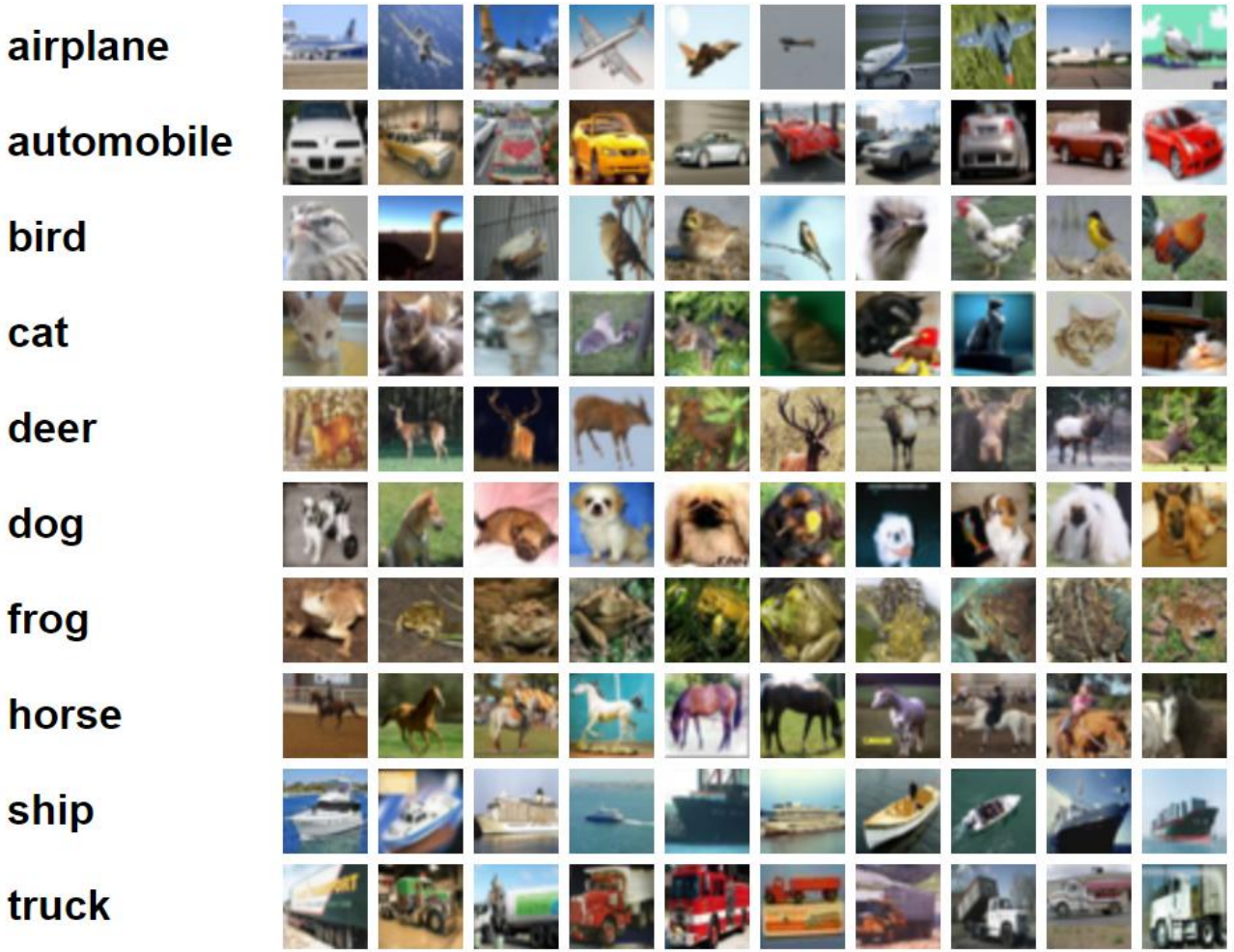
RBE474X/595-B01-ST: Deep Learning For Perception

Class 4: Advanced CNN Architectures And Image Warping

Prof. Wei Xiao


HW1: Nifty Neural Networks!
Due on: Nov 14, 2025 at 11:59:59 PM
(Group Submissions)

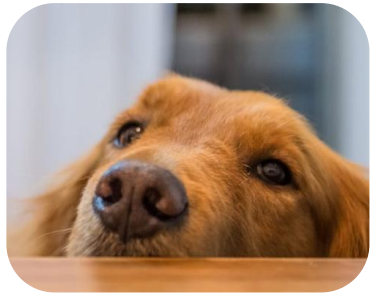
DIY Multi-Class Classifier

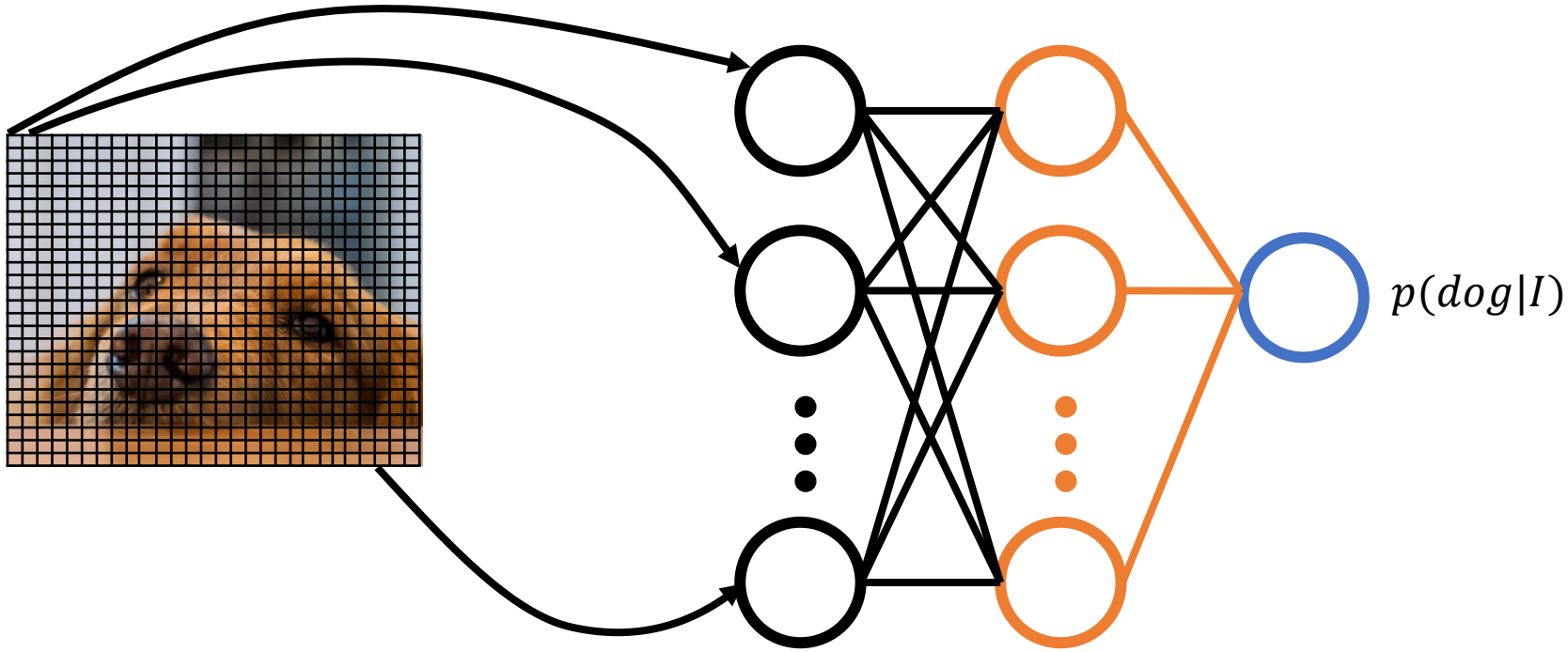


Magical Classifier

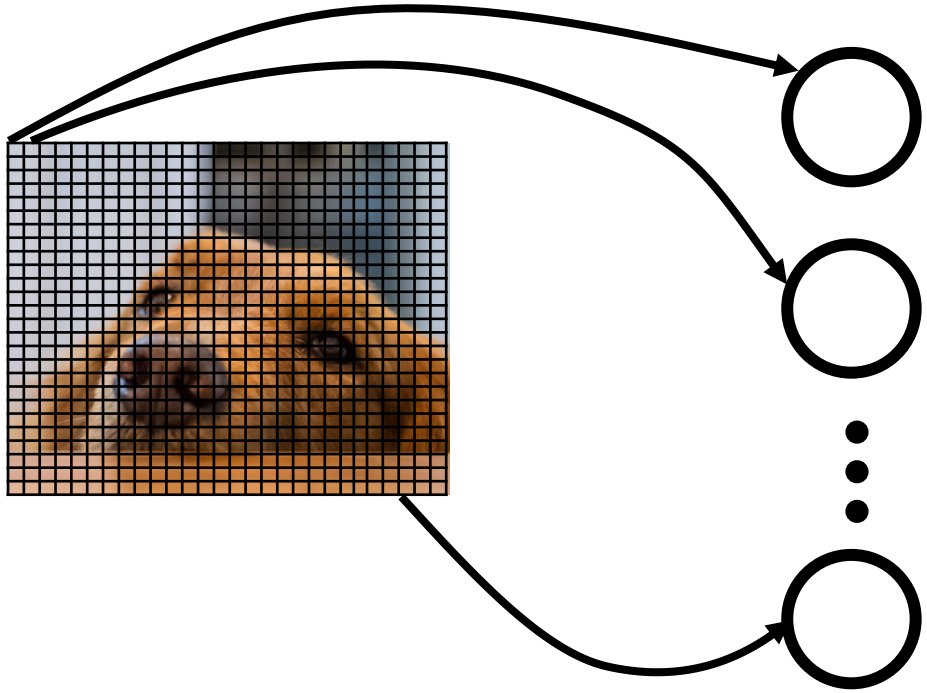
$p(\text{class}|I)$

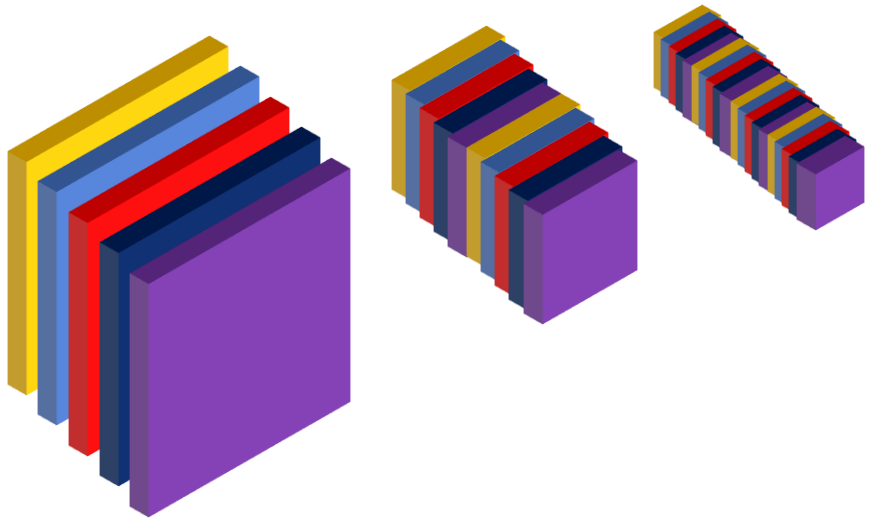
Let's Recap





Let’s Recap





Let’s Talk About CNN Architecture!

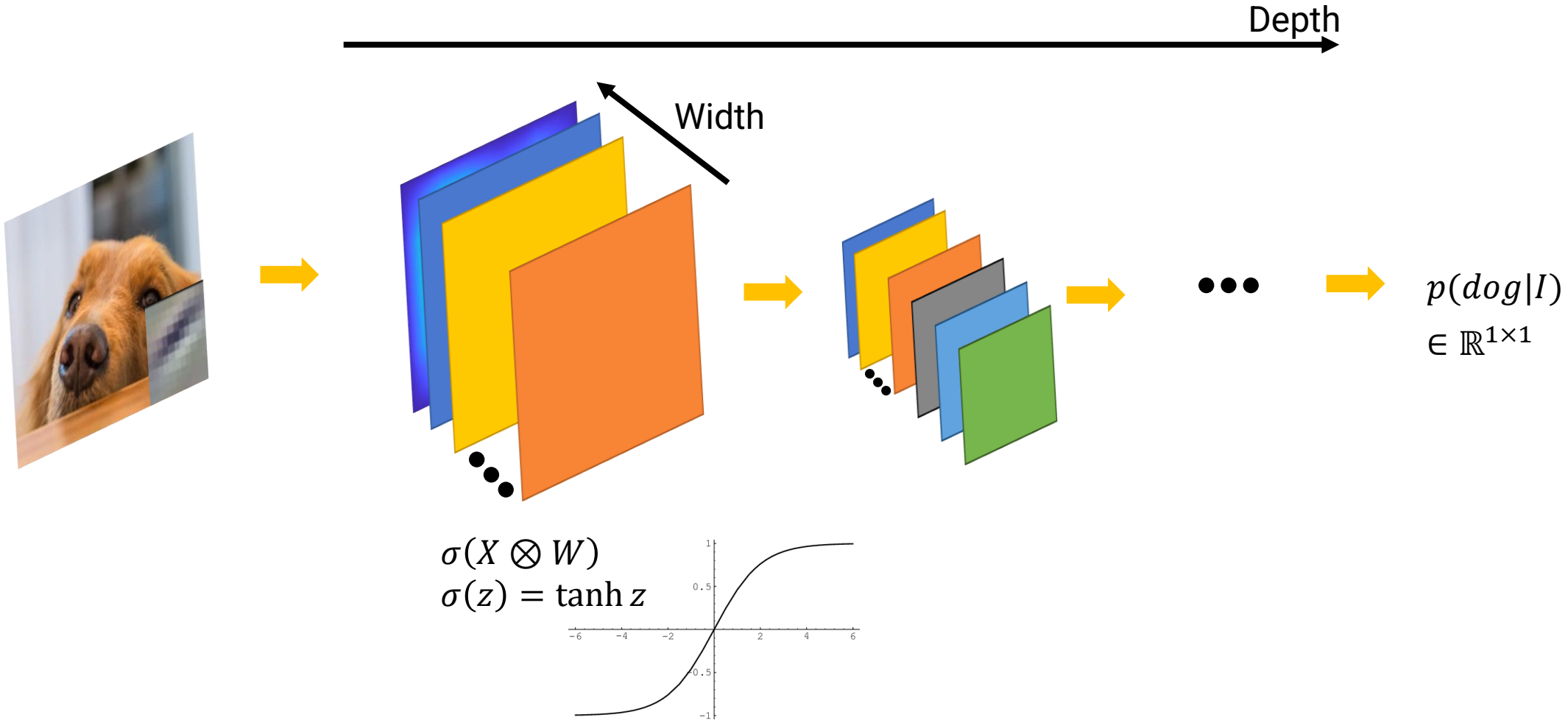
Let’s Start Simple

Let’s Start Simple

Let’s Start Simple

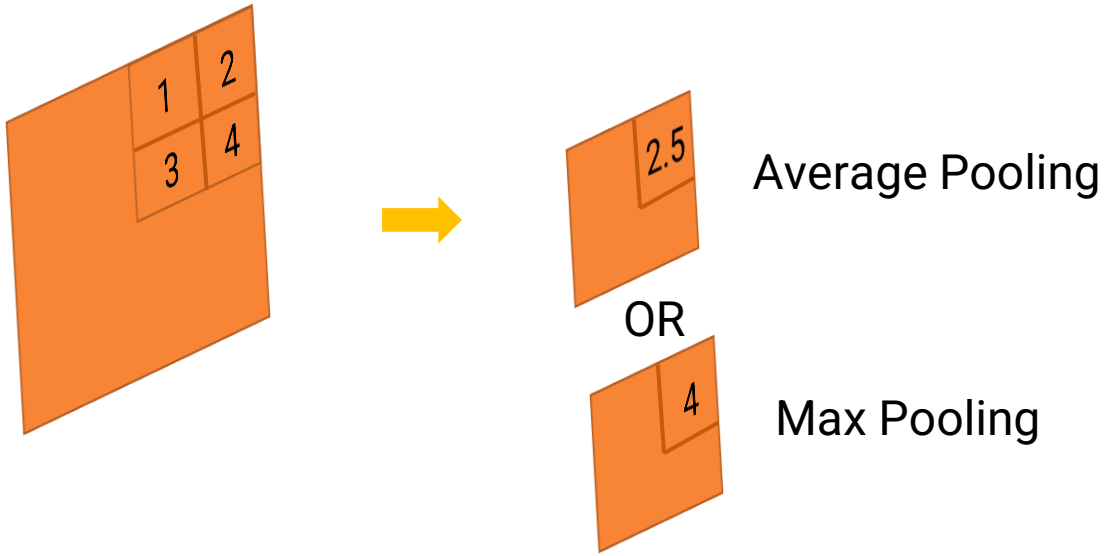
Let’s Start Simple

Let's Start Simple



Reduce Dimension/Resolution

How?



Can also do **Strided Convolution**

A Simple Architecture

Max-pooling

$\times N$ times

Dense (MLP)

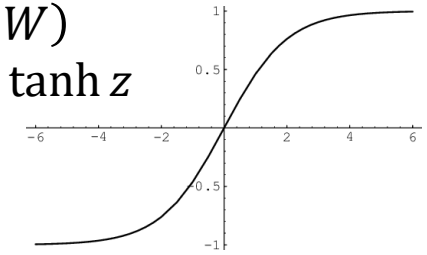
Flatten

$p(dog|I)$

$\in \mathbb{R}^{1\times 1}$

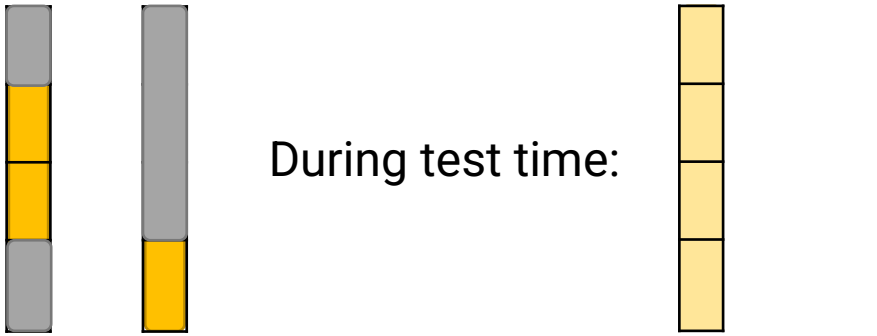
$\sigma(X \otimes W)$

$\sigma(z) = \tanh z$



“Rule Of Thumb”

- Drop filter size a factor of N (generally 2) every layer
- Increase number of neurons by a factor of M (generally 2) every layer
- Drop filter sizes as layer number decreases
  - Start at 7×7, then go to 5×5 and finally 3×3 (then maintain constant)
- Have as few fully connected or dense layers as possible to maintain small number of parameters
- Use **Dropout** to avoid overfitting
- If only convolutional layers are used: Fully Convolutional Network (FCN)
  - Can take **any input size**



Dropout = 0.5 0.75 Scale accordingly!

Story Time!

Turning Award in 2018! **A cool two decades later!**

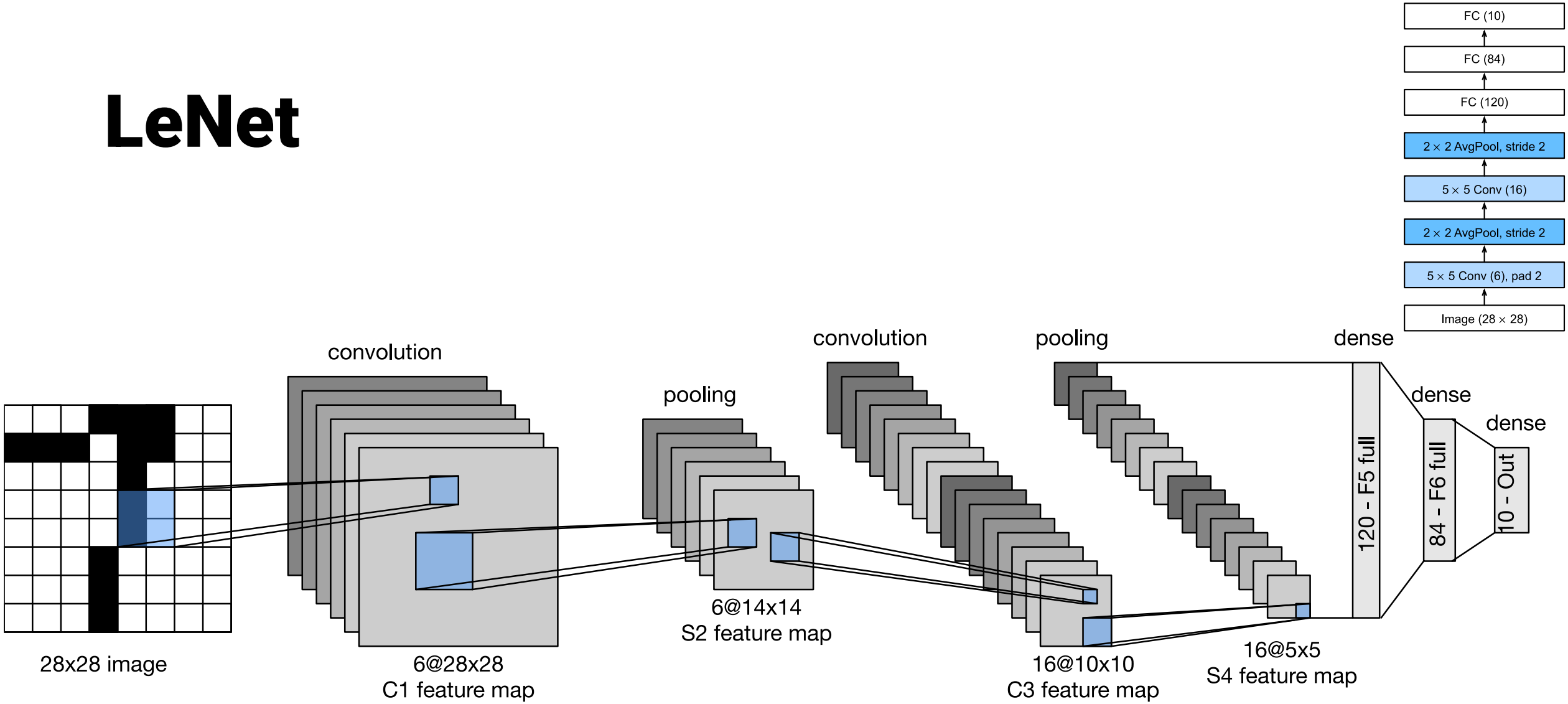
MNIST

Modified National Institute of Standards and Technology

For automatically characterizing zipcodes!

LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.

LeNet

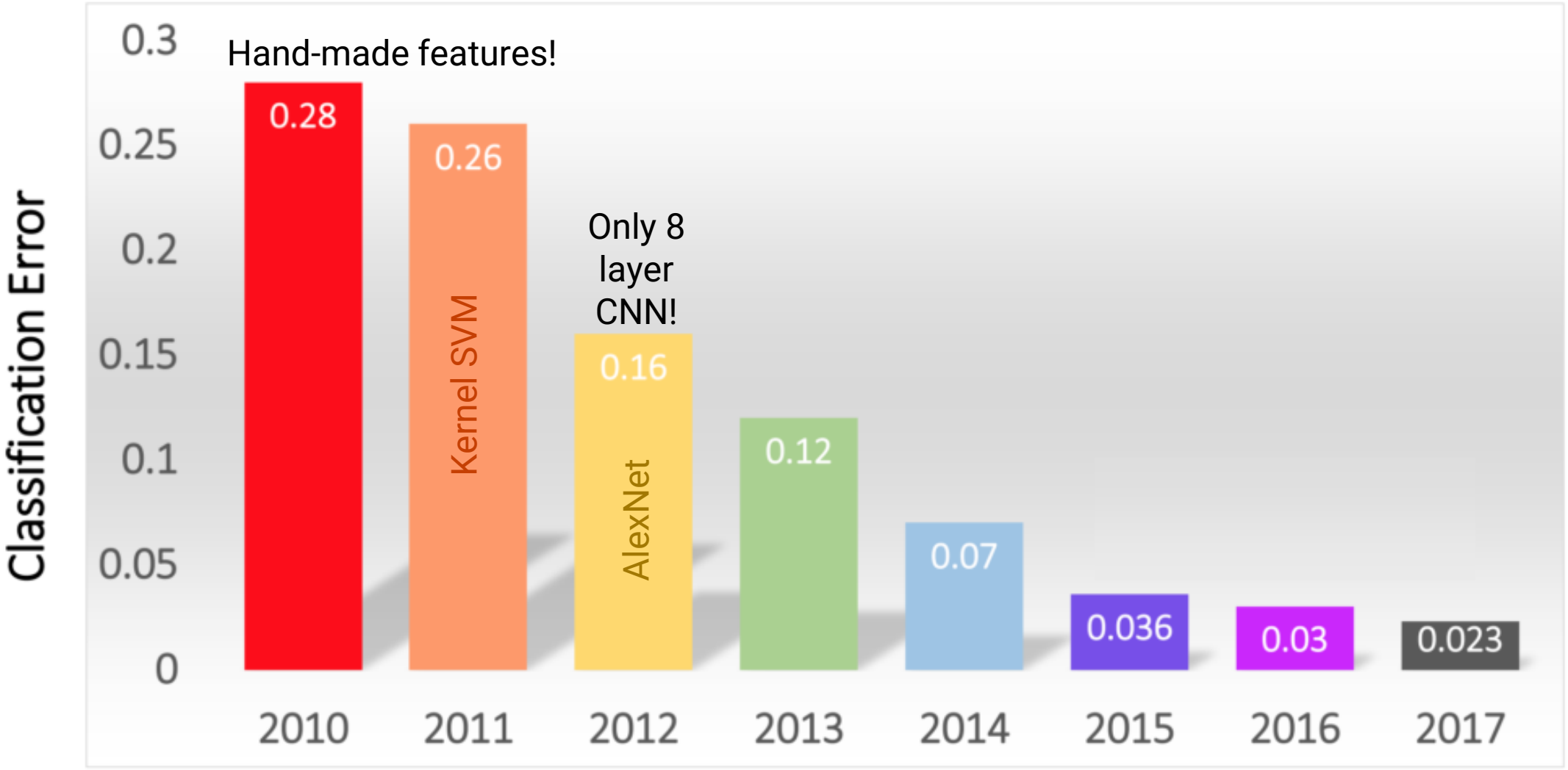

FC (10)
FC (84)
FC (120)
2 × 2 AvgPool, stride 2
5 × 5 Conv (16)
2 × 2 AvgPool, stride 2
5 × 5 Conv (6), pad 2
Image (28 × 28)
convolution
pooling
convolution
pooling
dense
dense
dense
120 - F5 full
84 - F6 full
10 - Out
28x28 image
6@28x28
C1 feature map
6@14x14
S2 feature map
16@10x10
C3 feature map
16@5x5
S4 feature map

ILSVRC Challenge

ImageNet Large Scale Visual Recognition Challenge

- Was built-on PASCAL VOC Challenge
- 20000 categories!
- 14 Million Images!
- Challenge uses 1000 classes over 1M images!
  - Usually 224×224 px. resolution

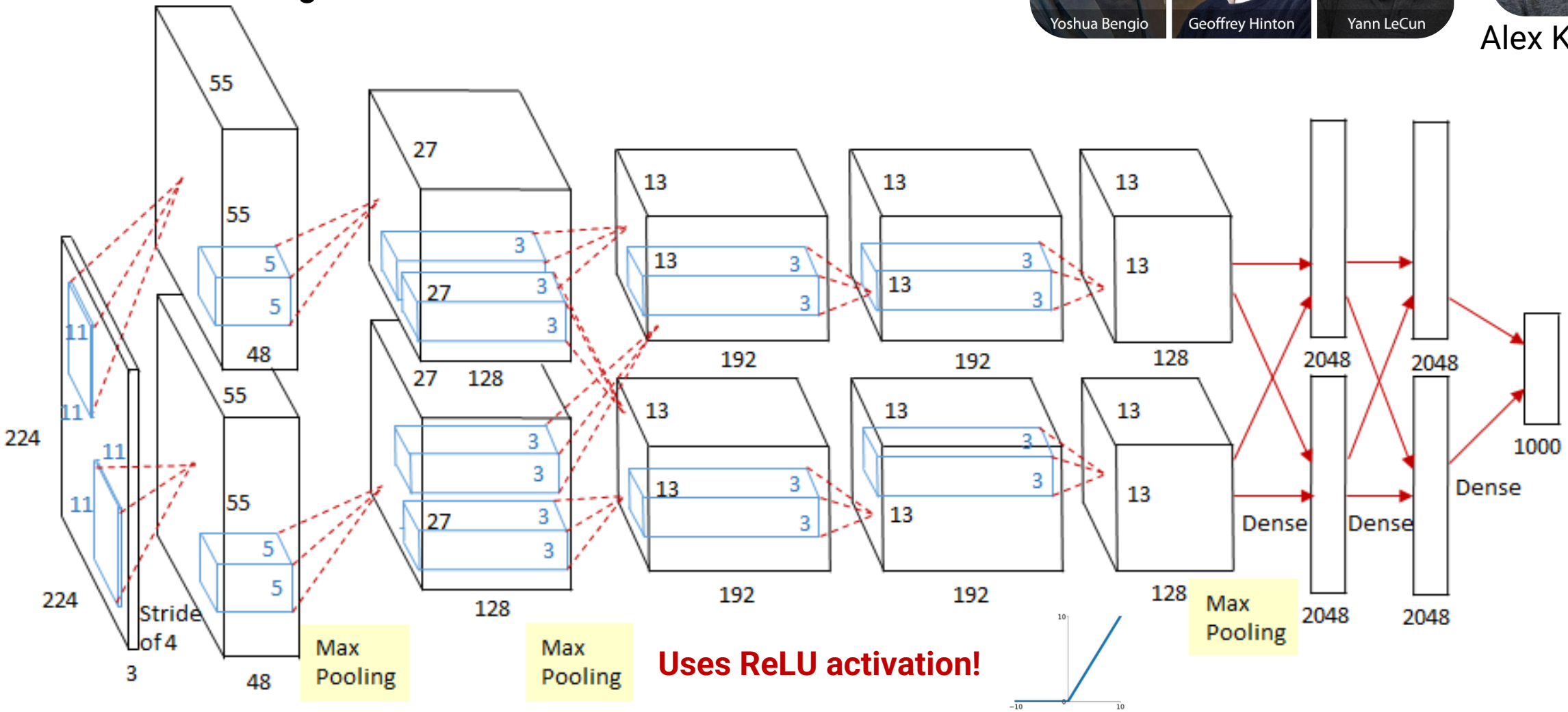
ILSVRC Winners



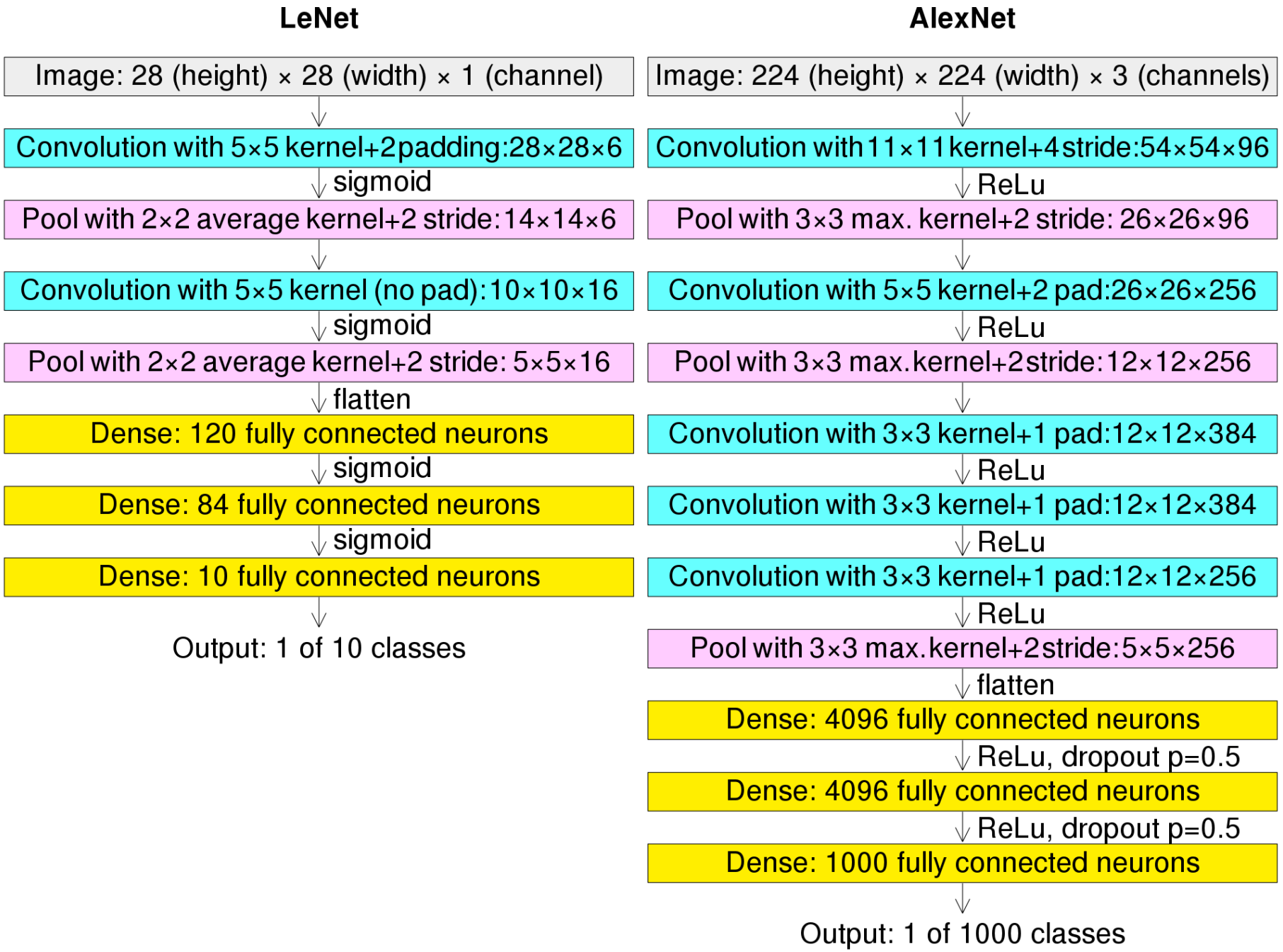
AlexNet

Trained using 2 NVIDIA GTX 580 3GB GPU

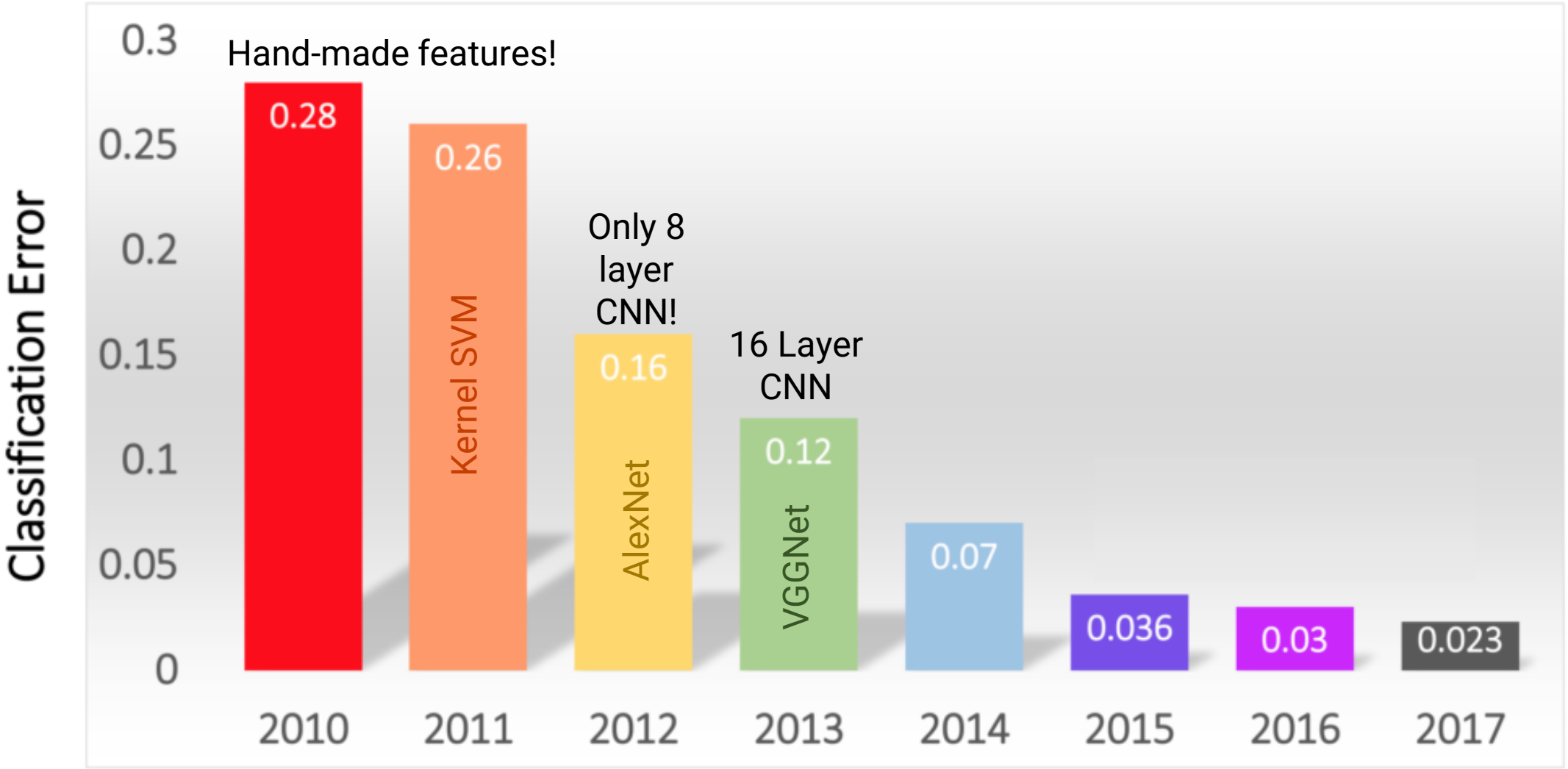
Alex Krizhevsky



Recap


LeNet
Image: 28 (height) × 28 (width) × 1 (channel)
Convolution with 5×5 kernel+2padding:28×28×6
sigmoid
Pool with 2×2 average kernel+2 stride:14×14×6
Convolution with 5×5 kernel (no pad):10×10×16
sigmoid
Pool with 2×2 average kernel+2 stride: 5×5×16
flatten
Dense: 120 fully connected neurons
sigmoid
Dense: 84 fully connected neurons
sigmoid
Dense: 10 fully connected neurons
Output: 1 of 10 classes
AlexNet
Image: 224 (height) × 224 (width) × 3 (channels)
Convolution with 11×11 kernel+4 stride:54×54×96
ReLu
Pool with 3×3 max. kernel+2 stride: 26×26×96
Convolution with 5×5 kernel+2 pad:26×26×256
ReLu
Pool with 3×3 max.kernel+2stride:12×12×256
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×256
ReLu
Pool with 3×3 max.kernel+2stride:5×5×256
flatten
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 1000 fully connected neurons
Output: 1 of 1000 classes

ILSVRC Winners



VGGNet

Visual Geometry Group, Oxford

Andrew Zisserman

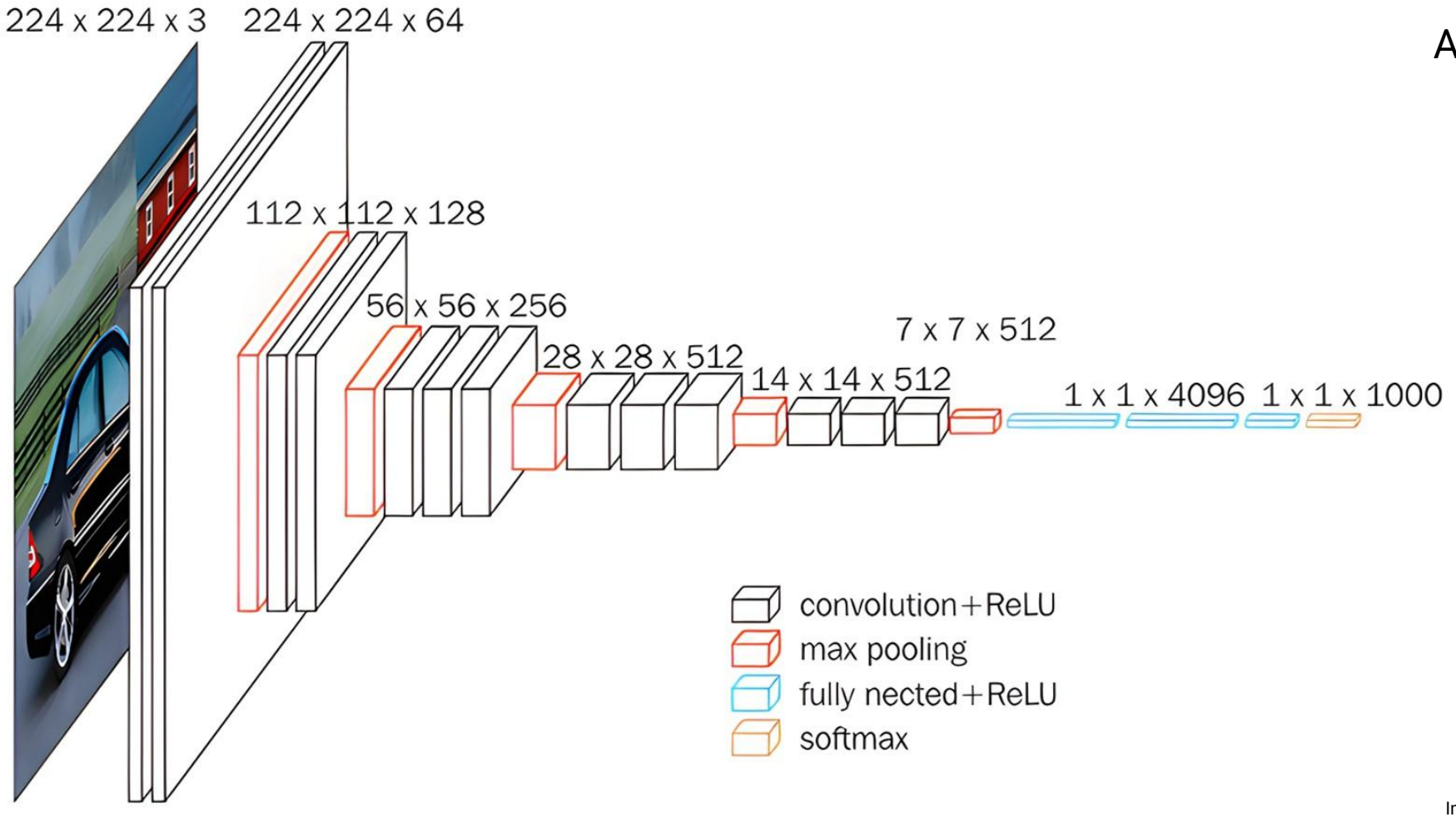
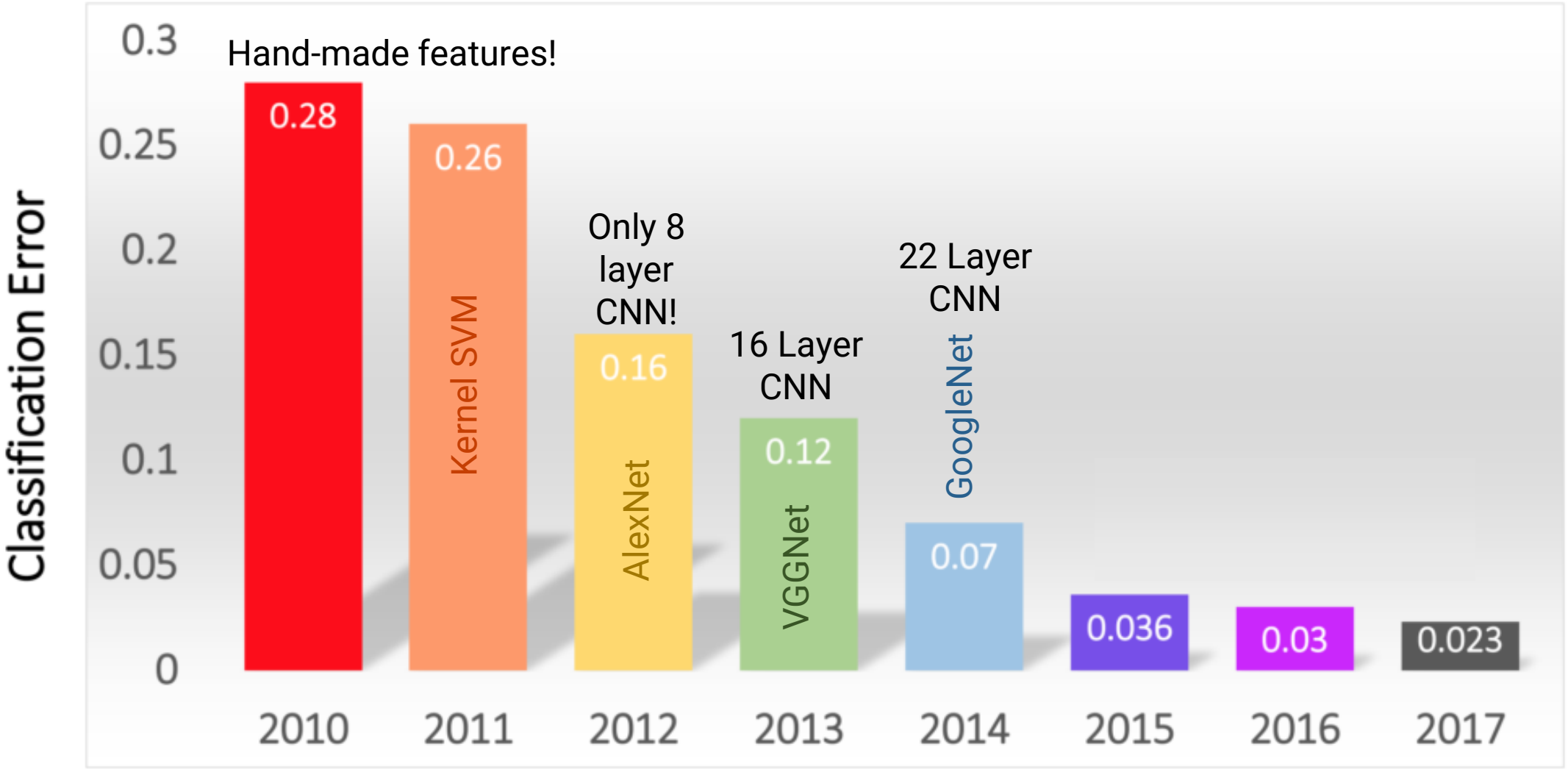
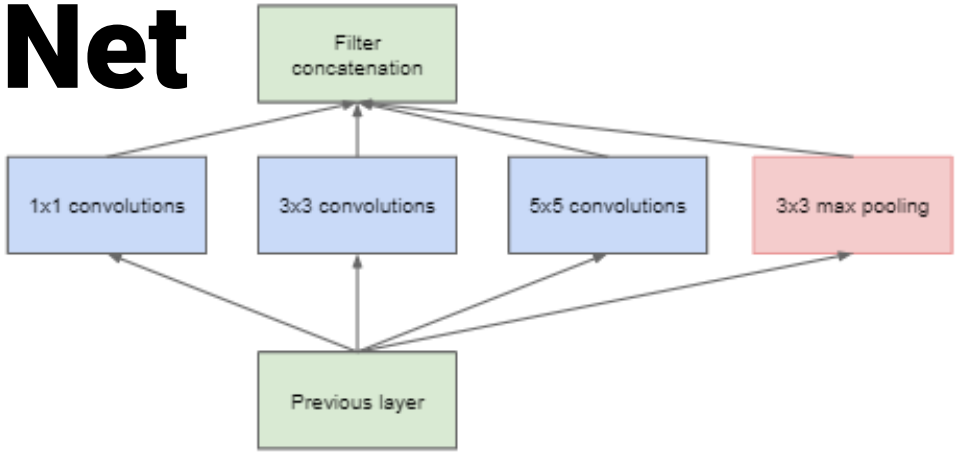


Image resolution enhanced using DeepAI

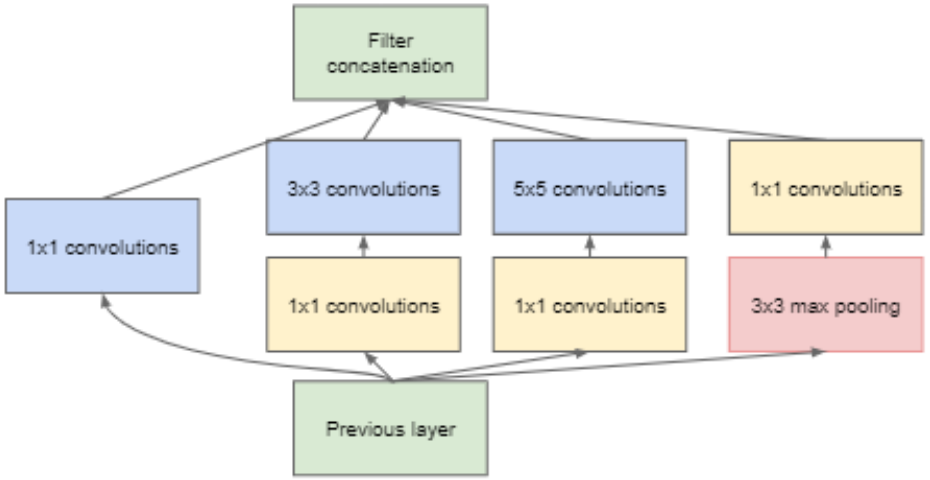
ILSVRC Winners



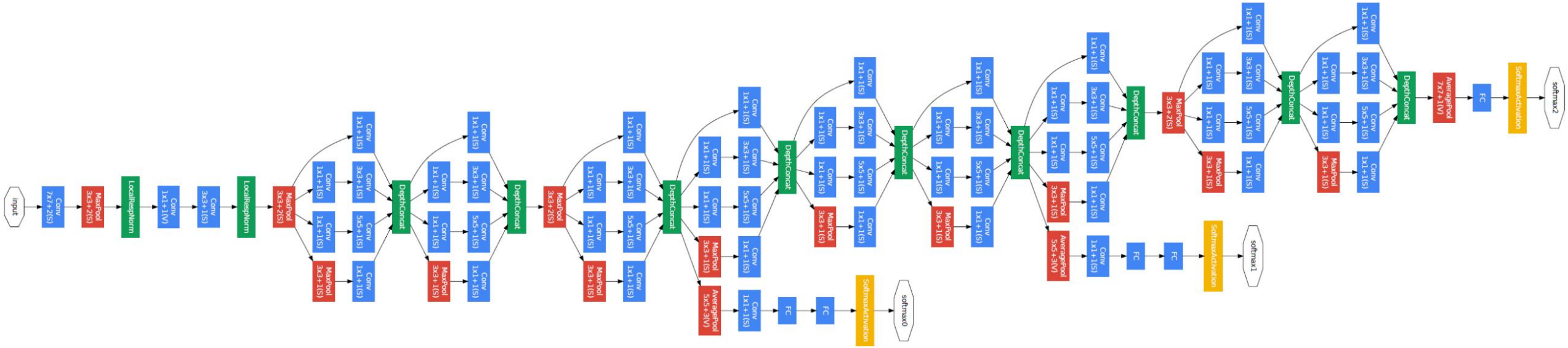
GoogleNet



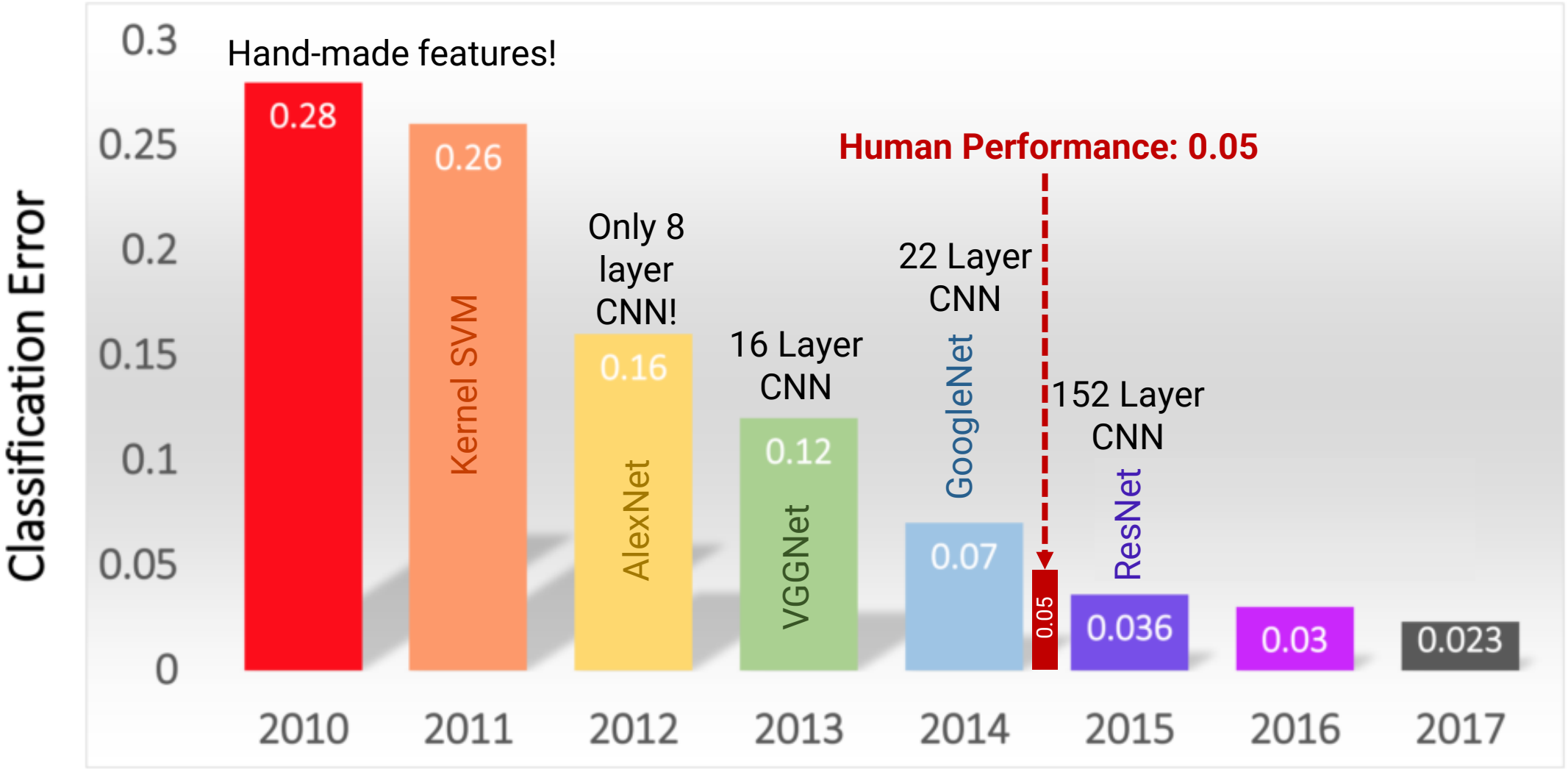
(a) Inception module, naïve version



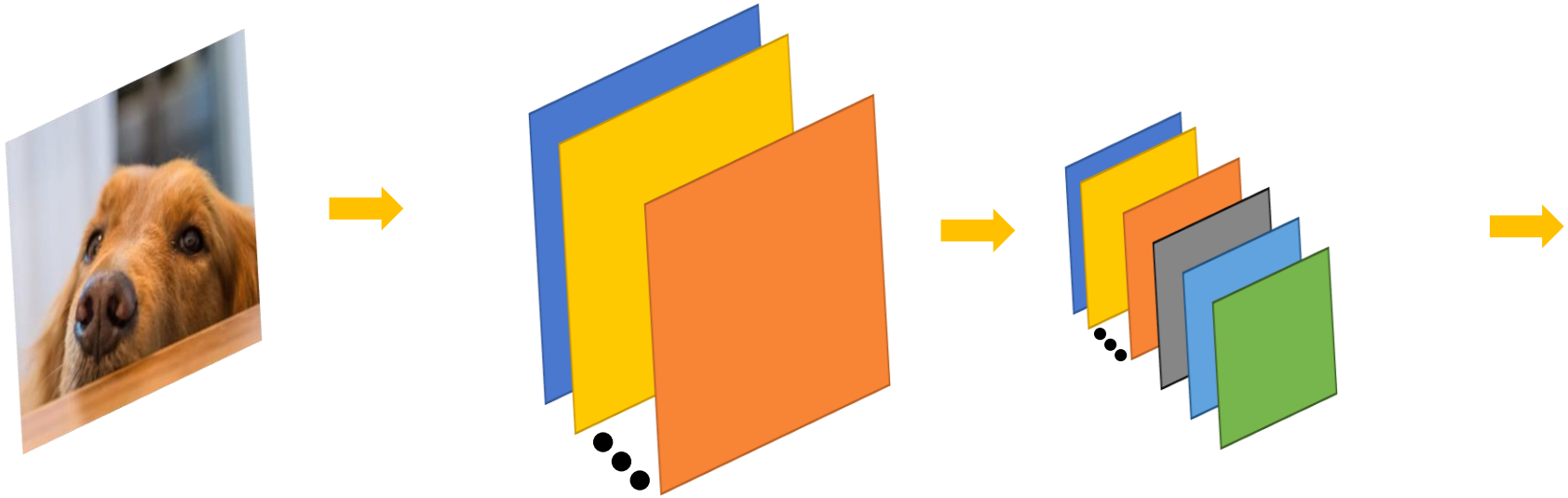
(b) Inception module with dimension reductions



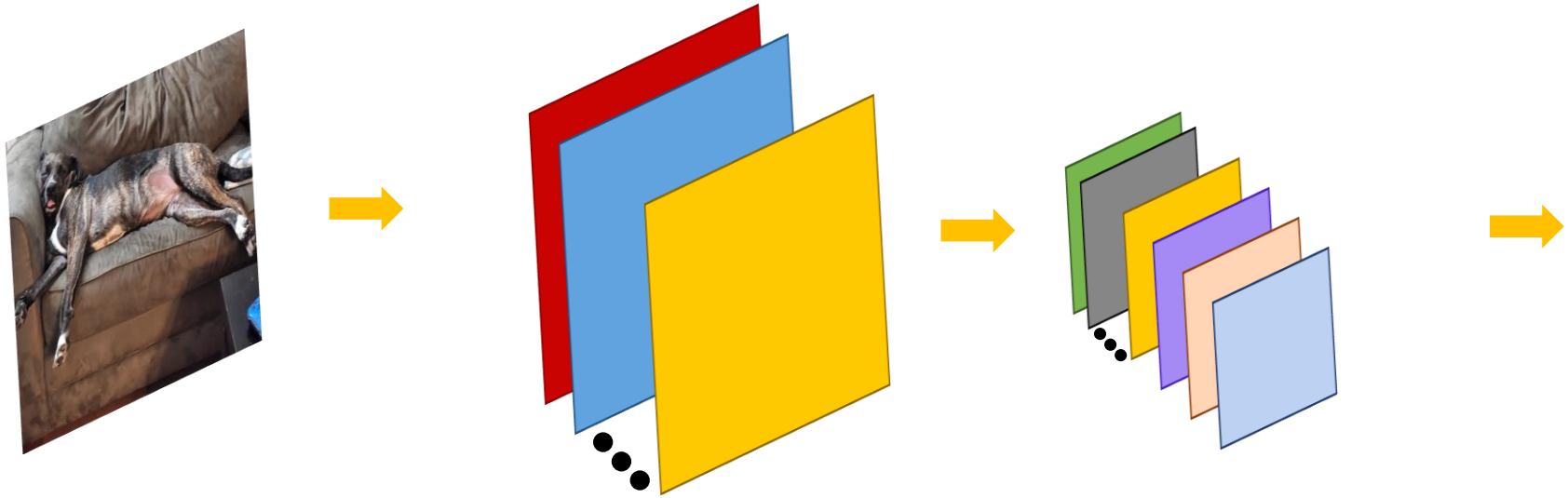
ILSVRC Winners



Sidestep: Batch Normalization

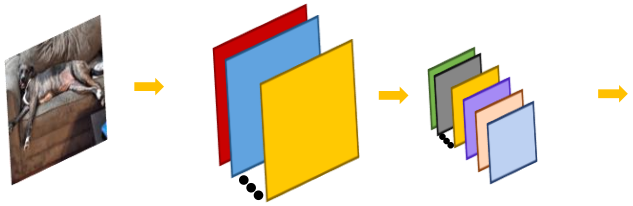


Sidestep: Batch Normalization



Activations shift from image to image and batch to batch: Covariance shift

Sidestep: Batch Normalization



Activations shift from image to image and batch to batch: Covariance shift

Standardize inputs (also called **whitening**) by subtracting mean and dividing by covariance!

Perform standardization over every feature map in every batch!

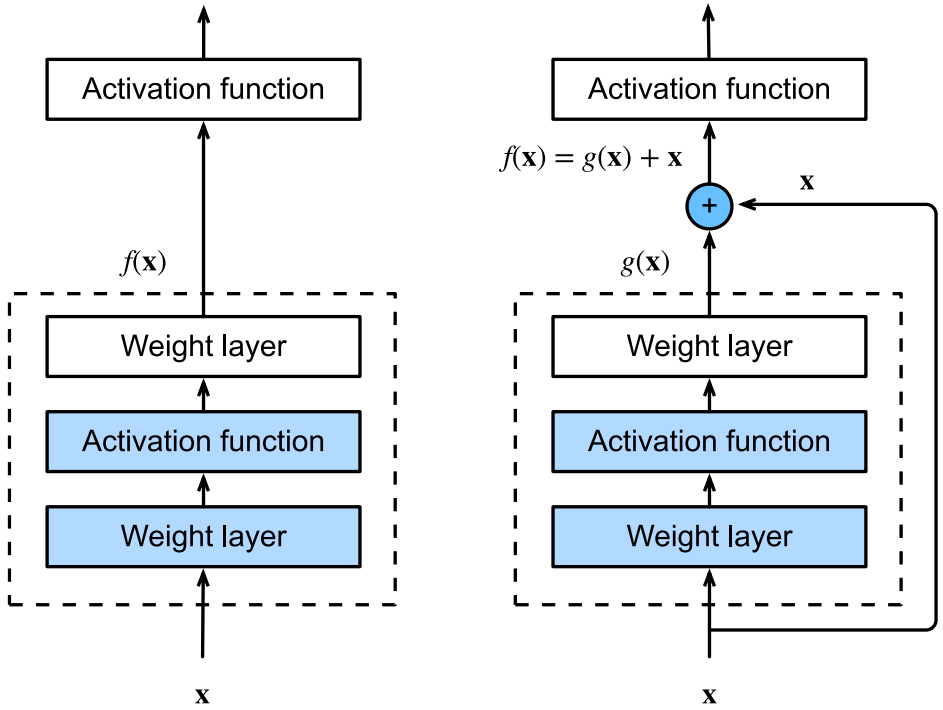
$$\gamma \frac{z - \mu}{\sigma} + \eta$$

Mean of feature map over training batch
Standard deviation over training batch
Training parameters for scaling and shifting

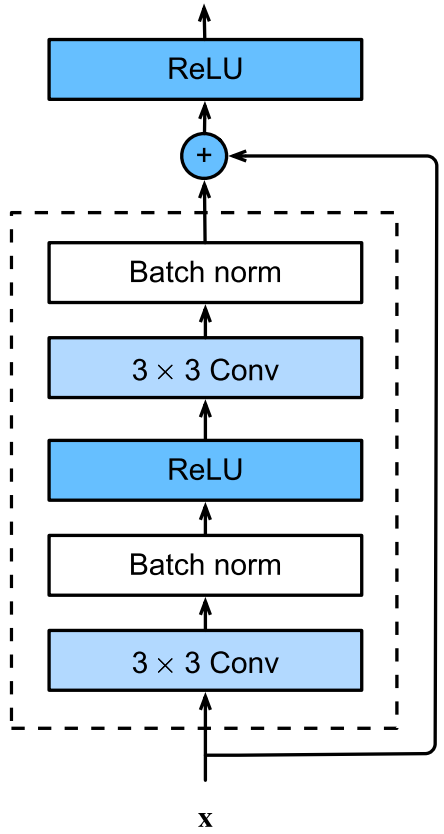
Debate as to whether if actually helps with covariance shift!
However, does speed up training!

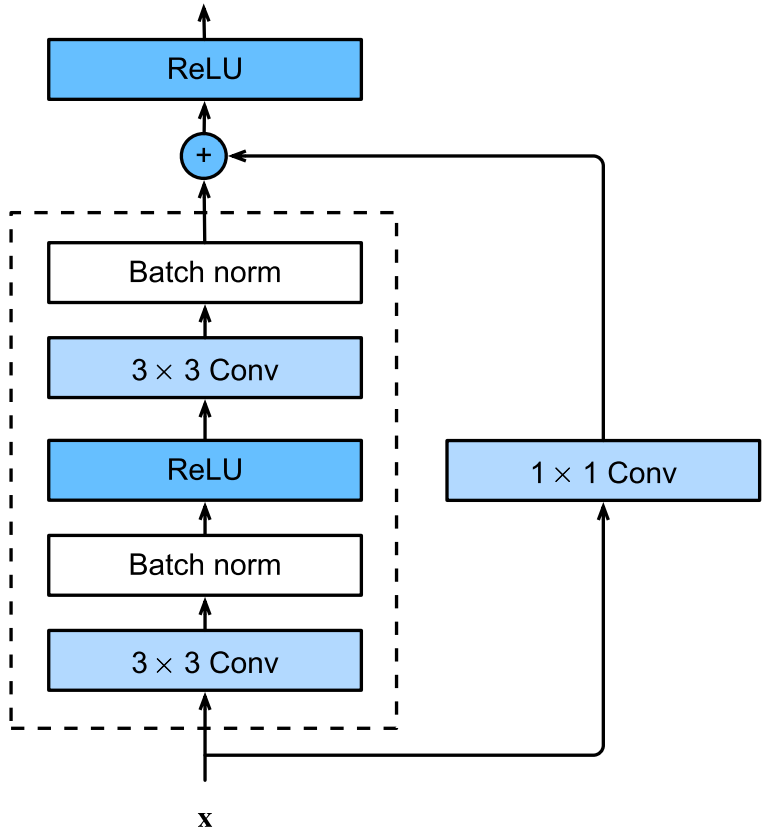
Santurkar, Shibani, et al. "How does batch normalization help optimization?." *Advances in neural information processing systems* 31 (2018).
Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International conference on machine learning*. PMLR, 2015.

Back To Main Story: ResNet



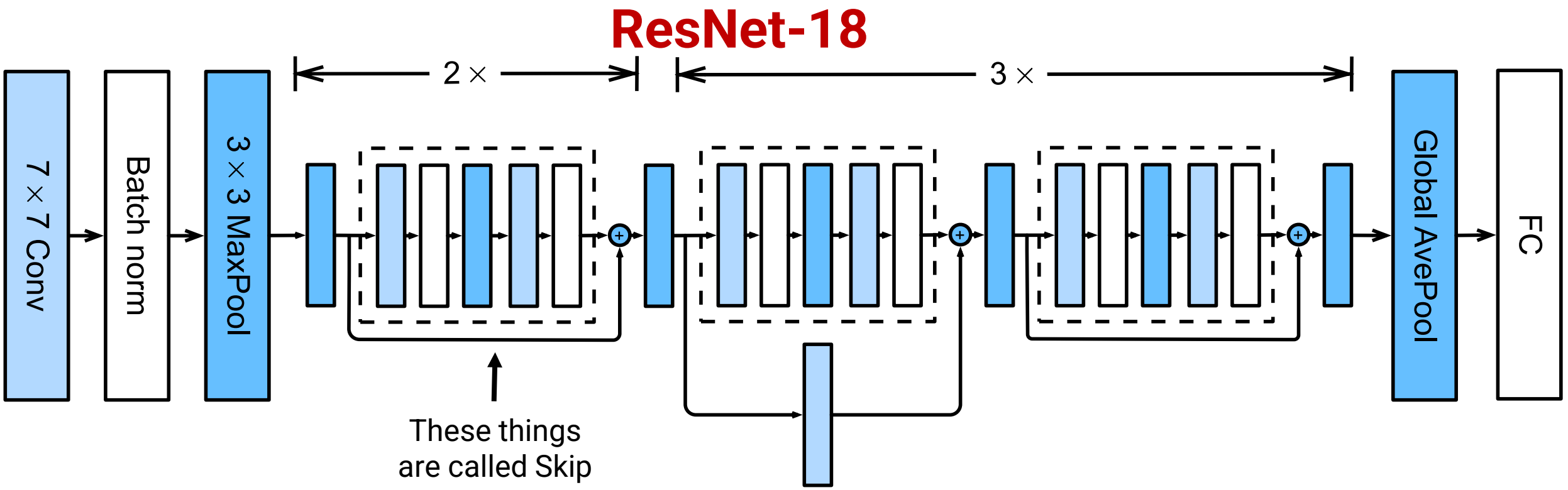
Helps with Vanishing Gradients





If dimension needs to be changed

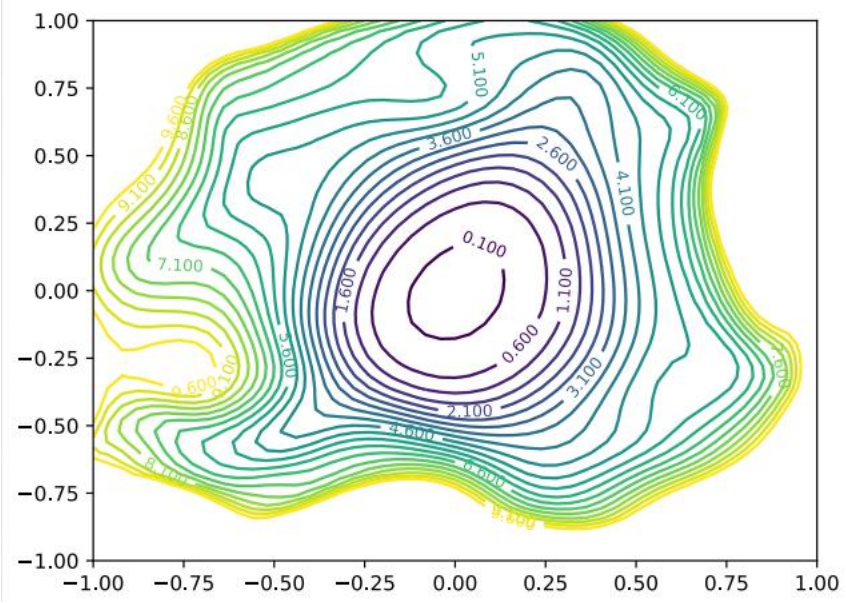
ResNet



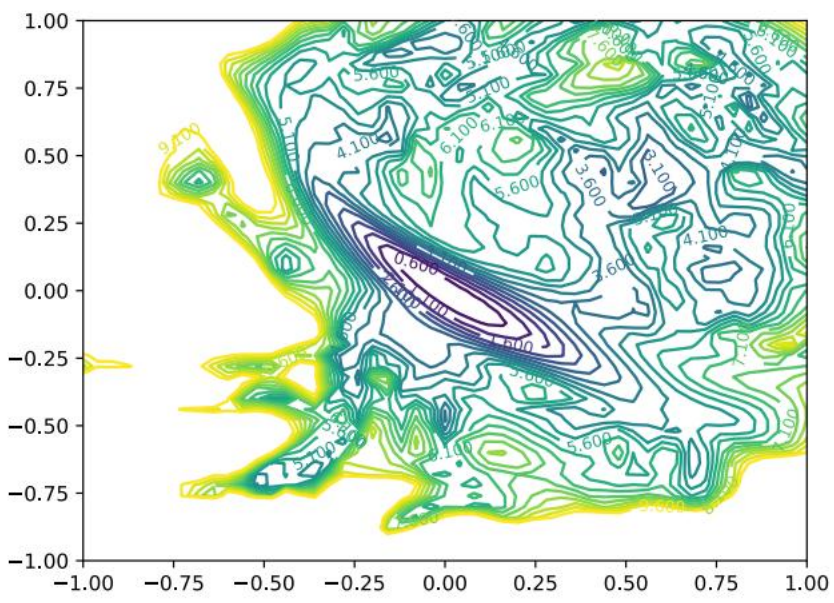
8.6. Residual Networks (ResNet) and ResNeXt — Dive into Deep Learning 1.0.0-alpha1.post0 documentation (d2l.ai)

Why Skip Connections?

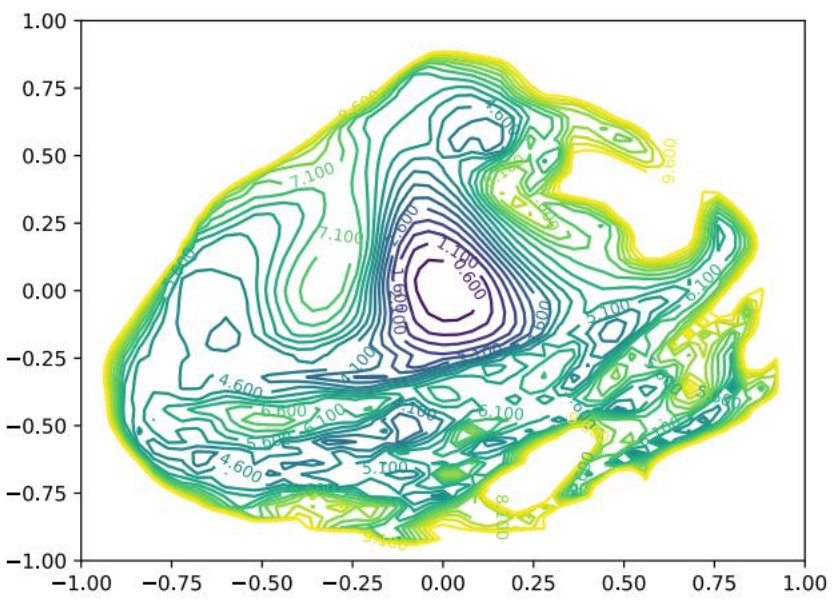
VGG-20



VGG-56



VGG-110



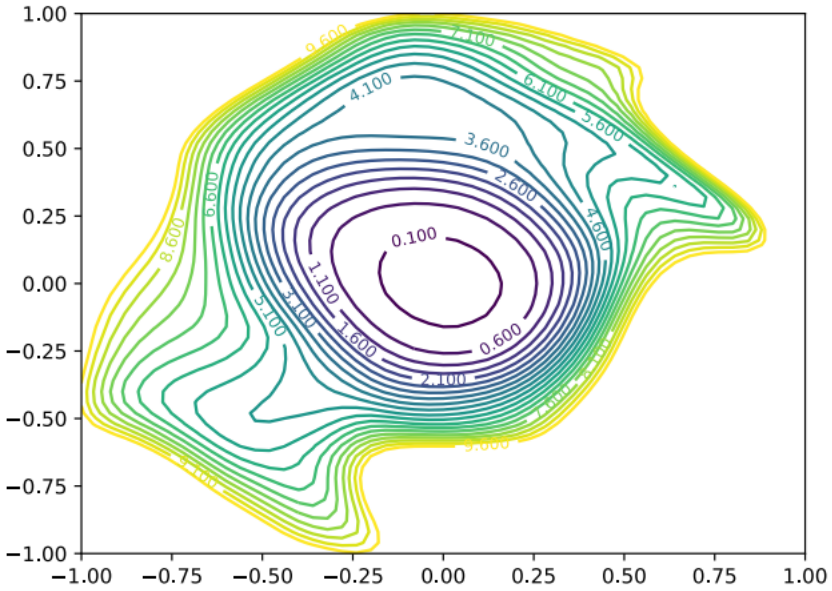
Convexity (Smoothness) → Chaos

Slide adapted from UMD's CMSC764

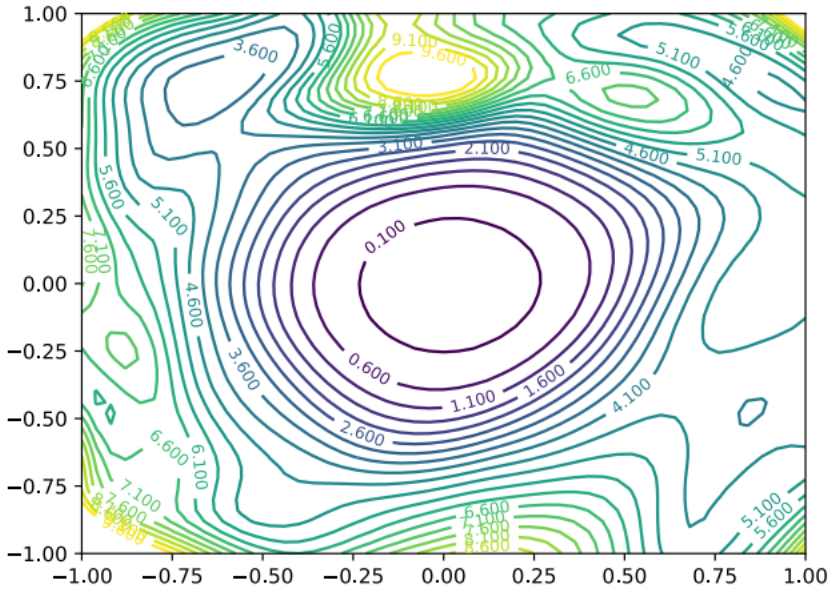
Li, Hao, et al. "Visualizing the loss landscape of neural nets." Advances in neural information processing systems 31 (2018).

Why Skip Connections?

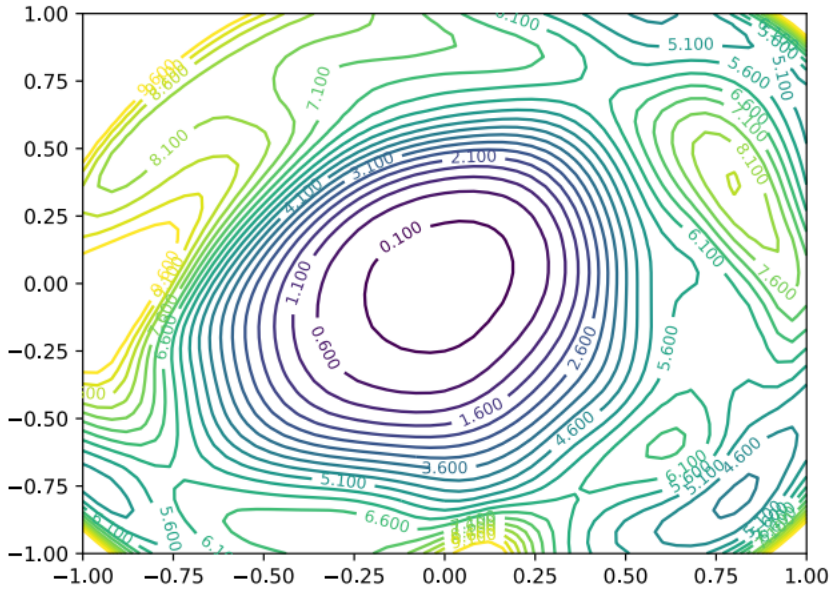
ResNet-20



ResNet-56

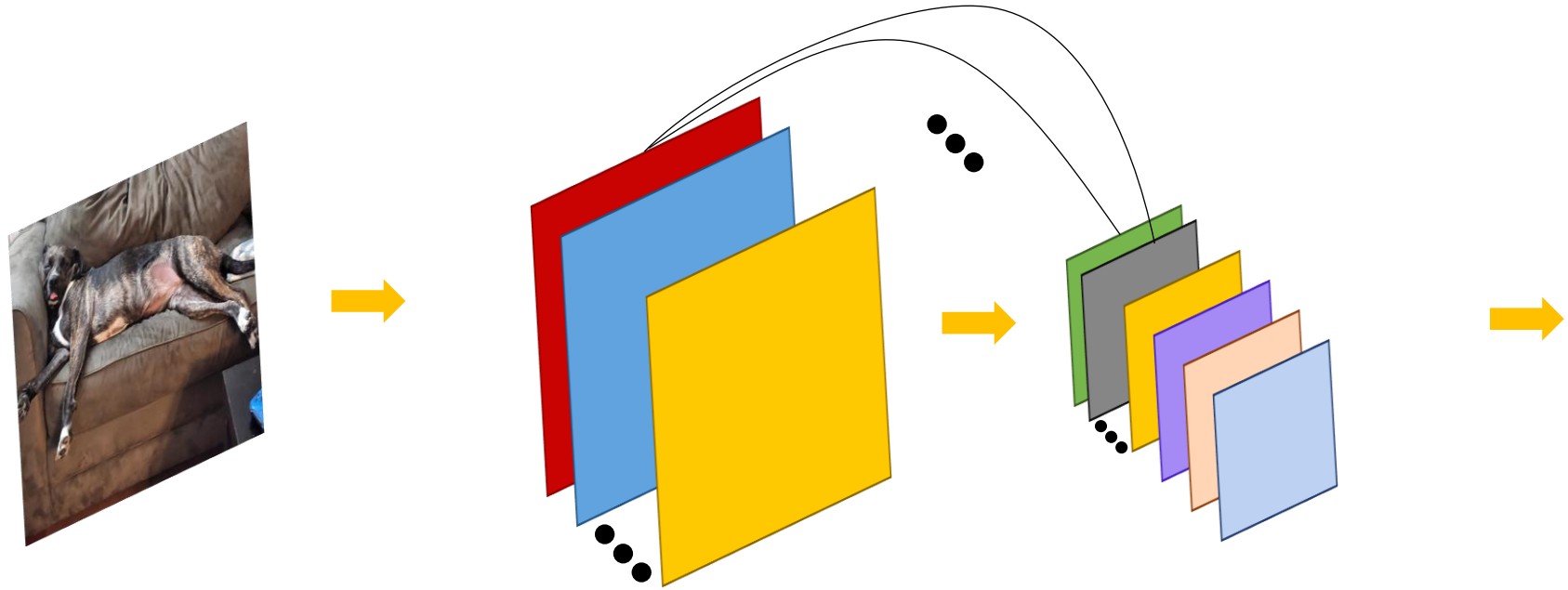


ResNet-110

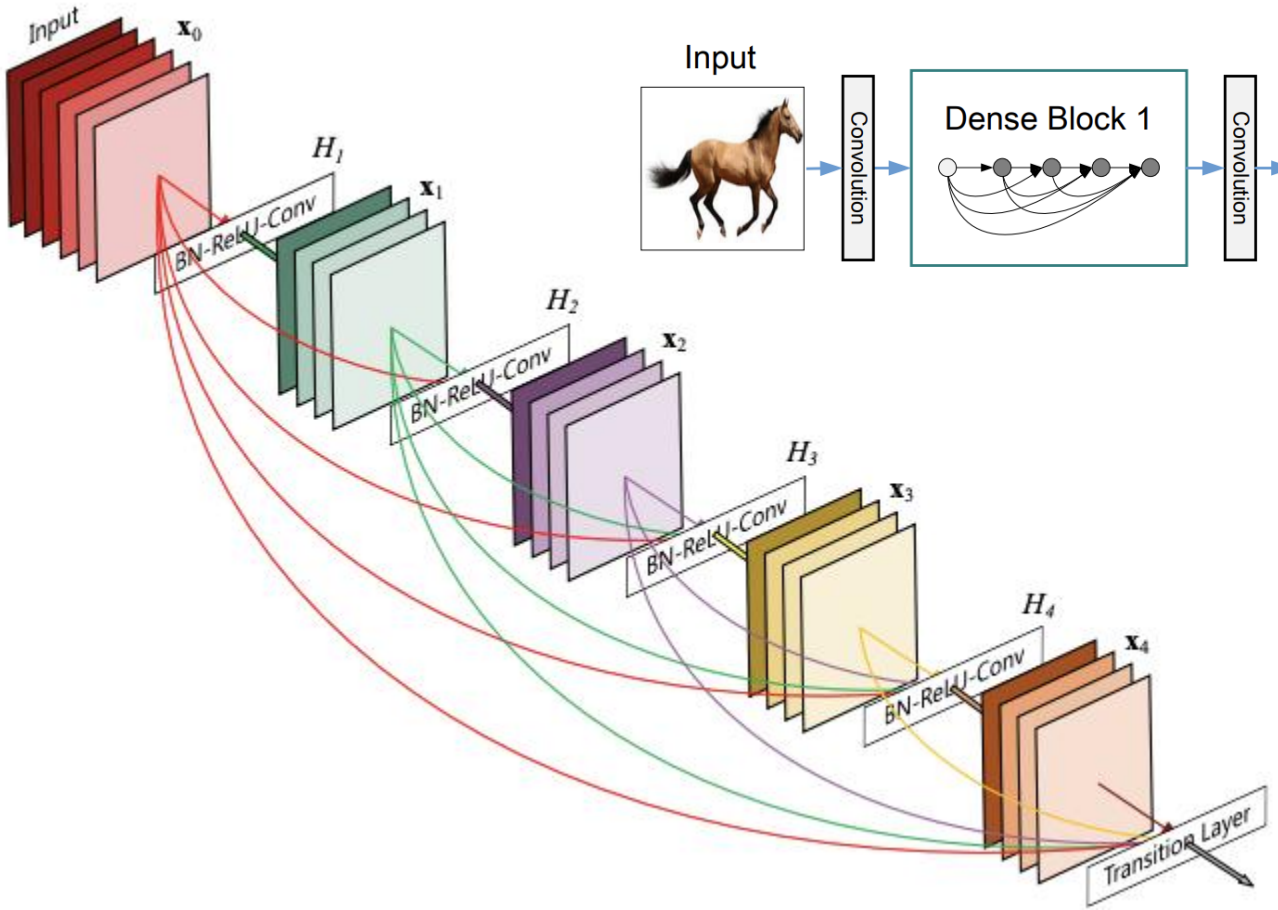


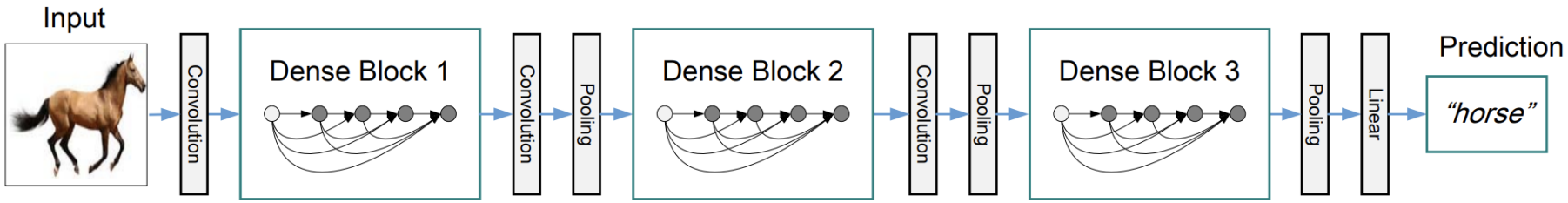
Convexity (Smoothness) → Chaos

How Can I Have Max. Skip Connections?



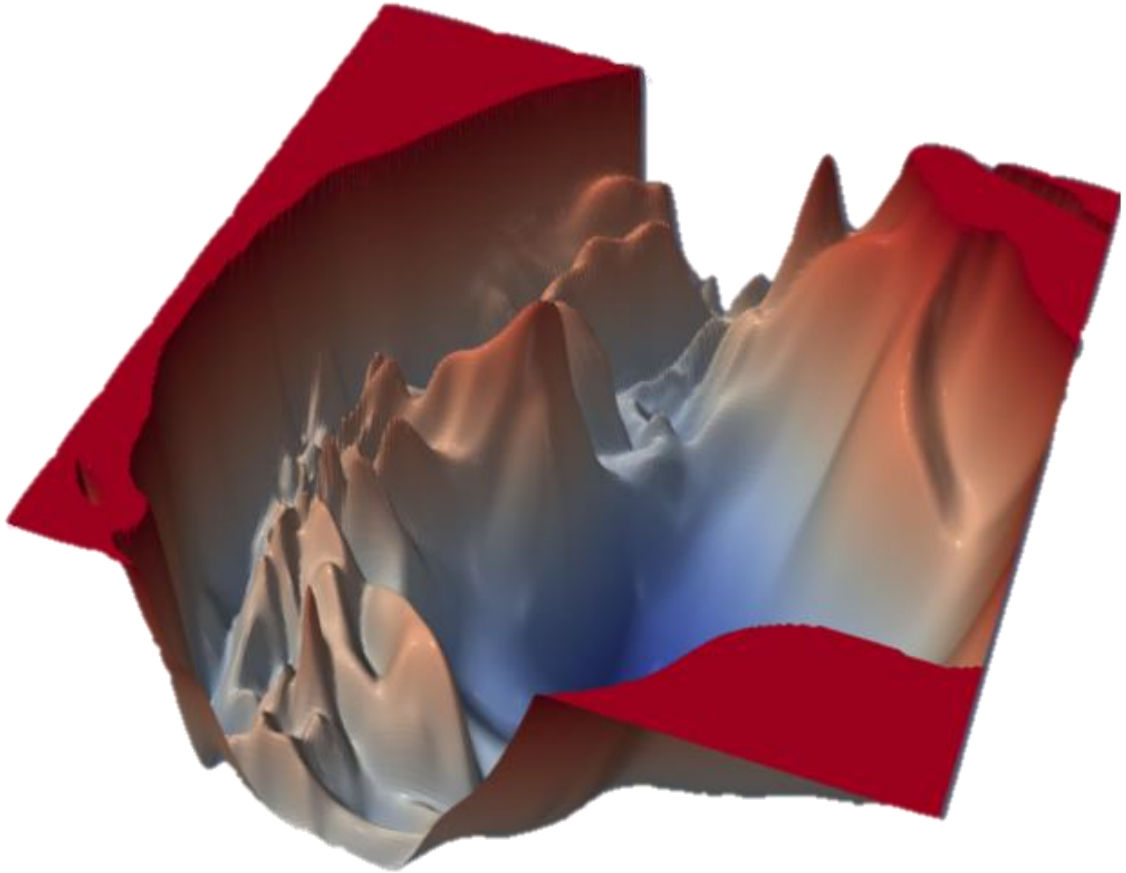
DenseNet



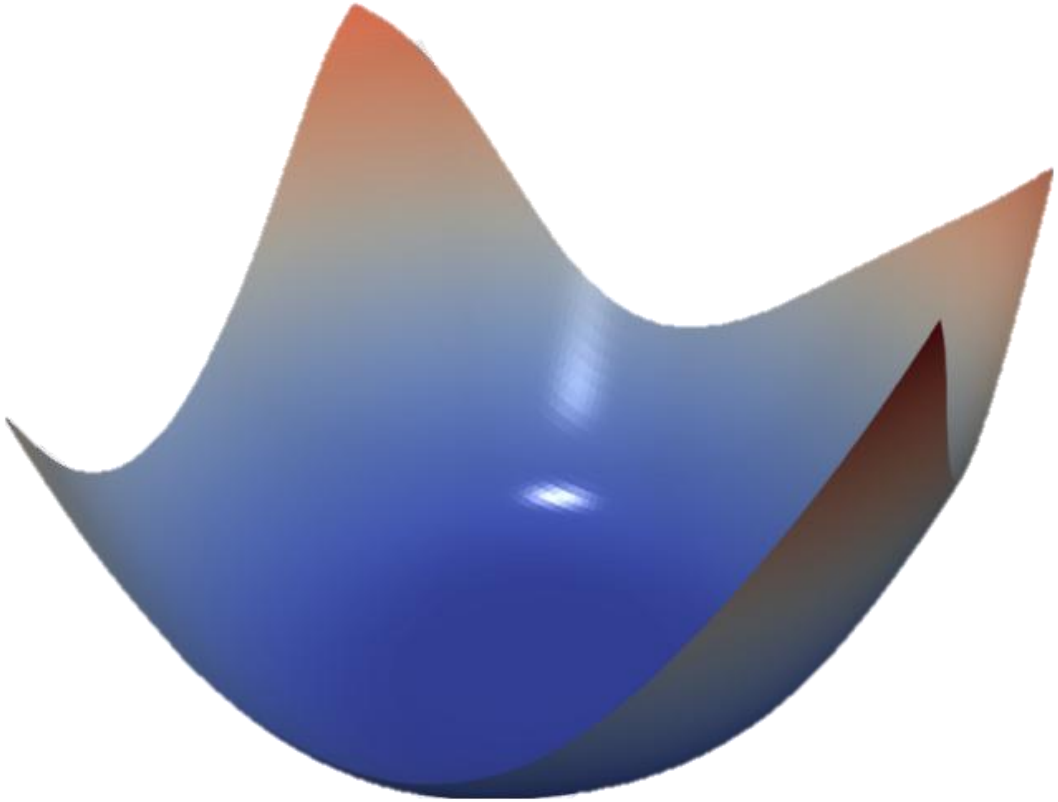


Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

Skip Connections Are Amazing!



VGG-110

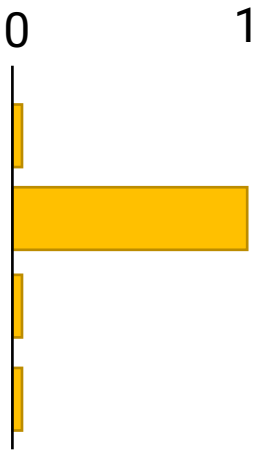


DenseNet 121 Layers

Loss Functions

- y is a class label such as “dog” or “cat”
- For multi-class case, can be many names and hence can be encoded as numbers
- But even better way is One-Hot Encoding!
- Sometimes people use “out of dataset” class as well!

“2” becomes



For 4 classes

- Raw output of network when values are $\mathbb{R}^{N\times 1}$ are called Logits!
- Logits can be converted to probabilities using softmax
- Be careful of what loss function you are using

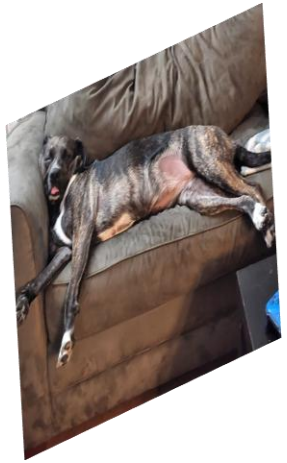
Some common loss functions (for classification) are:

- Cross entropy
- MSE
- Hinge Loss
- KL Divergence Loss

This will be covered in supplementary videos!

How to Choose Loss Functions When Training Deep Learning Neural Networks (machinelearningmastery.com)

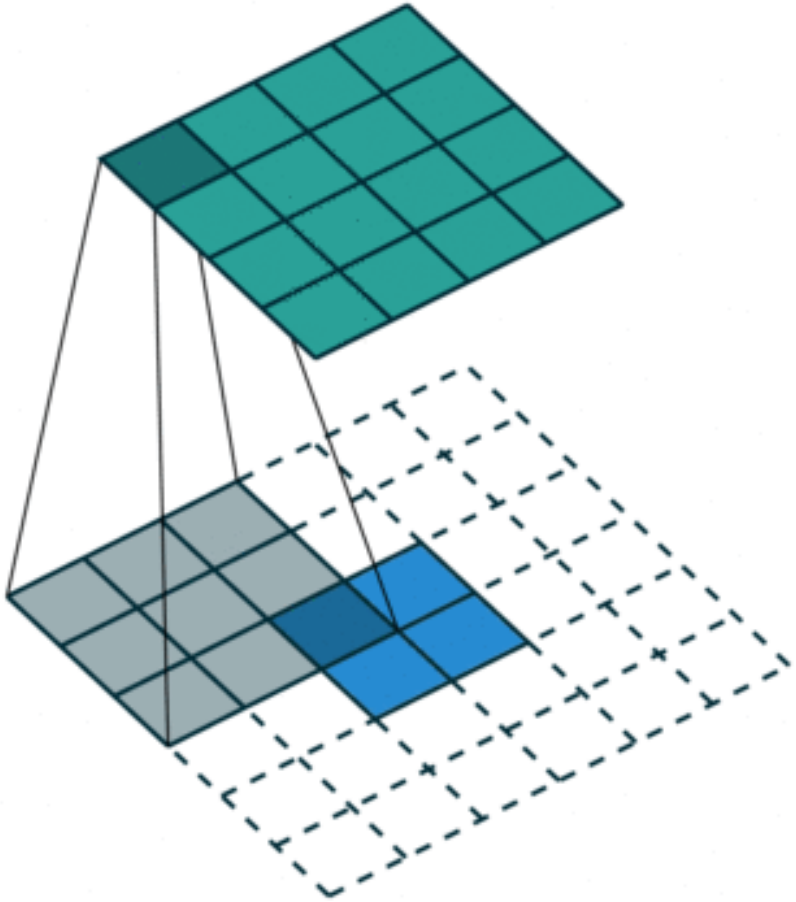
What If?

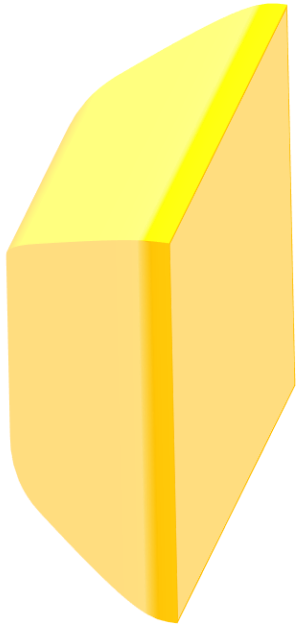


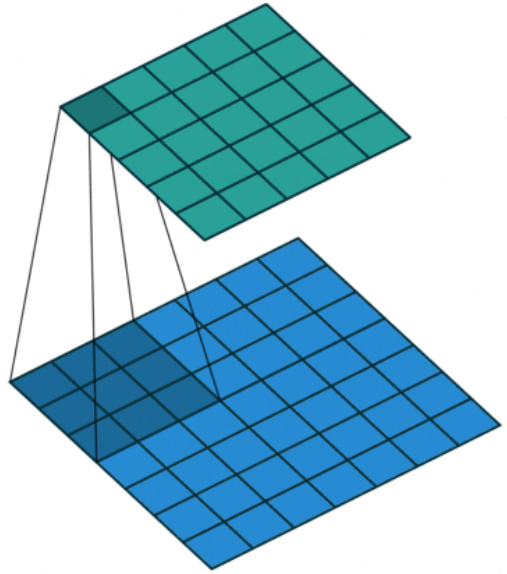
Magical NN

Deconvolution

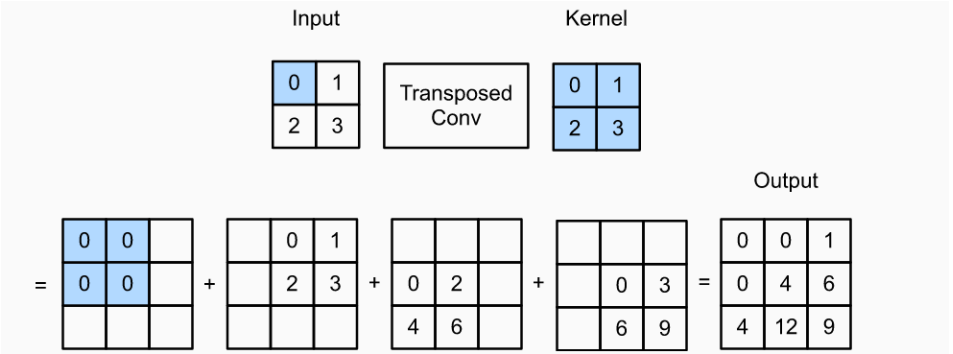
AKA Transposed Convolution





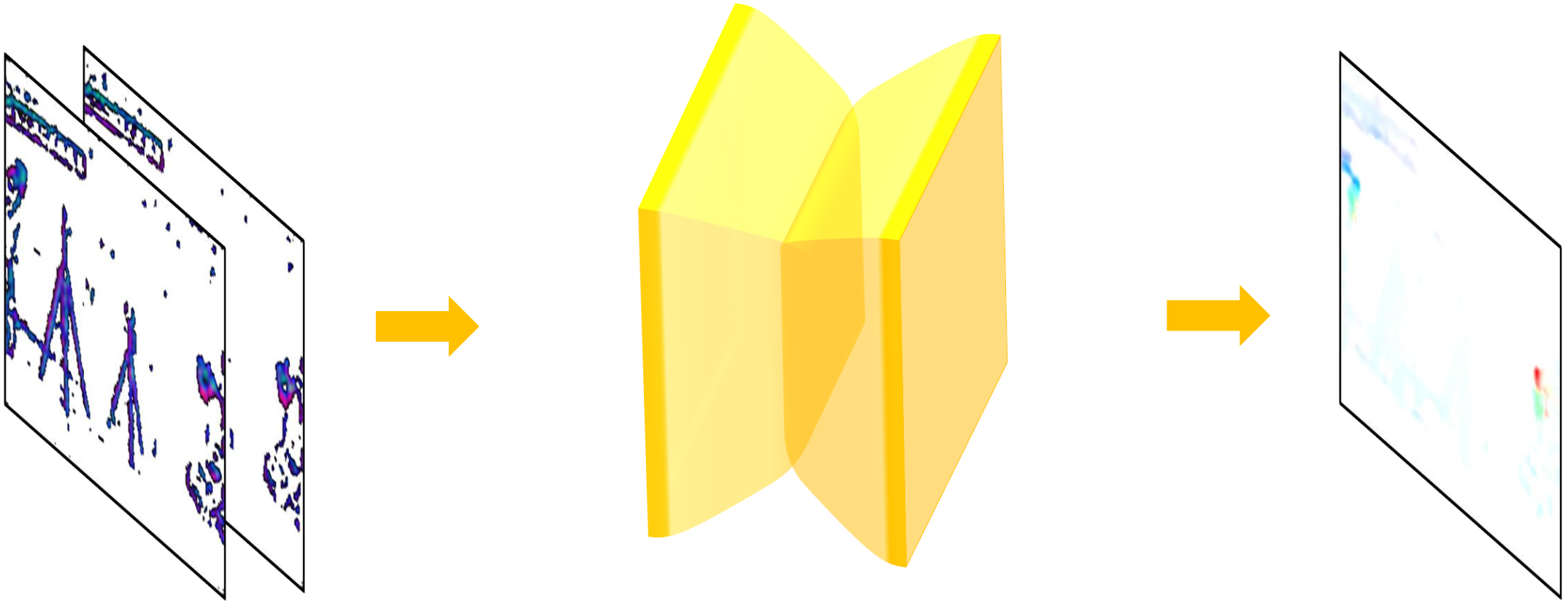


Padded Deconvolution



https://github.com/vdumoulin/conv_arithmetic

Encoder-Decoder Architecture



Sanket, Nitin J., et al. "Evdodgenet: Deep dynamic obstacle dodging with event cameras." *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

Monocular Depth Estimation!

AKA Do The Impossible With Data!

Monocular Depth Estimation!

AKA Do The Impossible With Data!

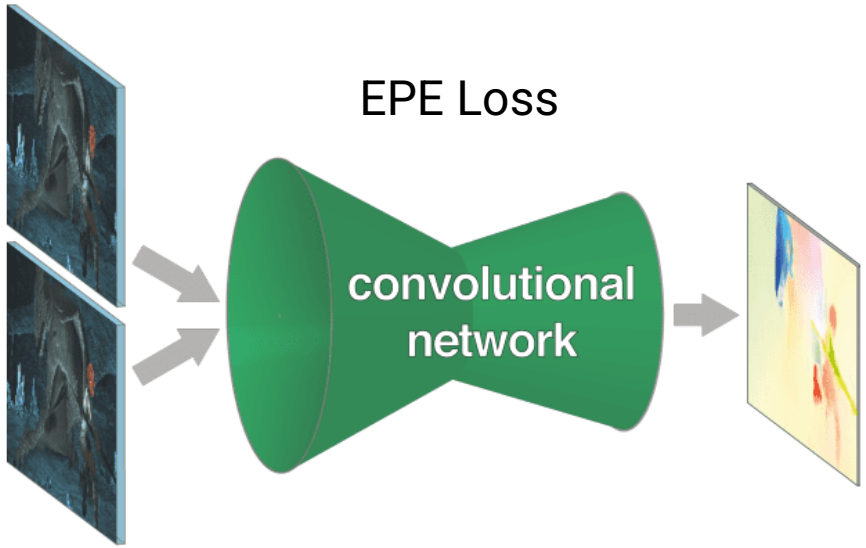
Monocular Depth Estimation!

AKA Do The Impossible With Data!

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K. and Koltun, V., 2019. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv preprint arXiv:1907.01341*.

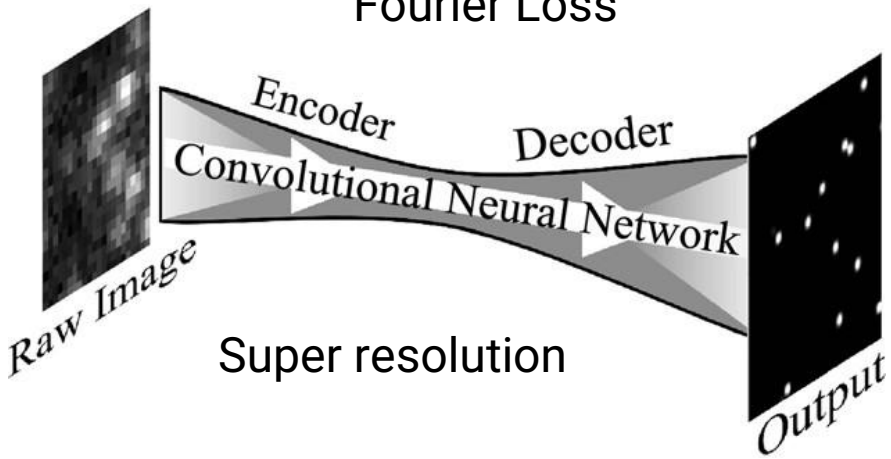
Can Do More Fancy Things!

EPE Loss



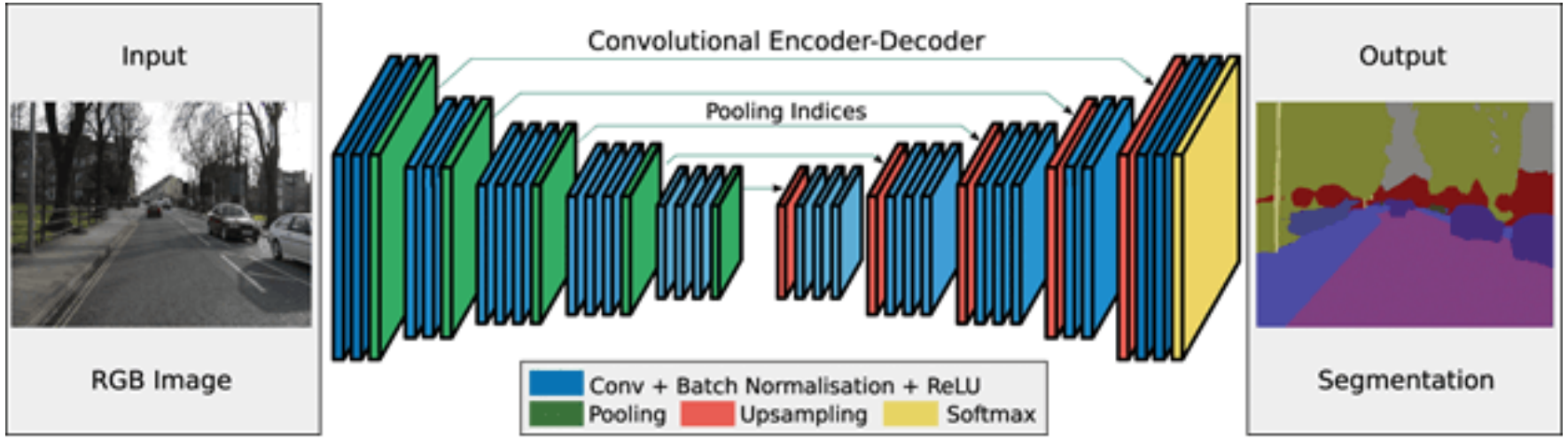
Optical Flow

Fourier Loss



Super resolution

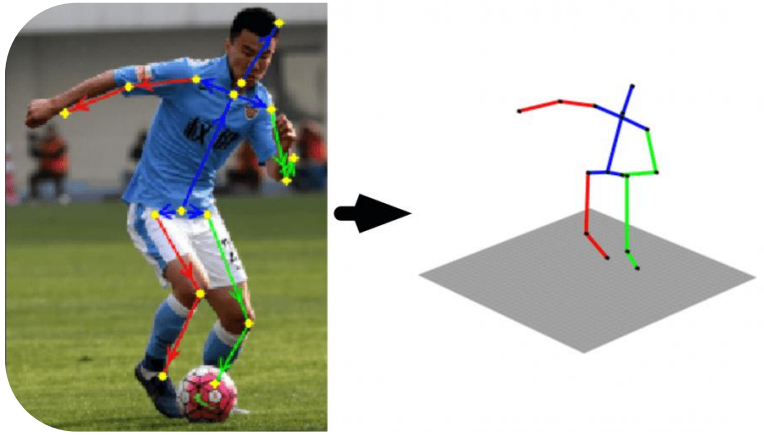
Semantic Segmentation



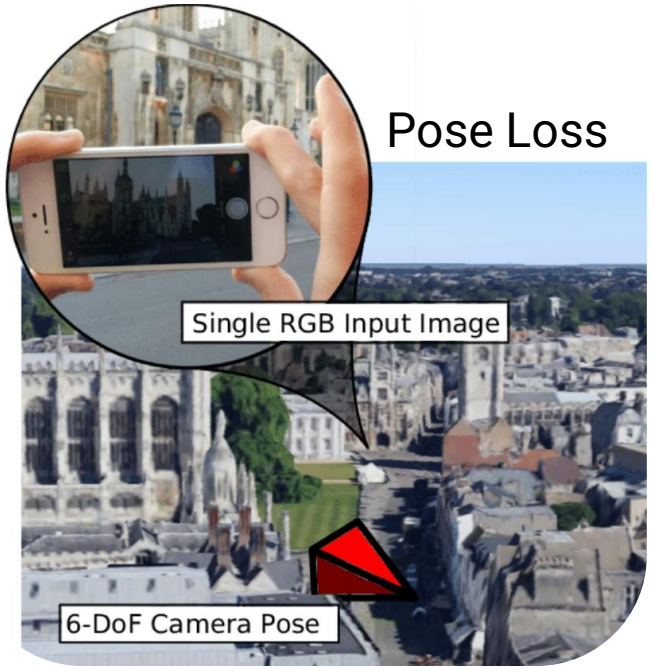
Dice Loss

And many more ….

Don’t Forget About Regression!



Human Pose
BCE and Smoothness Loss



Pose Loss

Camera Pose

Human Mesh

Perceptual and BCE Loss

And many more ….


Transformations Galore!

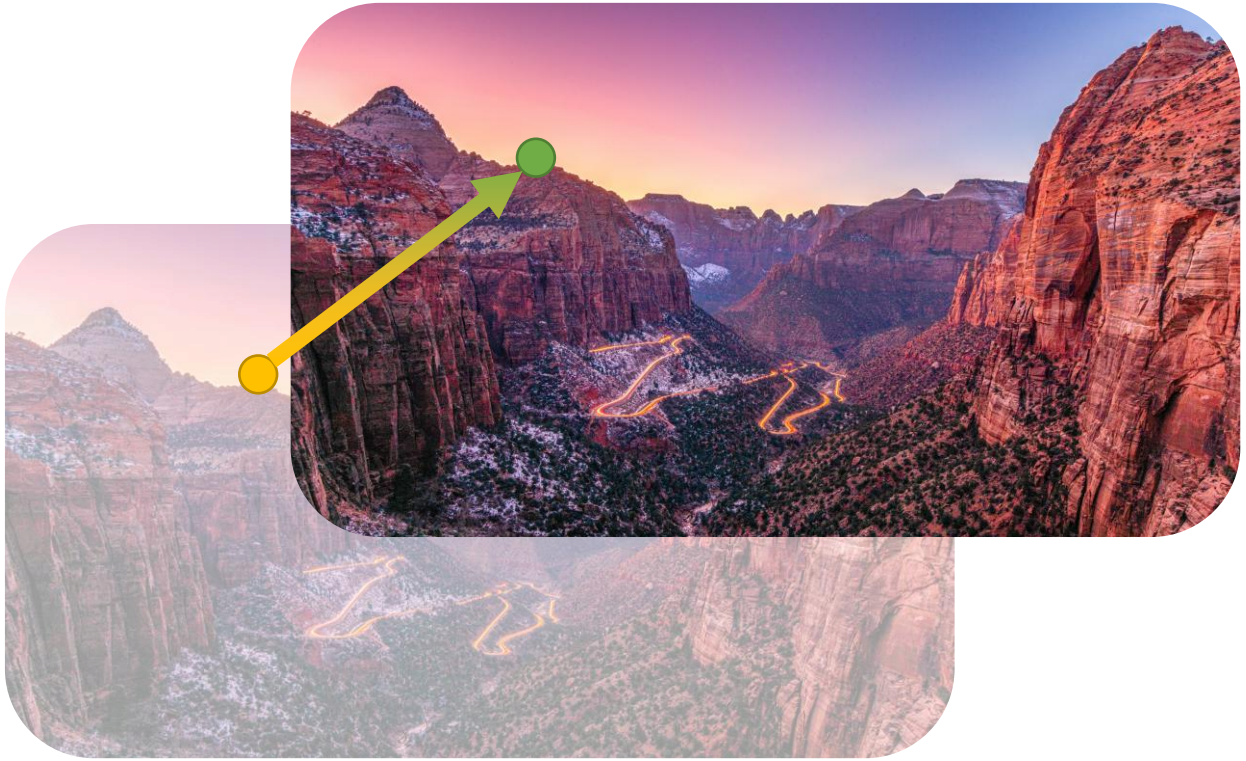
Let’s Start Simple

I_1

I_2

Let’s Start Simple

Translation Only



$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{x_2} = f(\mathbf{x_1})$$

Consider A Single Pixel

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{x_2} = f(\mathbf{x_1})$$

$$\mathbf{x_2} = \mathbf{x_1} + T_1^2$$

$$\begin{bmatrix} \mathbf{x_2} \\ 1 \end{bmatrix} = H \begin{bmatrix} \mathbf{x_1} \\ 1 \end{bmatrix}$$

$$H \in \mathbb{R}^{3\times 3}$$

$$H = \begin{bmatrix} 1 & 0 & T_{1\,x}^2 \\ 0 & 1 & T_{1\,y}^2 \\ 0 & 0 & 1 \end{bmatrix}$$

Num. of Variables?
2

Num. of Eqs.?
2

Num. Point Pairs?
1

A Little More Advanced

Rotation Only

$$\mathbf{x_2} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \mathbf{x_1}$$

$$\begin{bmatrix} \mathbf{x_2} \\ 1 \end{bmatrix} = H \begin{bmatrix} \mathbf{x_1} \\ 1 \end{bmatrix}$$

$$H \in \mathbb{R}^{3\times 3}$$

$$H = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Num. of Variables?
1

Num. of Eqs.?
1

Num. Point Pairs?
1

Rotation And Translation

AKA Euclidian AKA SE(2)

Rotation And Translation

Rotation And Translation

$$\mathbf{x_2} = H\mathbf{x_1}$$
$$H = H_{Rot}H_{Trans}$$

What is invariant?

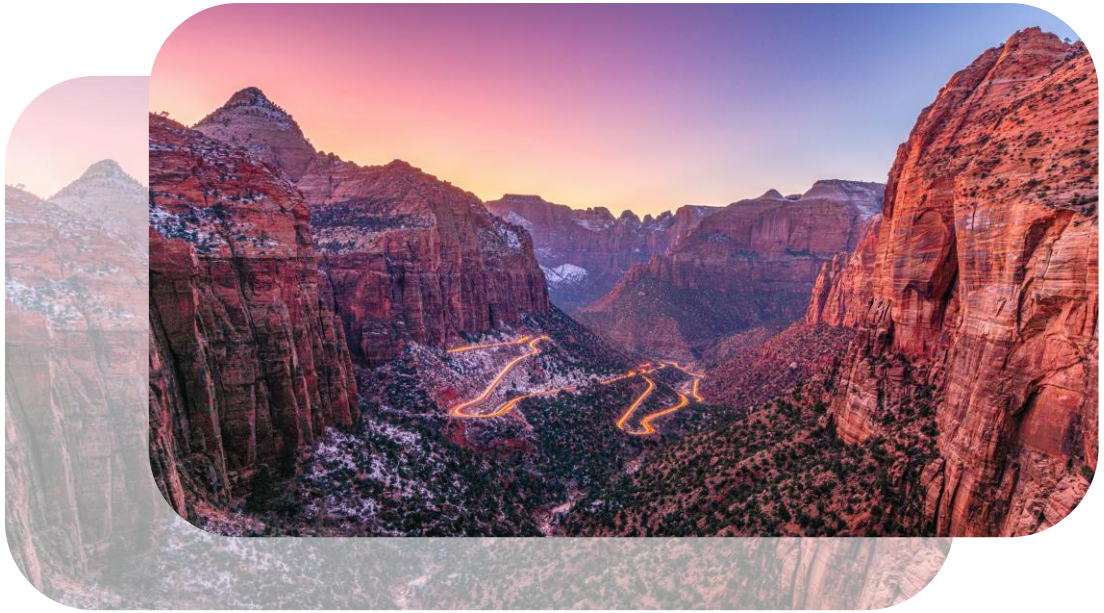
- Length
- Angle
- Area

Num. of Variables (DoF)?
3

Num. Point Pairs?
2

Rotation And Translation

Rotation And Translation



Rotation And Translation

$$H?$$

$$H = H_{Trans}H_{Rot}$$

Post-multiply your transformations!

Pre-multiply if you're doing it with respect to a global axis!

Scaling

Scaling

$$\mathbf{x_2} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \mathbf{x_1}$$

$$\begin{bmatrix} x_1 \\ 1 \end{bmatrix} = H \begin{bmatrix} x_2 \\ 1 \end{bmatrix}$$

$$H = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shear

Shear

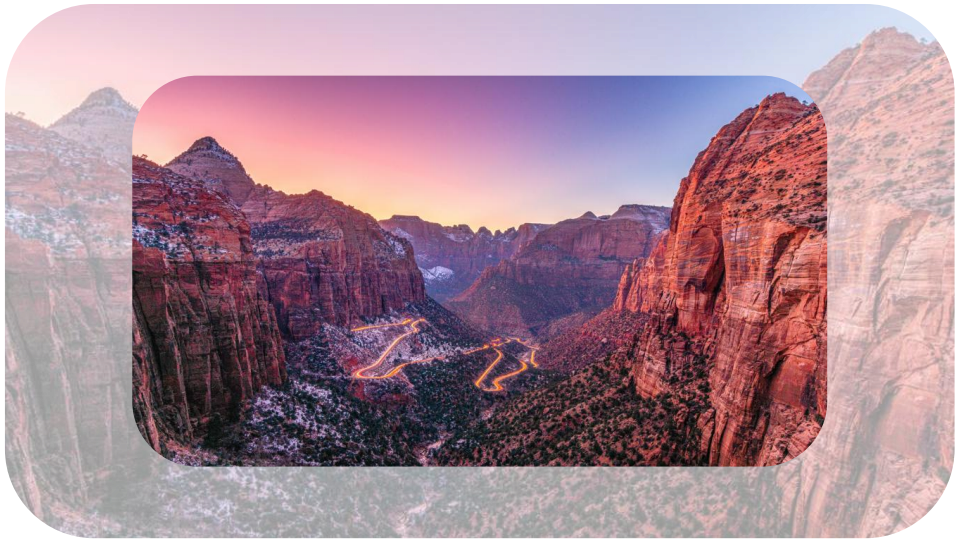
$$\mathbf{x_2} = \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix} \mathbf{x_1}$$

$$\begin{bmatrix} \mathbf{x_2} \\ 1 \end{bmatrix} = H \begin{bmatrix} \mathbf{x_1} \\ 1 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & s & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Similarity

Similarity



Similarity

Similarity

AKA Euclidean + Uniform Scaling

$$H = H_{Scale} H_{Rot} H_{Trans}$$

What is invariant?

- Length Ratio
- Angle
- Shape

Num. of Variables (DoF)?
4

Num. Point Pairs?
2

Affine

Affine

$$H = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

What is invariant?

- Collinearity
- Parallelism
- Length Ratio

Num. of Variables (DoF)?
6

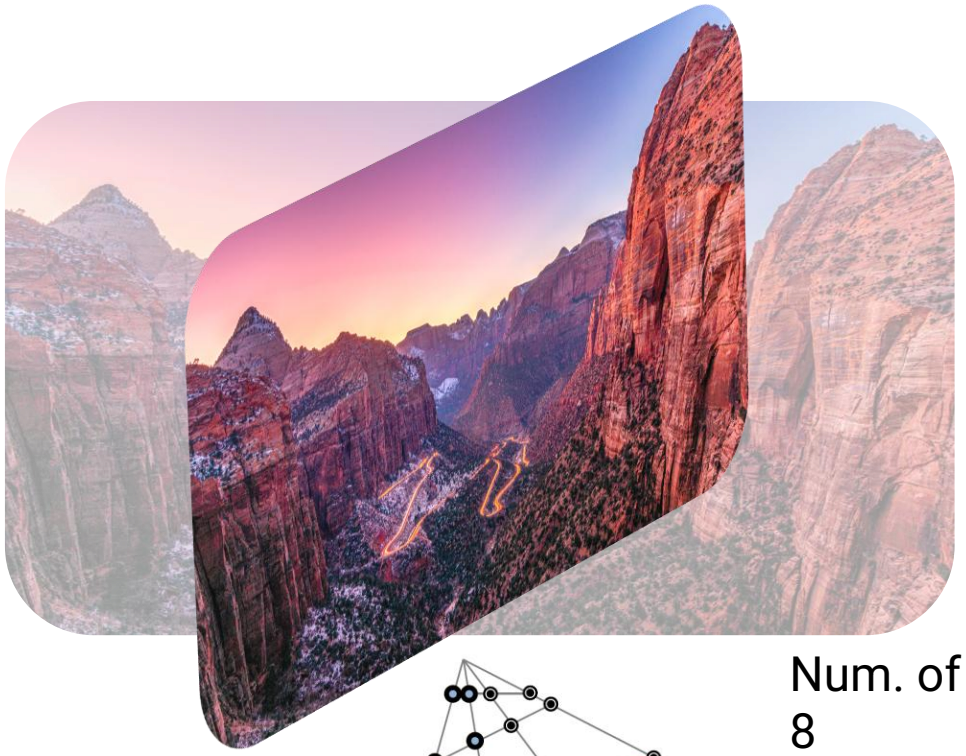
Num. Point Pairs?
3

Homography

AKA Projective

Homography

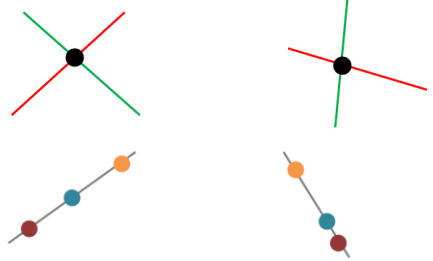
AKA Projective



$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}$$

What is invariant?

- Lines
- Cross-ratio
- Concurrency
- Colinearity



Num. of Variables (DoF)?
8

Num. Point Pairs?
4

Estimating Homography

Estimating Homography

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1 & -y_1y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{x_2} \\ 1 \end{bmatrix} = H \begin{bmatrix} \mathbf{x_1} \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x_1} \\ 1 \end{bmatrix}$$

$$x_2 = \frac{h_{11}x_1 + h_{12}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + 1}$$

$$y_2 = \frac{h_{21}x_1 + h_{22}y_1 + h_{23}}{h_{31}x_1 + h_{32}y_1 + 1}$$

$$h_{11}x_1 + h_{12}y_1 + h_{13} - h_{31}x_1x_2 - h_{32}y_1x_2 - x_2 = 0$$

$$h_{21}x_1 + h_{22}y_1 + h_{23} - h_{31}x_1y_2 - h_{32}y_1y_2 - y_2 = 0$$

We got 2 equations per point pair!
We need to find 8 unknowns ⇒ 4 point pairs!

Estimating Homography

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1 & -y_1y_2 \\ & & & & & \vdots & & \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1 & -y_1y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_2 \\ y_2 \\ \vdots \\ x_2 \\ y_2 \end{bmatrix}$$

We got 2 equations per point pair!
We need to find 8 unknowns ⇒ 4 point pairs!

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1 & -y_1y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$$

Estimating Homography

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1 & -y_1y_2 \\ & & & & A & & & \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1 & -y_1y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_2 \\ y_2 \\ x_2 \\ y_2 \end{bmatrix}$$

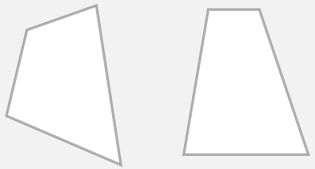
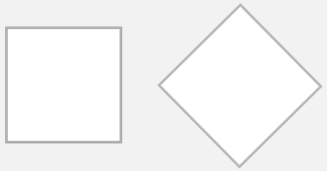
x b

We got 2 equations per point pair!
We need to find 8 unknowns ⇒ 4 point pairs!

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1 & -y_1y_2 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$$

$$Ax = b \Rightarrow x = \left(A^T A\right)^{-1} A^T b$$

Recap

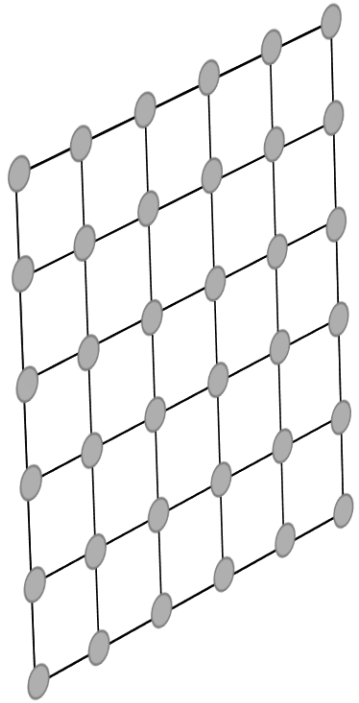
Group	Matrix	Distortion	Properties
Perspective or Homography 8 DOF	$\begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}$		Maps lines to lines but parallelism may not be preserved
Affine 6 DOF	$\begin{pmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}$		Collinearity and parallelism preserved
Similarity 4 DOF	$\begin{pmatrix} s_{11} & s_{12} & t_x \\ s_{21} & s_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}$		Angles and ratio of lengths are preserved
Euclidean/ Isometries 3 DOF	$\begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}$		Length and area preserved

Slide adapted from UMD's ENAE788M

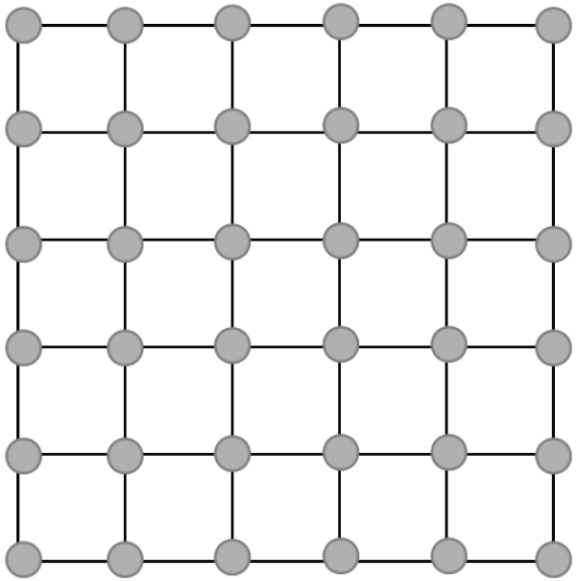
Did You Notice A Thing?

Did You Notice A Thing?

Did You Notice A Thing?

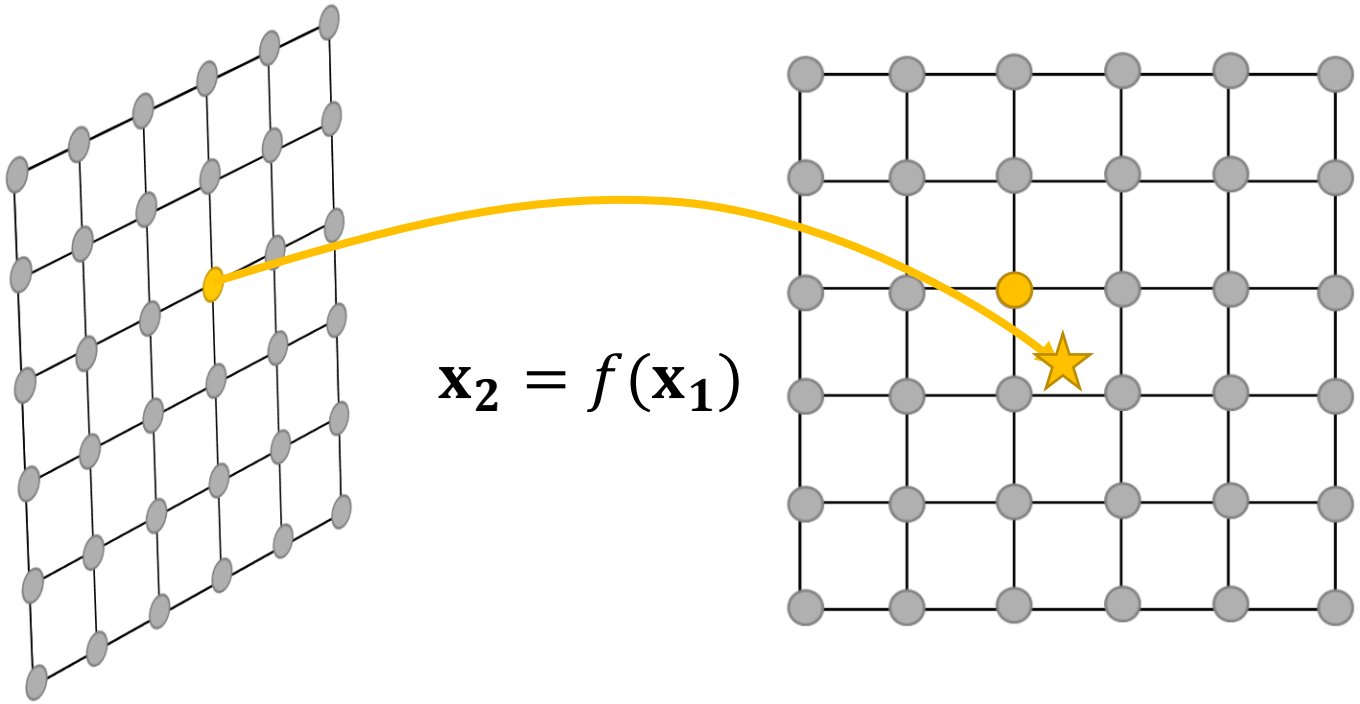


Did You Notice A Thing?



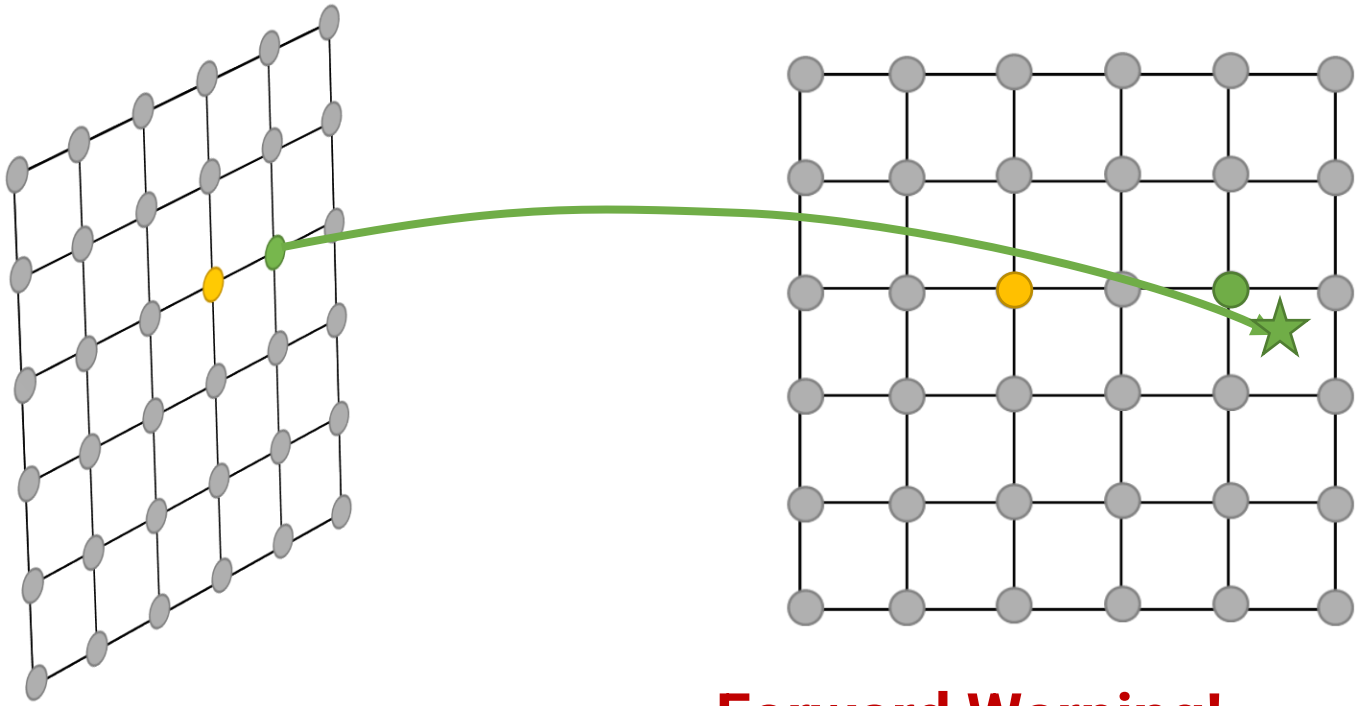
Warping

Remember we need colors!

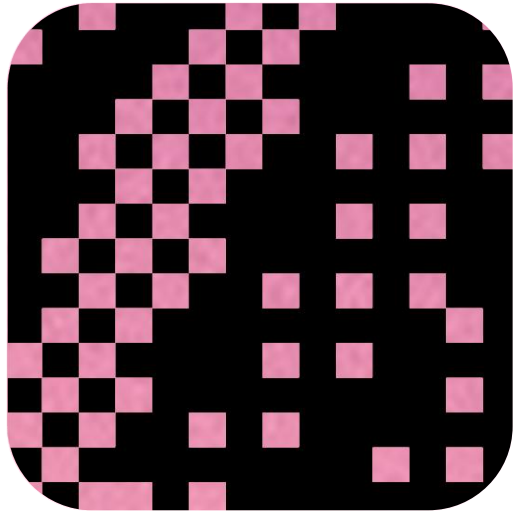


Warping

Remember we need colors!



Problem?
A lot of holes!

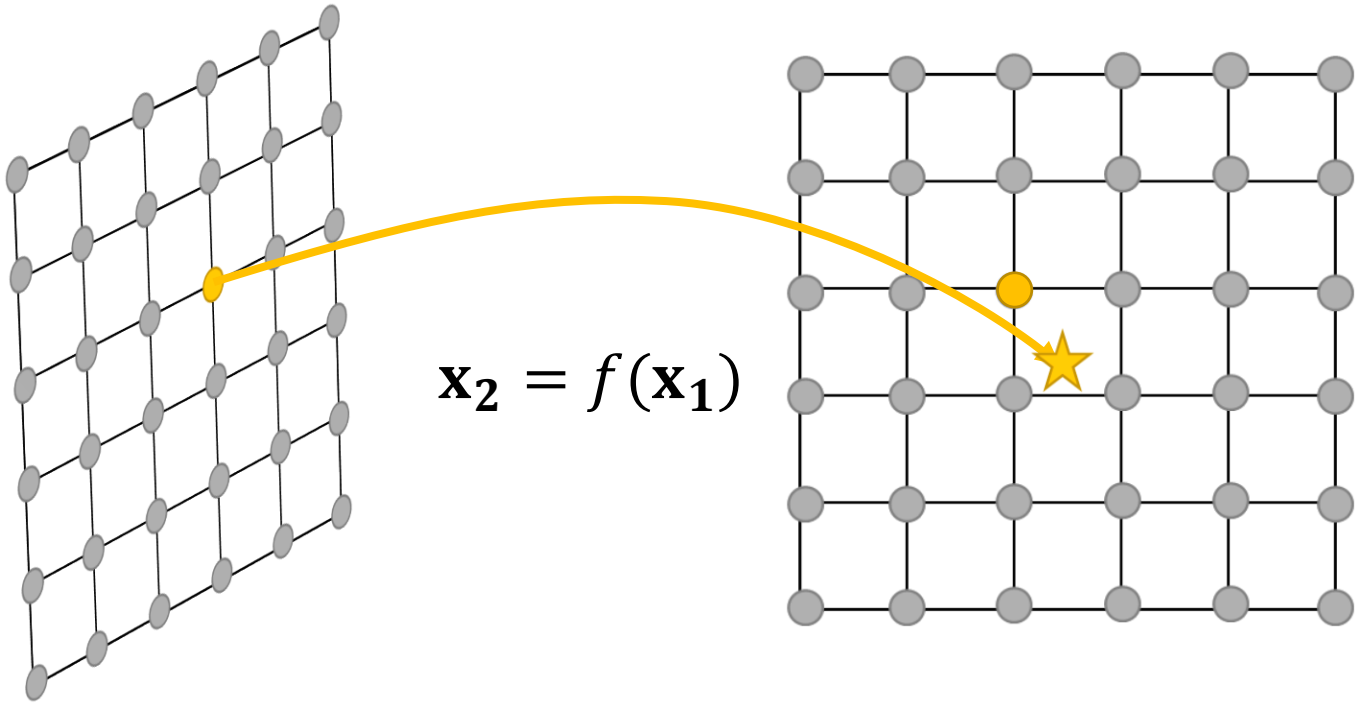


Forward Warping!

How do you distribute color amongst pixels?
You need to do splatting!
See `griddata` function

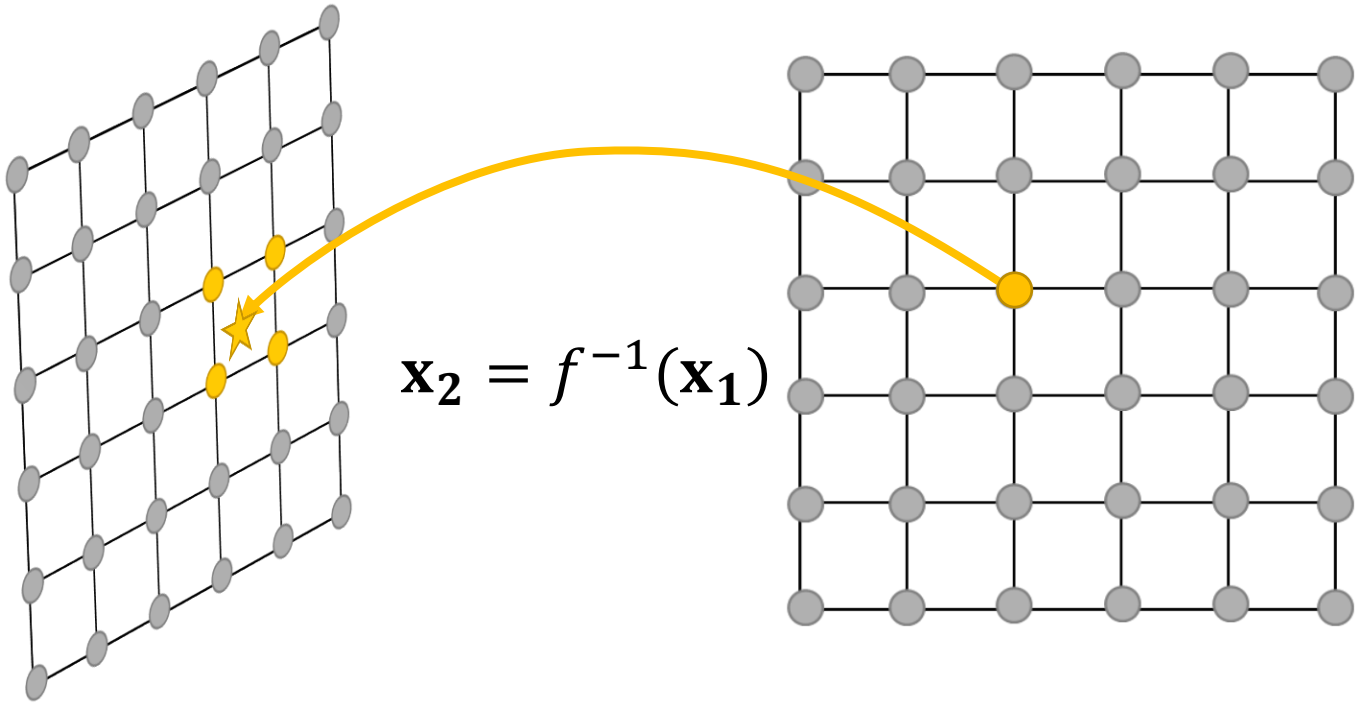
Fixing Warping Issues

Remember we need colors!



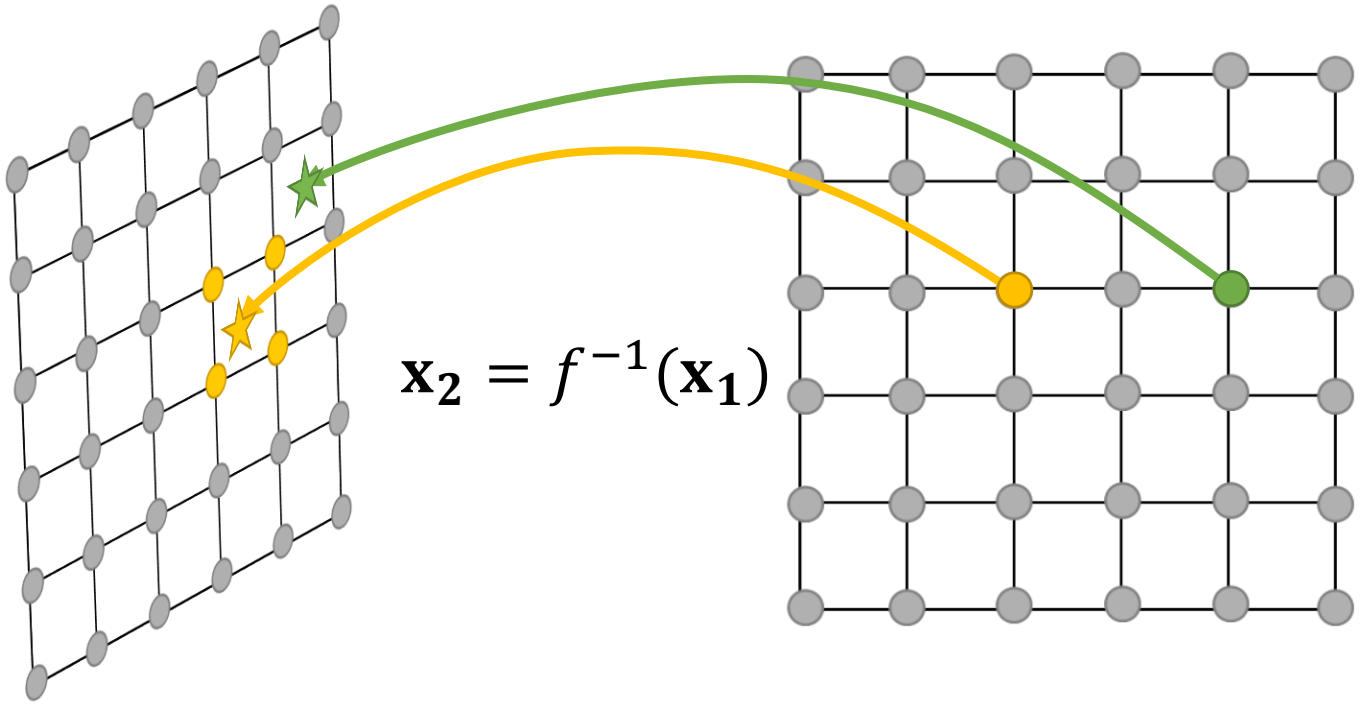
Fixing Warping Issues

Remember we need colors!



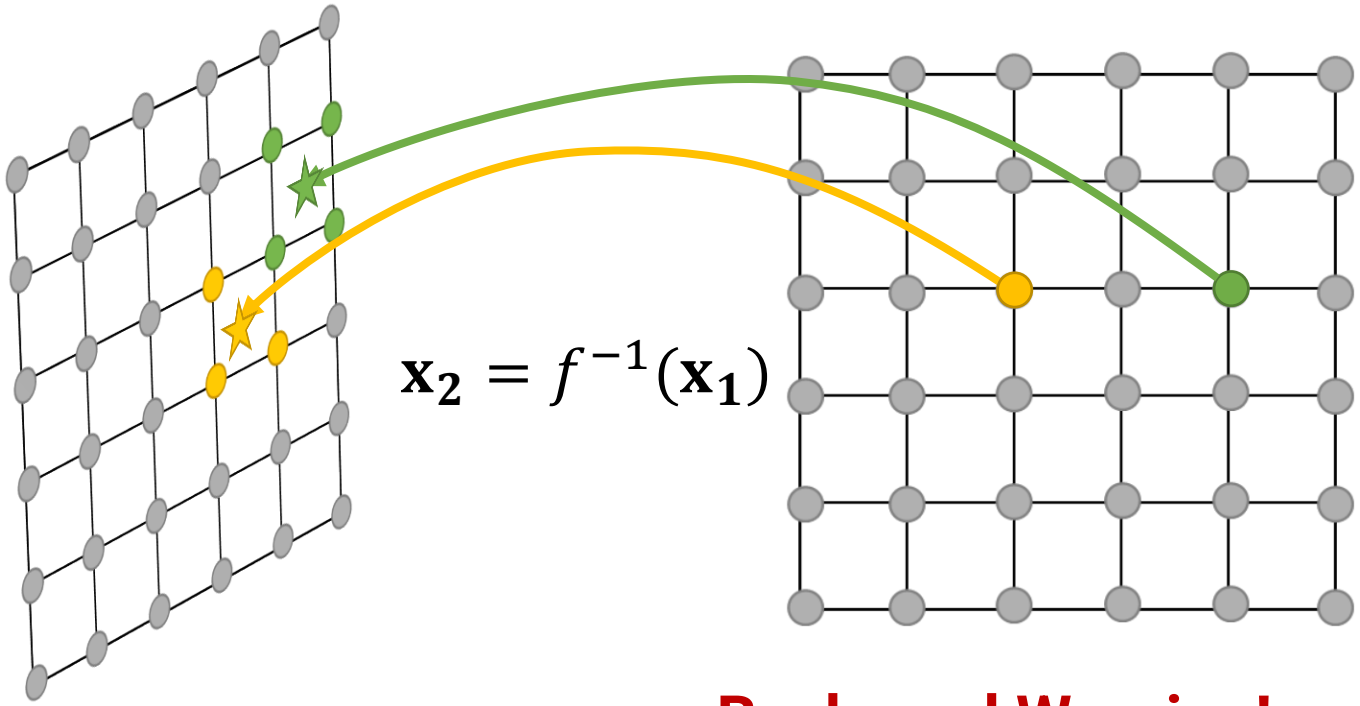
Fixing Warping Issues

Remember we need colors!



Fixing Warping Issues

Remember we need colors!



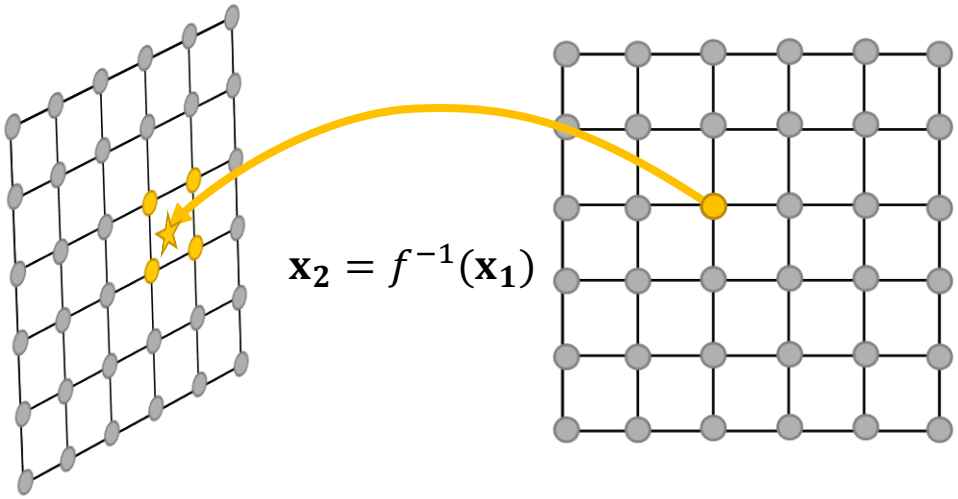
Backward Warping!

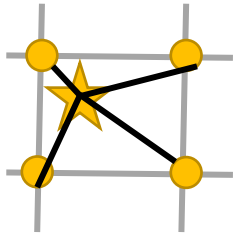
How do you distribute color amongst pixels?
Interpolation!
See `interp2` function

When To Use Either?

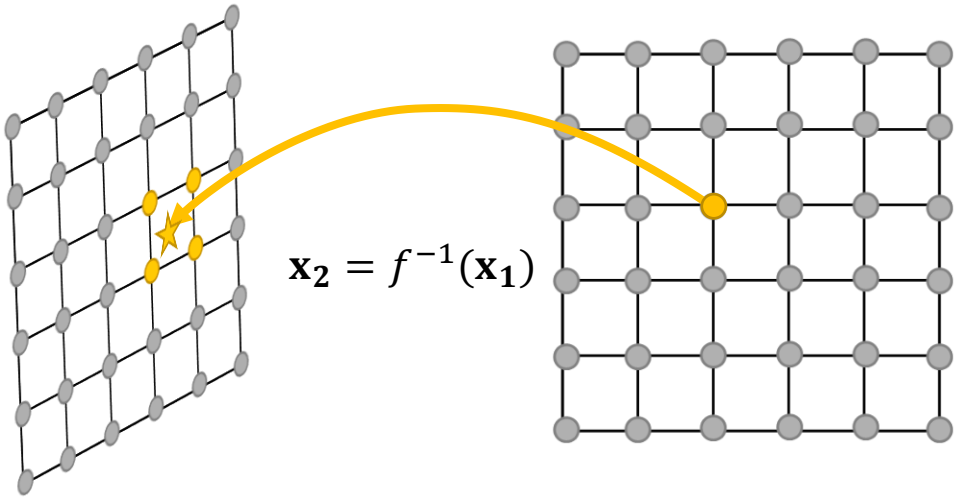
Inverse: Most of the times!
Forward: When Inversion is not possible!

Interpolation

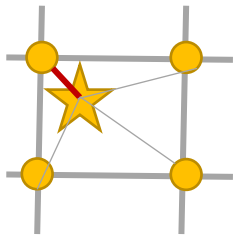




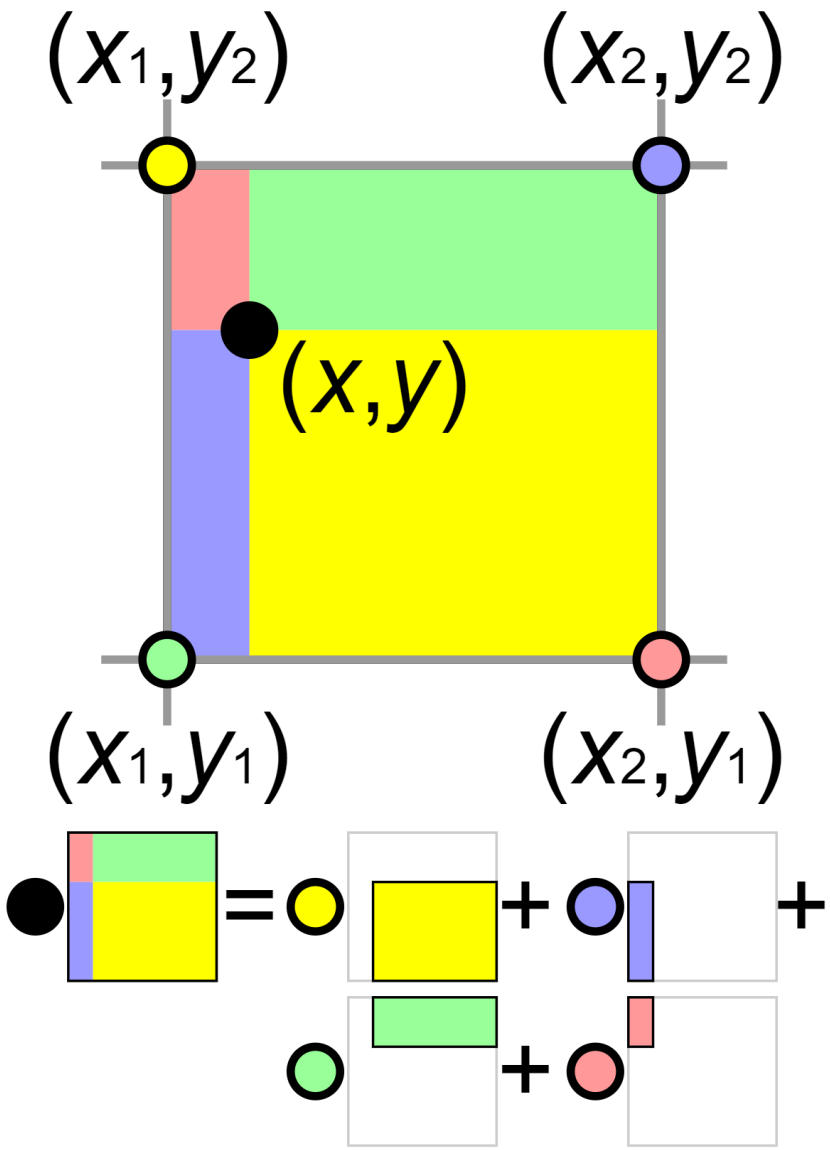
Interpolation



Nearest Neighbor

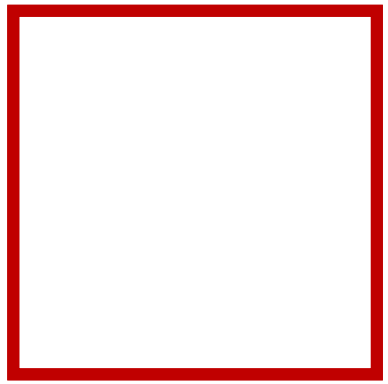


Bilinear



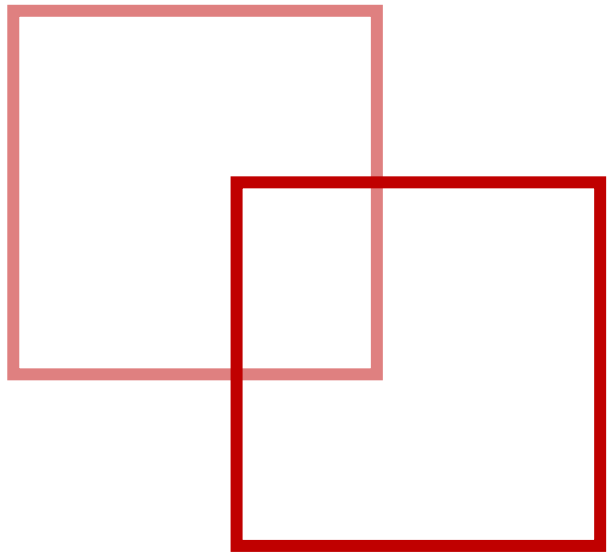
H Can Be Built Compositionally!

$$H =$$



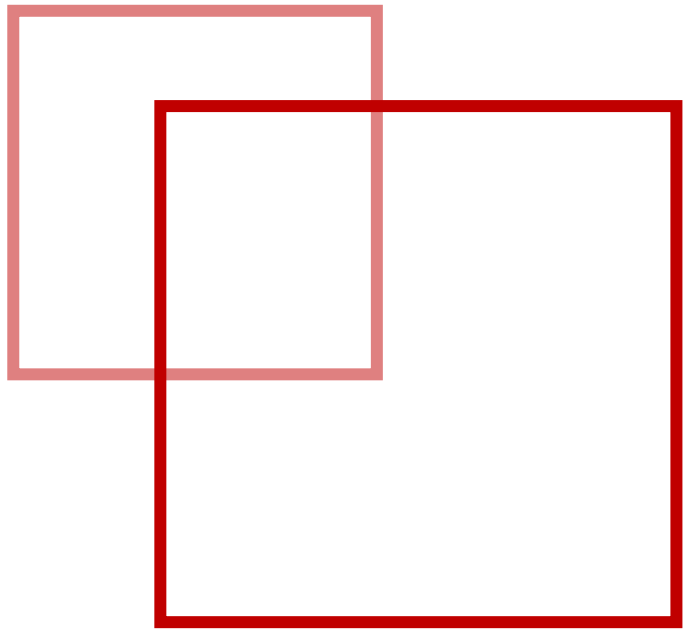
H Can Be Built Compositionally!

$$H = H_{Trans}$$



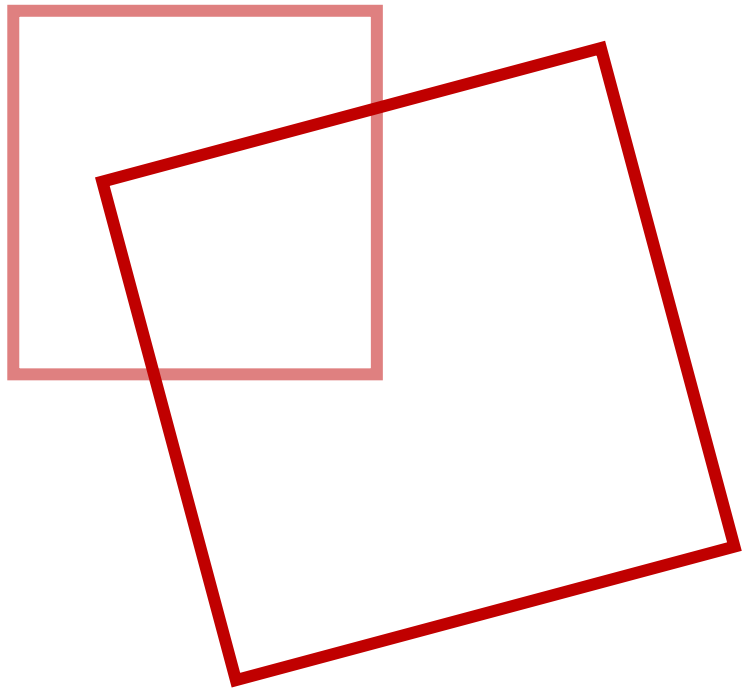
H Can Be Built Compositionally!

$$H = H_{Trans}H_{Scale}$$



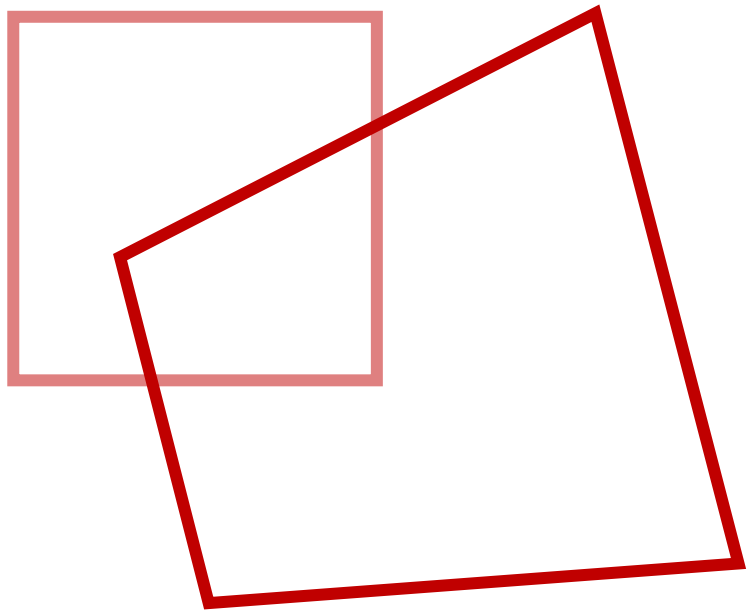
H Can Be Built Compositionally!

$$H = H_{Trans} H_{Scale} H_{Yaw}$$



H Can Be Built Compositionally!

$$H = H_{Trans} H_{Scale} H_{Yaw} H_{Distort}$$



Be sure to pay attention to transformation order here (It depends on your implementation)

Next Class!

Simulation for Data Generation and Sim2Real