Intro to Neural Networks

BA865 – Mohannad Elhamod

BOSTON UNIVERSITY

Boston University Questrom School of Business

CNNs

Convolutional Networks

BOSTON UNIVERSITY

A Problem of Scalability

- How many parameters in this network?
- Do we really need to learn all these parameters?

3 x 224 x 224 = ~150k pixels

Figure courtesy of Robert Bond

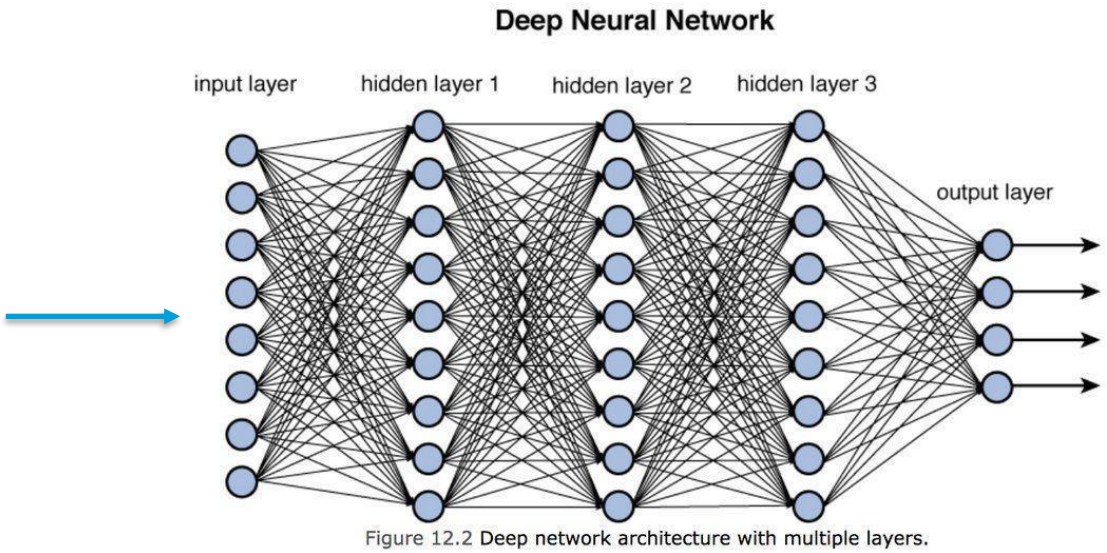


Figure 12.2 Deep network architecture with multiple layers.

Figure courtesy of Ravindra Parmar

Structure in Images

- Interesting images have:
  - Locality of information.
  - Spatial invariance.

Figure courtesy of Robert Bond

vs.

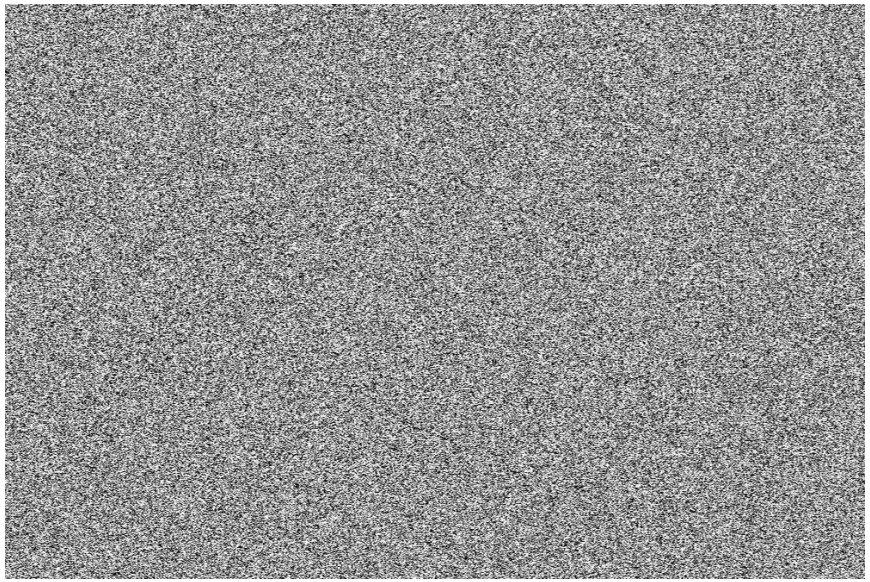


Figure courtesy of Jorge Stolfi

Convolutional Filters

- Instead of learning a mesh of all possible parameters, let's learn local descriptors *(kernels or filters)* that can be reused across the image!

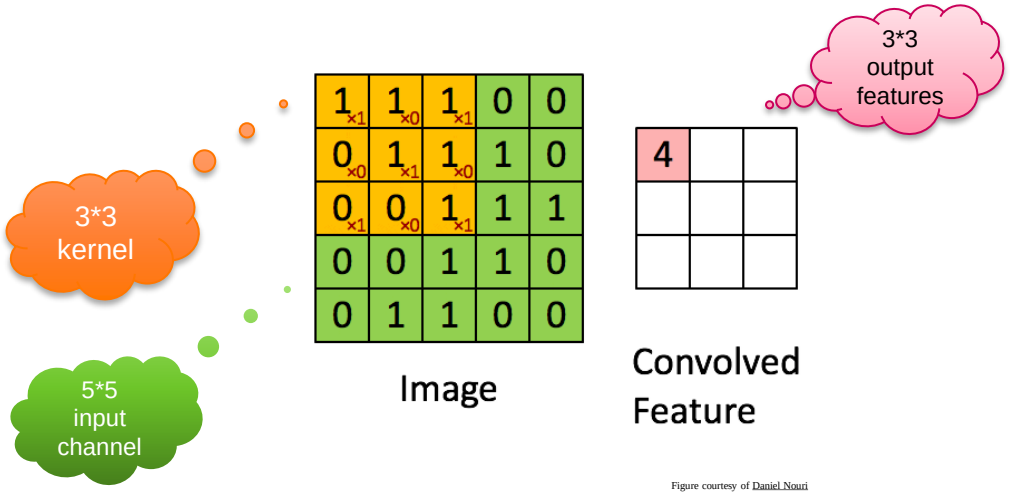


Figure courtesy of Daniel Nouri

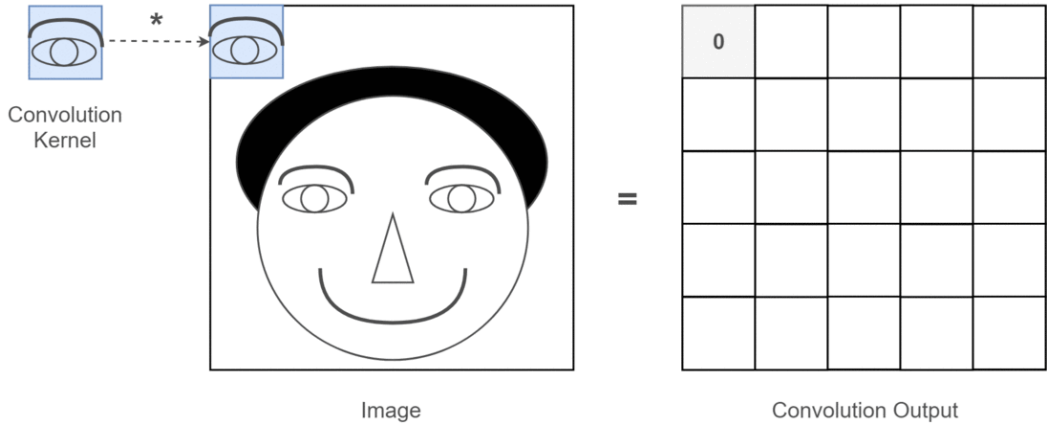
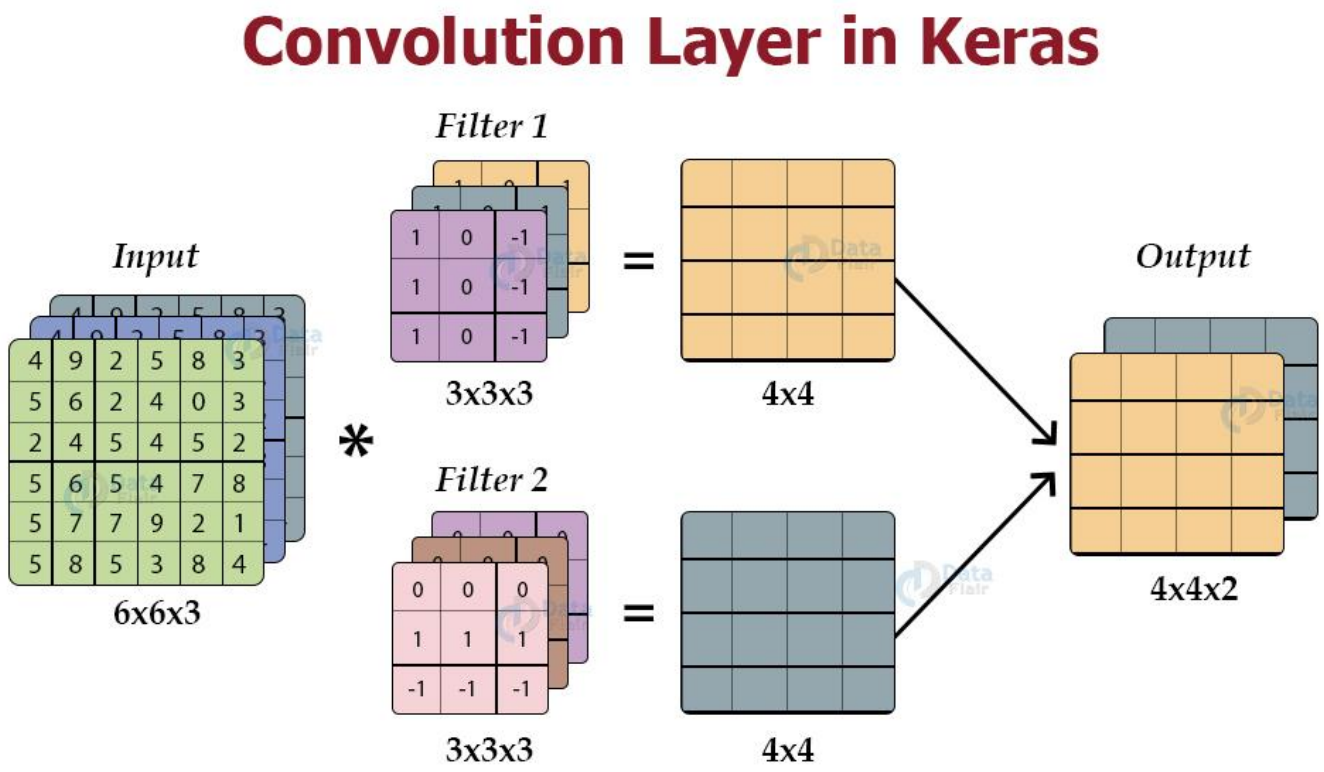


Figure courtesy of Thushan Ganegedara

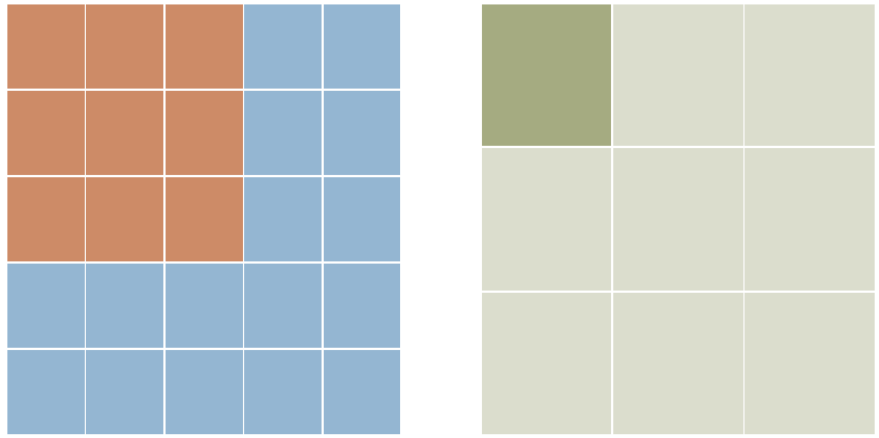
Mathematically Speaking...



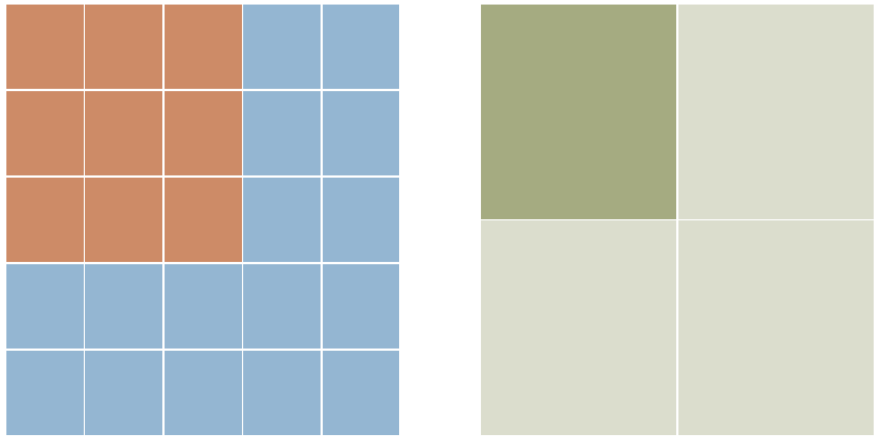
Stride and Padding

- Stride controls feature field overlap.
- Padding controls the down-sampling.

Type: conv - Stride: 1 Padding: 0



Type: conv - Stride: 2 Padding: 0

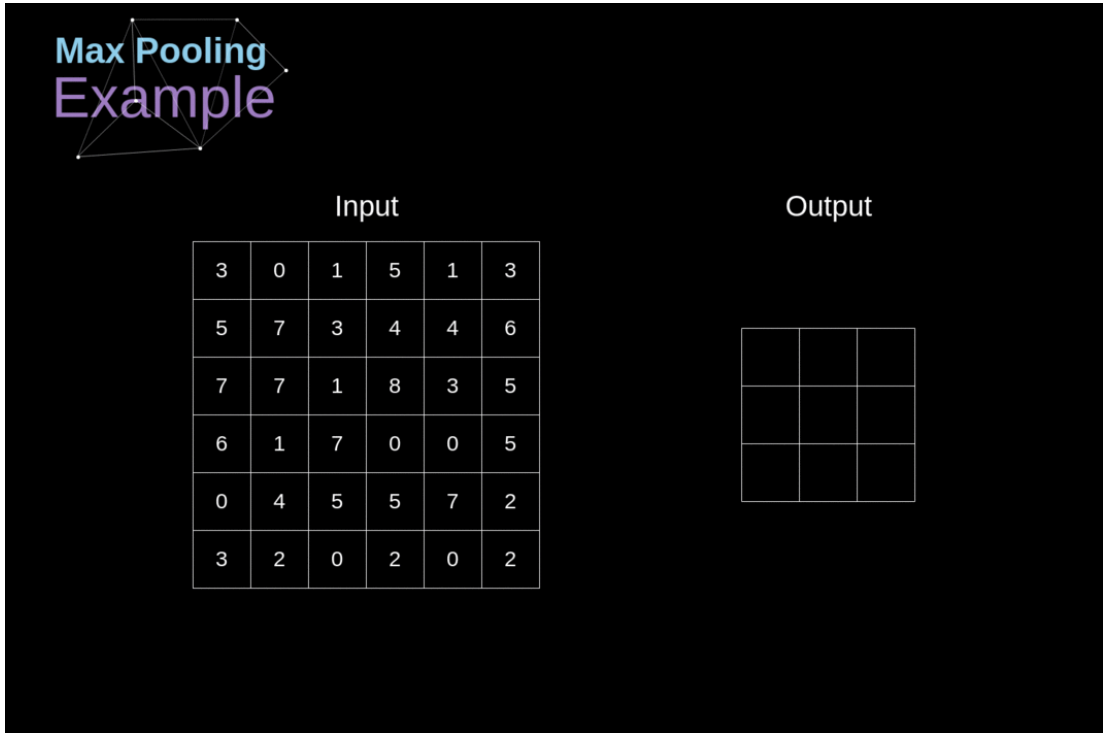


Type: conv - Stride: 1 Padding: 1

Aqeel Anwar

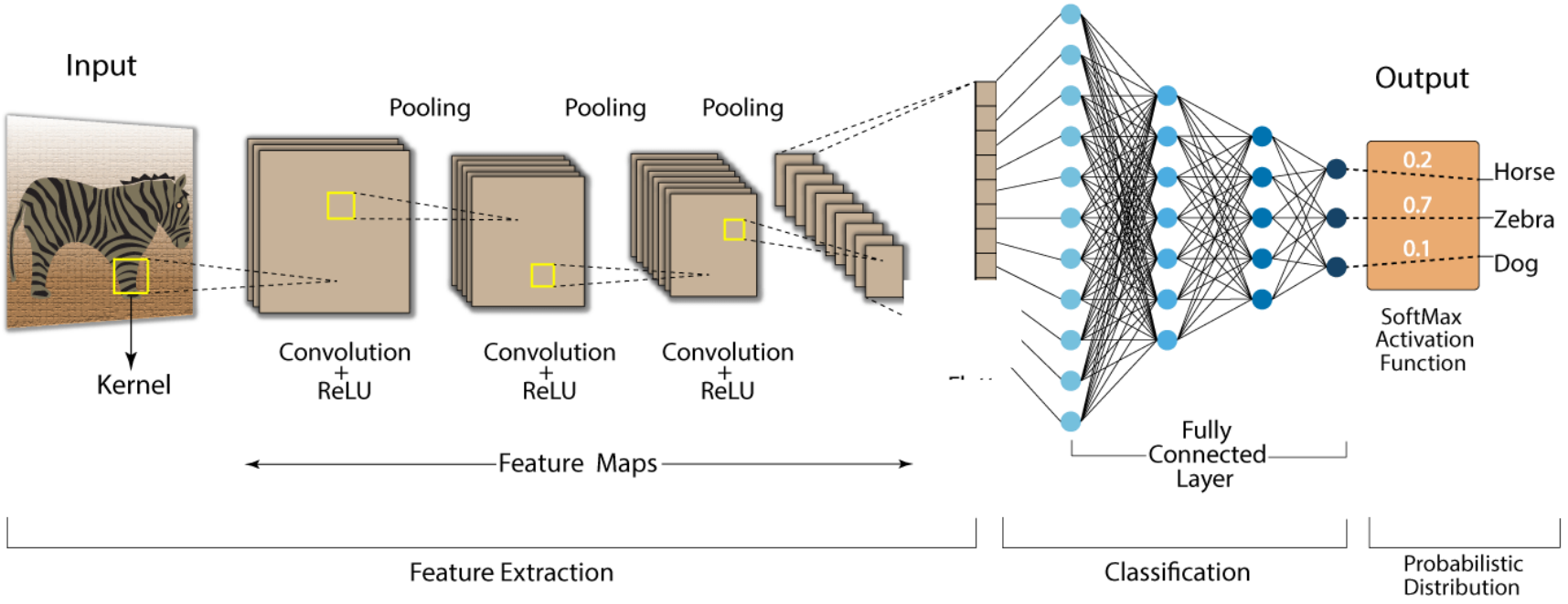
Non-Linearity: MaxPooling

- In addition to being a non-linearity…
  - it helps down-sample the image.
  - It helps summarize information in terms of larger blocks.



Putting It All Together

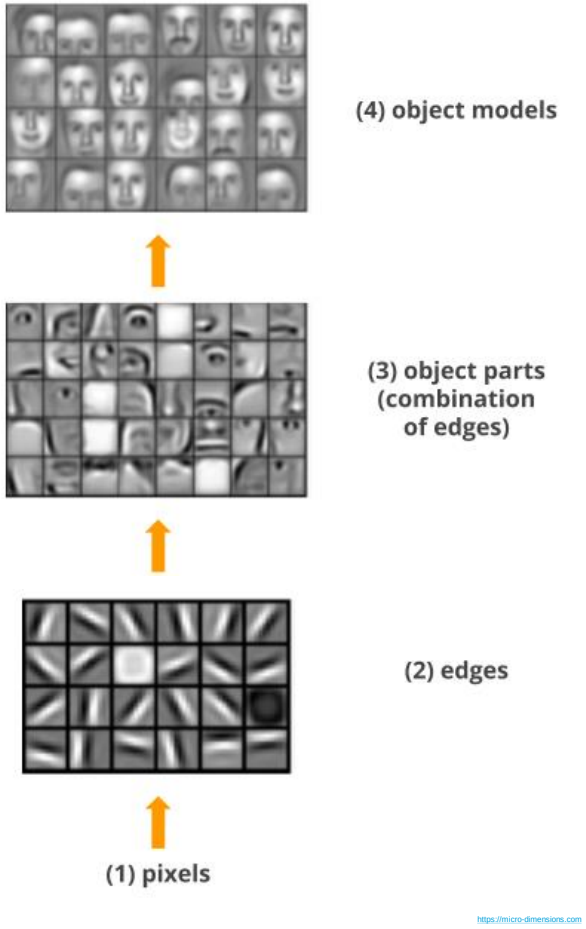
- Deeper layers generally have more kernels that are smaller.



Afaque Umer

Learned Features

- Early layers learn low-level features.
  - spots, edges, etc.
- Later layers learn to detect high-level features as a combination of low-level features.
  - Eyes, ears, hair, etc.
- Demo



Hyper-Parameters

Continued…

BOSTON UNIVERSITY

Batch Normalization

- Even if input data is properly normalized, the gradient in subsequent layers may vanish or explode.
- For that, you may add "*batch normalization"* at every layer.

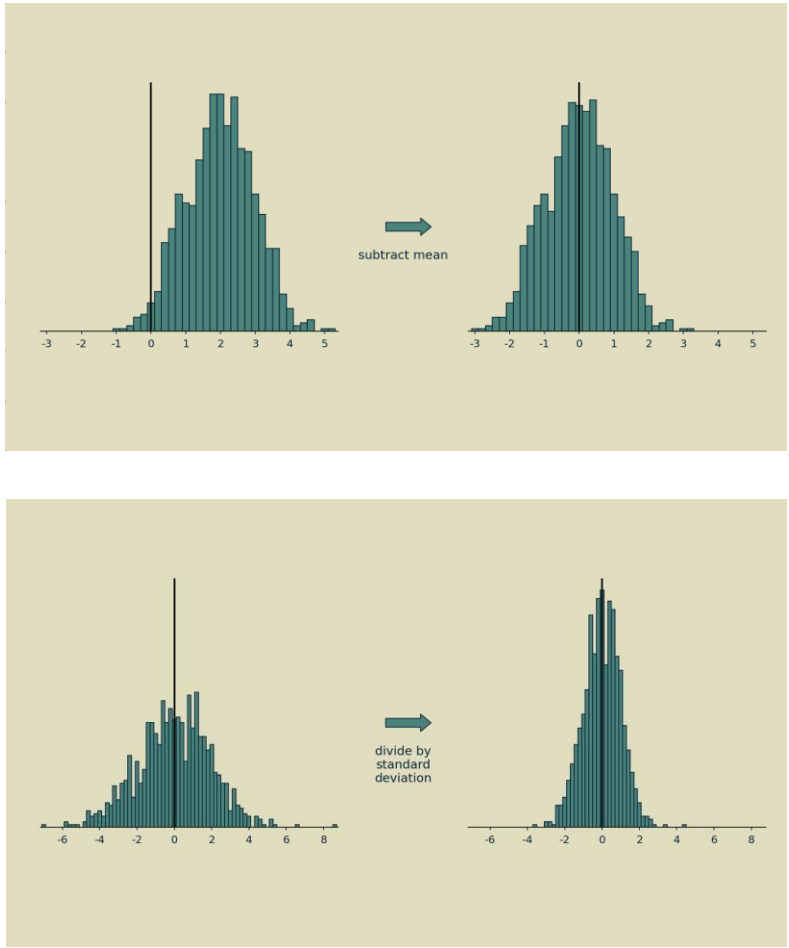


Figure courtesy of Brandon Rohrer

Learning Rate: Schedulers

- Since larger learning rates may converge faster but smaller ones are more stable, you could adjust the learning rate in phases to get the best of both worlds!
  - This way, you still converge but faster.
- Using a scheduler is a common practice.

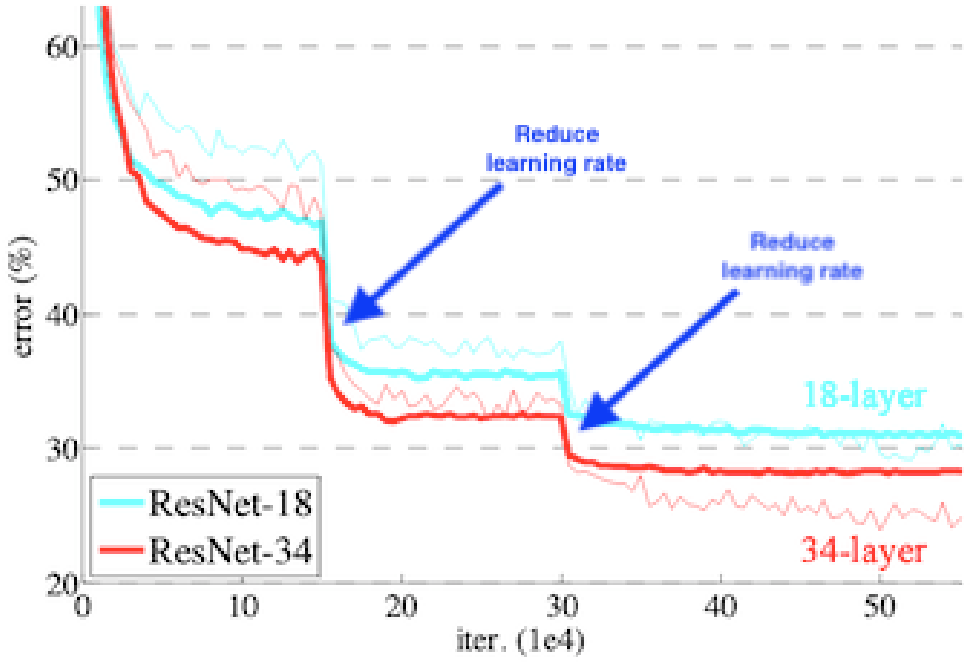


Figure courtesy of B. D. Hammel

Hyper-Parameter Tuning

BOSTON UNIVERSITY

Be Smart About It

- It is expensive!
  - 1 hyper-parameter with 3 values → 3 experiments
  - 2 hyper-parameter with 3 values each → 9 experiments
  - 3 hyper-parameter with 3 values each → 27 experiments
  - … exponential growth!

Be Smart About It

- It is expensive!
- Start with generally accepted wisdom:
  - Start with good initial guesses.
  - Different settings work better for different models/problems (e.g., SGD + momentum for computer vision vs. Adam otherwise)
- Be picky about what to fine-tune.
  - Use early stopping.
  - Learning rate is the most important parameter!

Hyper-Parameter Tuning Methods

- Generally, use log-scale for numerical hyper-parameters.
- Random and Adaptive searches generally find optimal values faster than grid searches.

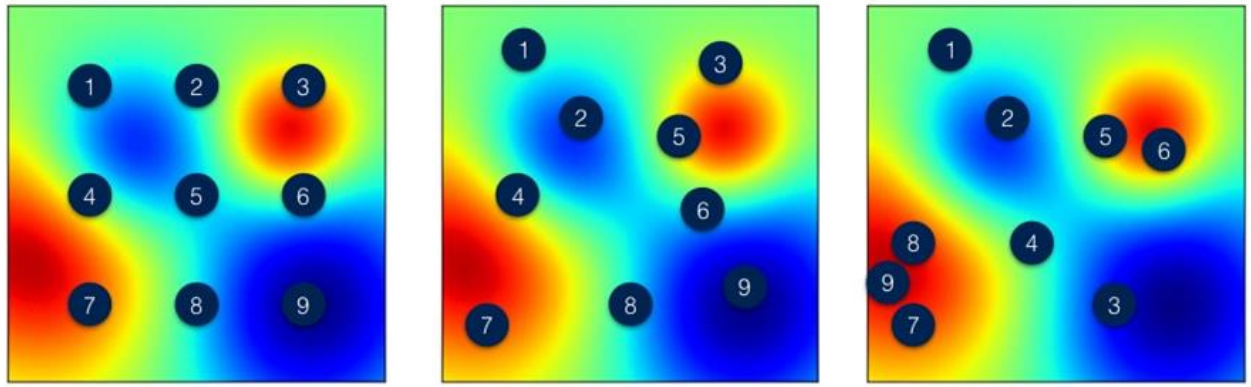


Figure courtesy of Liam Li

Troubleshooting

BOSTON UNIVERSITY

Results are bad?

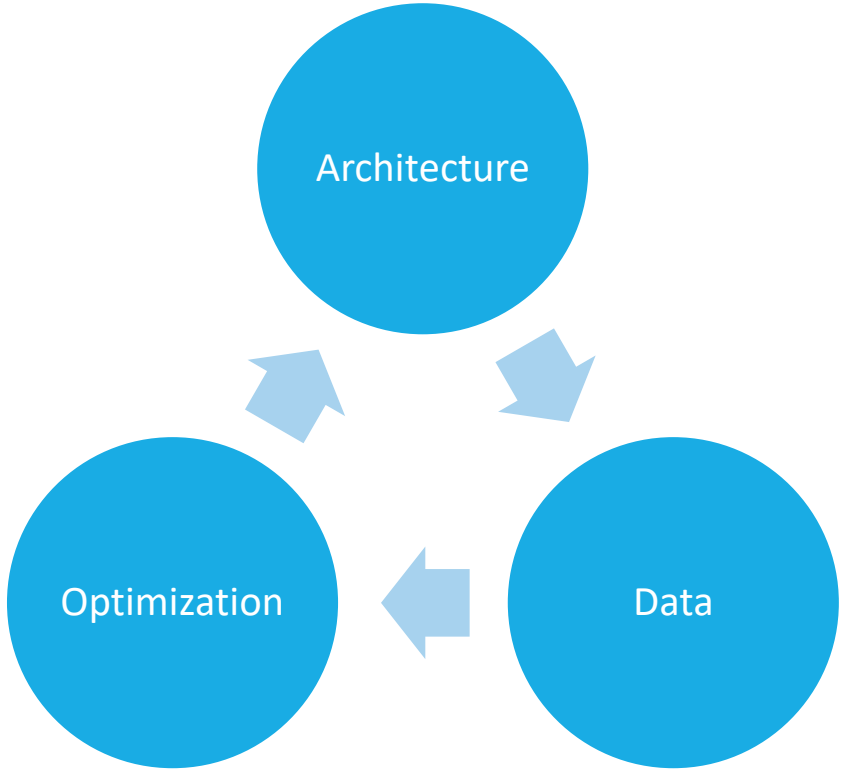
- Check against a benchmark!
  - paperswithcode.com
  - kaggle.com
- Are you overfitting or underfitting?

How do I improve my results?

- Best way: Get more **GOOD** data
  - If not, clean-up existing data.
- Are you overfitting or underfitting?
  - Overfitting: get more data or use a less complex model.
  - Underfitting: get a more complex model.
- Keep it simple!
  - Start with a simple model, simple data, simple code.
  - Test by component
  - Test by example

Debugging Practices

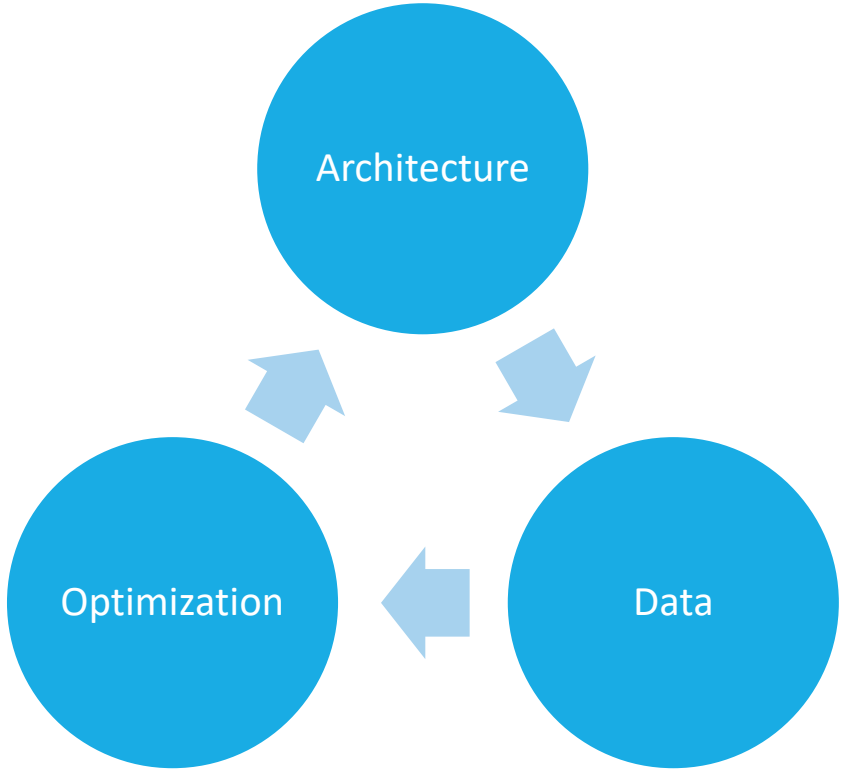
- If data comes from a benchmark, go look at results published by others to compare to (e.g., leaderboards[1]).
- The most straight forward way to improve performance is getting more data!
  - Is your data good quality? Low-quality data could lead to bad performance.
- Ask the question: Is my model…
  - Underfitting? (high training error)
  - Overfitting? (low training error, high test error)



1- https://paperswithcode.com/sota

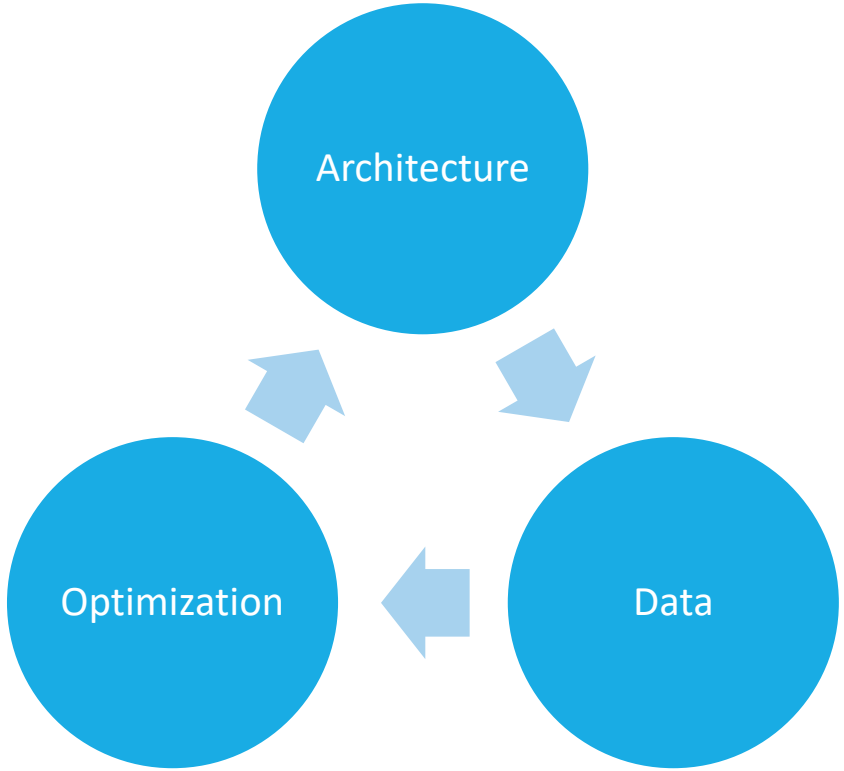
Debugging Practices

- If overfitting,
  - Use smaller models.
  - Add regularization.
  - Add/generate more data.
    - Data augmentation...
- If underfitting:
  - Might need a deeper model. But a deeper model does not guarantee finding the best minima without proper optimization.



Debugging Practices

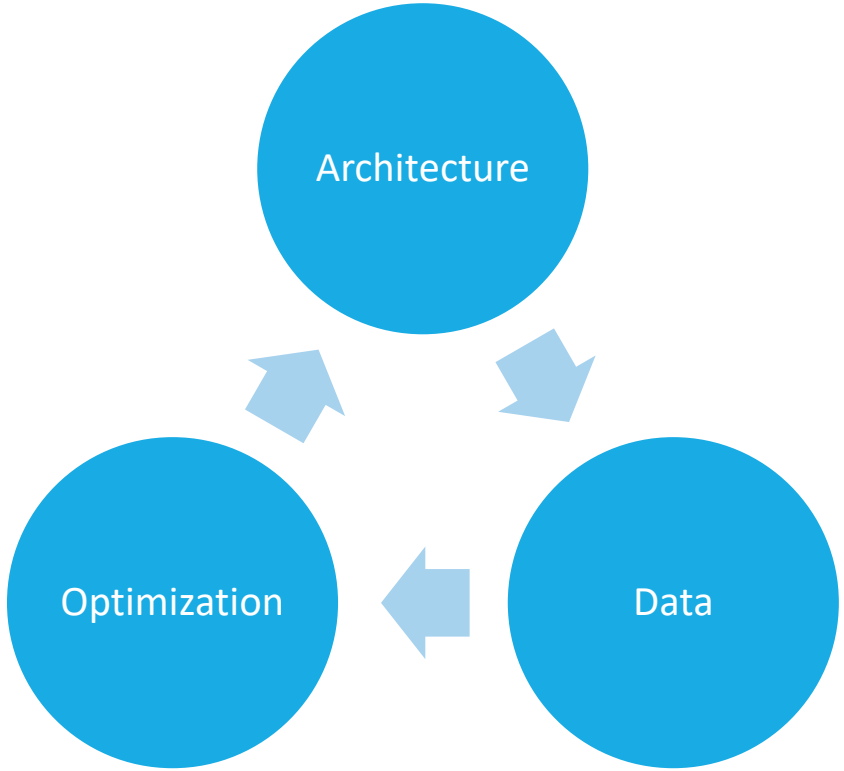
- **Start simple[1]:** Simple model. Few data point. Something quick.
- Test by module.
- Test by example.
  - Look at extreme cases, outliers, …



[1]- Checklist for debugging neural networks

Debugging Practices

- Look at the distributions of gradients and weights:
  - The updates should be in the order of magnitude of 1% of weight values.
  - The eigenvalues of the weight matrices should be close to 1. Otherwise, there will be a vanishing/exploding gradient problem.[1]



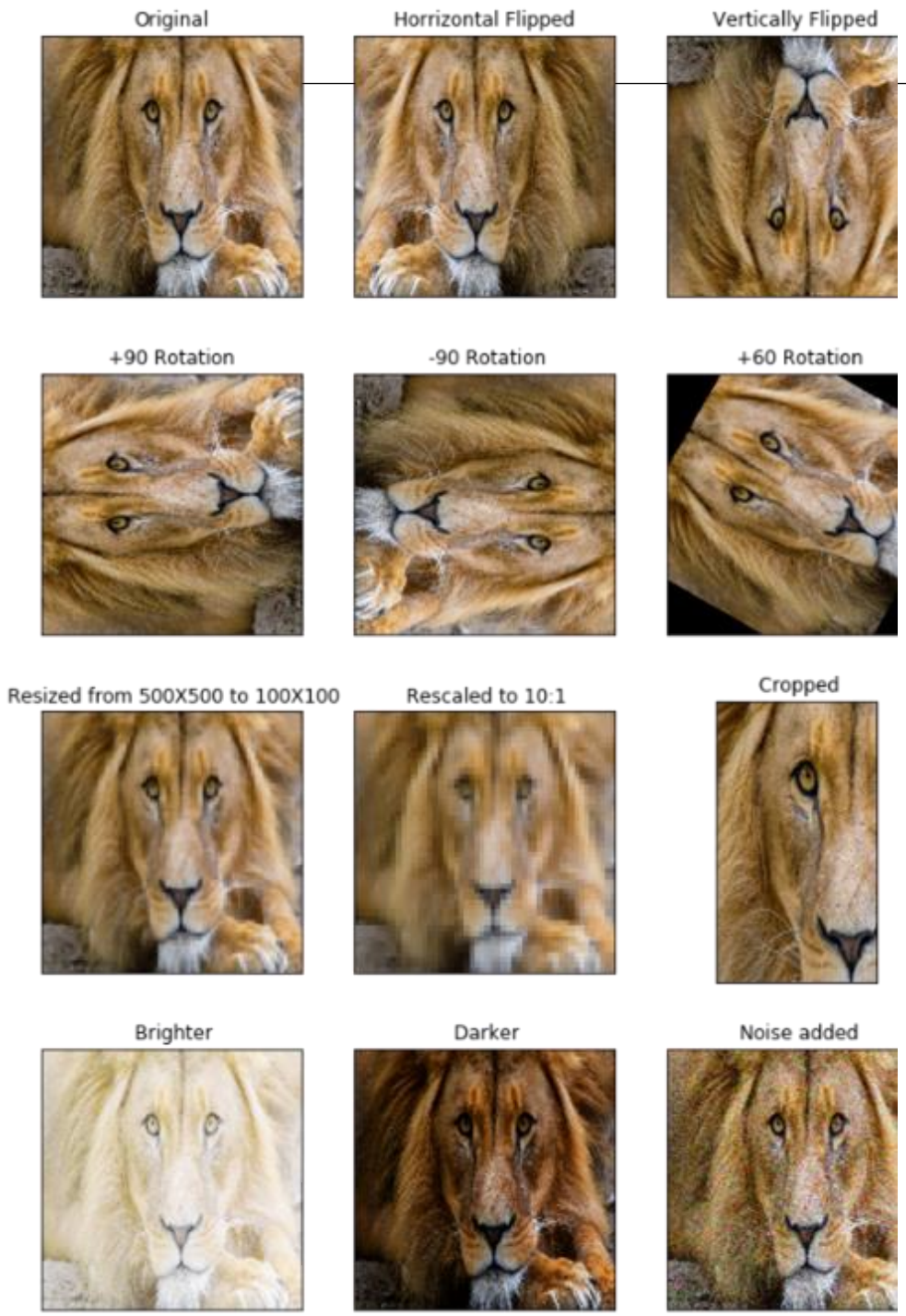
1- Deep Learning Book

CNNs more

BOSTON UNIVERSITY

Common Practices in Vision: Transfer Learning

- Deeper networks are highly non-convex and hard to train with little data. Hence, weight initialization becomes crucial.
- How about we start with weights that are pre-trained on a large set of images?
  - Such a model would already have learned useful features for a target problem.
- So, we start with a pretrained model on ImageNet, and then fine tune on the task at hand.



Common Practices: Data Augmentation

- What is a lion exactly?
- We need to make sure the network is not learning too specific (spurious) features.
- So, let's create as many variations as possible of this image to force the model to learn the true meaning of being a lion.
- This will also increase amount of data!

Figure courtesy of Great Learning Team

Historical perspective

- Advanced in GPU have enabled the use of deep models (e.g., ResNet) with millions of parameters.
- Availability of large labeled datasets (e.g., ImageNet) have also contributed to deeper models.
- Deeper models generally lead to higher accuracy.

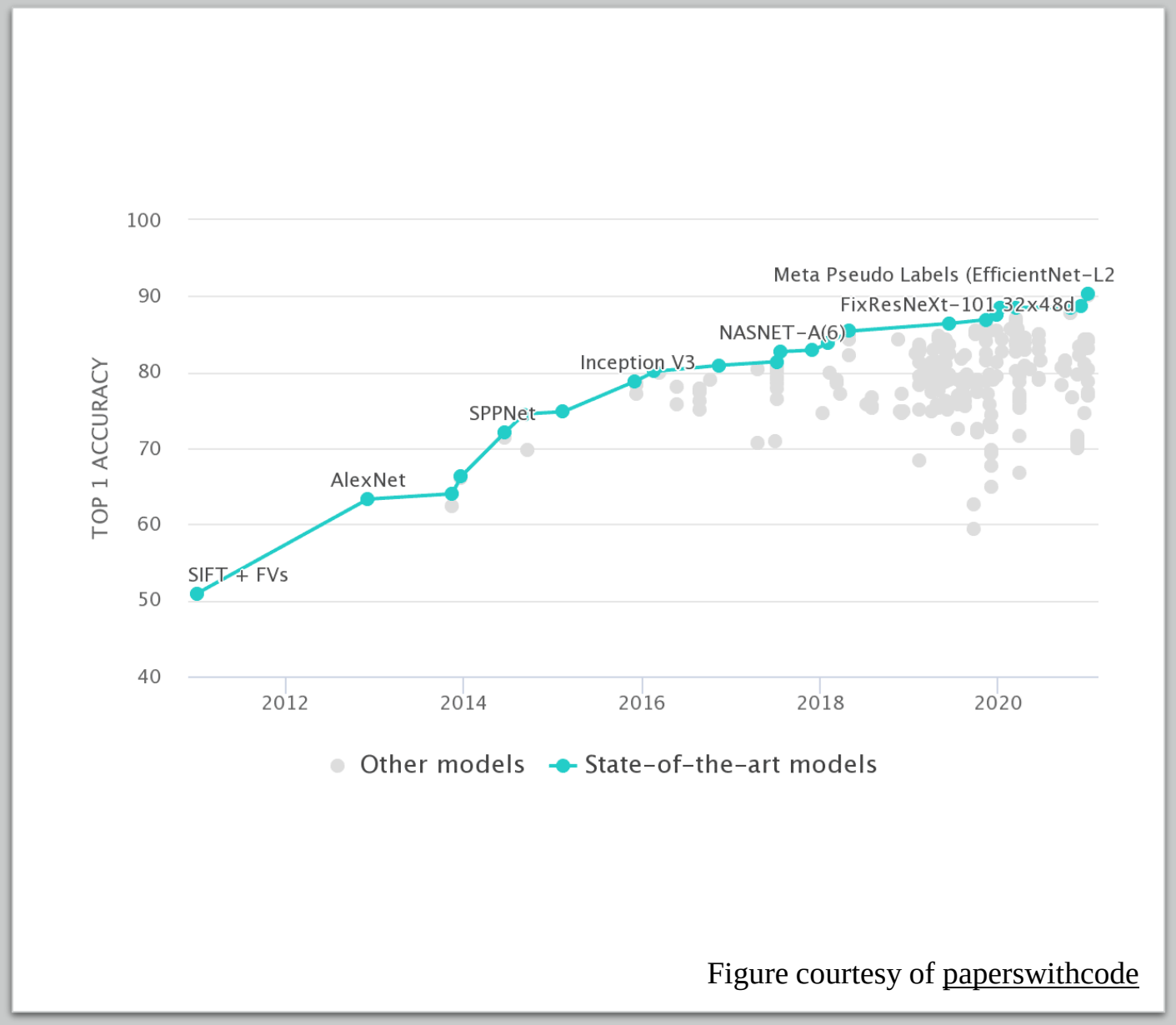


Figure courtesy of paperswithcode

Historical perspective

- AlexNet (2012):
  - Used data augmentation.
  - Used ReLU.
  - Dropout.
- VGGNet (2014):
  - Replacing large kernels with multiple smaller sized kernels. This results in fewer parameters.

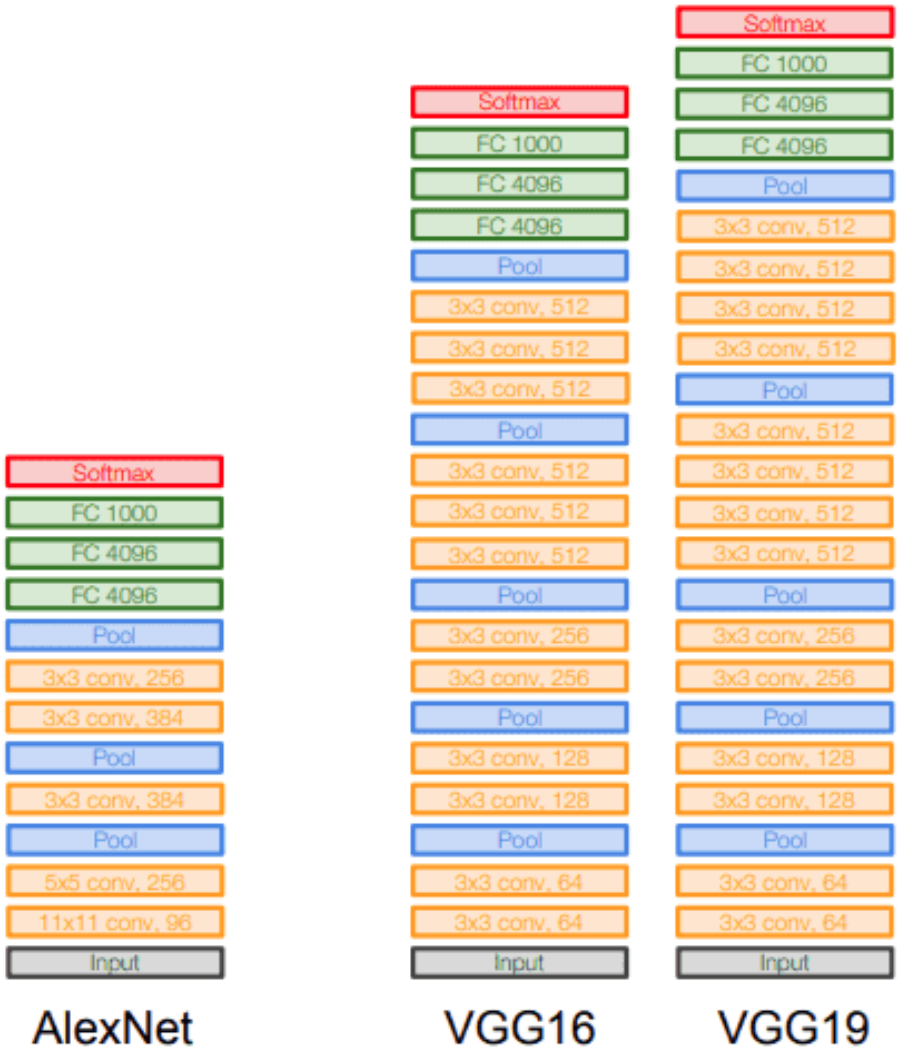


Figure courtesy of Standford 2017 Deep Learning Lectures

Historical perspective

- AlexNet (2012):
  - Used data augmentation.
  - Used ReLU.
  - Dropout.
- VGGNet (2014):
  - Replacing large kernels with multiple smaller sized kernels. This results in fewer parameters.

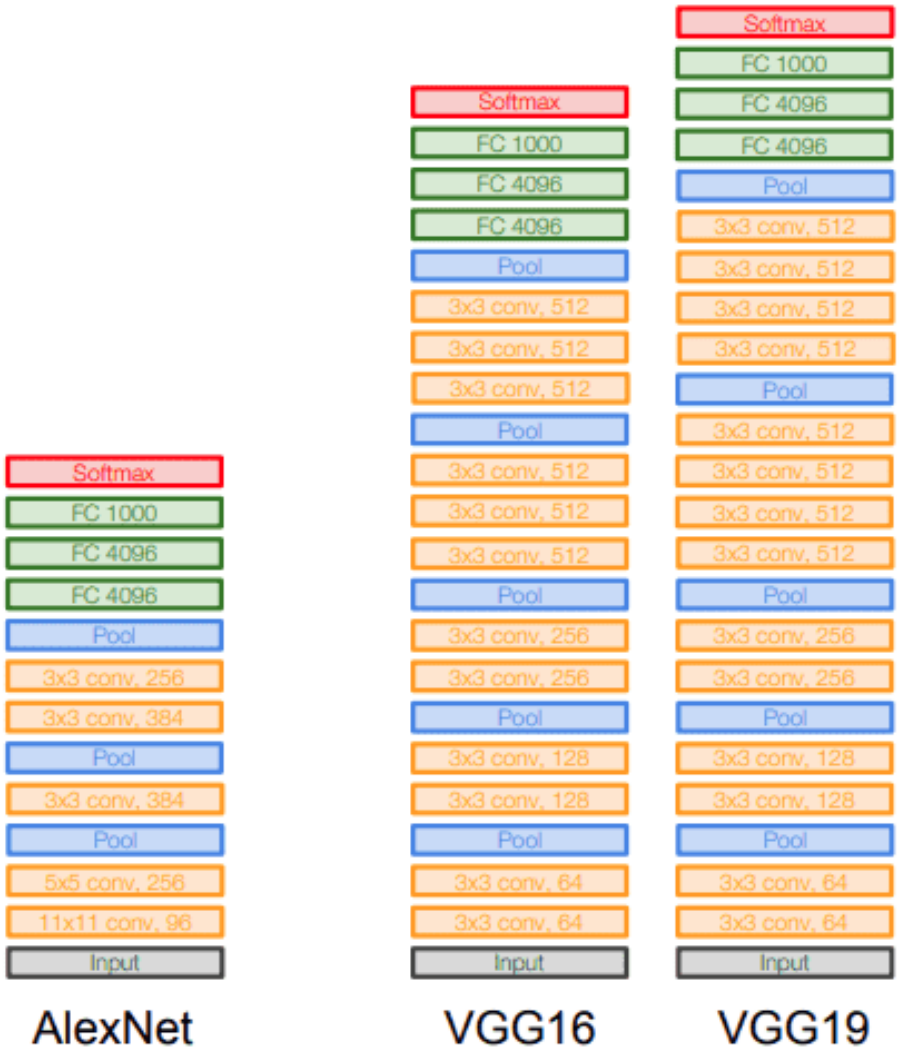


Figure courtesy of Standford 2017 Deep Learning Lectures

Historical perspective

- GoogLeNet/Inception (2014):
  - Having multiple branches of different convolutional sizes concatenated.

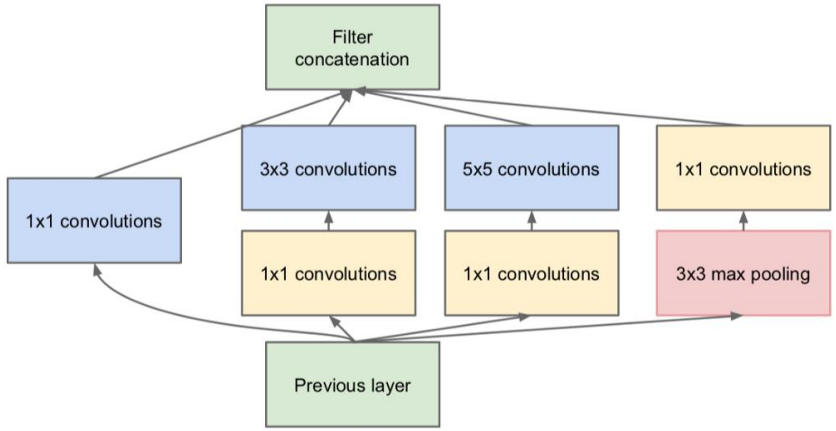


Figure courtesy of Bo Zhao

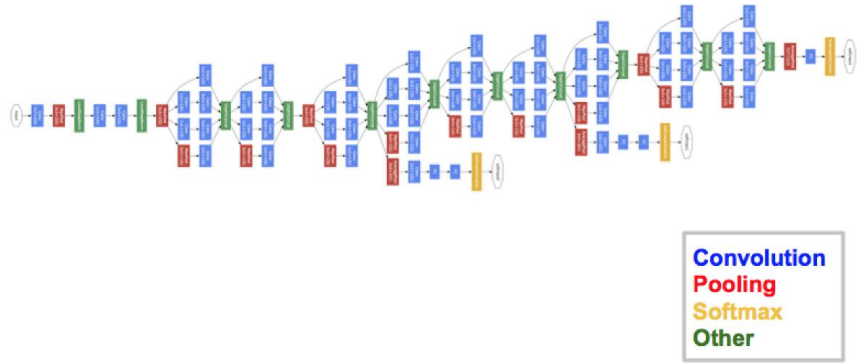


Figure courtesy of Siddharth Das

Recent Trends: Residual Networks (ResNets)

- ResNet (2015):
  - Adding shortcuts (residual connections) helps let information flow through the network, unimpeded by nonlinearities.

- Currently the most commonly used network in computer vision.

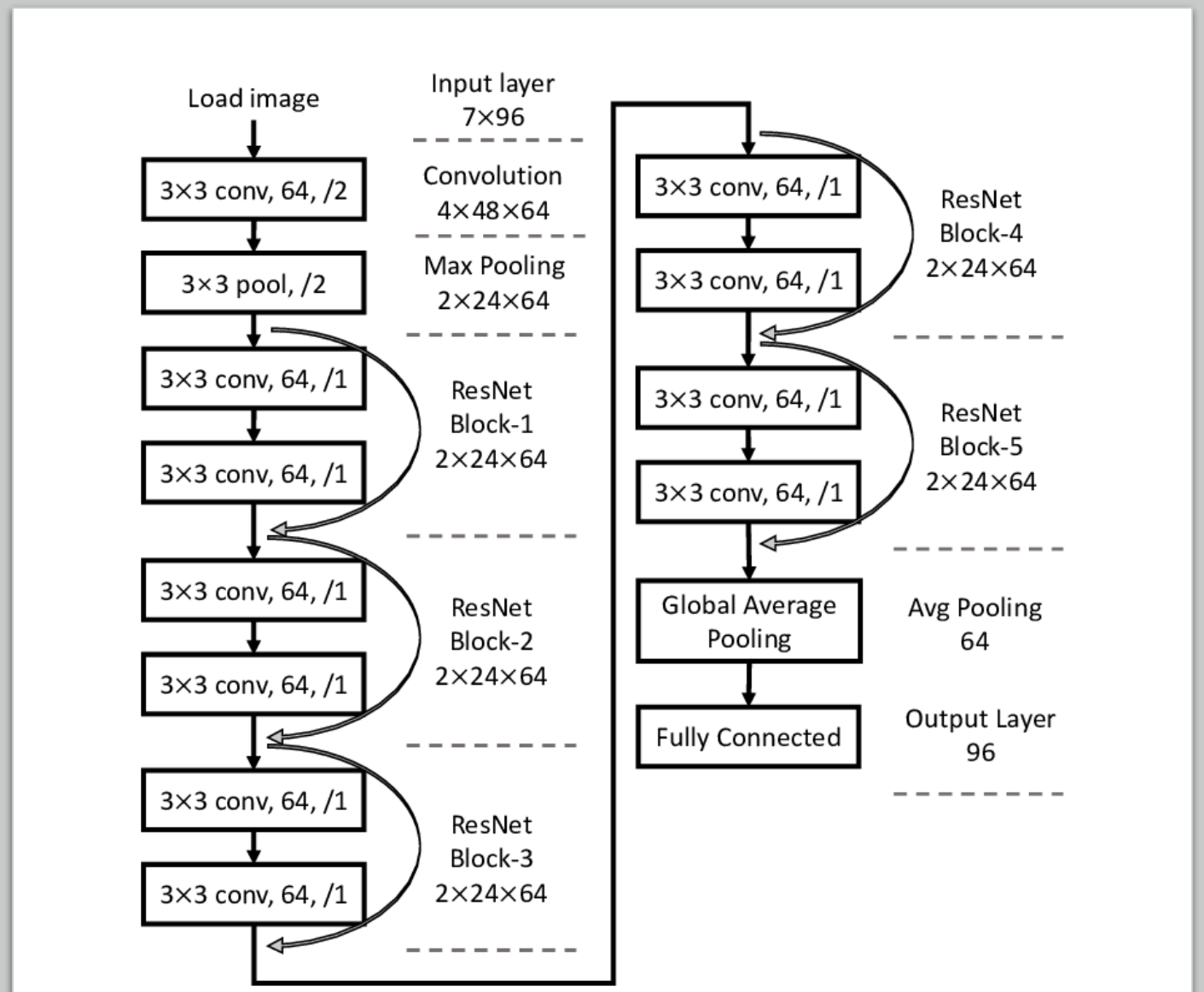


Figure courtesy of Seunghyoung Ryu

BOSTON UNIVERSITY

Historical perspective

- DenseNet (2018)
  - Extends on the idea of ResNet's skip connections by connecting each layer to every other layer in a feed-forward fashion.

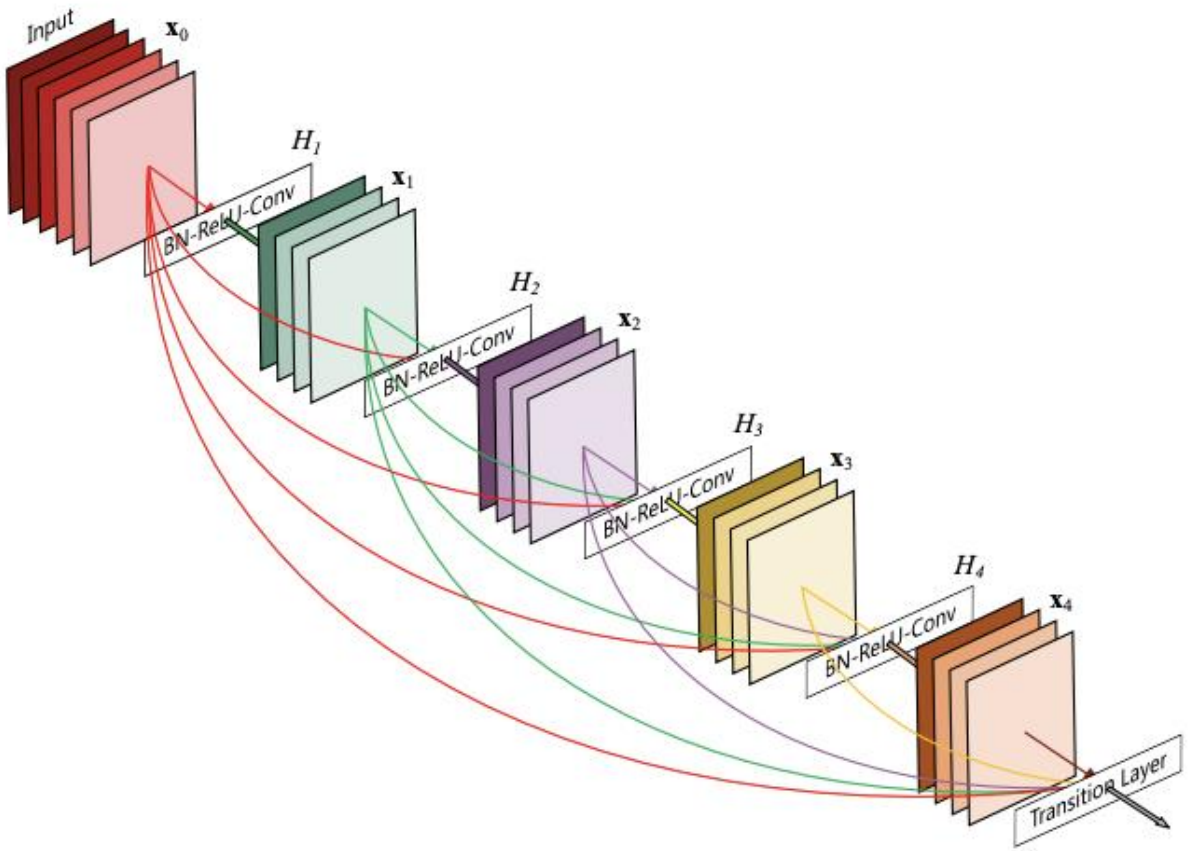


Figure courtesy of Huang et. al.

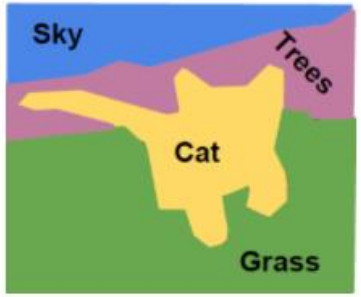
Computer Vision Tasks

Label	Probability
Car	0.001
Dog	0.2
Human	0.001
Cat	0.71
Chair	0.0002
Etc...	

Classification

Instance Segmentation

Object detection



Semantic segmentation

Figure courtesy of Rahul Purohit

Visualization and Understanding

- Visualizing the activations of hidden layers for an input image can help understand how the model works.
  - As we move to deeper layers into the model, the image resolution goes down because deeper layers learn more global (high-level) descriptors.
  - Deeper activations are less visually interpretable.

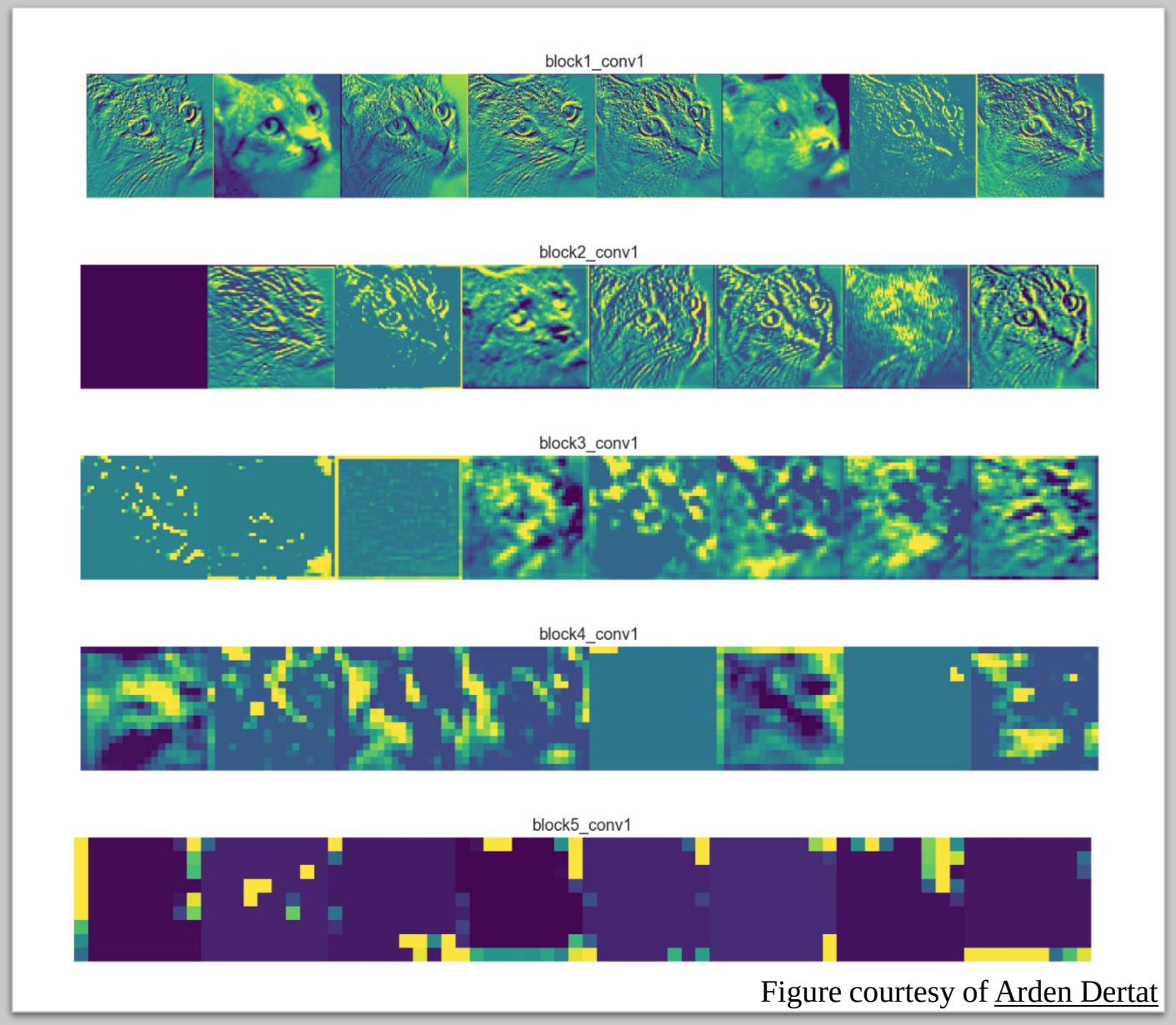


Figure courtesy of Arden Dertat

Visualization and Understanding

- Visualizing the kernels (filters) or what kind of features they respond to can reveal how the network works.

Visualization of the filters of early layers.

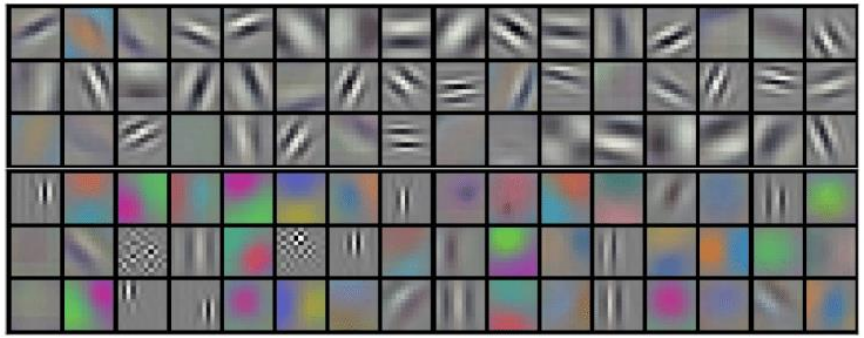


Figure courtesy of Bruno Eidi Nishimoto

Activation Maximization

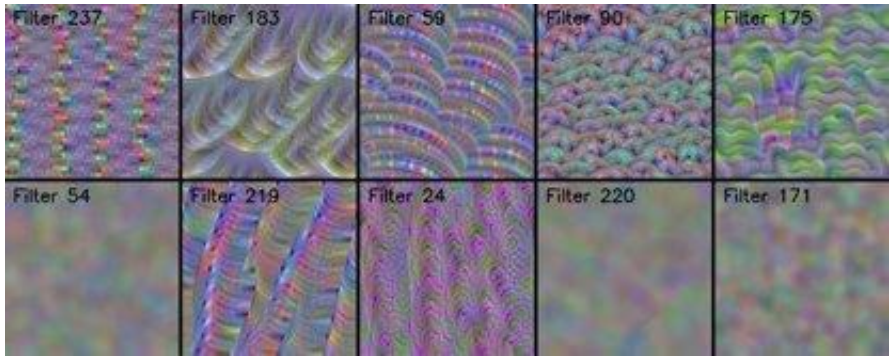


Figure courtesy of Keras-vis

Visualization and Understanding

- Saliency maps can be a great tool to understand which pixels of the image contribute to the highest change in model output.
- This method is based on using “guided backpropagation” which finds the gradients of the output w.r.t the **input.**

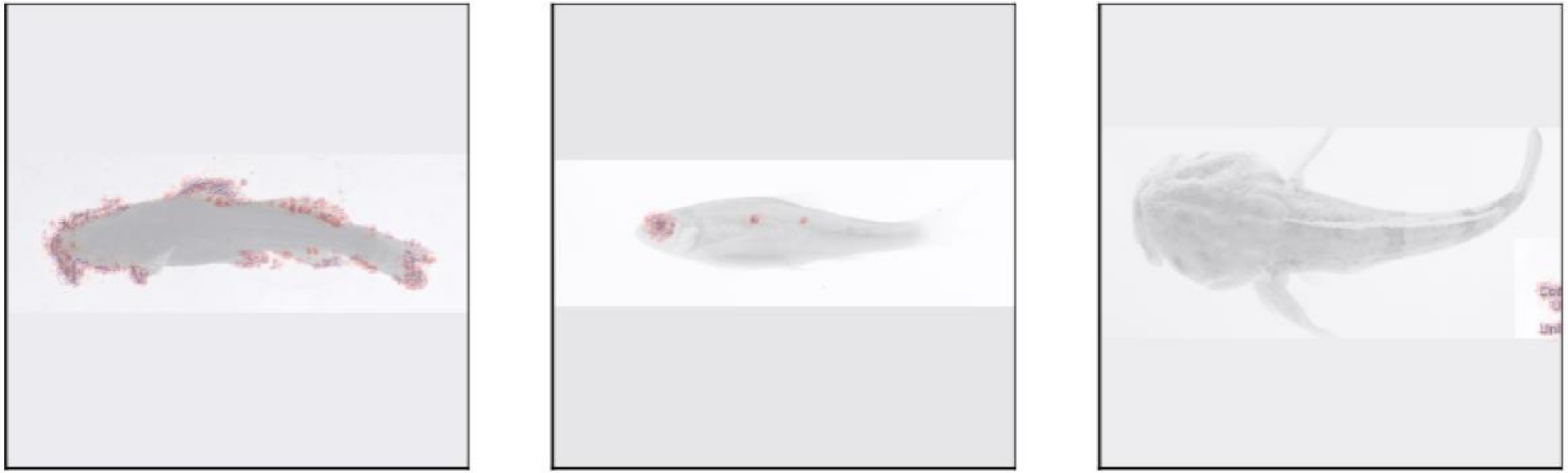


Figure courtesy of Mohannad Elhamod

Visualization and Understanding

- Class Activation Maps (CAMs) are another technique that is less noisy and focuses more on regions rather than pixels.
- This method is based on weighted averaging of the different feature maps.

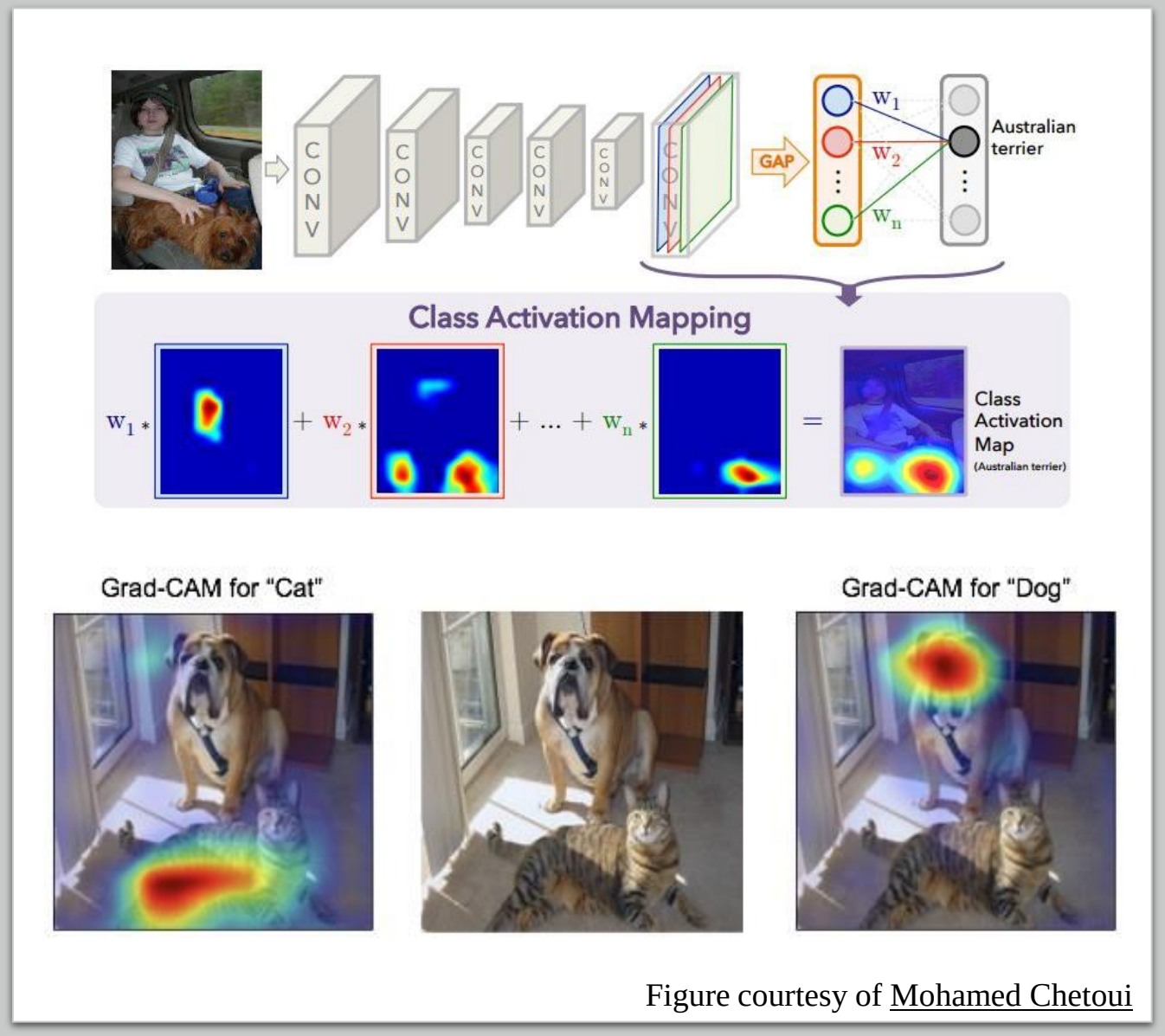


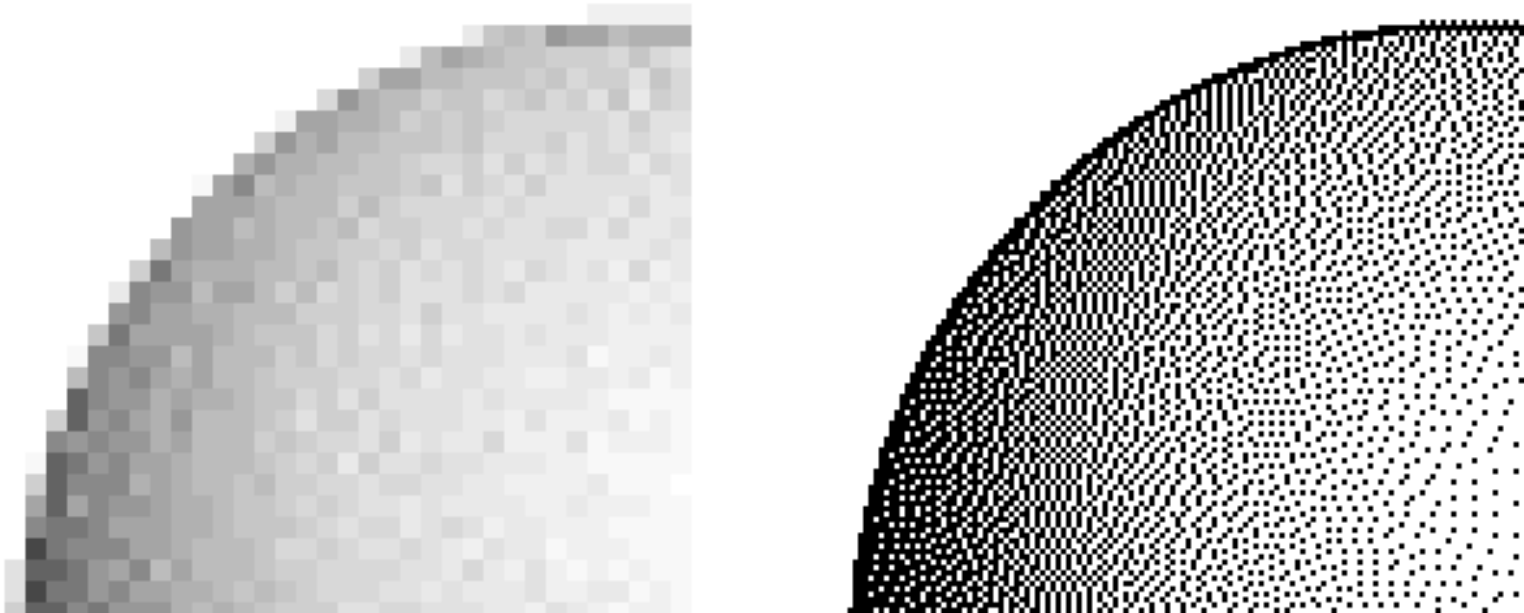
Figure courtesy of Mohamed Chetoui

Auto-encoder

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?*

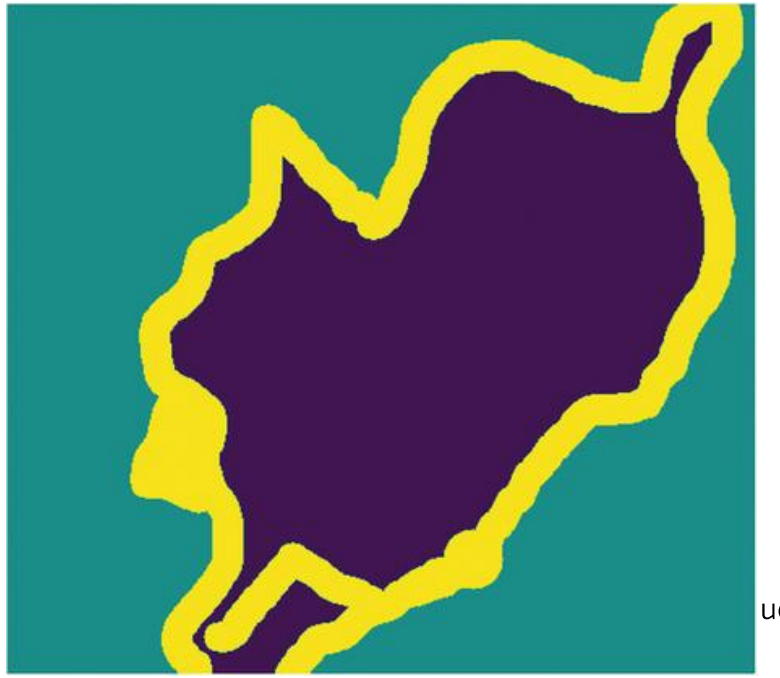


BOSTON UNIVERSITY

Image-to-Image Prediction

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.