Input image (real)

$(3 \times 1024 \times 1024)$

Conv2d(kernel_size = (4x4), stride = 2, padding = 1)

$(8 \times 512 \times 512)$

LeakyReLU(alpha = 0.2)

Conv2d(kernel_size = (4x4), stride = 2, padding = 1)

$(16 \times 256 \times 256)$

BatchNorm2d

LeakyReLU(alpha = 0.2)

$(16 \times 256 \times 256)$

In case of an image size of **(3x512x512)**:

Conv2d(kernel_size = (4x4), stride = 2, padding = 1)

LeakyReLU(alpha = 0.2)

In case of an image size of **(3x256x256)**:

Conv2d(kernel_size = (3x3), stride = 1, padding = 1)

LeakyReLU(alpha = 0.2)

In both cases, the dimensions are after the first block:
$(16 \times 256 \times 256)$

main

Conv2d(kernel_size = (4x4), stride = 2, padding = 1)

$(32 \times 128 \times 128)$

BatchNorm2d

LeakyReLU(alpha = 0.2)

$(16 \times 128 \times 128)$

Conv2d(kernel_size = (3x3), stride = 1, padding = 1)

$(32 \times 128 \times 128)$

BatchNorm2d

LeakyReLU(alpha = 0.2)

$(32 \times 128 \times 128)$

direct

AvgPooling2d(kernel_size = (2,2), stride = 2, padding = 0)

Conv2d(kernel_size = (1x1), stride = 1, padding = 0)

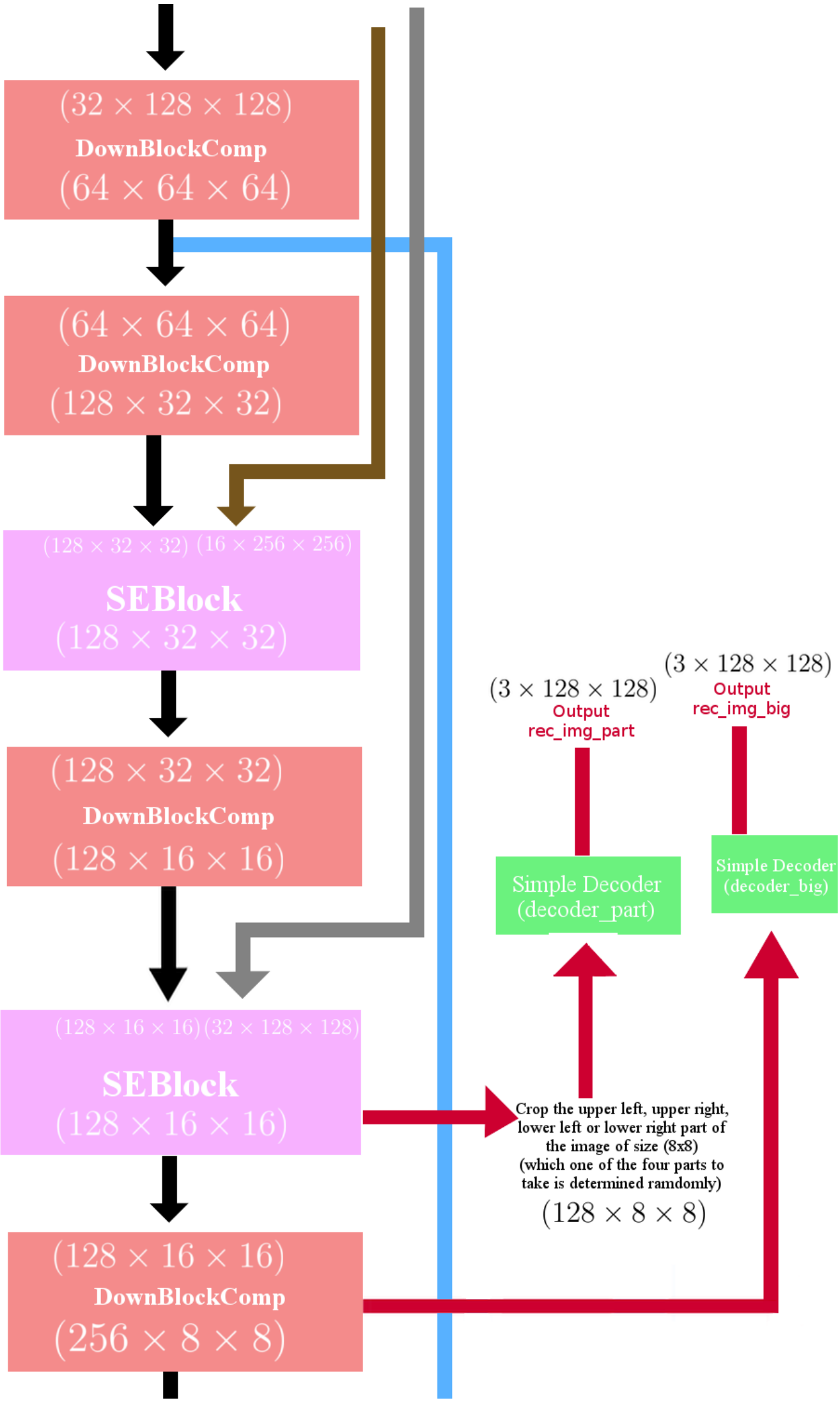
BatchNorm2d

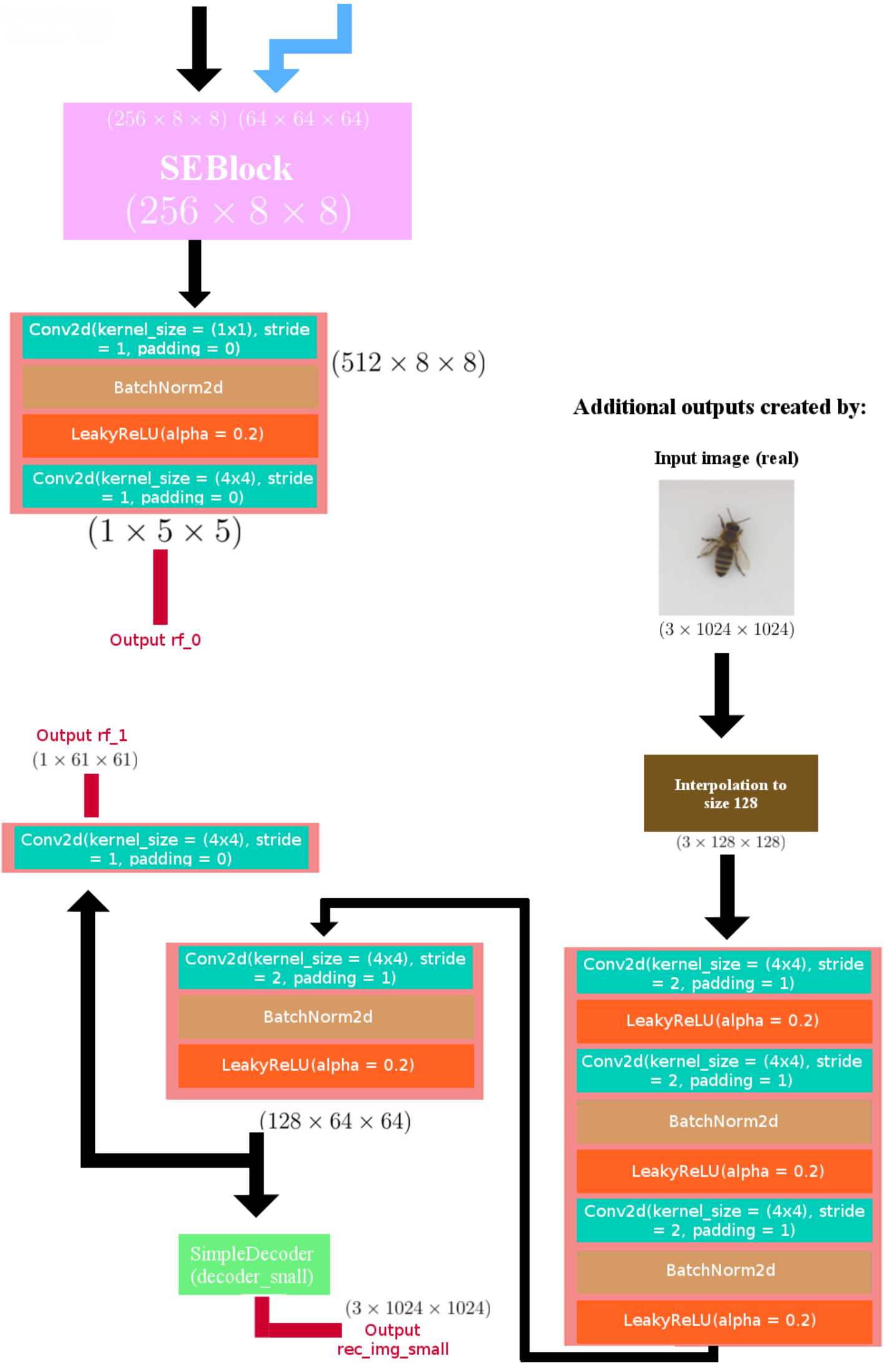
LeakyReLU(alpha = 0.2)

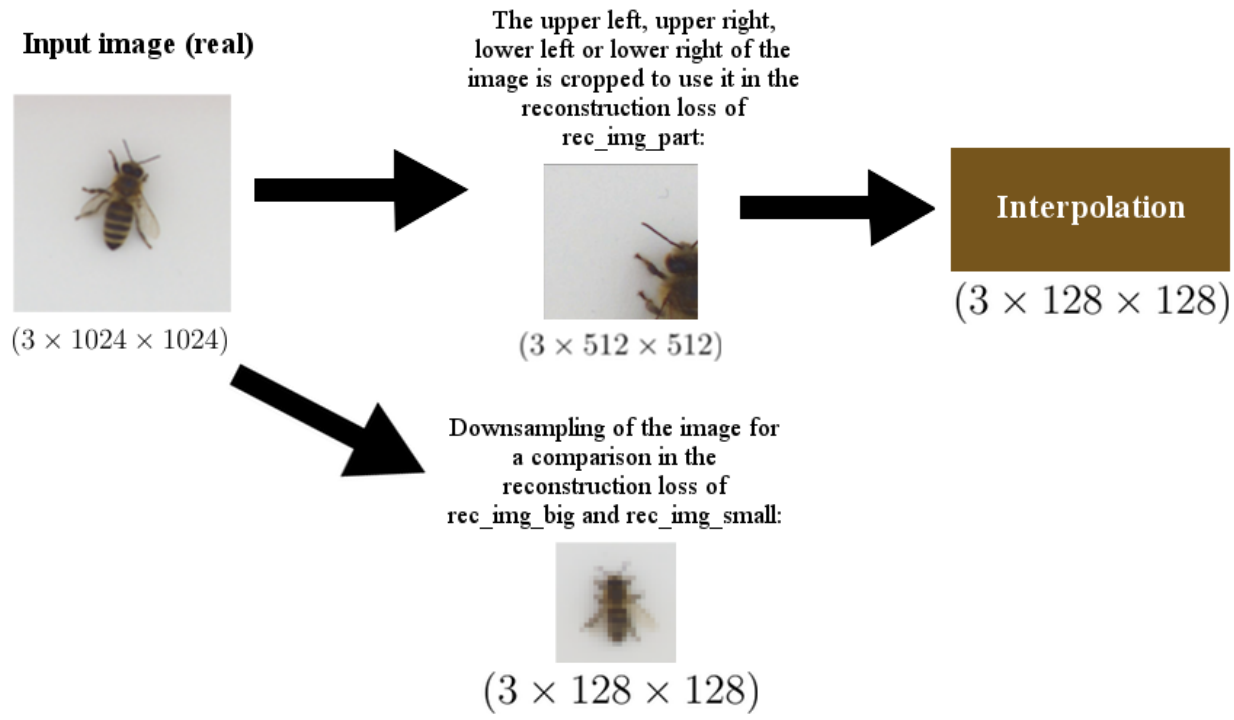
$(32 \times 128 \times 128)$

$\bar{x}$

DownBlockComp


(32 × 128 × 128)
DownBlockComp
(64 × 64 × 64)
(64 × 64 × 64)
DownBlockComp
(128 × 32 × 32)
(128 × 32 × 32) (16 × 256 × 256)
SEBlock
(128 × 32 × 32)
(128 × 32 × 32)
DownBlockComp
(128 × 16 × 16)
(128 × 16 × 16) (32 × 128 × 128)
SEBlock
(128 × 16 × 16)
(128 × 16 × 16)
DownBlockComp
(256 × 8 × 8)
(3 × 128 × 128)
Output
rec_img_part
(3 × 128 × 128)
Output
rec_img_big
Simple Decoder
(decoder_part)
Simple Decoder
(decoder_big)
Crop the upper left, upper right,
lower left or lower right part of
the image of size (8x8)
(which one of the four parts to
take is determined ramdomly)
(128 × 8 × 8)


$(256 \times 8 \times 8)$ $(64 \times 64 \times 64)$
SEBlock
$(256 \times 8 \times 8)$
Conv2d(kernel_size = (1x1), stride = 1, padding = 0)
BatchNorm2d
LeakyReLU(alpha = 0.2)
Conv2d(kernel_size = (4x4), stride = 1, padding = 0)
$(512 \times 8 \times 8)$
$(1 \times 5 \times 5)$
Output rf_0
Additional outputs created by:
Input image (real)
$(3 \times 1024 \times 1024)$
Interpolation to size 128
$(3 \times 128 \times 128)$
Conv2d(kernel_size = (4x4), stride = 2, padding = 1)
LeakyReLU(alpha = 0.2)
Conv2d(kernel_size = (4x4), stride = 2, padding = 1)
BatchNorm2d
LeakyReLU(alpha = 0.2)
Conv2d(kernel_size = (4x4), stride = 2, padding = 1)
BatchNorm2d
LeakyReLU(alpha = 0.2)
Conv2d(kernel_size = (4x4), stride = 2, padding = 1)
BatchNorm2d
LeakyReLU(alpha = 0.2)
$(128 \times 64 \times 64)$
Conv2d(kernel_size = (4x4), stride = 1, padding = 0)
Output rf_1
$(1 \times 61 \times 61)$
SimpleDecoder (decoder_small)
$(3 \times 1024 \times 1024)$
Output rec_img_small



Simple Decoder is given by: