Deep Learning en Imágenes con GANs y Modelos de Difusión - Parte III

Prof. Peter Montalvo

Contenido

- SR-GAN
- Función de Pérdida
- Función de Pérdida del Discriminador
- Pixel Shuffle
- Parametrized RELu

SRGAN

- SR-GAN

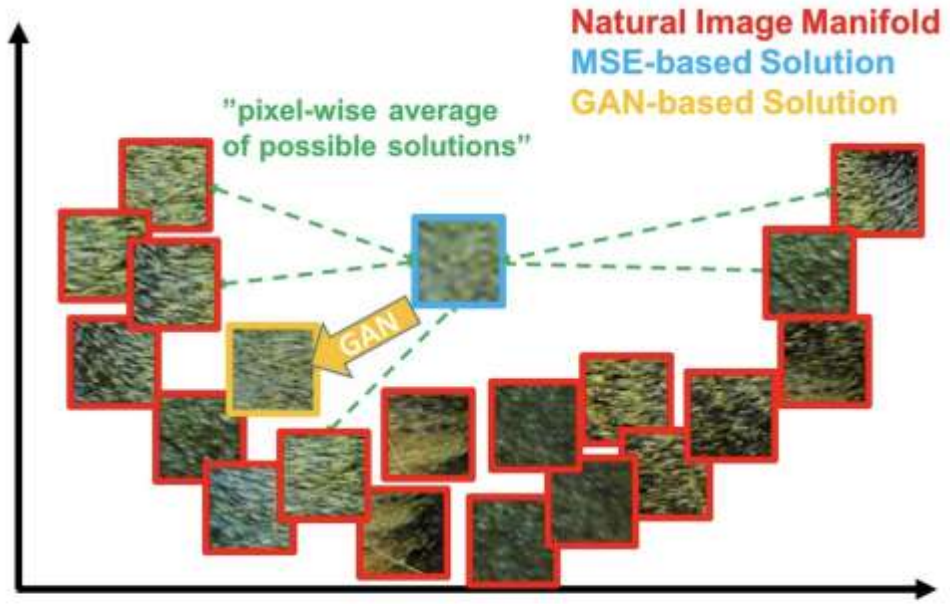
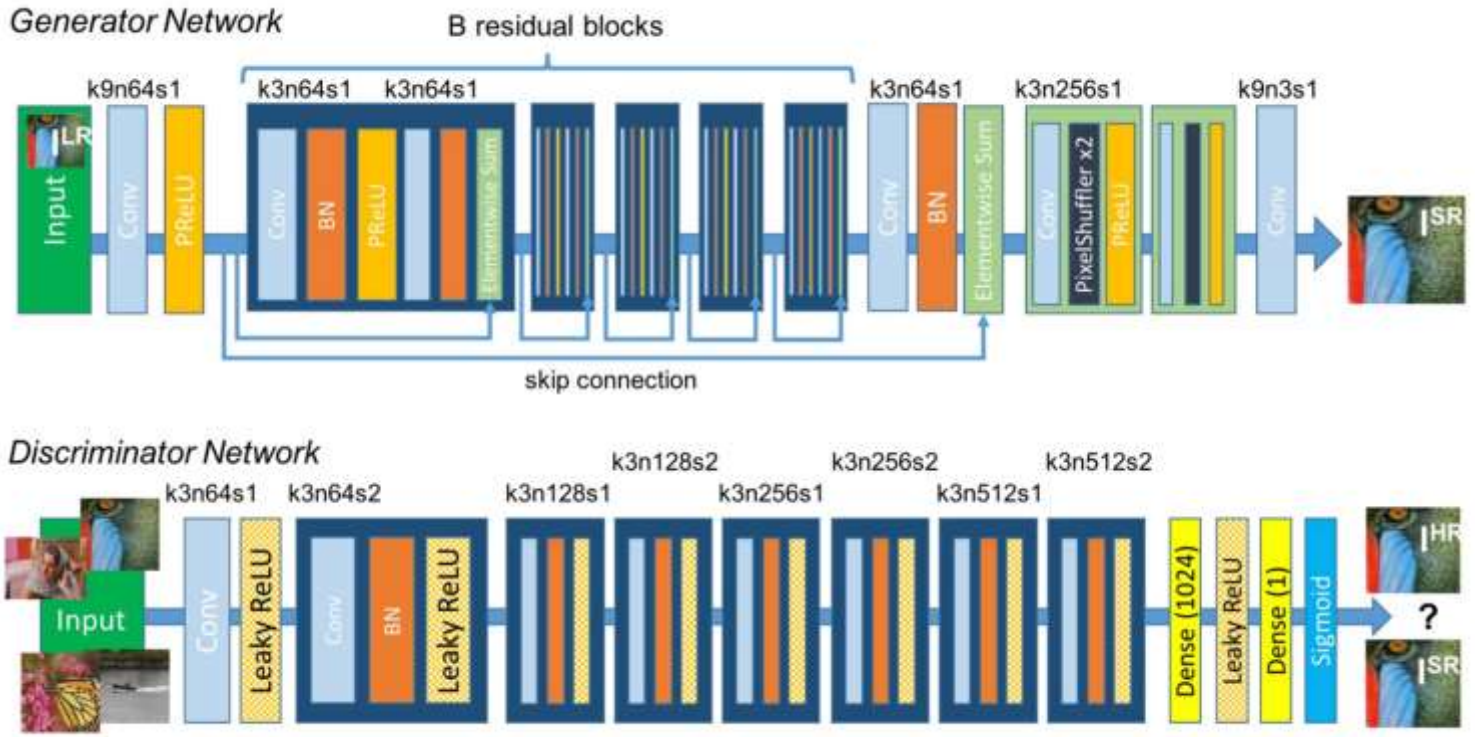
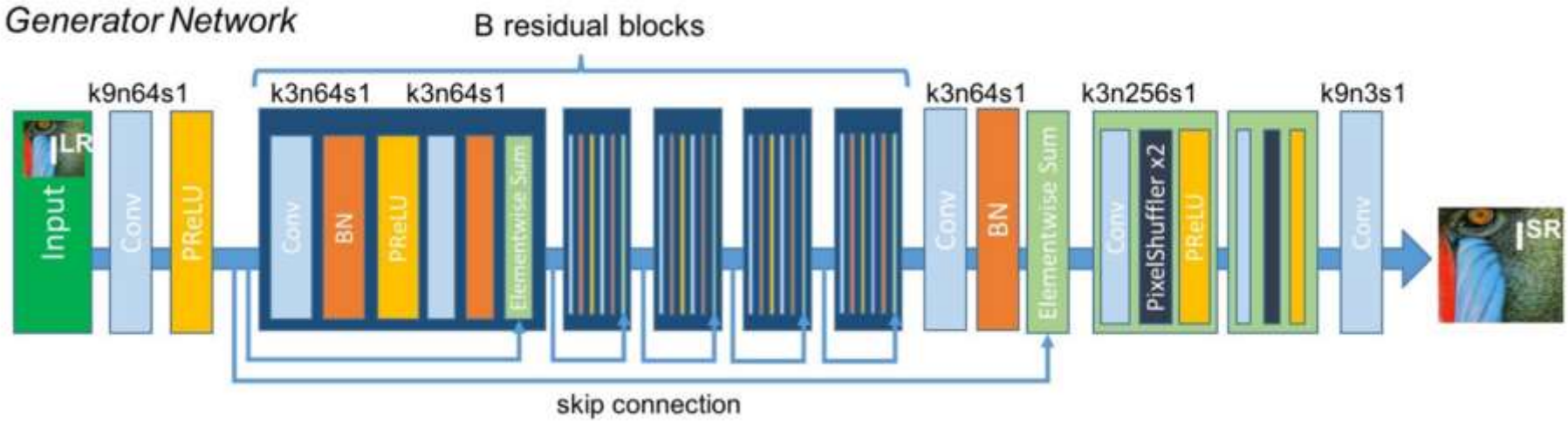


Figure 3: Illustration of patches from the natural image manifold (red) and super-resolved patches obtained with MSE (blue) and GAN (orange). The MSE-based solution appears overly smooth due to the pixel-wise average of possible solutions in the pixel space, while GAN drives the reconstruction towards the natural image manifold producing perceptually more convincing solutions.

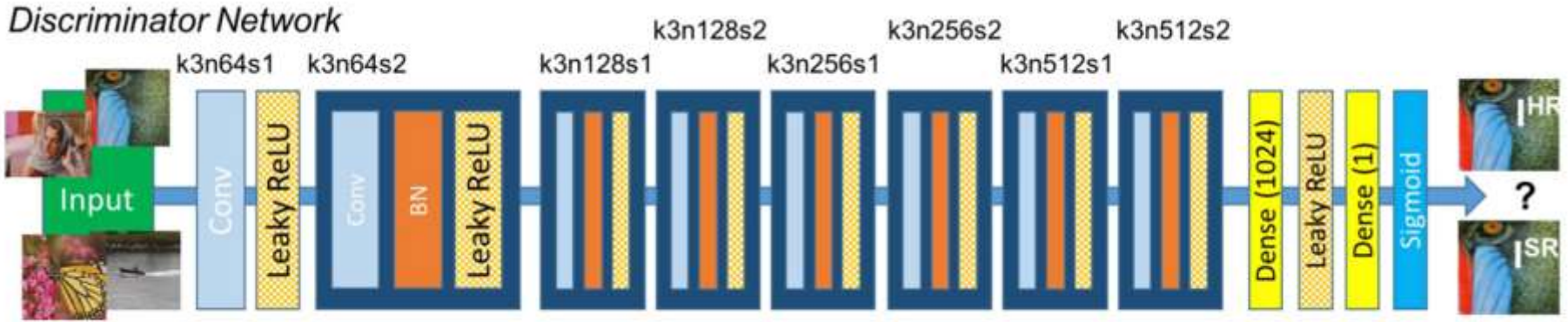
SRGAN


Generator Network
B residual blocks
k9n64s1
k3n64s1
k3n64s1
k3n64s1
k3n256s1
k9n3s1
LR
Input
Conv
PReLU
Conv
BN
PReLU
Elementwise Sum
Conv
BN
Elementwise Sum
Conv
PixelShuffler x2
PReLU
Conv
SR
skip connection
Discriminator Network
k3n64s1
k3n64s2
k3n128s1
k3n128s2
k3n256s1
k3n256s2
k3n512s1
k3n512s2
Input
Conv
Leaky ReLU
Conv
BN
Leaky ReLU
Dense (1024)
Leaky ReLU
Dense (1)
Sigmoid
HR
?
SR

SRGAN


Generator Network
B residual blocks
k9n64s1
k3n64s1
k3n64s1
k3n64s1
k3n256s1
k9n3s1
ILR
Input
Conv
PReLU
Conv
BN
PReLU
Elementwise Sum
Conv
BN
Elementwise Sum
Conv
PixelShuffler x2
PReLU
Conv
ISR
skip connection

SRGAN


Discriminator Network
k3n64s1
k3n64s2
k3n128s1
k3n128s2
k3n256s1
k3n256s2
k3n512s1
k3n512s2
Input
Conv
Leaky ReLU
Conv
BN
Leaky ReLU
Dense (1024)
Leaky ReLU
Dense (1)
Sigmoid
I^HR
?
I^SR

SRGAN - Función de Pérdida

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

SRGAN - Función de Pérdida

$$l_{MSE}^{SR} = \frac{1}{r^2 W H} \sum_{x=1}^{rW} \sum_{y=1}^{rH} \left(I_{x,y}^{HR} - G_{\theta_G}(I^{LR})_{x,y} \right)^2$$

Usualmente se calcula la diferencia entre la Imagen en Alta Resolución y el resultado del Generador con la imagen en Baja Resolución

SRGAN - Función de Pérdida

$$l_{MSE}^{SR} = \frac{1}{r^2WH} \sum_{x=1}^{rW} \sum_{y=1}^{rH} (I_{x,y}^{HR} - G_{\theta_G}(I^{LR})_{x,y})^2$$

Este error también es parte del Peak signal-to-noise ratio…

Más info: https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio

SRGAN - Función de Pérdida

$$MSE = \frac{1}{m\,n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2.$$

La relación señal-ruido máxima (PSNR) es un término de ingeniería para la relación entre la potencia máxima posible de una señal y la potencia del ruido corruptor que afecta la fidelidad de su representación.

$$\begin{aligned} PSNR &= 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \\ &= 20 \cdot \log_{10} \left(\frac{MAX_I}{\sqrt{MSE}} \right) \\ &= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE). \end{aligned}$$

Más info: https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio

SRGAN - Función de Pérdida

$$MSE = \frac{1}{m\,n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2.$$

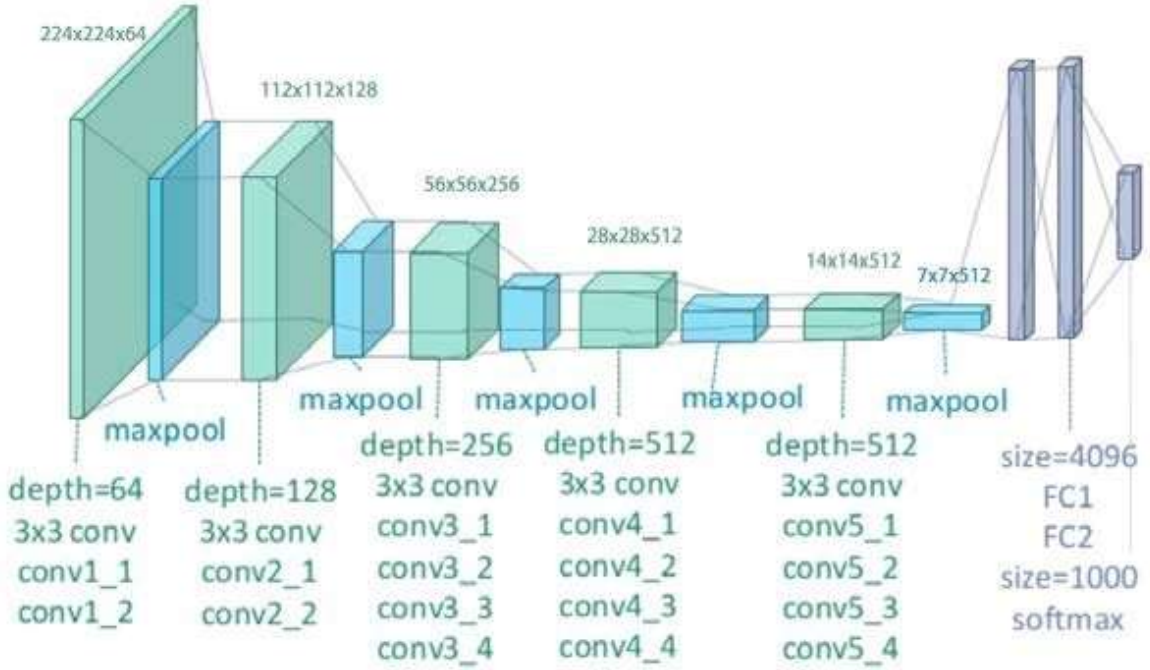
$$\begin{aligned} PSNR &= 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \\ &= 20 \cdot \log_{10} \left(\frac{MAX_I}{\sqrt{MSE}} \right) \\ &= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE). \end{aligned}$$

Debido a que muchas señales tienen un rango dinámico muy amplio, la PSNR generalmente se expresa como una cantidad logarítmica usando la escala de decibeles.

PSNR se usa comúnmente para cuantificar la calidad de reconstrucción de imágenes y videos sujetos a compresión con pérdidas.

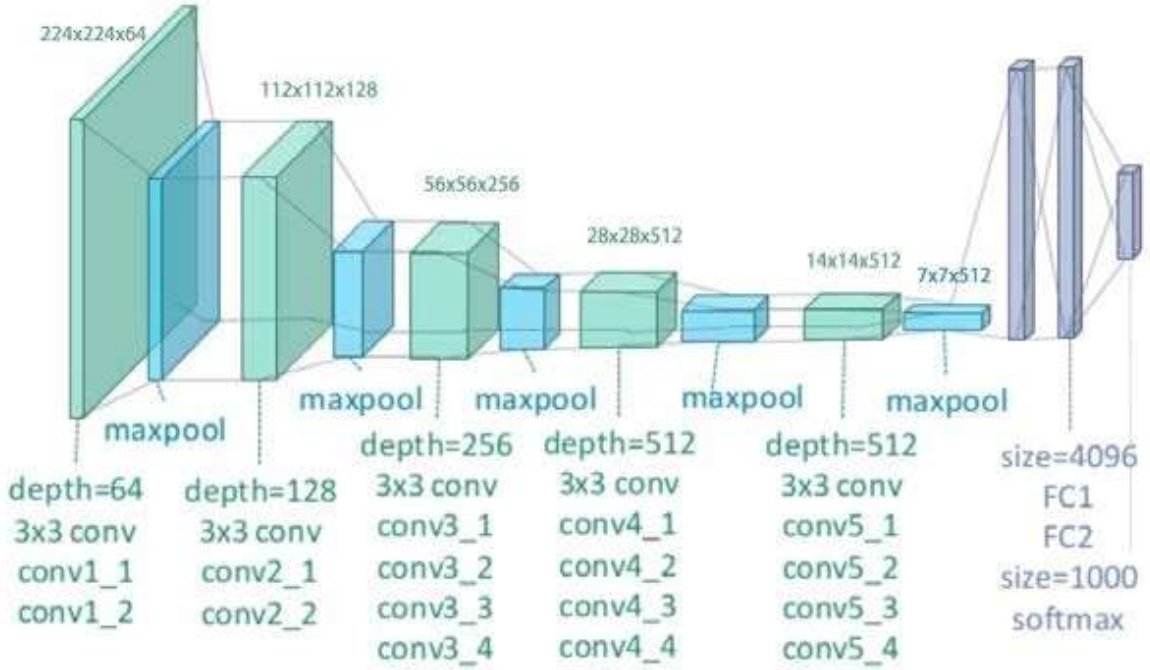
Más info: https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio

SRGAN - Función de Pérdida



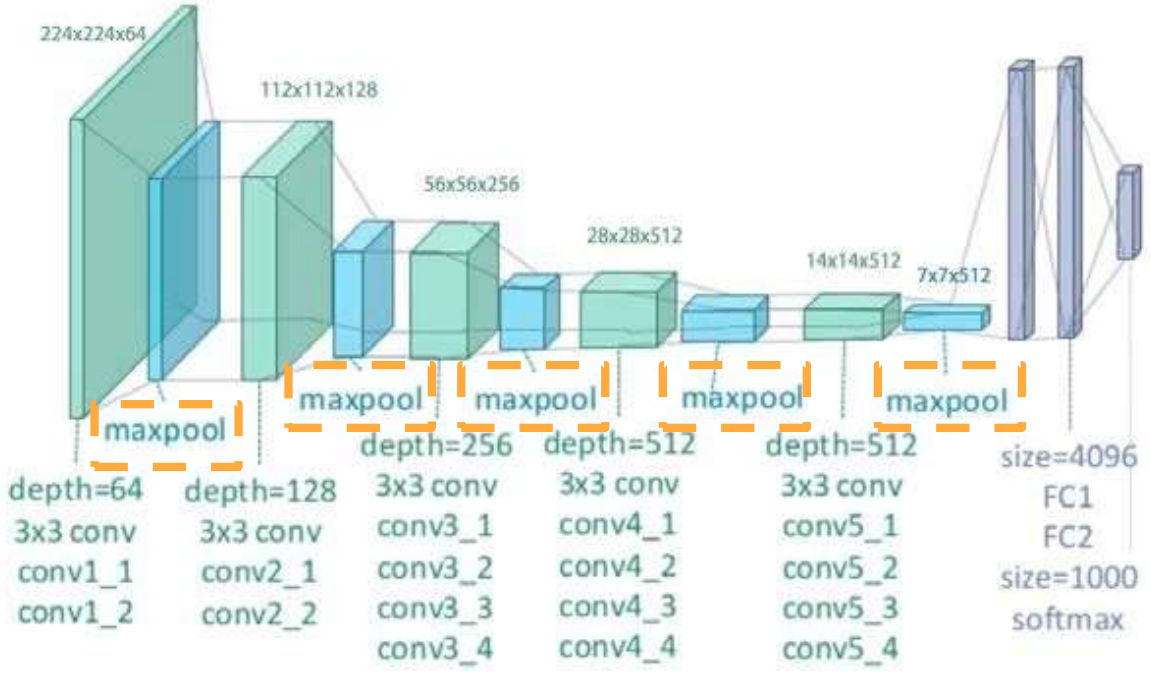
Consideremos que el discriminador es un VGG-19…

SRGAN - Función de Pérdida



Consideremos que el discriminador es un VGG-19… cuya tarea es clasificar 1000 cat.

SRGAN - Función de Pérdida



Consideremos cada maxpool como un feature map y será llamada ϕ

SRGAN - Función de Pérdida

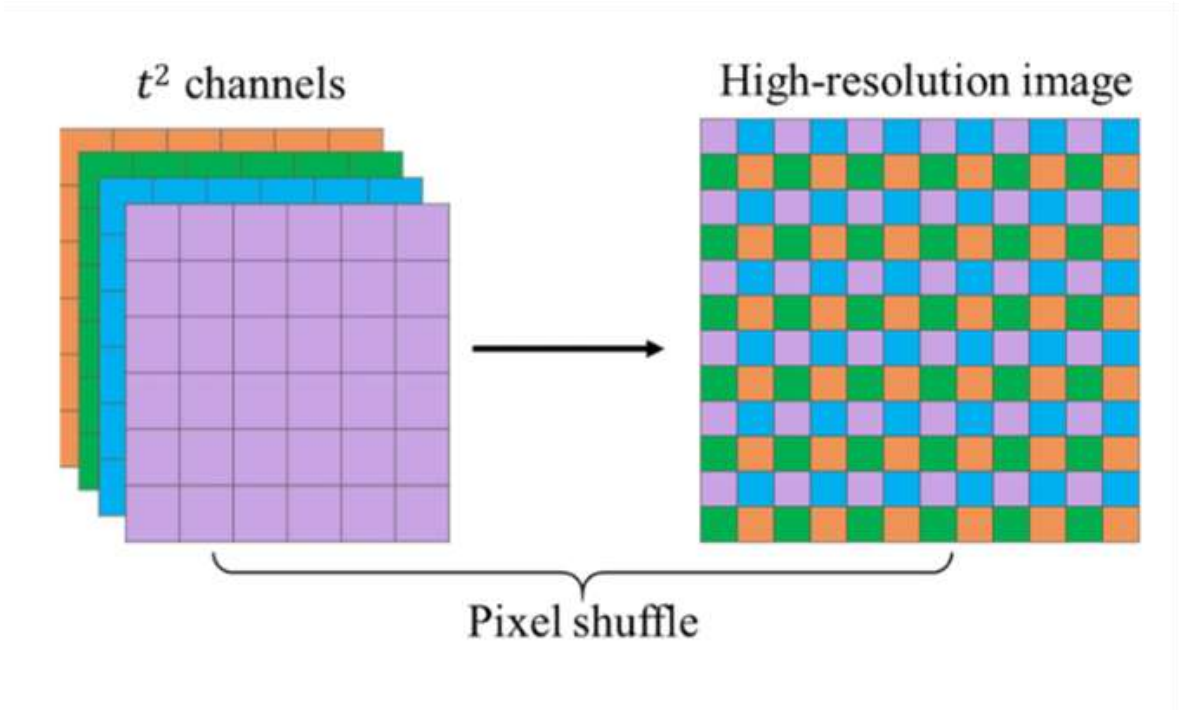
$$l^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

De manera que el error es la diferencia del mapa de activación de la imagen en alta resolución vs. el mapa de activación de la imagen obtenida por el Generador con la imagen de Baja Resolución

SRGAN - Función de Pérdida del Discriminador

$$l_{Gen}^{SR} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

Pixel Shuffle



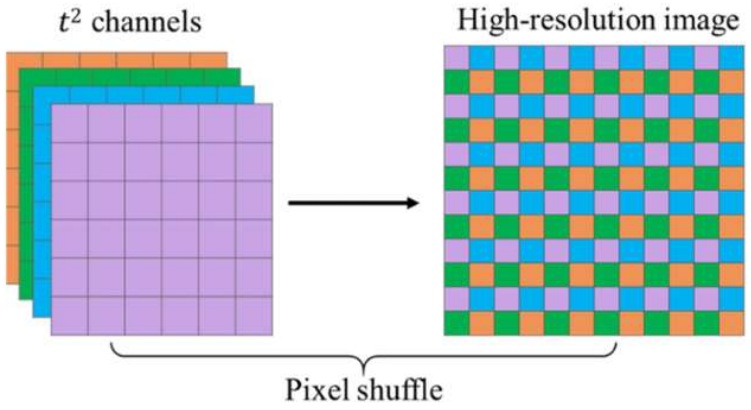
Más info:
https://paperswithcode.com/method/pixelshuffle

Pixel Shuffle

PixelShuffle is an operation used in super-resolution models to implement efficient sub-pixel convolutions with a stride of $1/r$. Specifically it rearranges elements in a tensor of shape $(*, C \times r^2, H, W)$ to a tensor of shape $(*, C, H \times r, W \times r)$.

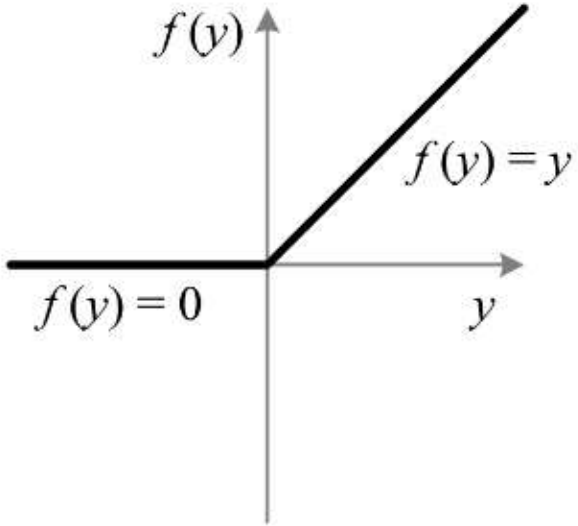
Image Source: Remote Sensing Single-Image Resolution Improvement Using A Deep Gradient-Aware Network with Image-Specific Enhancement

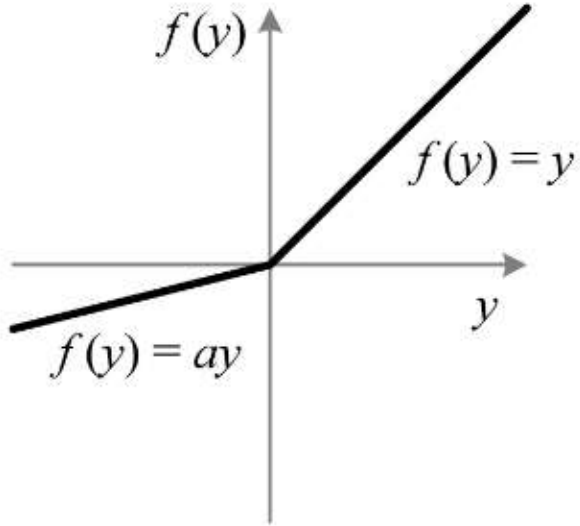
Source: Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network



Más info:
https://paperswithcode.com/method/pixelshuffle

Parametrized RELu





Más info:
https://paperswithcode.com/method/prelu

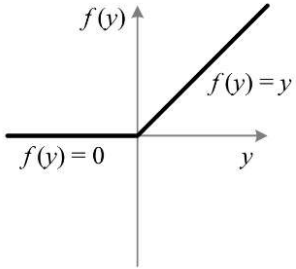
Parametrized RELu

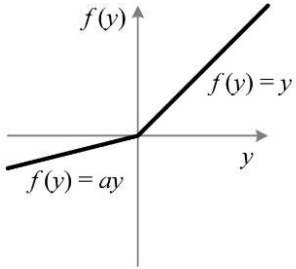
A **Parametric Rectified Linear Unit**, or **PReLU**, is an activation function that generalizes the traditional rectified unit with a slope for negative values. Formally:

$$f(y_i) = y_i \text{ if } y_i \geq 0$$

$$f(y_i) = a_i y_i \text{ if } y_i \leq 0$$

The intuition is that different layers may require different types of nonlinearity. Indeed the authors find in experiments with convolutional neural networks that PReLus for the initial layer have more positive slopes, i.e. closer to linear. Since the filters of the first layers are Gabor-like filters such as edge or texture detectors, this shows a circumstance where positive and negative responses of filters are respected. In contrast the authors find deeper layers have smaller coefficients, suggesting the model becomes more discriminative at later layers (while it wants to retain more information at earlier layers).





Más info:
https://paperswithcode.com/method/prelu

Fuentes

https://arxiv.org/abs/1609.04802v5