Deep Convolutional Generative Adversarial Networks (DCGAN)

ECE57000: Artificial Intelligence, Fall 2019

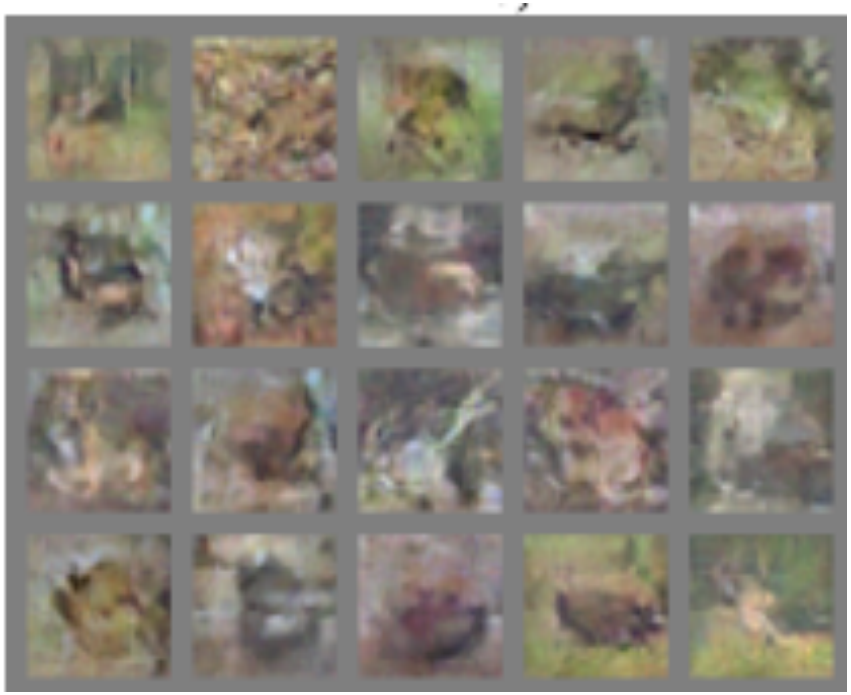
David I. Inouye

Announcements

- Quiz on Wednesday

DCGAN: Randomly generated bedrooms show slightly odd but almost realistic bedrooms

Original GAN (CIFAR10)



DCGAN (bedrooms)

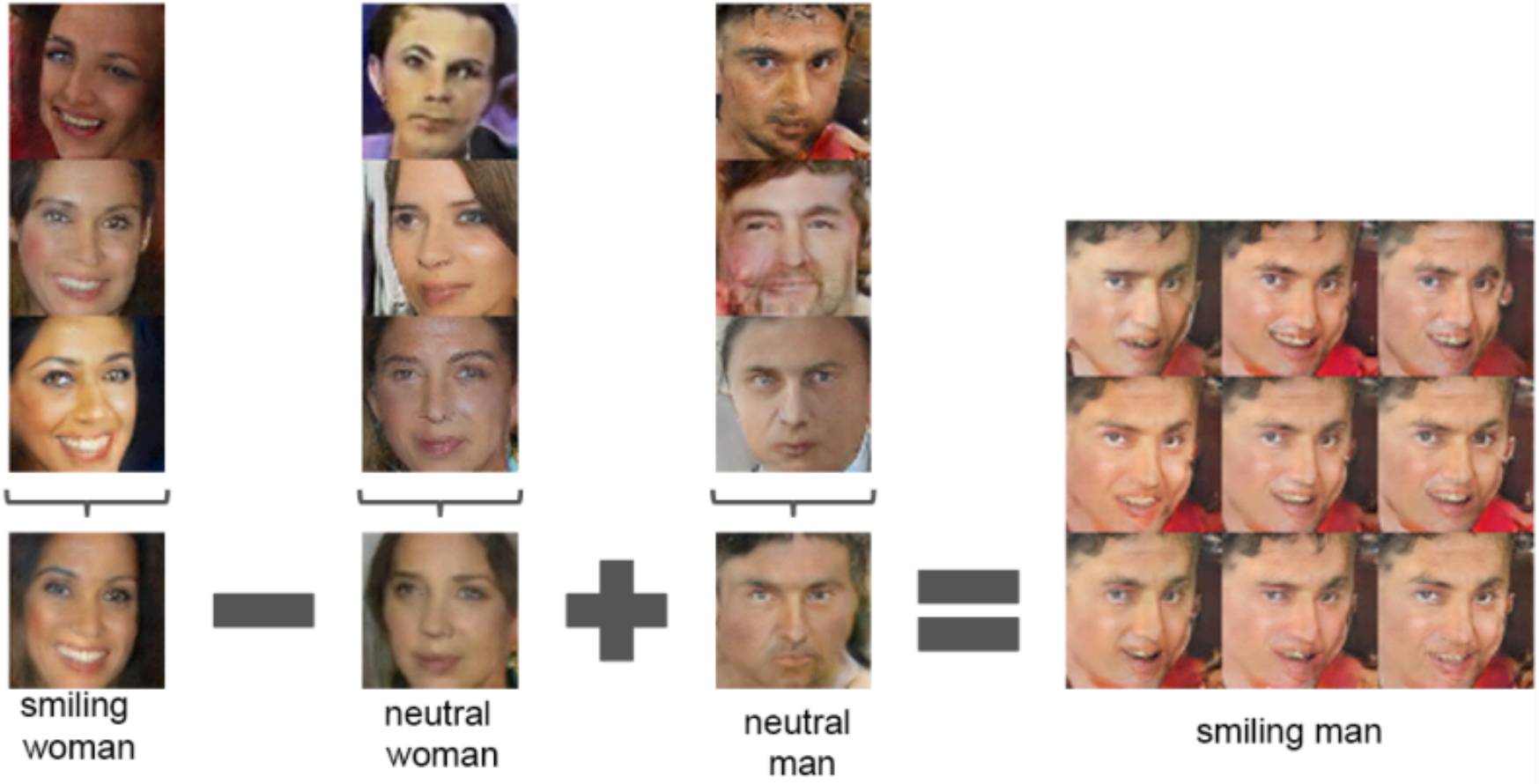
DCGAN can show interpolation between imaginary hotel rooms

https://arxiv.org/pdf/1511.06434.pdf

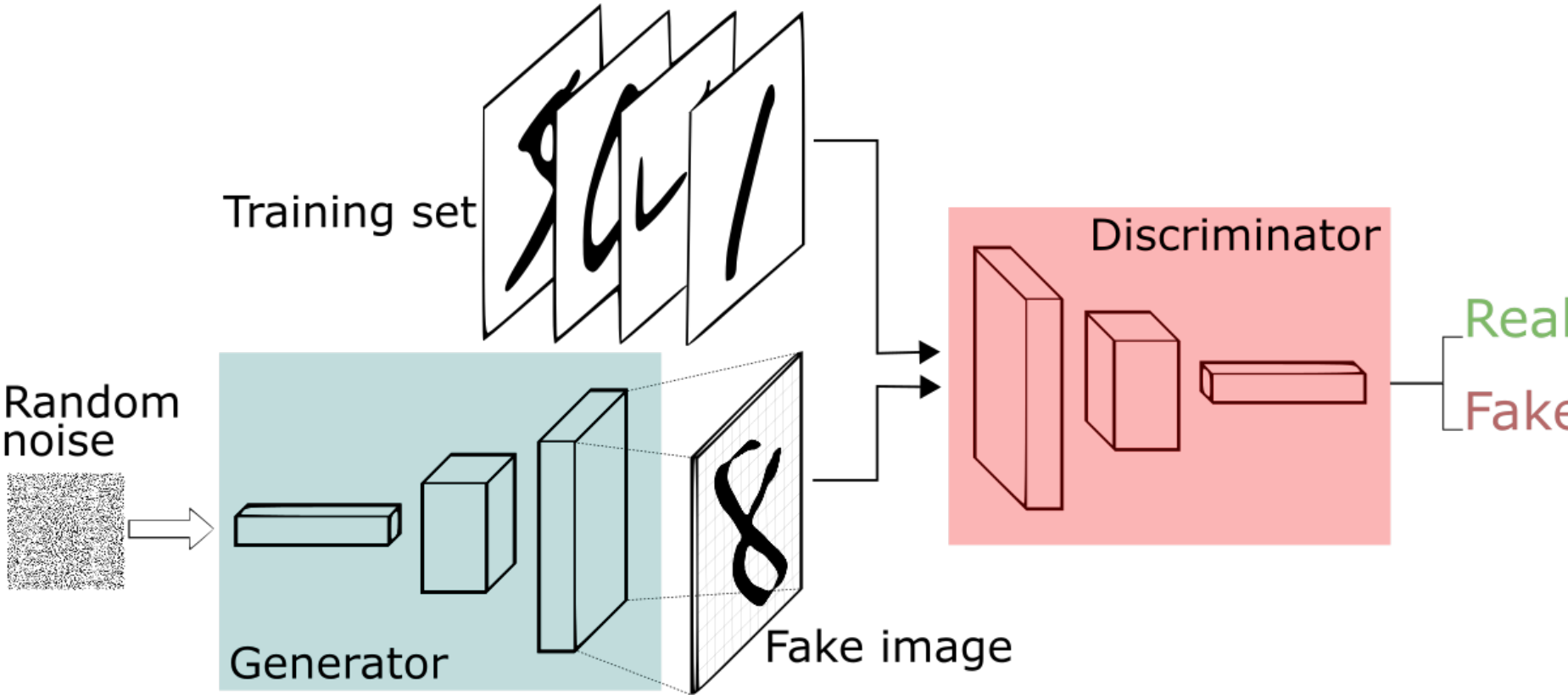
Removing certain filters can modify the generated images (in this case, a “window” filter)

Figure 6: Top row: un-modified samples from model. Bottom row: the same samples generated with dropping out ”window” filters. Some windows are removed, others are transformed into objects with similar visual appearance such as doors and mirrors. Although visual quality decreased, overall scene composition stayed similar, suggesting the generator has done a good job disentangling scene representation from object representation. Extended experiments could be done to remove other objects from the image and modify the objects the generator draws.

Simple vector arithmetic in **latent space** of DCGAN can generate new faces

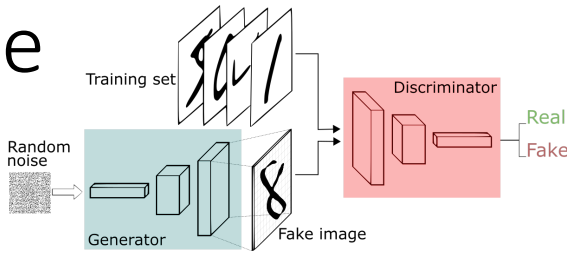


How do we learn this implicit generative model? Train two deep networks simultaneously



https://www.freecodecamp.org/news/an-intuitive-introduction-to-generative-adversarial-networks-gans-7a2264a81394/

DCGAN generator upsamples the size of the image in multiple stages



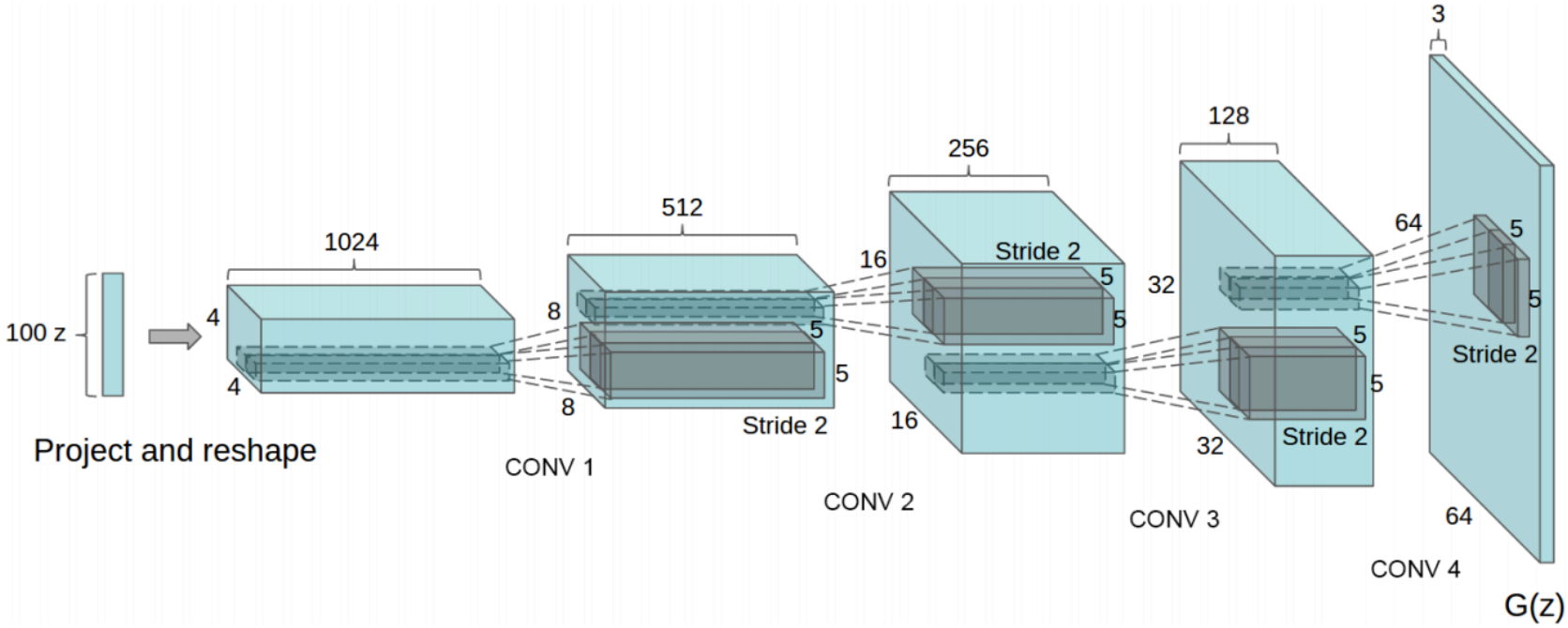


Figure 1: DCGAN generator used for LSUN scene modeling. A 100 dimensional uniform distribution Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64×64 pixel image. Notably, no fully connected or pooling layers are used.

Transposed convolution can be used to **upsample** an tensor/image to have higher dimensions

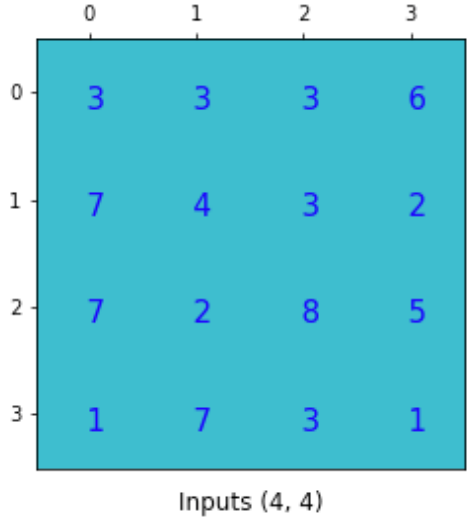
- Also known as:
  - Fractionally-strided convolution
  - Improperly, deconvolution
- Remember: Convolution can be seen as a matrix multiplication

$$y = x\ (*)\ f \Leftrightarrow \mathrm{vec}(y) = A_f \mathrm{vec}(x)$$

- Transpose convolution is the transpose of A_f:

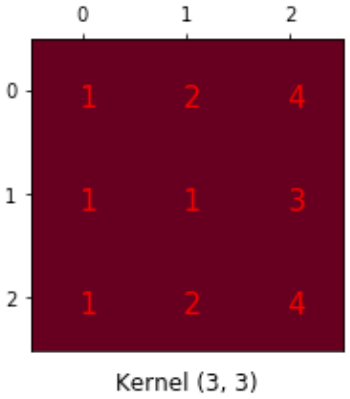
$$\mathrm{vec}(y) = A_f^T \mathrm{vec}(x)$$

Convolution operator with corresponding matrix



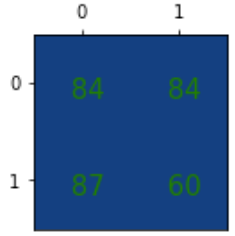
Inputs (4, 4)

(*)

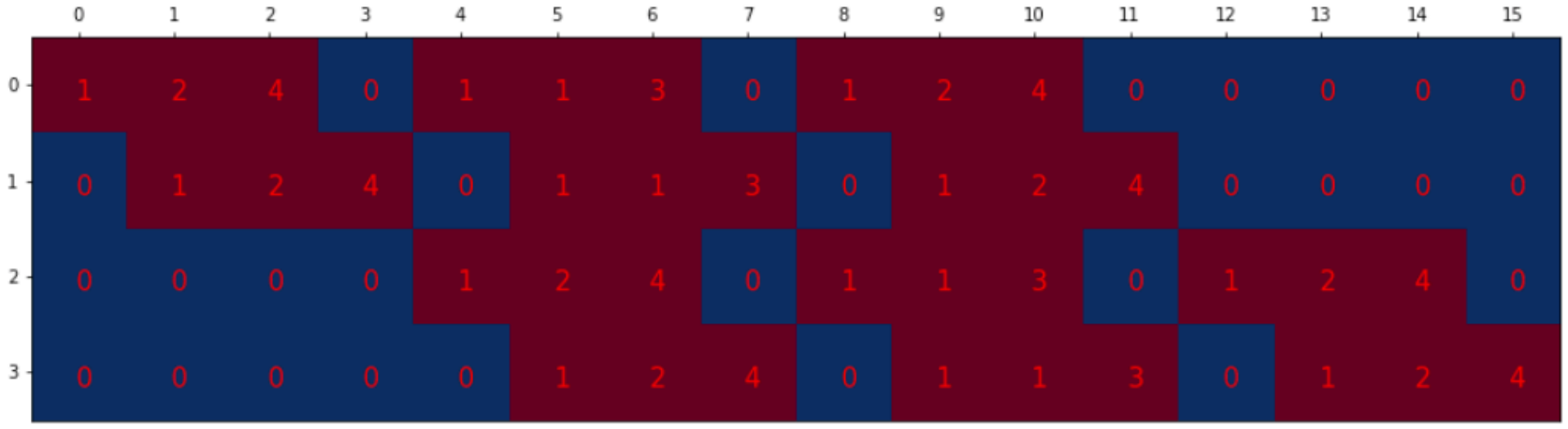


Kernel (3, 3)

=



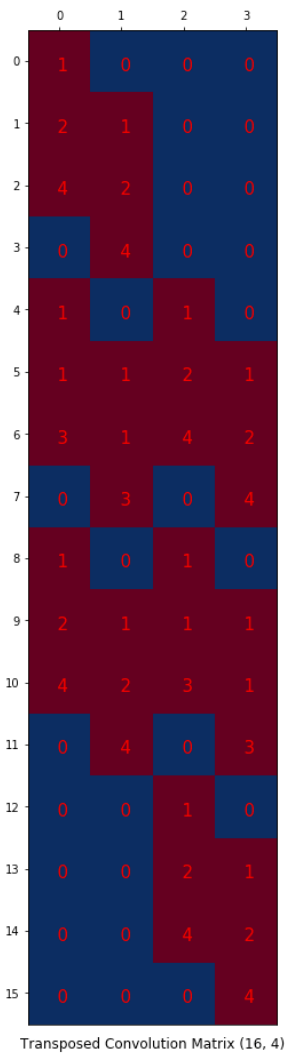
Output (2, 2)



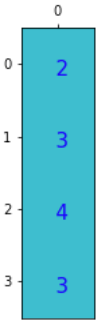
Convolution Matrix (4, 16)

https://github.com/naokishibuya/deep-learning/blob/master/python/transposed_convolution.ipynb

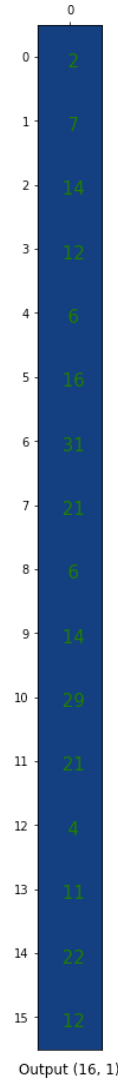
Transposed convolution operator with corresponding matrix



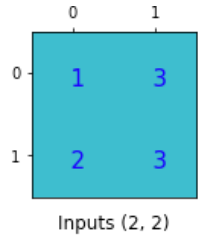
*



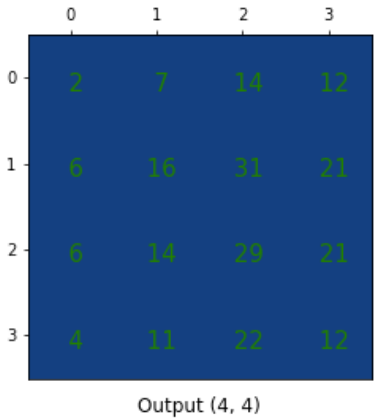
=



Reshaped input

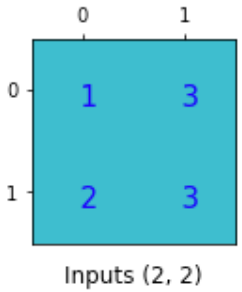


Reshaped output



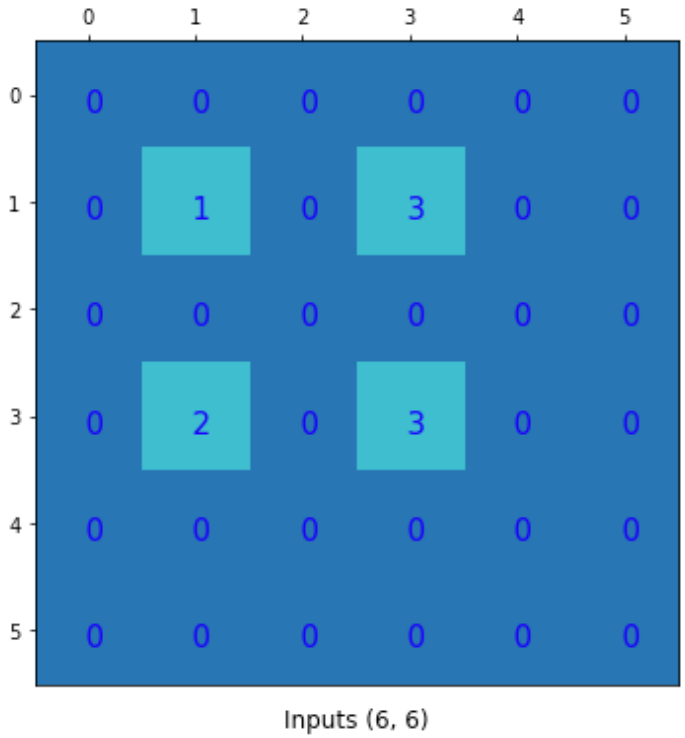
https://github.com/naokishibuya/deep-learning/blob/master/python/transposed_convolution.ipynb

Transposed convolution can be **equivalent** to a simple convolution with zero rows/columns added (added zeros simulate fractional strides)



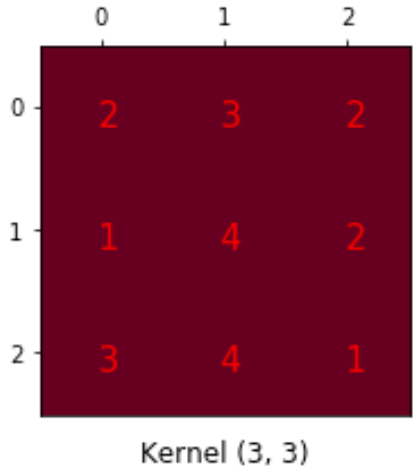
Original input

Zero-padded input



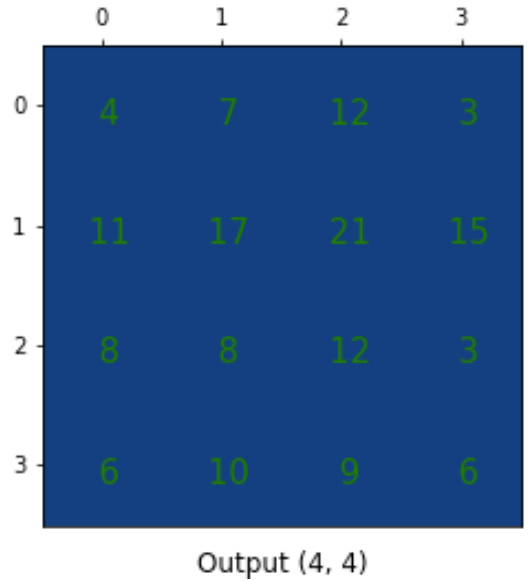
(*)

Kernel



=

Output



Implementation in PyTorch: DCGAN Tutorial

- DCGAN Tutorial
  https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html

- GAN training tips/hacks
  - https://github.com/soumith/ganhacks