Intro to Neural Nets

Week 5: Convolutional NNs (ConvNets)

Today’s Agenda

Modern Image-Network Architectures

- Mitigating vanishing gradients in deep networks.

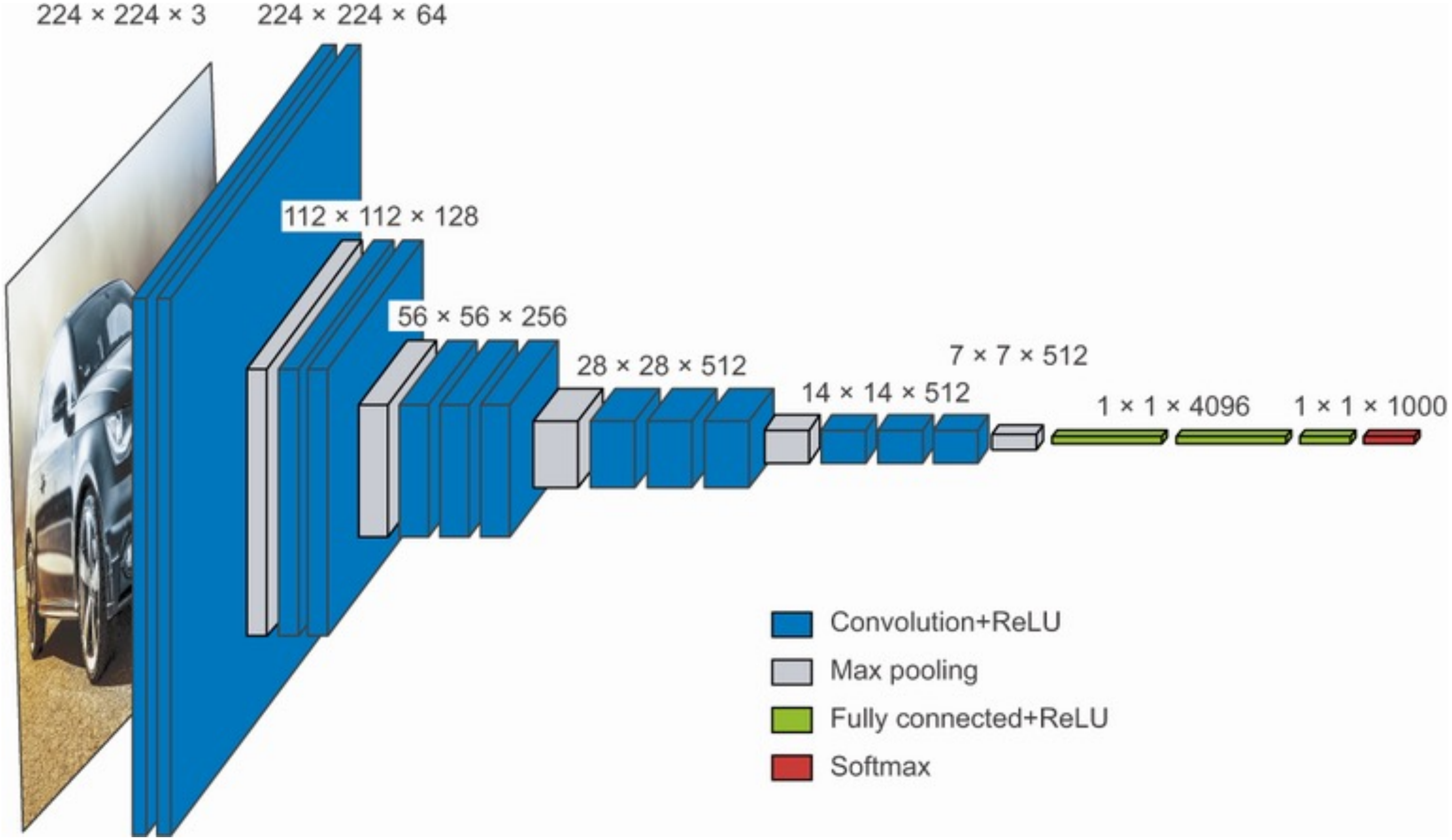
Image-to-Image Prediction Tasks

- Image segmentation, Image super-resolution.
- Transpose convolution operations for up-sampling.

Visualizing What a CNN is Learning

- Try to visualize what components of an image your model’s feature extractions are detecting.

Image-Classification Architectures



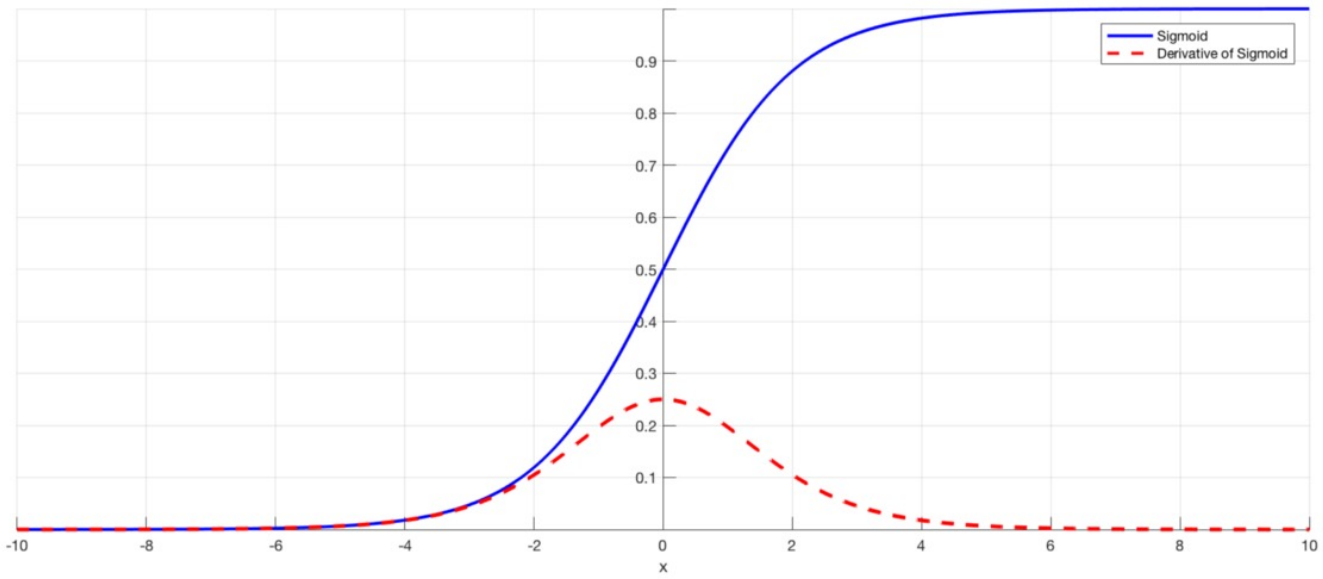
Vanishing Gradients

Recall That Certain Activations Make Vanishing Gradients More Likely

- As we approach 0 or 1 at a node's output, changes to input parameters have little impact...

However, Regardless of Activation This Can Be an Issue

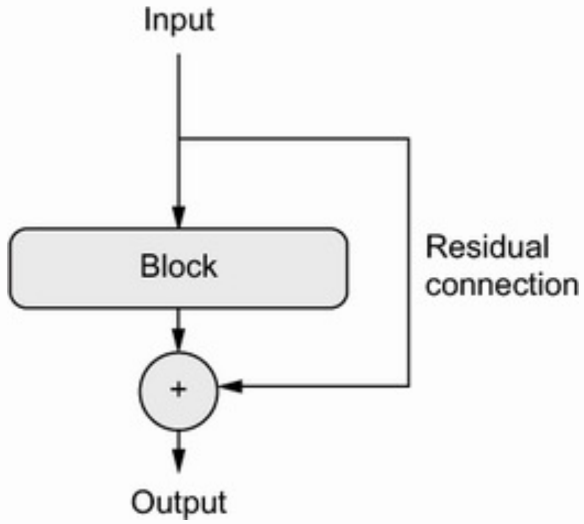
- For deep networks, parameters at front of network tend to have a much smaller influence on ultimate loss function. Accordingly, gradients for those early parameters can be very small.

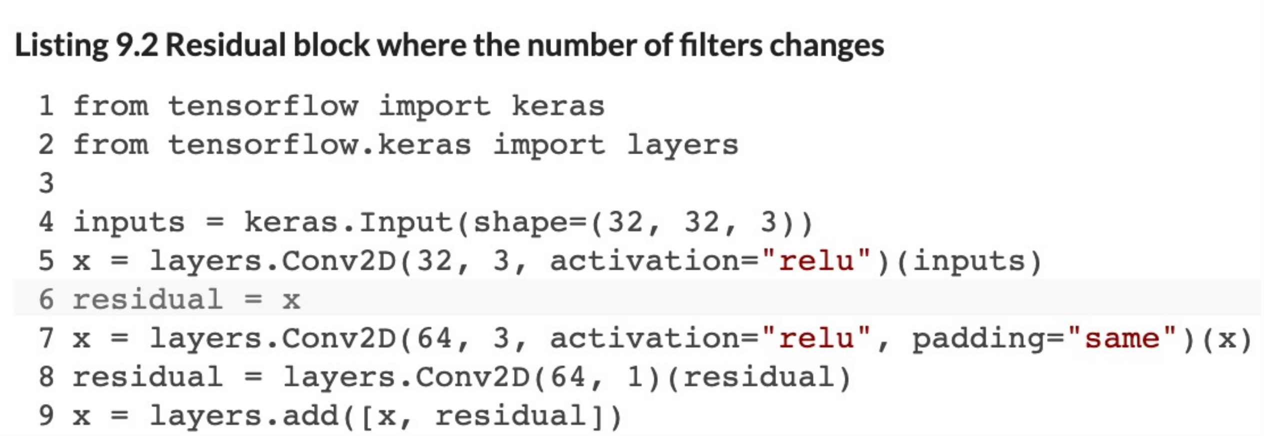


Solution: Residual Connections

Provide a ’Short-cut’ From Loss Function to Front-end Weights

- We include feed-forward layers as usual, but we also add short-cut connections *around* the layers.
- We typically incorporate these residual connections either via an ‘Add’ layer or a ‘Concatenate’ layer.
- Note that conformity of the tensor shapes will be very important here – you’ll start encountering shape conformity errors if you are not careful!





Listing 9.2 Residual block where the number of filters changes

```
from tensorflow import keras
from tensorflow.keras import layers

inputs = keras.Input(shape=(32, 32, 3))
x = layers.Conv2D(32, 3, activation="relu")(inputs)
residual = x
x = layers.Conv2D(64, 3, activation="relu", padding="same")(x)
residual = layers.Conv2D(64, 1)(residual)
x = layers.add([x, residual])
```

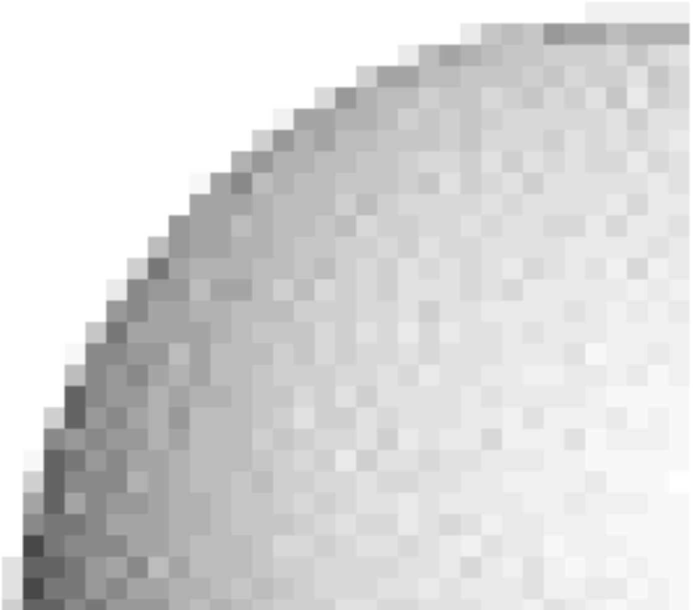
Image-to-Image Prediction

Super-Resolution

- Take a high-resolution image, pixelate it, then try to predict the high resolution from the pixelated version.
- *Q: What kind of activation and loss function will make sense in this task?*

Activation function: non-negative

Loss function: identifying worst case errors (customized)



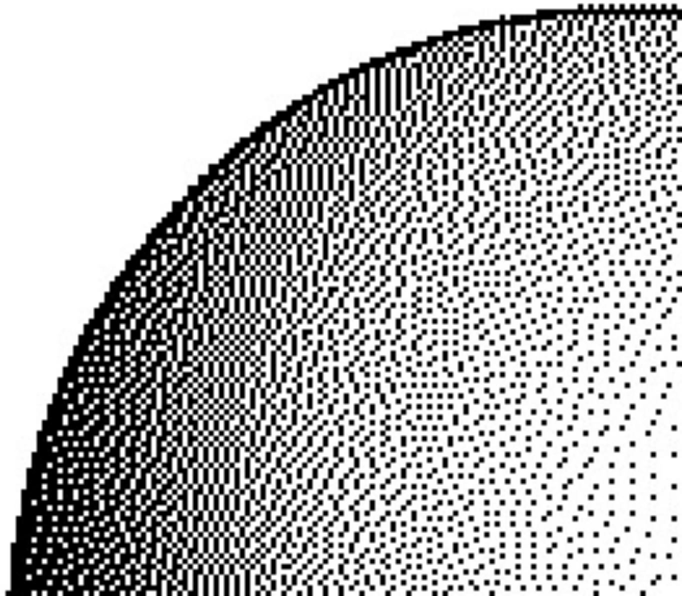
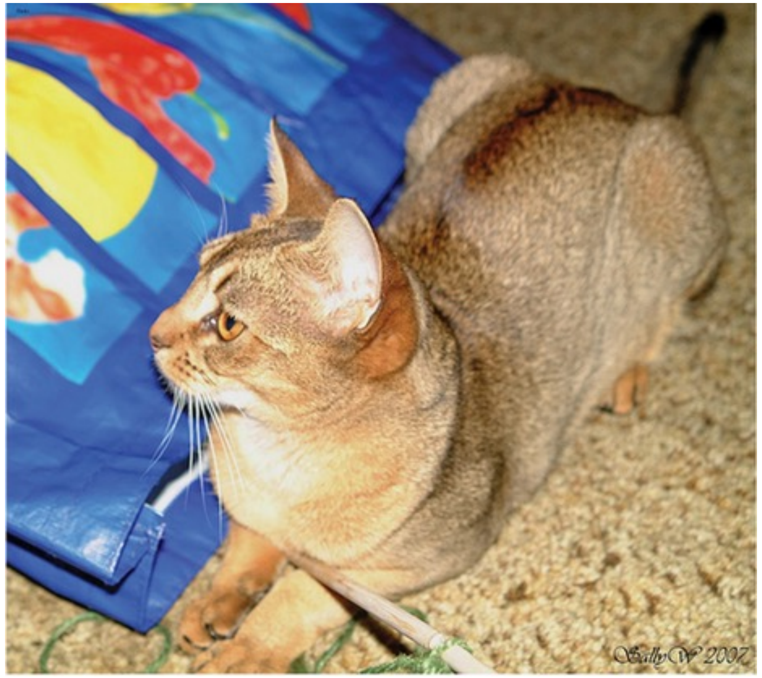


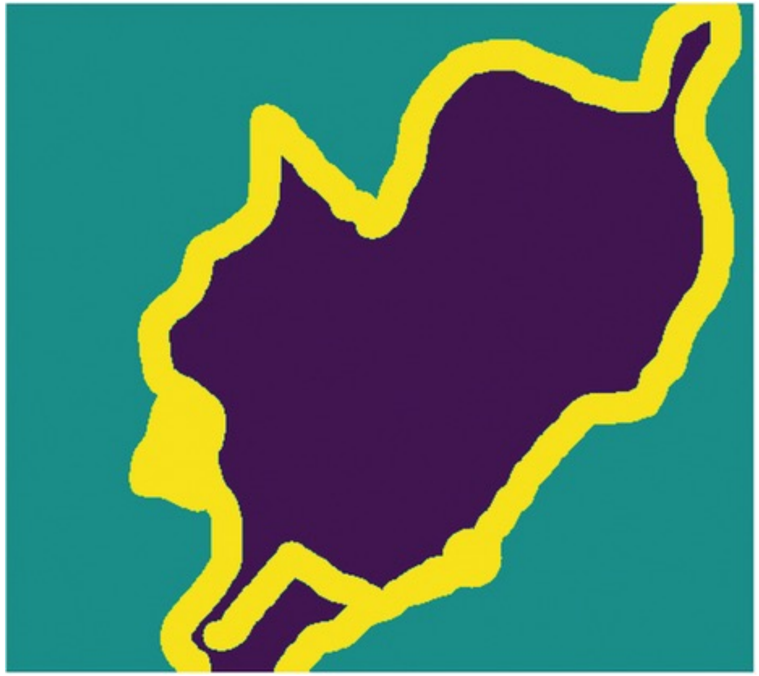
Image-to-Image Prediction

Activation: Softmax
Loss: Cross entropy

Image-Segmentation

- Take an image and its segment mask, then try to predict the segment associated with each pixel from the original picture.
- *Q: What kind of activation function and loss function will make sense in this task?*

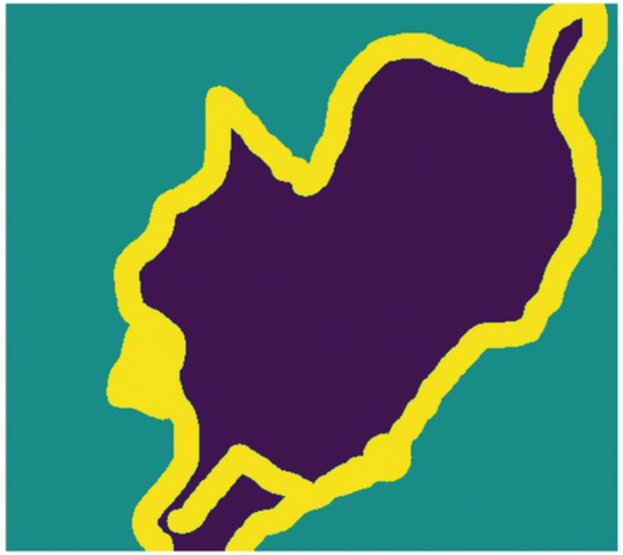




Topology for Image-to-Image

Auto-Encoder Architecture

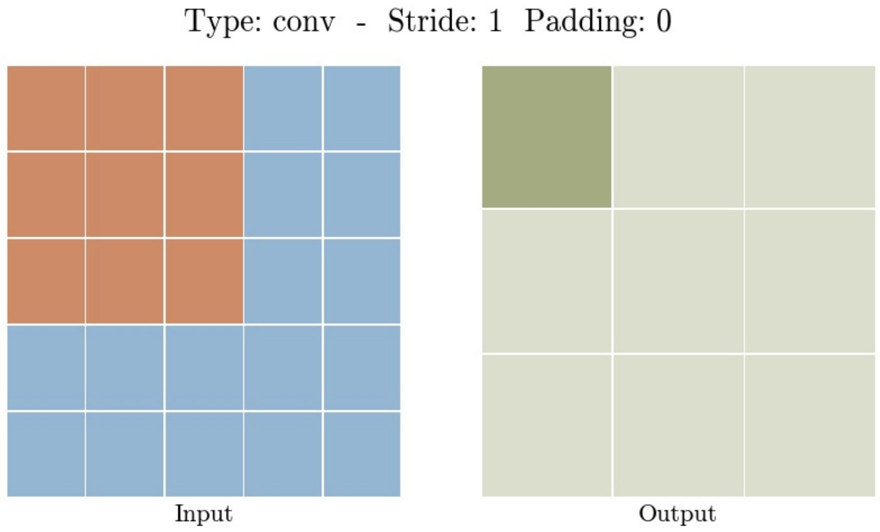
- Down-sample and then Up-sample back to same dimensionality
- We do not use down-sample using pooling (because they force attention to the whole image as we learn higher level features). Instead, we use larger strides. This enables ‘dimensionality’ reduction while maintaining a focus on local portions of the image.
- We then ‘up-sample’ back to the original dimensionality using a transpose of the convolutional operation. This is a form of autoencoder architecture.

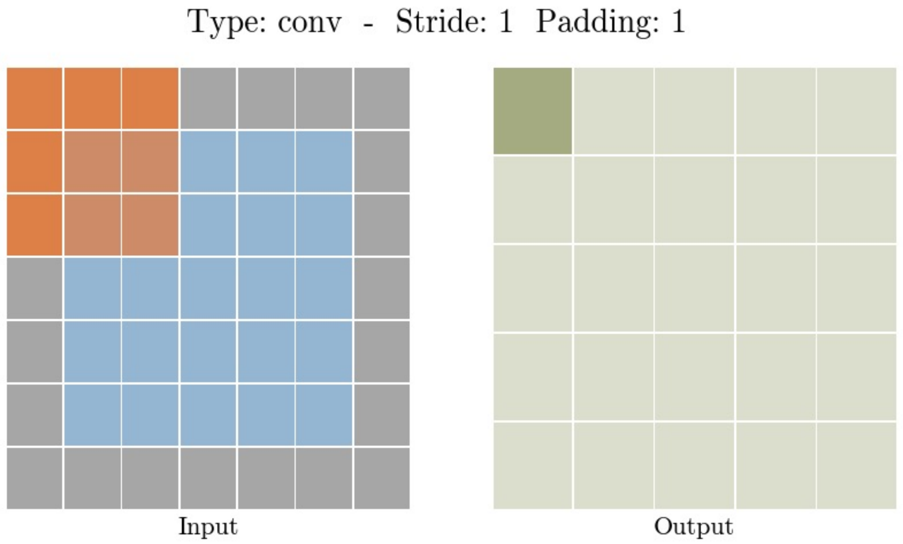


Convolution

Recall What A Convolution Is…

- It’s a down-sampling approach (it compresses information)

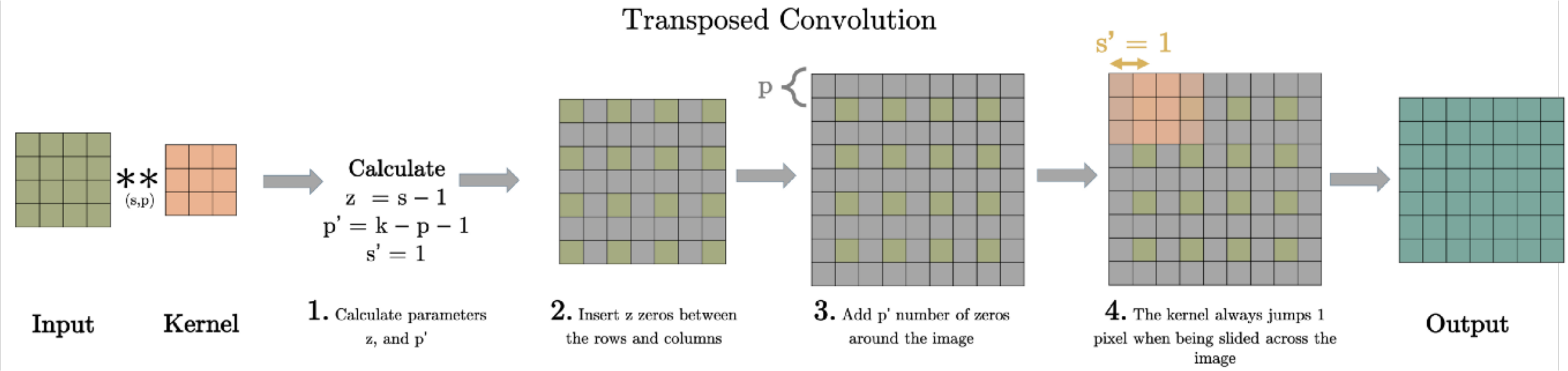




Transpose Convolution

Transpose of a Convolution

- Instead of Input (+padding) * filter (+stride) = output, this calculates the inverse operation, to up-sample.
- Here, s is the Convolution stride, p is the Convolution padding, k = is the Convolution kernel width/height; thus, s’ is the stride of the transpose convolution (always 1), z = padding around each pixel element, p’ is the final padding around the image for the transpose operation.

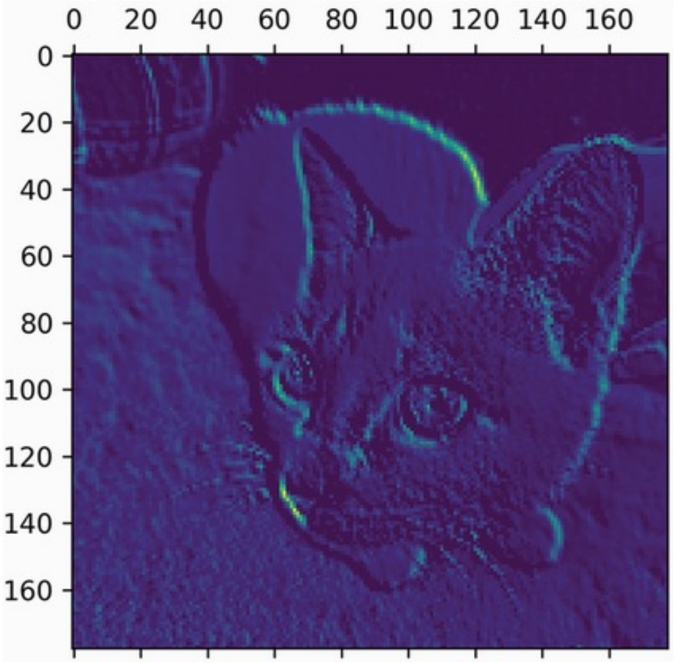


Transposed Convolution
s’ = 1
p
**
(s,p)
Calculate
z = s − 1
p’ = k − p − 1
s’ = 1
Input
Kernel
1. Calculate parameters z, and p'
2. Insert z zeros between the rows and columns
3. Add p' number of zeros around the image
4. The kernel always jumps 1 pixel when being slided across the image
Output

Visualizing What a CNN is Learning

Visualizing Layer Activations

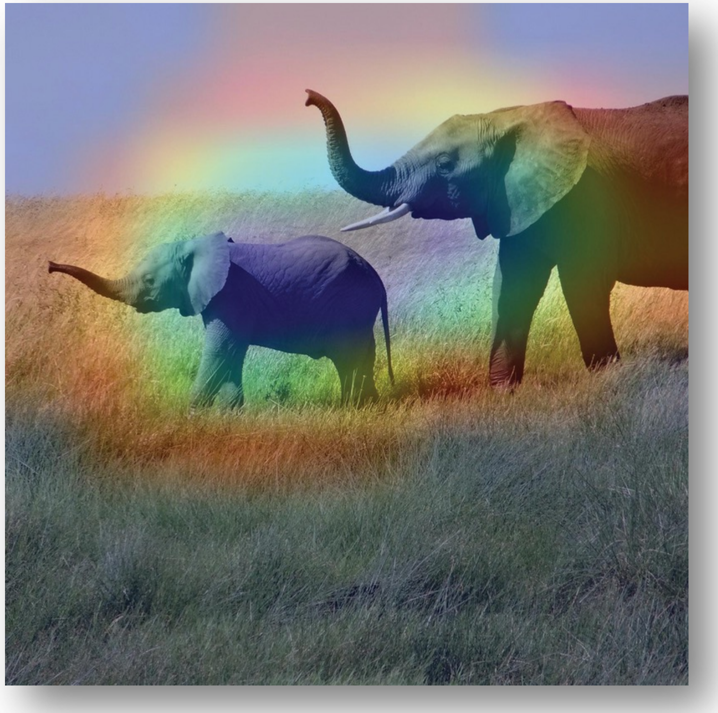
- For a given layer, we plot the 2D feature map for each channel (filter). Each one will capture independent features of the image – plotting each layer's activation (output) can show us what pixels are being identified in the pictures to produce a prediction.



Visualizing What a CNN is Learning

Class Activation Maps

- Generate a heatmap on the original input image that indicates what ‘components’ (feature maps) of the input image most drive the final classification assignment.
- The idea here is to calculate the gradient of the output class w.r.t. mapped features from the input image, and then highlight the pixel where the high-gradient features were most present.



Questions?