Neural Networks

Terminology

- Inspired by biology
- Better picture: nested linear algebra functions, fully parametrized
- Layers: Differentiable functions on previous intermediate results with free parameters for training
- Deep learning: more layers

Key ingredients

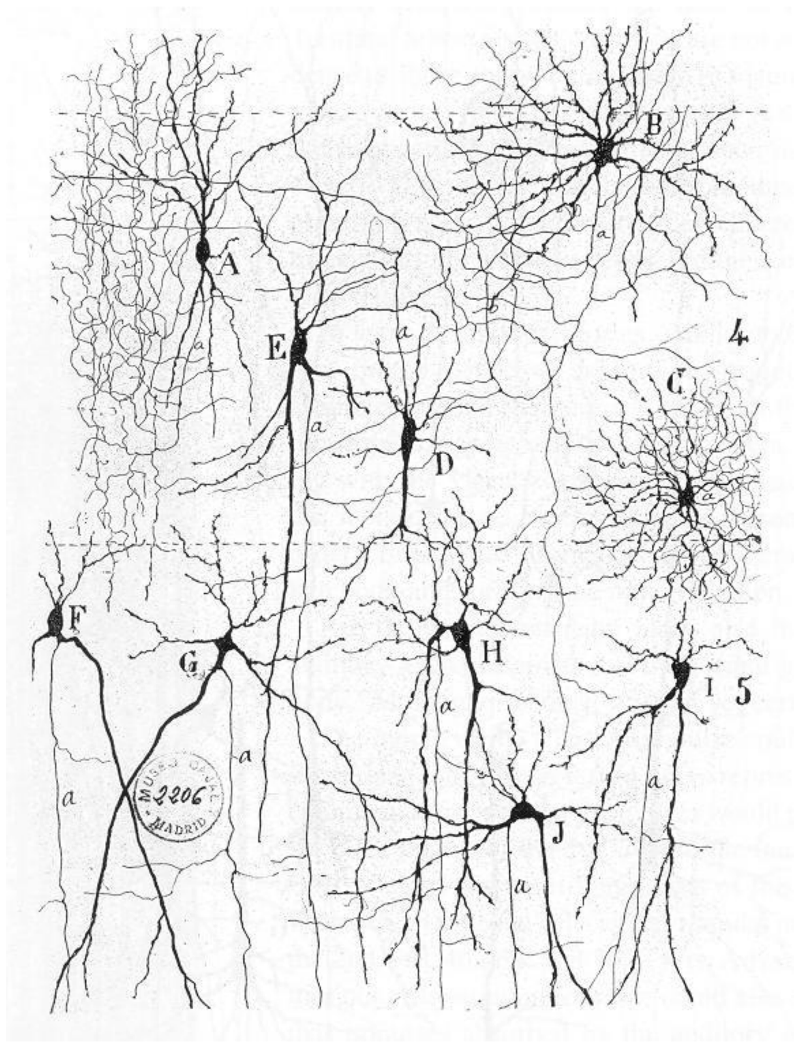
- Non-linear function
- Linear model
  - w: some weights
  - b: bias

$$g(\mathbf{X})$$

$$\hat{y} = \mathbf{w}g(\mathbf{X}) + \mathbf{b}$$

Universal Approximation Theorem

Given enough layers and width, any function can be represented.



Layer types

Convolution layer: includes a fixed neighborhood
- Weighted combination of close entries, fixed weights

Pooling layer: reduce dimensionality
- Aggregate (typically max) values

Dense layer: general use

$$\underbrace{\sigma}_{\text{Activation function}}(\underbrace{\mathbf{W}}_{\text{Weights}}\overbrace{\mathbf{x}}^{\text{Input}}) + \underbrace{\mathbf{b}}_{\text{Bias}}$$

Layer ingredients

Activation function
- Differentiable
- Non-linear
- Hyperparameter
- Common: rectified linear unit (ReLU) = max(0, x)

Weights
- Matrix, free parameters
- Shape determines dimensionality of output

Bias
- Vector, free parameters

$$\underbrace{\sigma}_{\text{Activation function}}(\underbrace{\mathbf{W}}_{\text{Weights}}\overbrace{\mathbf{x}}^{\text{Input}}) + \underbrace{\mathbf{b}}_{\text{Bias}}$$

Layers with are no layers

Hidden layer: anything the result of which is not inspected by the user

Activation layer: Explicit application of the activation function

Dropout layer: stochastic method of ignoring features to reduce overfitting

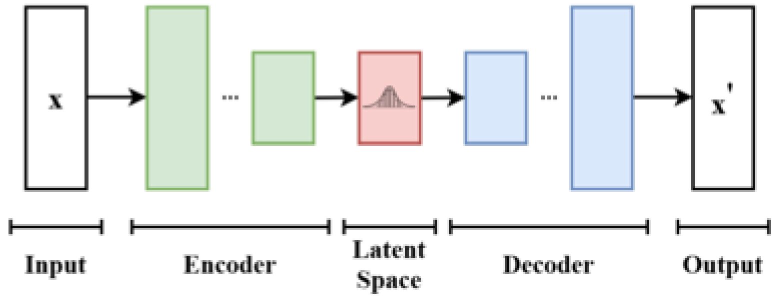
Input layer: tensor of input features

Output layer: tensor (often vector) of observables or probabilities

Latent space

Implicitly learned representation
- Other methods: require features
- NN: implicitly find one at the cost of complexity and data

Use cases
- Generate more similar systems / elements of similar properties



Common types of networks

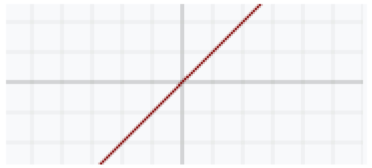
Graph neural network (GNN): Input is a graph

Convolutional neural network (CNN): Uses at least one convolution, often for image-like data

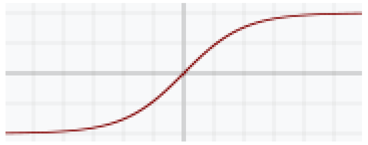
Autoencoder: small layer somewhere in the network (everything before: encoder, everything after: decoder)

Common types of activation functions

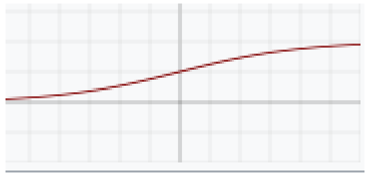
Step



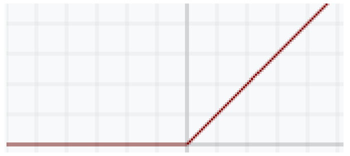
Tanh



Logistic



Rectified Linear Unit (ReLU)



Vanishing gradient
Is the function almost constant anywhere?

Cost
Can be evaluated quickly?

Normalisation
Is there a finite output domain?

Differentiable
Otherwise hard to optimize

Summary Neural networks

- Chained, heavily parametrized functions
- Applied in order („layers“)
- Implictly find a representation
- Somewhat intransparent
- Need to be trained iteratively