FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Dimension Reduction & Introduction to Neural Networks

LIAD MAGEN

Review – Approaching ML Project

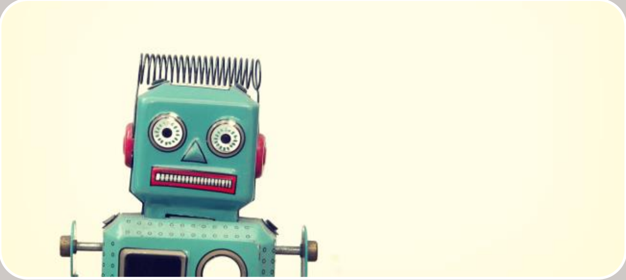
- Explore the data (always look at the data)
  - Formulate questions – and find their answers
  - Plot the data
- Plan experiments
  - Decide which features should be used
  - Decide on the models
- Perform Experiments
- Optimize Hyperparameters
- Use the test set to choose the best model

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Agenda

- Dimension Reduction:
  - Principle Component Analysis (PCA)
  - T-SNE
  - U-MAP
  - pyMDE
- Introduction to Neural Networks
- Word2Vec

The Big Picture

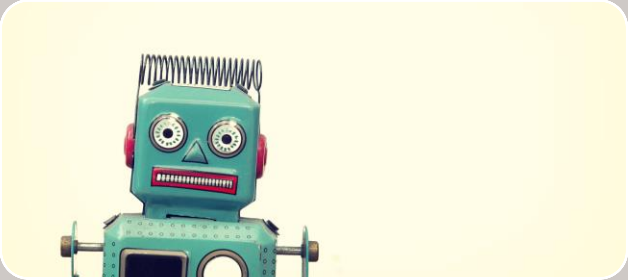


Supervised
- Decision Tree
- Random Forest
- Logistic Regression
- Naïve Bayes
- K-Nearest Neighbor
- Support Vector Machine
- Neural Networks

Unsupervised
- Latent Dirichlet Allocation
- K-Means
- Dimension Reduction
  - PCA

Reinforcement Learning

The Big Picture



Supervised

- Decision Tree
- Random Forest
- Logistic Regression
- Naïve Bayes
- K-Nearest Neighbor
- Support Vector Machine
- **Neural Networks**

Unsupervised

- Latent Dirichlet Allocation
- K-Means
- Dimension Reduction
  - **PCA (Principal Component Analysis)**

Reinforcement Learning

Dimension Reduction

> We're given a dataset (e.g., documents)
> How can we visualize them (e.g., plot items by their label) in a single, 2-D chart?

> Method #1: Common words

> Method #2: One-hot-encoding bag-of-words … ?

Dimension Reduction

> A one-hot-vector / CountVector has the size of the vocabulary. E.g., 20,000

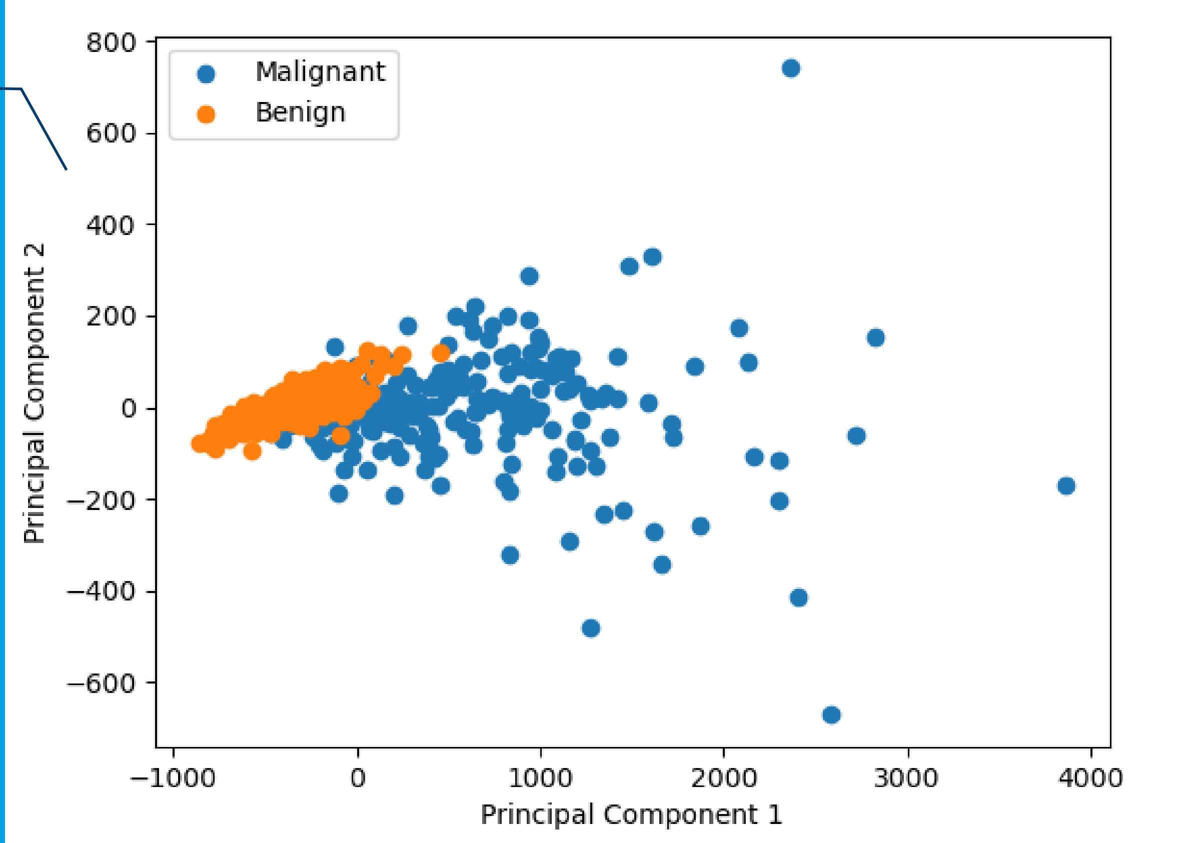
> The dimension of a chart is 2 to 3.

> Can we squeeze a 20,000 vector into 2?

> Can we squeeze m documents – $20{,}000 \times m$ into a $2 \times m$ matrix for visualization?

Principal Component Analysis (PCA)

The result of the PCA on the data from the cells found in breast growths. **30** features reduces to **2**



Principal Component Analysis (PCA)

How does it work?

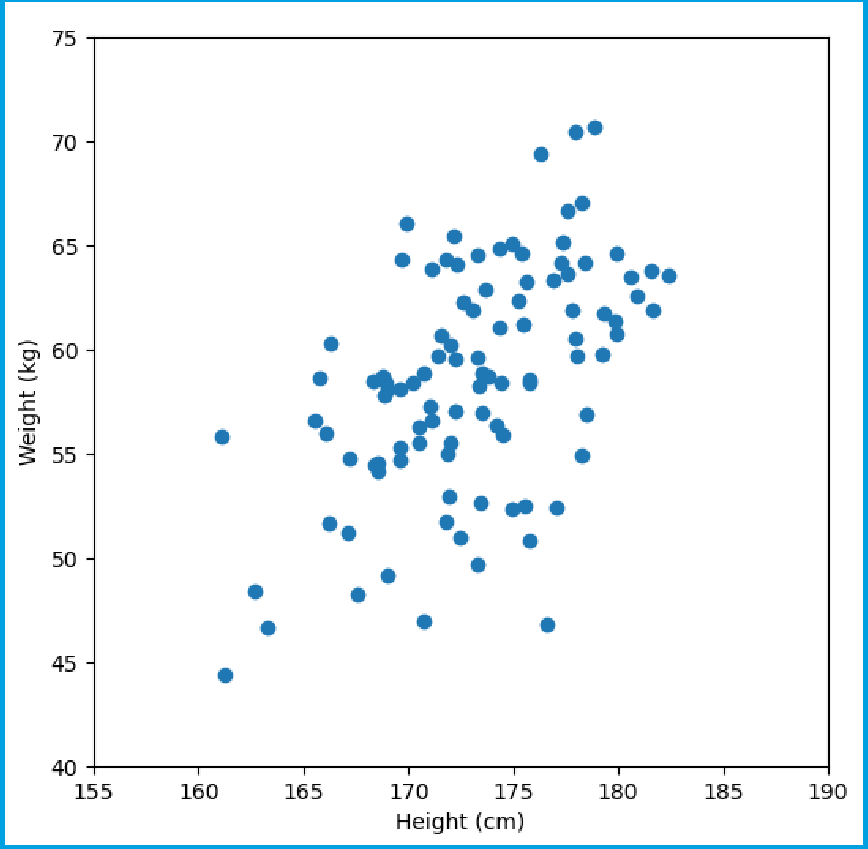
With a lot of math!

Main idea:
Detect and return the *eigenvectors* of the *covariance matrix*.

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

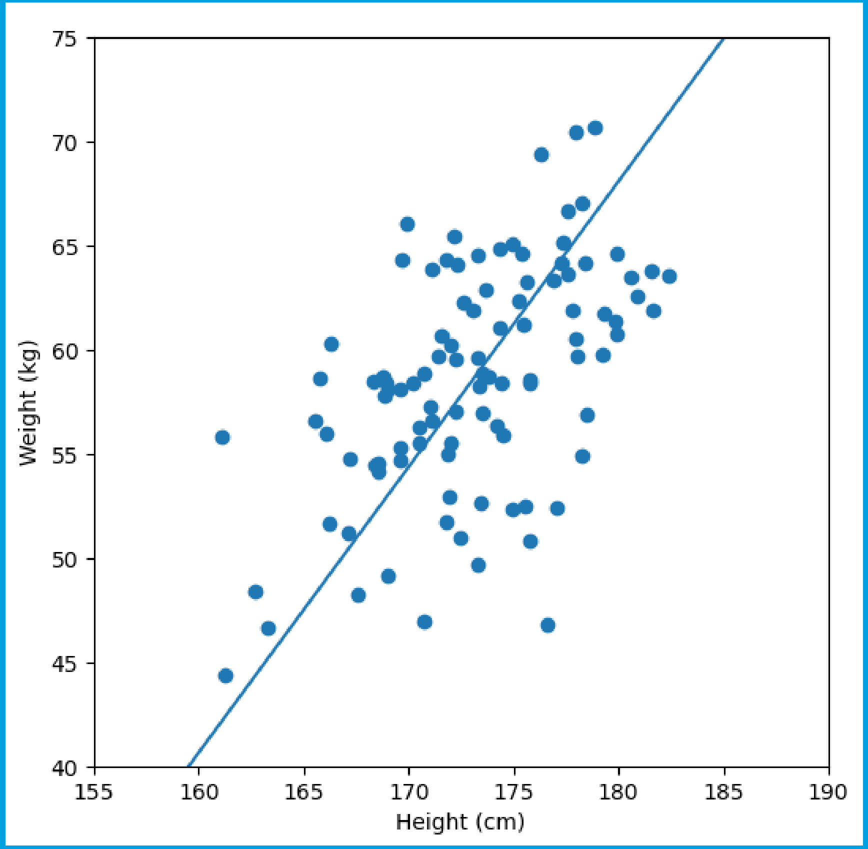
Principal Component Analysis – Example: 2-D to 1-D

Height vs Weight of 50 People



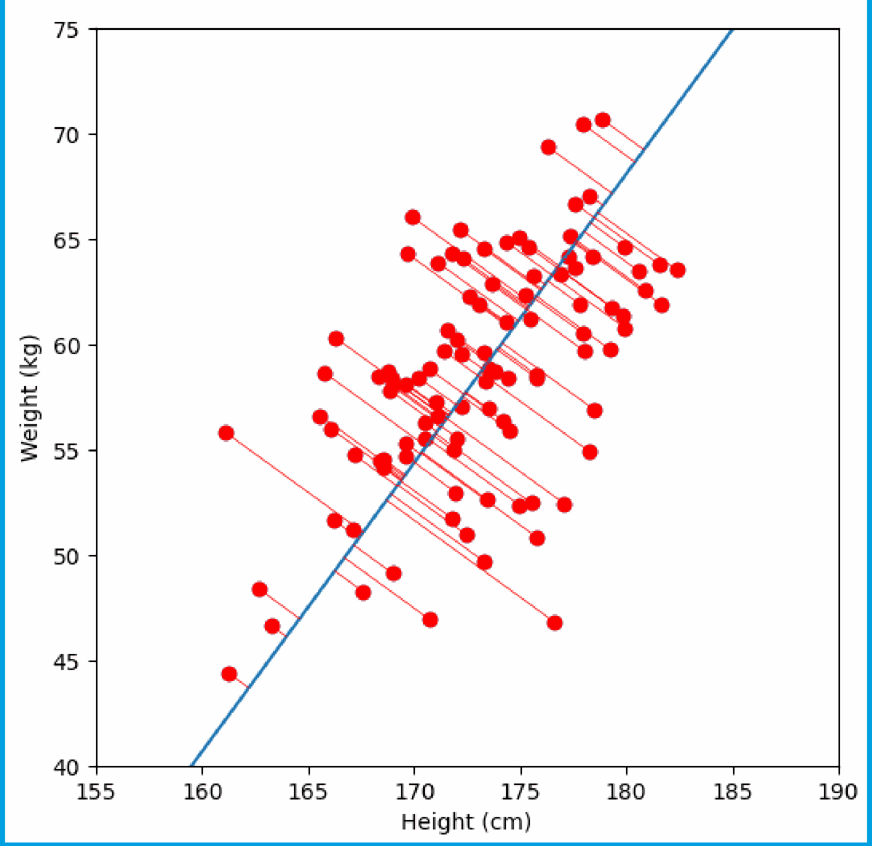
Principal Component Analysis – Example: 2-D to 1-D

Best fitted line (the axis of greatest variation).



Principal Component Analysis – Example: 2-D to 1-D

“Projecting” the points onto the line (aka first principal component).



Every point is mapped into a single number:

its *distance* from the line.

Principal Component Analysis – Covariance Matrix

> $cov(x,y) = \frac{\sum_{i=0}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1}$ (subtract the mean from each value)

> Covariance matrix for 3 values: A, B, C

$$\begin{matrix} cov(A,A) & cov(A,B) & cov(A,C) \\ cov(B,A) & cov(B,B) & cov(B,C) \\ cov(C,A) & cov(C,B) & cov(C,C) \end{matrix}$$

Notes:

> Covariance measures the directional relationship between two variables.

> Along the diagonal is the *variance*

Additional Read (For deeper understanding)

- https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202
- Principal Component Analysis (PCA) For Dummies | Bill Connelly

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Additional Dimension Reduction Methods

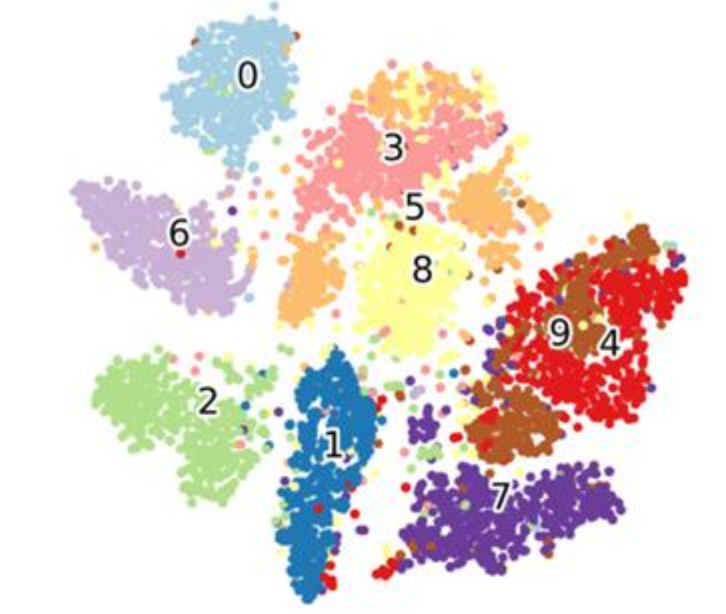
> SVD (Singular Value Decomposition)
  Breaking a matrix into factors
  ($12 \rightarrow 3 \times 4$ // $3 \times 2 \times 2$ // 2.4×5)

> T-SNE - T-distributed Stochastic Neighbor Embedding

> U-MAP - Uniform Manifold Approximation and Projection for Dimension Reduction
  python package: **umap-learn**

> Phate – (more for biological data)

> PyMDE - Minimum-Distortion Embedding



Resources

- sklearn.decomposition.PCA — scikit-learn 1.2.0 documentation
- sklearn.manifold.TSNE — scikit-learn 1.2.0 documentation
- UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction — umap 0.5 documentation (umap-learn.readthedocs.io)
- PyMDE: Minimum-Distortion Embedding — pymde 0.1.15 documentation

Take-Aways

- There are methods to reduce the dimensionality of the data which can be used to:
  - Visualize the data and plot it
  - Clustering
  - Preprocessing step before a ML Classifier
- These methods are based on a mathematical representation of the data
- The representation is relatively faithful to the original data
- But… may suffer from missing information

Introduction to

Neural Networks

Basic Machine Learning for NLP

- N-Grams
- Bag-of-Words
- Word-Classes (WordNet, Stemming, Lemmas)

- Unsupervised Dimensionality Reduction (PCA)
- Unsupervised Clustering (K-Means, LDA)
- Supervised classification (Logistic Reg, SVM, Naïve Bayes,…)

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Issues with ML for NLP

- Bag-of-words:
  - Sum of one-hot codes
  - Ignore the word order
- N-gram Language Models
  - Probabilities are estimated from counting on large corpus
  - Smoothing is used to prevent unseen events (OOV) → Zero probabilities.
- Word Classes
  - Similar words should share parameter estimation
  - Require an external, labeled, dataset (hard to obtain, single-lingual).

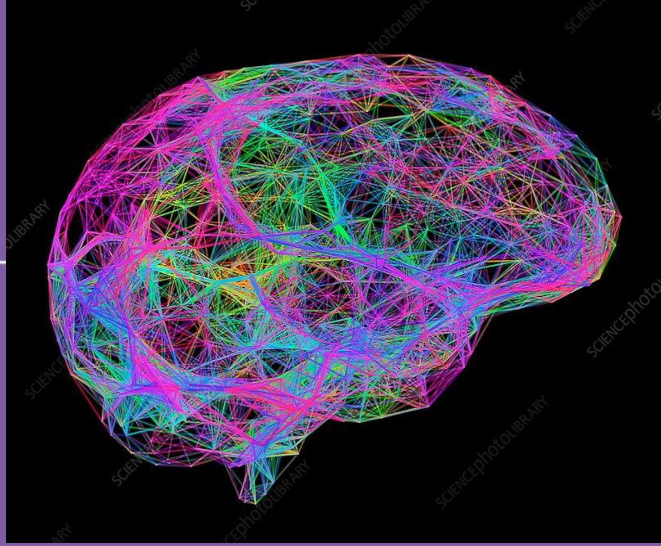
Neural Networks: NLP Motivation

> We would like to have better techniques than plain word-counting.

> A method that can learn on its own, without too much need of a specialized person.

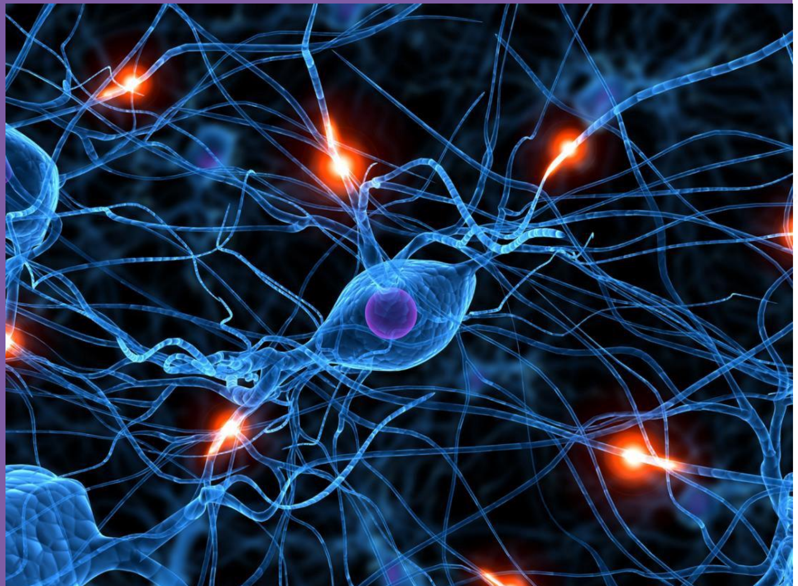
> (Are we there yet?)

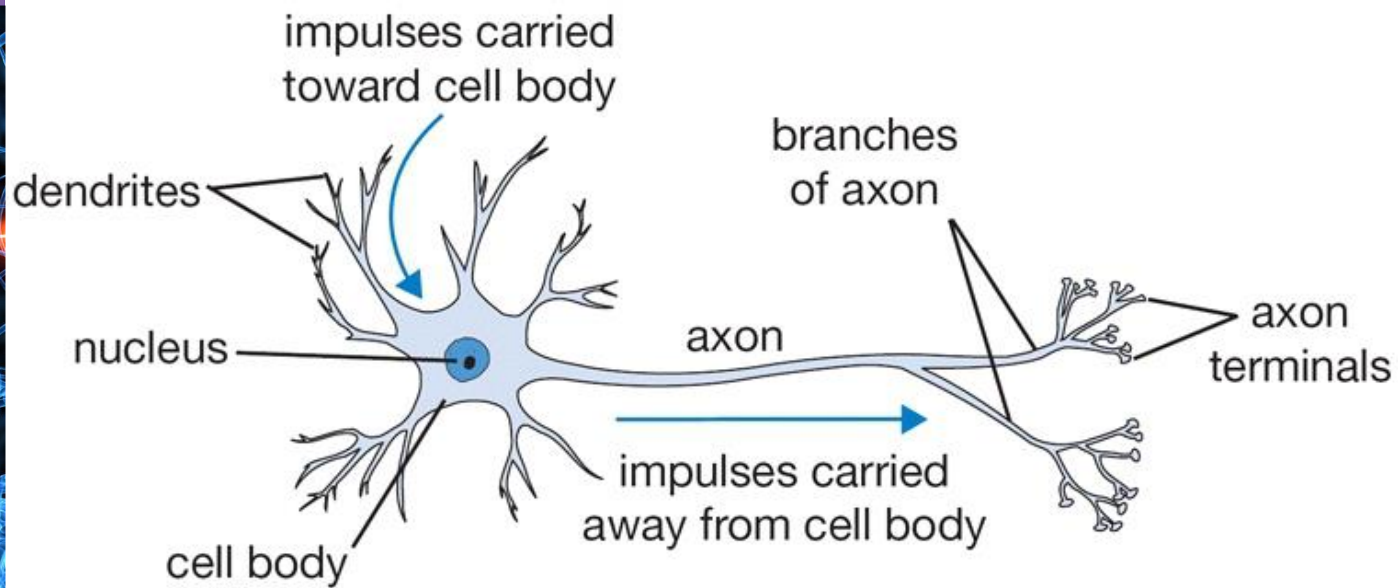
FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES



Biological Motivation

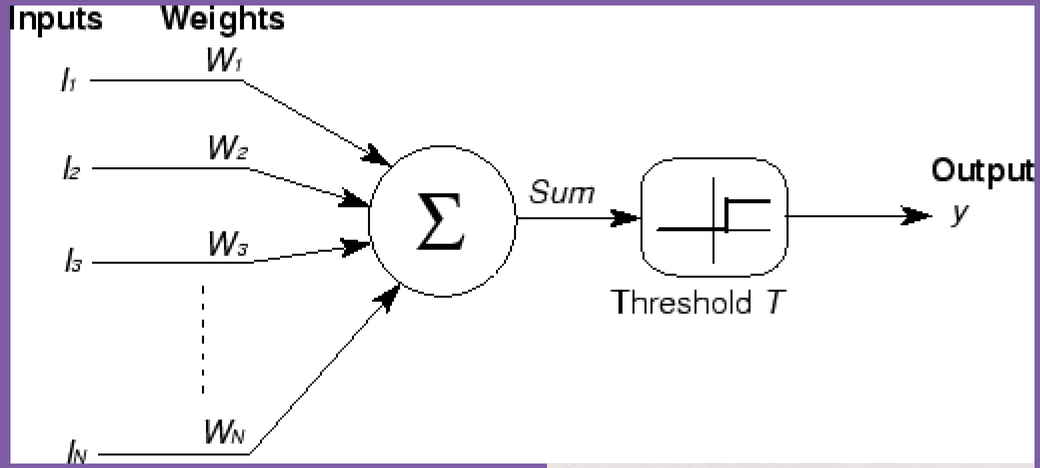
> There are ~86B neurons in our brain
> Neurons are connected to other neurons through an axon
> The structure resemble to a network, where information is passed from one neuron to another



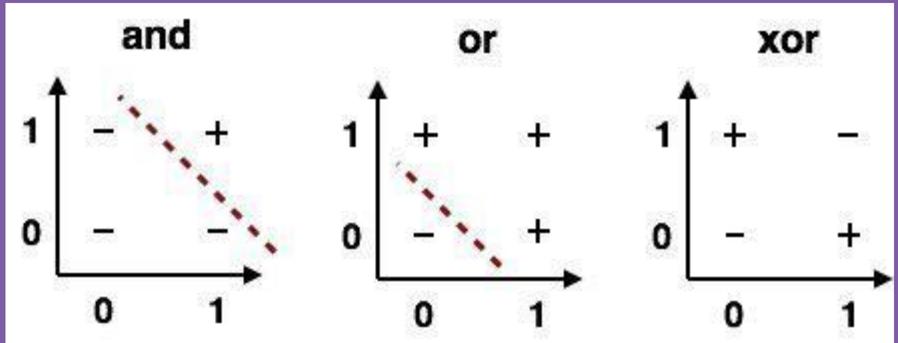


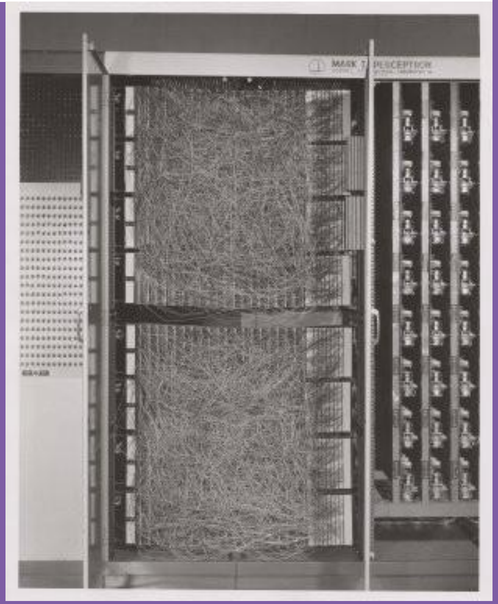
FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

From Biology to Computation

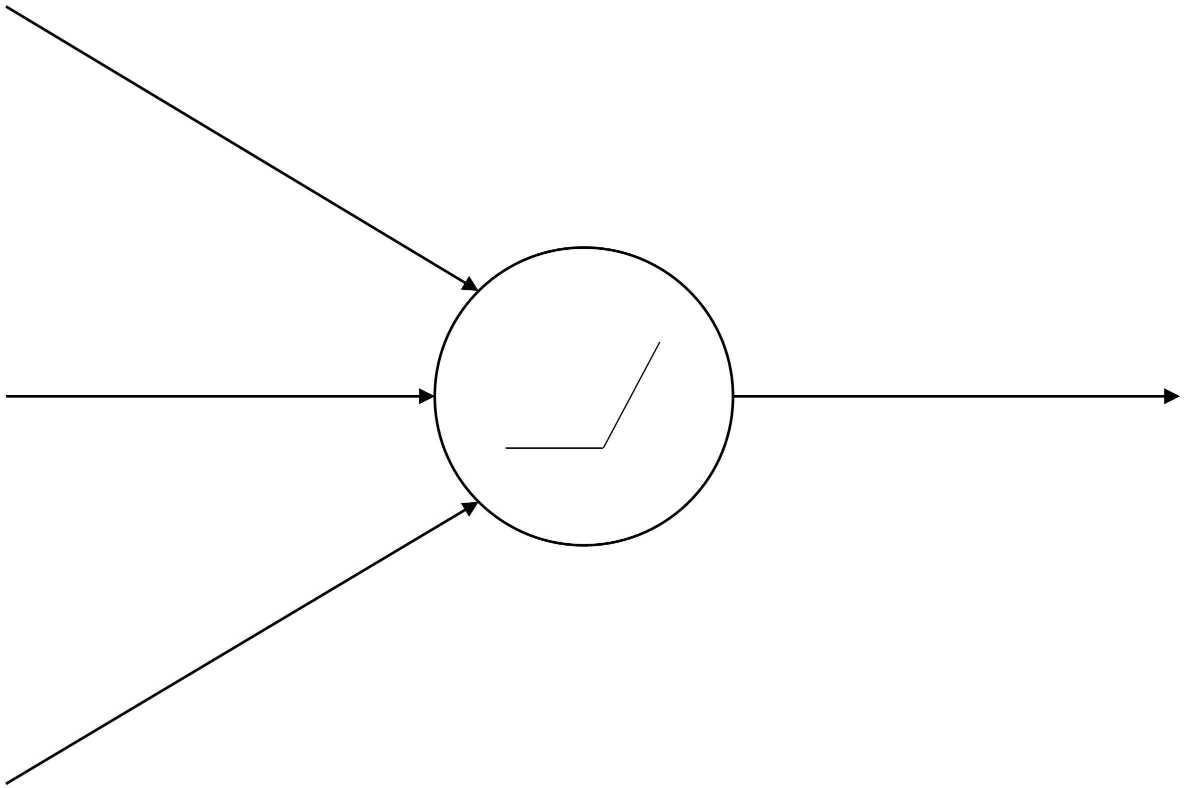


- > 1943 – McCulloch-Pitts neuron
  A linear threshold gate
- > 1958 – The Perceptron
  The weights were learned through data-examples
- > Only capable of simple linear decision boundaries



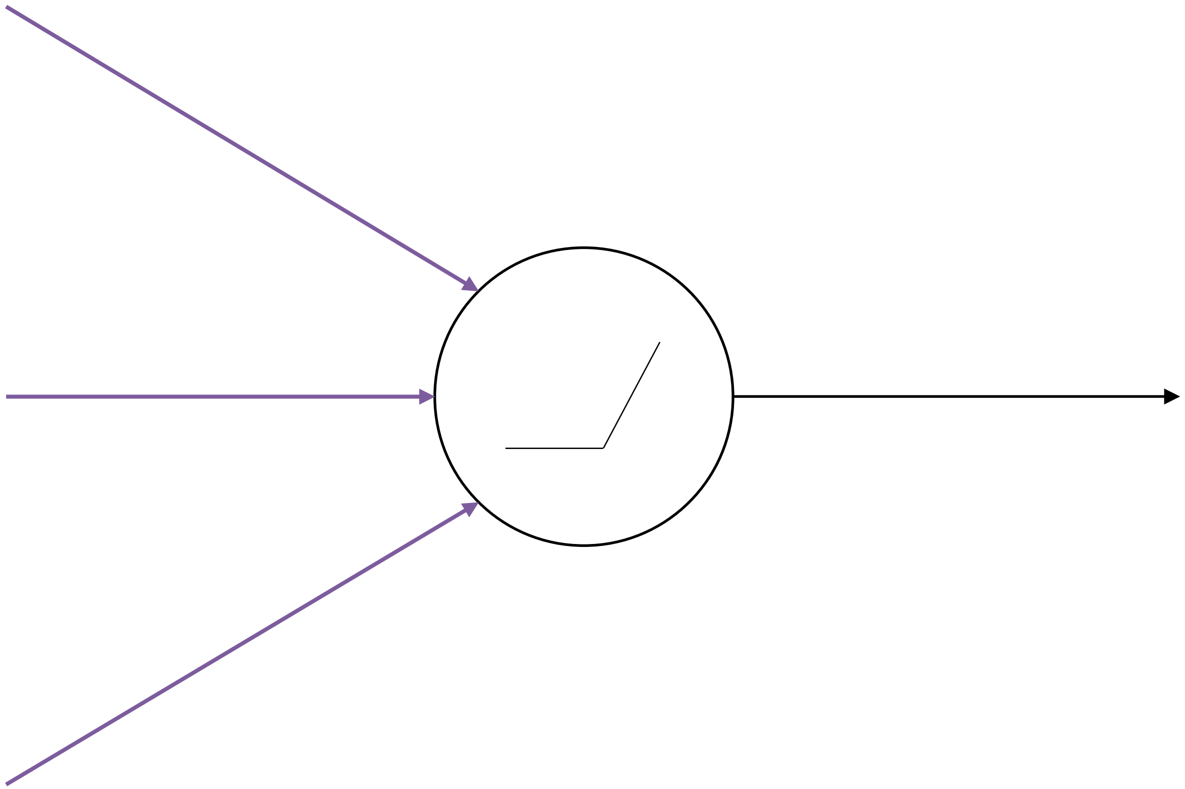


Neuron (Perceptron)



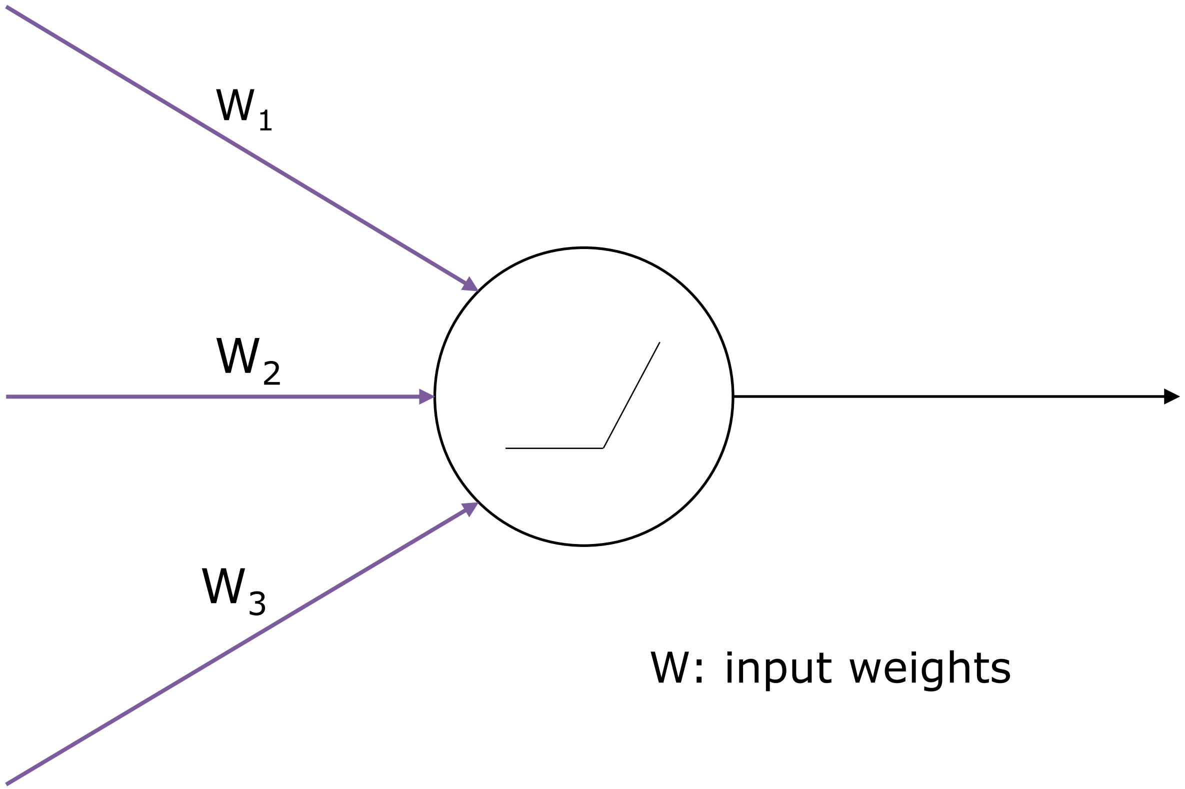
Neuron (Perceptron)

Input synapses



Neuron (Perceptron)

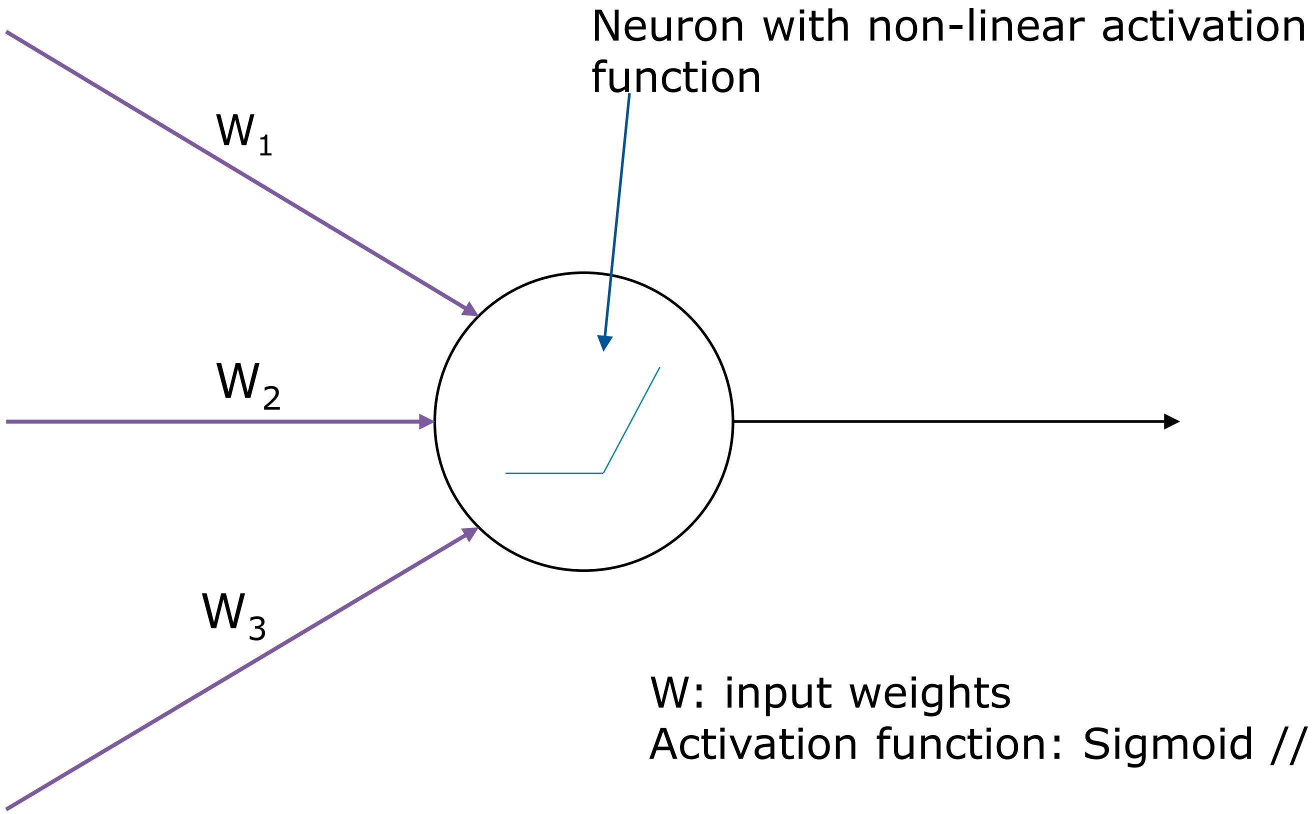
Input synapses



W: input weights

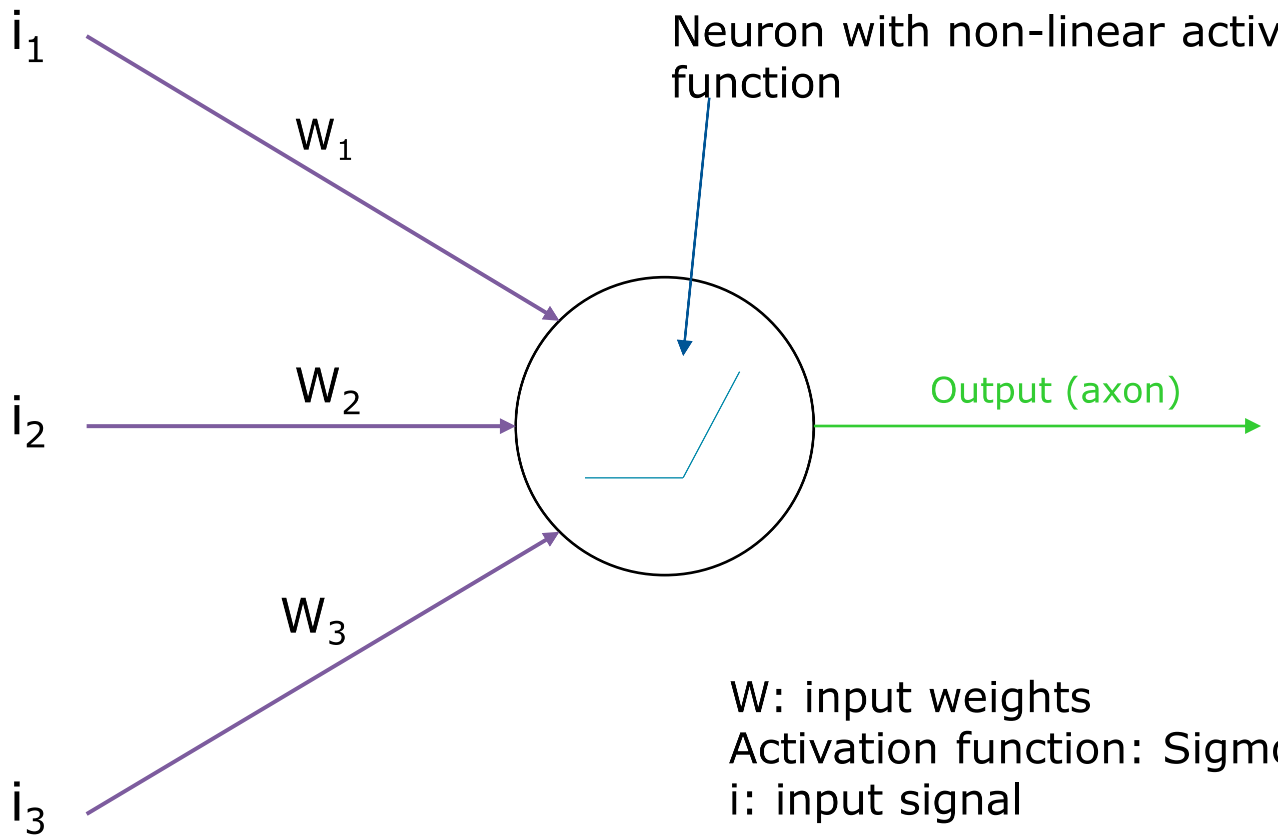
Neuron (Perceptron)

Input synapses



W: input weights
Activation function: Sigmoid // max(0, value)

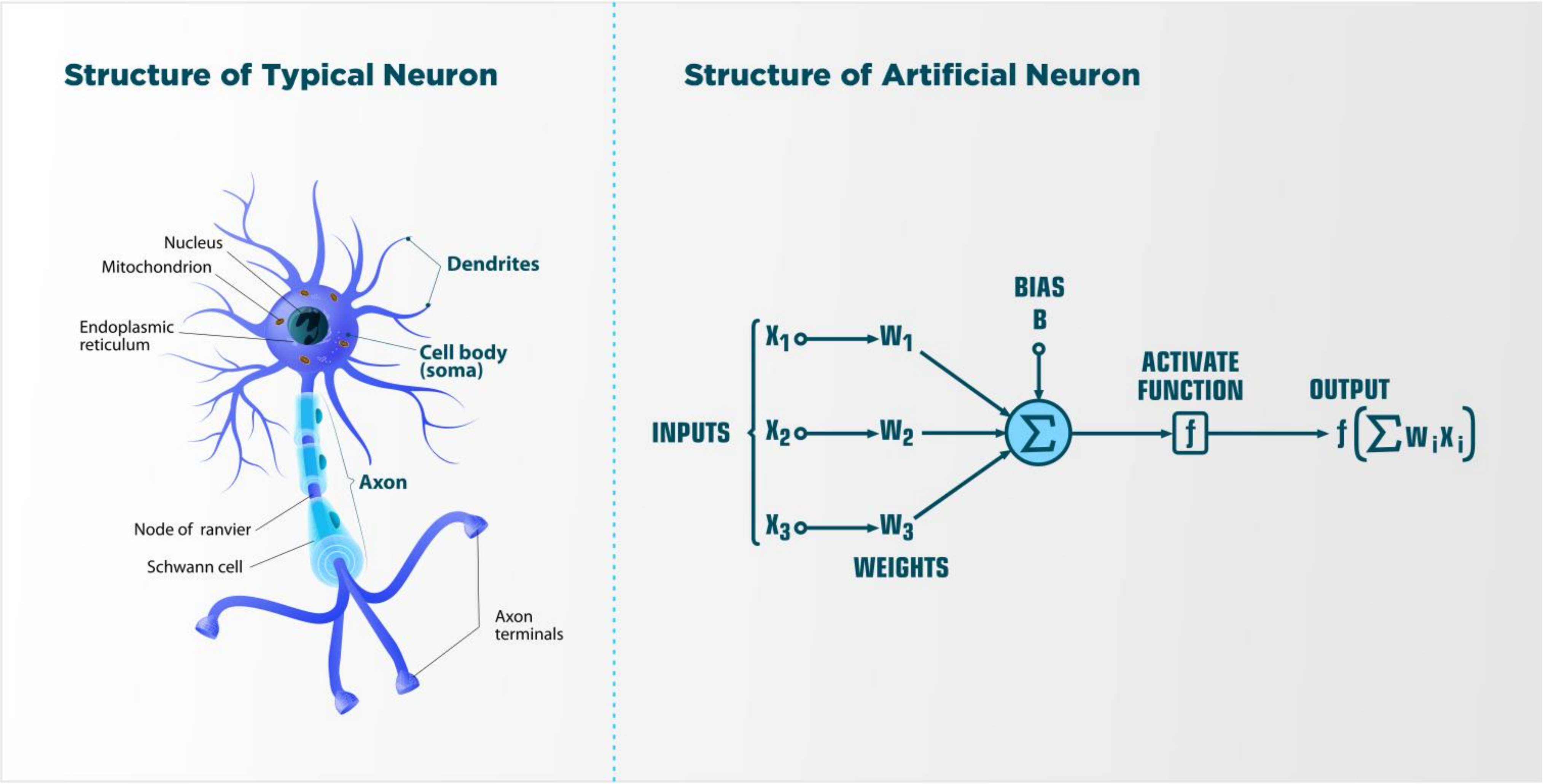
Neuron (Perceptron)



W: input weights
Activation function: Sigmoid // max(0, value)
i: input signal

$$Output = \max(0, I \cdot W)$$

Perceptron vs Artificial Neuron

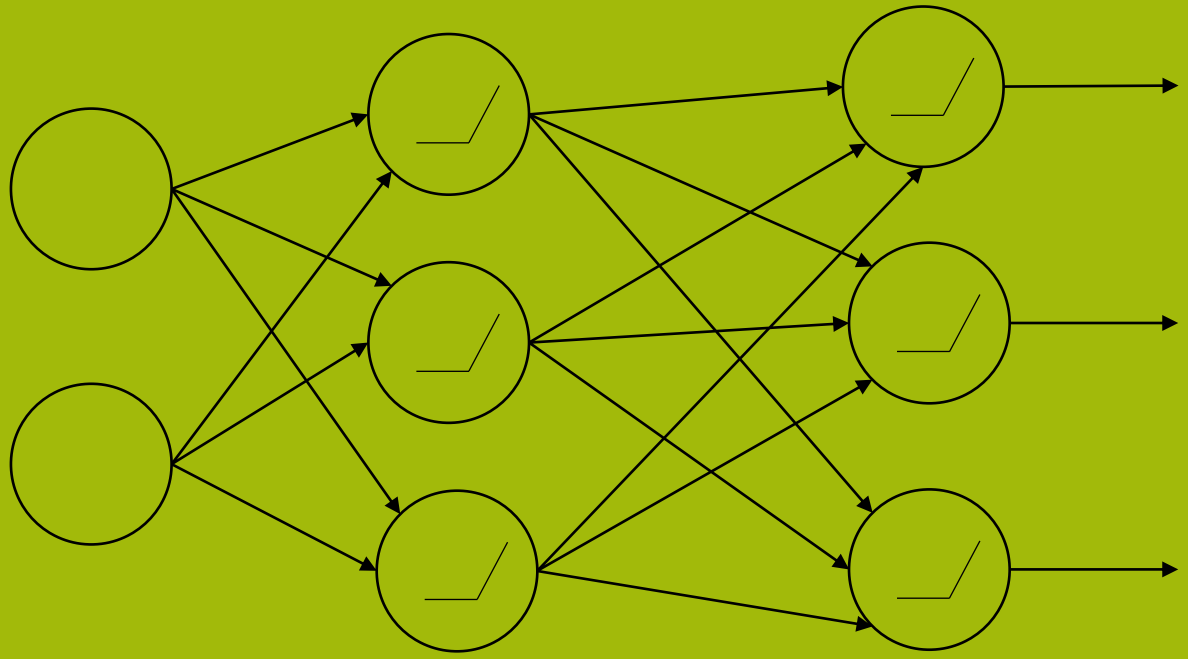


Neuron (Perceptron)

> The perceptron model is quite different from the biological neurons:
  > Those communicate by sending spike signals at various frequencies
  > The learning in brains seems also quite different

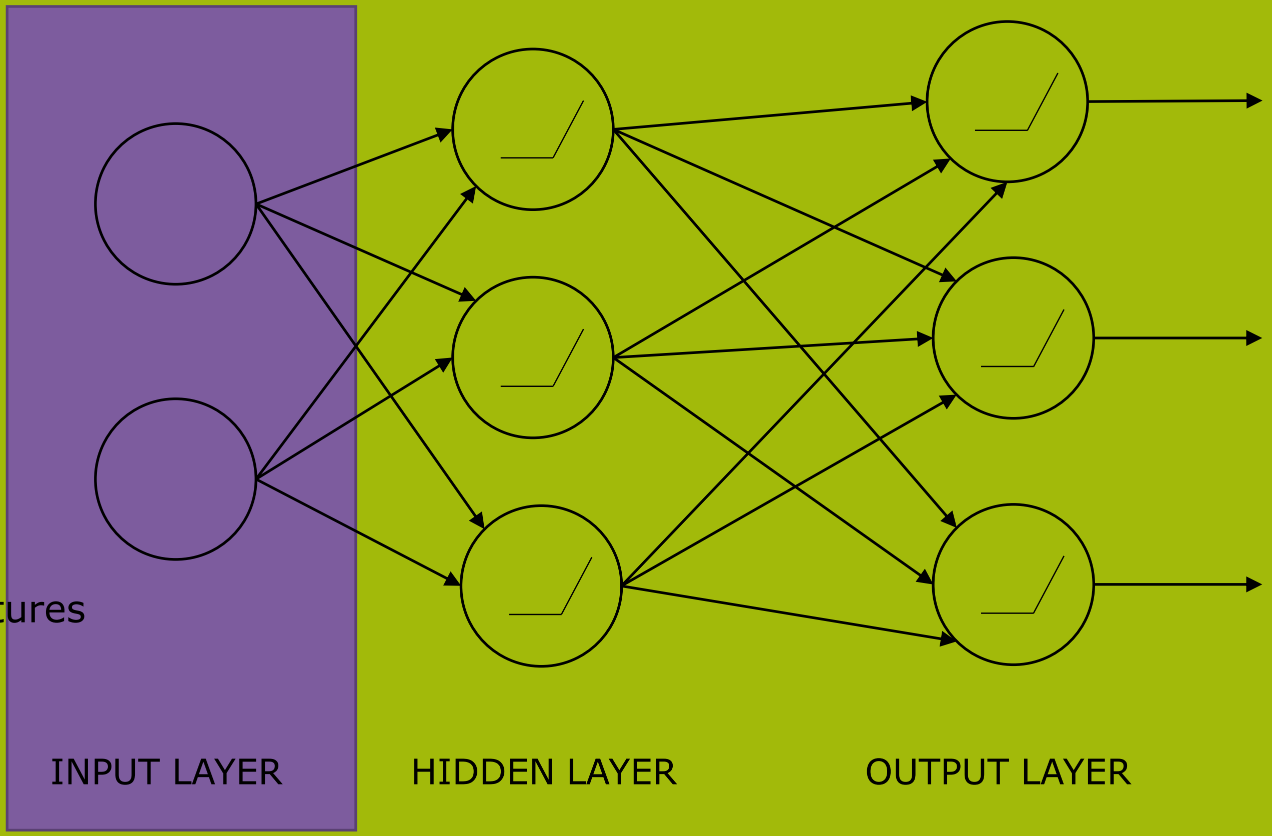
> It would be better to think of artificial neural networks as non-linear projections of data (and not as a model of brain)

Neural Network Layers



INPUT LAYER HIDDEN LAYER OUTPUT LAYER

Neural Network Layers



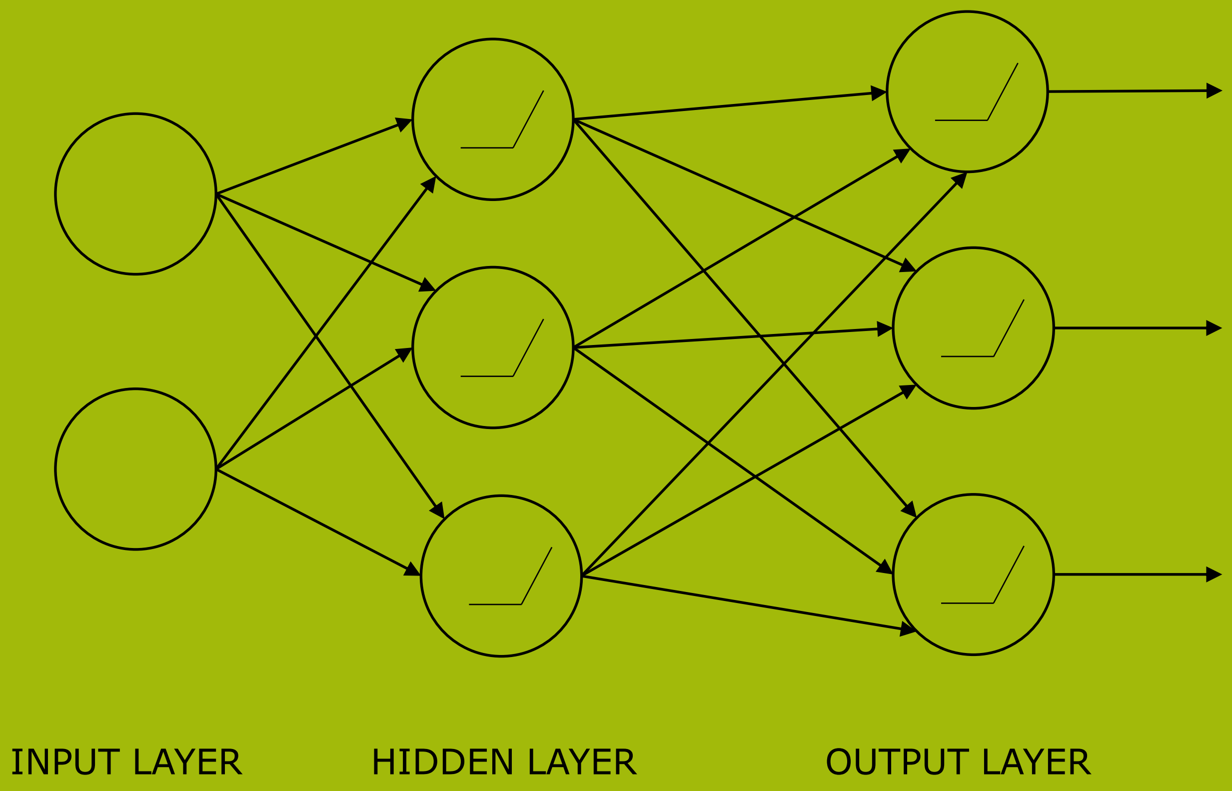
Input layer size

As many as the features vector size (length)

Output layer size

For classification:
Number of the classes

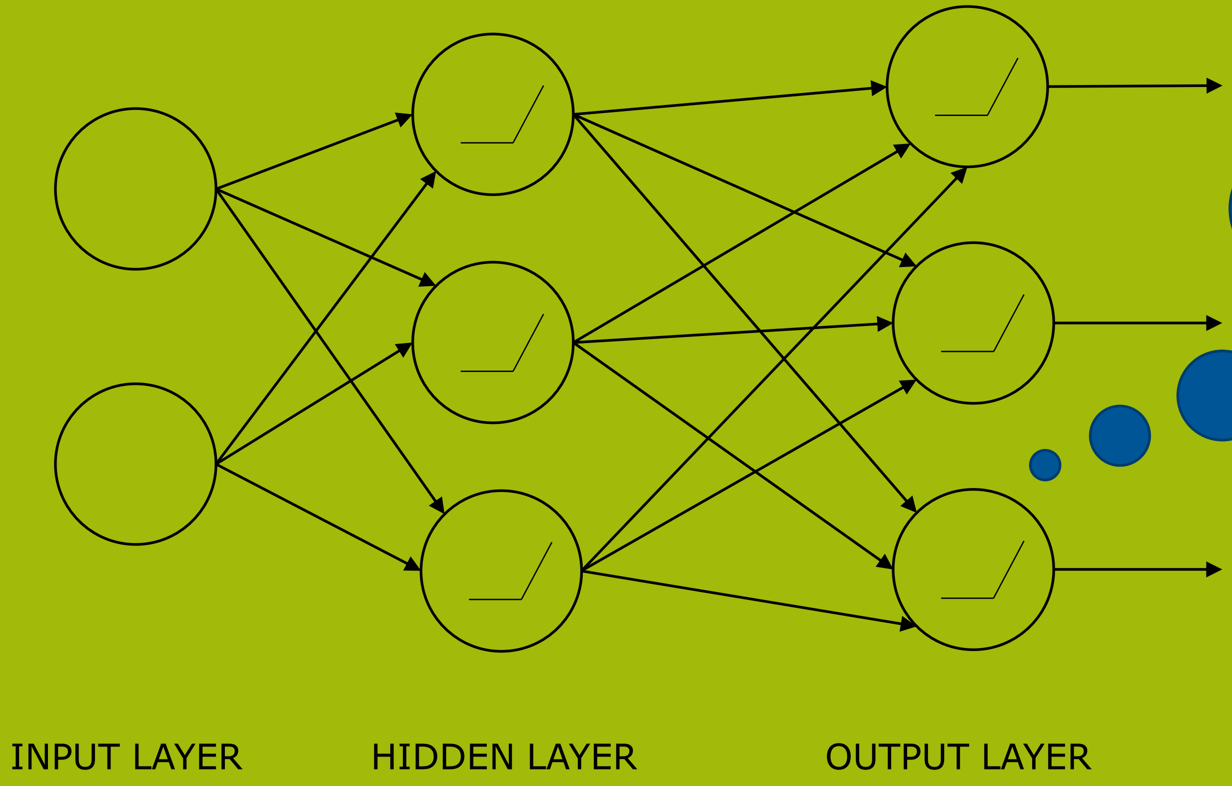
Neural Network Layers



Output layer size

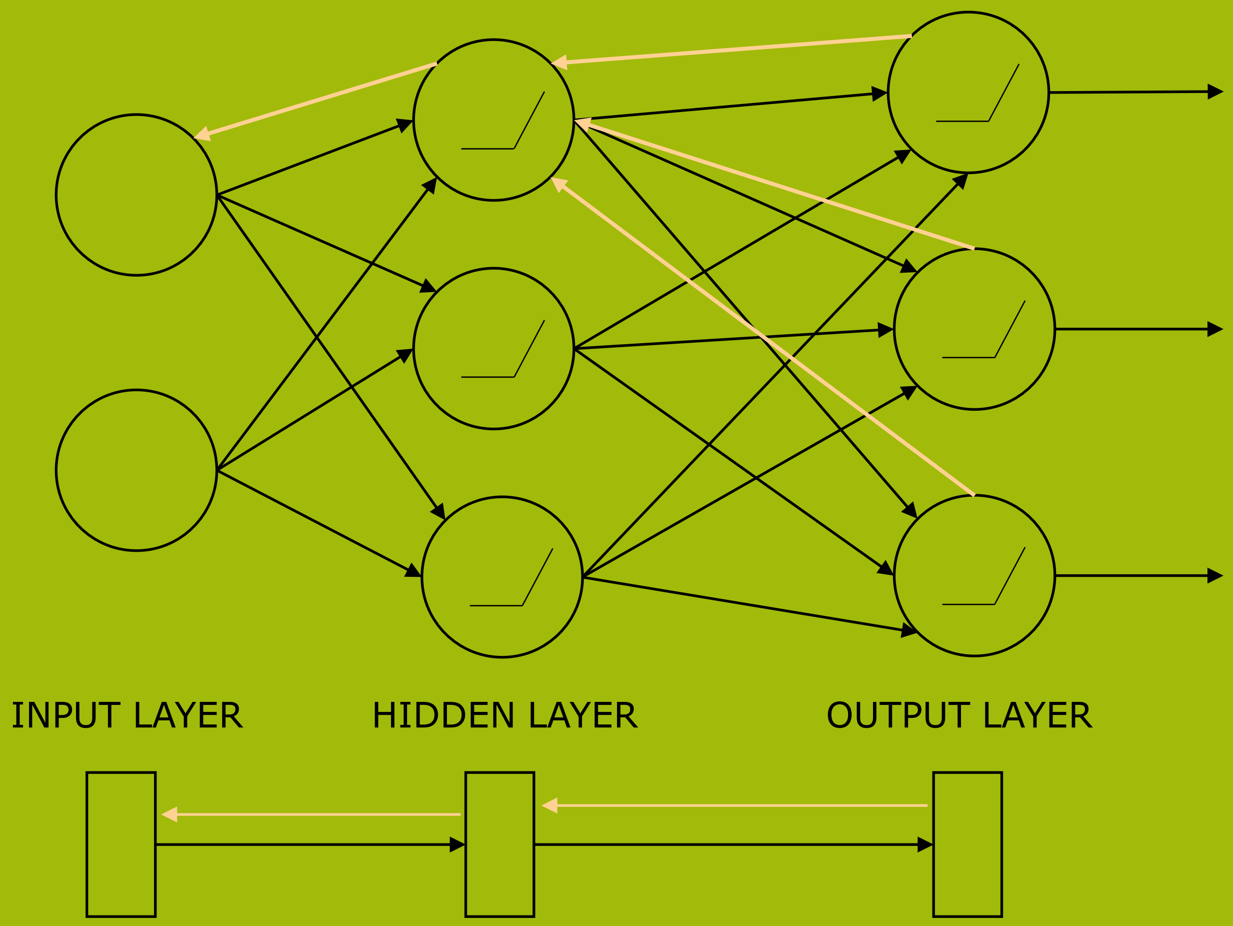
For classification:
Number of the classes

Neural Network Layers



How big should the input/output layer (# of neurons) be if for a NN that translates a single sentence from English to German?

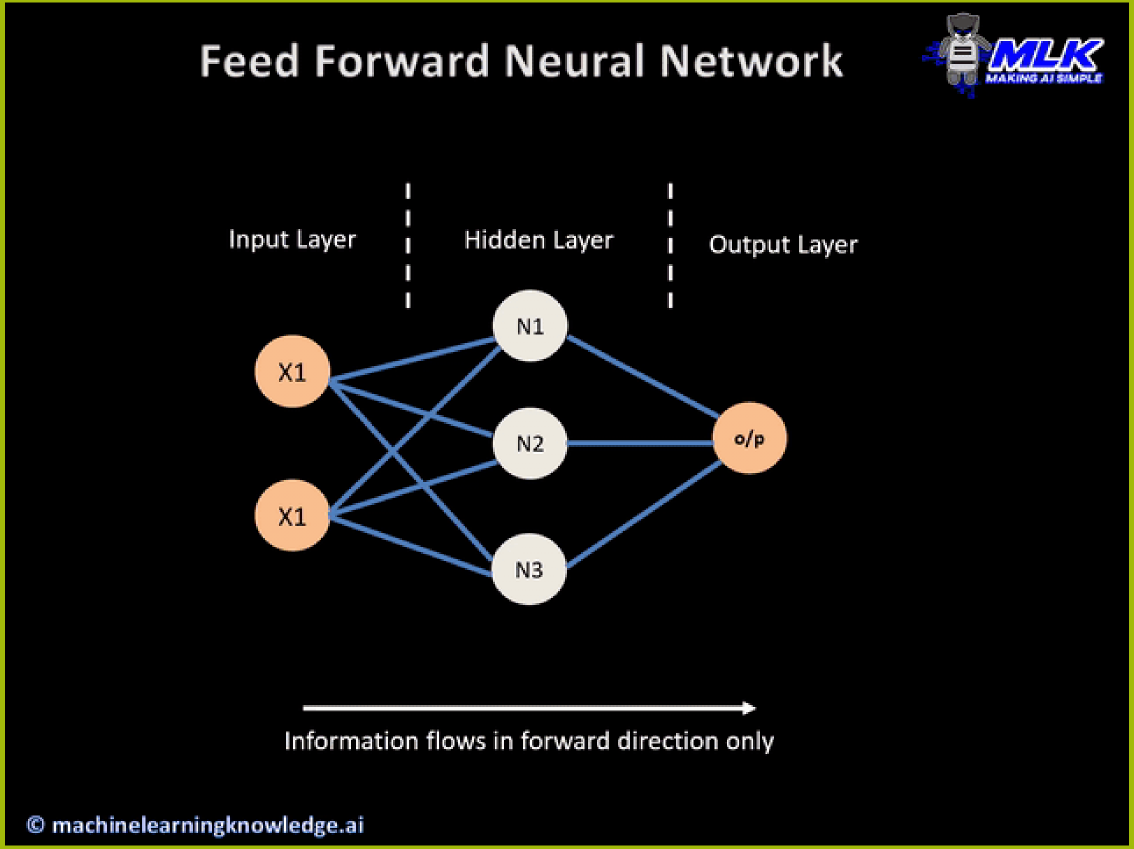
Neural Network Layers



- During the training, the network computes the values for a certain input through the network and compares the result to the given output.
- The gradient of the error is used to correct the neuron weights.

The Inference Part – Feed Forward

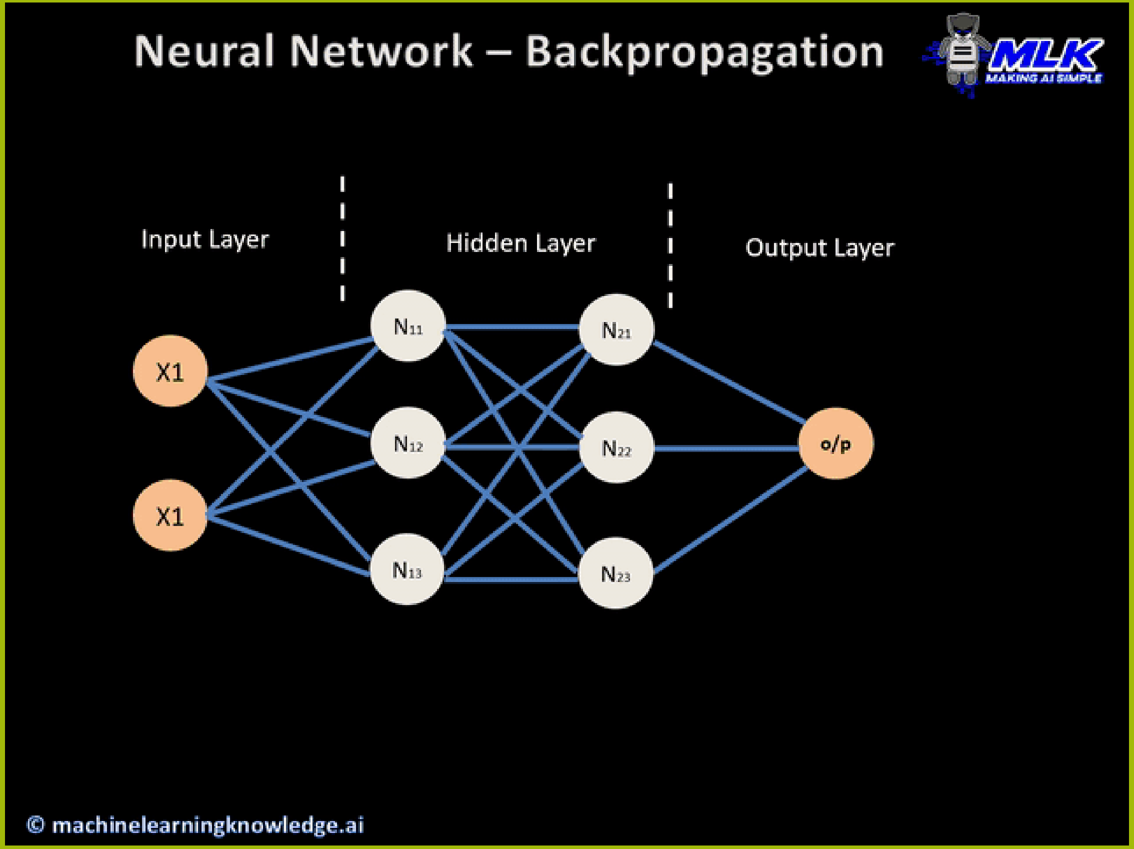
Given an input, predicting the output by calculating the values through the network



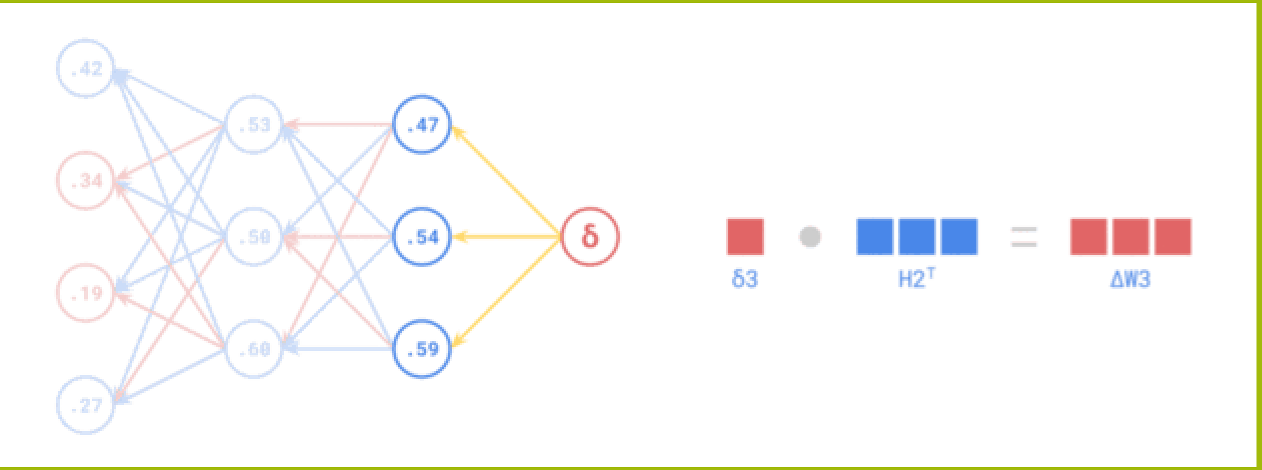
Backpropagation – Training Networks to Learn

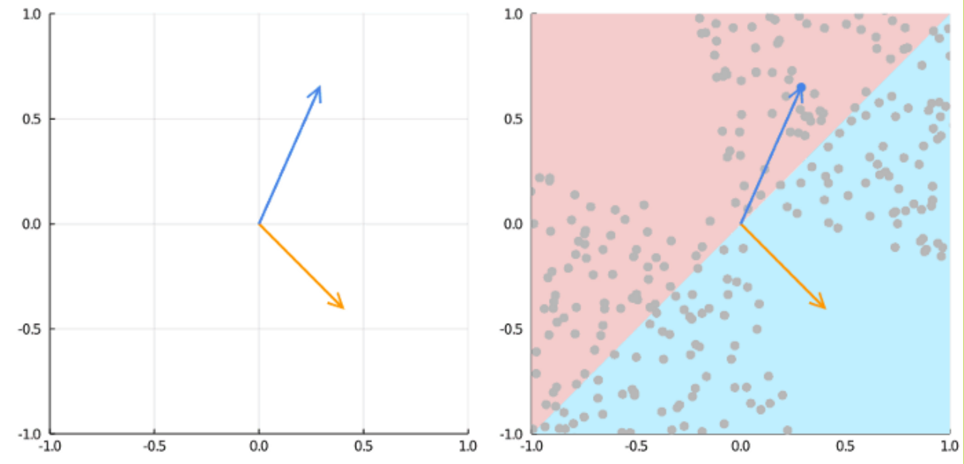
Backpropagation occurs only during the training.

The network calculates the error (**loss function**) and gently updates the network weights, so next time when they “see” this sample, the prediction is closer to the desired result.



For a Deeper Dive





- The Building Blocks of Deep Learning | by Tyron Jung | The Feynman Journal | Medium
- What Makes Backpropagation So Elegant? | by Tyron Jung | The Feynman Journal | Medium

Training Does Not Include:

- > Hyper-parameters setting must be done manually
  - > Choice of activation function (sigmoid, RELU)
  - > Number of layers / number of neurons per layer (network architecture)
  - > Learning rate
  - > Number of epochs (training cycles)
  - > Regularization

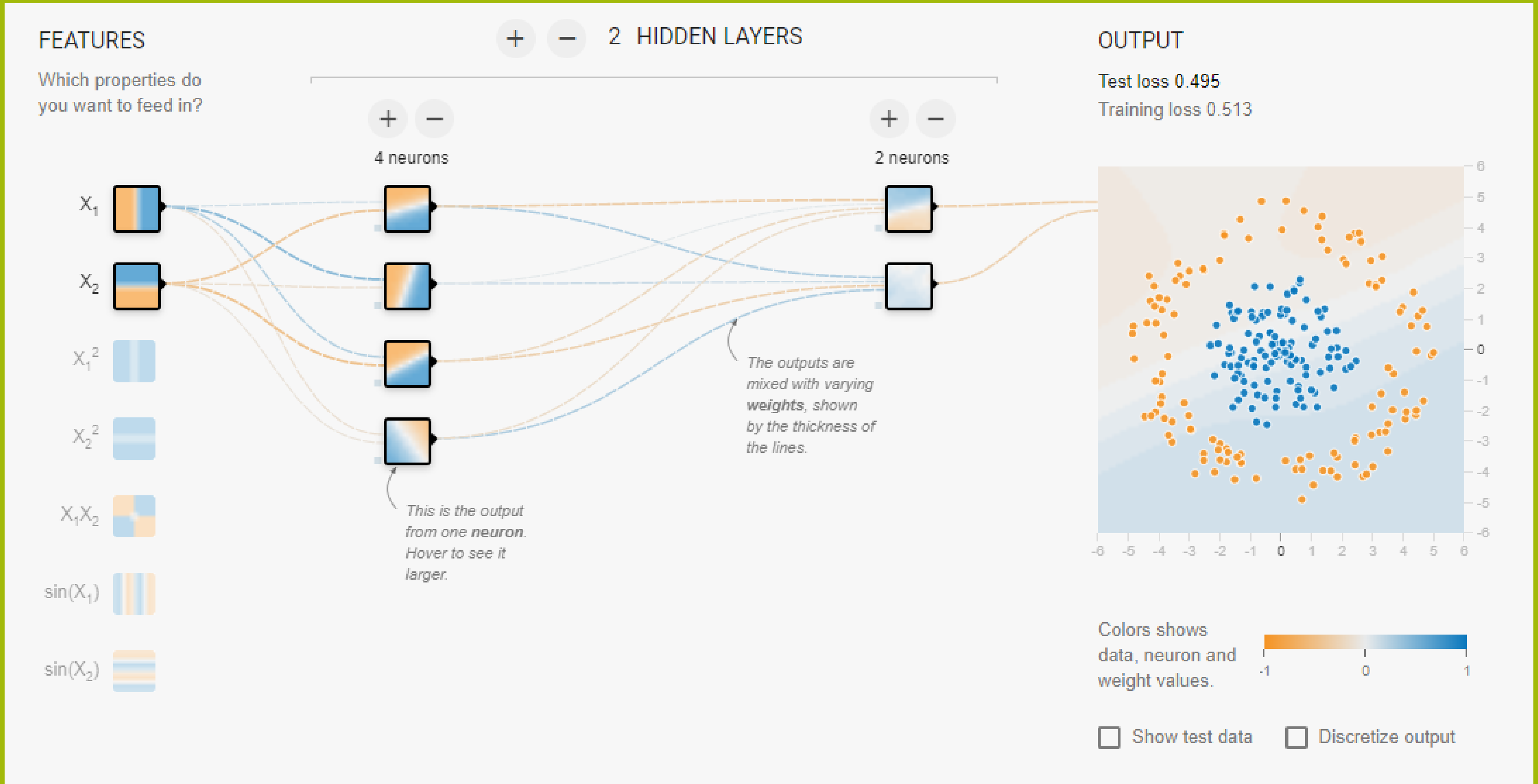
Complicated?

> Many existing frameworks do half of the job for you:
> > Fast.AI
> > Tensorflow + Keras
> > PyTorch + PyTorch-lightning // HuggingFace // spaCy // flair

> Best to start by re-using an existing model and modifying it if needed.

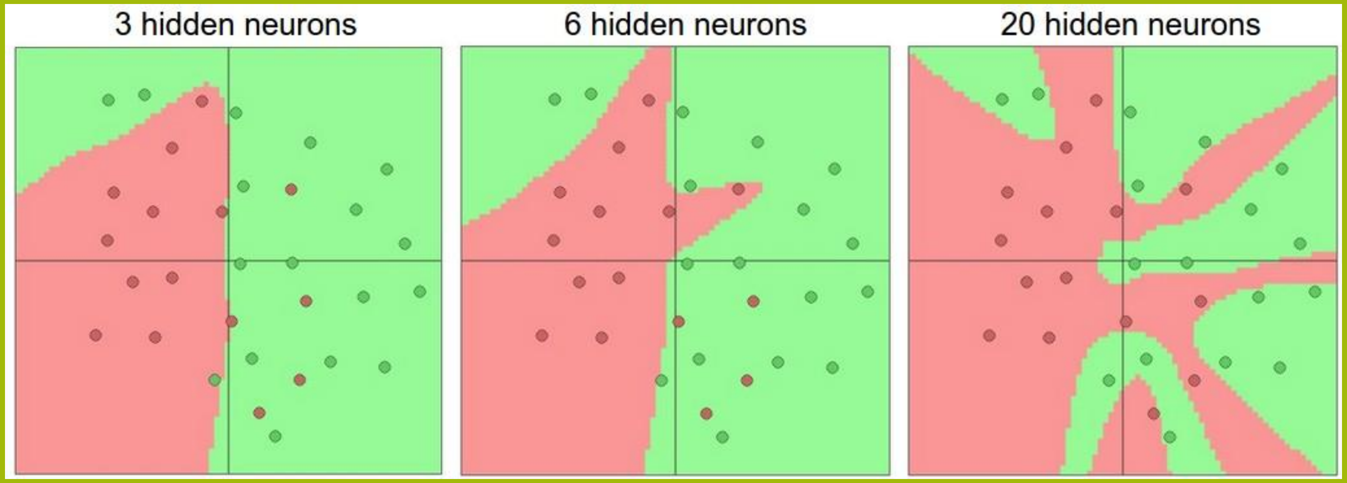
Neural Network - Demo

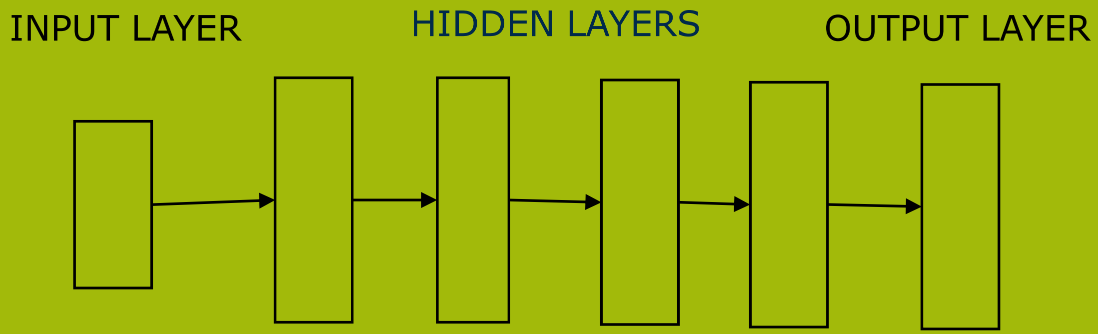
> https://playground.tensorflow.org/



Deep Learning

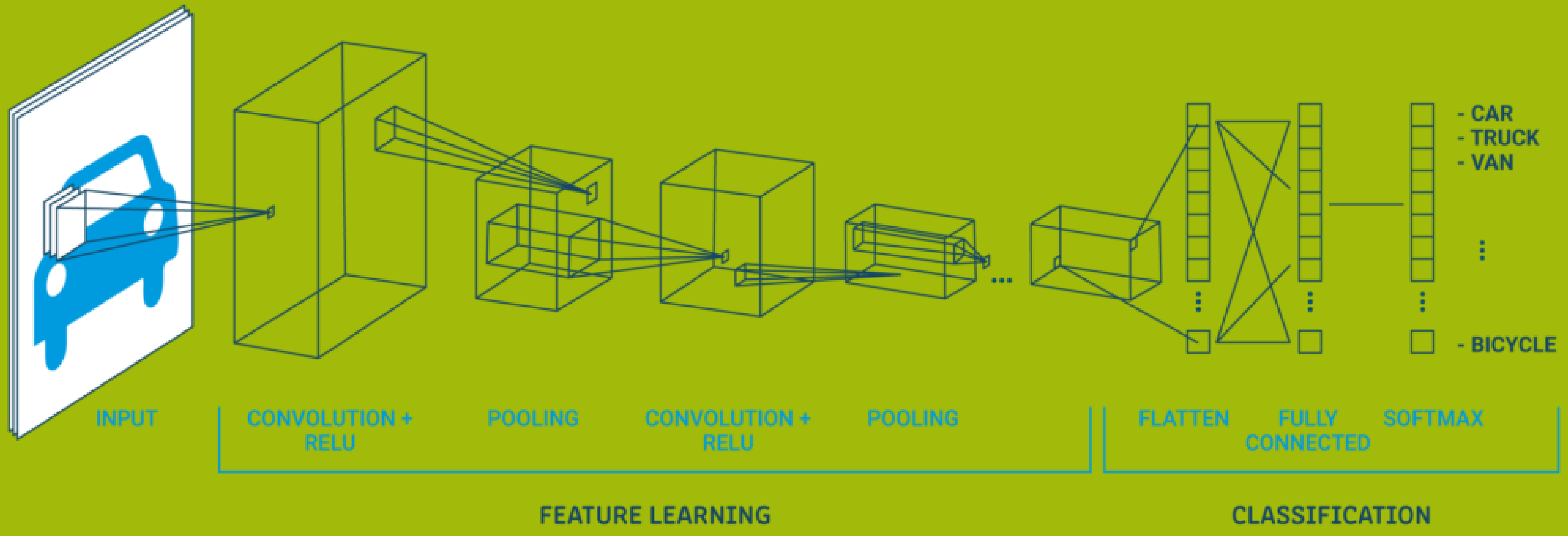
- > Deep Learning means more hidden layers → More computational steps, more degree of freedom
- > It can learn patterns that cannot be learned efficiently with shallow models.





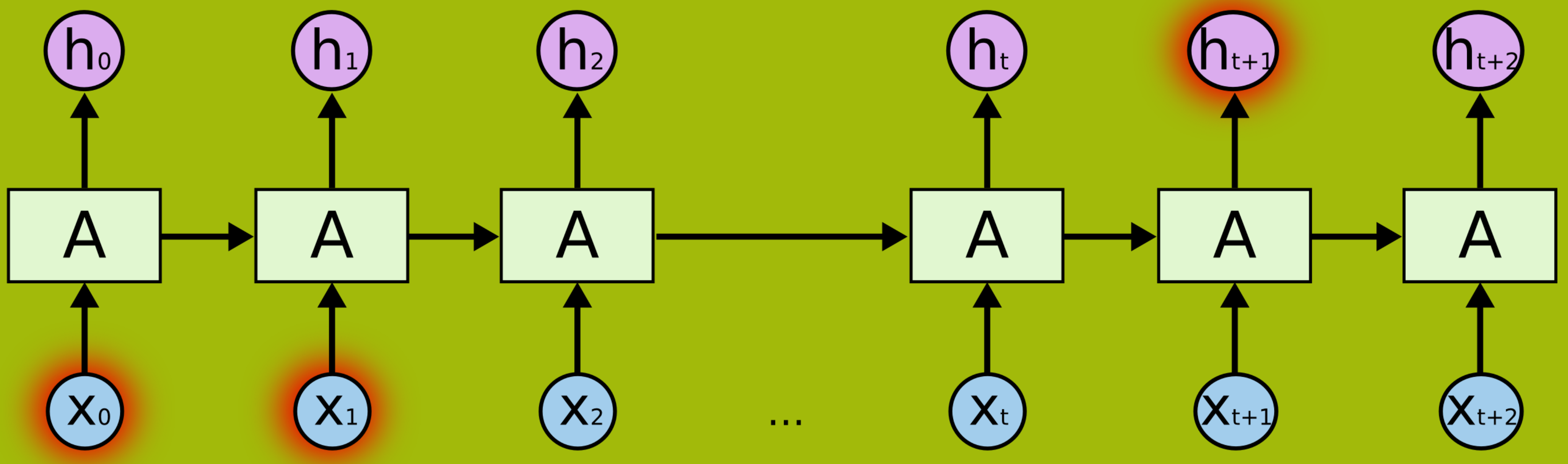
Deep Learning

> Many architectures are researched daily.
> Still an open and (very) active research problem

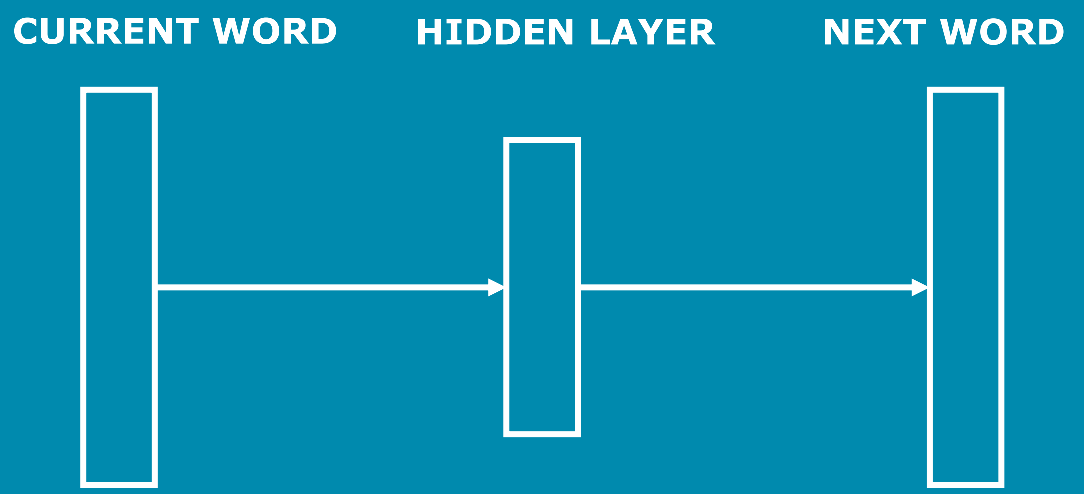


Deep Learning

> For NLP, a common architecture is called RNN or LSTM
> In every step, the network output is re-used as an additional input for the next step.

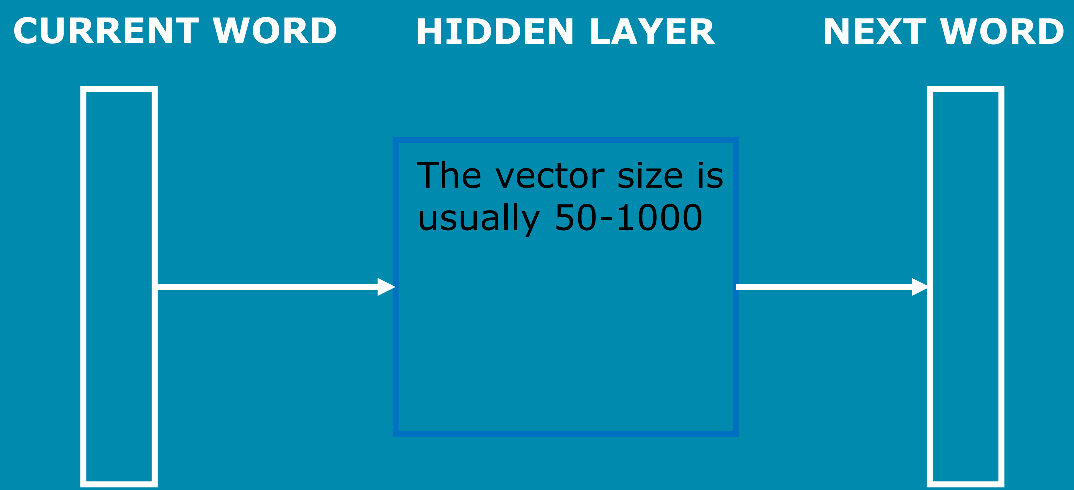


Basic Neural Network applied to NLP



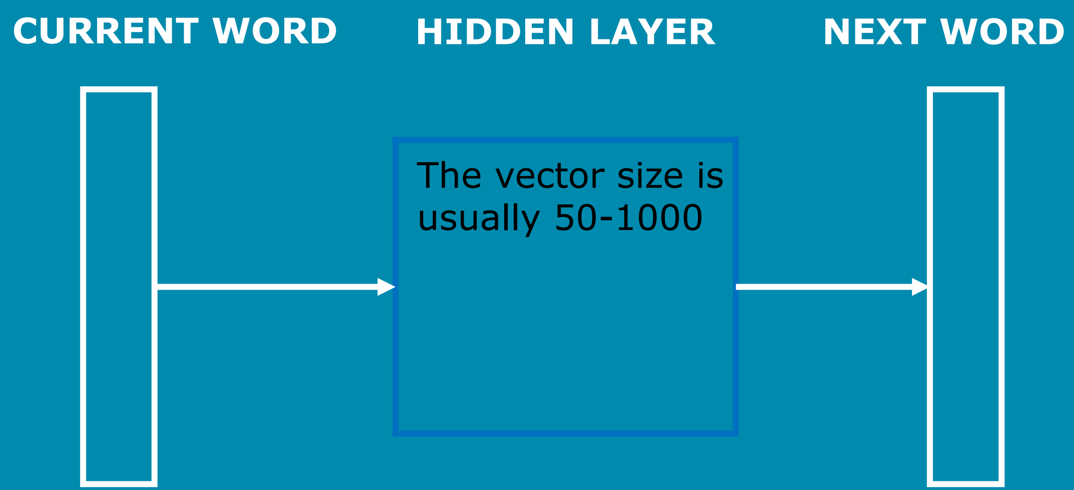
- Bi-gram neural language model: predicts the next word
- The input is encoded as a one-hot-encoder
- The model will learn compressed (dense), continuous representations of words (usually the matrix of weights between the input and hidden layers)

Word Vectors (Mikolov et. al 2013)



- We call the vectors in the matrix between the input and hidden layer word vectors (also known as word embeddings)
- The word vectors have similar properties to word classes (similar words have similar vector representations)

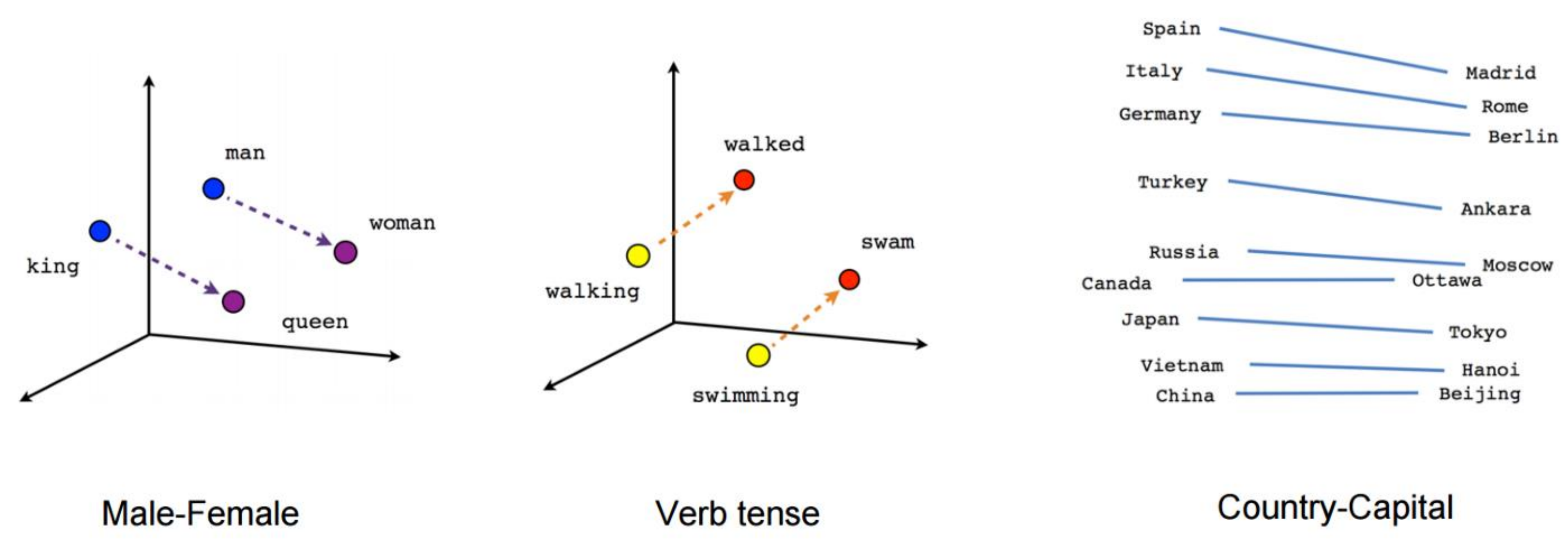
Word Vectors



> The word vectors can be used as feature-inputs for many NLP tasks.

> Word vectors are trained in a ***semi-supervised*** way

Word Vectors – Semantic Properties



Word Vectors

- This is only the beginning.
- Other frameworks for Word Vectors:
  - FastText – character based
  - BytePair Embedding (BPE) – frequent sub-words (letters that often appear together)
- Contextual Embedding:
  - ELMo
  - ULMFiT
  - BERT
  - RoBERTa
  - GPT

Additional Resources

- https://lena-voita.github.io/nlp_course.html
- Animated Explanation of Feed Forward Neural Network Architecture - MLK - Machine Learning Knowledge
- Word Embedding Demo: Tutorial (cmu.edu)
- WebVectors: distributional semantic models online (nlpl.eu)

- Embedding projector - visualization of high-dimensional data (tensorflow.org)

Final Presentations

Choice between analyzing an NLP task or reviewing an academic paper

Verbal presentation - ~10-15min + 5min Q&A

ML/NLP Task

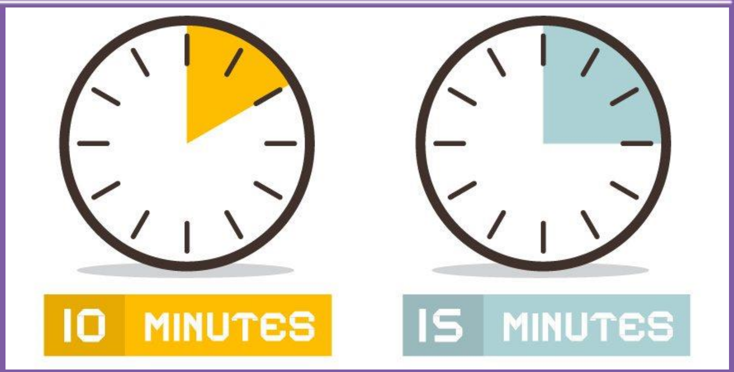
- Describe the problem you want to solve.
- Which dataset(s) will you use?
  - Create your own? Find somewhere? Annotate yourself?
- Which features would you use?
  - What is your expectation from each one?
- Which ML Methods would you think to be the most efficient for this problem?
  - What is their loss function? Which hyperparameters would you aim for tuning?

Academic Paper

- What is the problem the researchers are trying to solve?
- What other solutions already exist? How well do they work?
- How do the researches solve the problem?
- How well did they solve it?
- What is the novelty in their method?
- What is still open regarding this problem?
- What could still be improved with the researches approach?
- What methods would YOU use to tackle it?

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Presentation



- ~10-15 minutes.
- 5 minutes Q&A

- Communication is an important aspect for researchers and data-scientists as one.
- Construct your presentation in a clever way, tell a story, lead the audience across the plot.
  - Use visuals
  - Don’t go crazy with the text

- Some helpful resources:
  - How to Speak – YouTube
  - How to Tell a Story With Data: Steps, Tips and Examples for Leaders | ThoughtSpot