

# Intro to Neural Networks

**BA865 – Mohannad Elhamod**

# CNNs

## Convolutional Networks

# A Problem of Scalability

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

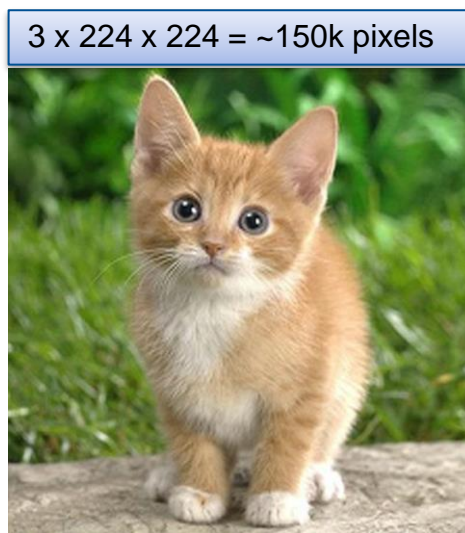


Figure courtesy of Robert Hod

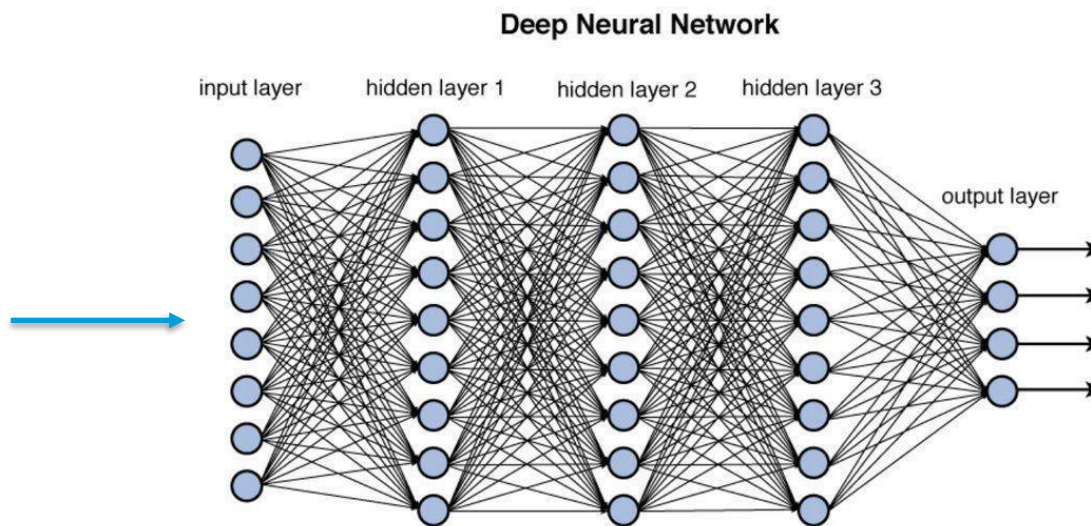


Figure 12.2 Deep network architecture with multiple layers.

Figure courtesy of Ravindra Pamar

# Structure in Images

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



Figure courtesy of Robert Bradi

vs.

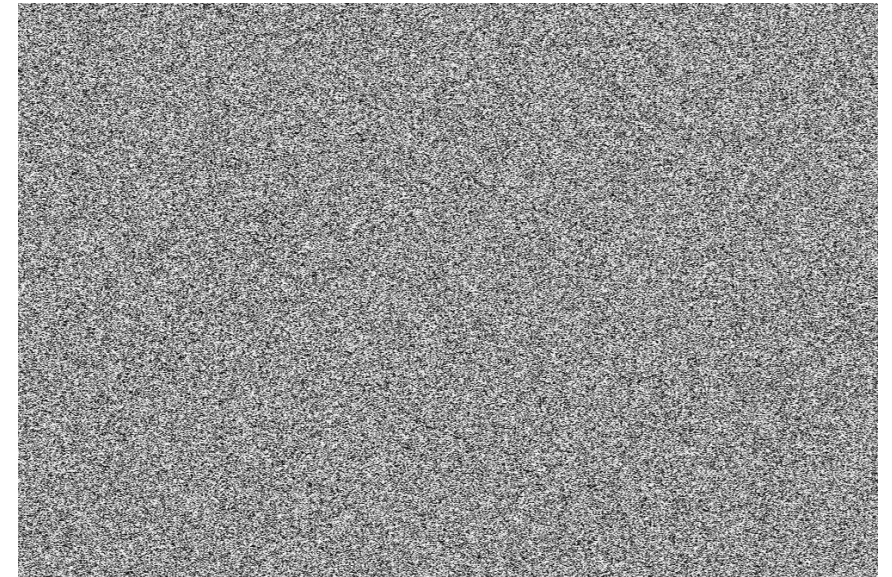


Figure courtesy of Jorge Stolfi

# 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!

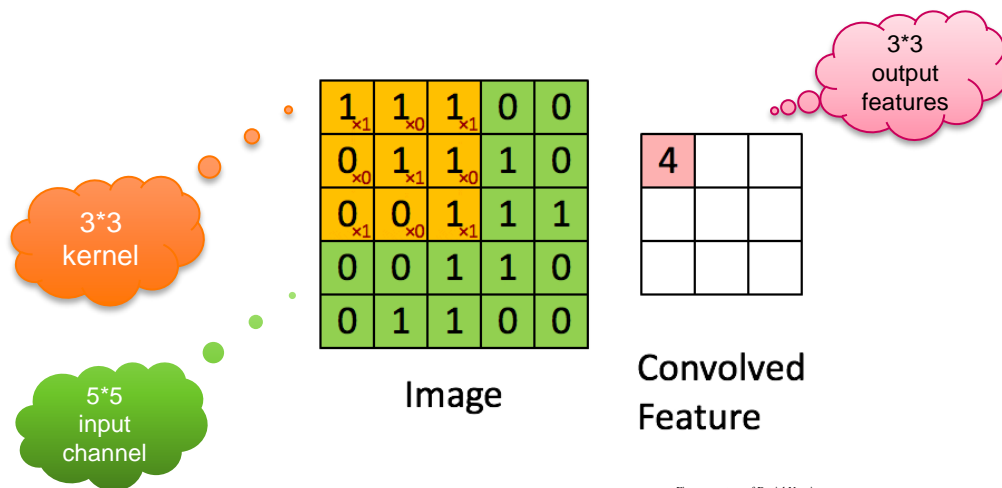


Figure courtesy of Daniel Nouri

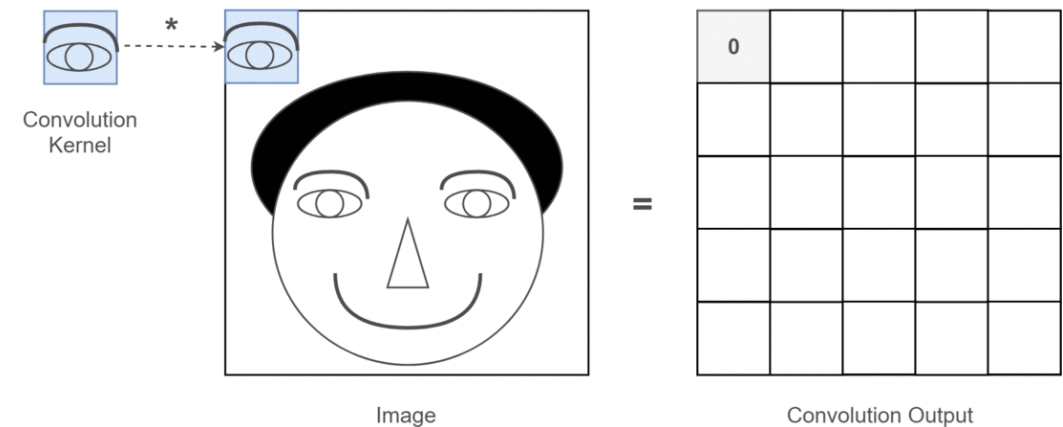
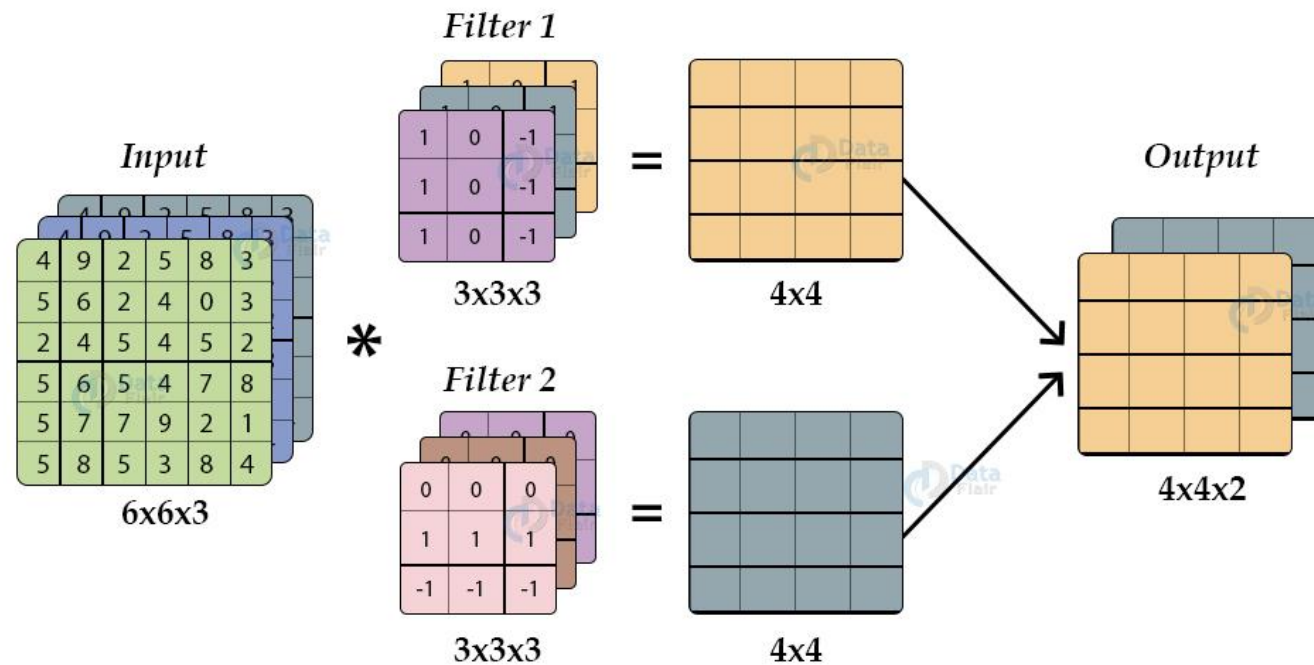


Figure courtesy of Thushan Ganegodara

# Mathematically Speaking...

## Convolution Layer in Keras

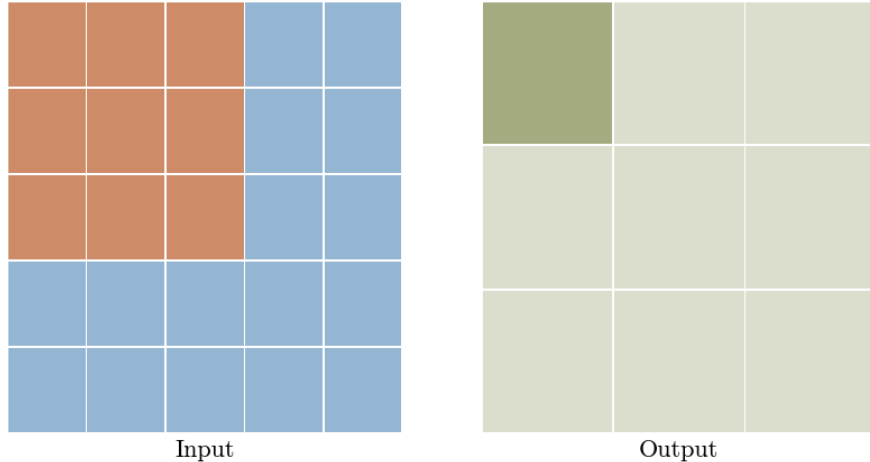




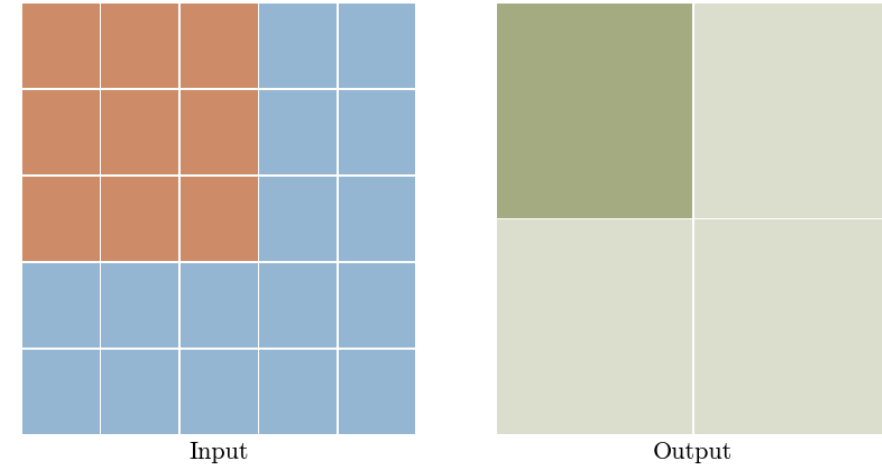
# Stride and Padding

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

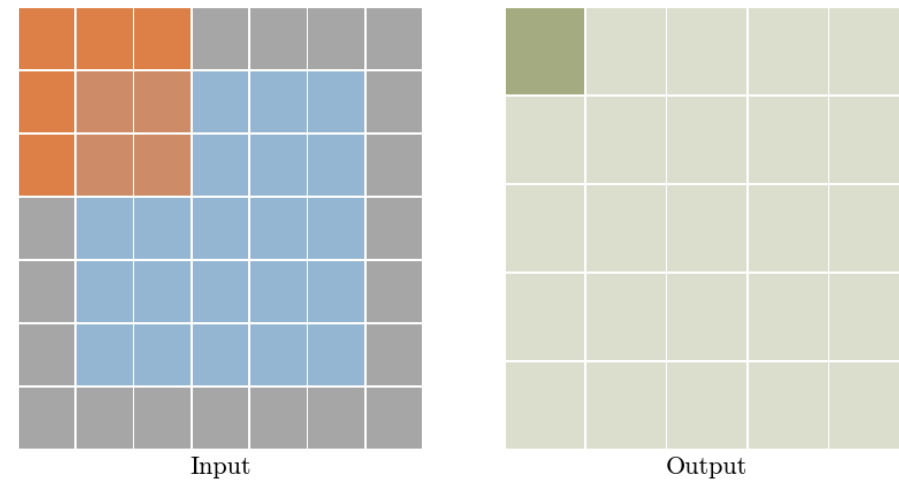
Type: conv - Stride: 1 Padding: 0



Type: conv - Stride: 2 Padding: 0



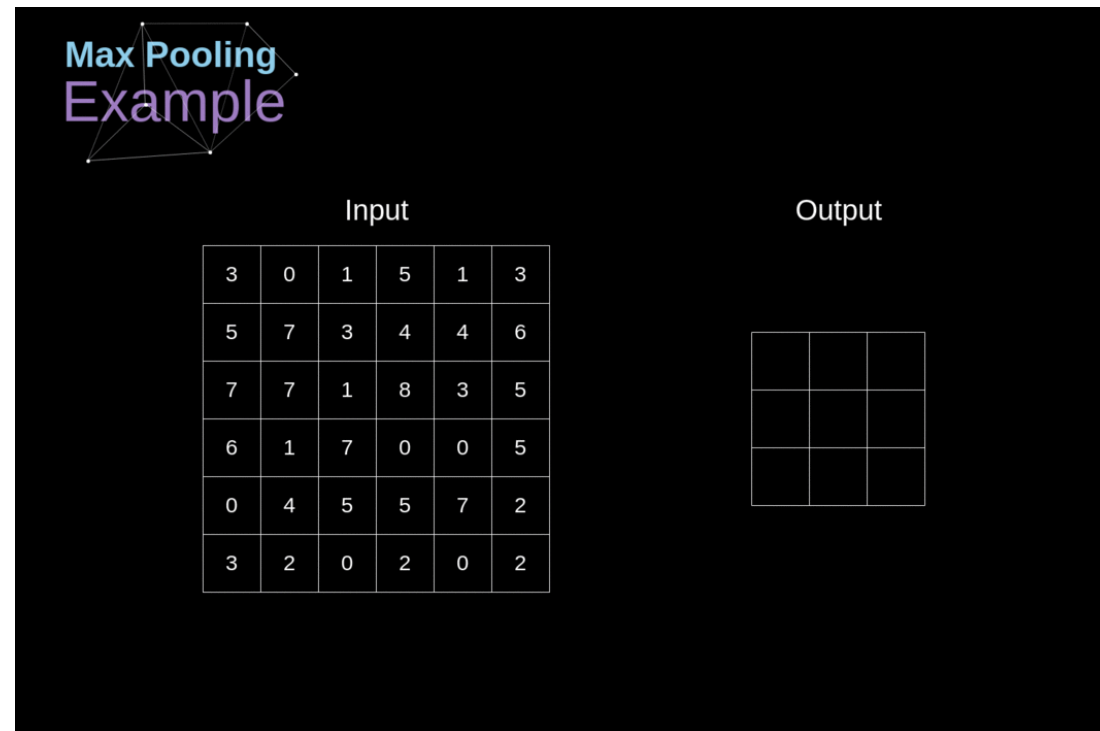
Type: conv - Stride: 1 Padding: 1



[Aqeel Anwar](#)

# Non-Linearity: MaxPooling

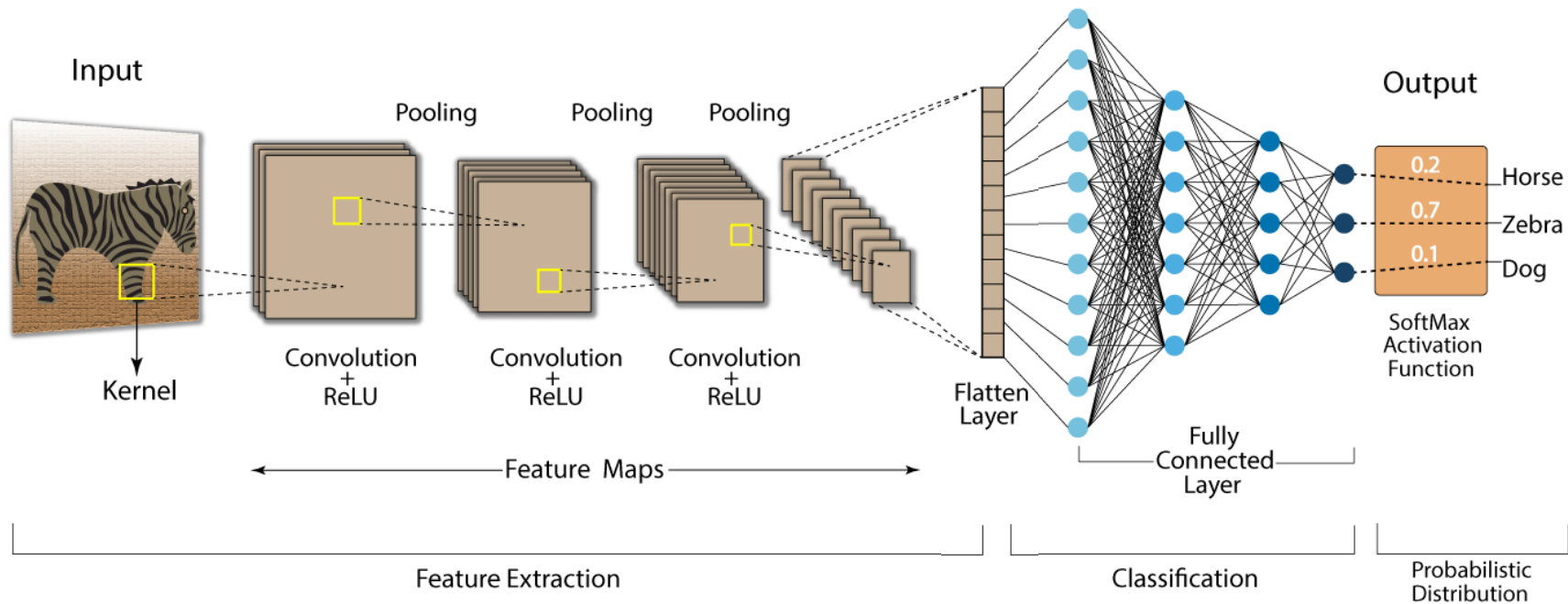
- In addition to being a non-linearity...
  - it helps down-sample the image.
  - It helps summarize information in terms of larger blocks.





# Putting It All Together

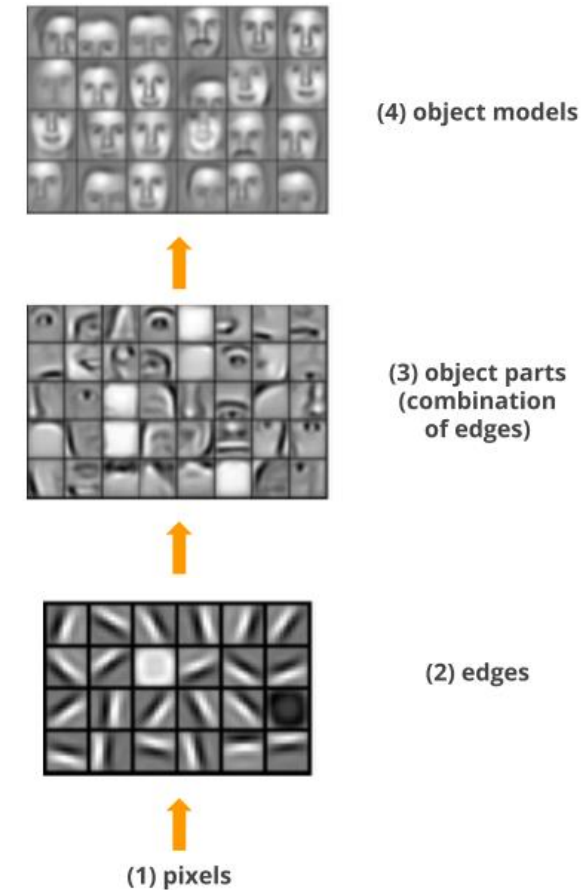
- Deeper layers generally have more kernels that are smaller.



Afaq 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.
- Interpretability is not guaranteed (But there is great research interest...)
- [Demo](#)



# Hyper-Parameters

**Continued...**

# 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.

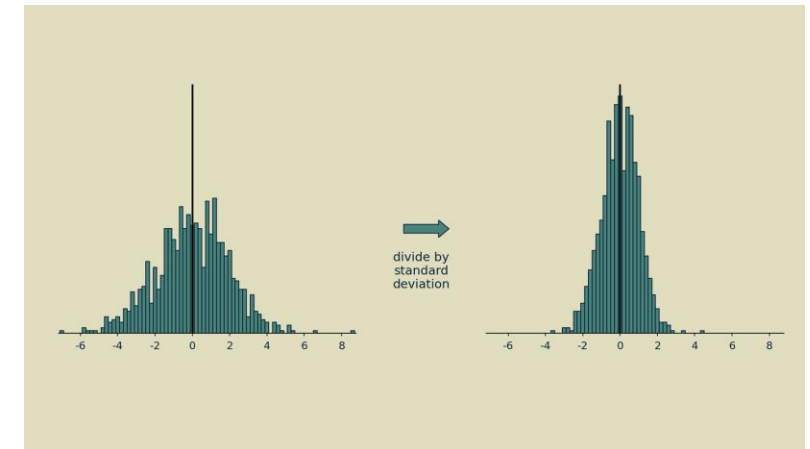
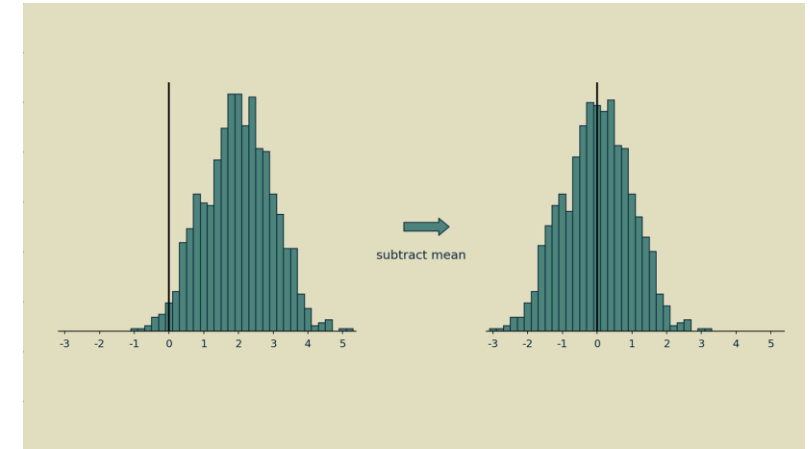


Figure courtesy of Brandon Rohrer

# 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.

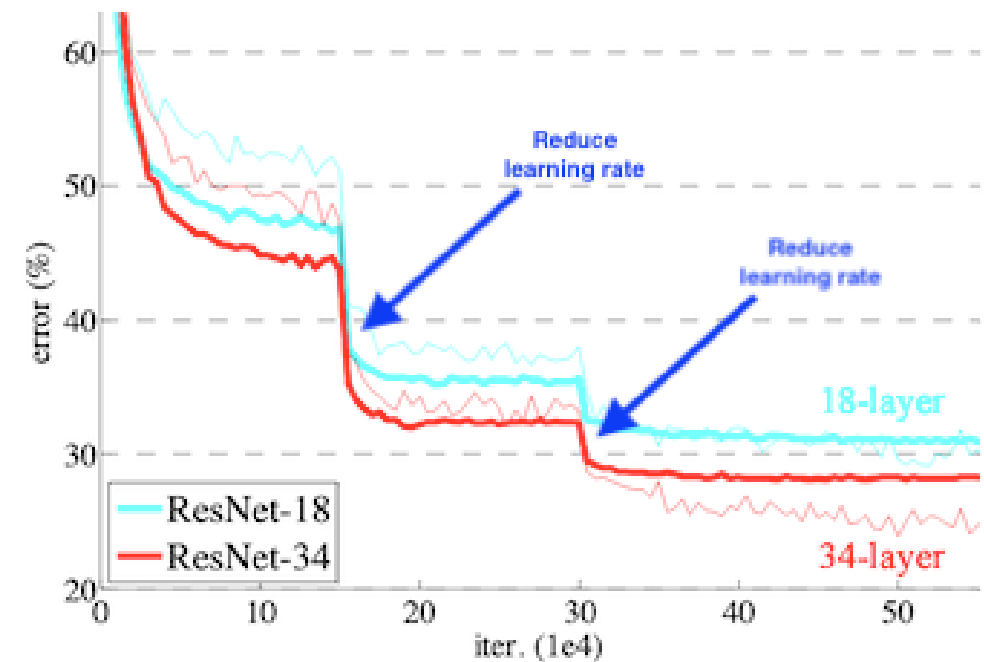


Figure courtesy of [B. D. Hammel](#)

# Hyper- Parameter Tuning

# 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!

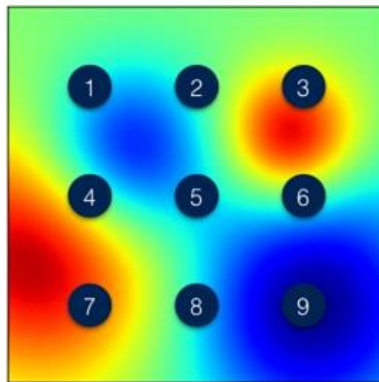


# Be Smart About It

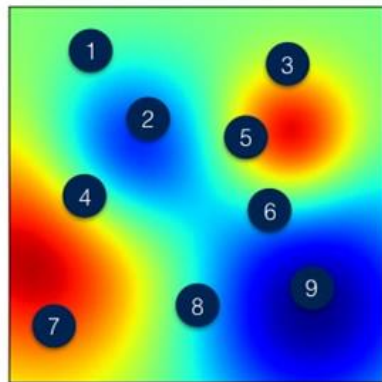
- It is expensive!
- Start with generally accepted wisdom:
  - Start with good initial guesses.
  - Different settings work better for different models/problems (e.g., SGD + momentum for computer vision vs. Adam otherwise)
- Be picky about what to fine-tune.
  - Use early stopping.
  - Learning rate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

# Hyper-Parameter Tuning Methods

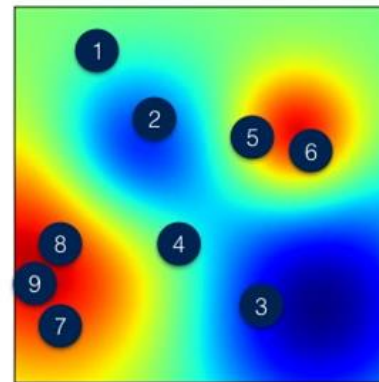
- Generally, use log-scale for numerical hyper-parameters.
- Random and Adaptive searches generally find optimal values faster than grid searches.



Grid Search



Random Search



Adaptive Selection

Figure courtesy of Liam Li

# Advanced Techniques

# Transfer Learning

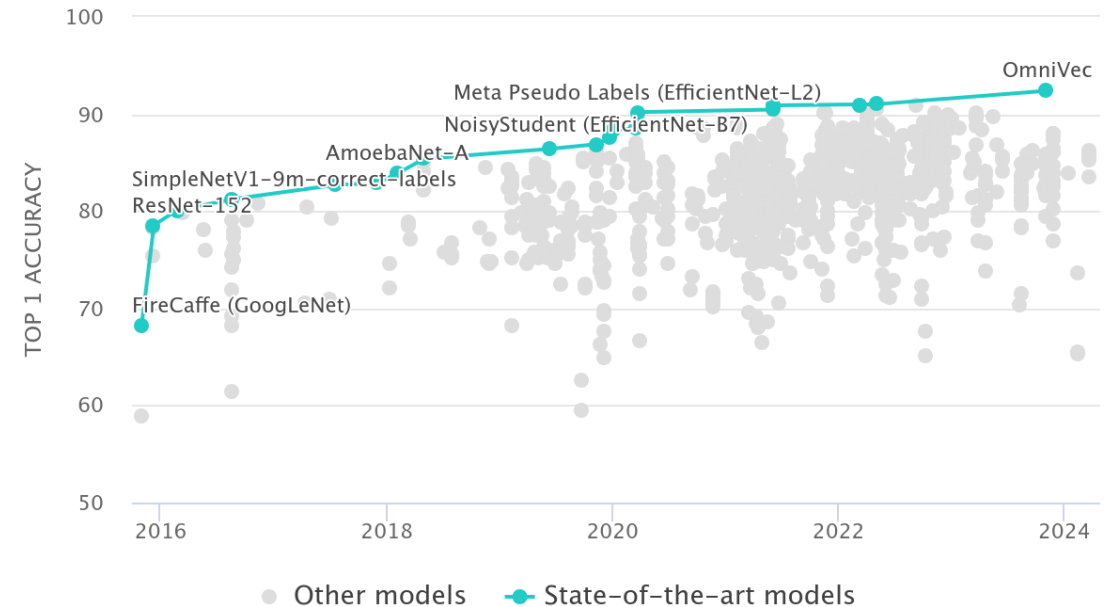
- Deeper networks have highly irregular loss surfaces. They are hard to train:
  1. They need relatively large amounts of data.
  2. They need relatively much compute resources to train and tune hyper-parameters.
- We need to somehow start with “*an advantage*”.

# Transfer Learning

- We need to somehow start with “*an advantage*”.
- Large tech companies and research institutes are more capable than individuals in terms of data and compute resources.
  - They can afford to train their models from scratch.
  - Can we capitalize on their pre-trained models?

# Transfer Learning

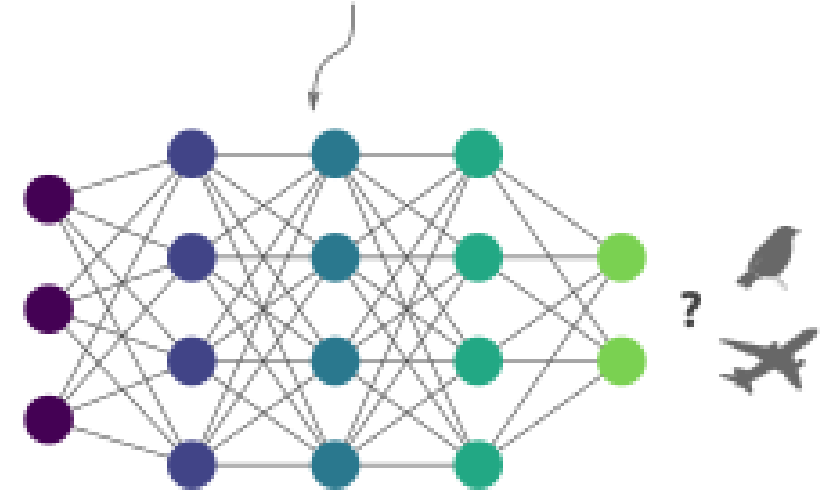
- A pre-trained model would already have learned useful features for a target problem.
  - For example, we can start with a model (e.g., ResNet) that was pre-trained on a large dataset (e.g., [ImageNet](#): ~1.4M images. 1000 classes. ~3\*469x387 pixels).



# Transfer Learning

- The pre-trained network was trained on Task A (e.g., bird vs. plane).
- Replace the network's FC layers with new ones appropriate for Task B (e.g., other species).

The original, "**pre-trained**" network, trained for binary classification of birds vs planes

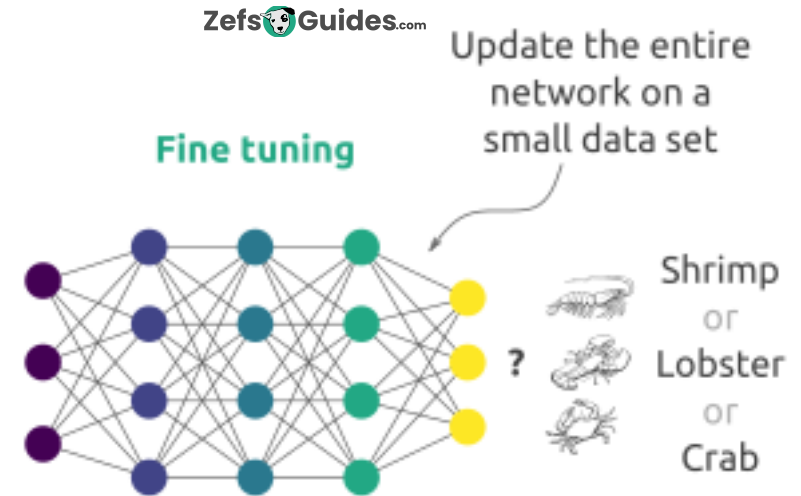
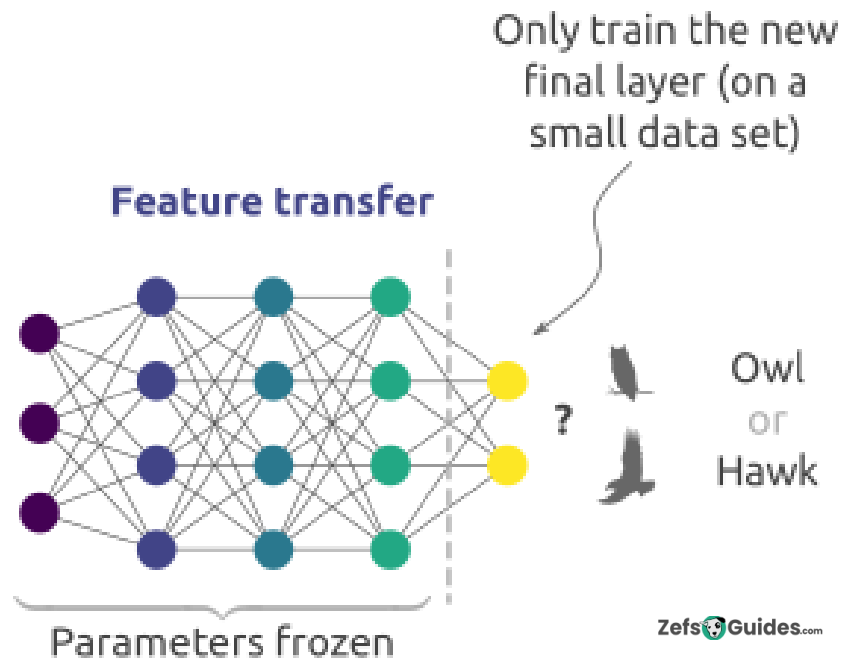


Zefsky Guides.com



# Transfer Learning

- You could freeze all but the FC layers and train.
- You could fine-tune the whole network.



# Transfer Learning

Generally easy to do:

- You could load from [HuggingFace](#).
- You could load from [TorchVision](#)
- Or even other [GitHubs](#)...

How to use from the Transformers library

```
# Use a pipeline as a high-level helper
from transformers import pipeline

pipe = pipeline("image-classification", model="microsoft/resnet-18")

# Load model directly
from transformers import AutoImageProcessor, AutoModelForImageClassification

processor = AutoImageProcessor.from_pretrained("microsoft/resnet-18")
model = AutoModelForImageClassification.from_pretrained("microsoft/resnet-18")
```

Quick Links

[Read model documentation](#)  
[Read docs on high-level-pipeline](#)  
[Read our learning resources](#)

PyTorch

Get Started Ecosystem Edge Blog Tutorials Docs Resources GitHub

<

MOBILENET V2

By Pytorch Team

Efficient networks optimized for speed and memory, with residual blocks

View on Github

Open on Google Colab

Open Model Demo

(a) Residual block

(b) Inverted residual block

```
import torch
model = torch.hub.load('pytorch/vision:v0.10.0', 'mobilenet_v2', pretrained=True)
model.eval()
```

# Data Augmentation

- As mentioned earlier, lack of large amounts of data is a problem.
  - Model may overfit (learn “spurious” features).
  - Model may not generalize well to “out-of-distribution” data.

# Data Augmentation

- What is a lion exactly?
- To increase the amount of data and add make sure the learned features are diverse...
  - introduce as much valid variations as possible to the dataset.

