Neural networks, CNNs, and AlphaFold

Brian Kissmer

USU Department of Biology

Dec. 3rd, 2024

Learning objectives

1. Gain a conceptual foundation of neural networks and deep learning
2. Have a basic understanding of inference with neural networks
3. Learn about some examples of neural networks used in the study of biology

Final exam

1. Scheduled for 1:30-3:20pm, but will be available all day Tuesday
2. Similar to midterm but focusing on second half of the course

Course laptops

1. Get these back to me once you’re done with them
2. I’ll be in the classroom on Thursday (Dec. 12) at our normal meeting time to receive laptops. Otherwise let me know when you can get it to me
3. Will receive an incomplete grade if not returned

Outline

1. Overview of neural networks
2. CNNs
3. AlphaFold
4. Programming Project 6

Machine learning

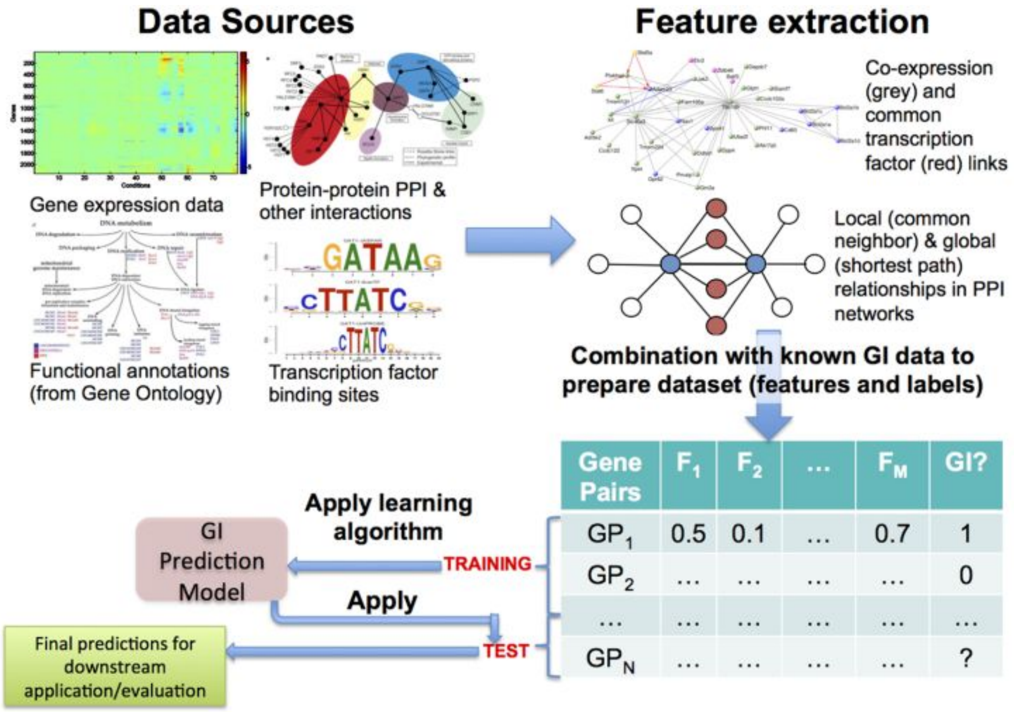
Machine learning comprises a variety of computaitnal methods including many that are popular in bioinformatics, as well as computational biology more broadly:

- Random Forest
- Neural networks
- Support-vector machines
- LASSO regression

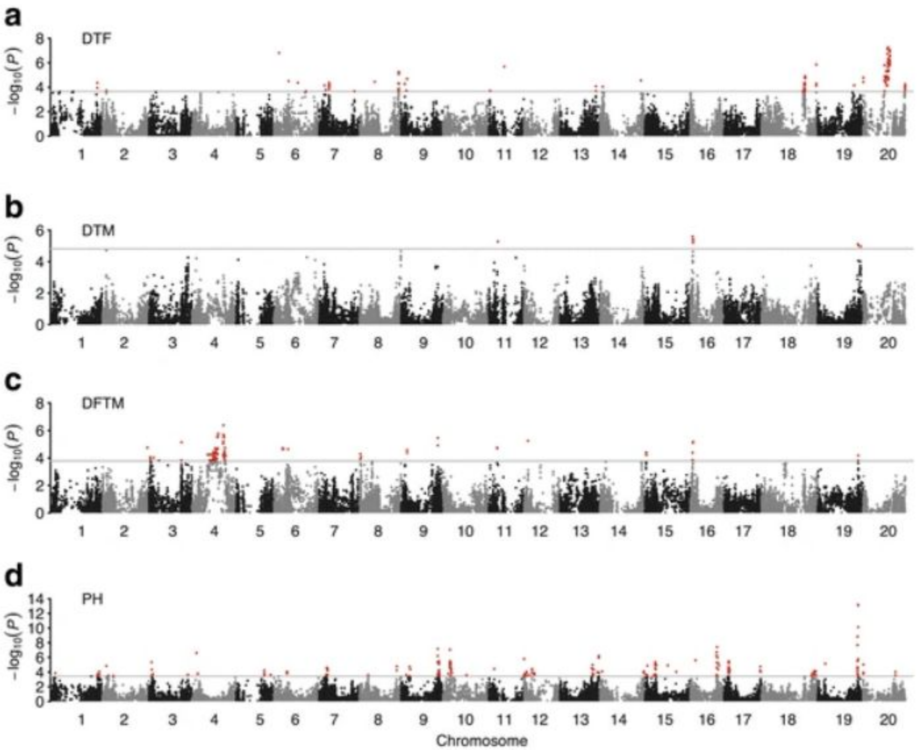
Applications of machine learning - image recognition

Dog?

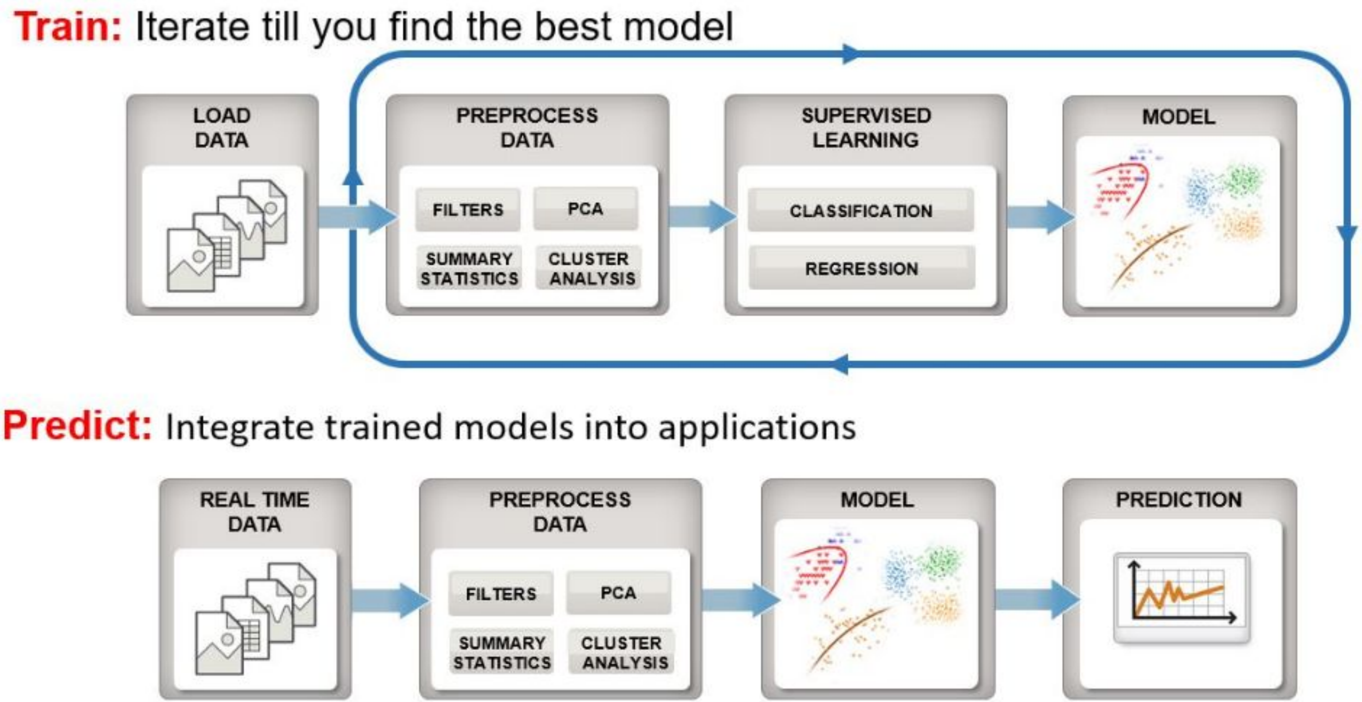
Applications of machine learning - prediction of gene interactions



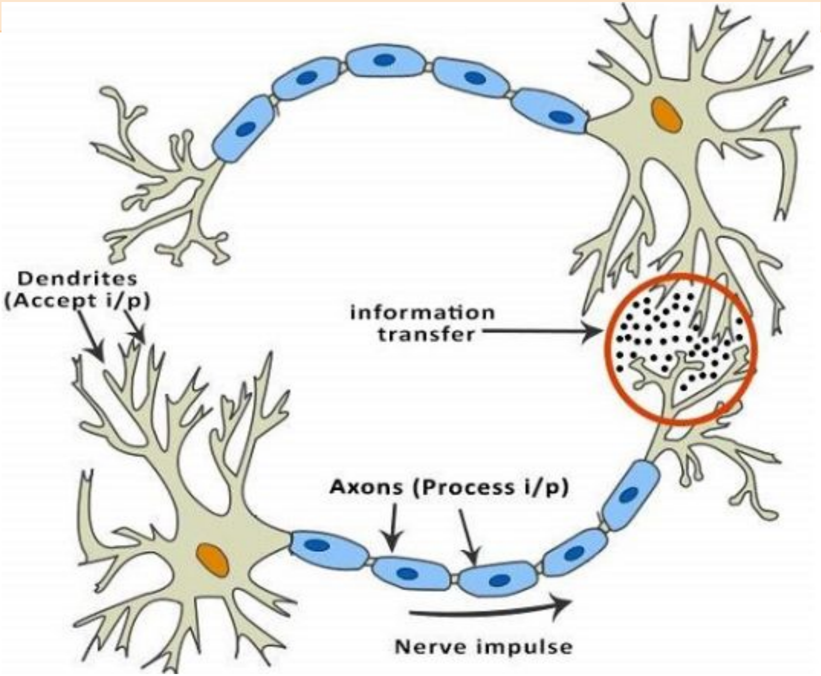
Applications of machine learning - prediction of phenotype from genotype



Generalized machine learning workflow

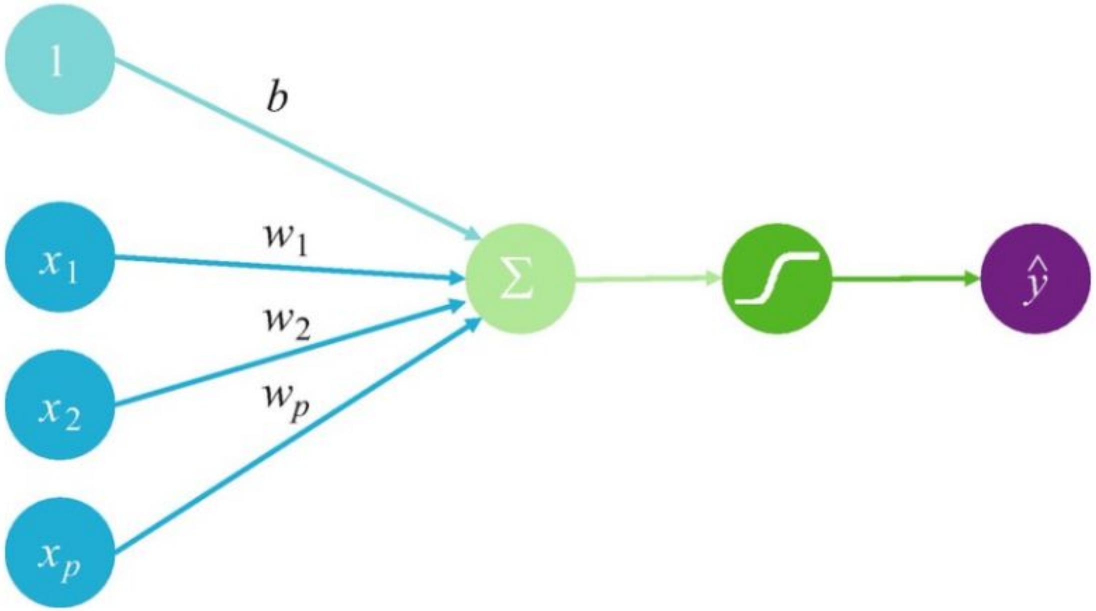


Artificial neural networks



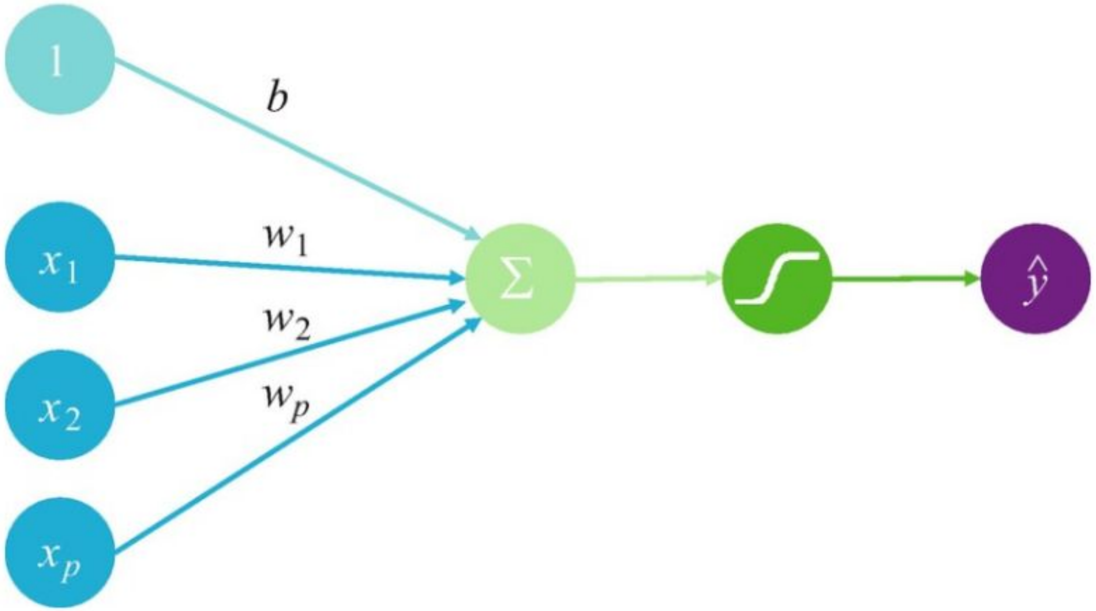
Machine learning models (loosely) inspired by biological neural networks

Artificial neural networks



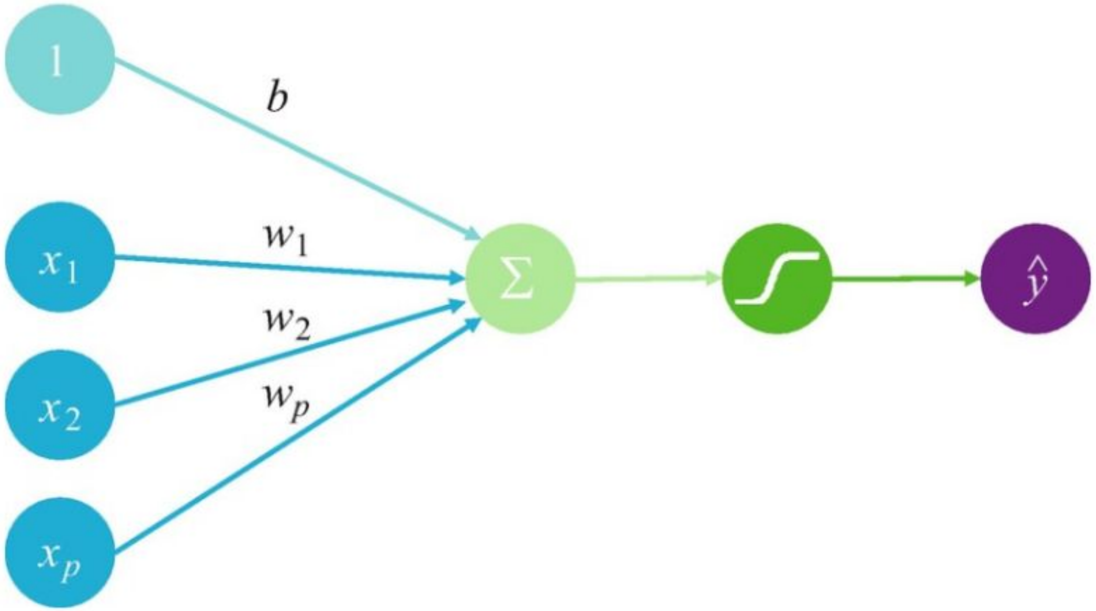
Models are composed of units that combine multiple inputs to produce outputs

Artificial neural networks



Simple NN (perceptron) model, classification ($\hat{y}$) is determined by whether the weighted sum of input elements exceeds a threshold

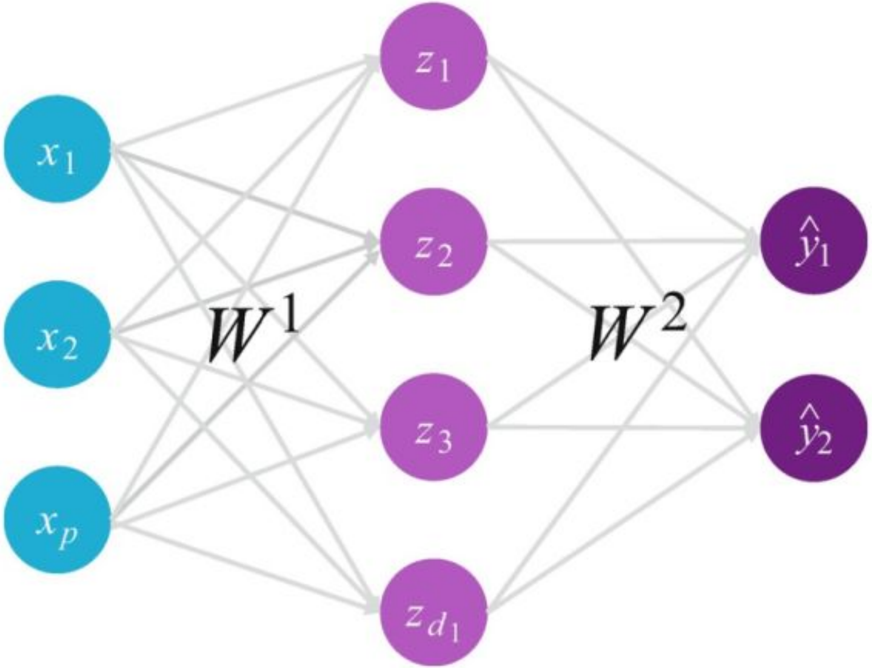
Artificial neural networks



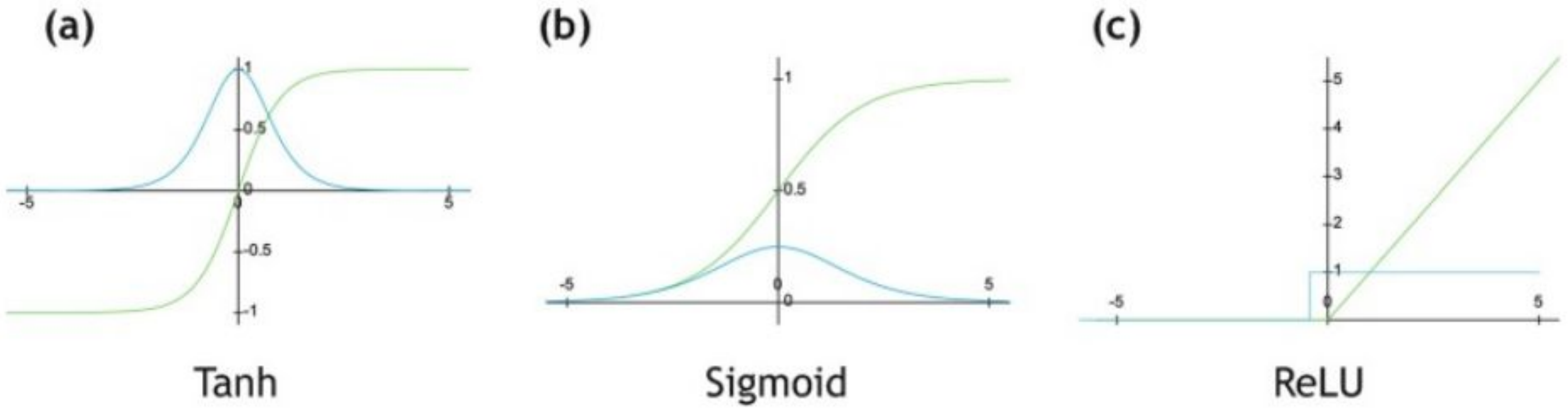
Neural networks are trained, increasing or decreasing the weights (w) iteratively

Multilayer neural networks

Multilayer networks consist of at least 3 layers: the <u>input layer</u>, <u>hidden layer</u> (z), and <u>output layer</u>

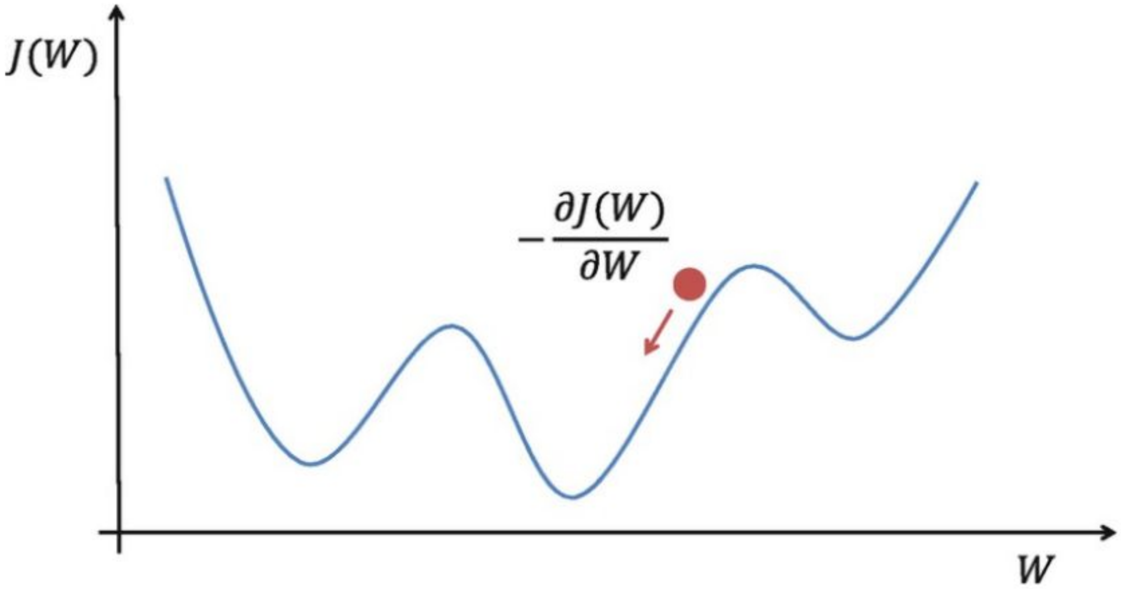


Activation functions for multilayer neural networks



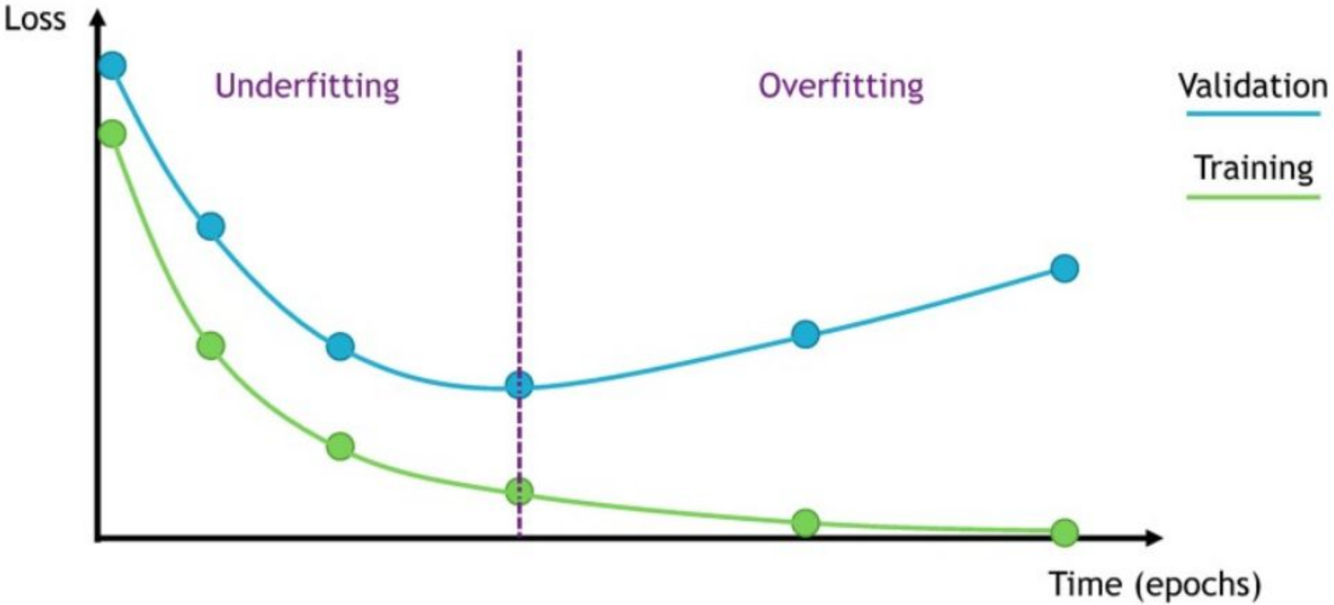
Common non-linear activation functions (green) and their derivatives (blue); ReLU = rectified linear

Learning involves tuning weights



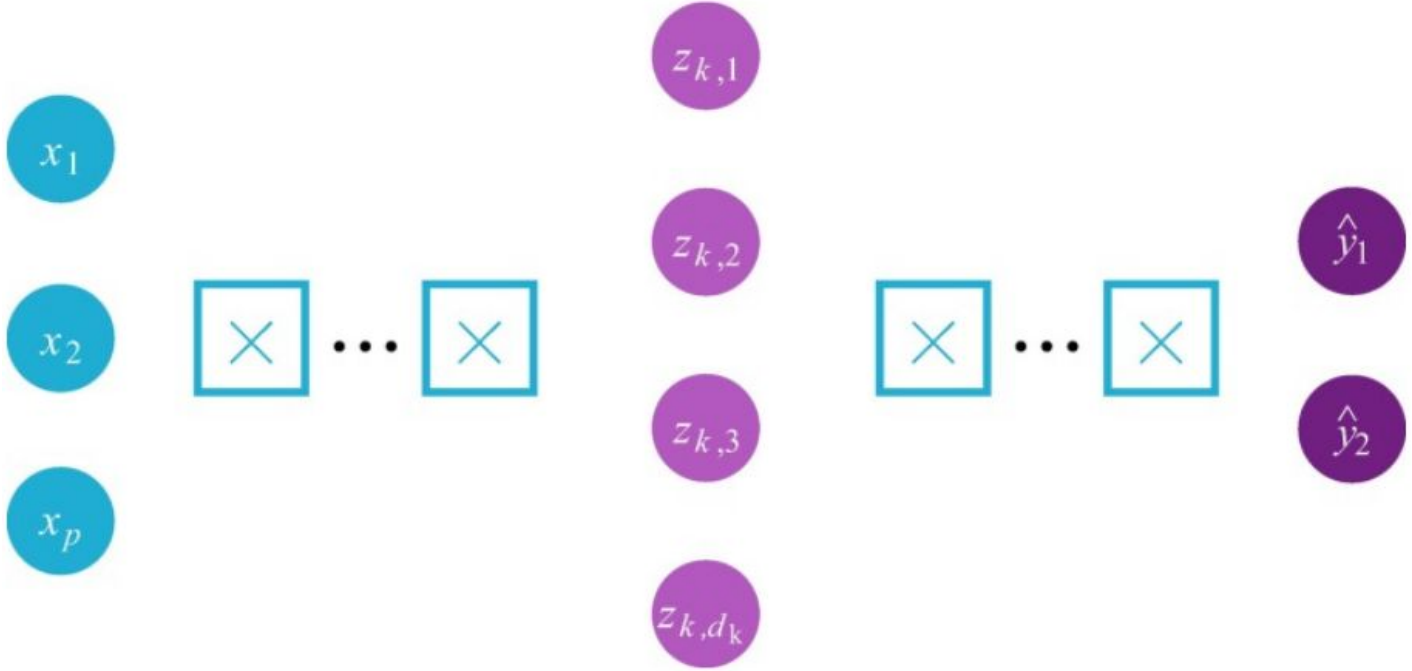
Gradient descent algorithms adjust weights (W) in the multivariate direction of deecreasing error (loss) *(J(W))*

Iterative training improves models until it doesn't



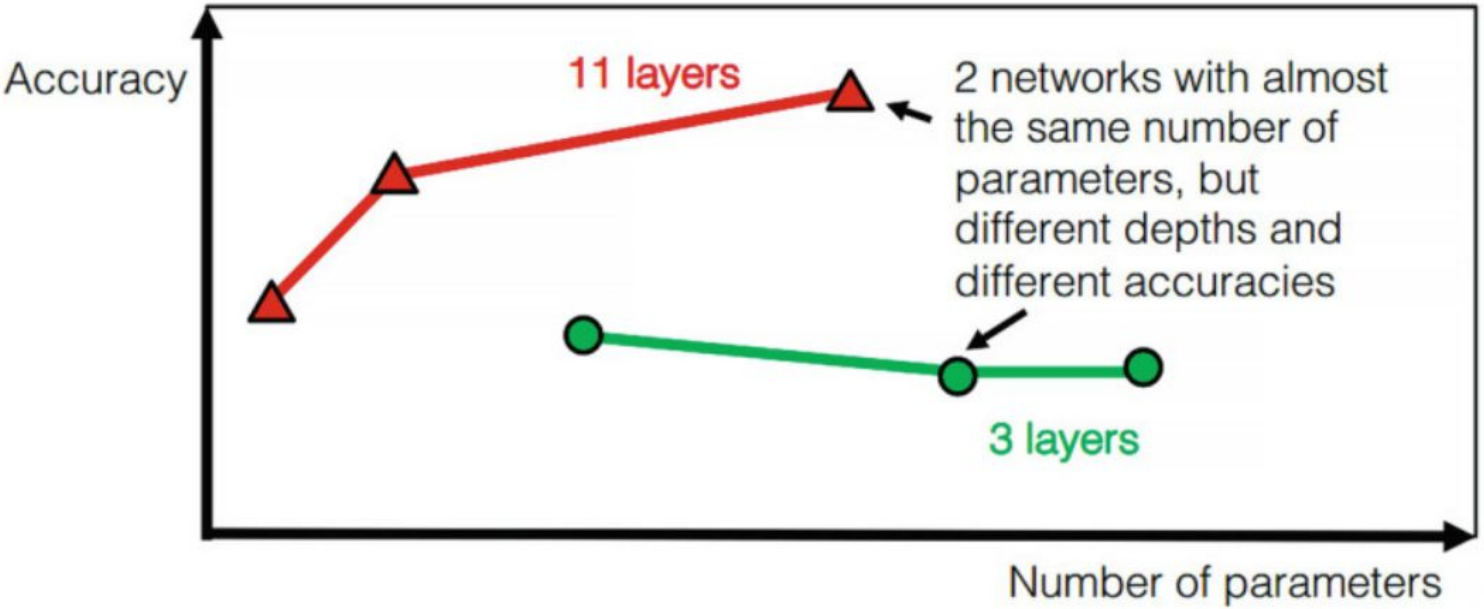
Too much training can focus the network on the training data and lead to poor performance with out-of-bag validation data (epoch = passes through the training data

Deep neural networks



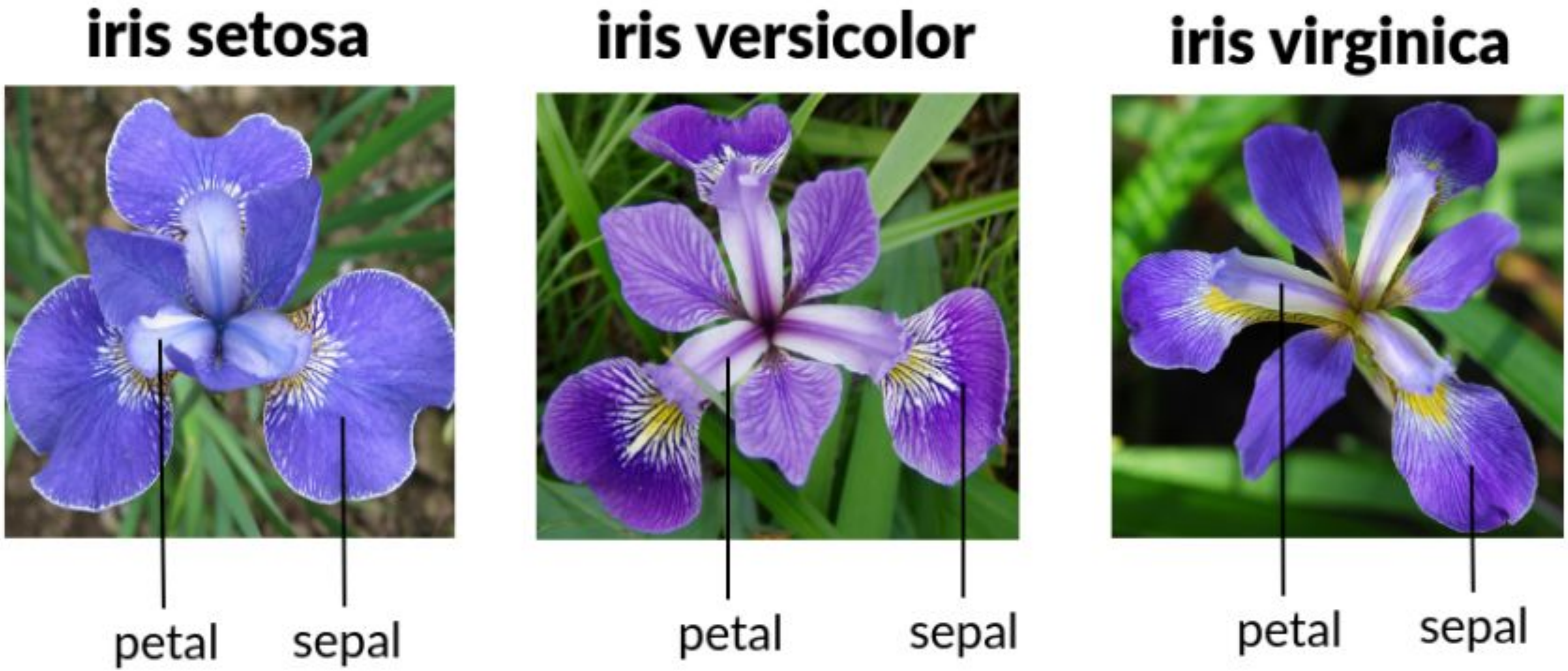
Deep neural networks learn with many hidden layers

Deep versus shallow neural networks



Even with a single layer, neural networks can approximate any function. However, deep networks do well with fewer parameters than shallow networks

Neural network example: *Iris* classification example



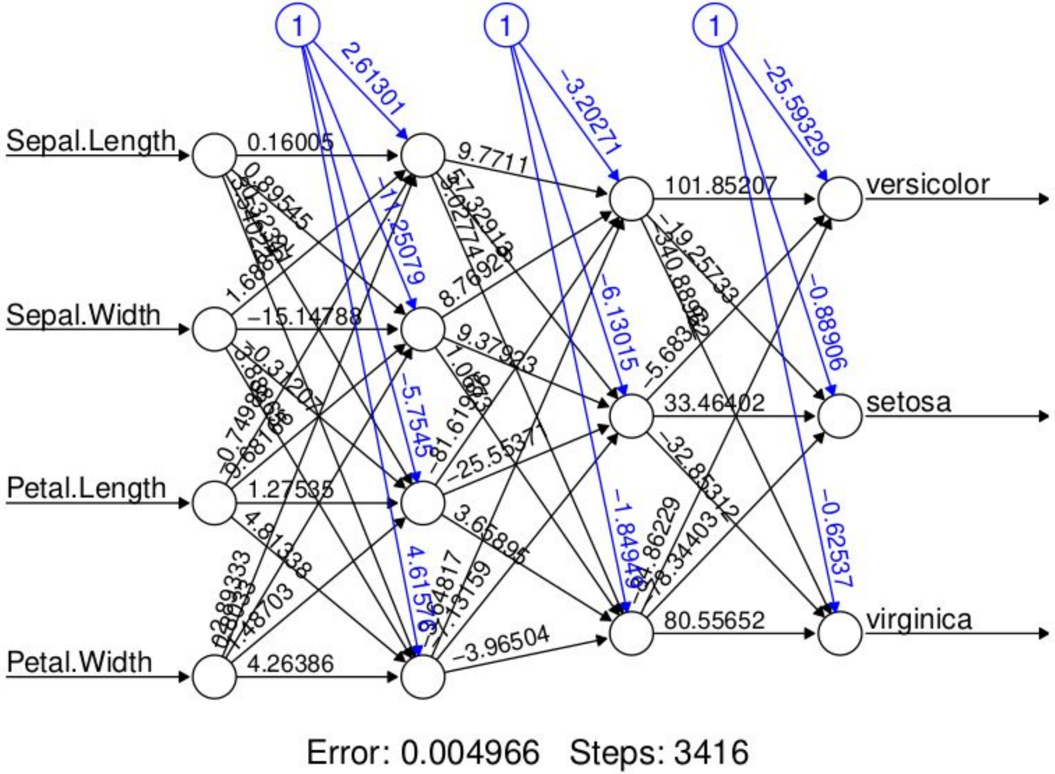
Iris classification neural network

```
library(neuralnet)
data(iris)
## divide into test (20) and training (80)
ntrain <- floor(0.80 * nrow(iris))
trainIndices <- sample(c(1:nrow(iris)), ntrain,
replace=FALSE)
train_data <- iris[trainIndices,]
test_data <- iris[-trainIndices,]
```

Iris classification neural network

```
model <- neuralnet(
Species~Sepal.Length+Sepal.Width+
Petal.Length+Petal.Width,
data=train_data,
hidden=c(4,3),
linear.output = FALSE)
pred <- predict(model, test_data)
preds <- apply(pred,1,which.max)
unique(as.character(test_data$Species))[preds]
```

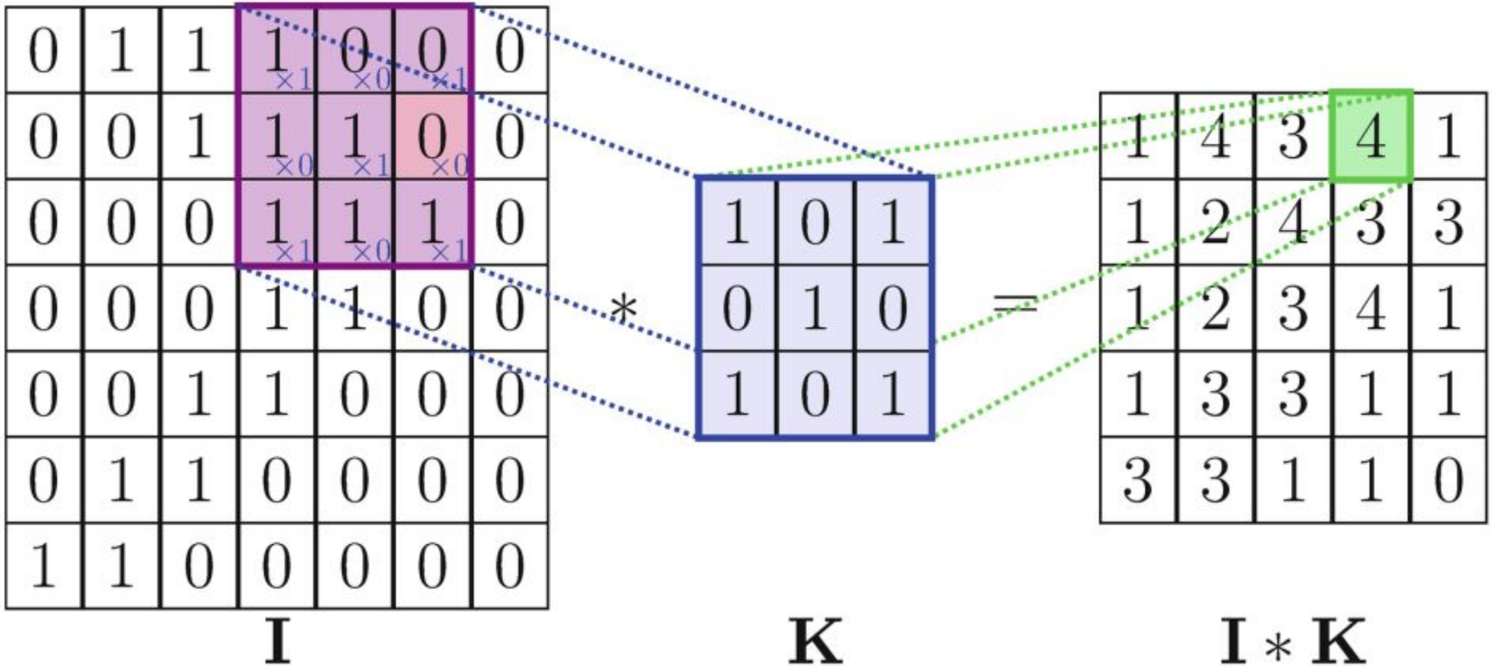
Iris classification neural network



Iris classification neural network: success

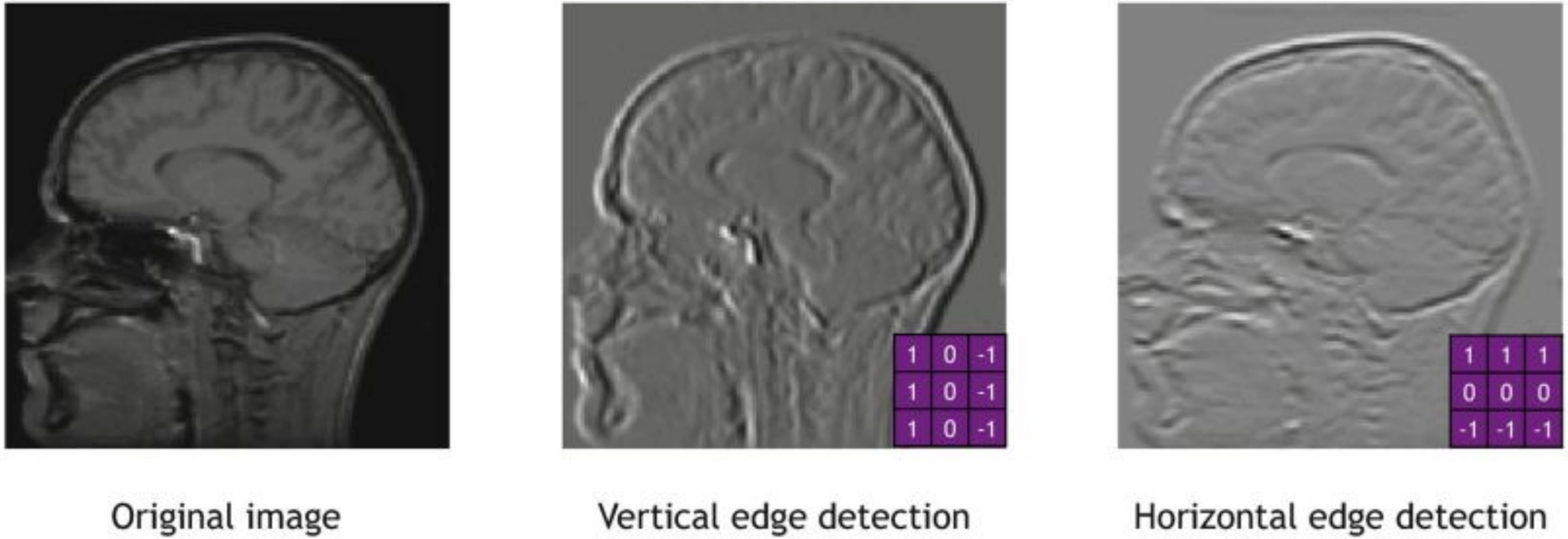
	setosa	versicolor	virginica
setosa	11	0	0
versicolor	0	8	0
virginica	0	0	11

Convolutional neural networks



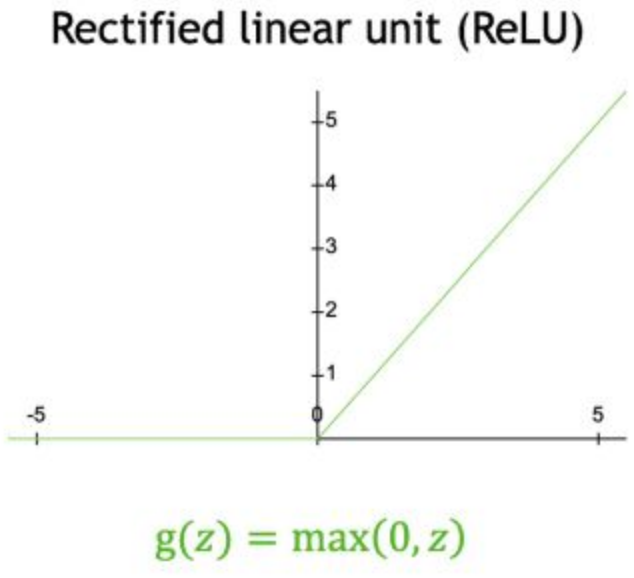
Visualization of a 2D convolution of image (I) with a filter (K)

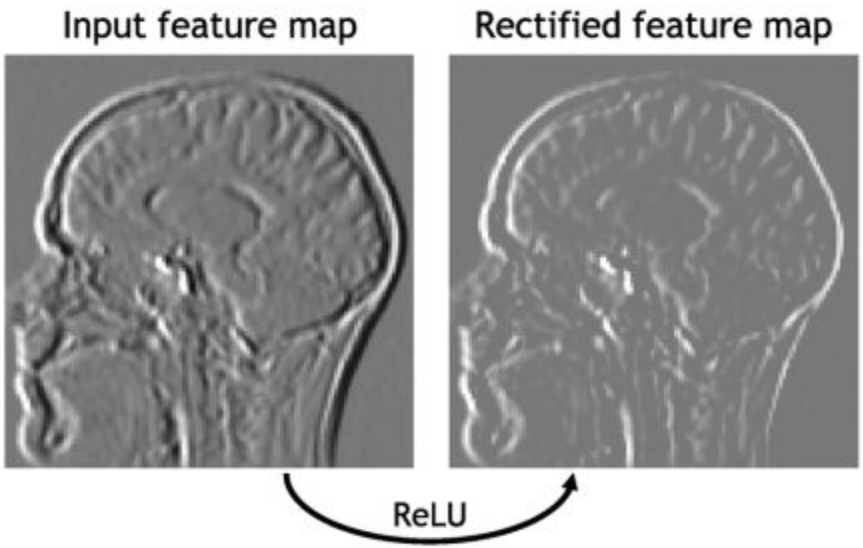
Convolutions applied to an image



The first convolution emphasizes vertical edges, the second emphasizes horizontal edges

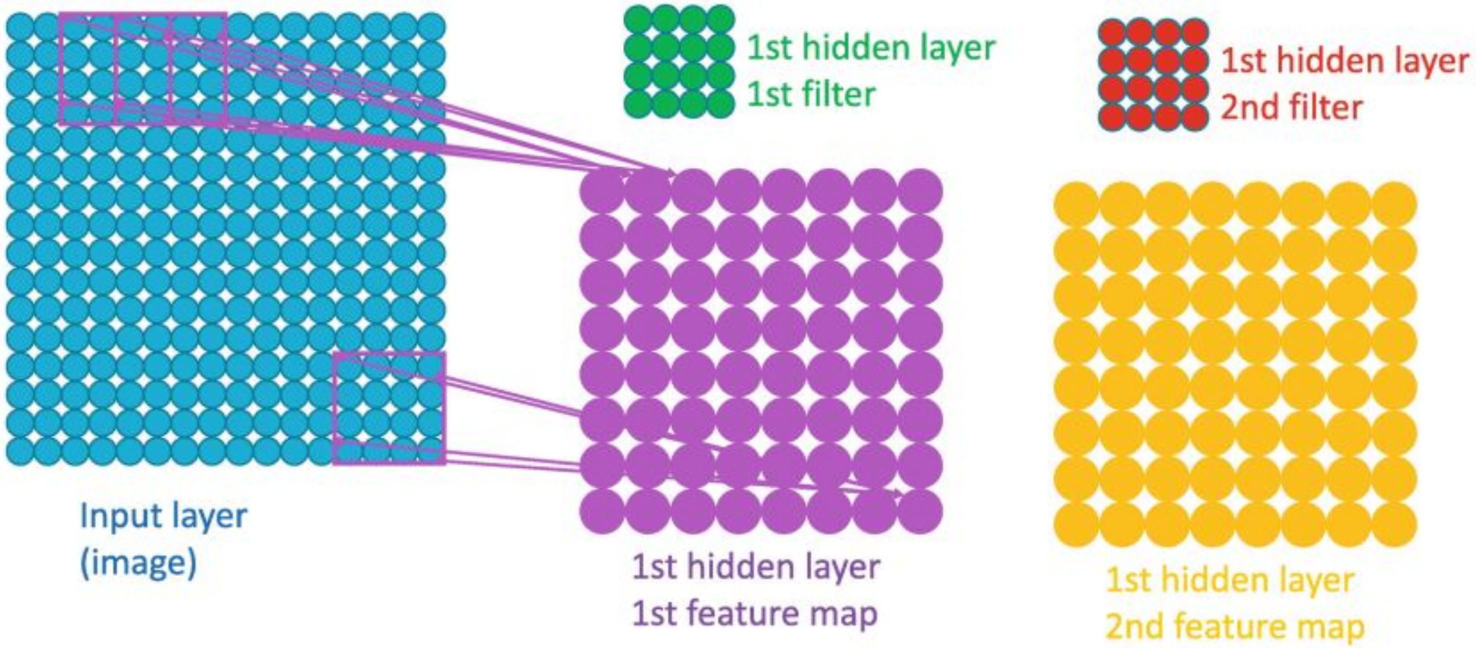
Convolutions plus non-linear activation applied to an image





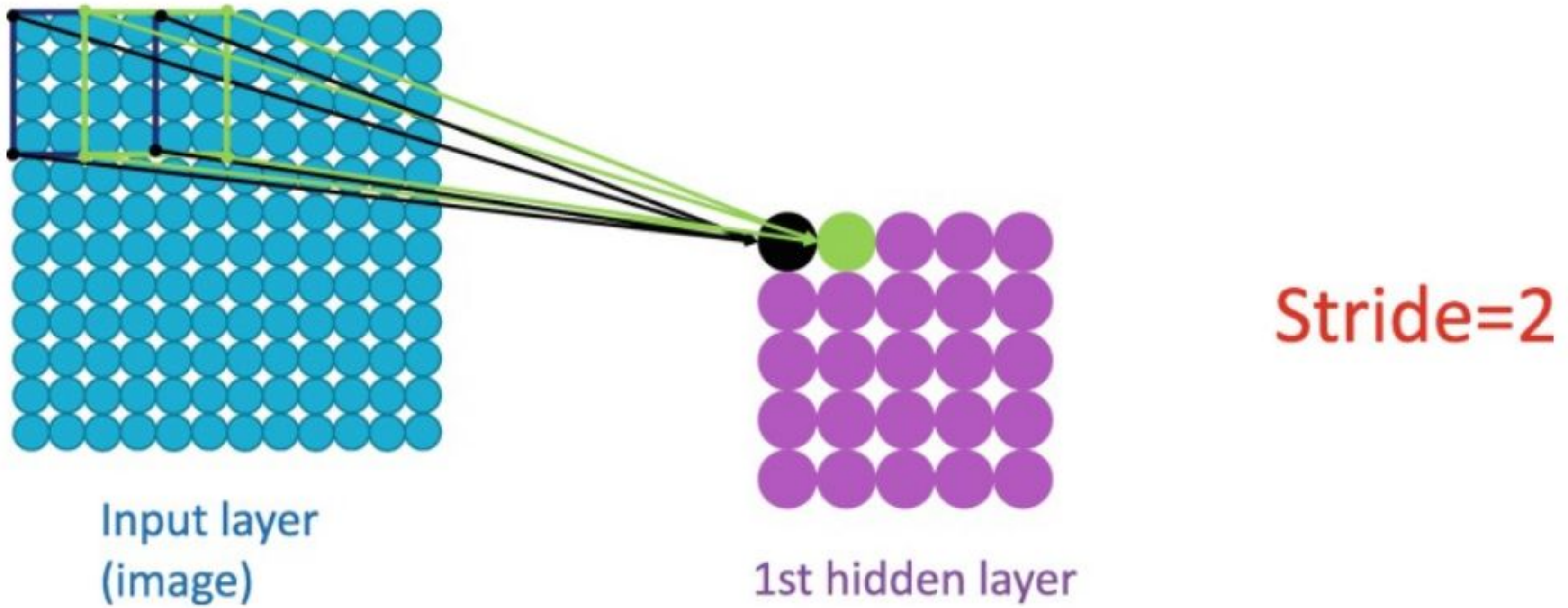
Application of the ReLU activation function, note the preservation of positive values

Many filters can be learned and combined



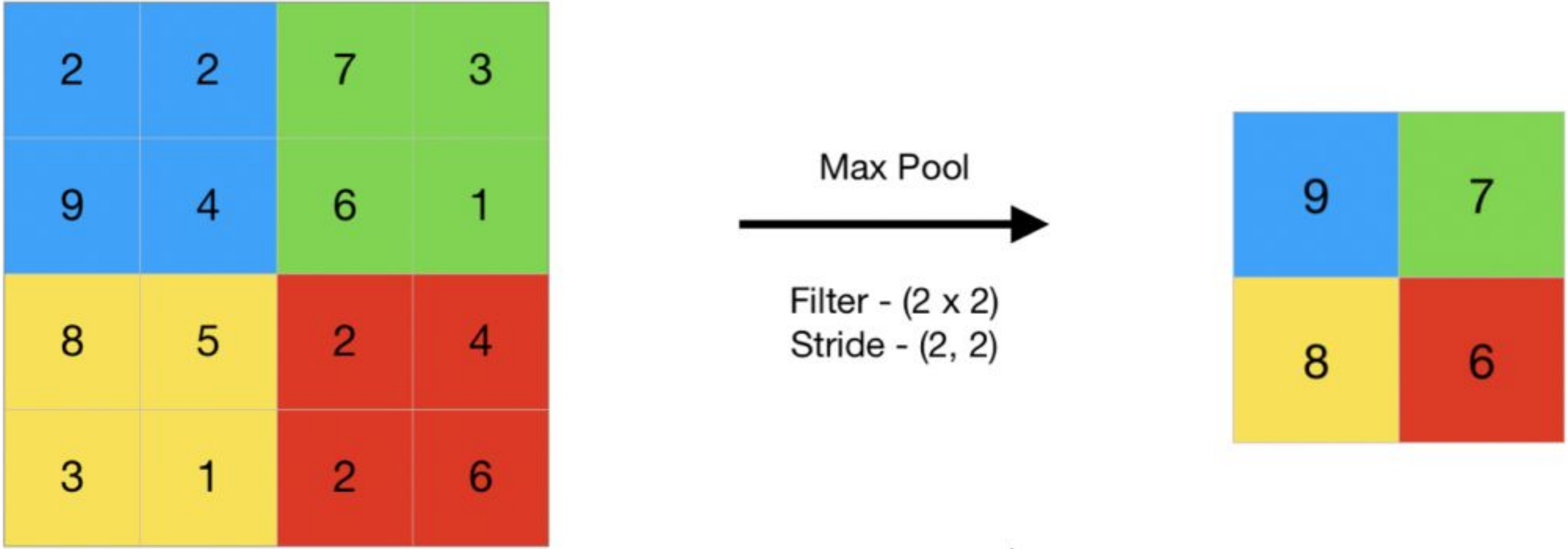
Each feature can detect different features of an image, creating mutiple feature maps

Pooling layers reduce the dimensions of feature maps



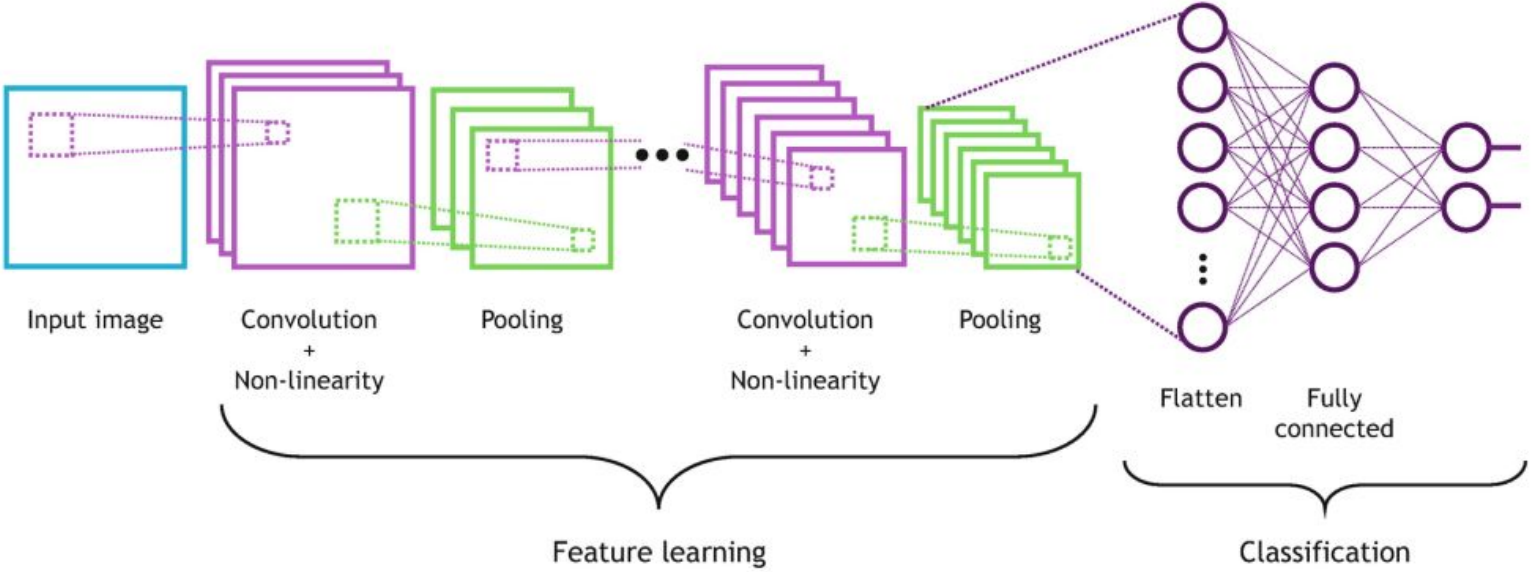
take the maximum (or average) value of elements in a filter with stride > 1

Pooling layers reduce the dimensions of feature maps



take the maximum (or average) value of elements in a filter with stride > 1

Basic CNN architecture

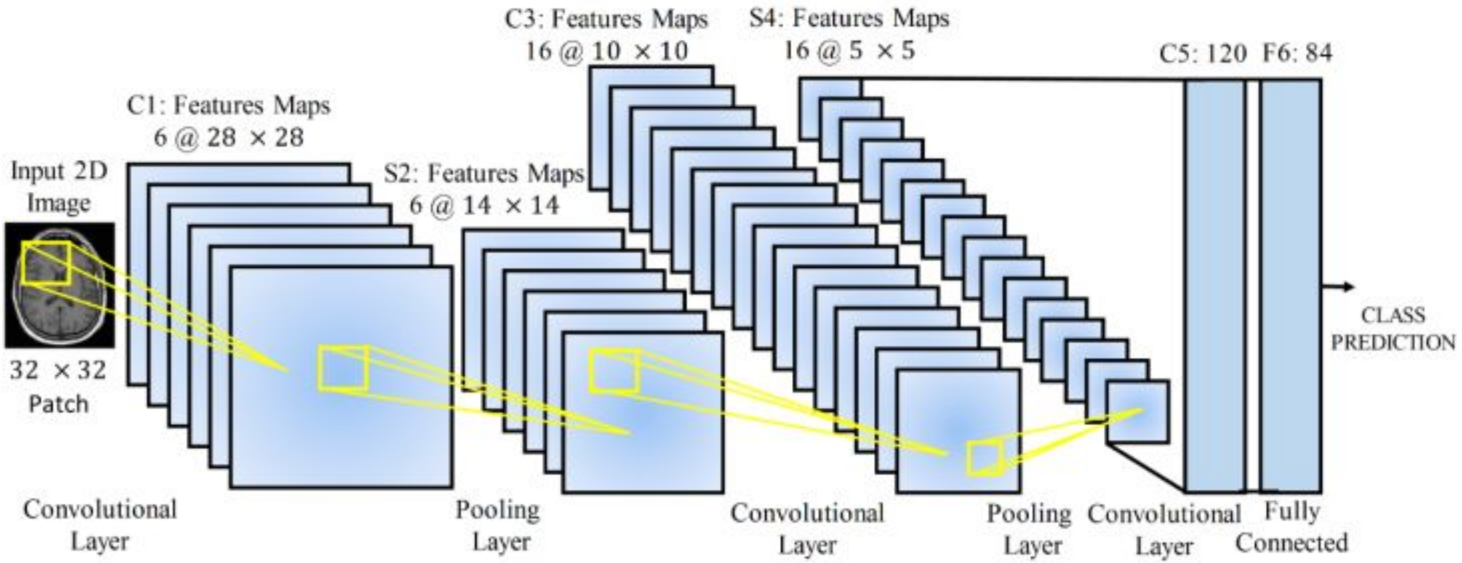


Combines convolutions for feature learning and classic multilayer networks for classification and regression

CNNs in R

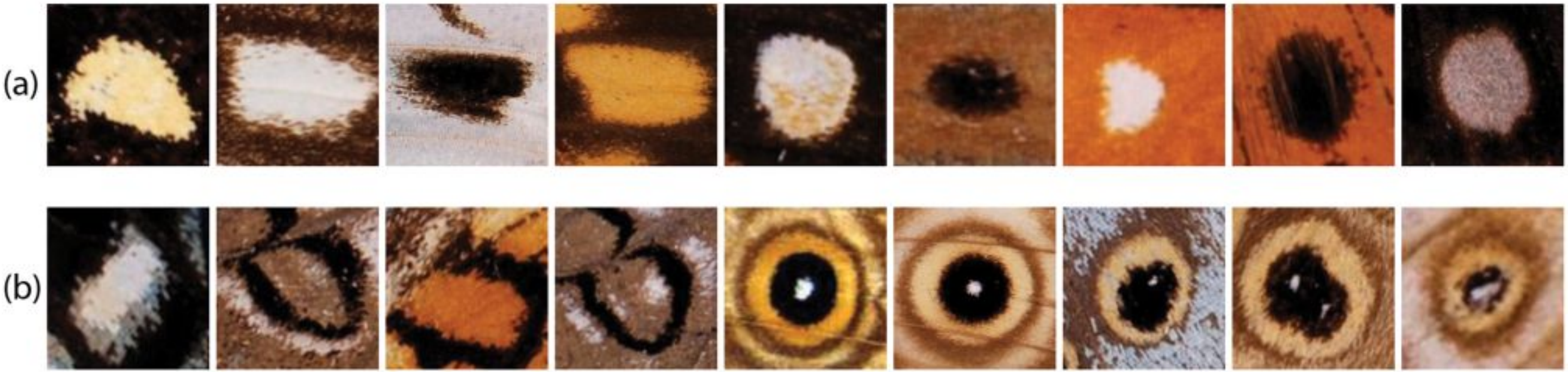
- You can fit CNNs in R using the `keras` package
- Other languages (e.g. Python) interface with `keras` more easily and with other packages for CNNs
- Process is somewhat involved, but not too crazy
- Here's an example for you to try if interested:
  - https://www.r-bloggers.com/2018/07/convolutional-neural-networks-in-r/

CNNs in biology- abnormality detection/disease classification/diagnosis



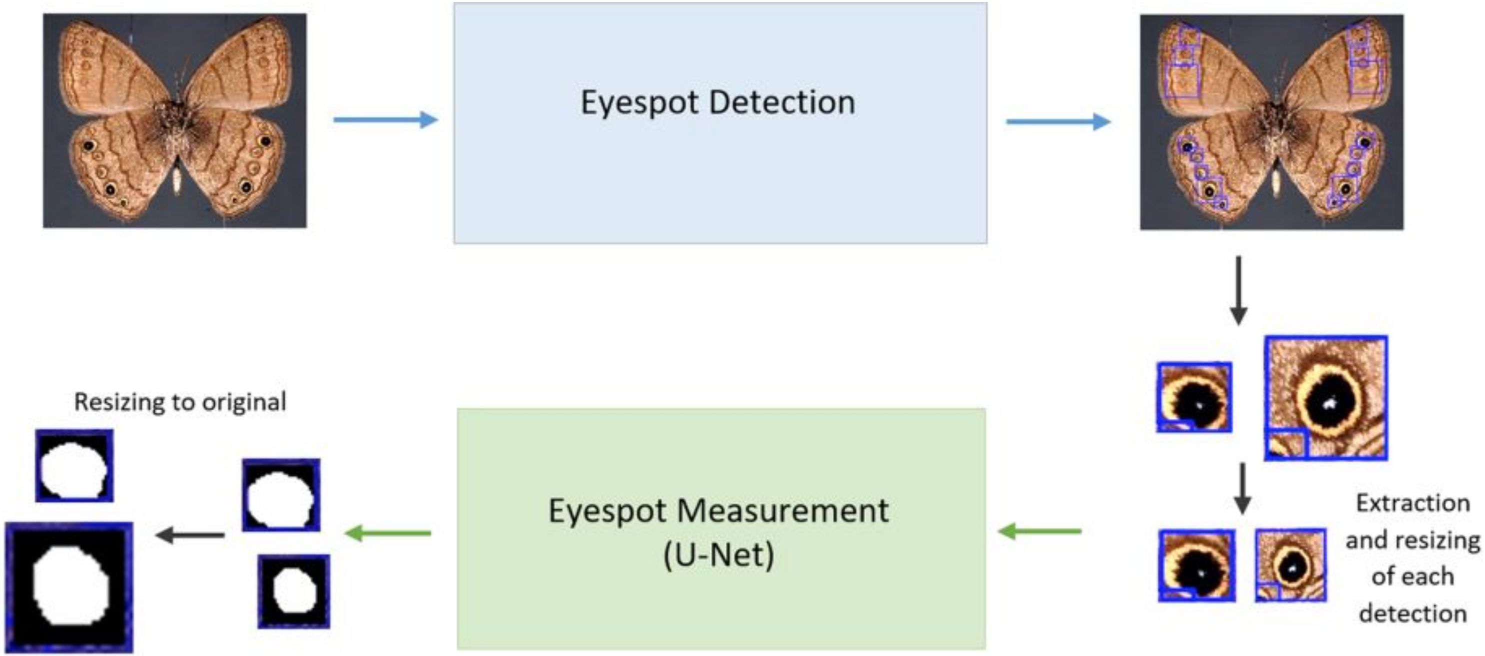
CNN architecture for medical image classification

CNNs in biology- identify/measure eyespots on butterfly wings



Examples of butterfly wing-pattern spots

CNNs in biology- identify/measure eyespots on butterfly wings



Uses 2 networks: one for detection and one for measurement

CNNs in biology

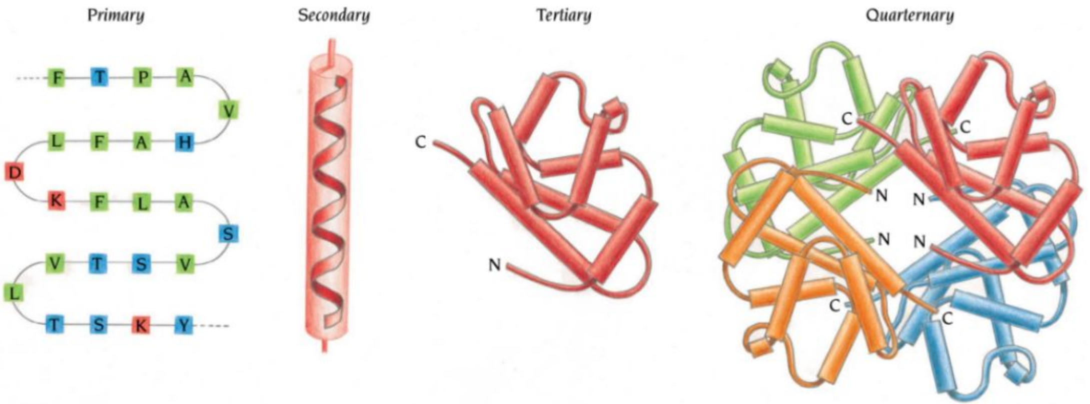
There are many other examples, but we could spend an entire course talking about this, so…

Protein folding problem

One of the biggest challenges in biology's recent history has been to predict the 3-dimensional structure of a protein from its amino acid sequence

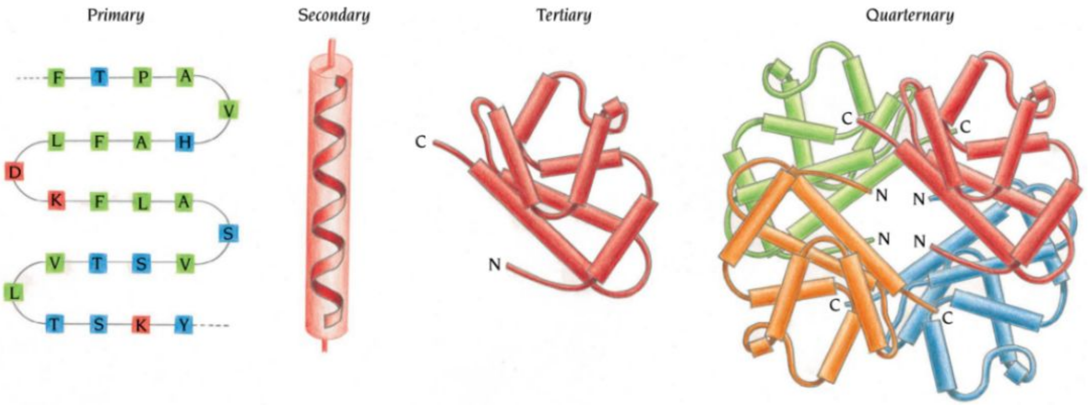
The protein folding problem

- ➢ Primary structure = amino acids held together by peptide bonds, can be determined via DNA sequencing
- ➢ Secondary structure = local motives. α-helixes and β-sheets held together by hydrogen nonds



The protein folding problem

- ➢ Tertiary structure = 3-dimensional shape of the protein, determined by interactions and bonds between protein side chains
- ➢ Quaternary structure = structure of proteins composed of multiple polypeptides or protein chains



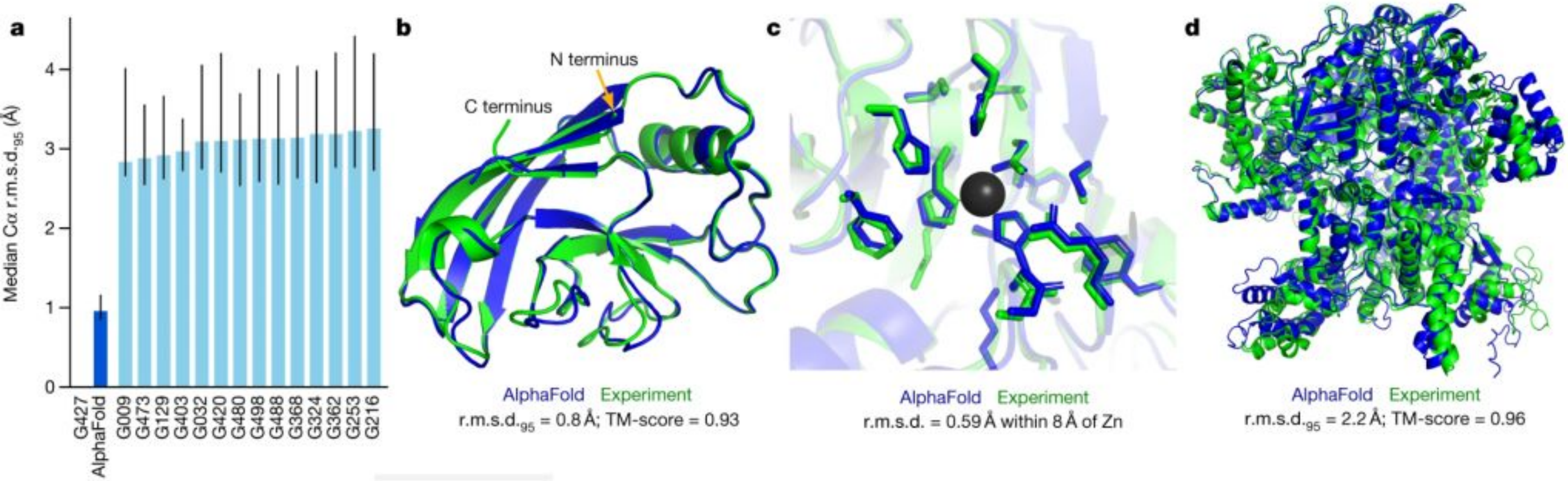
Protein folding problem

- Experimental determination of protein structure is expensive and time consuming
- Protein structure prediction is a major aim in computational biology
- Annual competition CASP (Critical Assessment of Techniques) since 1994

AlphaFold “solved” the protein folding problem

- DeepMind’s (from Google) AlphaFold, a powerful neural network, was introduced in the 2018 competition
- AlphaFold2 (with improvements) dominated the 2022 competition with near experimental accuracy

Performance relative to top competitors at CASP

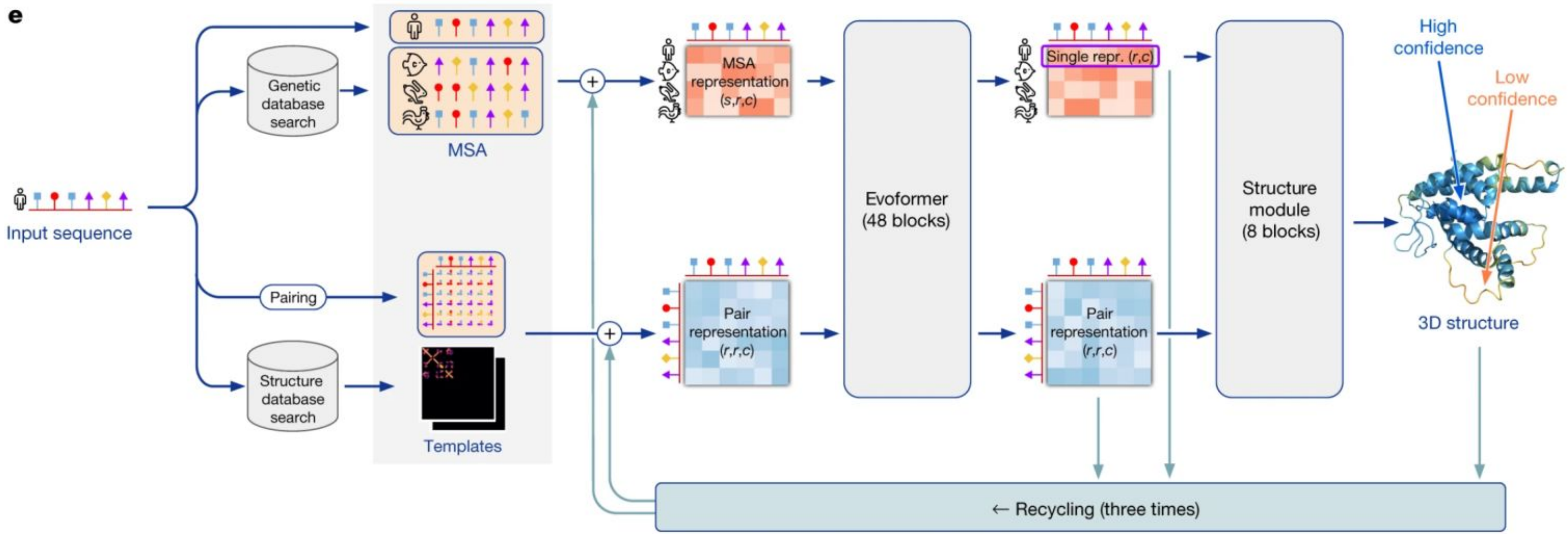


Major steps of AlphaFold

Alphafold consists of ___ major steps:

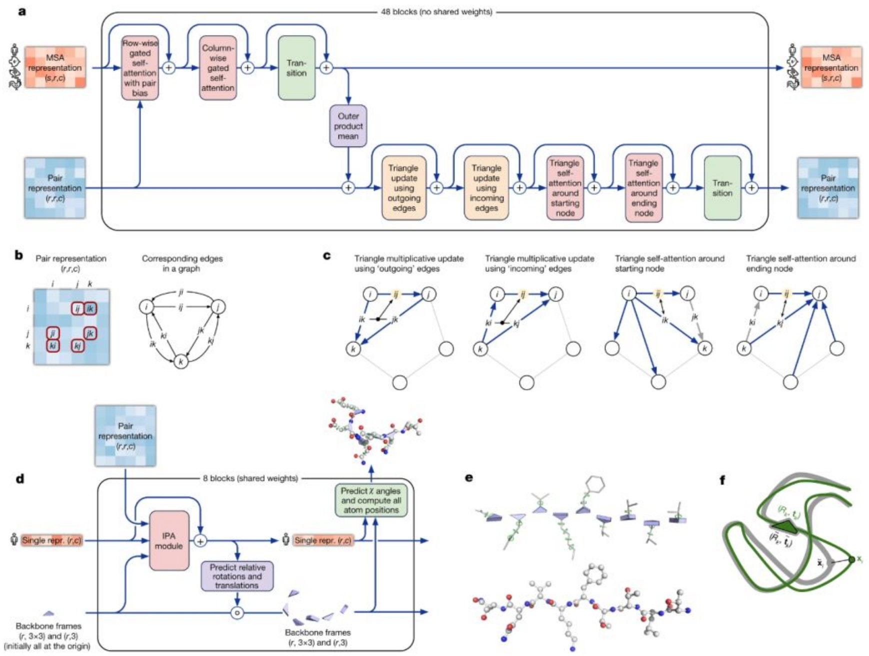
- ➢ Multiple sequence alignment (MSA) and pairwise distance matrix between residues for known homologs
- ➢ Evoformer neural network develops and refines structural representation of protein in 3D space, treating it as a graph problem

Architecture of AlphaFold network



Model architecture, includes MSA

AlphaFold approach



Evoformer NN treats structure prediction as a graph problem (note step c)

Alphafold outputs predictions in hours to days

https://github.com/google-deepmind/alphafold/blob/main/imgs/casp14_predictions.gif

Programming Projcet 6

Programming project 6 involves predicting 3D protein structure with AlphaFold