Introduction to Deep Neural Networks

Neo Christopher Chung

Lecture 10, 1000-719bMSB

Why Deep Learning?

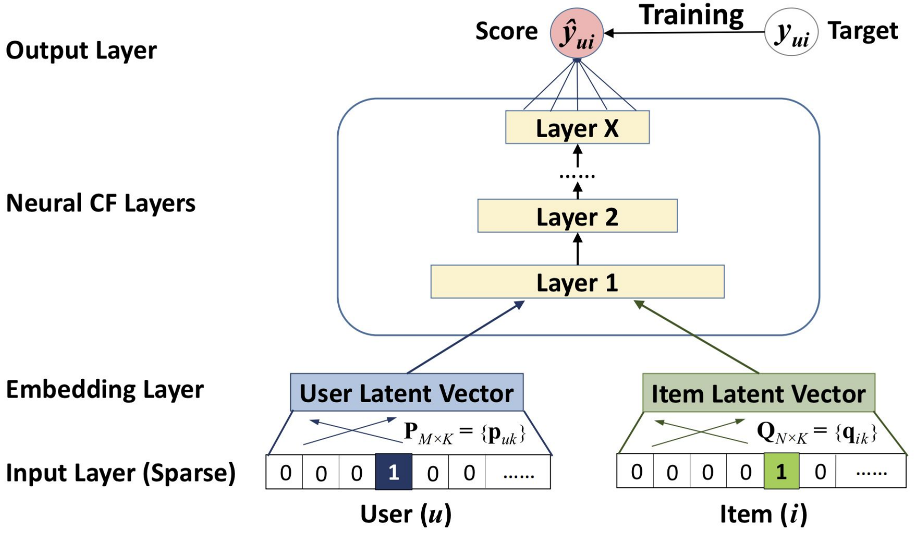
- Image recognition: handwritten digits, ImageNet (1.2 mio images, 1000 classes) Krizhevsky,Sutskever, Hinton (2012)

Model	**Top-1 (val)**	**Top-5 (val)**	**Top-5 (test)**
SIFT + FVs [7]	—	—	*26.2%*
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

Table 2: Comparison of error rates on ILSVRC-2012 validation and test sets. In *italics* are best results achieved by others. Models with an asterisk* were "pre-trained" to classify the entire ImageNet 2011 Fall release. See Section 6 for details.

Why Deep Learning?

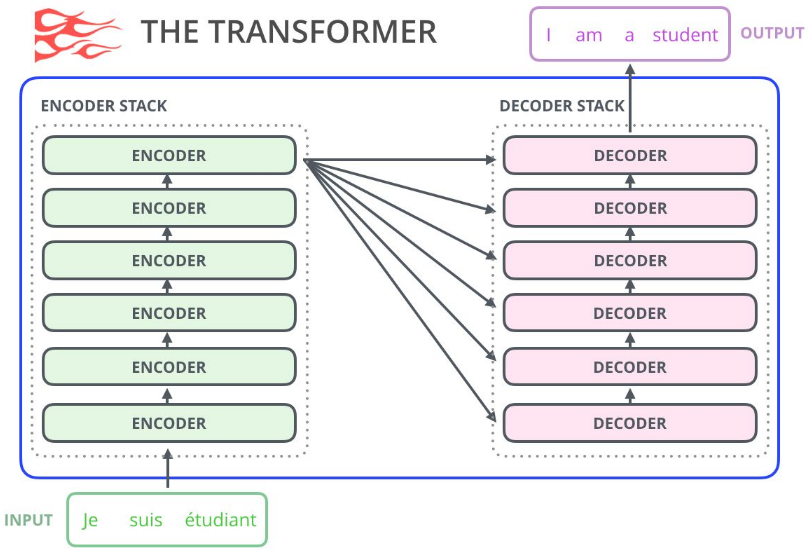
- Recommendation system



Neural collaborative filtering framework (He et al. 2017)

Why Deep Learning?

- Natural Language Processing



Why Deep Learning?

Style transfer

Super resolution

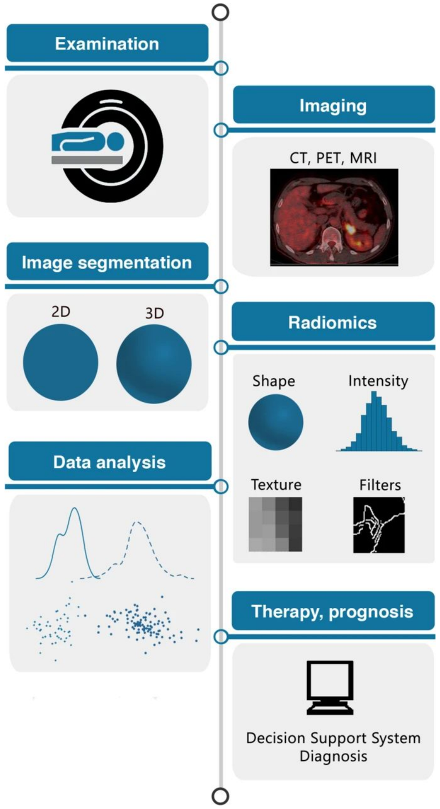
Deep Learning in Biology and Medicine

- Lots of challenges -- is it simply a fad?
- Learning from 50+ years of failures and successes
- Interpretability is important

e.g., Drug discovery, targets vs. off-targets

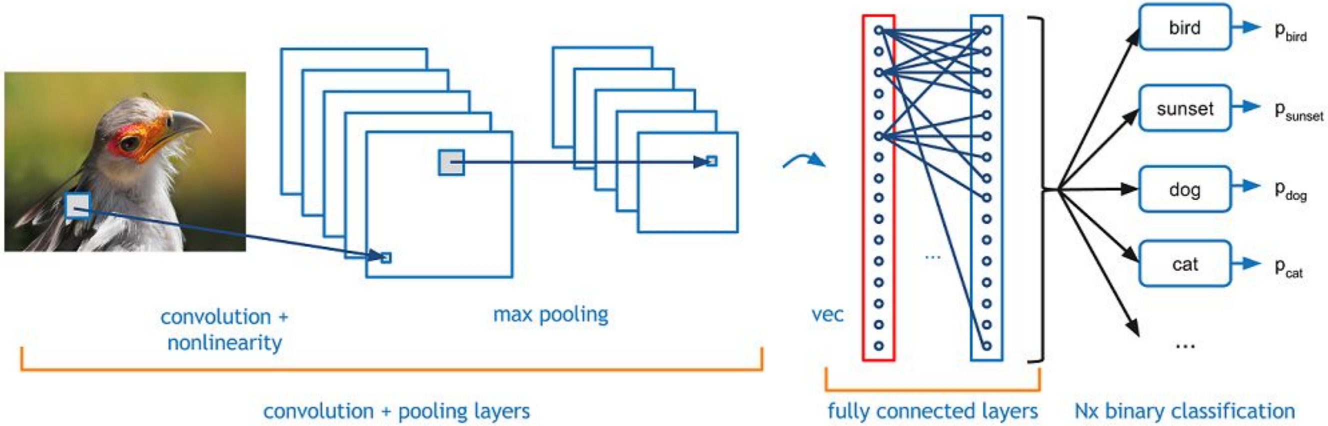
Toxic effects of biochemical or biologics

Predict cancer using medical images



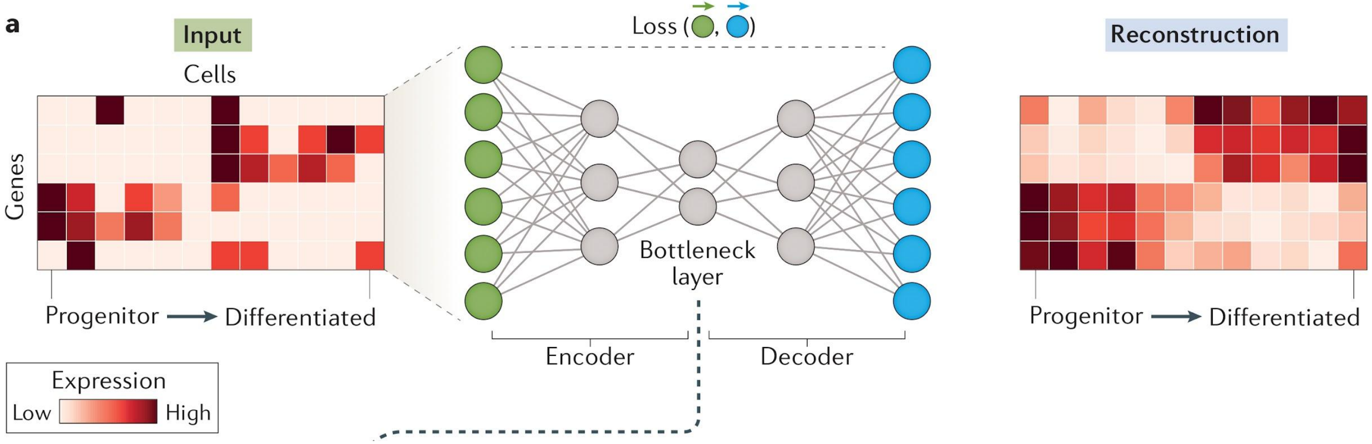
Supervised learning

- Outputs (labels) are given for input data
- Learn a mapping function between input and labels
- Most popular use of deep learning and machine learning generally
- Focus of this week



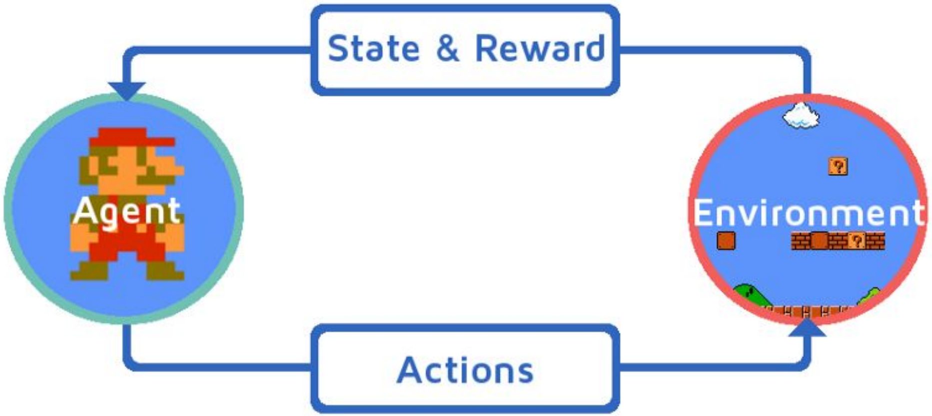
Unsupervised learning

- Labels are not available or not used
- Discover patterns or internal/compact representation
- Identify the latent space or latent variables underlying the data
- Focus of next week



Reinforcement Learning

- Agent in an environment
- Learn to maximize reward
- Chess, Go, Starcraft, etc
- Self-driving cars, robotics, etc



Self-supervised learning

- Labels generated from input data
- Predict/generate next words
- Predict/generate next frames in video

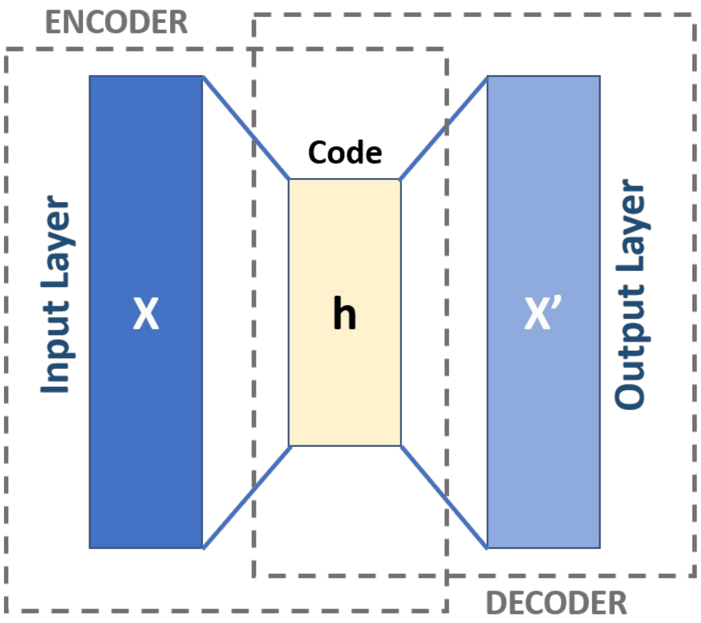
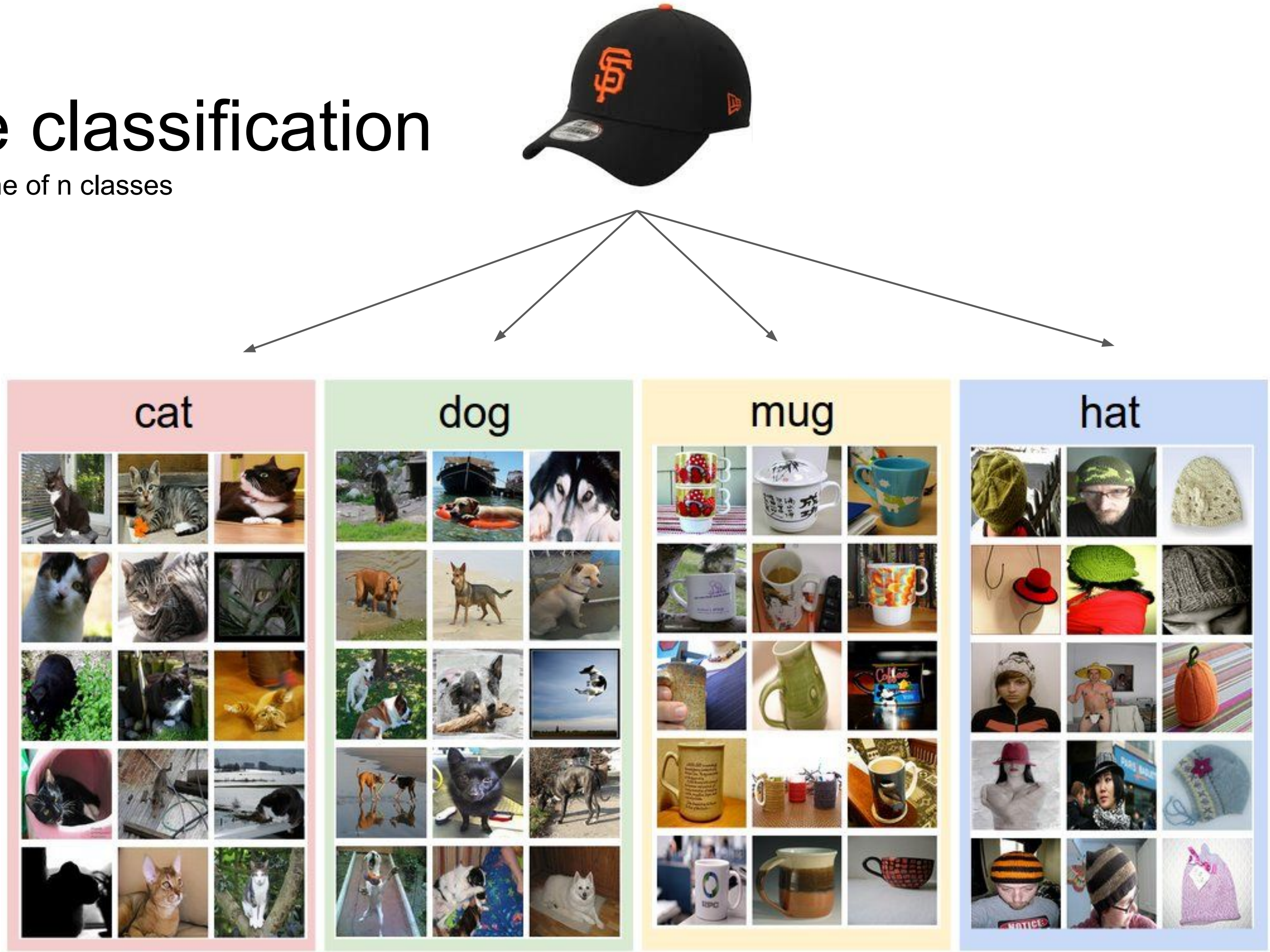


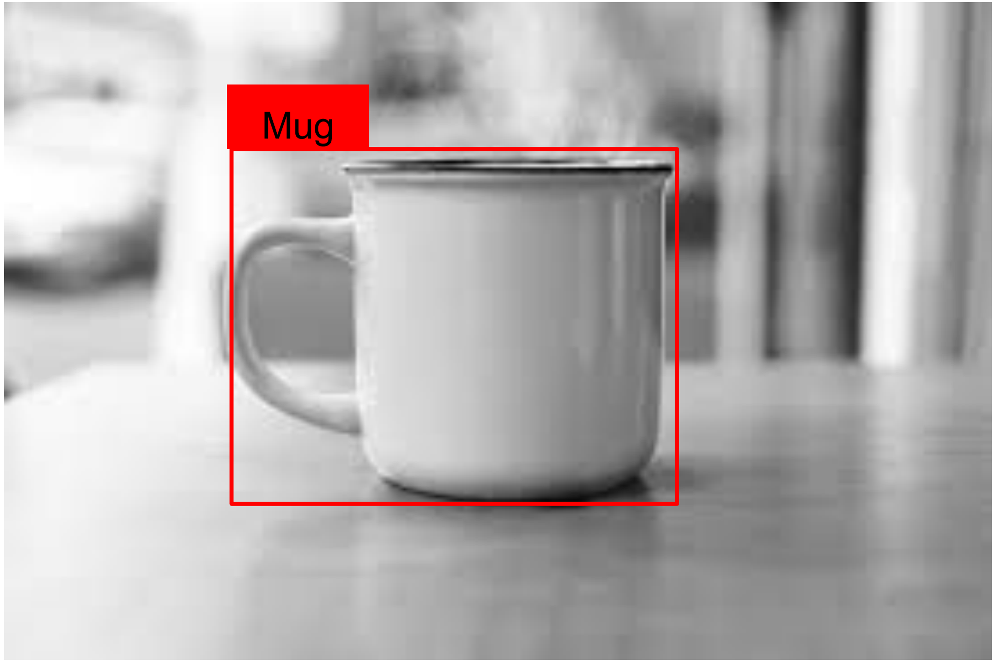
Image classification

Classify into one of n classes



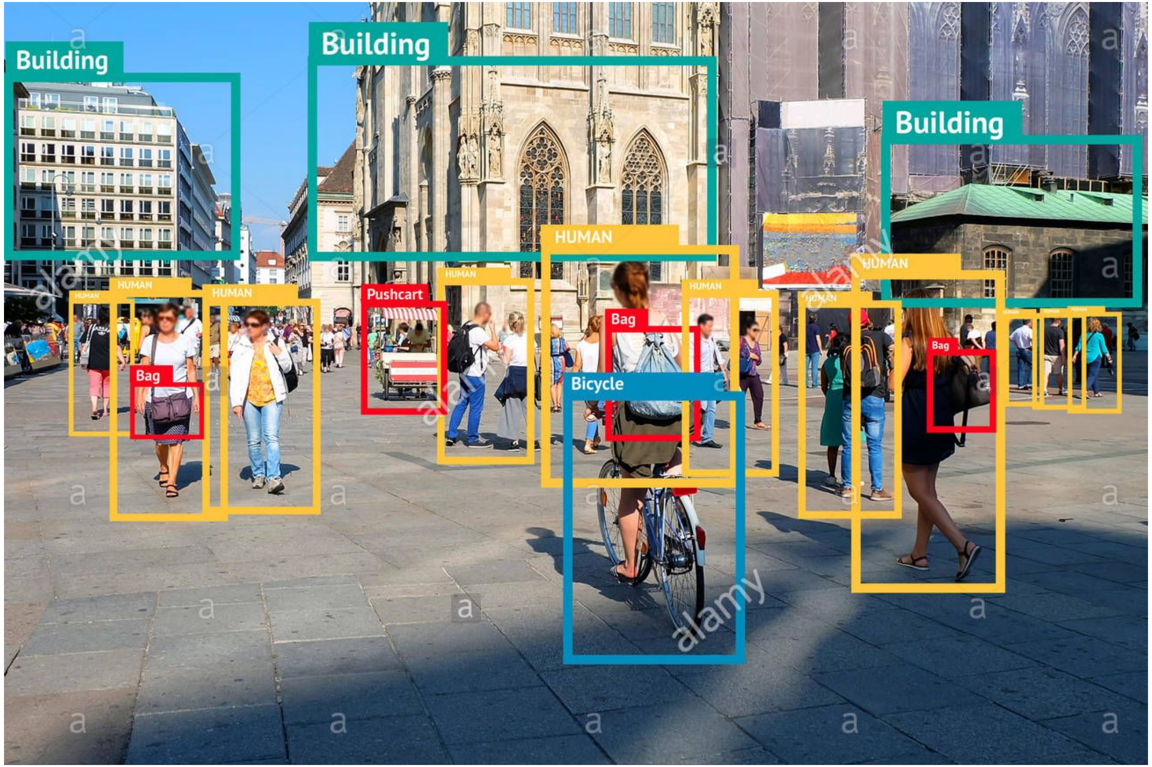
Object localization

categorizing and locating an object in position and size using a bounding box



Object Detection

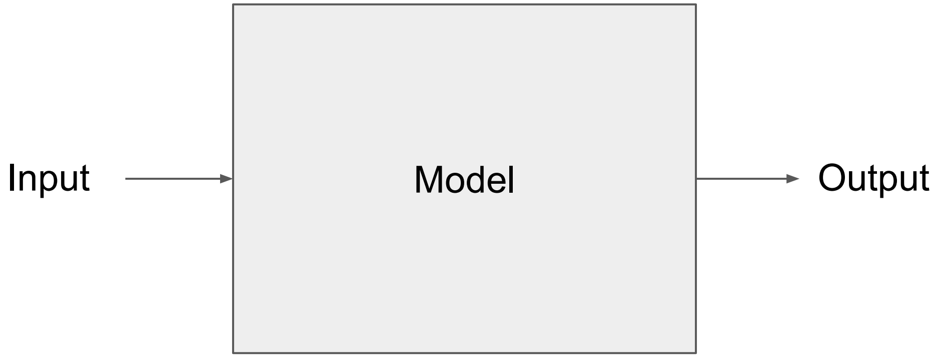
Identify the object category and locate the position using a bounding box for every known object within an image



Semantic segmentation

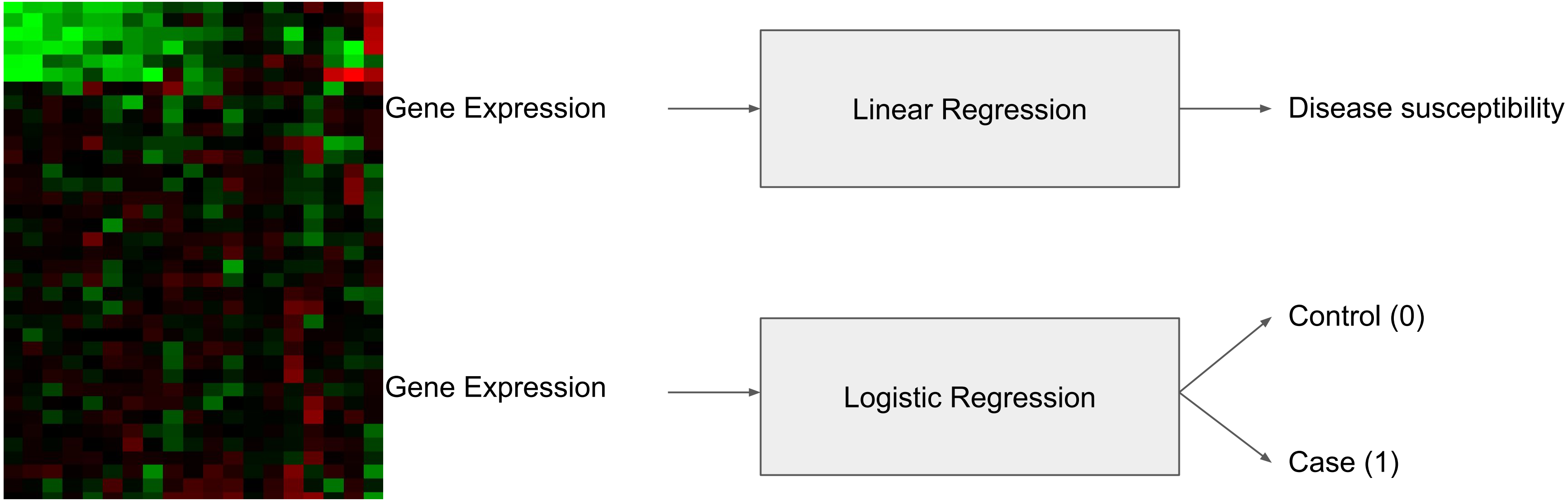
Identify the object category of each pixel for every known object within an image.
Labels are class-aware.

http://www.cs.toronto.edu/~tingwuwang/semantic_segmentation.pdf

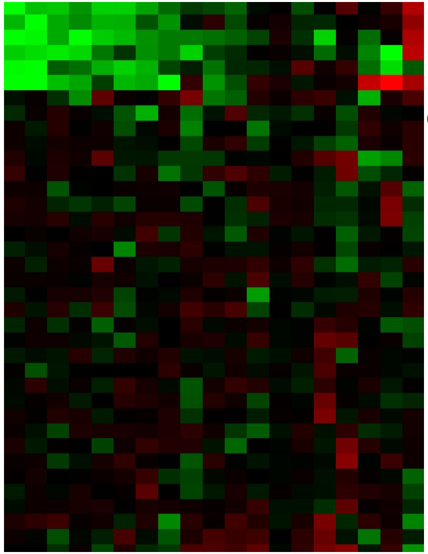

Input
Model
Output

White box models

how a certain inference/prediction is made is clear and explainable



As the number of variables grows



Gene Expression → Linear Regression → Disease susceptibility

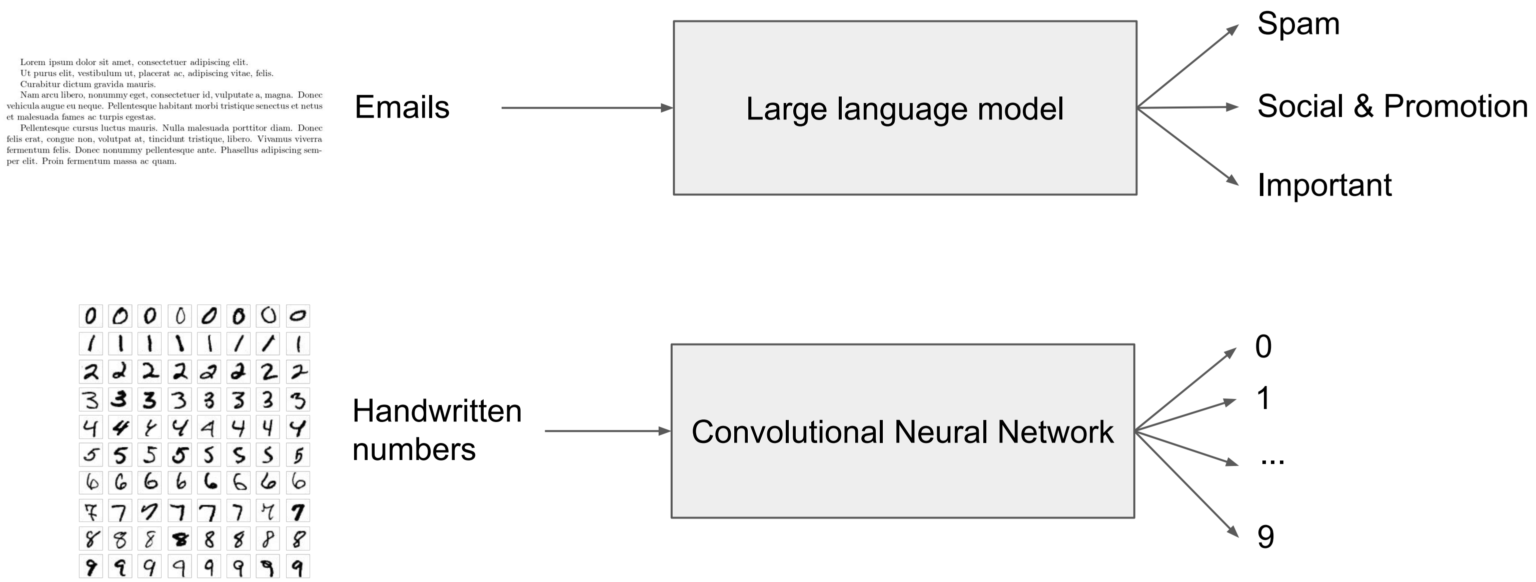
Even a simple linear regression may result in a **<u>massive number of predictors.</u>**

Then, we may use feature selection as a preprocessing step.

Alternatively, we employ sparse models, like the Lasso

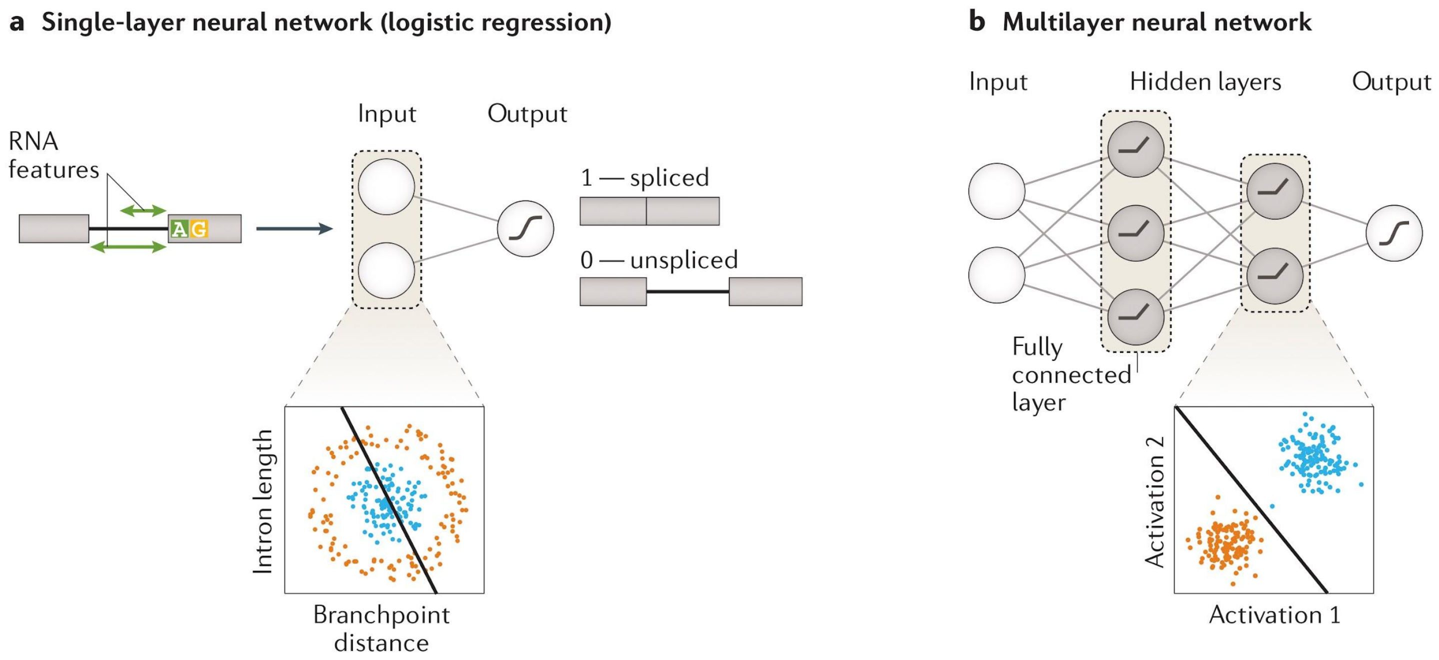
Black box models

Nearly impossible to understand why a prediction is made



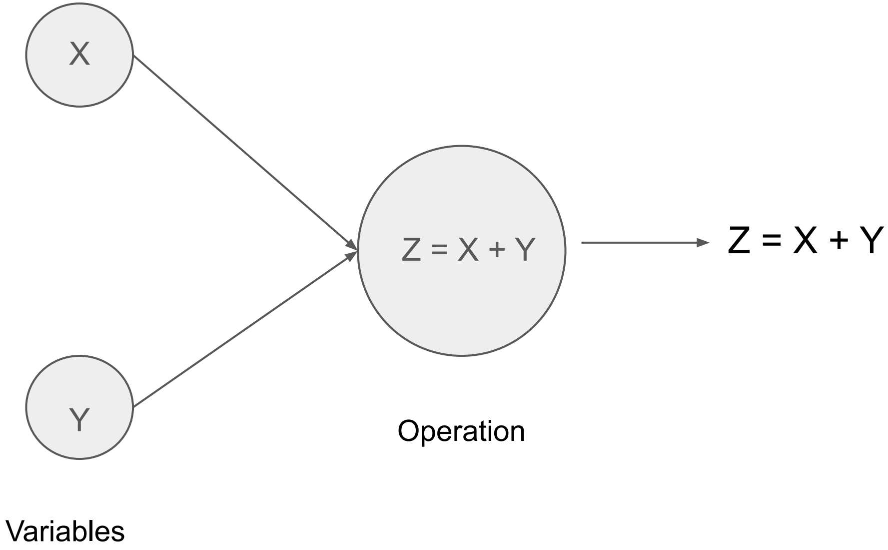
Due to # predictors, # parameters and non-linearity

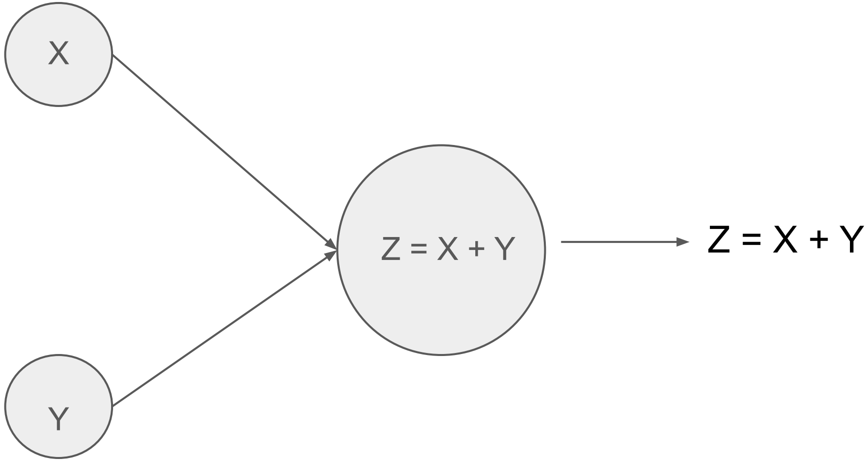
Deep Neural Network



Computational Graph: directed graph, w/ nodes are operations or variables

Variables

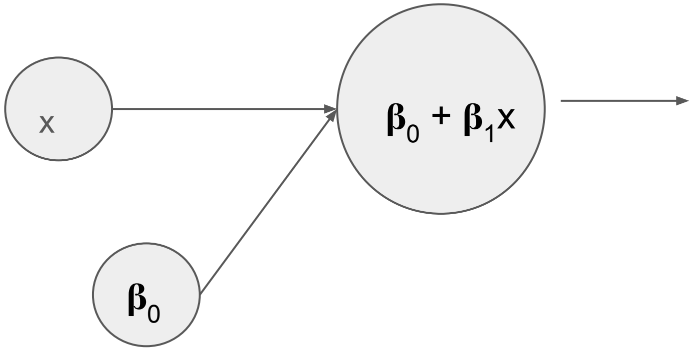

X
Y
Z = X + Y
Z = X + Y
Operation
Variables



Tensors = Inputs and outputs of nodes = (multi-dimensional) arrays

Simple Linear Regression

Consider an influence of 1 gene's expression on a disease susceptibility

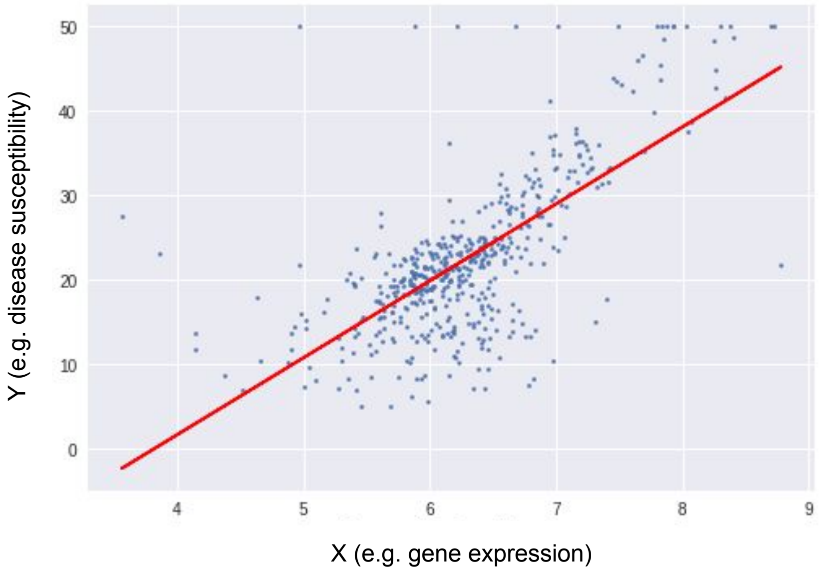


$y = \beta_0 + \beta_1 x$

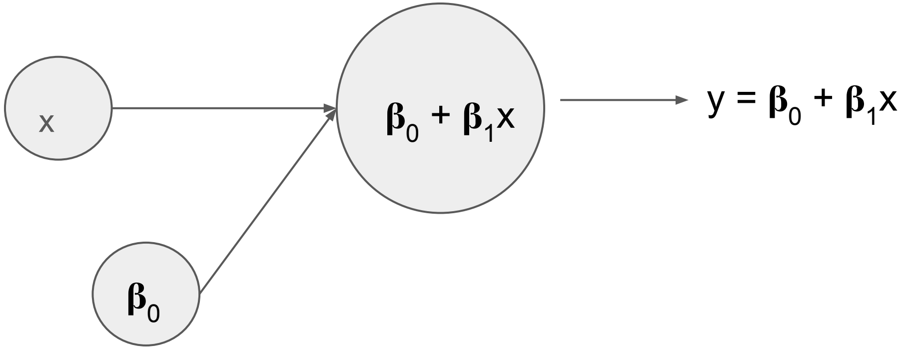
In deep learning, this is called a bias

Linear Regression

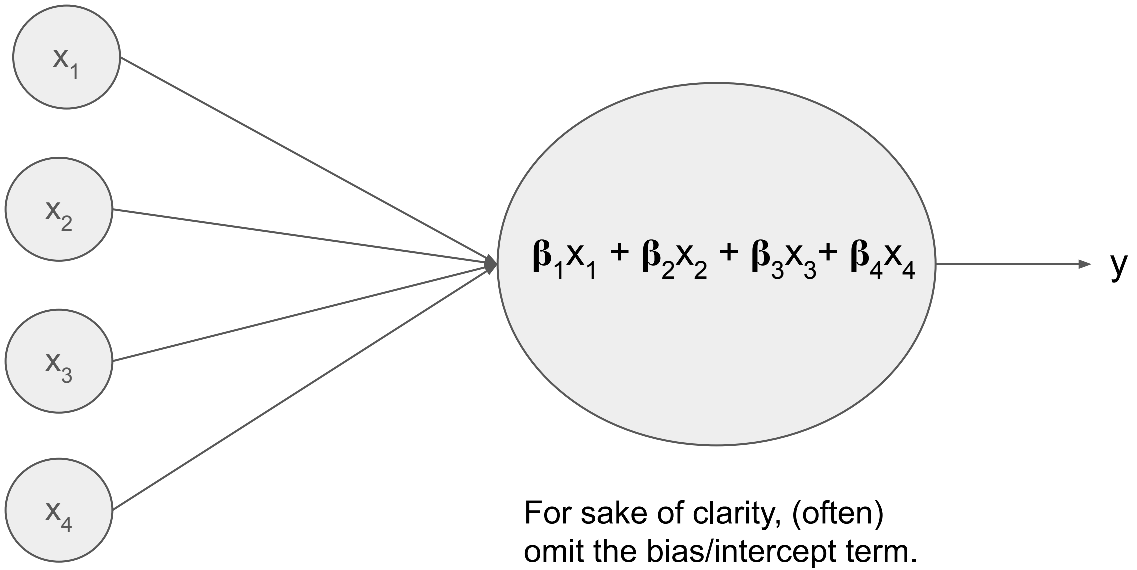
Prediction: $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$



Computational Graph



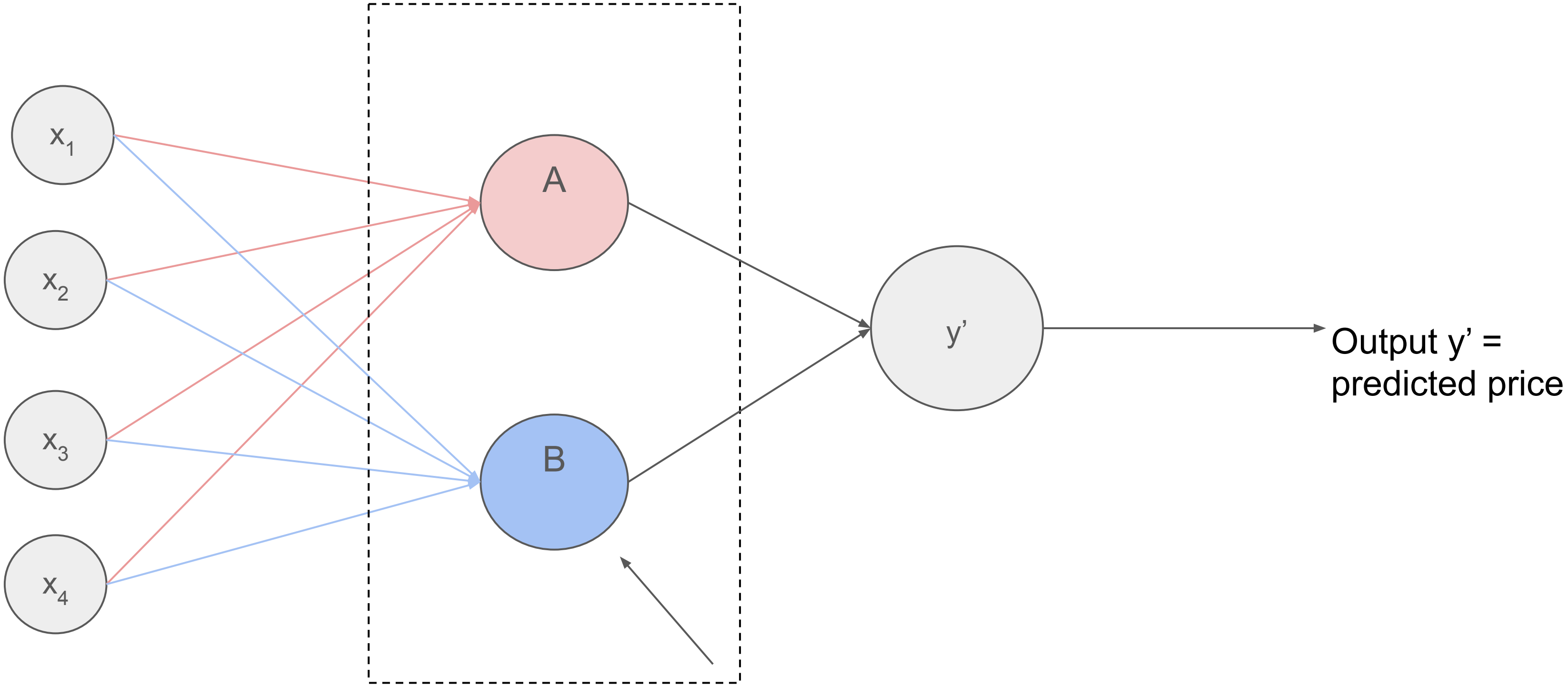
Multiple Linear Regression



For sake of clarity, (often) omit the bias/intercept term.

Consider 4 genes!

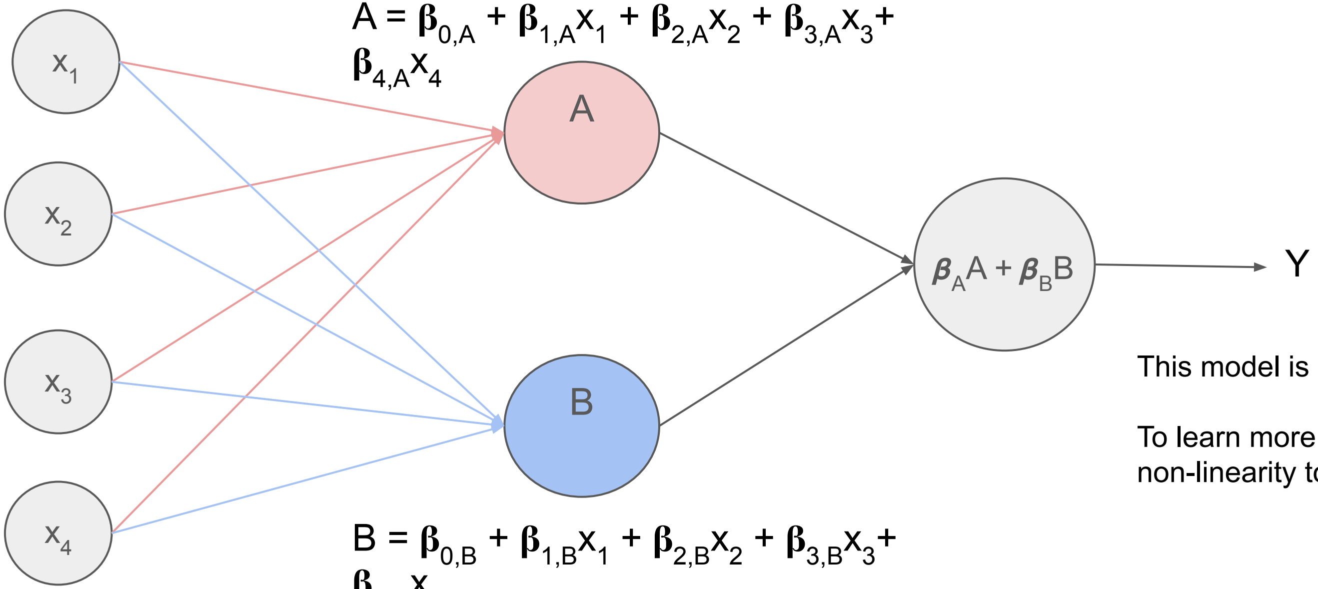
Hidden Layer



Intermediate layer of operations

combines x_i into a set of intermediate features, followed by combining again into the final node

$$A = \beta_{0,A} + \beta_{1,A}x_1 + \beta_{2,A}x_2 + \beta_{3,A}x_3 + \beta_{4,A}x_4$$



$$B = \beta_{0,B} + \beta_{1,B}x_1 + \beta_{2,B}x_2 + \beta_{3,B}x_3 + \beta_{4,B}x_4$$

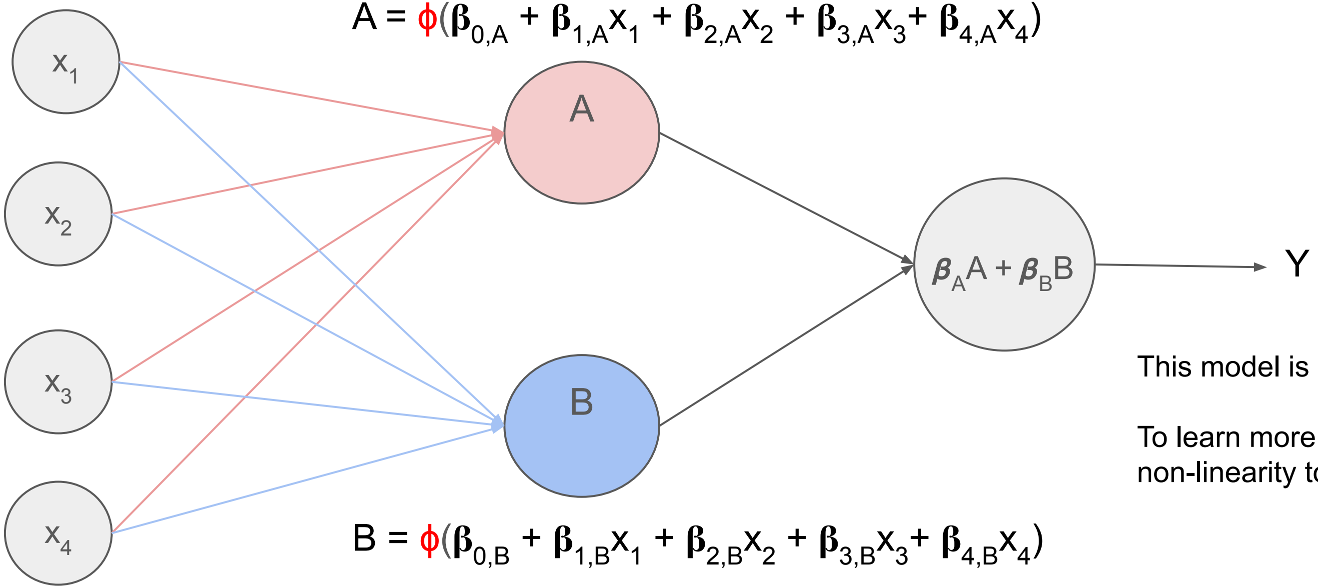
This model is linear.

To learn more complex relationship, we add non-linearity to this network

Non-linearity

Apply an activation function ϕ at the output of each node. E.g., Sigmoid function, Rectified Linear Unit (ReLU), etc

$$A = \phi(\beta_{0,A} + \beta_{1,A}x_1 + \beta_{2,A}x_2 + \beta_{3,A}x_3 + \beta_{4,A}x_4)$$



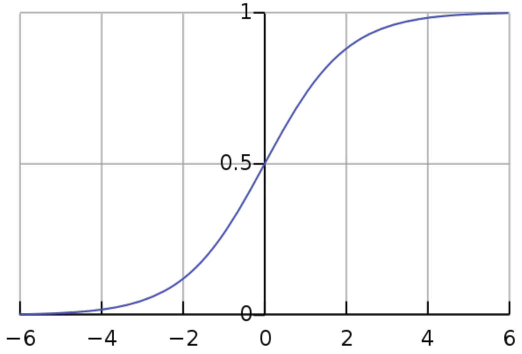
$$B = \phi(\beta_{0,B} + \beta_{1,B}x_1 + \beta_{2,B}x_2 + \beta_{3,B}x_3 + \beta_{4,B}x_4)$$

This model is linear.

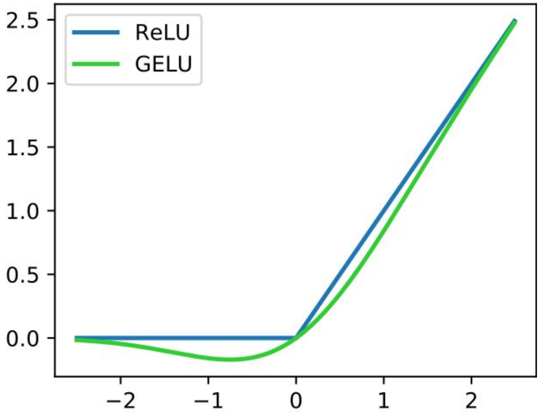
To learn more complex relationship, we add non-linearity to this network

Activation functions

Sigmoid function
$S(x) = 1 / (1+e^{-x})$



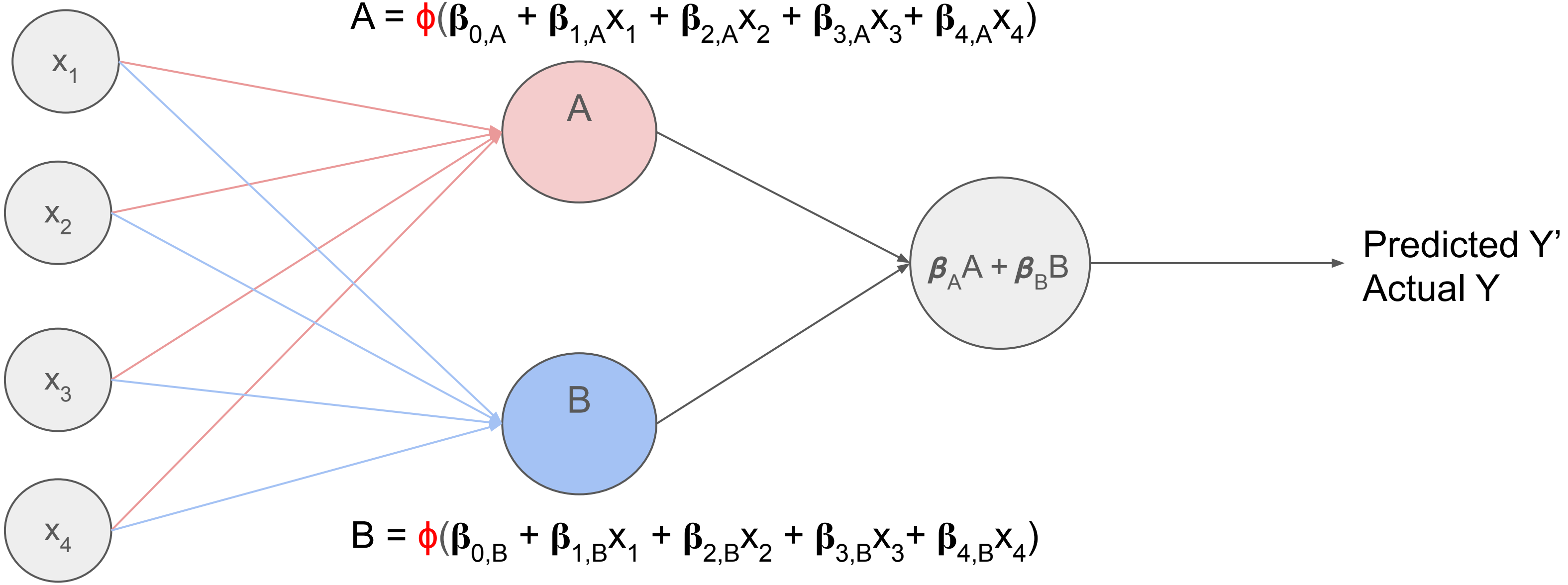
Rectified Linear Unit (ReLU)
$f(x) = \max(0,x)$



Wikipedia

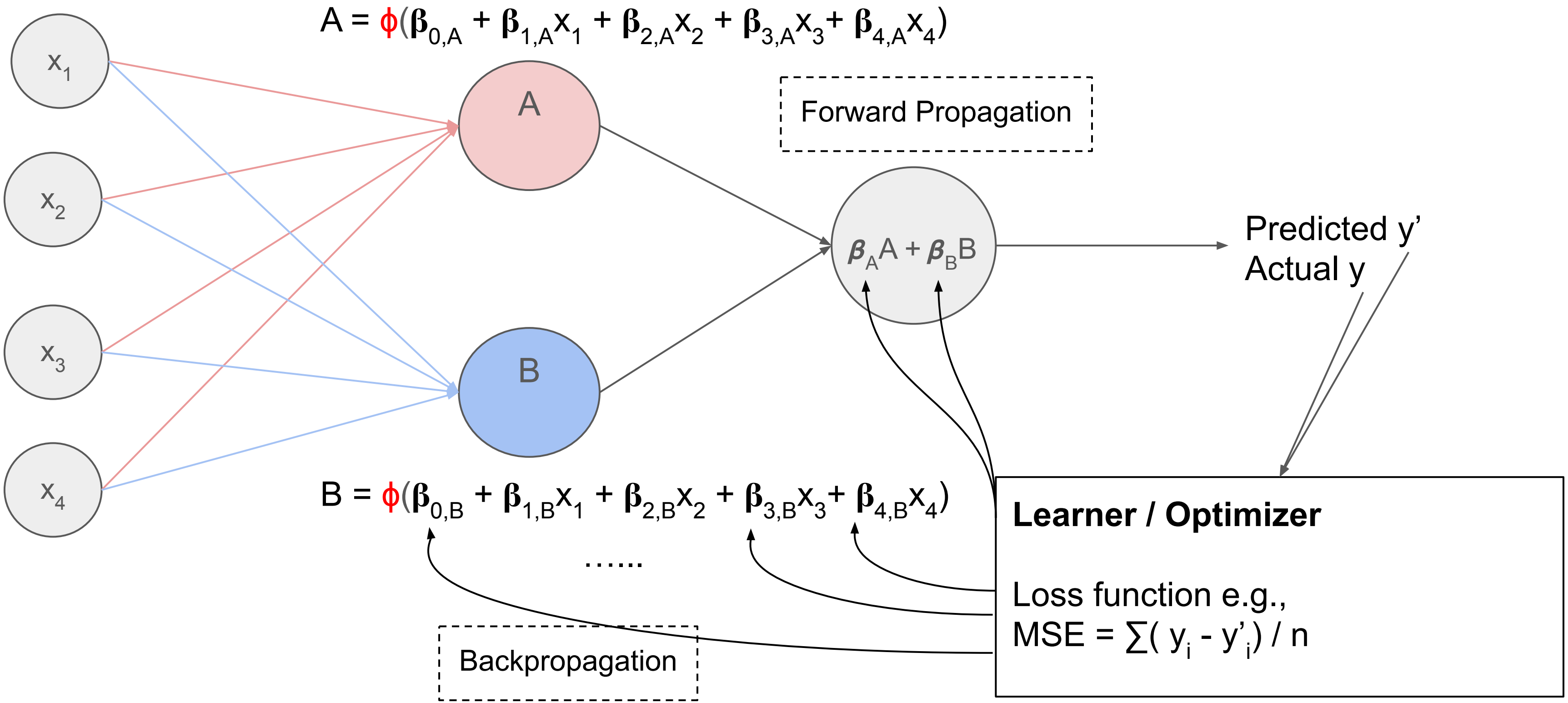
Training: learning weights

$$A = \phi(\beta_{0,A} + \beta_{1,A}x_1 + \beta_{2,A}x_2 + \beta_{3,A}x_3 + \beta_{4,A}x_4)$$



$$B = \phi(\beta_{0,B} + \beta_{1,B}x_1 + \beta_{2,B}x_2 + \beta_{3,B}x_3 + \beta_{4,B}x_4)$$

Training: learning weights

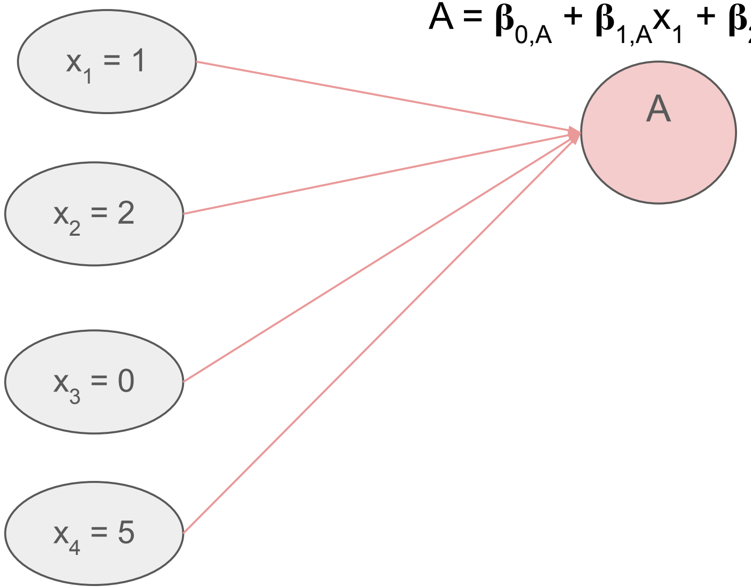


Forward propagation

Calculating the value for the chosen (or all) node.

Coefficients are known. Compute A, B, etc

$$A = \boldsymbol{\beta}_{0,A} + \boldsymbol{\beta}_{1,A}x_1 + \boldsymbol{\beta}_{2,A}x_2 + \boldsymbol{\beta}_{3,A}x_3 + \boldsymbol{\beta}_{4,A}x_4$$

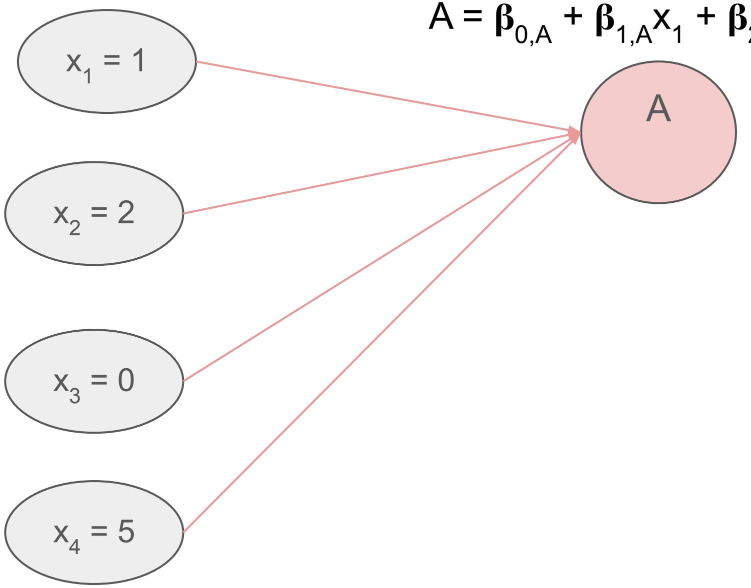


Backpropagation

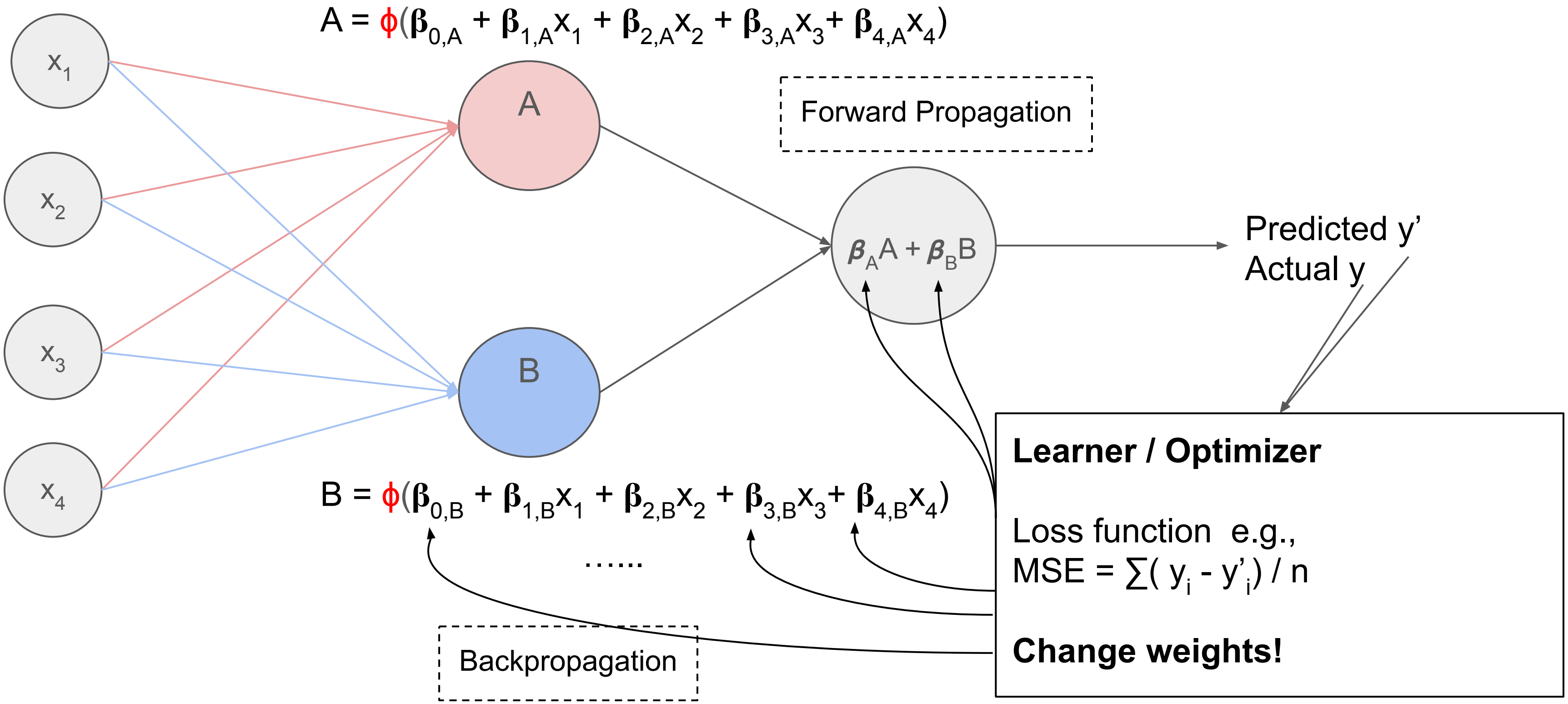
Calculating the derivative. Use the chain rule.

$dA/dx_1 = \boldsymbol{\beta}_{1,A}$

$A = \boldsymbol{\beta}_{0,A} + \boldsymbol{\beta}_{1,A}x_1 + \boldsymbol{\beta}_{2,A}x_2 + \boldsymbol{\beta}_{3,A}x_3 + \boldsymbol{\beta}_{4,A}x_4$

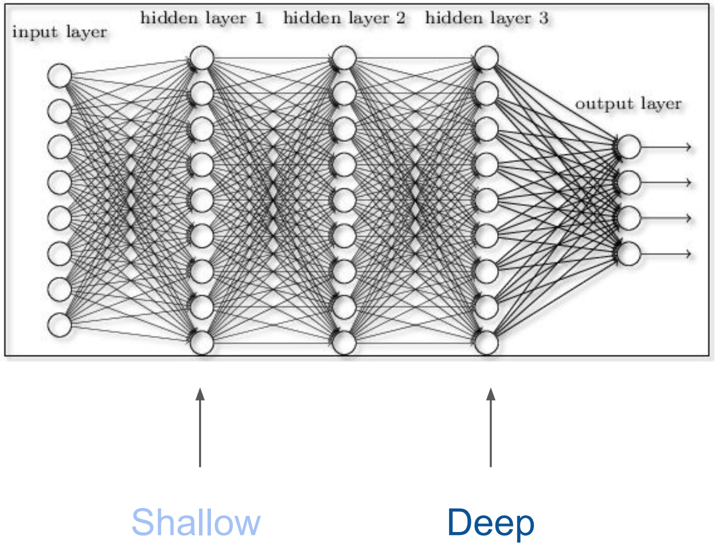


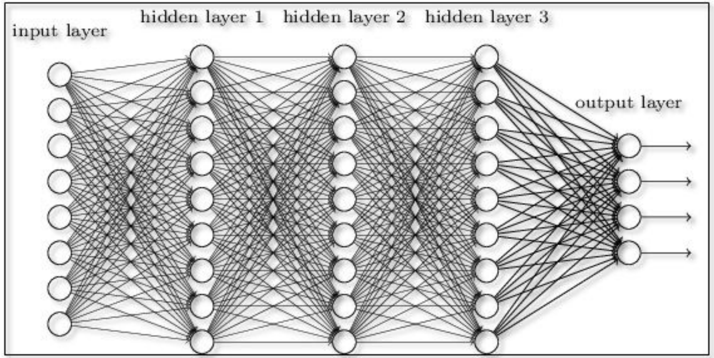
Backpropagation tells us how to change weights



Deep Neural Networks

By stacking many hidden layers, a “deep” neural network is created



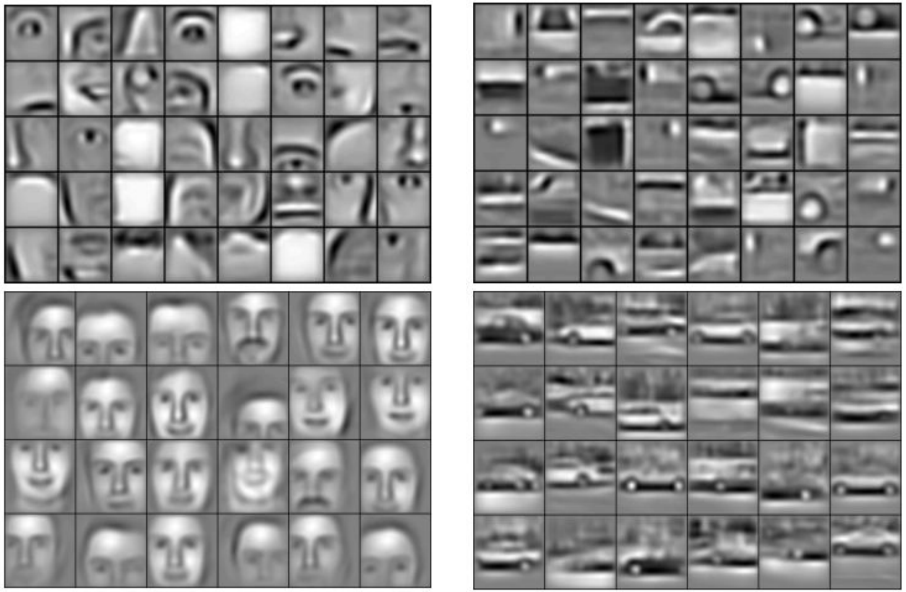


Shallow

Deep

Shallow Layer

Deep Layer

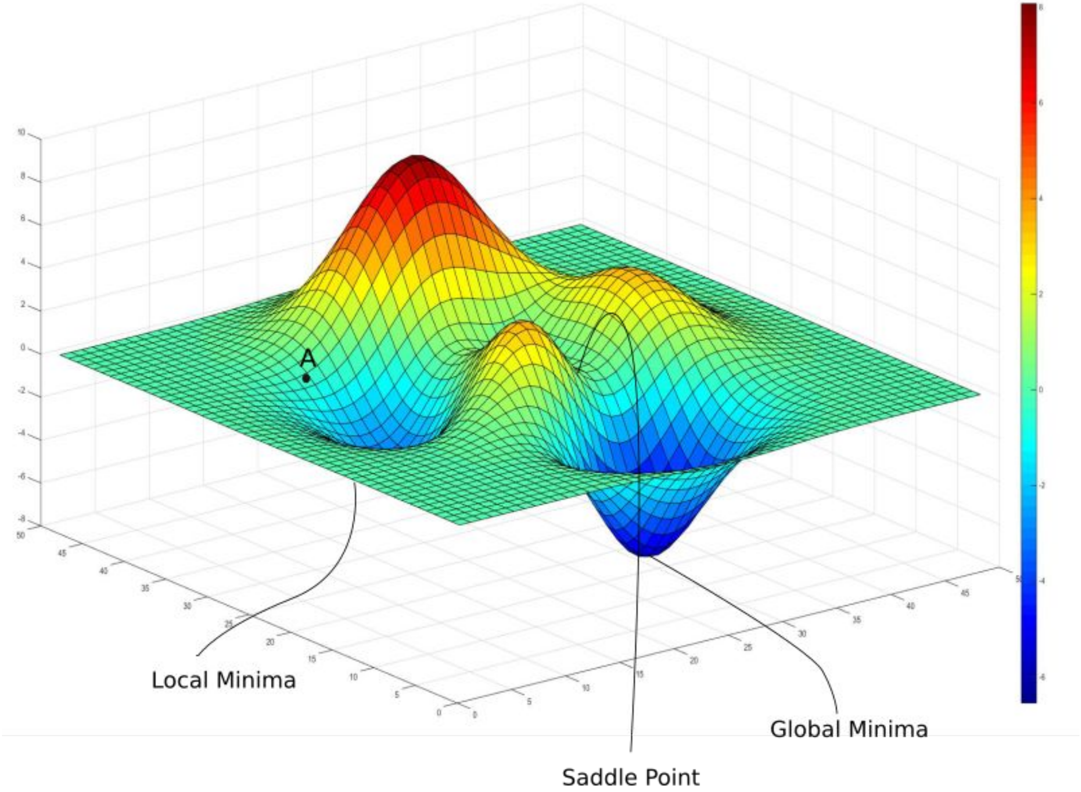


<u>Convolutional Deep Belief Networks</u> for Scalable Unsupervised Learning of Hierarchical Representations, Lee et al. 2009 ICML

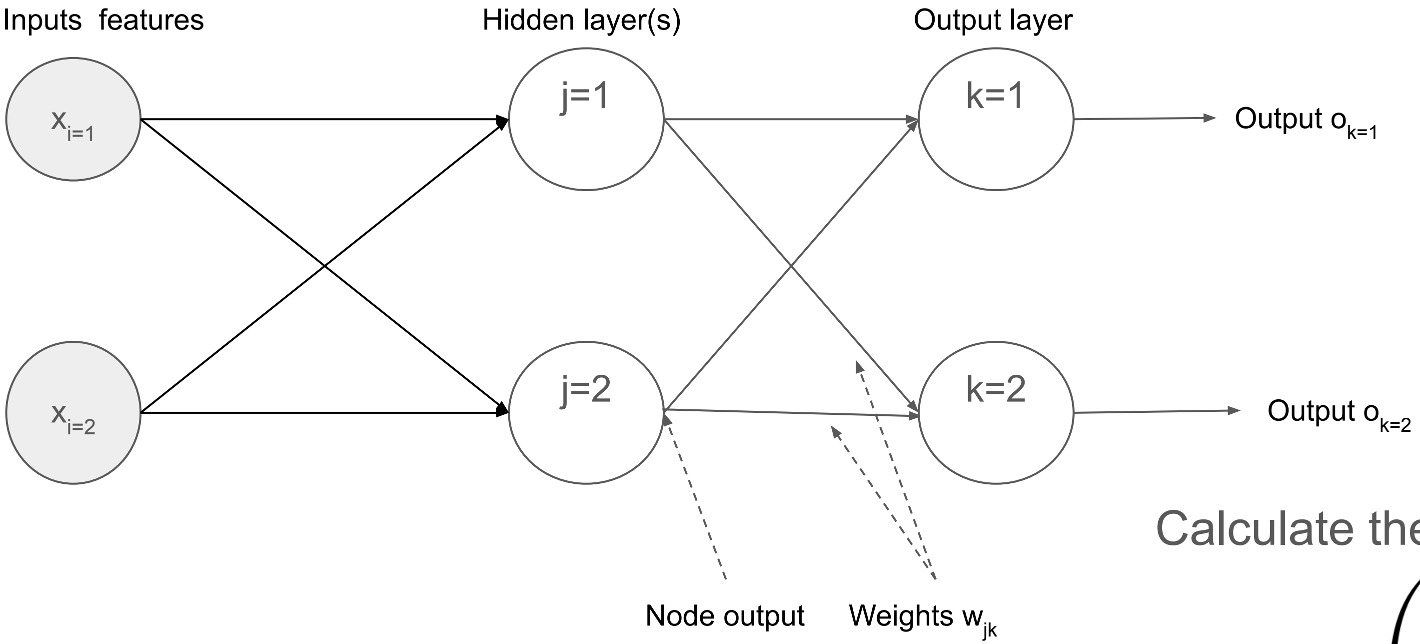
Why Deep Learning?

- End-to-end learning
- Deal with multimodal data effectively
- Abstraction from mathematical details
- Rapid prototyping, high-level libraries
- Unreasonable effectiveness
- Parameters >> Data
- …

Gradient-Based Optimization

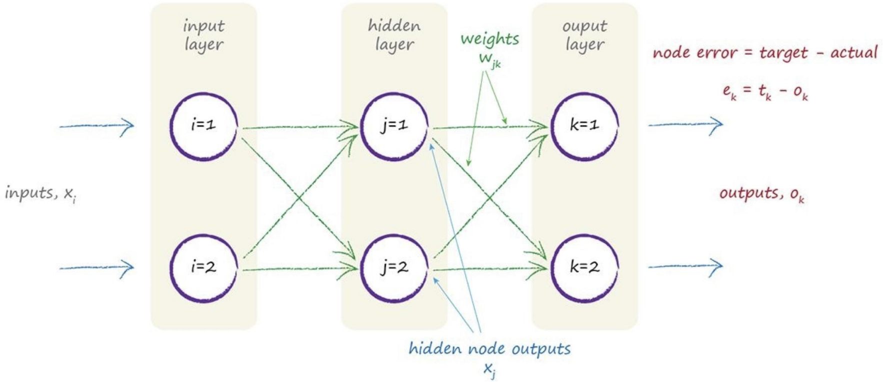


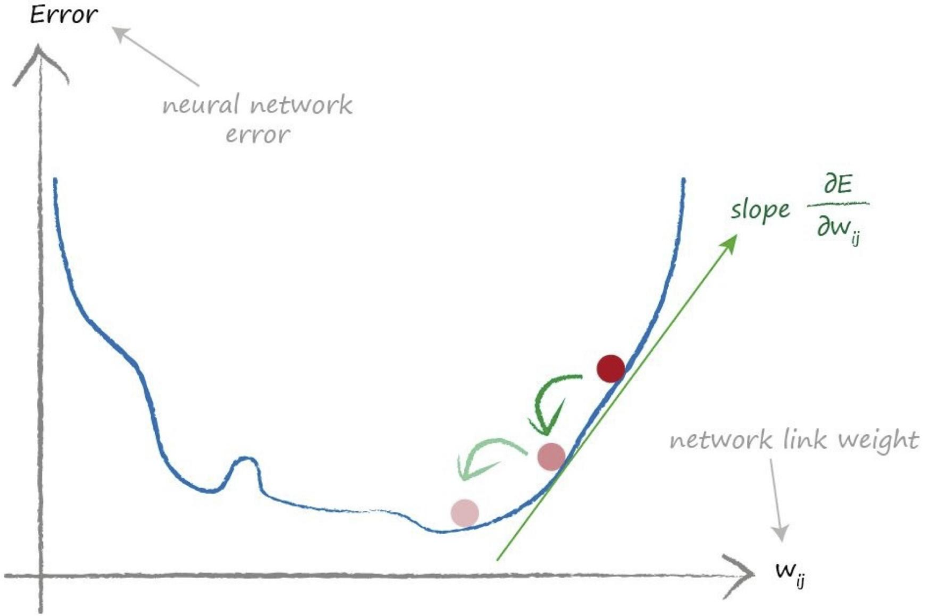
Gradient-Based Optimization



Calculate the node at kth layer

$$o_k = S\left(\sum_j w_{jk} o_j\right)$$

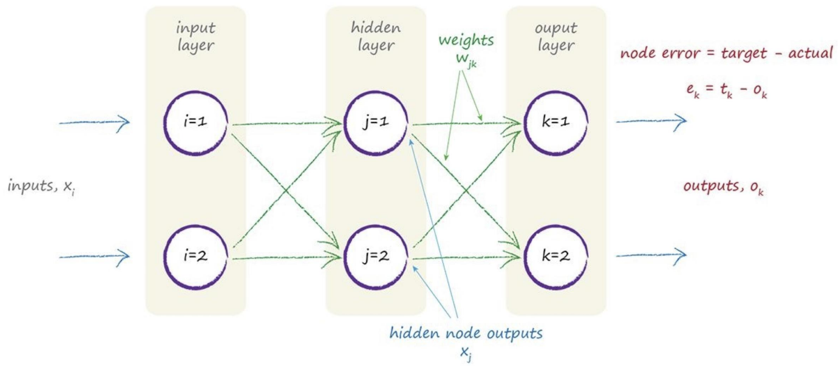




$$E = \sum_n (t_n - o_n)^2$$

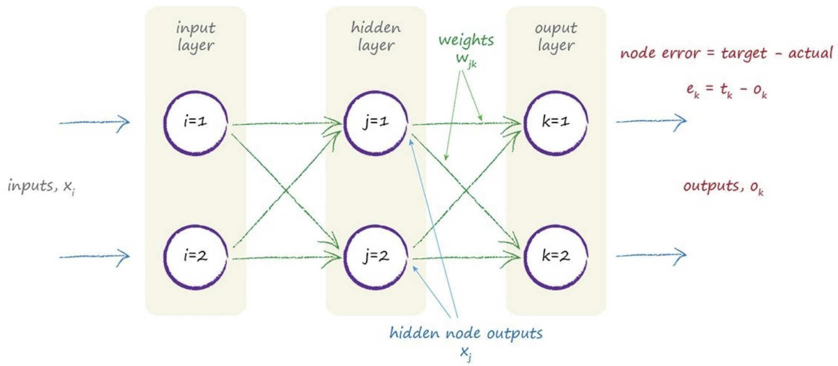
$$w_{ij}^{\text{new}} = w_{ij}^{\text{old}} - \alpha \frac{\partial E}{\partial w_{ij}}$$

α = learning rate, the very number (step size) taken into the gradient direction



$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial o_k}{\partial w_{jk}} \frac{\partial E}{\partial o_k}$$

$$= -2(t_k - o_k)\frac{\partial o_k}{\partial w_{jk}}$$



$$\frac{\partial o_k}{\partial w_{jk}} = \frac{\partial}{\partial w_{jk}} S(x_k) = \frac{\partial}{\partial w_{jk}} S\left(\sum_{j'} w_{j'k} o_{j'}\right)$$

$$= S(x_k)(1 - S(x_k)) \frac{\partial x_k}{\partial w_{jk}}$$

$$= S(x_k)(1 - S(x_k)) o_j$$

$$\boxed{\frac{\partial S(x)}{\partial x} = S(x)(1 - S(x))}$$

Putting everything together ..

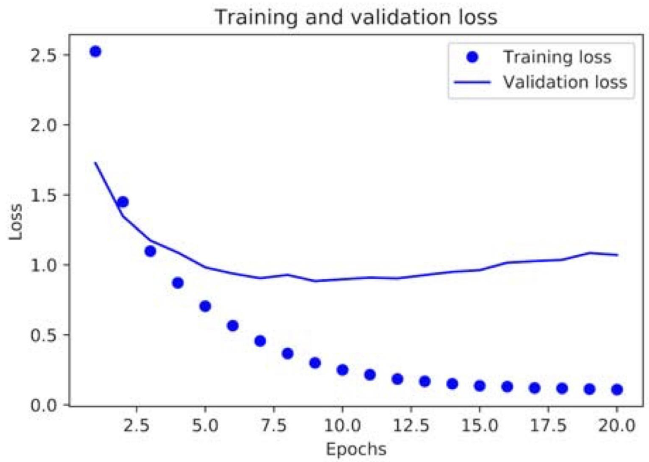
$$w_{jk}^{\text{new}} = w_{jk}^{\text{old}} - \frac{\partial E}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = -2(t_k - o_k)S(x_k)(1 - S(x_k))o_j$$

$$x_k = \sum_j w_{jk} o_j$$

Training may lead to overfitting

- Tension between optimization and generalization
- Optimization: performance on training data
- Generalization: performance on unseen data
- Split into training, test, AND validation sets



How to avoid overfitting

- More training data
  - Diverse, unbiased, random sampling
- Constrain information stored in the network
  - Smaller network
  - Weight regularization
  - Dropout
- Data augmentation
  - Geometric transformation
  - Combination of multiple parts
  - Erasing

The double-descent phenomenon

Reconciling modern machine-learning practice and the classical bias–variance trade-off

Mikhail Belkin[a,b,1], Daniel Hsu[c], Siyuan Ma[a], and Soumik Mandal[a]

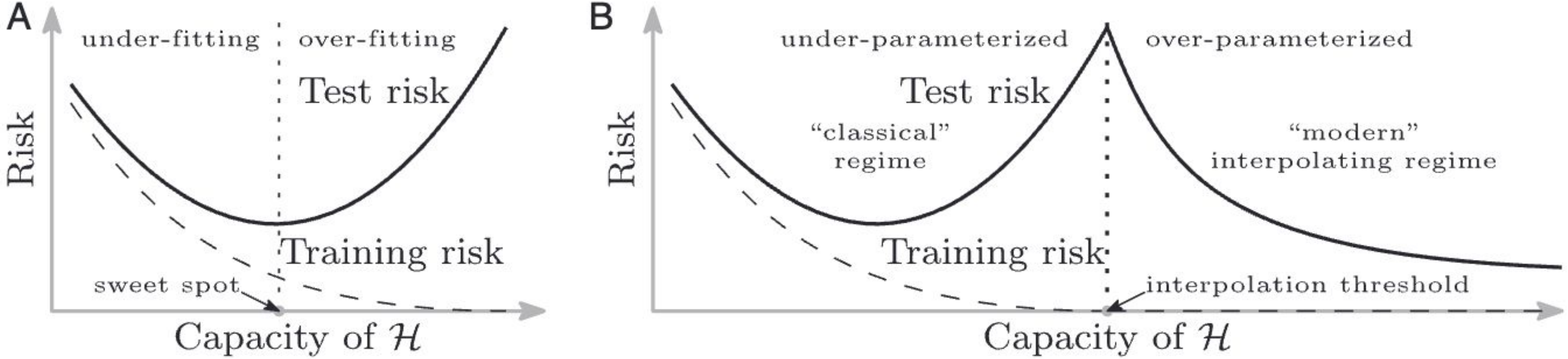
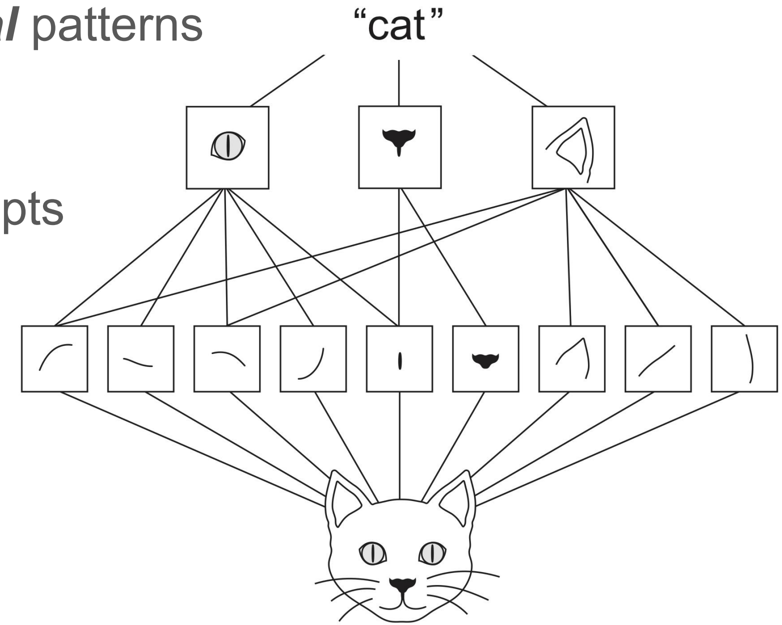


Fig. 1. Curves for training risk (dashed line) and test risk (solid line). (*A*) The classical U-shaped risk curve arising from the bias–variance trade-off. (*B*) The double-descent risk curve, which incorporates the U-shaped risk curve (i.e., the "classical" regime) together with the observed behavior from using high-capacity function classes (i.e., the "modern" interpolating regime), separated by the interpolation threshold. The predictors to the right of the interpolation threshold have zero training risk.

Convolutional Neural Networks

- Densely connected nets learn ***global*** patterns
- Convolutional neural network learn ***local*** patterns
- Learn translational invariant patterns
- Learn hierarchies of patterns and concepts



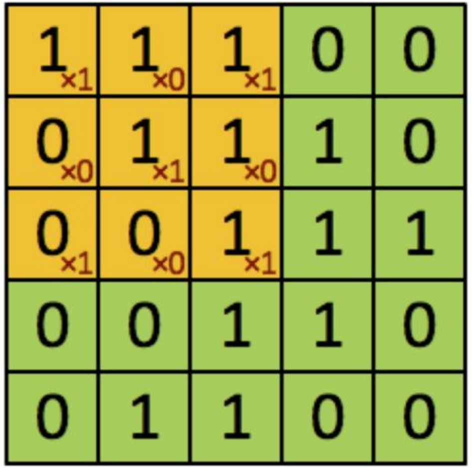
<u>Well suited to process images</u>

Convolution

Filter (3x3)

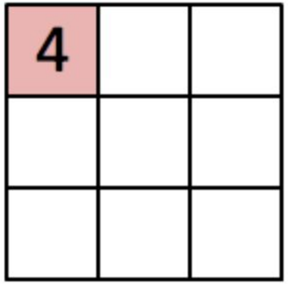
1	0	1
0	1	0
1	0	1

Filter on an Image (5x5 pixels)



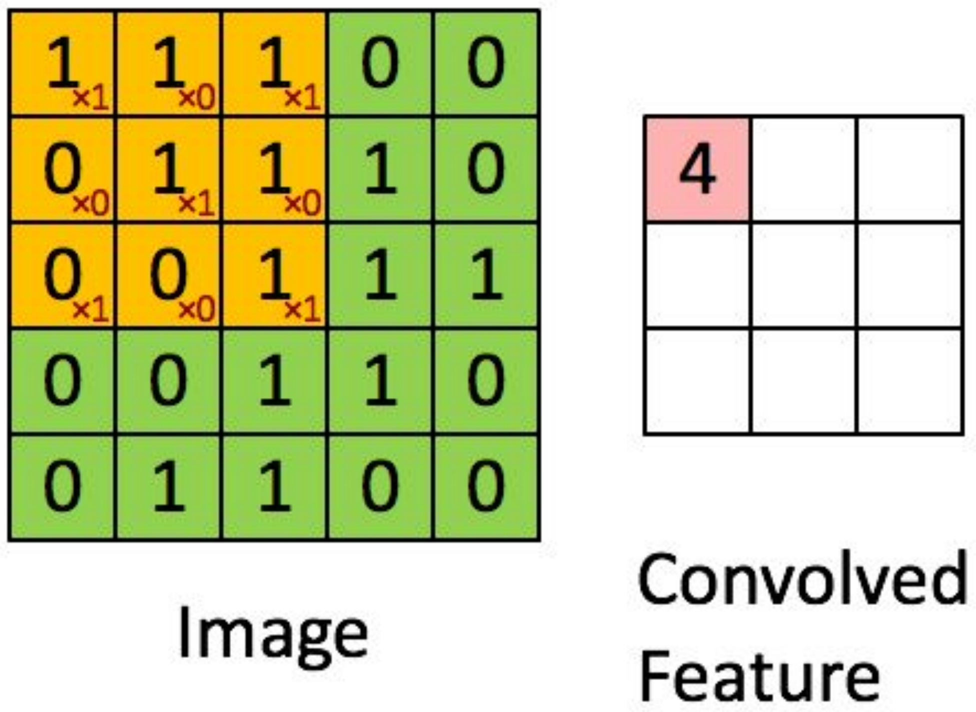
Convoluted Feature

A sum of element-wise multiplications



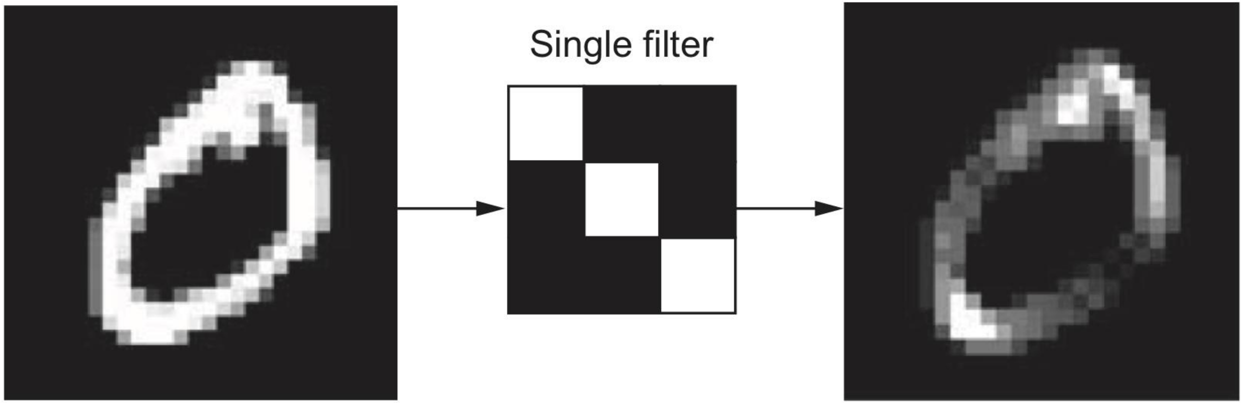
Convolution

A given filter is applied throughout an image

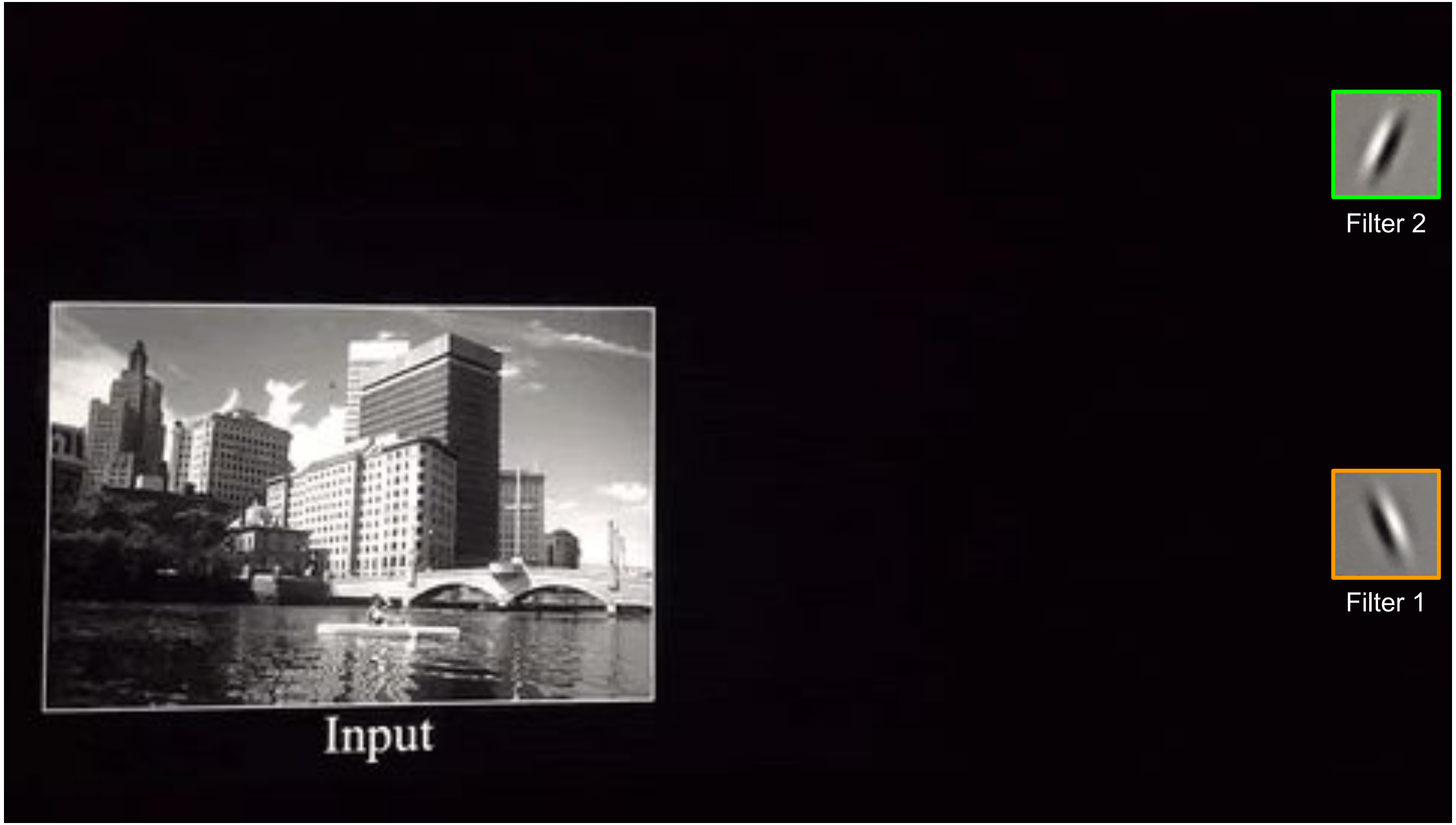


Convolution

Classically, a filter is created that can detect an edge, create a blur, etc



Convolutional Image Filters as Feature Extractors

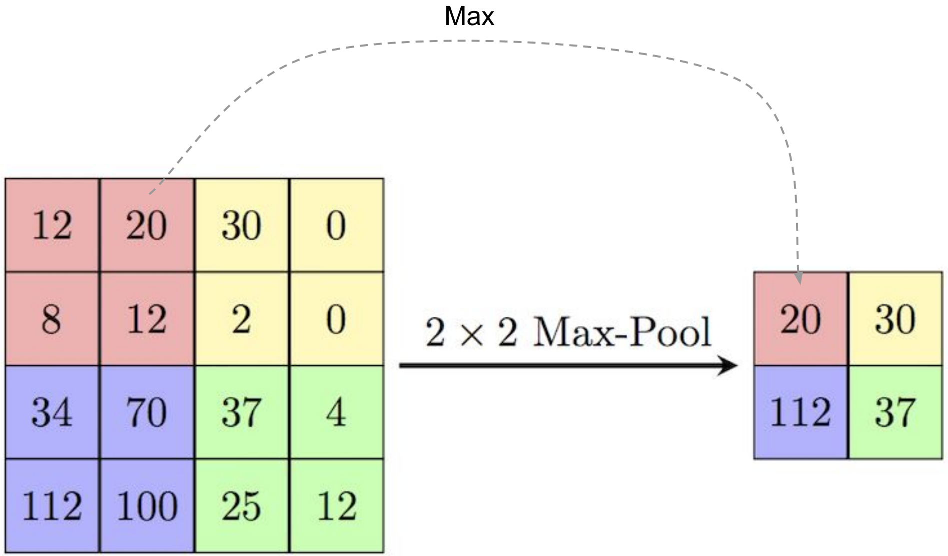


https://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/

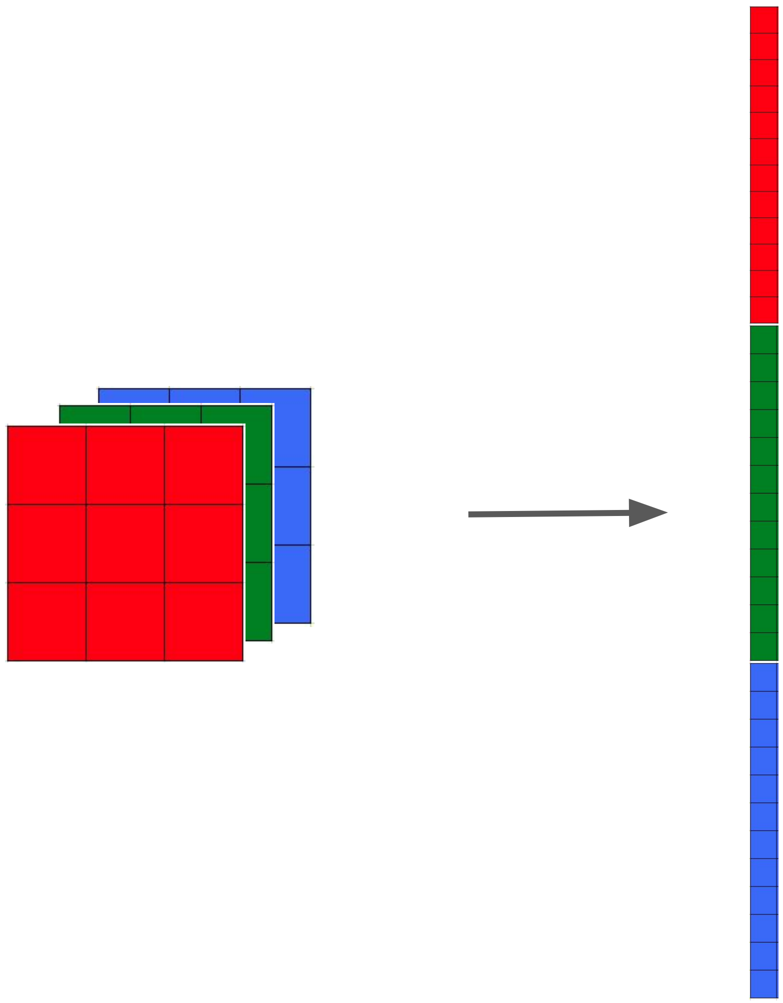
Pooling

- Reduces spatial dimension
- Lowers number of parameters
- Enables deeper layers to learn large high-level patterns

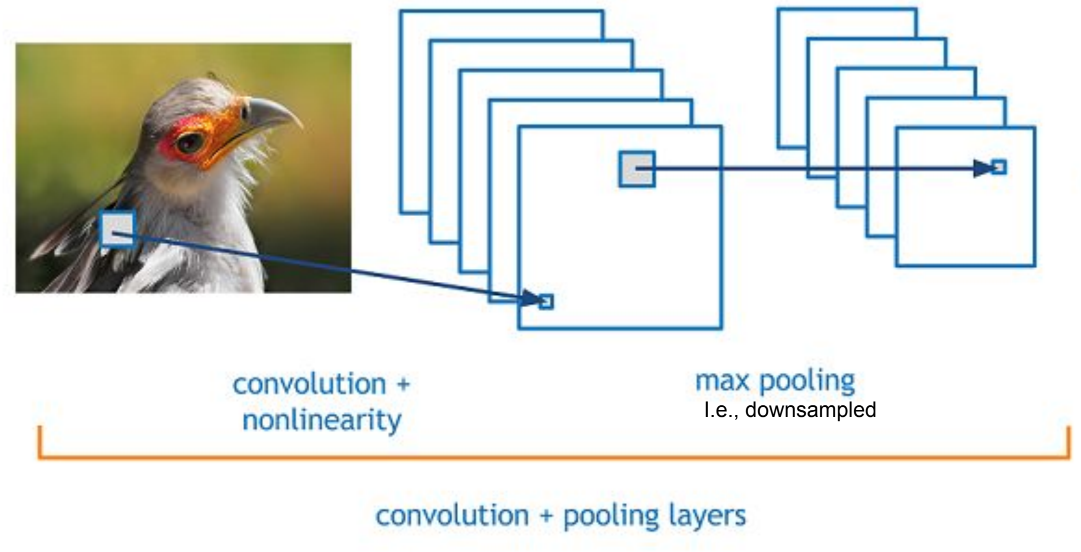
- Other downsampling strategies possible



Flatten



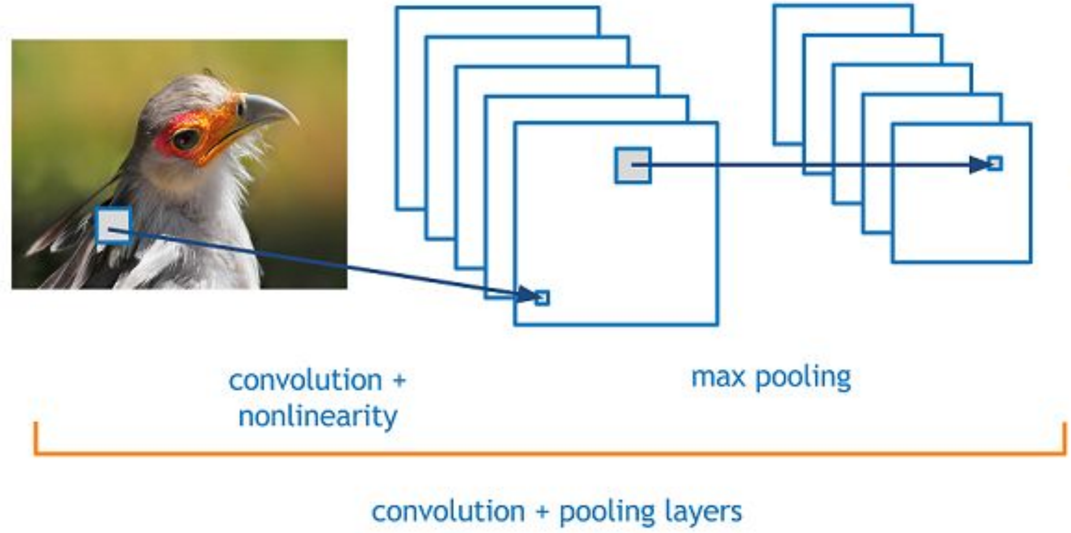
Convolutional Neural Networks



CNN takes an image +
perform convolutions using
a number of filters (per channel)

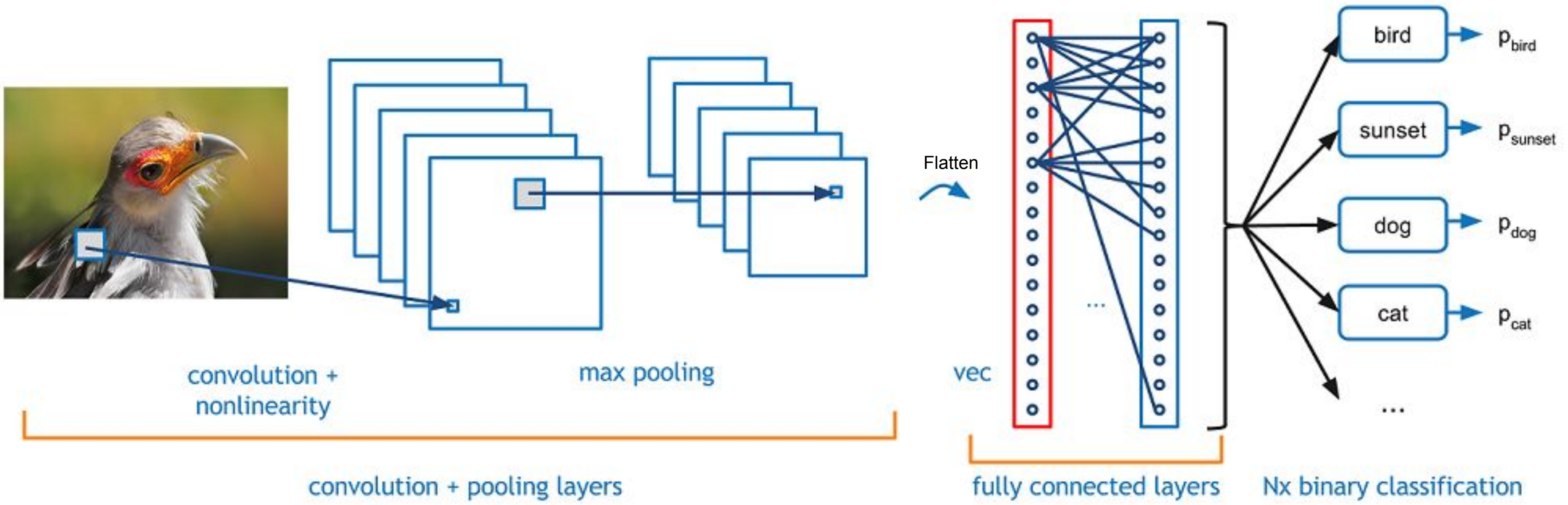
https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

Convolutional Neural Networks



Feature extraction +
Hierarchical representation

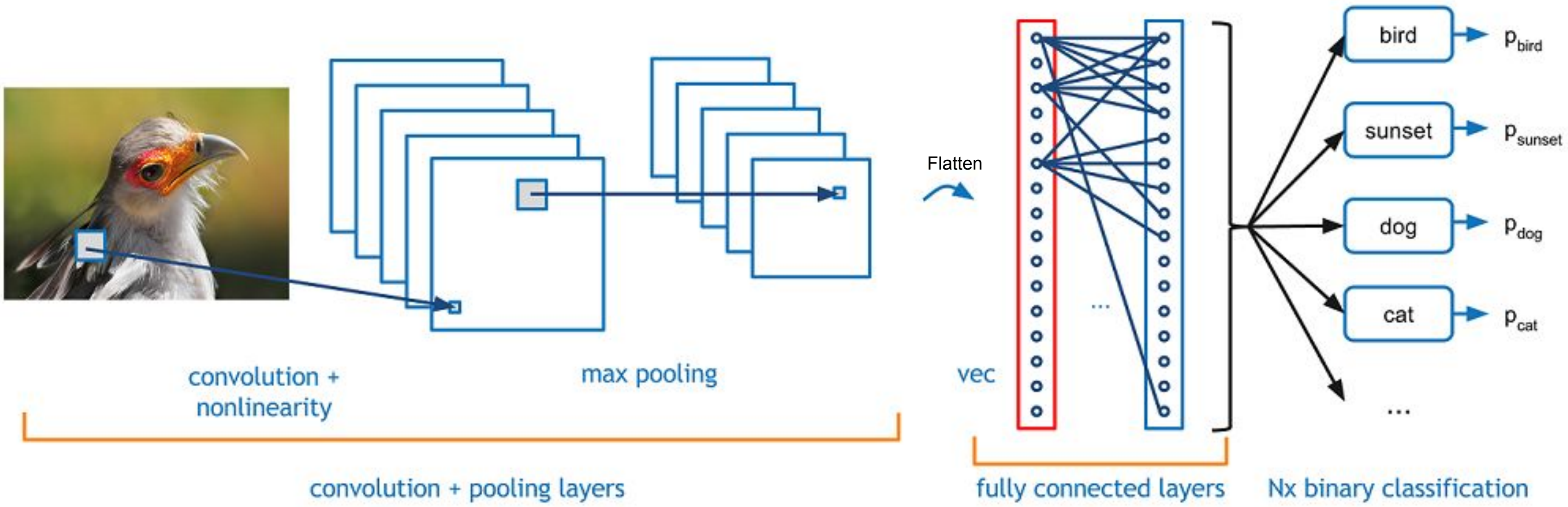
Convolutional Neural Networks



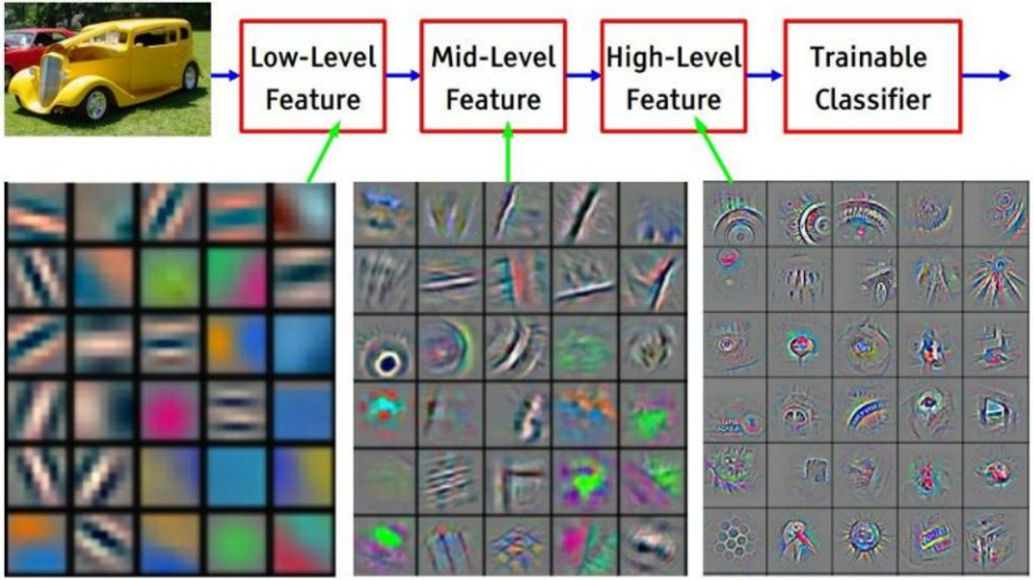
Feature extraction +
Hierarchical representation

Classification

Using combinations of convolutions, pooling, and other operations
Using different activation functions and regularization
Using many layers (deeper)

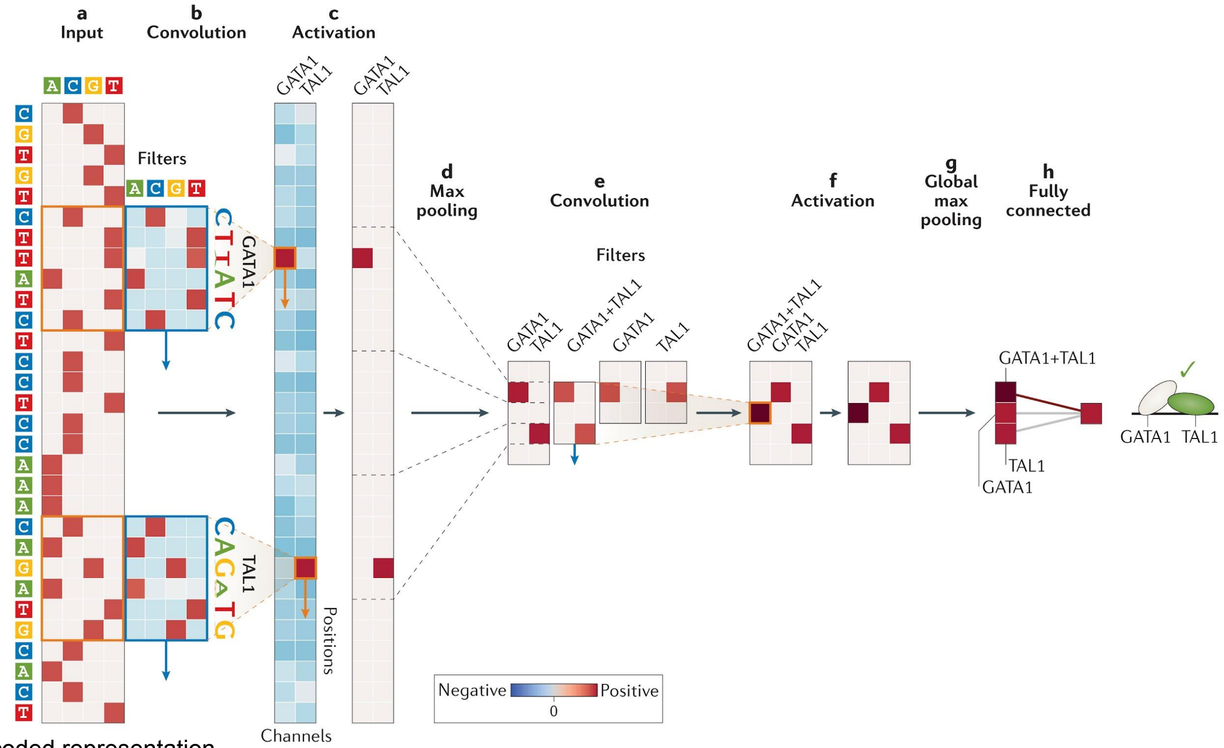


Hierarchy of concepts



https://towardsdatascience.com/build-your-own-convolution-neural-network-in-5-mins-4217c2cf964f

Modelling transcription factor binding sites



One-hot encoded representation of the DNA sequence

Eraslan et. al. (2019)

Visualization of Filter Response

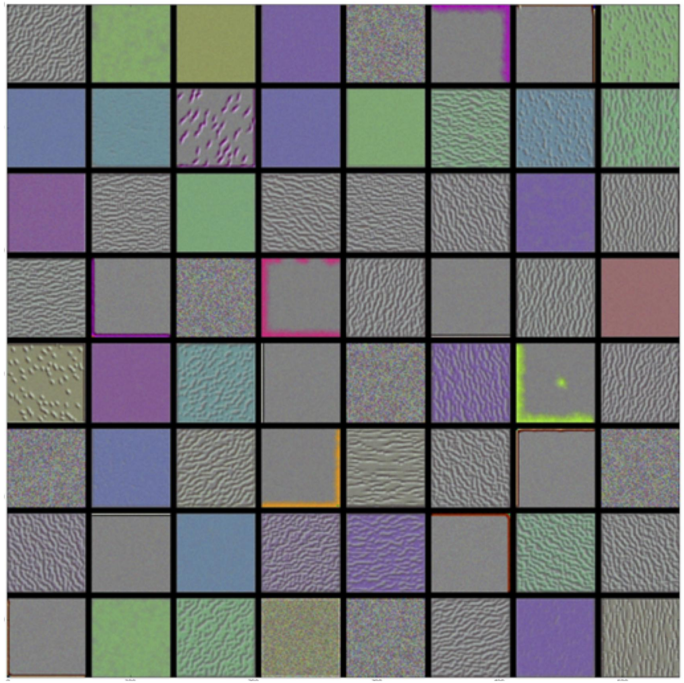
Gradient descent in input space

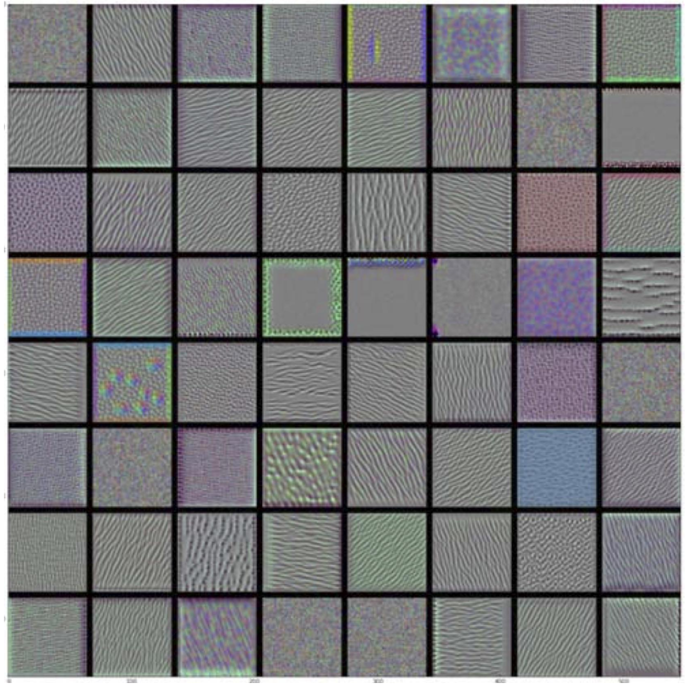
- Loss function maximizes mean activation of some filter
- Start with blank input image
- Gradient descent to change input image

⇒ Maximize filter activation

⇒ Obtain images each filter is most responsive too

Visualization of Filter Response





Visualization of Filter Response