Hardware for Machine Learning

Lecture 2: Deep Neural Networks

Sophia Shao

ECE1742S: Programming using CUDA, 2008, U of T

Convolutional Neural Networks for Object Classification

Alex Krizhevsky

Department of Computer Science, University of Toronto

akrizhevsky at gmail.com

Abstract I implemented a convolutional neural network with one layer of convolution. I tested it on the CIFAR-10 dataset, which consists of 6000 32×32 colour images in each of 10 classes. The convolutional net does well on the classification task and takes roughly 140x less time to train than a CPU implementation.

- Presentation: (pdf).
- Report: (pdf).
- Source code: (zip).

Convolutional Neural Networks for Object Classification in CUDA

Alex Krizhevsky (kriz@cs.toronto.edu)

April 16, 2009

1 Introduction

Here I will present my implementation of a simple convolutional neural network in CUDA. The network takes as input a 32 × 32 colour image and produces as its output one of ten possible class labels. The convolution operations, which account for 90% of the time required to train this network, are 125-150x faster on the GPU than on an Intel Core 2 Duo 2.4GHz.

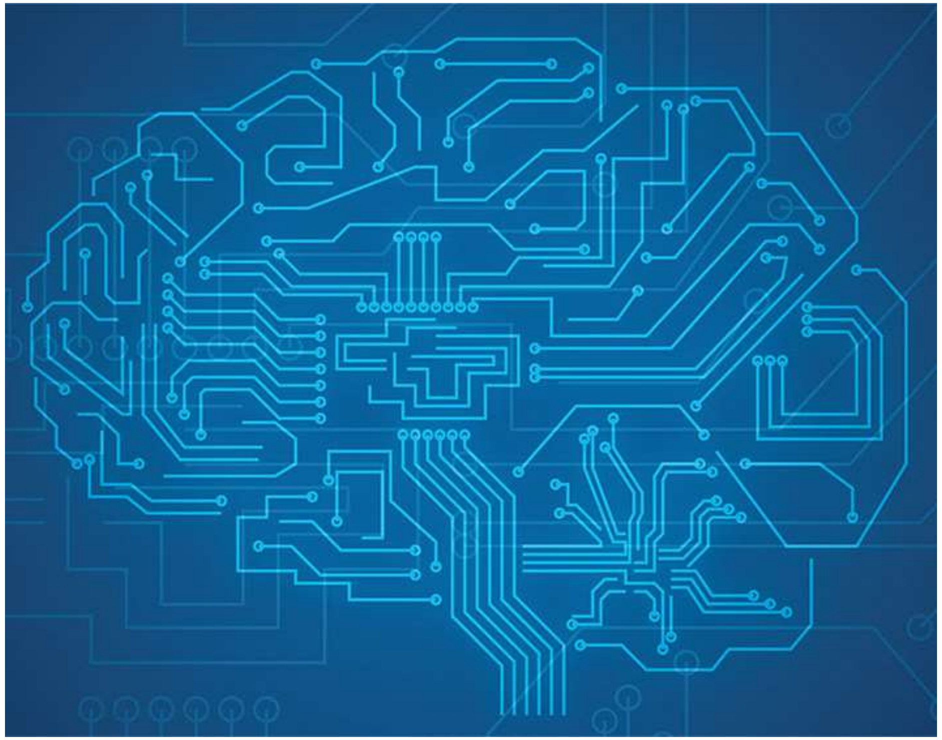
9 Conclusion

It works! The next step is to build convolutional nets with multiple layers of convolution.

http://www.eecg.toronto.edu/~moshovos/CUDA08/arx/convnet_report.pdf

Review

- Course overview:
  - A bridge between hardware and machine learning
  - **Build** efficient hardware for accelerating machine learning applications.
- Class website:
  - http://inst.eecs.berkeley.edu/~ee290-2/sp21/
- Lectures, reading/reviews, labs, and project



DNNs

- **AI, ML, and DL**
- ML Basics
- DL Overview
- AlexNet Example

Artificial Intelligence (AI)

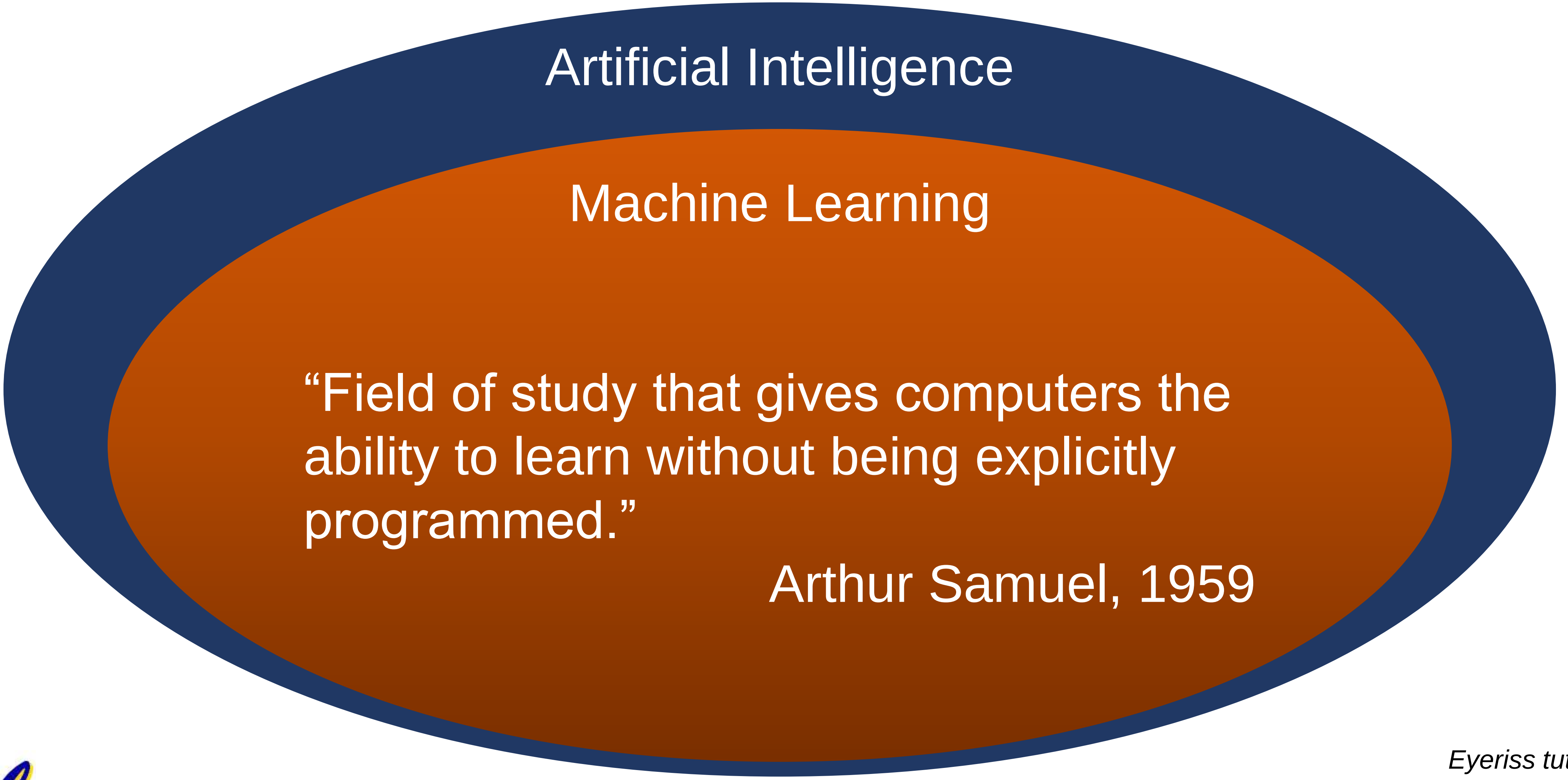
Artificial Intelligence

"The science and engineering of creating intelligent machines"

John McCarthy, 1956

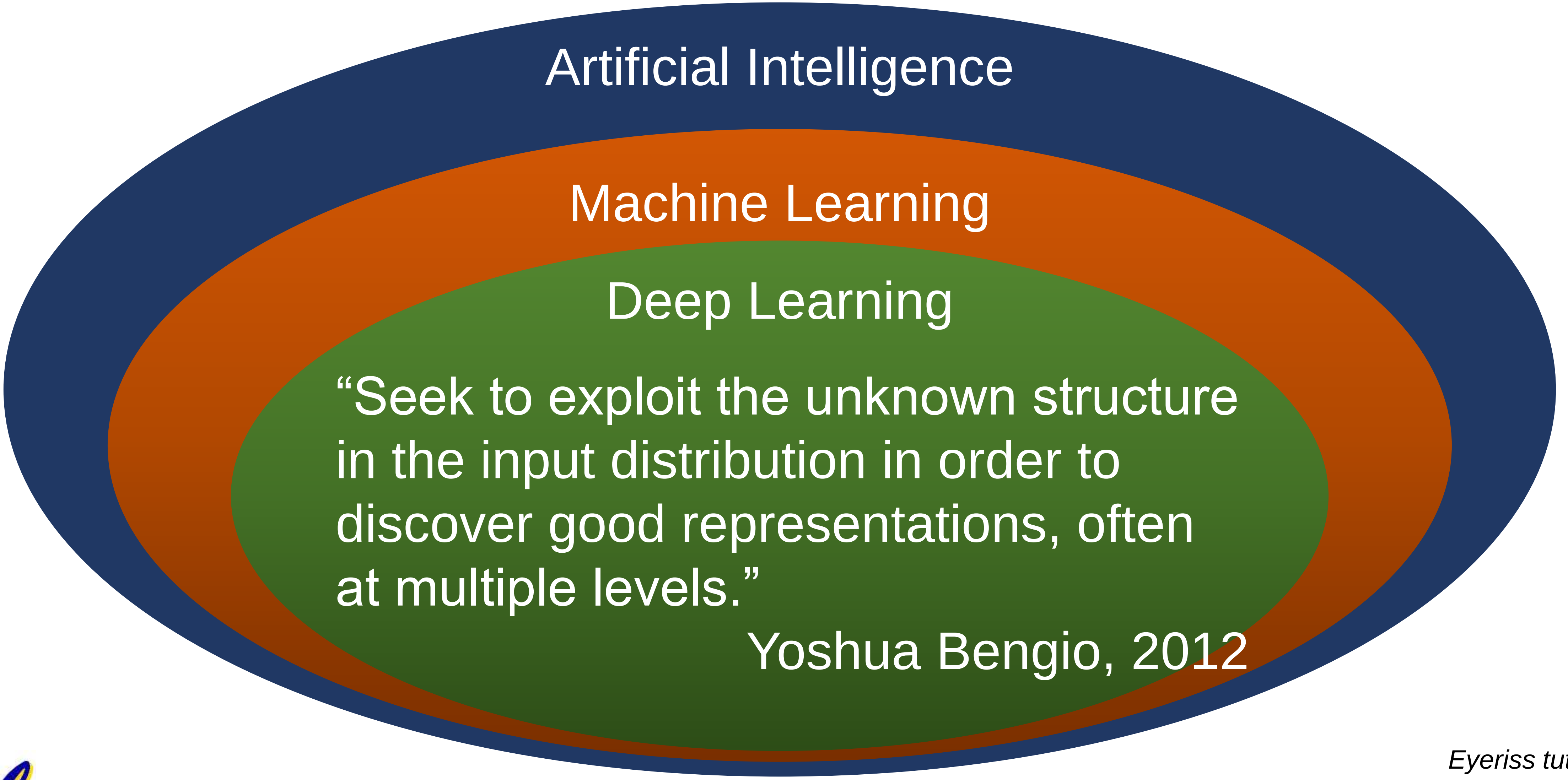
Eyeriss tutorial

Machine Learning (ML)



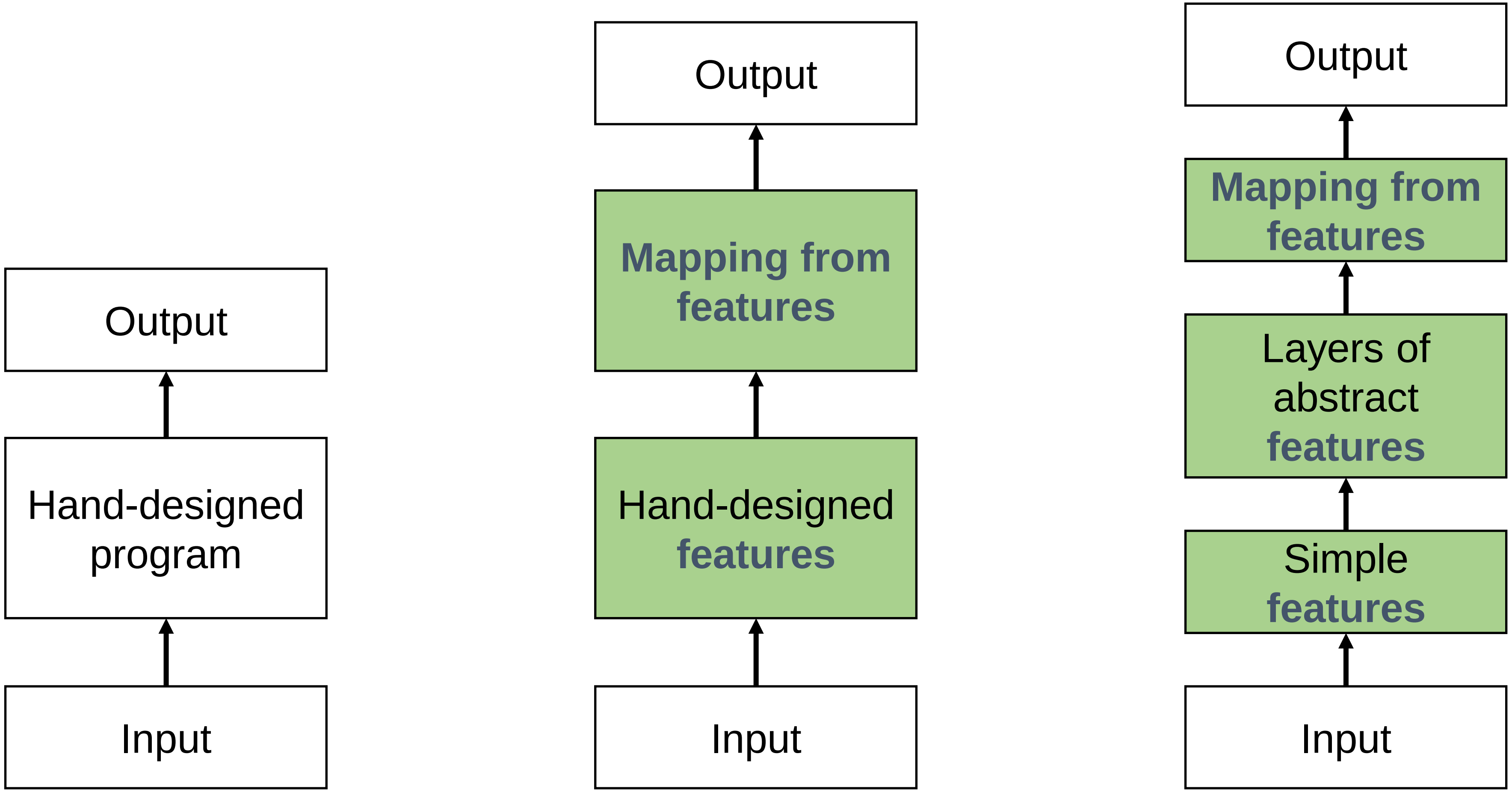
Eyeriss tutorial

Deep Learning (DL)



Eyeriss tutorial

Different AI Systems

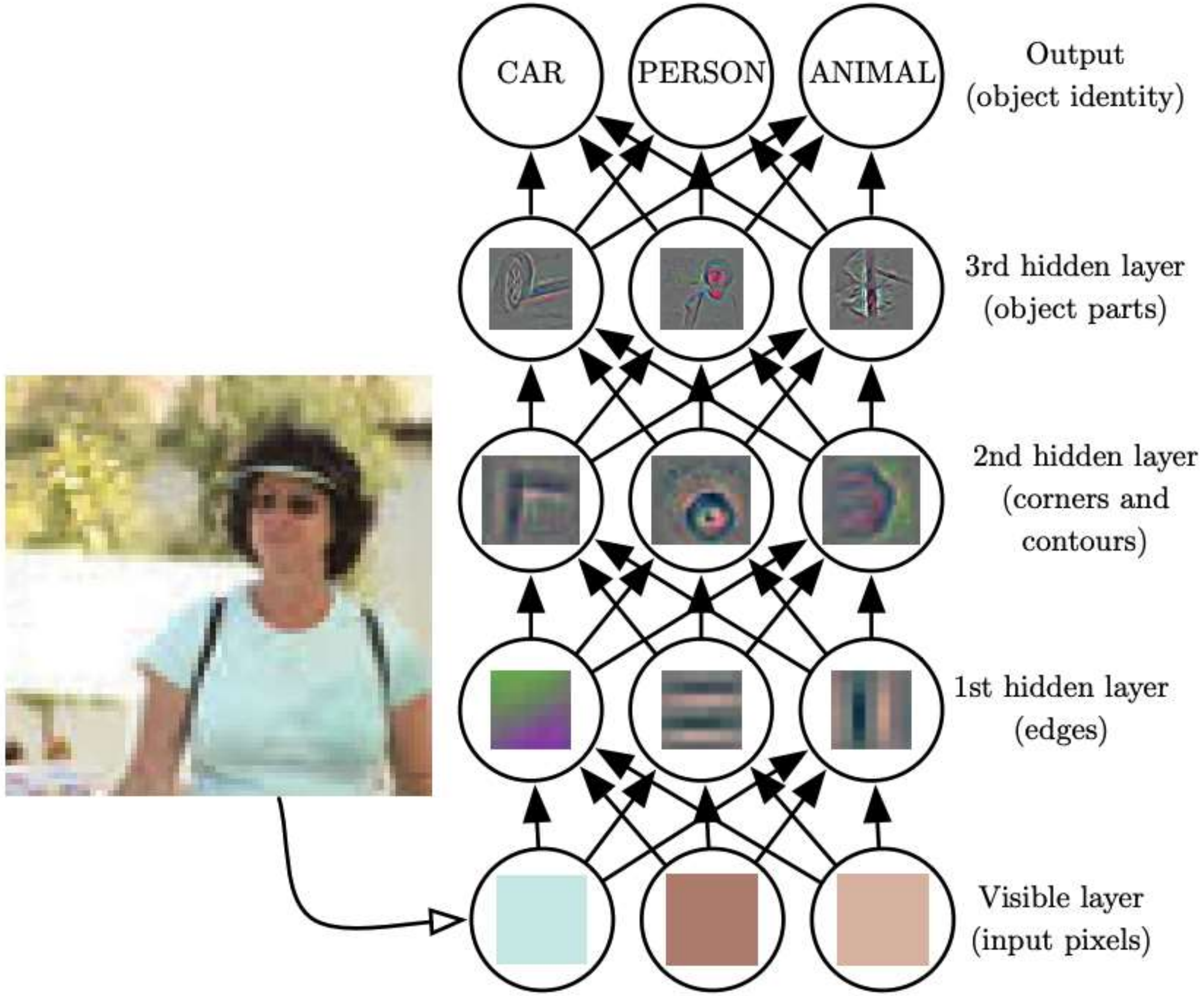


Rule-based system

Classic machine learning

Deep learning

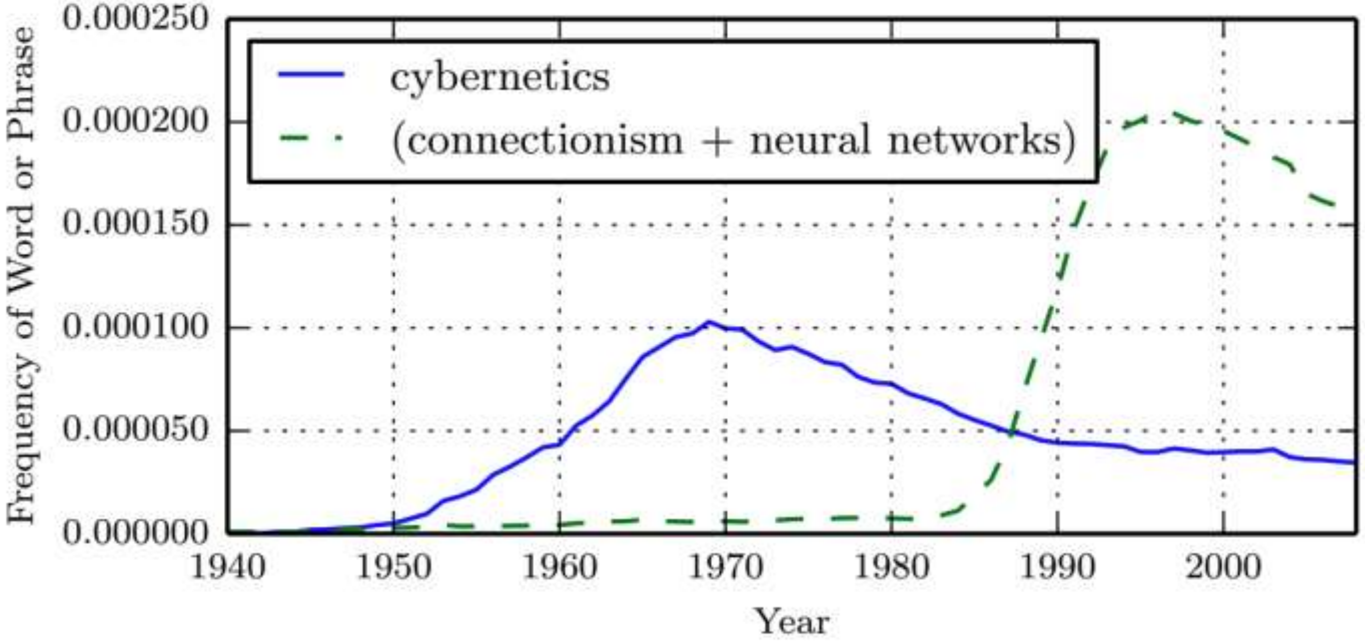
Deep Learning Example



Deep Learning, Goodfellow et. al.

Many Names of Deep Learning

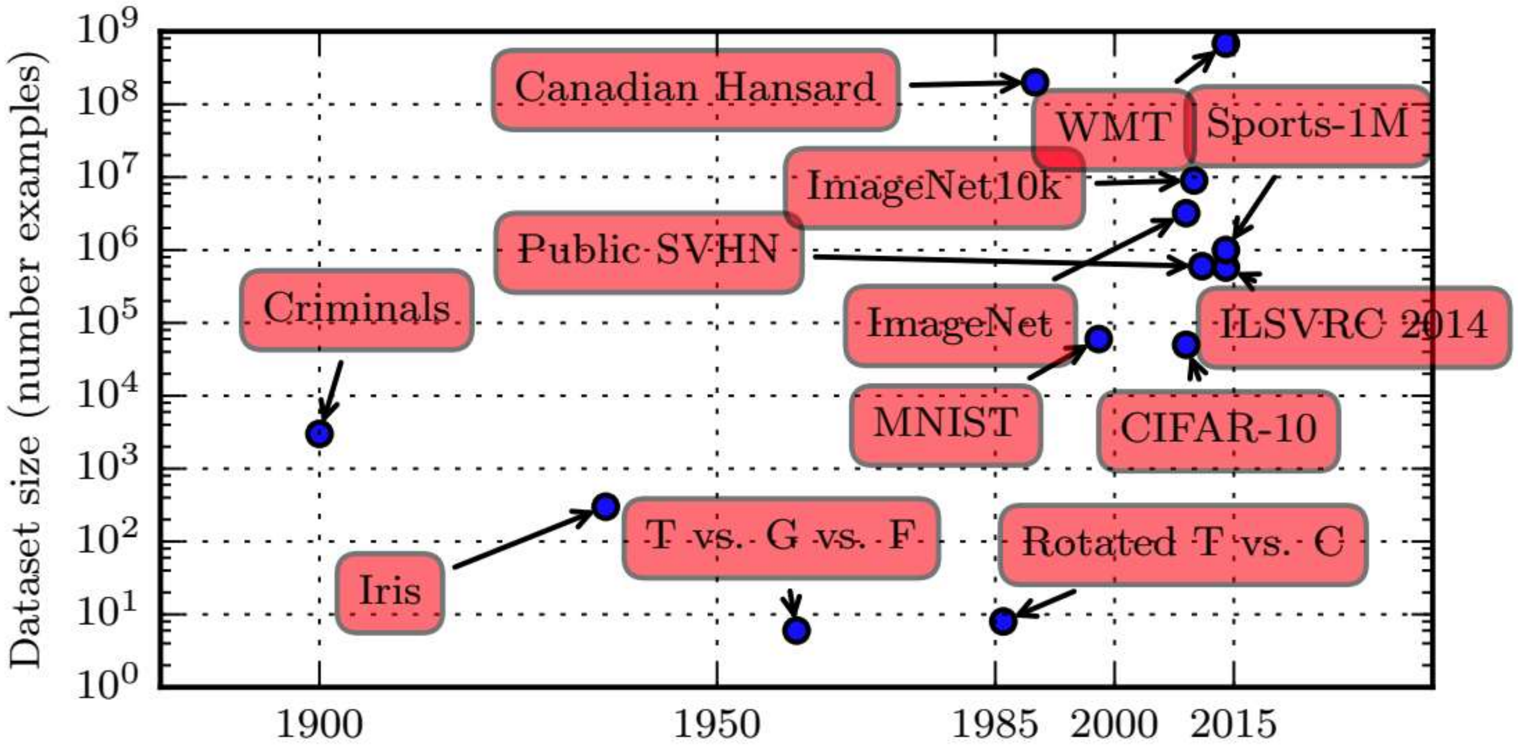
- Three waves of neural networks
  - Cybernetics
    - (1940s – 1960s)
  - Connectionism
    - (1980s – 1990s)
  - Deep learning
    - (2006s – now)
- Diminished role of neuroscience
  - Simply do not have enough information about the brain



Deep Learning, Goodfellow et. al.

Increasing Dataset Sizes

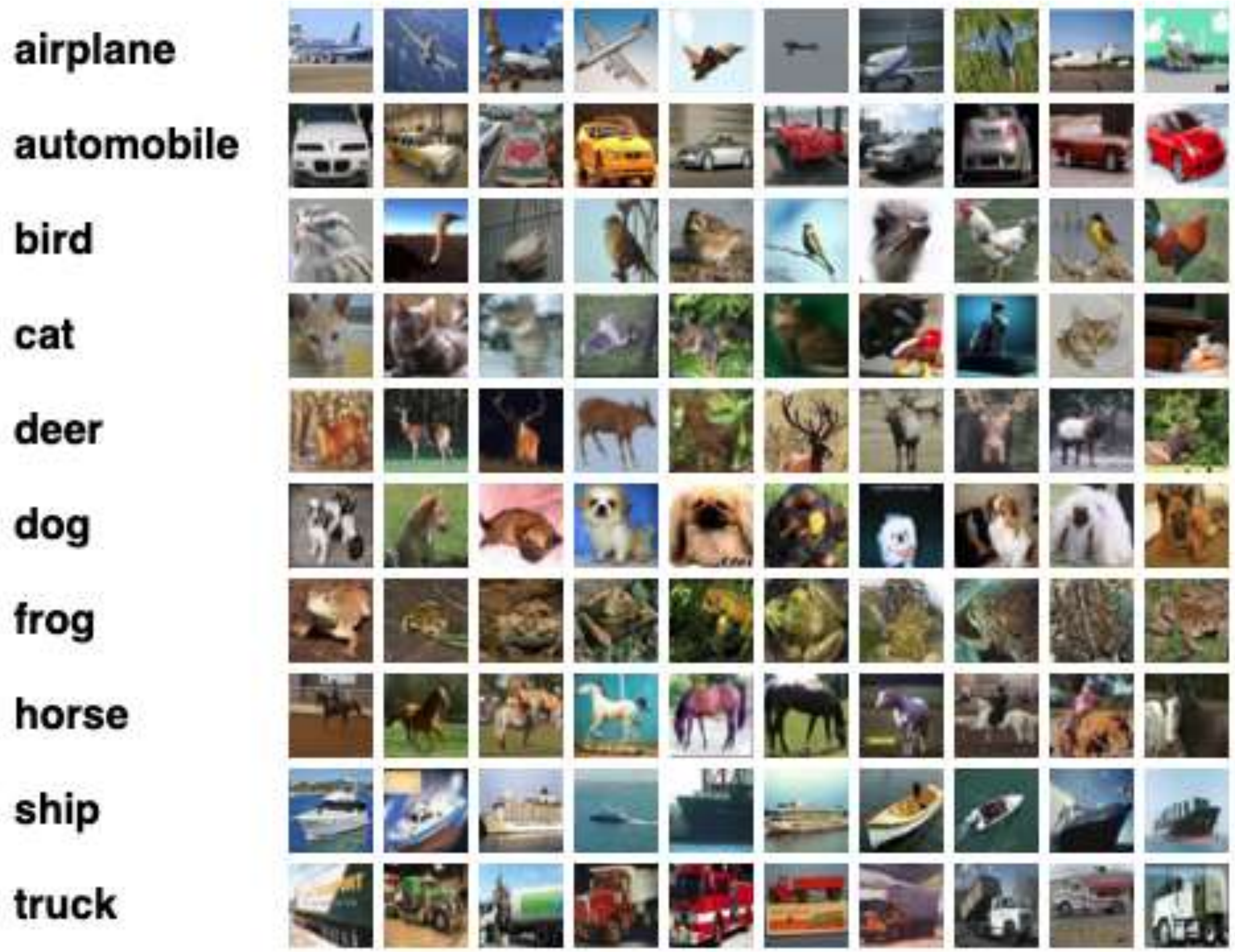
- The size of datasets has expanded remarkably over time.
- The age of “Big Data” has made machine learning easier.



Deep Learning, Goodfellow et. al.

Increasing Dataset Sizes: CIFAR10

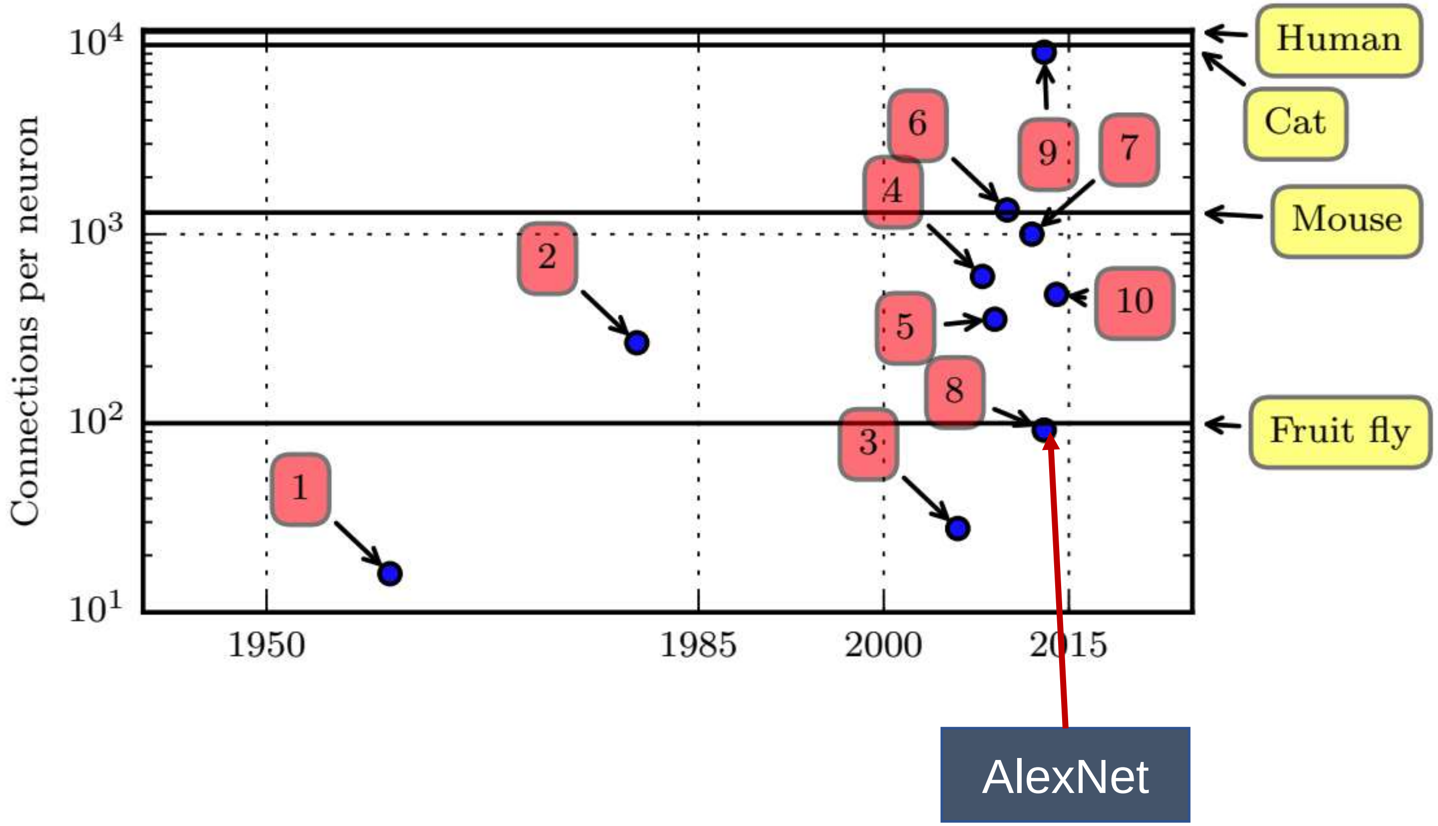
- 60,000 32x32 images
  - 50,000 training
  - 10,000 test
- 10 classes
  - 6,000 images/class



Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton

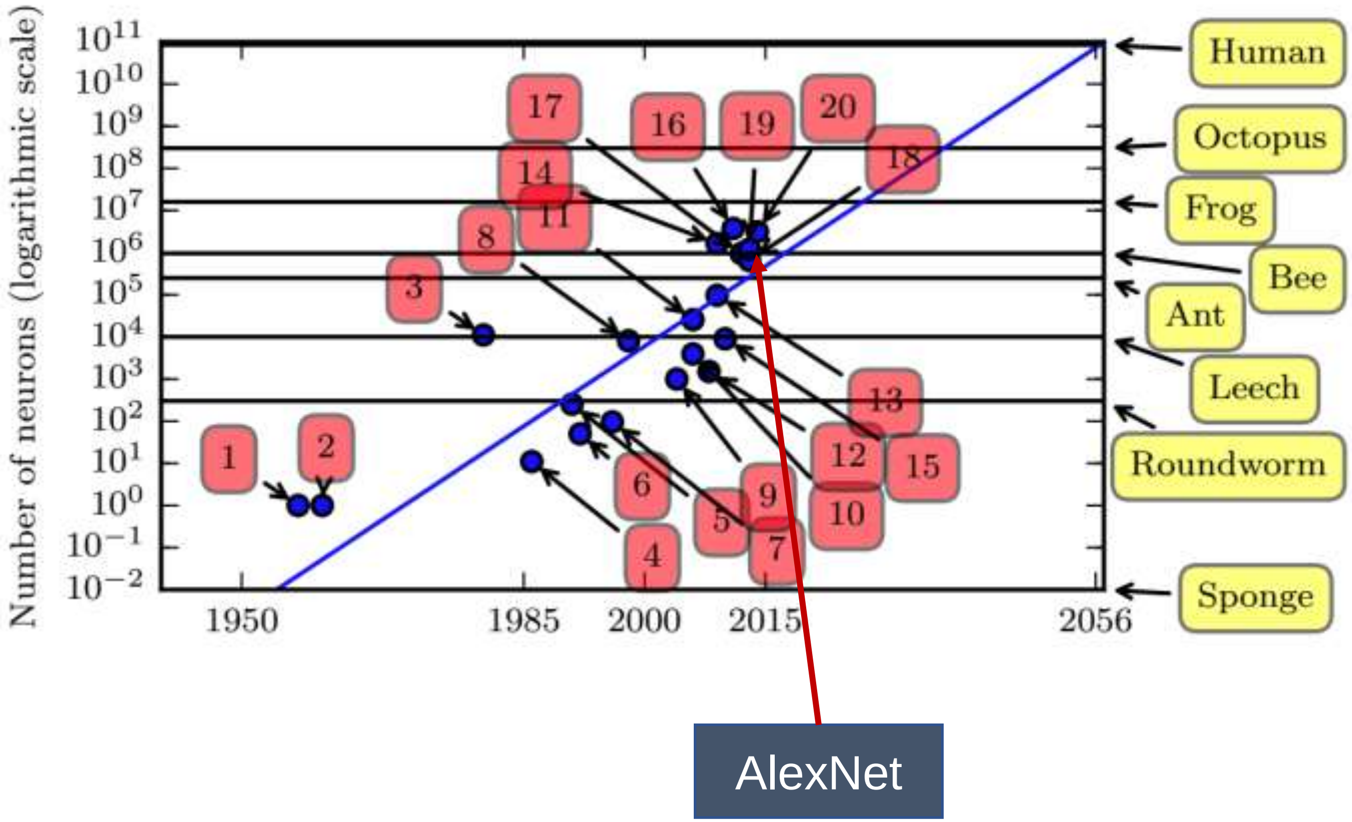
Increasing Model Sizes

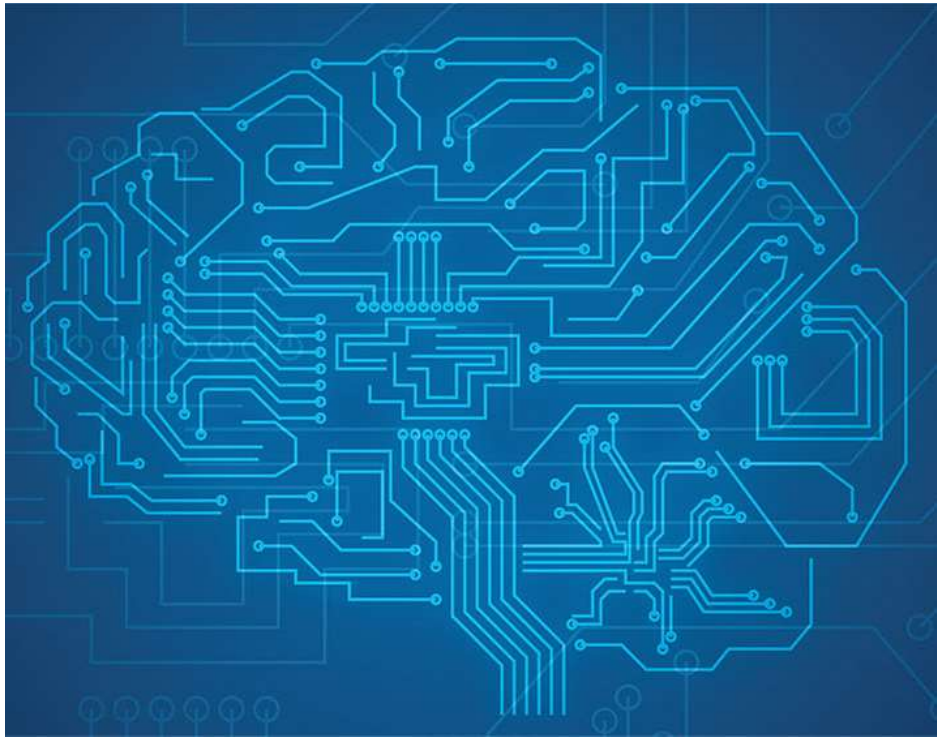
- Model size:
  - # of connections / neuron
  - # of neurons
- Largely due to the availability of faster hardware (CPUs and GPUs)
- Expect to continue



Increasing Model Size

- Model size:
  - # of connections / neuron
  - # of neurons
- Largely due to the availability of faster hardware (CPUs and GPUs)
- Expect to continue





DNNs

- AI, ML, and DL
- **ML Basics**
- DL Overview
- AlexNet Example

Machine Learning Algorithms

- “A computer program is said to ***learn*** from ***experience (E)*** with respect to some ***task (T)*** and some ***performance measure (P),*** if its performance on T, as measured by P, improves with experience E.”, Tom Mitchell, 1998

- Example: spam classification

  - Task (T): Predict emails as spam or not spam.
  - Experience (E): Observe users label emails as spam or not.
  - Performance (P): # of emails that are correctly predicted.

Machine Learning Algorithms

- Suppose a tumor classification program watches which tumor is being marked as “benign” or “malignant” by doctors, and it learns how to better predict benign/malignant tumors.

- What is the task T in this setting?

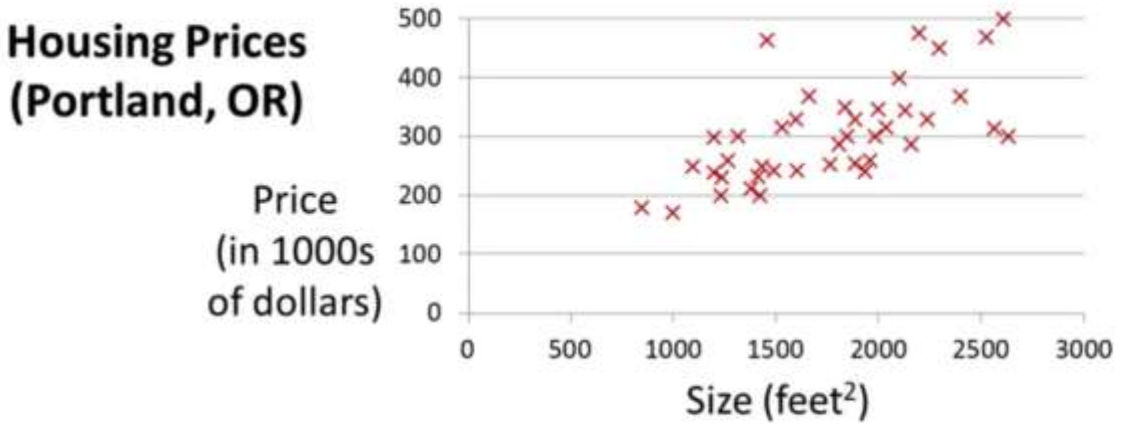
A. The number (or fraction) of tumors correctly classified as benign/malignant
B. Watching doctors label tumors as benign or malignant
C. Classifying tumors as benign or malignant
D. Non of the above. This is not a machine learning problem.

Building a ML Algorithm

- Nearly all ML algorithms can be described as particular instances of a simple recipe:
  - A dataset -> Experience (E)
  - A cost (loss) function -> Performance Measure (P)
  - A model + An optimization method -> Task (T)
- Use this recipe to see the different algorithms:
  - As part of a taxonomy of methods for doing related tasks that work for similar reasons
  - Rather than as a long list of algorithms that each have separate justifications

Example: Linear Regression

- Dataset:
  - (x, y) where x is size and y is price
  - m training examples
- Cost function:
  - Mean Squared error
  - $MSE = \frac{1}{m}\sum_{i=0}^{m}(h(x_i) - y_i)^2$
- Model:
  - $h = w_0 + w_1 * x$
- Optimization method:
  - Solve for where its gradient is 0.
  - Gradient descent



Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

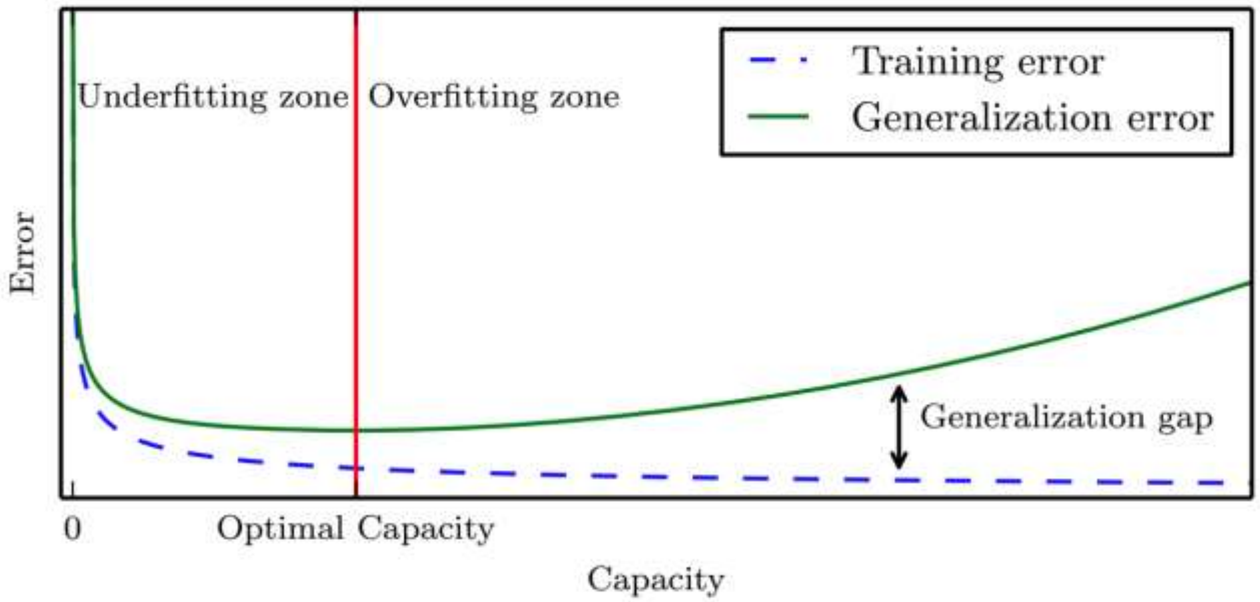
Machine Learning, Andrew Ng

Dataset

- ML tasks are usually described in terms of how the ML system should process an example.
- A dataset is a collection of many examples.
- ML algorithms can be broadly categorized as ***supervised*** and ***unsupervised*** by what kind of dataset they process.
  - Supervised: each example of the dataset is associated with a label or target
    - E.g., Classification, regression
  - Unsupervised: experience dataset without labels
    - E.g., Clustering
  - Reinforcement: Not a fixed dataset, interact with an environment

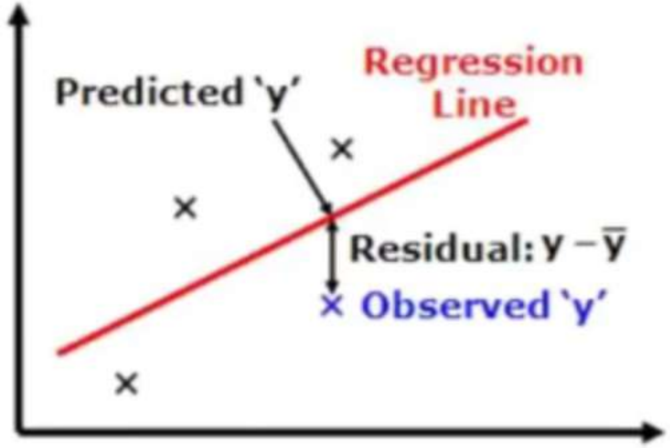
Dataset:

- Generalization:
  - Algorithm performs well on new, previously unseen inputs (not just those on which the model was trained).
- Partition the dataset into:
  - Training set: where the model was trained
  - Test set: where the trained model was tested
- Overfitting:
  - Large gap between training and test errors



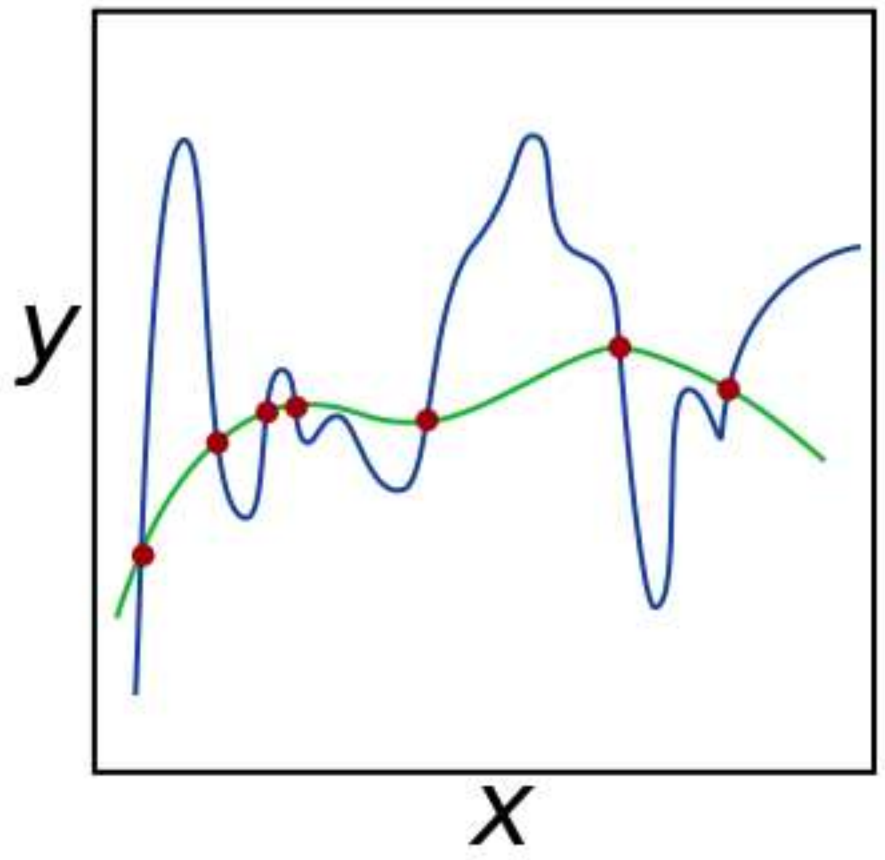
Cost function

- Measures the performance of a ML model for given data
- Quantifies the error between predicted and expected values in the ***training*** set.
  - $CF = MSE_{train} = \frac{1}{m}\sum_{i=0}^{m}(\hat{y}_i - y_i)^2$



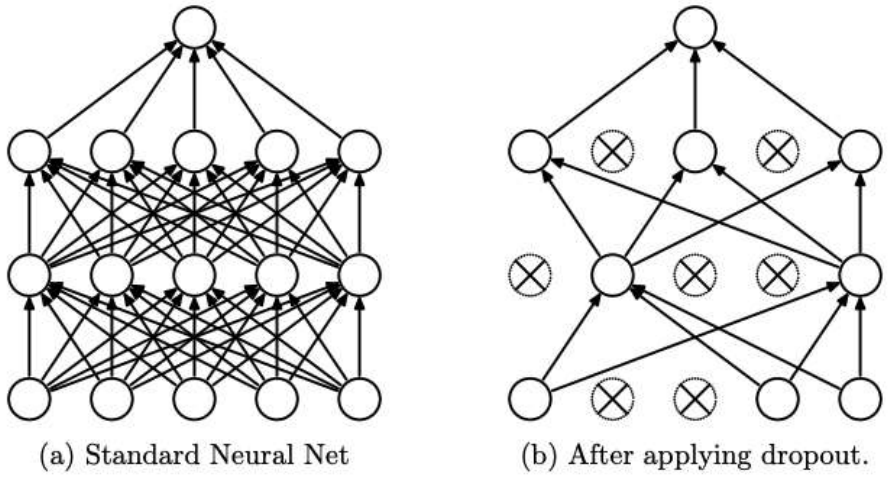
Cost function

- Regularization: modifications to reduce the generalization error but not the training errors
  - Weight decay (L2/1 regularization)
    - Expressing preferences of smaller weights
    - $CF = MSE_{train} + \lambda \sum_i w_i^2 \ (L2)$
    - $CF = MSE_{train} + \lambda \sum_i |w_i| \ (L1)$



Cost function

- Regularization: modifications to reduce the generalization error but not the training errors
  - Dropout
    - Temporally remove nodes from network
    - Train a large ensemble of models that share parameters



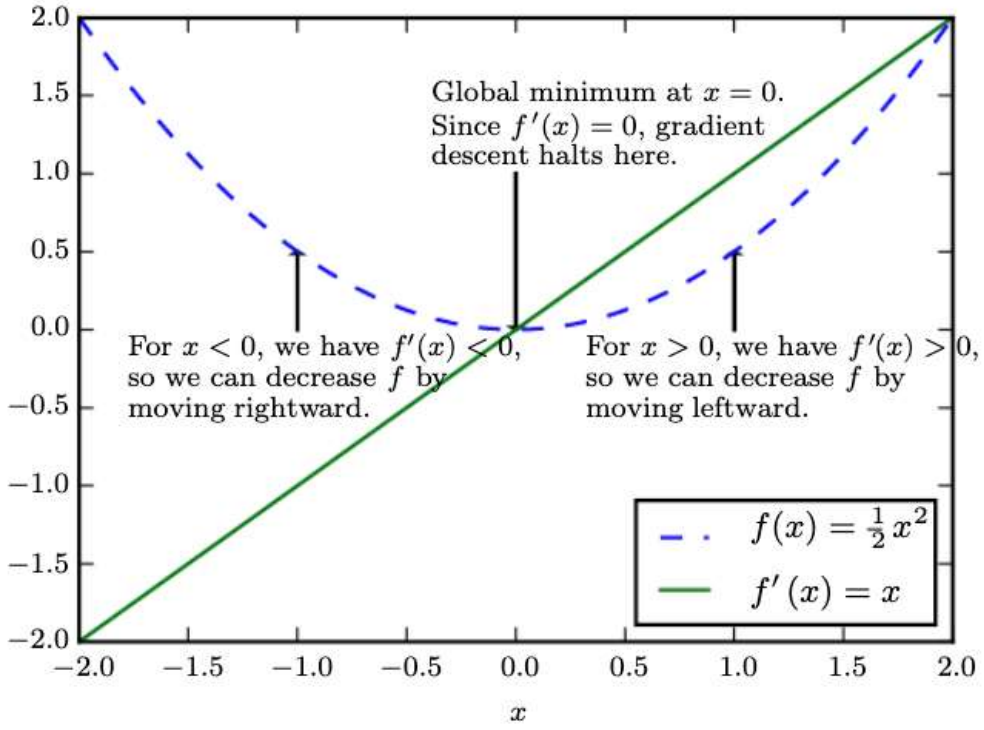
Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Optimization

- Follow the slope

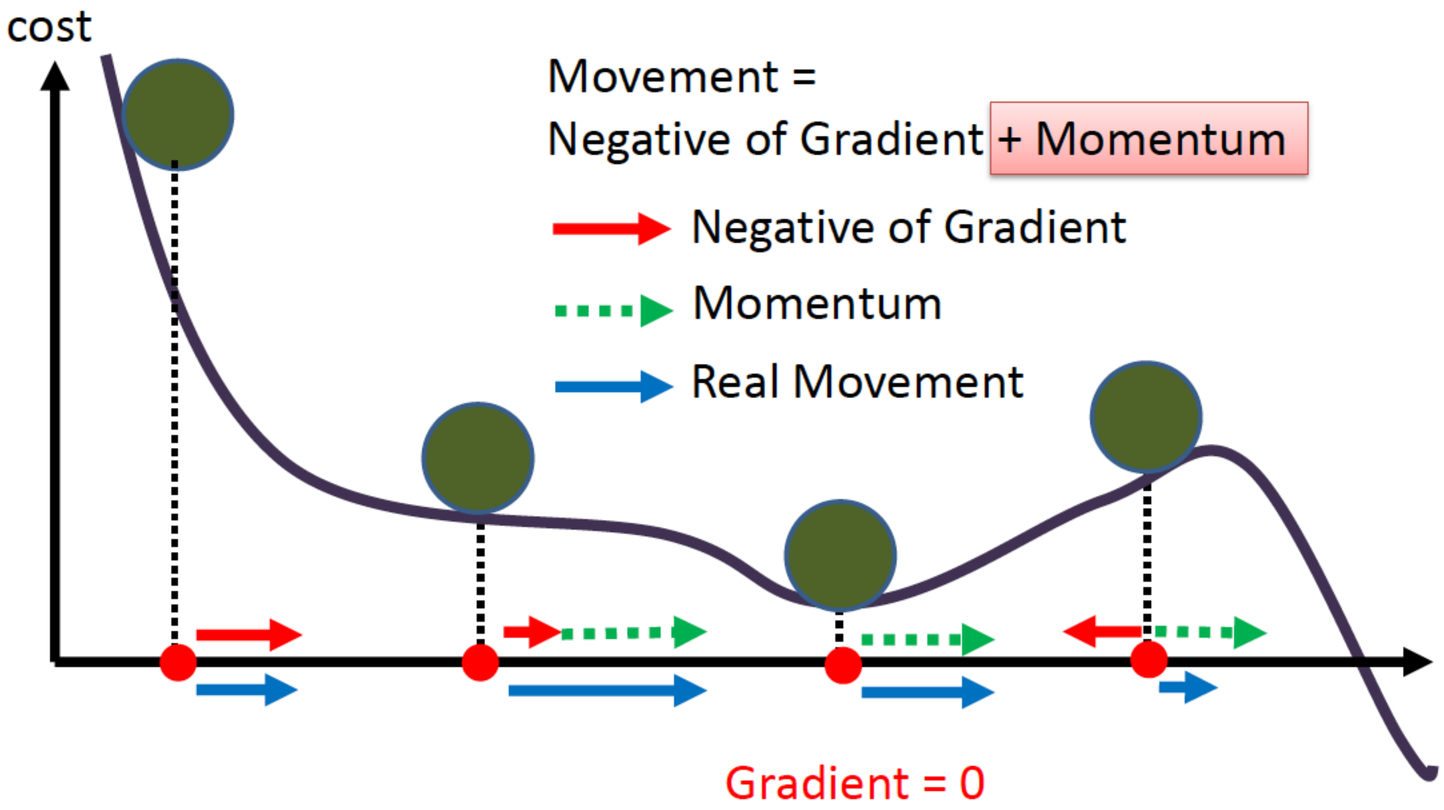
Optimization

- Method to minimize the ***cost function*** by updating weights
- Gradient descent:
  - Iteratively moving in the direction of steepest descent as defined by the negative of the gradient
- Stochastic gradient descent (SGD)
  - To handle large training sets
  - Only run a subset of the training sets (i.e., batch/minibatch) for each update
  - Easier to converge



Optimization

- Momentum:
  - Prefers to go in a similar direction as before



SGD

$$x_{t+1} = x_t - \alpha \nabla f(x_t)$$

```
while True:
  dx = compute_gradient(x)
  x -= learning_rate * dx
```

SGD+Momentum

$$v_{t+1} = \rho v_t + \nabla f(x_t)$$
$$x_{t+1} = x_t - \alpha v_{t+1}$$

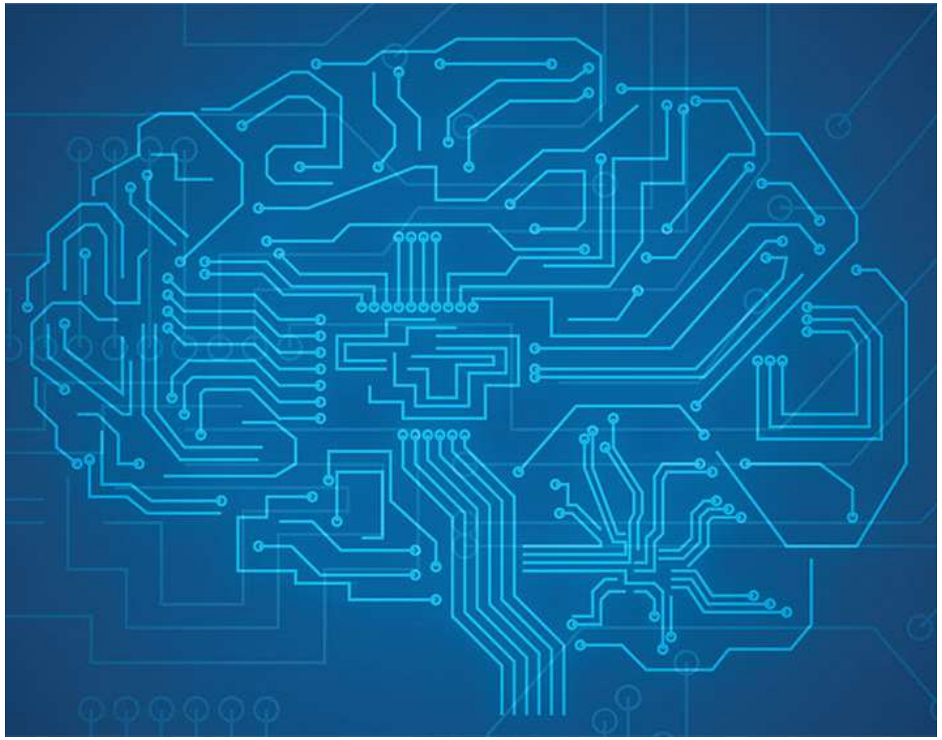
```
vx = 0
while True:
  dx = compute_gradient(x)
  vx = rho * vx + dx
  x -= learning_rate * vx
```

Challenges motivating DL

- The curse of dimensionality
  - Generalizing to new examples become exponentially more difficult when working with high-dimensional data
  - Challenging to learn complicated functions in high-dimensional spaces

Administrivia

- Lab 1 posted!
  - Due in two weeks (2/5)
- Reading for this week is posted.
  - Submit your review by Wednesday
- Joint project is OK.

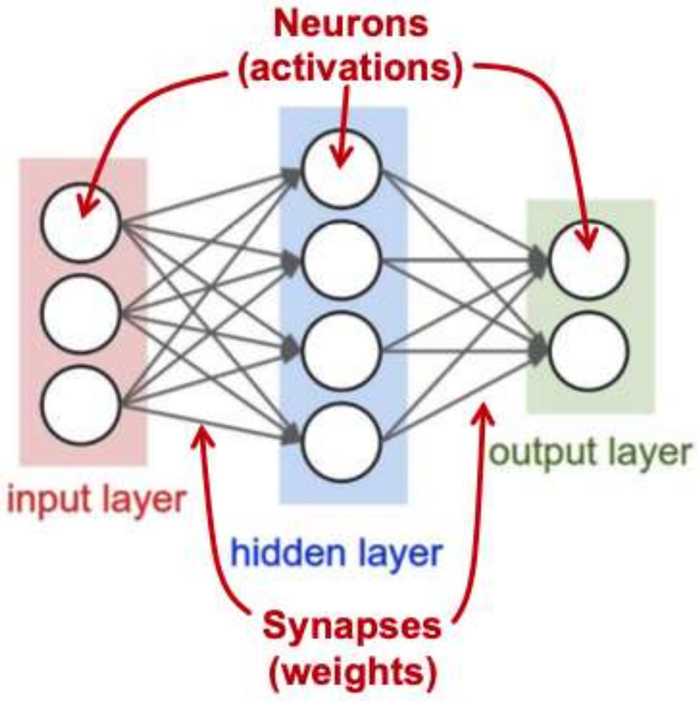


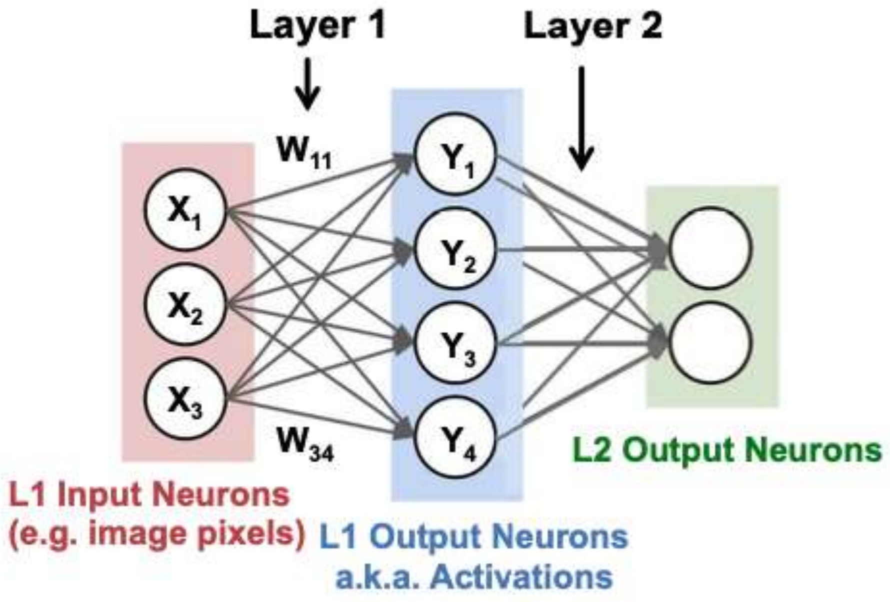
DNNs

- AI, ML, and DL
- ML Basics
- DL Overview
- AlexNet Example

Deep Neural Networks

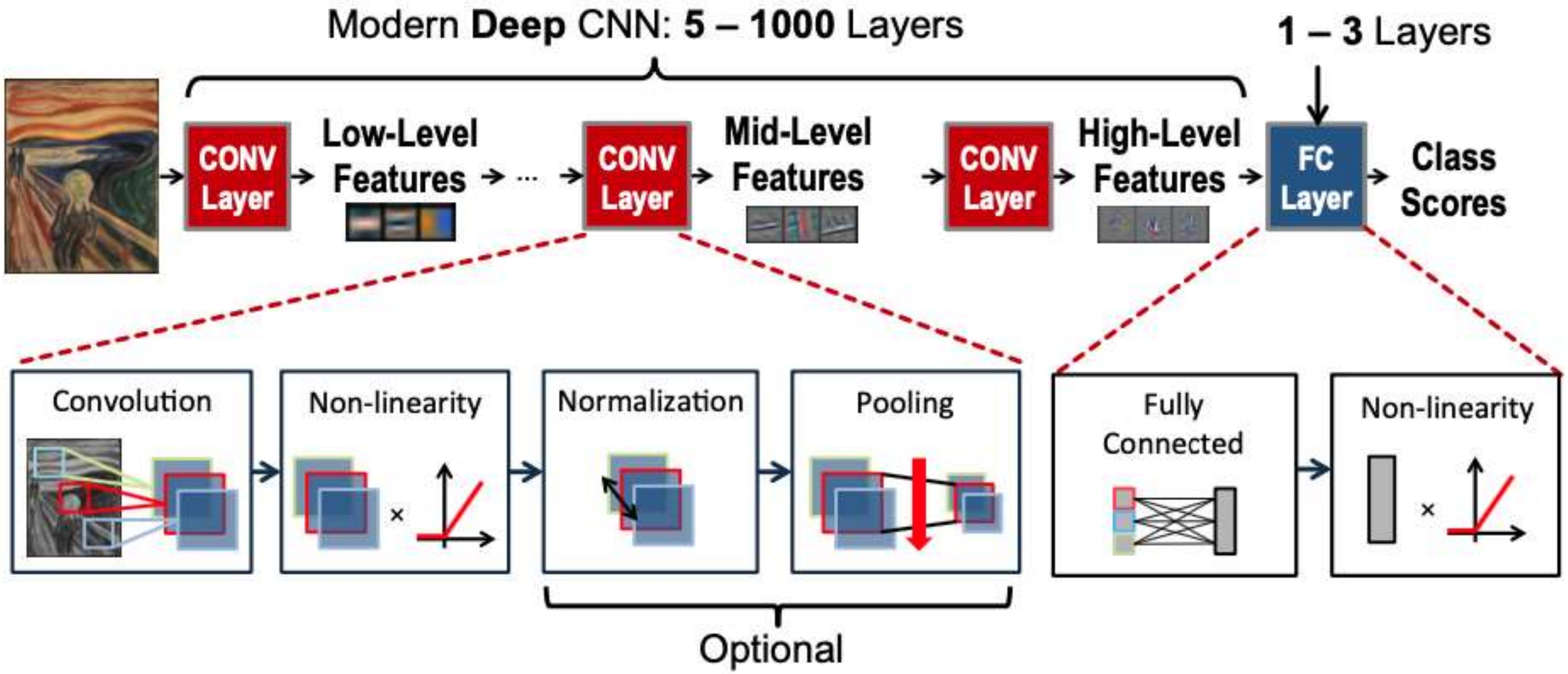
- A subset of machine learning models
  - Deep -> multiple layers
  - Linear model + nonlinear transformation





Eyeriss tutorial

Deep Neural Networks



Eyeriss tutorial

Deep Neural Networks: LeNet

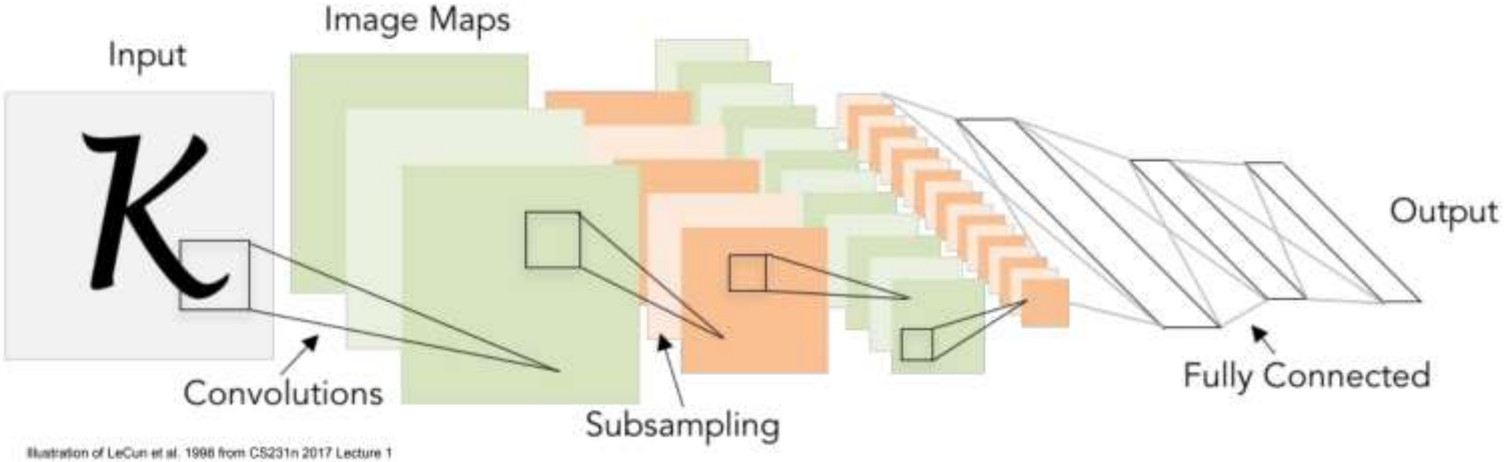
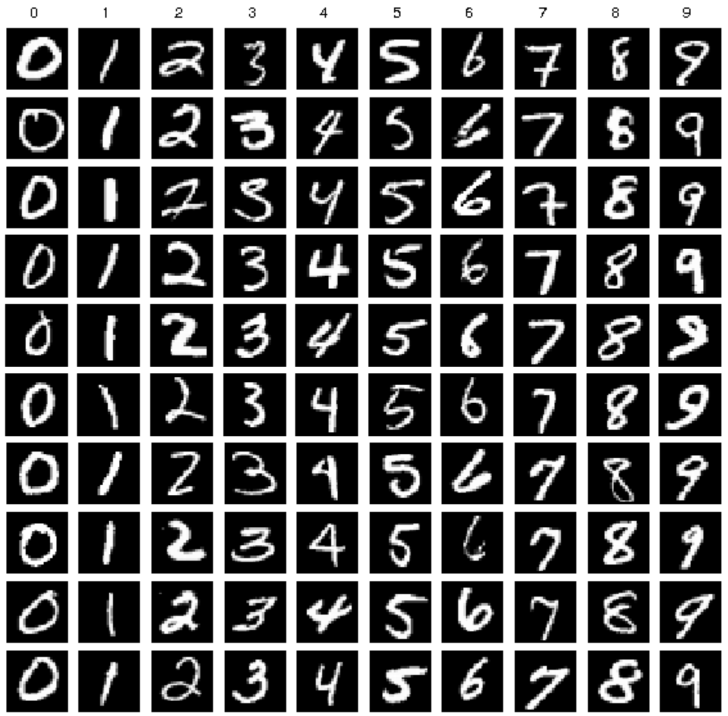


Illustration of LeCun et al. 1998 from CS231n 2017 Lecture 1

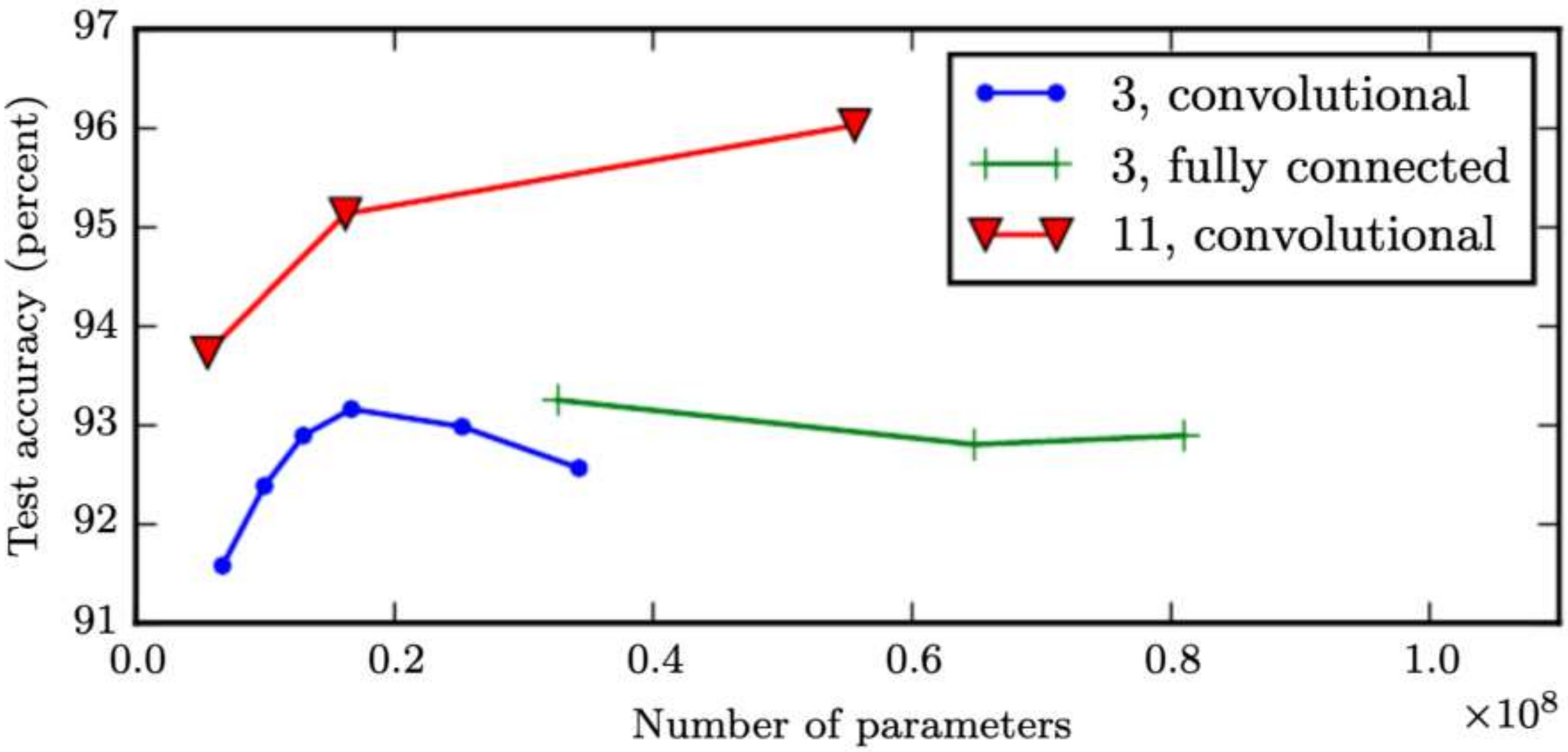


Deep Neural Networks

- Dataset
  - Problem dependent
  - https://github.com/pytorch/vision
  - Training set, validation set, test set
- Cost function
  - Similar to other parametric ML models
  - Backpropagation (using chain rule)
- Optimization
  - Stochastic Gradient Descent

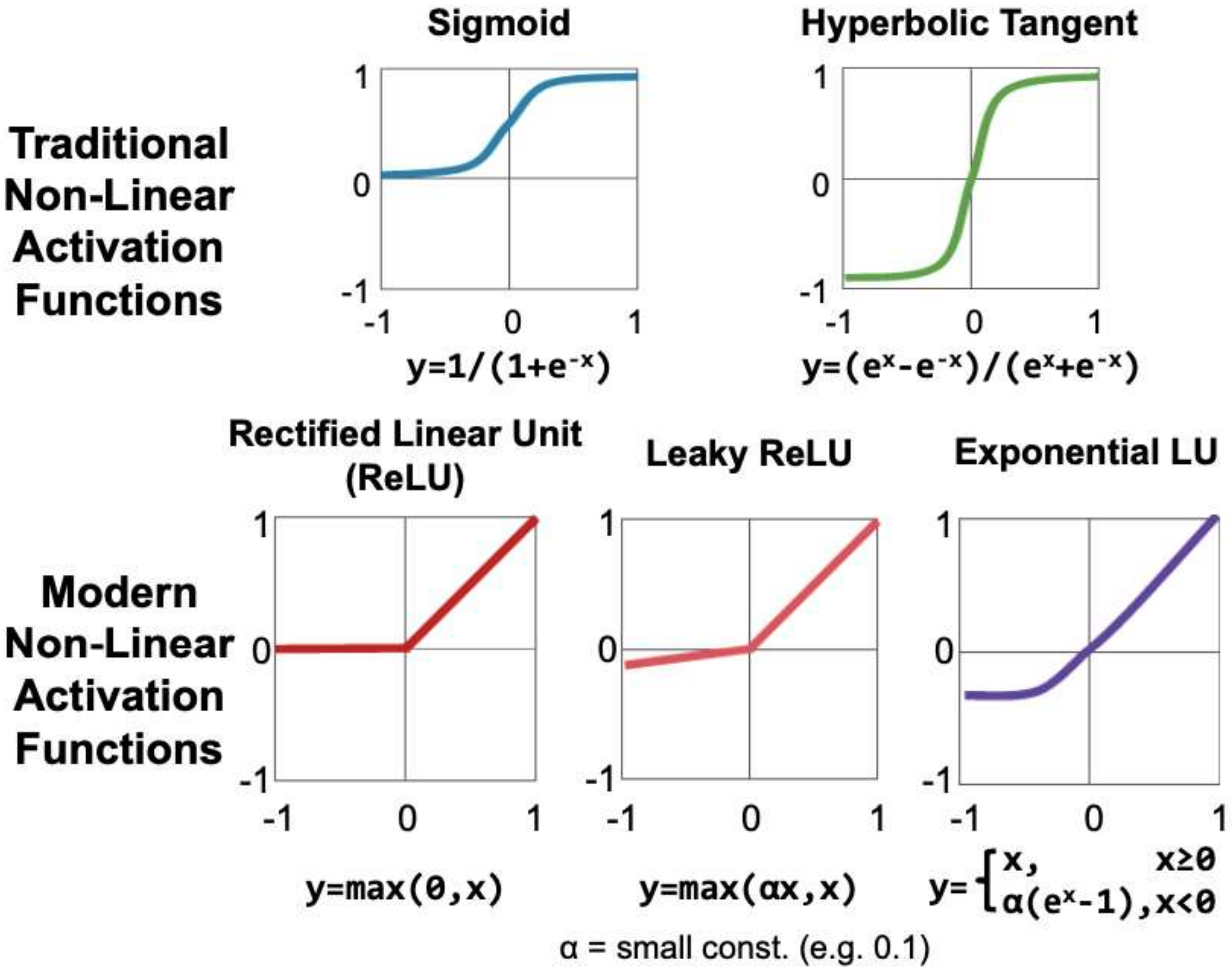
DNN Parameters

- Depth and width of DNNs



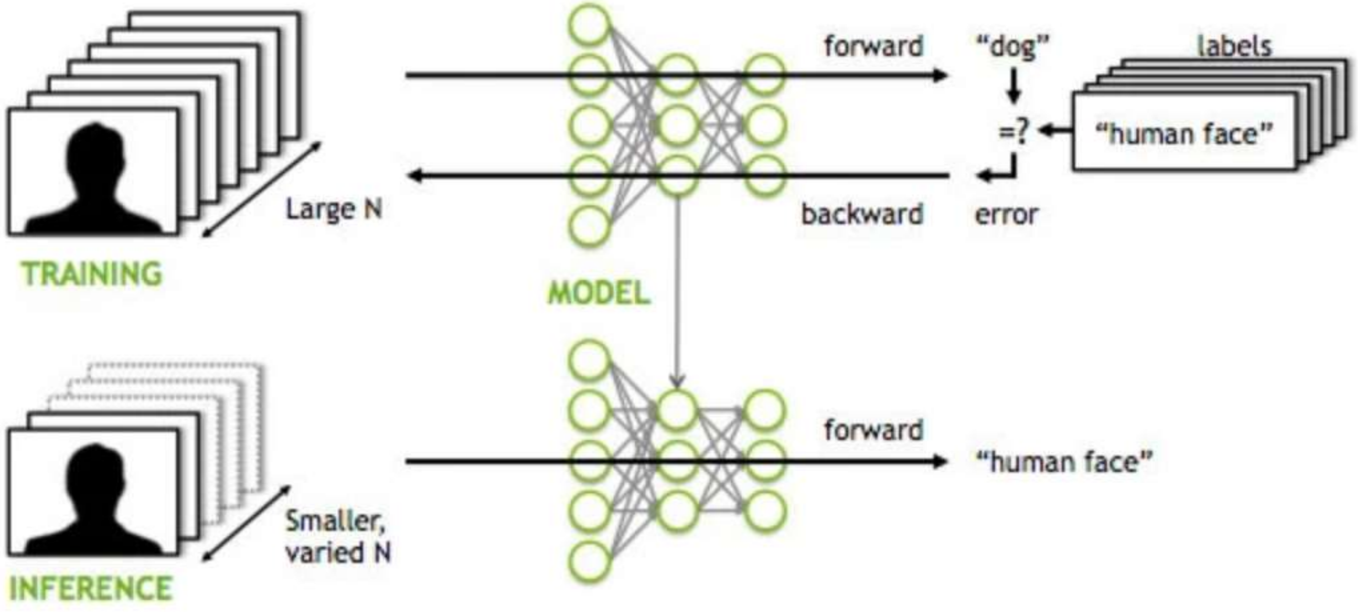
DNN Non-Linearity

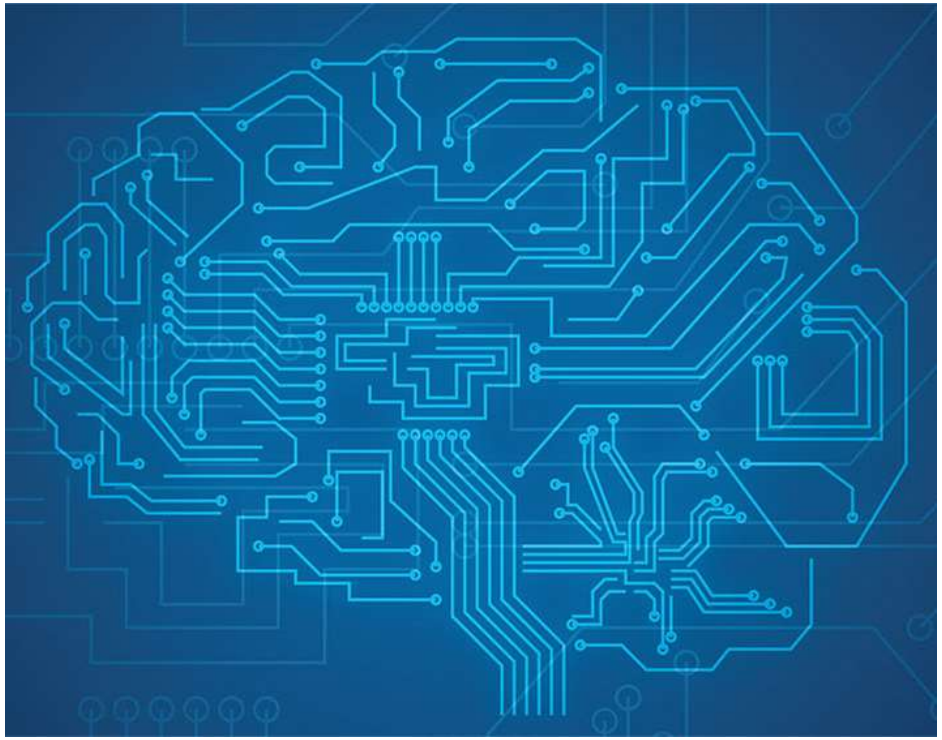
- Rectified Linear Units (ReLU)
  - $y(x) = \max\{0, x\}$
- Benefits:
  - Reduce the likelihood of the gradient vanishing problem.
  - Adding more sparsity/regularization
  - Easy to compute



Training vs Inference

- Training
  - Dataset
  - Cost function
  - Optimization function
  - Model
- Inference
  - Dataset
  - Model





DNNs

- AI, ML, and DL
- ML Basics
- DL Overview
- **AlexNet Example**

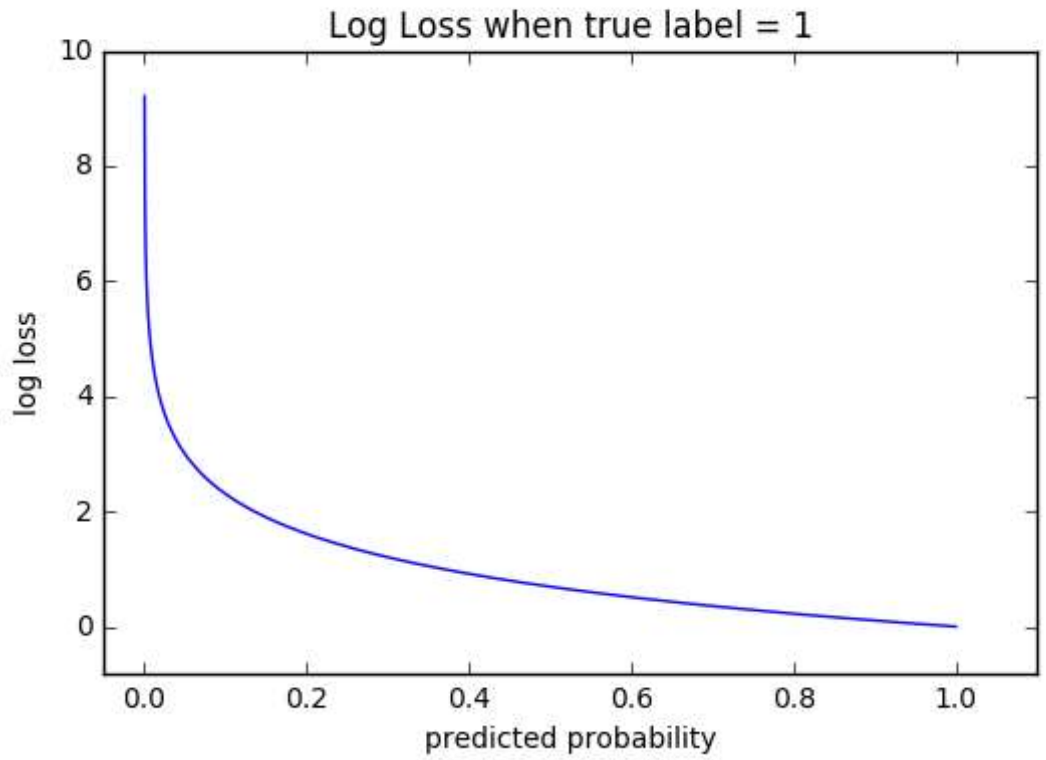
AlexNet Dataset

- ImageNet LSVRC-2010 with 1.2 million images
- Top-1 and top-5 error rate:
  - The fraction of images for which the correct label is not among the 1/5 labels considered most probable by the model.

[A. Krizhevsky, I. Sutskever, G.E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, 2012]

AlexNet Cost (loss) function

- Minimize the cross-entropy loss function
- Measures the performance of a classification model whose output is a probability value between 0 and 1
- $CF = -\frac{1}{N}(\sum_{i-1}^{N} y_i \cdot \log(\hat{y}_i))$



AlexNet Optimization Method

- Stochastic Gradient Descent
  - Batch size = 128
  - Update rule:

Momentum

Weight Decay

$$
\begin{aligned}
v_{i+1} &:= 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot w_i - \epsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i} \\
w_{i+1} &:= w_i + v_{i+1}
\end{aligned}
$$

Learning rate

Gradient of cost function

AlexNet Model

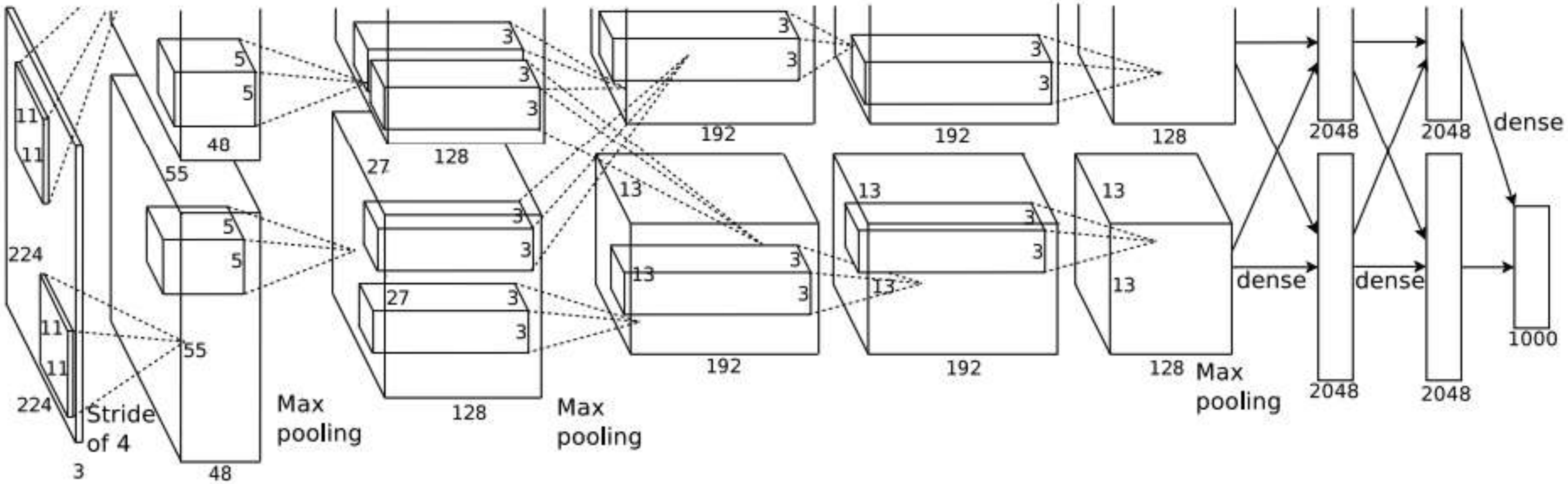


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

State-of-the-art Training Time

	Batch Size	Processor	DL Library	Time	Accuracy
He et al.	256	Tesla P100 x8	Caffe	29 hours	75.30%
Goyal et al.	8K	Tesla P100 x256	Caffe2	1 hour	76.30%
Smith et al.	8K→16K	full TPU Pod	TensorFlow	30 mins	76.10%
Akiba et al.	32K	Tesla P100 x1024	Chainer	15 mins	74.90%
Jia et al.	64K	Tesla P40 x2048	TensorFlow	6.6 mins	75.80%
This work	**34K→68K**	**Tesla V100 x2176**	**NNL**	**224 secs**	**75.03%**

https://news.developer.nvidia.com/sony-breaks-resnet-50-training-record-with-nvidia-v100-tensor-core-gpus/

Review

- Artificial intelligence, machine learning, and deep learning
- Building a machine learning algorithm:
  - Dataset
  - Cost function
  - Optimization function
  - Model
- Deep learning to automatically extract hierarchical data
  - Better at handle high-dimensional data
  - Key differences are in the Model
    - Multiple layers, i.e., deep
    - Linear + non-linear layers