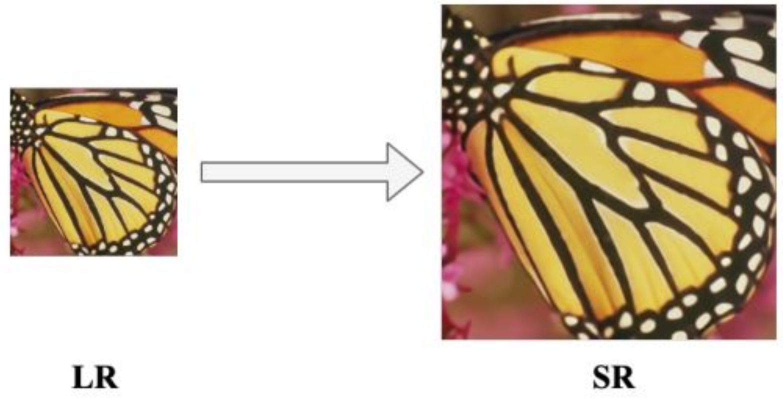


Image Source: https://arxiv.org/pdf/2109.14335.pdf

Types of Compression

Lossless compression

Every single bit of data that was originally in the file remains after the file is uncompressed.

All of the information is completely restored. This is generally the technique of choice for text or spreadsheet files, where losing words or financial data could pose a problem.

Examples: PNG, GIF,ZIP

Lossy Compression

It reduces a file by permanently eliminating certain information, especially redundant information.

When the file is uncompressed, only a part of the original information is still there (although the user may not notice it)

Example: JPEG

Compression used by Social Network

Social Media for example whatsapp uses compression format such as JPEG,JPEG200 to compress and downscale the data to be sent over internet.

How does it work?

There are particularly three components when it comes to compression of data.

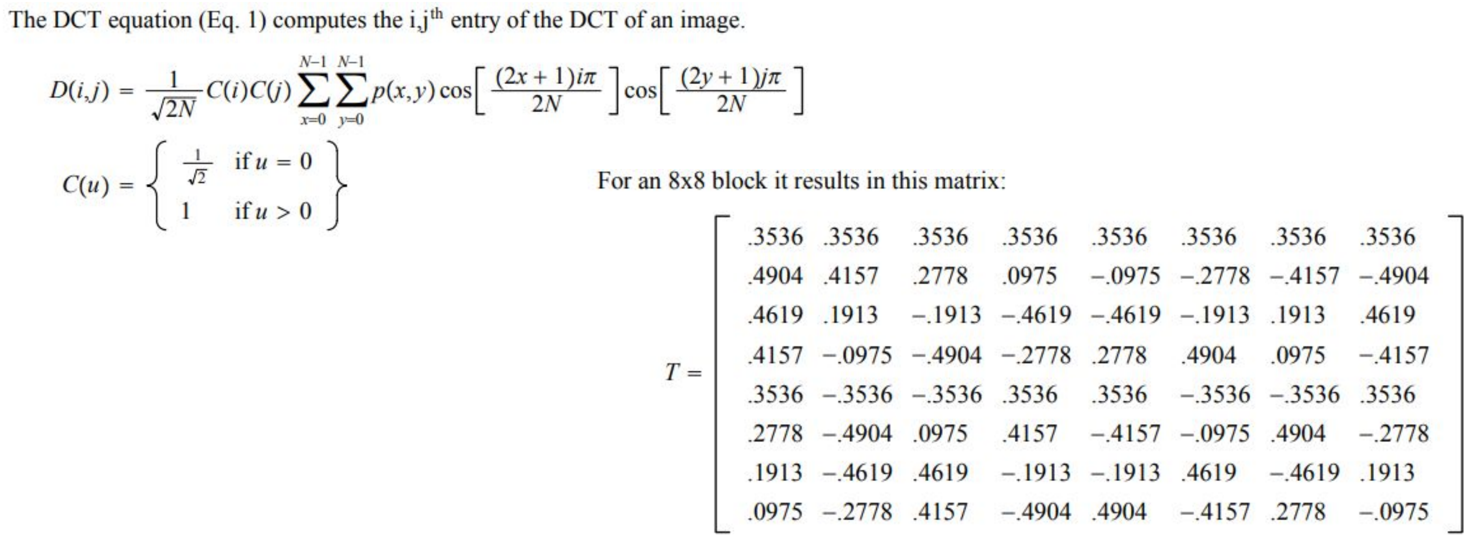
1)Source Encoder

2)Quantizer

3)Entropy Encoder

Source Encoder (or Linear Transformer):

Over the years, a variety of linear transforms have been developed which include Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT) , Discrete Wavelet Transform (DWT)



The DCT equation (Eq. 1) computes the i,jth entry of the DCT of an image.

$$D(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} p(x,y) \cos\left[\frac{(2x+1)i\pi}{2N}\right] \cos\left[\frac{(2y+1)j\pi}{2N}\right]$$

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } u = 0 \\ 1 & \text{if } u > 0 \end{cases}$$

For an 8x8 block it results in this matrix:

$$T = \begin{bmatrix} .3536 & .3536 & .3536 & .3536 & .3536 & .3536 & .3536 & .3536 \\ .4904 & .4157 & .2778 & .0975 & -.0975 & -.2778 & -.4157 & -.4904 \\ .4619 & .1913 & -.1913 & -.4619 & -.4619 & -.1913 & .1913 & .4619 \\ .4157 & -.0975 & -.4904 & -.2778 & .2778 & .4904 & .0975 & -.4157 \\ .3536 & -.3536 & -.3536 & .3536 & .3536 & -.3536 & -.3536 & .3536 \\ .2778 & -.4904 & .0975 & .4157 & -.4157 & -.0975 & .4904 & -.2778 \\ .1913 & -.4619 & .4619 & -.1913 & -.1913 & .4619 & -.4619 & .1913 \\ .0975 & -.2778 & .4157 & -.4904 & .4904 & -.4157 & .2778 & -.0975 \end{bmatrix}$$

$$Original = \begin{bmatrix} 154 & 123 & 123 & 123 & 123 & 123 & 123 & 136 \\ 192 & 180 & 136 & 154 & 154 & 154 & 136 & 110 \\ 254 & 198 & 154 & 154 & 180 & 154 & 123 & 123 \\ 239 & 180 & 136 & 180 & 180 & 166 & 123 & 123 \\ 180 & 154 & 136 & 167 & 166 & 149 & 136 & 136 \\ 128 & 136 & 123 & 136 & 154 & 180 & 198 & 154 \\ 123 & 105 & 110 & 149 & 136 & 136 & 180 & 166 \\ 110 & 136 & 123 & 123 & 123 & 136 & 154 & 136 \end{bmatrix}$$

If this is my original matrix, then to get the transformed matrix we do matrix multiplication,

D=TOT’

Quantization

- Quantizer: A quantizer simply reduces the number of bits needed to store the transformed coefficients by reducing the precision of those values. Since this is a many-to-one mapping, it is a lossy process and is the main source of compression in an encoder.
- The top left coefficient correlates to low frequencies of the original image block,and the down right coefficient correlates to the higher frequency.
- It is important to note that human eye is more sensitive to low frequencies.
- Therefore in the quantized matrix, the top values are lower so that when the elements of matrix D are divided there is not much change observed where as the bottom right corner values are higher, to nullify the higher frequency data.

Quantization Example:

```
[ [16,  11,  10,  16,  24,  40,  51,  61],
  [12,  12,  14,  19,  26,  58,  60,  55],
  [14,  13,  16,  24,  40,  57,  69,  56],
  [14,  17,  22,  29,  51,  87,  80,  62],
  [18,  22,  37,  56,  68, 109, 103,  77],
  [24,  35,  55,  64,  81, 104, 113,  92],
  [49,  64,  78,  87, 103, 121, 120, 101],
  [72,  92,  95,  98, 112, 100, 103,  99]]
```

$$C = \begin{bmatrix} 10 & 4 & 2 & 5 & 1 & 0 & 0 & 0 \\ 3 & 9 & 1 & 2 & 1 & 0 & 0 & 0 \\ -7 & -5 & 1 & -2 & -1 & 0 & 0 & 0 \\ -3 & -5 & 0 & -1 & 0 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\text{Quantized Value}(i, j) = \frac{\text{DCT}(i, j)}{\text{Quantum}(i, j)} \text{ Rounded to the nearest integer}$$

Entropy Encoder:

- An entropy encoder further compresses the quantized values lossless to give better overall compression.
- It uses a model to accurately determine the probabilities for each quantized value and produces an appropriate code based on these probabilities so that the resultant output code stream will be smaller than the input stream.
- The most commonly used entropy encoders are the Huffman encoder
- Now all coefficients of matrix C are converted into Binary data.One of the popular once is Huffman coding. When the most frequently occurred character is given the smallest code for example : 01. This encoded data is then sent over the network.

Deep Convolutional AutoEncoder based Lossy Compression

As we have already seen the , traditional image compression algorithms such as JPEG,JPEG 2000 used fixed transform matrix.

Drawback with this approach is:

Not optimal and flexible image coding solution for all image formats, therefore we would   be using deep learning approach which contains autoencoder with convolution layers and PCA.

1) To replace the transform and inverse transform in traditional codecs, we design a symmetric CAE structure with multiple downsampling and upsampling units to generate feature maps with low dimensions. We optimize this CAE using an approximated rate-distortion loss function.

2) It’s a principal components analysis (PCA)-based rotation to generate more zeros in the feature maps. Then, the quantization and entropy coder are utilized to compress the data further.

Block Diagram

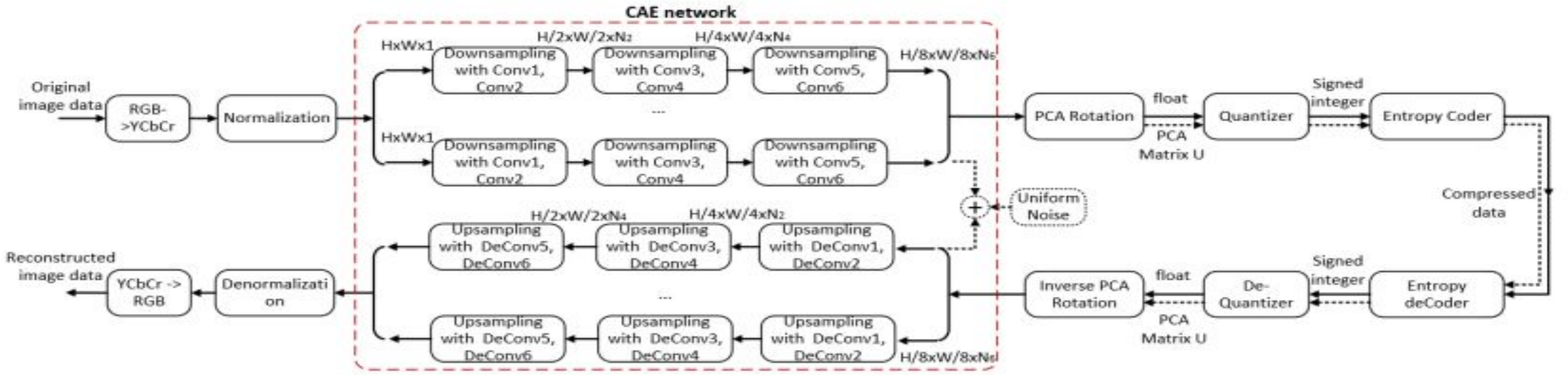
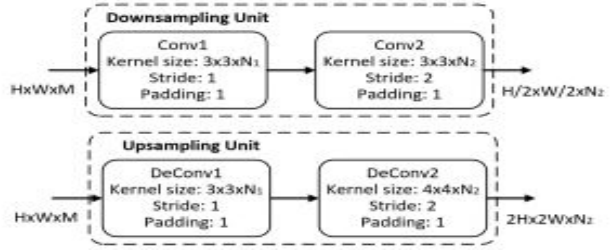


Fig. 1: Block diagram of the proposed CAE based image compression. (The detailed block for downsampling/upsampling is shown in Fig. 2)



Downsampling Unit
HxWxM
Conv1 Kernel size: 3x3xN1 Stride: 1 Padding: 1
Conv2 Kernel size: 3x3xN2 Stride: 2 Padding: 1
H/2xW/2xN2
Upsampling Unit
HxWxM
DeConv1 Kernel size: 3x3xN1 Stride: 1 Padding: 1
DeConv2 Kernel size: 4x4xN2 Stride: 2 Padding: 1
2Hx2WxN2

Fig. 2: Downsampling/Upsampling Units with two (De)Convolution Filters.

Explanation:

1) The RGB image is converted into YCbCr form a and then normalized to get the values in the range [0,1].

2) This normalized data is sent to the CAE network where we have 3 convolutional layers to downsample the data.

3) As for the activation function in each convolution layer,we utilize the Parametric Rectified Linear Unit (PReLU)

4) by analyzing the produced feature maps from the pre-trained CAE, we utilize the PCA rotation to produce more zeros for improving the coding efficiency further.

5) Subsequently, quantization and entropy coder are used to compress the rotated feature maps to generate the compressed bitstream.

6) We train this CAE greedily using an RD loss function, which looks like this:

$$\begin{aligned} J(\theta,\phi;x) &= ||x-\hat{x}||^2 + \lambda \cdot ||y||^2 \\ &= ||x - g_\phi(f_\theta(x)+\mu)||^2 + \lambda \cdot ||f_\theta(x)||^2 \end{aligned}$$

- It achieves a 13.7% BD-rate decrement compared to JPEG2000 on the Kodak database images.

PCA (Principal Component Analysis)

- Standardize the range of continuous initial variables
- Compute the covariance matrix to identify correlations
- Compute the eigenvectors and eigenvalues of the covariance matrix to identify the principal components
- Create a feature vector to decide which principal components to keep
- Recast the data along the principal components axes

Convolutional Neural Network

The convolutional neural network, or CNN for short, is a specialized type of neural network model designed for working with two-dimensional image data.

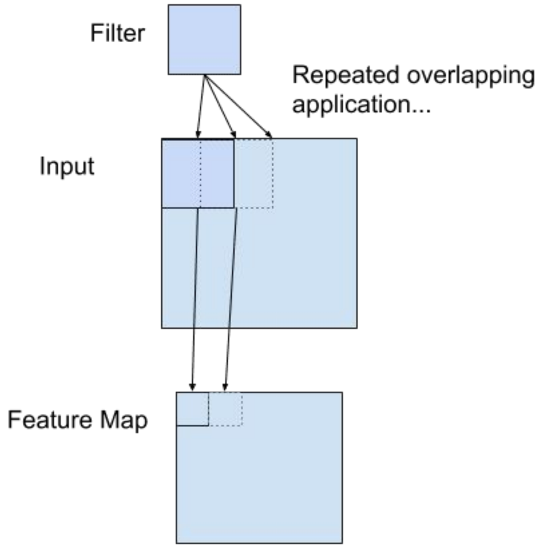
Different types of CNN:

1) LeNet

2) AlexNet

3) ResNet

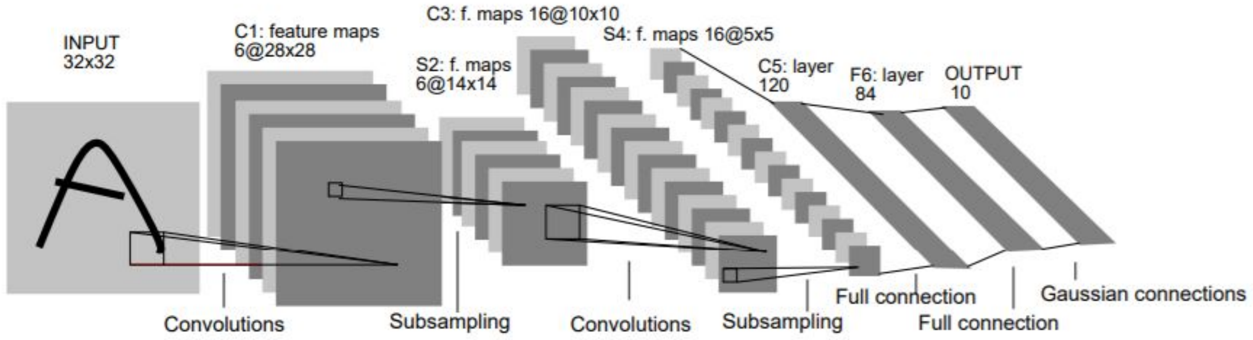
4) Super resolution CNN



Les Net

It has 2 convolutional layers, 2 sampling layers using average sampling, 2 fully connected layer and an output layer to classify the output .

Activation method used is Relu.



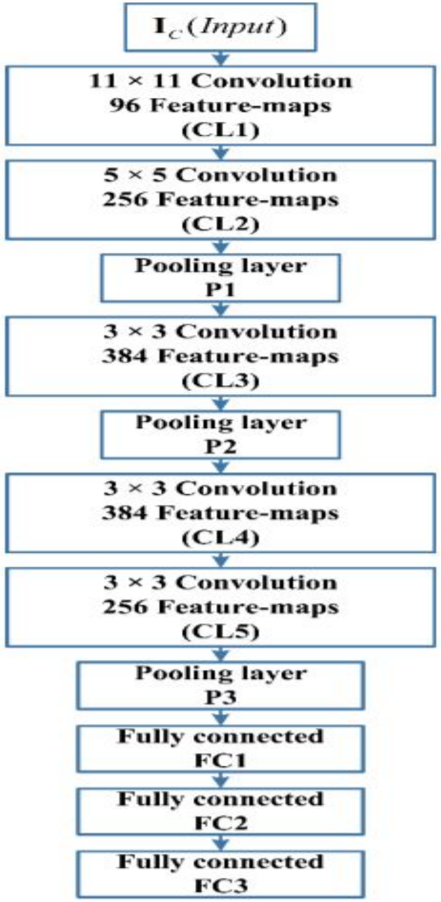
AlexNet

Starting with an 11x11 kernel,

It has 5 conv layers.

3 max-pooling layers, ReLu activation functions.

The network was used to classify images into 1000 different categorie



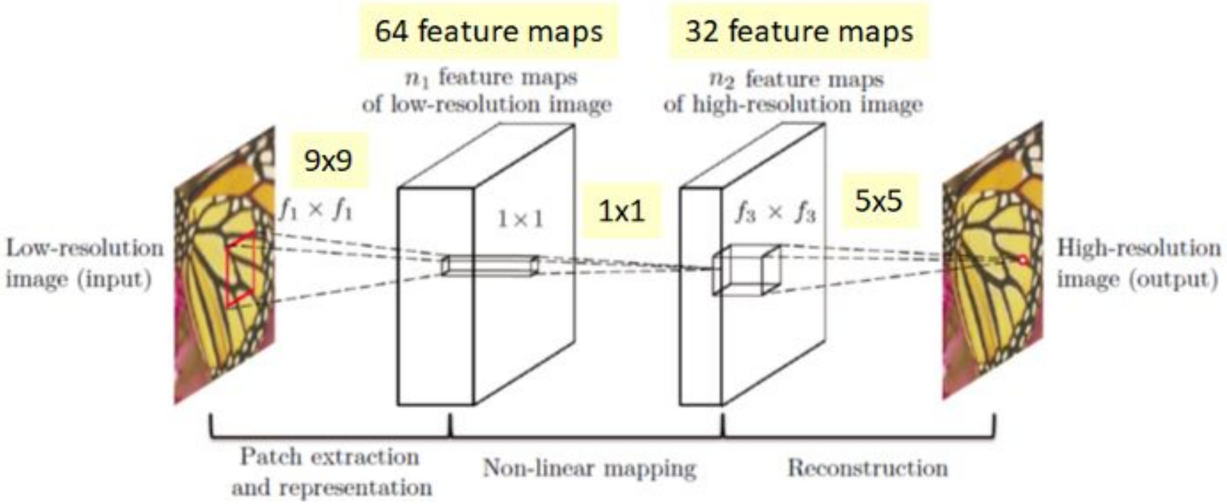
Convolutional Neural Network (CNN)

Super-Resolution CNN (SRCNN) [4]

The SRCNN consists of the following operations:

1. **Preprocessing**: Up-scales LR image to desired HR size.
2. **Feature extraction**: Extracts a set of feature maps from the up-scaled LR image.
3. **Non-linear mapping**: Maps the feature maps representing LR to HR patches.
4. **Reconstruction**: Produces the HR image from HR patches.

In [7] , author used SRCNN to improve the resolution of the image.


64 feature maps
n_1 feature maps of low-resolution image
32 feature maps
n_2 feature maps of high-resolution image
9x9
$f_1 \times f_1$
1×1
1x1
$f_3 \times f_3$
5x5
Low-resolution image (input)
High-resolution image (output)
Patch extraction and representation
Non-linear mapping
Reconstruction

SRCNN comprises three convolutional blocks corresponding to patch extraction, non-linear mapping, and reconstruction.

- The first layer convolves the input image with a 9x9 kernel with padding and expands the three-channel image into 64 feature maps.
- The second layer applies a 1x1 kernel to condense to 32 feature maps.
- The third layer applies a 5x5 kernel to generate the output image.
- Both the first and second convolutional layers are succeeded by a ReLU activation function.

- The 9x9 kernels of the first SRCNN layer apply the same 81 parameters (values for each pixel of the kernel) to a full image, such that each pixel of the resulting image corresponds to the level of similarity between the corresponding 9x9 pixel patch of the input image and that kernel.
- The nonlinear mapping step connects these input feature maps to a reduced set of output feature maps.
- The 5x5 kernels can be viewed as higher-resolution/granularity image features represented by the pixels of the output feature maps.
- Loss function used is MSE and the metric to check for the image quality used is peak-signal-to-noice-ratio(PSNR)

Photo-Realistic Single Image Super-Resolution using a GAN

- Super resolution enhances resolution of an image
- Uses Generative Adversarial Network which is a Generator and Discriminator architecture.
- Original paper that introduced GANs for Super Resolution

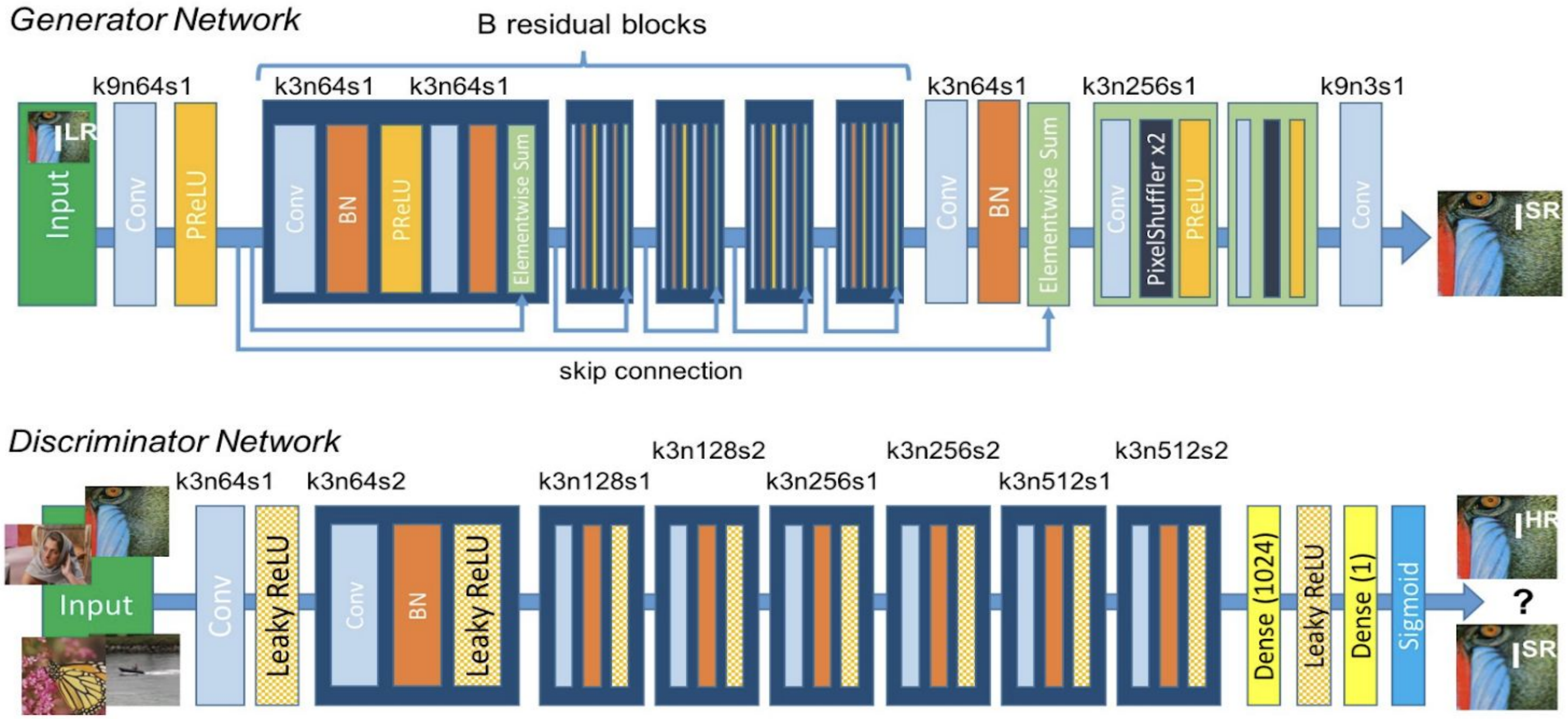
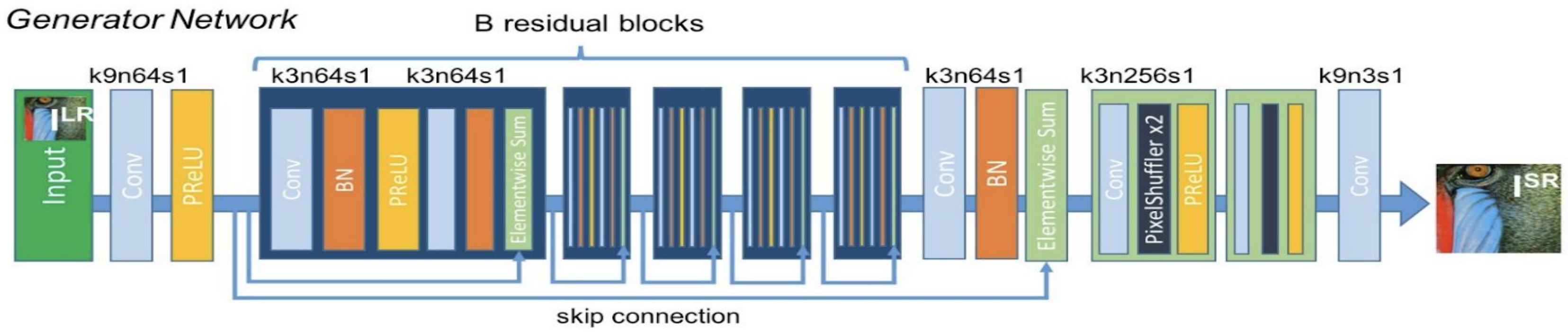
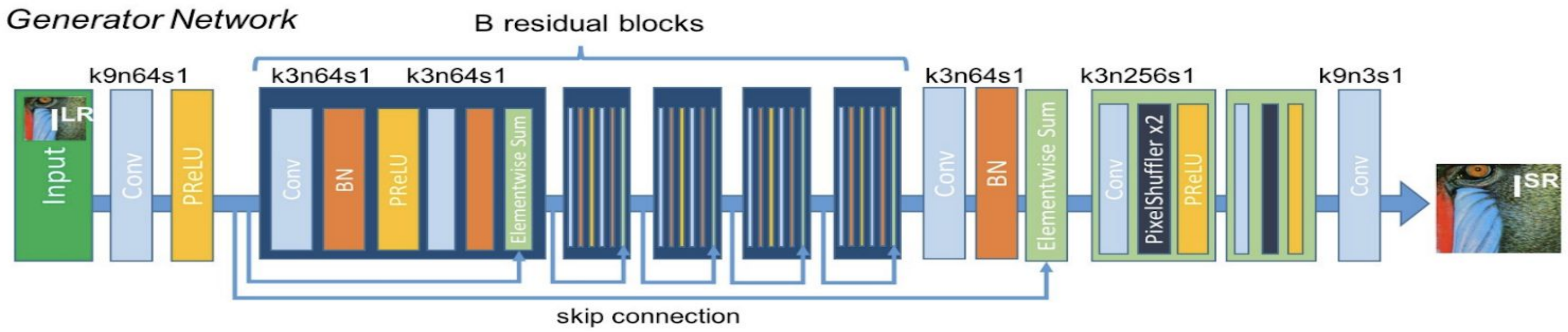


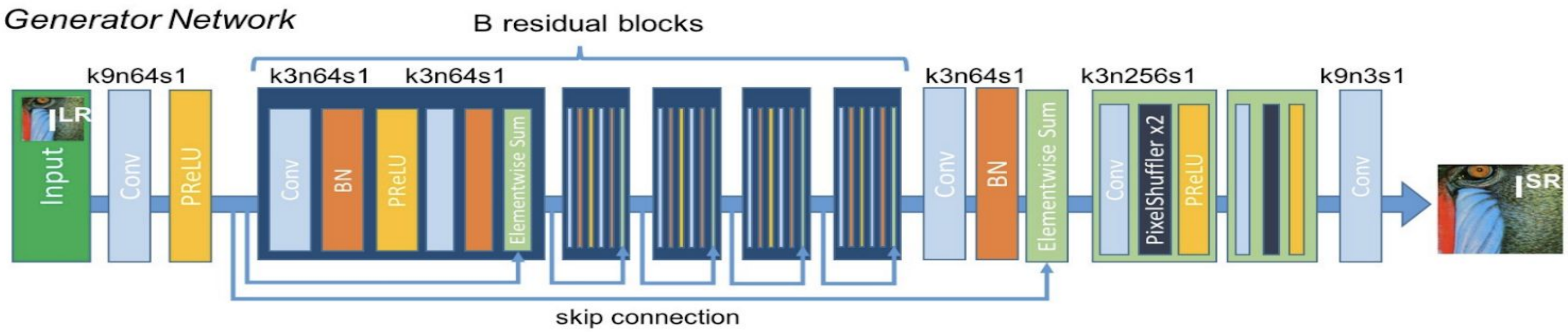
Figure 4: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer.



- First convolutional layer with 64 filters of 9x9 size
- Use Parametric ReLU for activation function. In PReLU the network choses the constant fraction on it’s own for negatives
- Doesn’t have max pooling layer as we don’t want to downscale. Rather, upscaling is the goal
- Uses multiple residual blocks with skip connection
- Each residual block extracts features from images

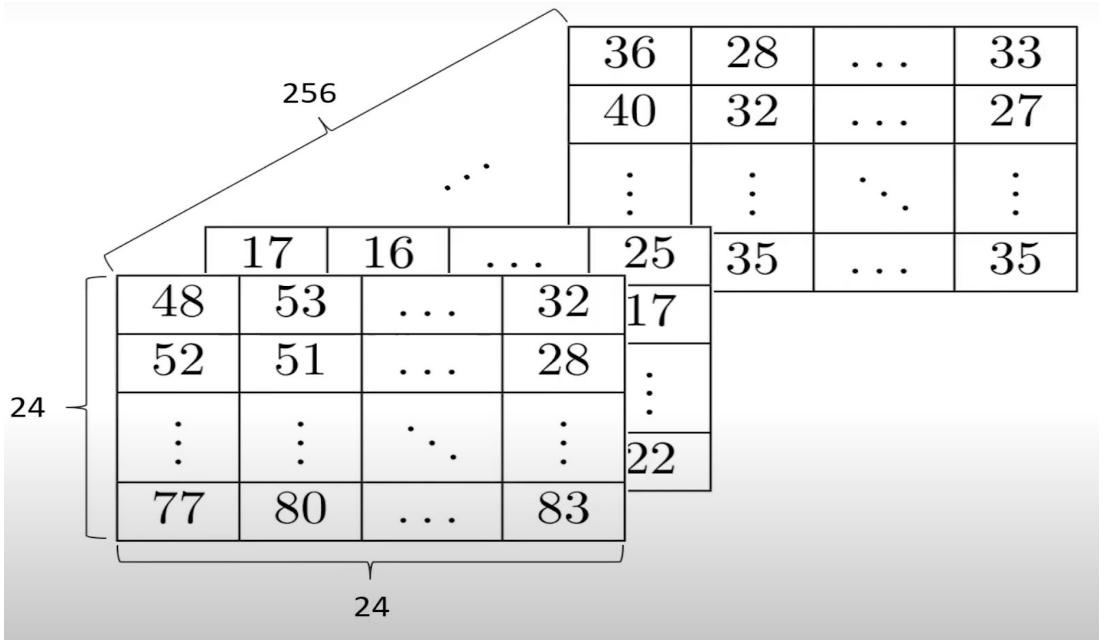


- Uses SRResNet Architecture
- Each Residual Block(RB) consists of convolutional layers, batch normalization layer, followed by PReLU layer
- At the end of each RB there is Elementwise Sum layer which adds the input of RB to output of RB
- The skip connection helps with degradation problem problem.
- Let h(x) = g(x)+x, layers with skip connections. Here +x term denotes the skip connection. Layer g(x) has to learn just the changes in the value
- After B-RB there is Convolution Layer, Batch Normalization Layer and another Elementwise Sum Layer which adds initial input to the output



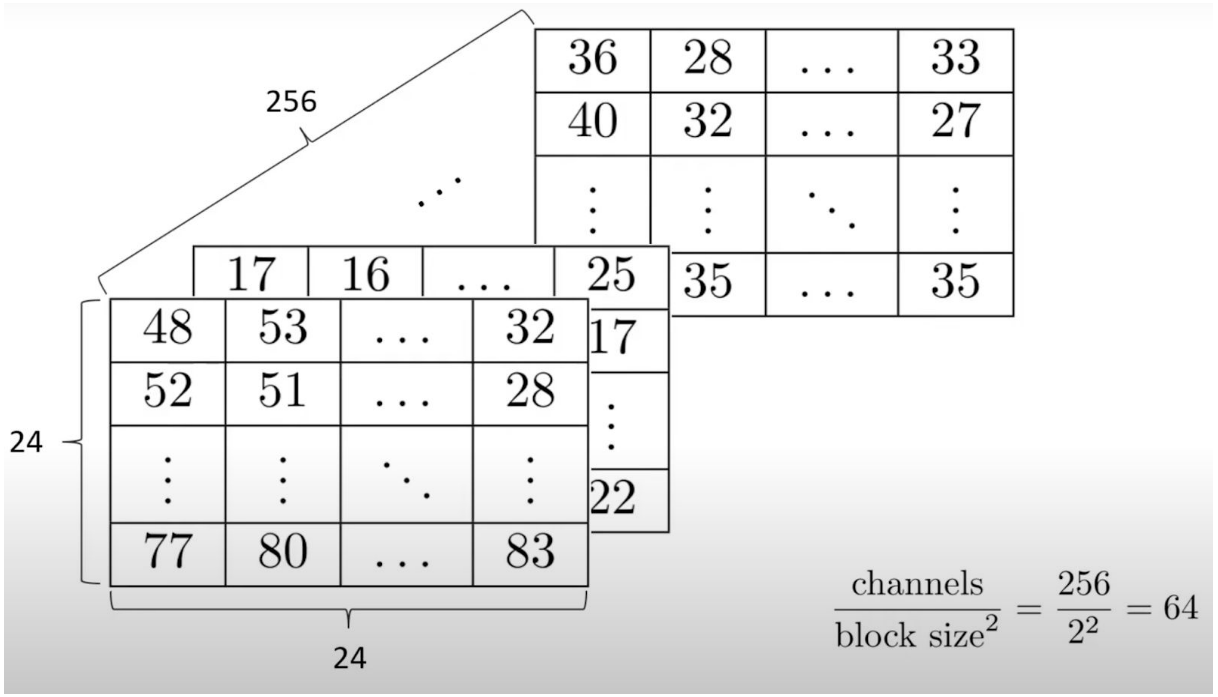
- After this we have upscaling block
- Increase width and height of image
- Has 3 layers; Conv, PixelShuffler and PReLU
- There are 2 upsampling blocks

Pixel Shuffler



- Activation map of 24x24
- 256 Channels

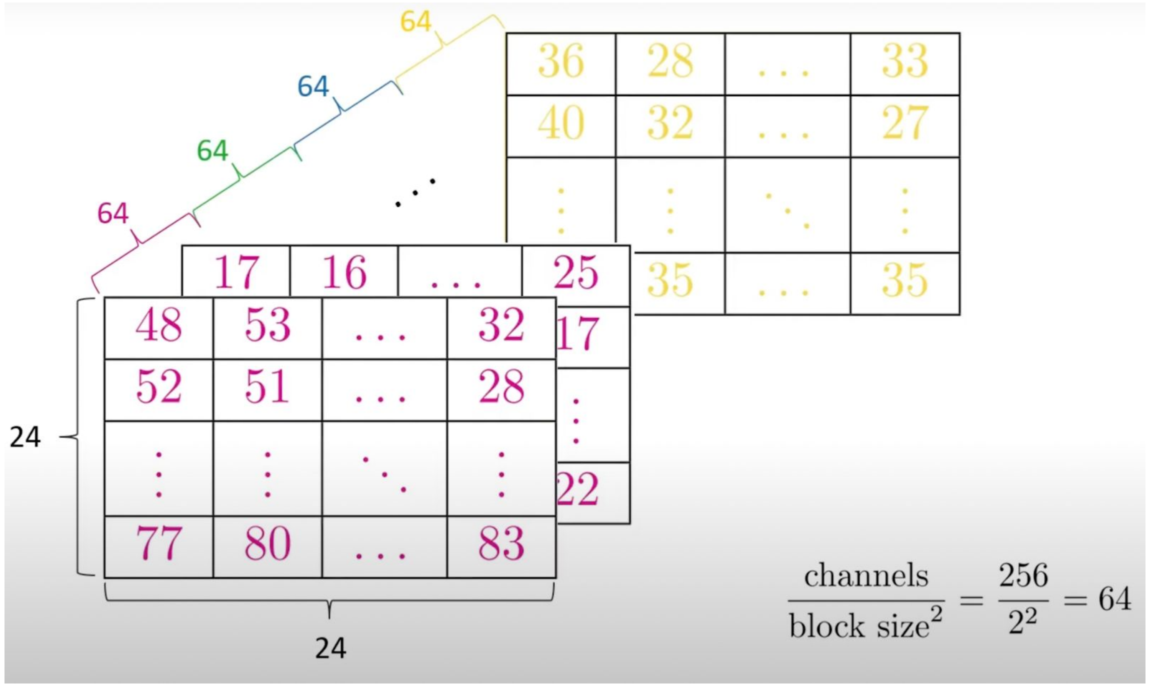
Pixel Shuffler



$$\frac{\text{channels}}{\text{block size}^2} = \frac{256}{2^2} = 64$$

- Block size is 2
- We get 64

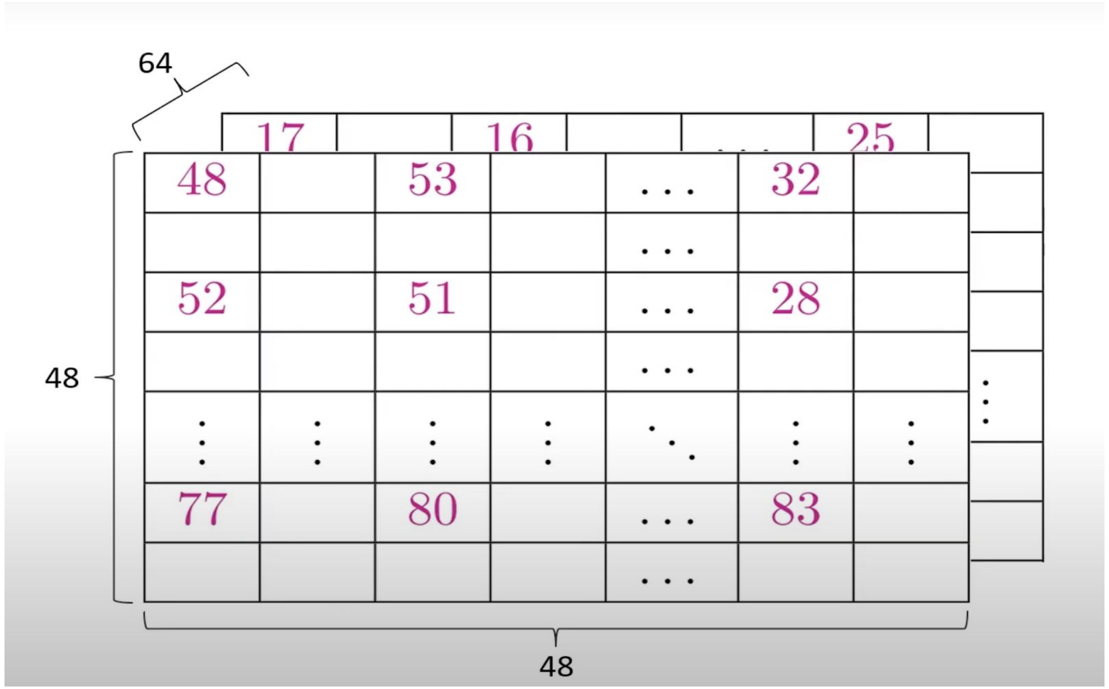
Pixel Shuffler



$$\frac{\text{channels}}{\text{block size}^2} = \frac{256}{2^2} = 64$$

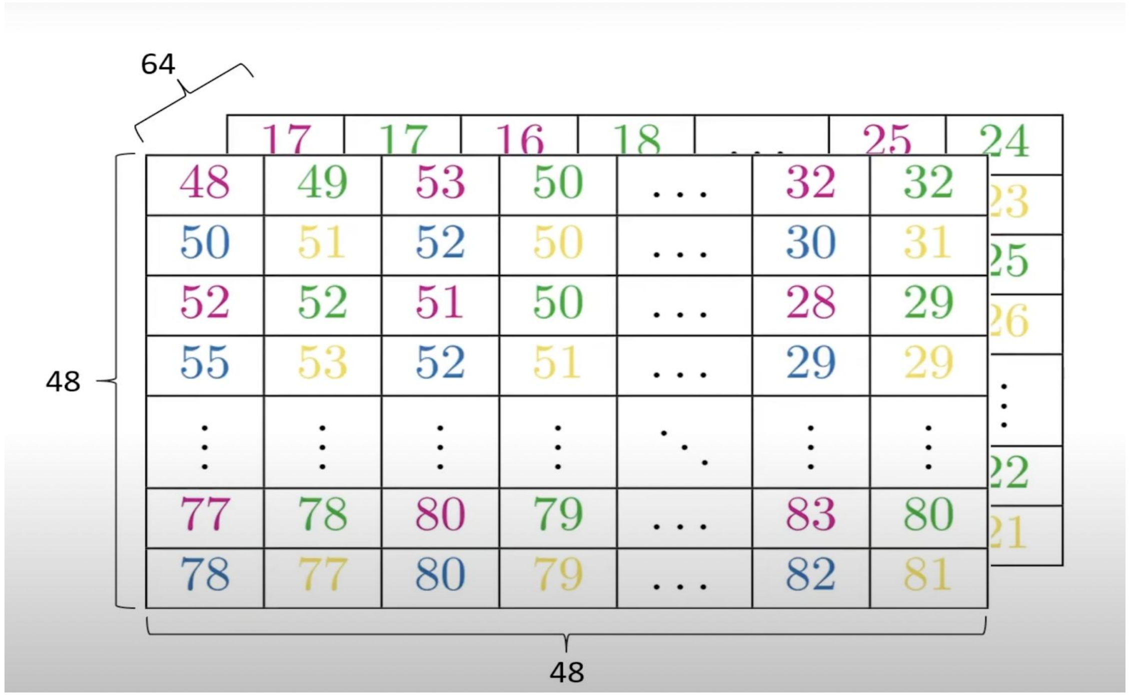
- Divide the channel into groups of 64

Pixel Shuffler



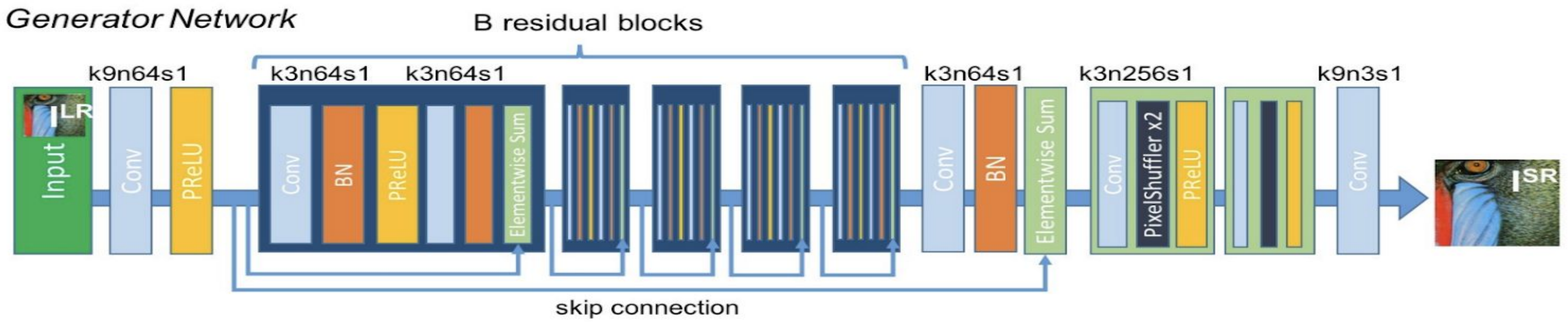
- Block size is 2
- Double the width and height dimensions of activation map

Pixel Shuffler

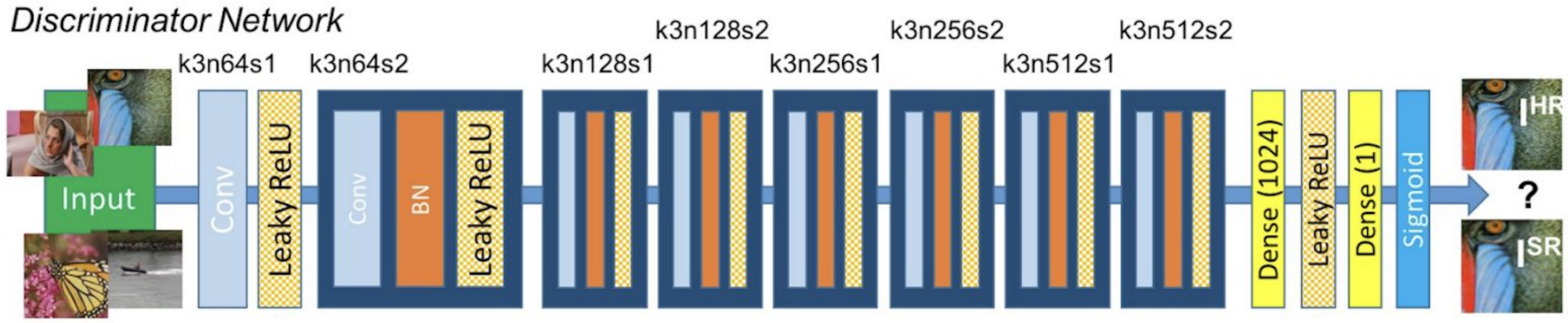


48	49	53	50	. . .	32	32
50	51	52	50	. . .	30	31
52	52	51	50	. . .	28	29
55	53	52	51	. . .	29	29
⋮	⋮	⋮	⋮	⋱	⋮	⋮
77	78	80	79	. . .	83	80
78	77	80	79	. . .	82	81

- Fill gaps with values from other channels



- Finally we have an image whose width and height have become 4 times



- Uses Leaky ReLU instead of PReLU
- Uses image classification CNN
- Compares HR image with SR image

Loss Function

- Uses Perceptual Loss
- Perceptual loss looks at how image is perceived instead of comparing pixel by pixel
- Perceptual loss = content Loss + Adversarial Loss
- MSE and Peak Signal to Noise Ratio does not capture perceptual difference such as high texture detail
- Uses high level feature map of VGG

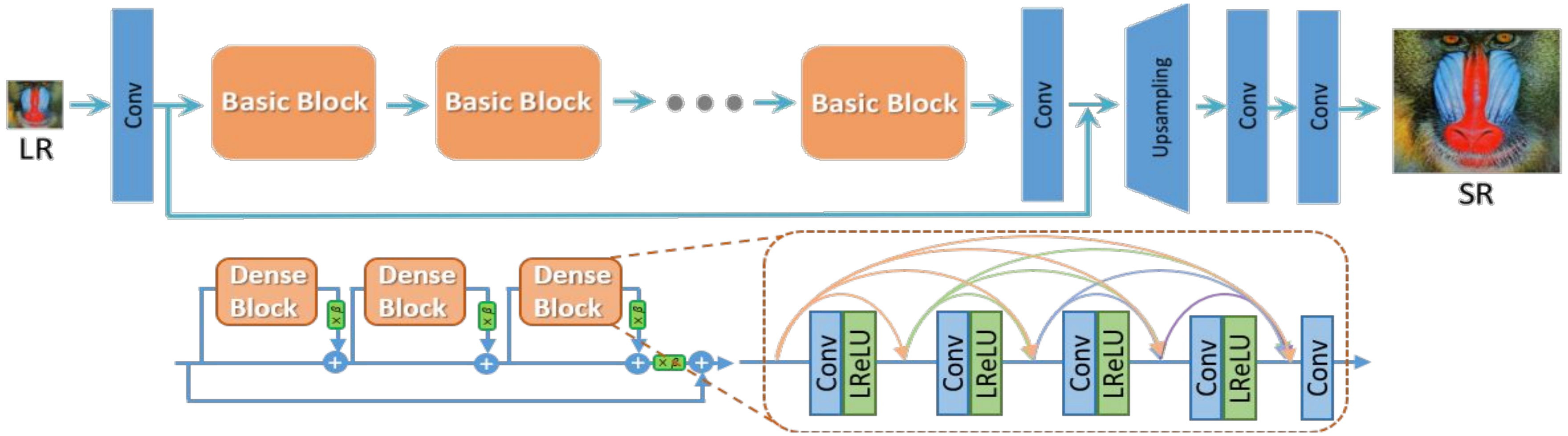
Loss Function

$$l^{SR} = \underbrace{\underbrace{l^{SR}_{X}}_{\text{content loss}} + \underbrace{10^{-3} l^{SR}_{Gen}}_{\text{adversarial loss}}}_{\text{perceptual loss (for VGG based content losses)}}$$

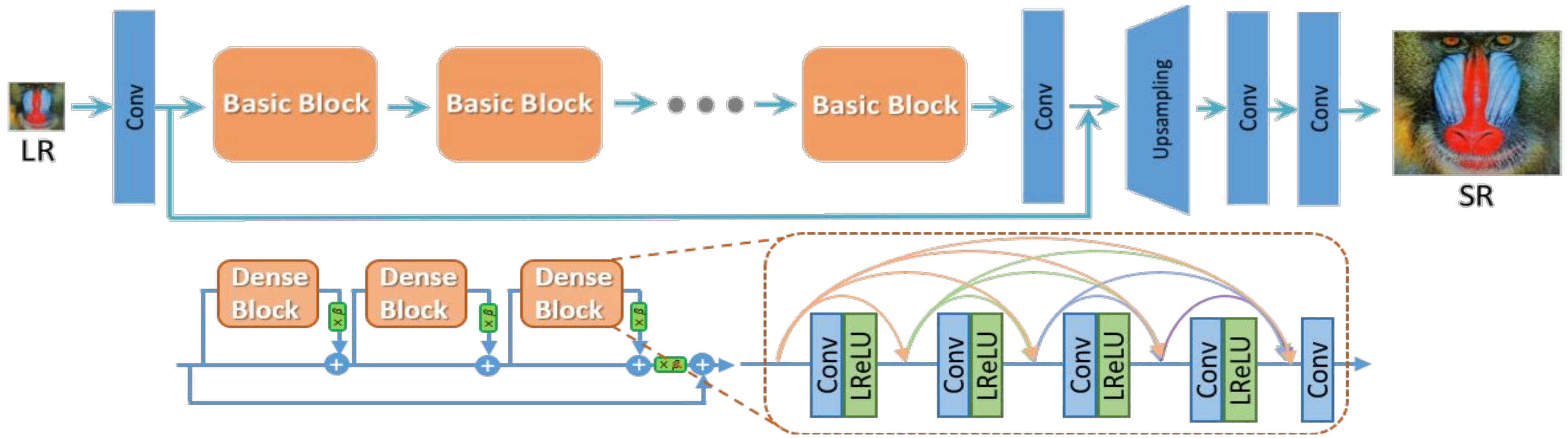
- VGG loss based on the ReLU activation layers of the pre-trained 19 layer VGG.
- Define VGG loss as the euclidean distance between the feature representations of a reconstructed image and the reference image
- Adversarial loss pushes neural network to the natural image using a discriminator network that is trained to differentiate between the super-resolved images and original photo-realistic images

ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks

- Improve the perceptual quality of Single Image Super-Resolution
- Some changes to SRGAN architecture



- Replaces original basic blocks with a proposed residual-in-residual dense block
- Follows DenseNet architecture and connects all layers within the residual block directly with each other



- No Batch Normalization in residual block
- No BN layers increases performance and reduce the computational complexity
- Because statistics of each layer are very different for every image and also for test images
- Against the assumption of BN that the features for train/test images are similar

Real-ESRGAN

- Extension of ESRGAN
- Synthesizes training pairs with a more practical degradation mechanism to recover general real-world low-resolution pictures
- Real-ESRGAN is able to repair most real-world photos and produce superior visual performance than prior works

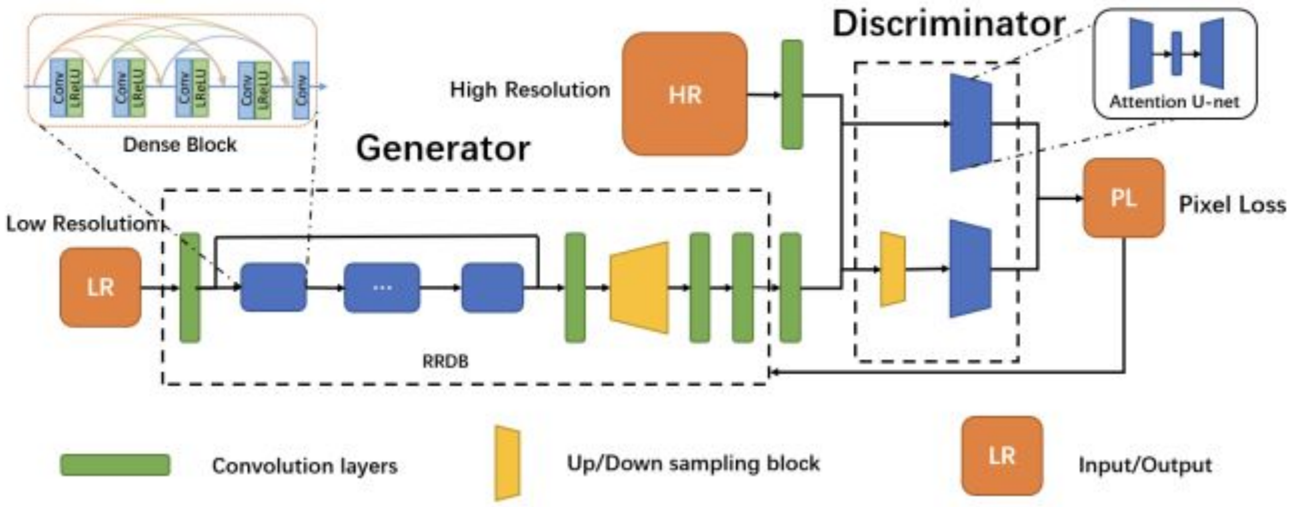
Blind Super-Resolution

- Image reconstruction is to obtain a high-resolution image from a sequence of low-resolution images

- Low-resolution images are degraded by unknown blur, noise, and down sample

A-ESRGAN: Training real-world blind super-resolution with attention U-net discriminators

- Most work has focused on generator
- Simulating a more complex and realistic degradation process
- This architecture focuses on discriminator architecture
- Discriminator provides the generator the direction to generate better images
- Can be added to any generator architecture



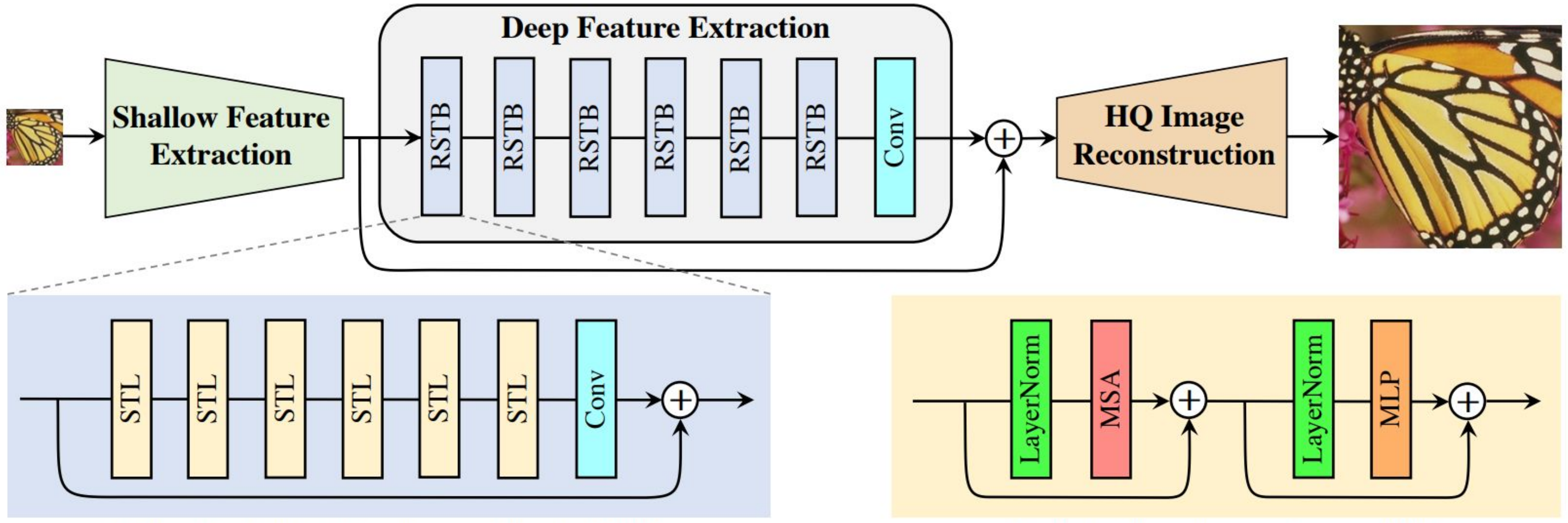
- Uses residual in residual dense block as generator
- Uses Attention U-net structure in discriminator
- 2 discriminators with identical network structure but operate at different image scales, original and 2 X downsampled image
- U-Net Structure ito provide per-pixel feedback to the generator
- Attention layer to distinguish the outline of the subject area

- The normal image discriminator emphasizes more on lines.
- Downsampled input images with blurred edges force the other discriminator to focus more on larger patches

Transformer

- Self-attention mechanism
    - Learns relationships between elements of a sequence
    - Captures long-range dependency
    - Aggregate global information from complete input sequence
- Pre-trained on large-scale datasets
    - Finetune to specific tasks

SwinIR



(a) Residual Swin Transformer Block (RSTB)

(b) Swin Transformer Layer (STL)

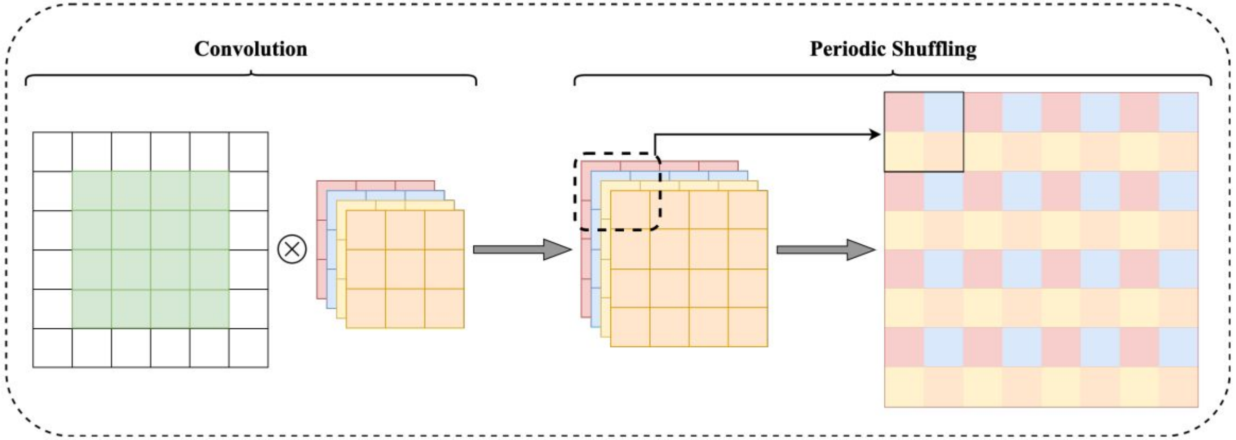
SwinIR

- Shallow features contain low-frequencies
- Deep features focus on restoring high-frequencies
- Local attention and shifted window mechanism
- Skip connection directly from shallow feature extraction to reconstruction module
- Multihead self-attention (MSA) and Multi-layer perceptron (MLP) in each Swin Transformer layer

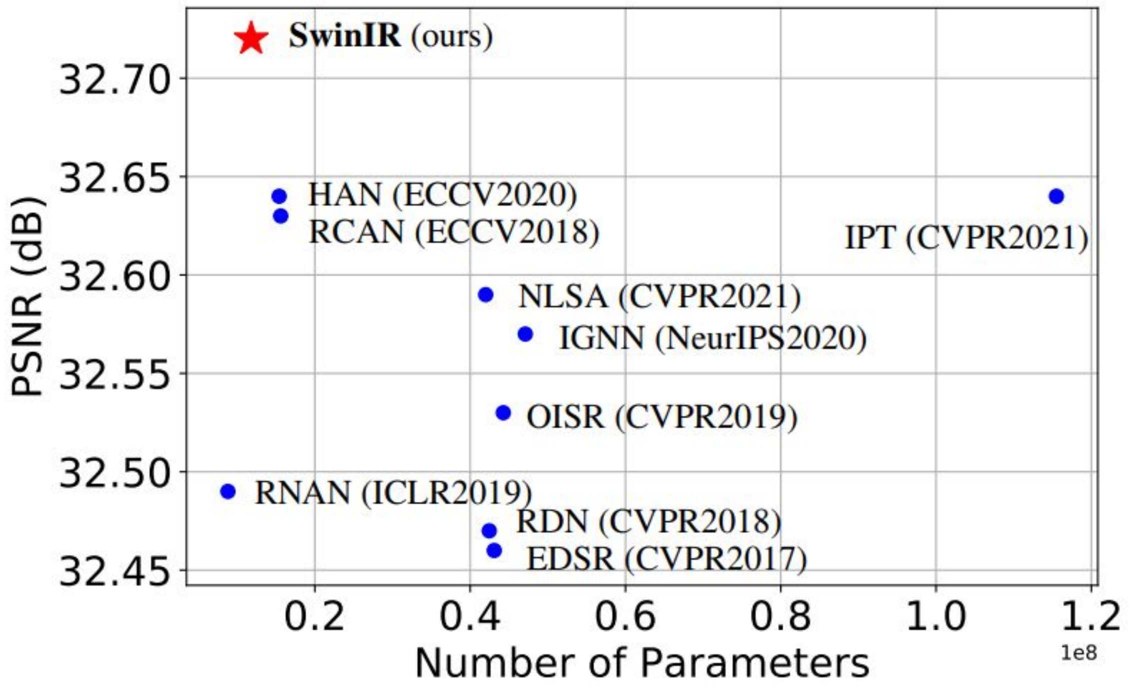
Reconstruction

Upsampling method: sub-pixel convolution layer

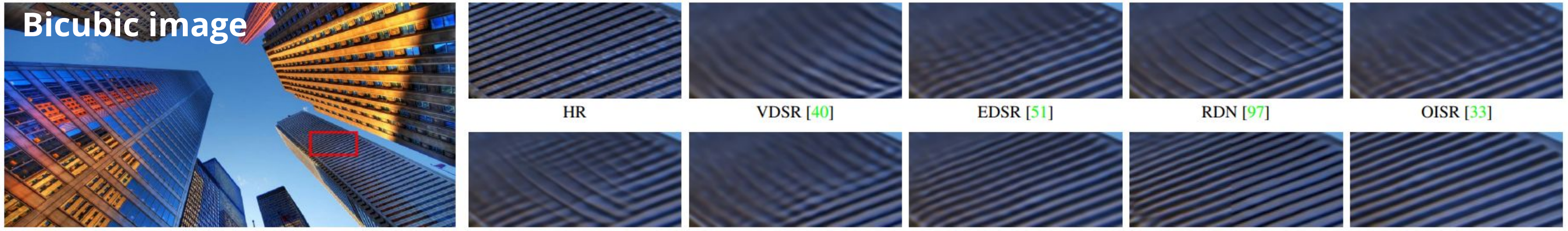
Loss function: L_1 pixel loss, GAN loss, perceptual loss



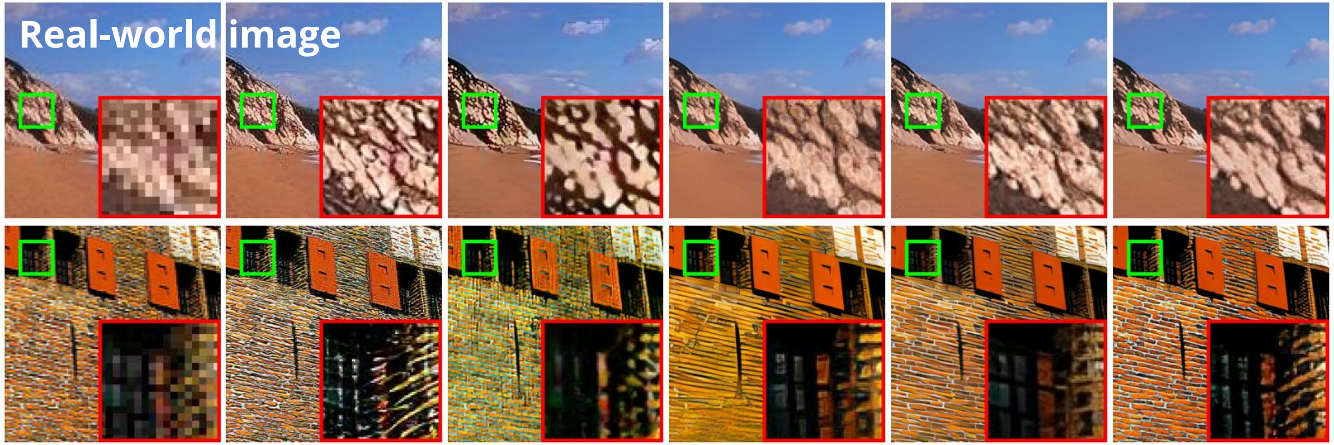
Comparisons



Comparisons



Urban100 (4×):img_012



Proposed Method

1. Train a Convolutional Autoencoder (CAE)
2. Obtain new training dataset
3. Transfer-learning on pre-trained SwinIR model on new dataset
4. Compress images through CAE + restore the output through new SwinIR model
5. Evaluate results

CAE Compression Model Results

Images	JPEG_PSNR	CAE_PSNR
img1	25.133571730052722	68.80298894094759
img2	25.133571730052722	72.0766678288864
img3	23.492463587258207	71.06848423751096
img4	23.782298740309713	69.24729347565386
img5	24.498634719616142	70.9987425569537

Images	JPEG_SSIM	CAE_SSIM
img1	0.7324647203460863	0.9985165564310678
img2	0.8404630172197501	0.9985165564310678
img3	0.9898393374143969	0.9898393374143969
img4	0.6912463978940243	0.9980587167189803
img5	0.7678400774761748	0.9957950897580261

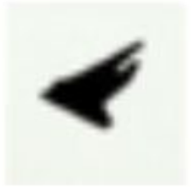
Transfer-Learning on SwinIR Results

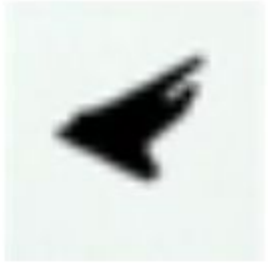
Dataset - 60 high-resolution images
- 50 Training
- 10 Validation

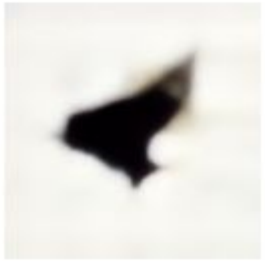
Training
- 5 epochs
- Adam optimizer

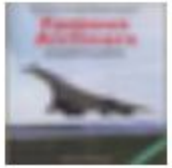
Epoc	AVG PSNR
1	24.309989679055892
2	26.74641404859804
3	26.69878608011634
4	27.317388404477526
5	26.18571235406673

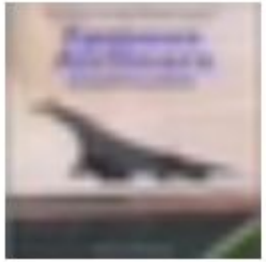
Visual Results on Restoration

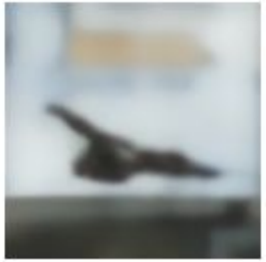

Original Image


JPEG Compression


CAE Compression + SWIN IR Restoration


Original Image


JPEG Compression


CAE Compression + SWIN IR Restoration

References

[1] V. Punj, “Whatsapp update on ios let’s beta users send HD photos and how it works.” https://www.livemint.com/technology/apps/whatsapp-update-on-ios-lets-beta-users-send-hd-photos-how-it-works-11627203919427.html, 2021.

[2] R. Y. Tsai and T. S. Huang. “Multipleframe image restoration and registration.” In Advances in Computer Vision and Image Processing, pages 317–339. Greenwich, CT: JAI Press Inc., 1984.

[3] Sean Borman and Robert L. Stevenson. “Super-resolution from image sequences - A review.” In Proceedings of the 1998 Midwest Symposium on Circuits and Systems, pages 374–378, 1998.

[4] Dong, C., Loy, C.C., He, K., et al. “Learning a deep convolutional network for image super-resolution.” *European Conf. on Computer Vision*, Zurich, Switzerland, 2014, pp. 184–199

[5] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, 2016.

[6] A. Vaswani et al. “Attention is all you need.” *arXiv preprint arXiv:1706.03762*, 2017.

[7] B. Garber et at., “Image Super-Resolution Via a Convolutional Neural Network”, http://cs229.stanford.edu/proj2020spr/report/Garber_Grossman_Johnson-Yu.pdf

[8] Wei, Zihao, et al. "A-ESRGAN: Training Real-World Blind Super-Resolution with Attention U-Net Discriminators." in *arXiv preprint arXiv:2112.10046*, 2021. Available: https://arxiv.org/abs/2112.10046

[9] J. Liang, J. Cao, G. Sun, K. Zhang, L. V. Gool, and R. Timofte. “SwinIR: Image Restoration Using Swin Tranformer,” in *arXiv:2108.10257v1 (eess.IV)* 23 Aug 2021. [Online]. Available: https://arxiv.org/abs/2108.10257