Neural Networks

Alex Olson

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Centre for Analytics and Artificial Intelligence Engineering

Programming Languages for AI

- Python
  - Dominates AI development due to extensive libraries like TensorFlow, PyTorch, and scikit-learn.
  - Easy syntax, strong community, rich ecosystem for AI research and production.
  - Slower execution compared to lower-level languages, can struggle with very large-scale, performance-critical systems.

Programming Languages for AI

- R
  - Specialized for statistics and data visualization, with packages like caret and ggplot2.
  - Most useful for data preprocessing, statistical modeling, and some ML tasks.
  - Less suited for general-purpose AI development or production-grade systems.

Programming Languages for AI

- C++
  - High performance for real-time applications like gaming or embedded AI. Libraries like dlib and OpenCV excel in computer vision.
  - Core of many Python-based AI tools (e.g., TensorFlow's backend).
  - Steeper learning curve, verbose syntax, slower development time.
- Java
  - Scalability and enterprise use, with frameworks like Weka and Deeplearning4j.
  - Can interact with Python tools via APIs or frameworks like Apache Spark.
  - Verbose and less favored for rapid prototyping.
- Julia
  - High-performance numerical computing, increasingly adopted for AI and optimization.
  - Growing interoperability with Python (e.g., PyCall).
  - Smaller ecosystem and less community support than Python.

Machine Learning Libraries PyTorch

- PyTorch
  - Intuitive and flexible, with dynamic computation graphs allowing for easier debugging and experimentation.
  - Strong adoption in research, supported by an active community.
  - Less mature deployment tools compared to TensorFlow (though this gap is narrowing).
  - Can be slower in some production scenarios without optimization.

Machine Learning Libraries

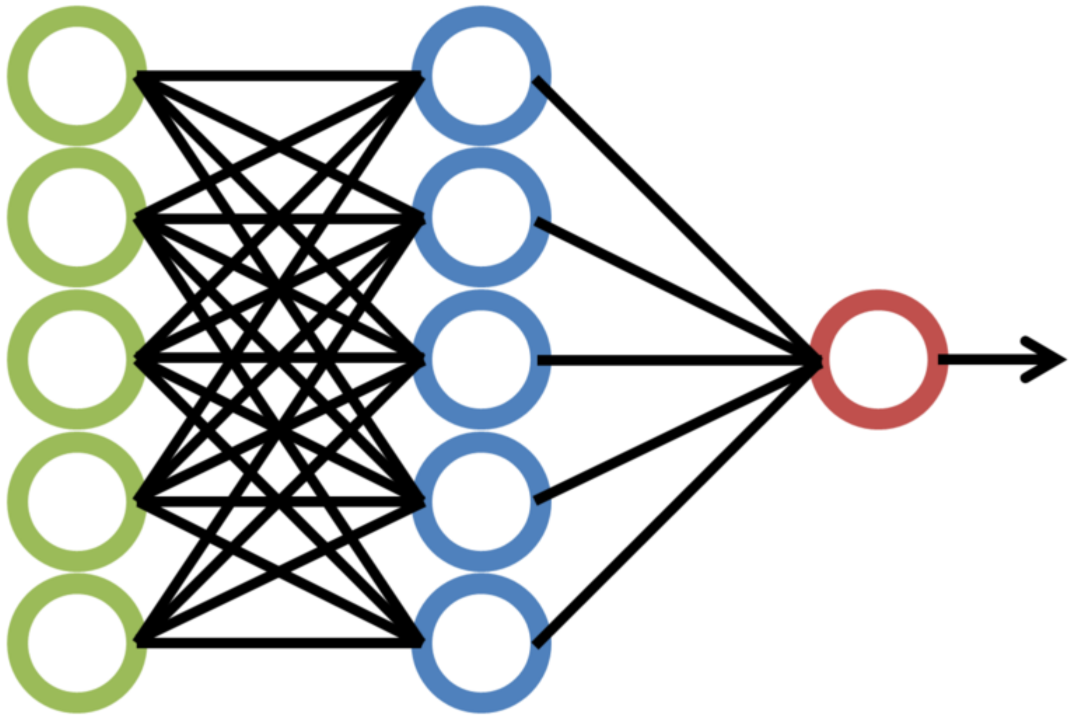
- TensorFlow
  - Comprehensive ecosystem with tools for training (TensorFlow), deployment (TensorFlow Serving, TensorFlow Lite), and explainability (What-If Tool).
  - TensorFlow.js and TensorFlow Lite make it suitable for web and mobile development.
  - Strong community and corporate support (Google).
  - Integration with Keras offers a high-level API for beginners.
  - Steeper learning curve compared to PyTorch.
  - Debugging can be less straightforward due to static computation graphs (though this has improved with TensorFlow 2.x).

Machine Learning Libraries

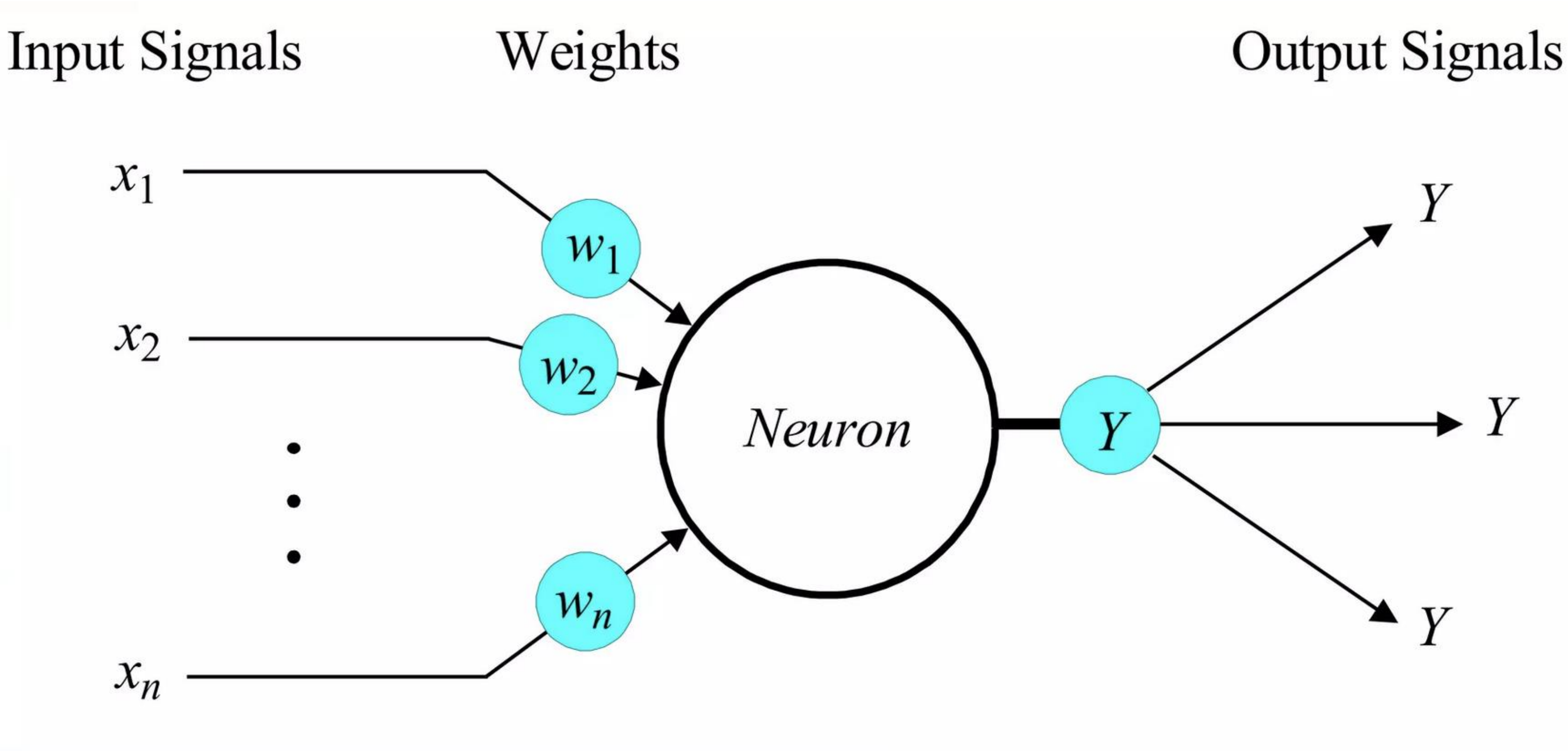
- Scikit-learn
  - Easy-to-use interface for classical machine learning tasks like regression, classification, and clustering.
  - Excellent for preprocessing and feature engineering (e.g., PCA, scalers).
  - Strong documentation and wide adoption in education and small-scale projects.
- XGBoost
  - Extremely efficient and scalable for tabular data tasks.
  - Known for achieving high accuracy with minimal tuning.
  - Distributed training support for large datasets.
- HuggingFace Transformers
  - Simplifies the use of pre-trained transformers for NLP, vision, and multimodal tasks.
  - Strong community and regularly updated with state-of-the-art models.
  - Easy fine-tuning and deployment of large language models (LLMs).

What is a neural network?

- Complex structure of interconnected computing nodes (neurons)
- Can identify patterns and trends in complex data
- NNs operate on the principle of "learning" from data, using a process that mimics how biological brains learn

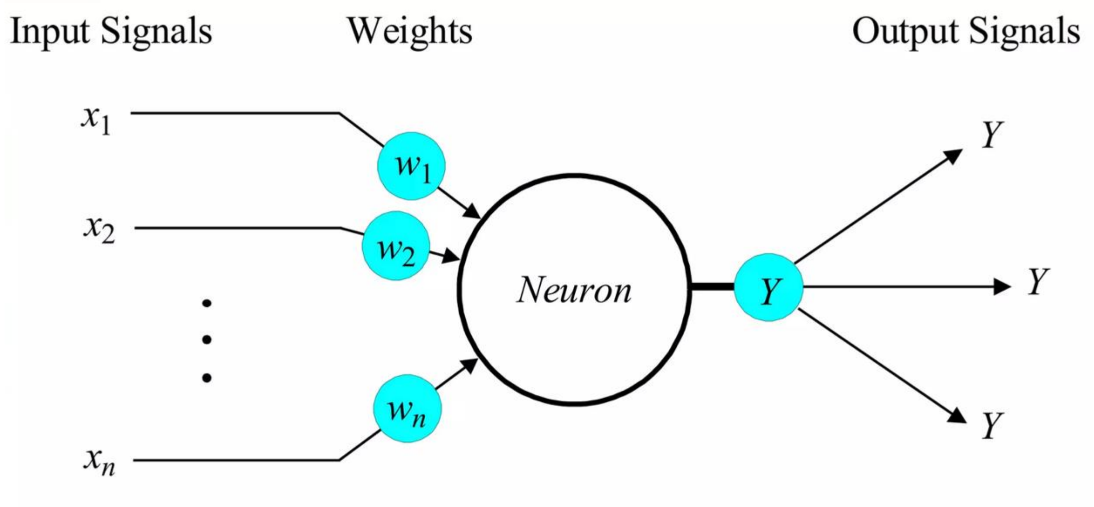


View inside an artificial neuron



View inside an artificial neuron

- Behaves like a linear regression model:
- $w_1x_1 + w_2x_2 + \ \dots + w_nx_n$
- Weights correspond to how much the neuron "cares" about each input

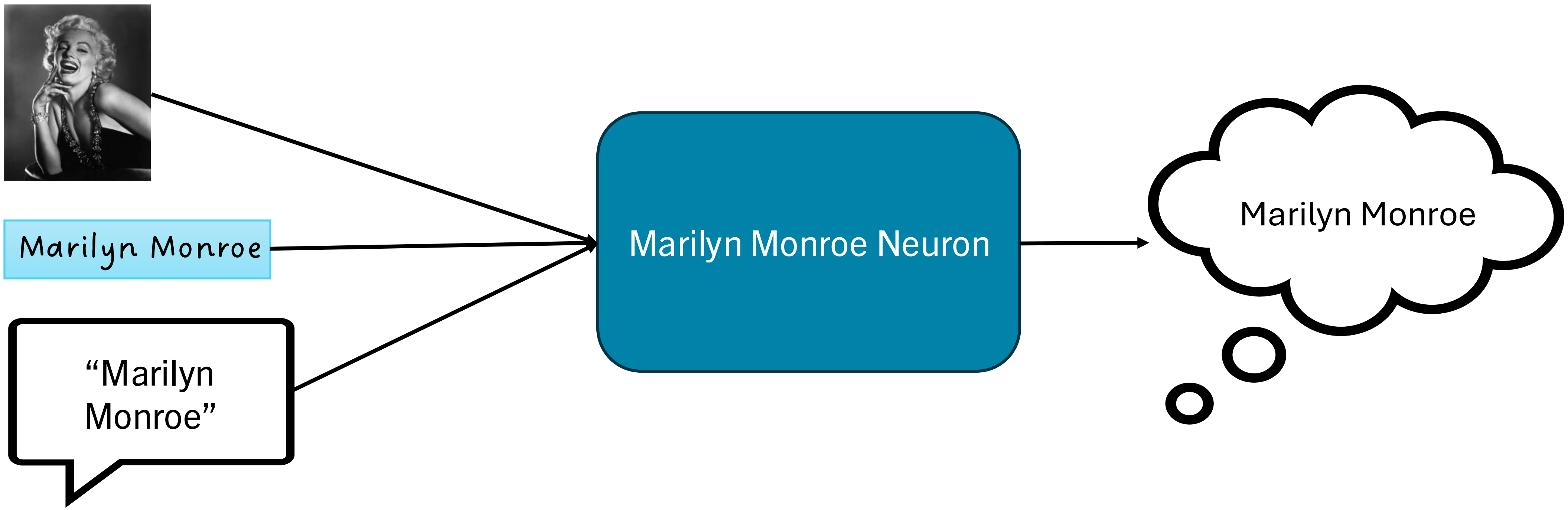


Back to the brain: the Marilyn Monroe Neuron

- Study conducted on patients with epilepsy
- Researchers use specialized equipment to measure the “excitement” of individual neurons in a patient’s brain
- Measuring a neuron, the researchers showed patients a series of images
- In each patient, they found around five neurons that fired when the patient looked at a specific person

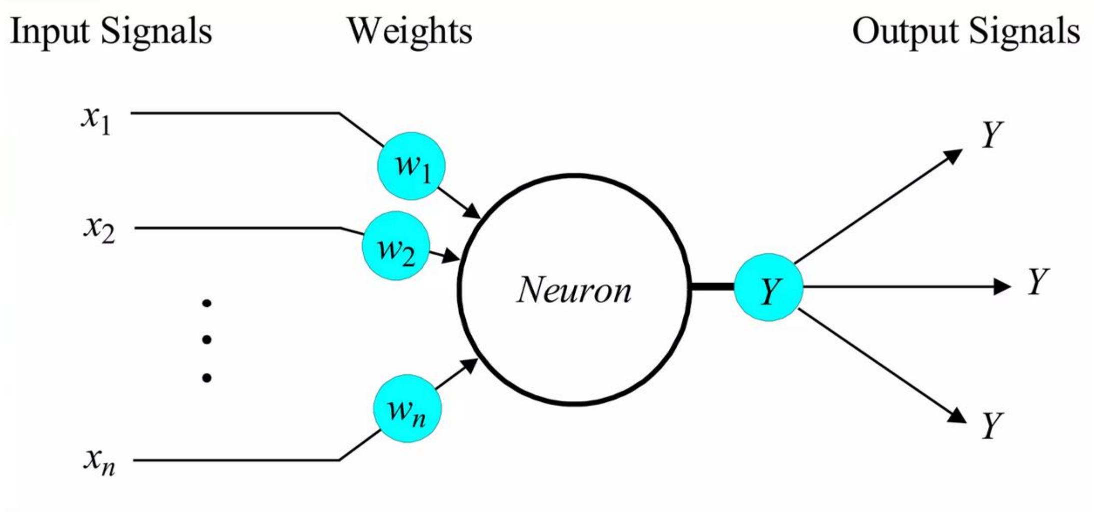
Back to the brain: the Marilyn Monroe Neuron

- Once a “celebrity” neuron was identified, the researchers wanted to know if it would still fire for representations other than images



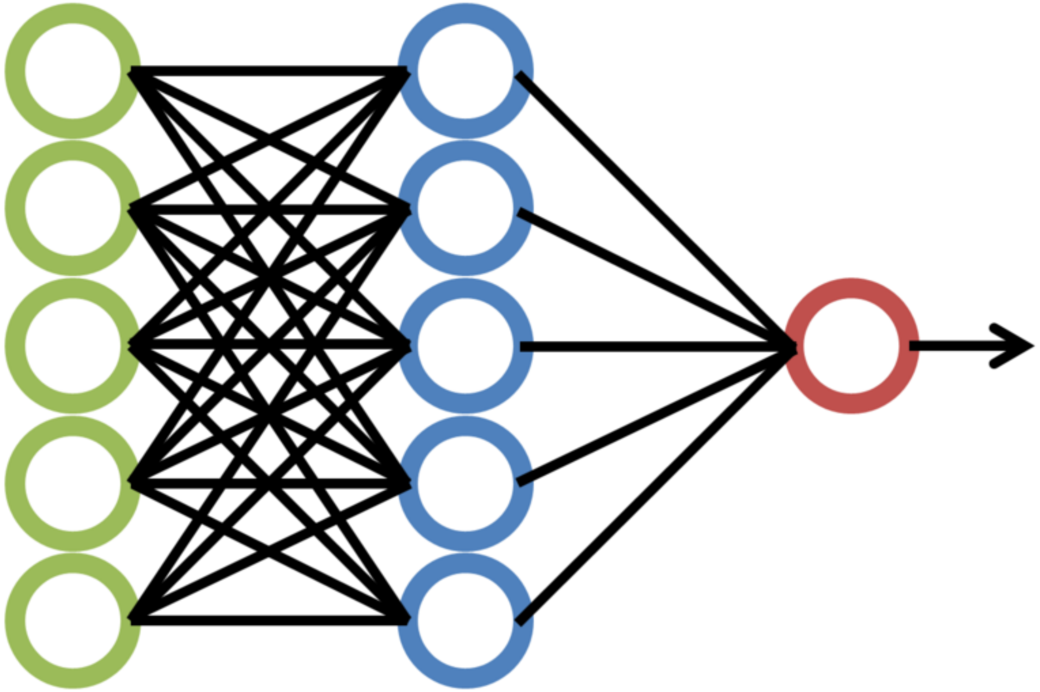
Marilyn Monroe ANN

- Weights would be high from neurons that react to different representations of Marilyn Monroe
- Weights would be low for neurons that react to other people, or concepts



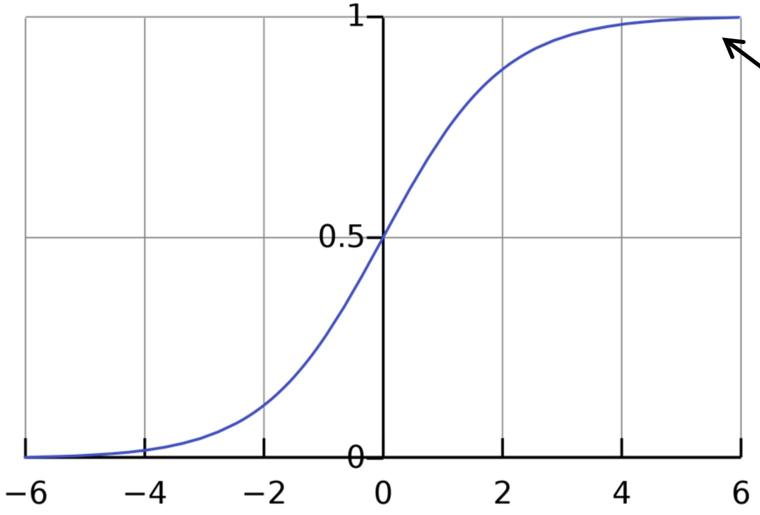
ANNs

- Each neuron considers the responses of the neurons in the previous layer
- It learns to pay attention to the neurons that are excited about what it's excited about
- Ignores the neurons that are excited about other things



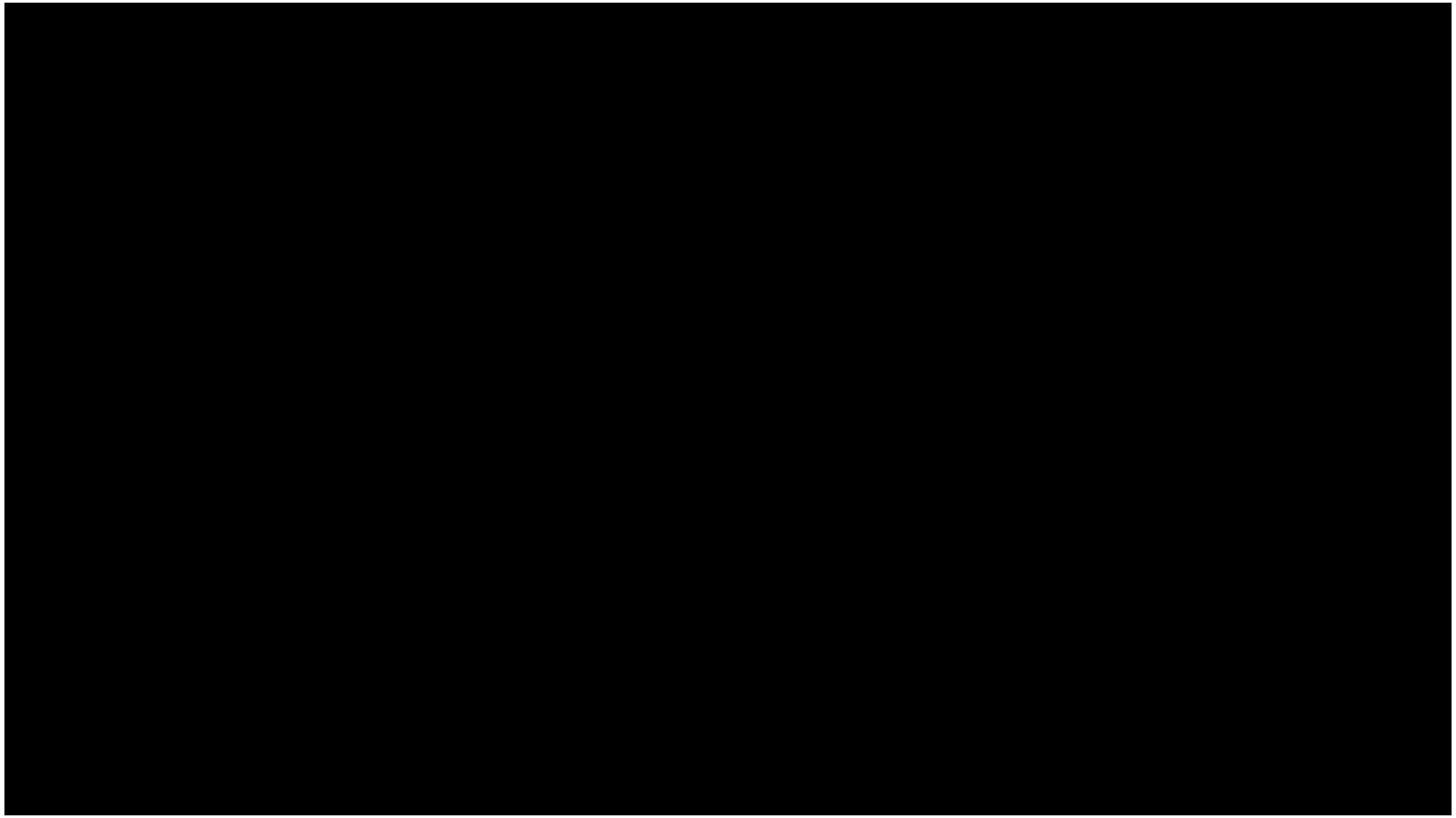
Activation Functions

- Basic approach: when I see enough activity, I get excited
- Below threshold: 0
- Above threshold: 1
- More useful: gradually increase excitement as we see more activity
- In practice: many different activation functions!



Desired value gets max output

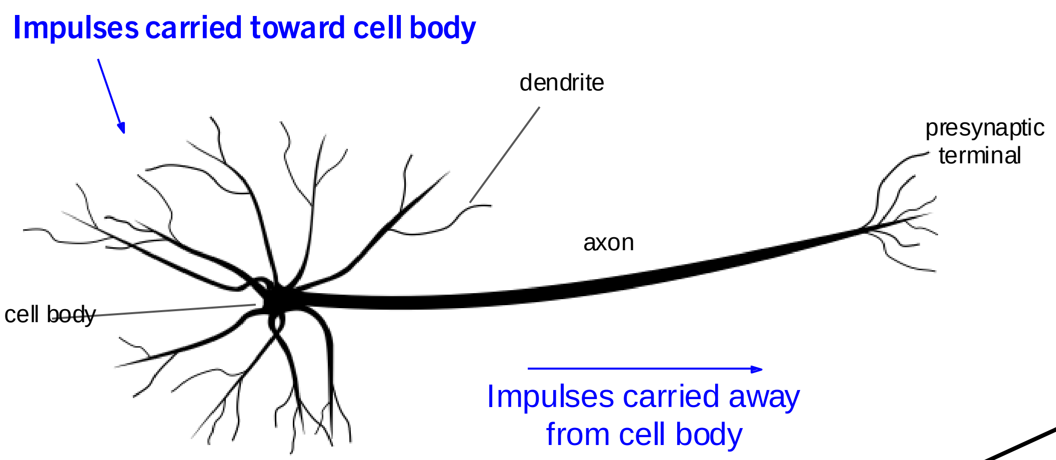
How do Neural Networks actually work?

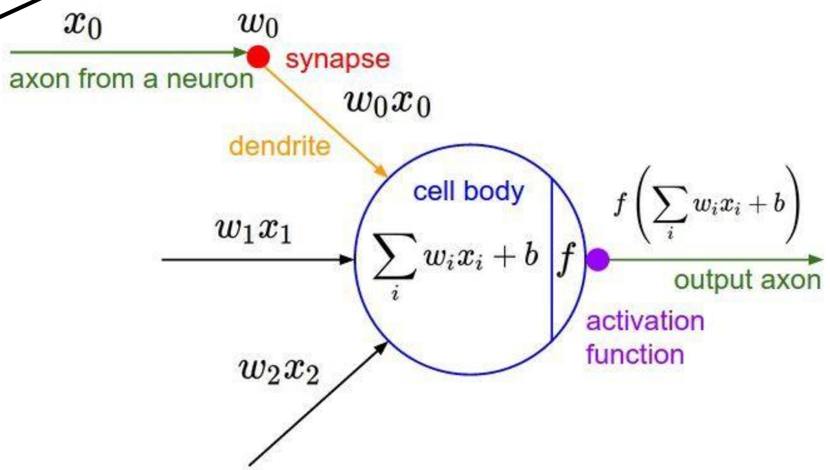


The Neuron Metaphor

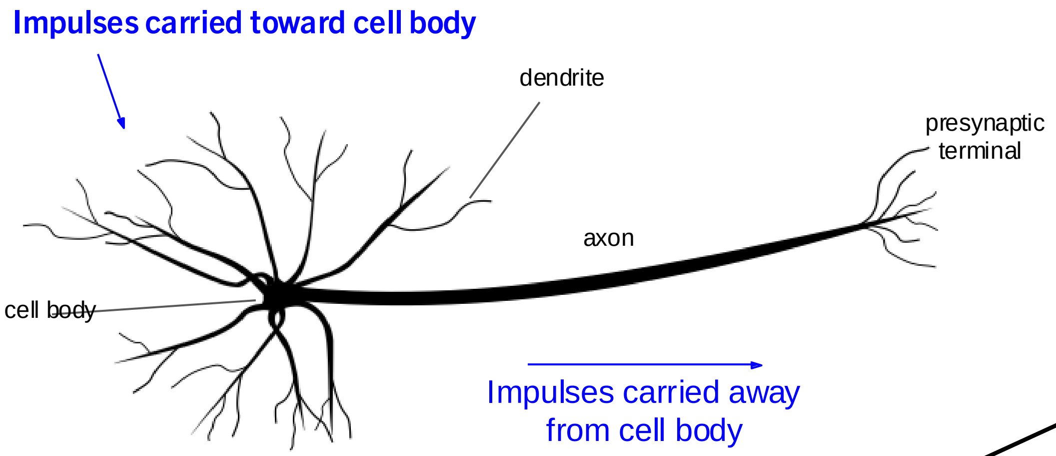
- Neural networks were inspired by our understanding of the brain and how neurons interact.
- An artificial neuron in a neural network takes in multiple inputs, applies a function to them, and generates an output – mirroring the basic functionality of a biological neuron.
- This analogy has been extremely useful for explaining and visualizing how these artificial structures work.

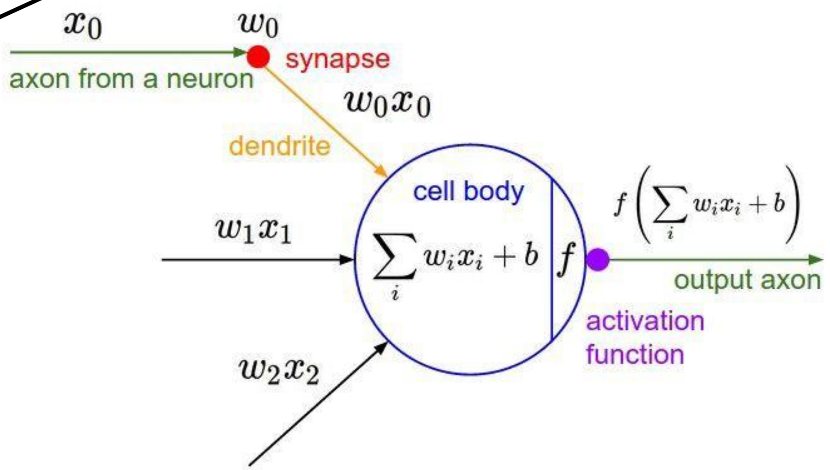
The Neuron Metaphor

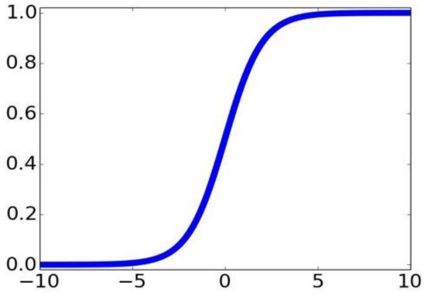




The Neuron Metaphor







sigmoid activation function

$$\frac{1}{1+e^{-x}}$$

Training the Network

- Find parameters that **minimize the total error**
- **Loss** for a given sample is the total error in predictions made
- Going through the network, the predictions are dependent on the settings of the parameters
- We have a mathematical function representing the network
- A way of measuring how ”good” it is
- How do we find the *parameters* that minimize the *loss*?

Gradient Descent

- Let's imagine we have a single parameter, p
- We can compute the relationship between p and our prediction: the $derivative\ of\ the\ loss\ with\ respect\ to\ p$
- The derivative tells us whether increasing p will increase the error, or decrease it
- To minimize loss, we make a set of predictions, compute the derivative using the total error, and adjust p *away* from the error

Gradient Descent

- We can use gradient descent to play "guess what number I'm thinking of"
- If your guess is too high, you decrease it
- If your guess is too low, you increase it
- The error function is a parabola
- By finding the lowest point on the parabola, you find the best guess

Visualizing Gradient Descent

https://uclaacm.github.io
/gradient-descent-visualiser

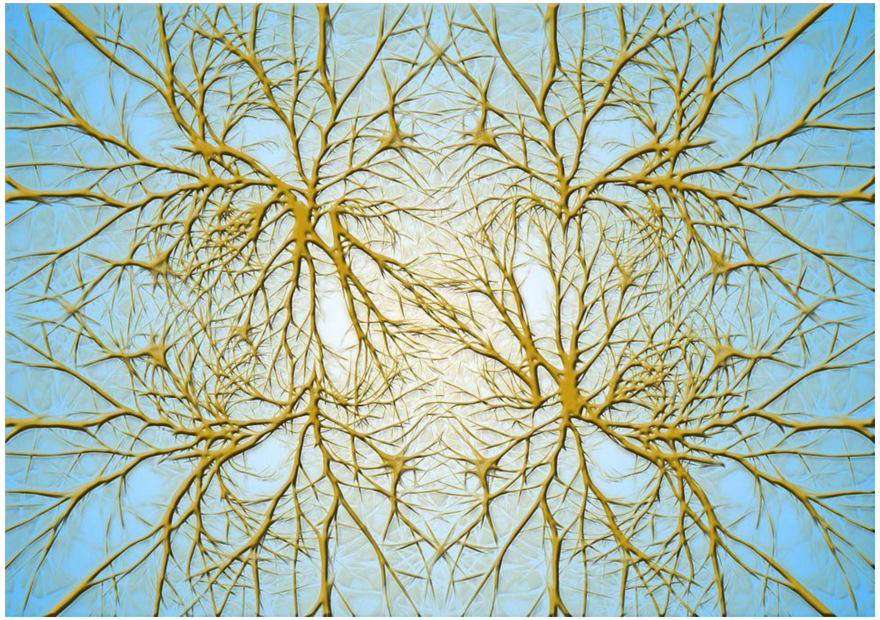
Stochastic Gradient Descent

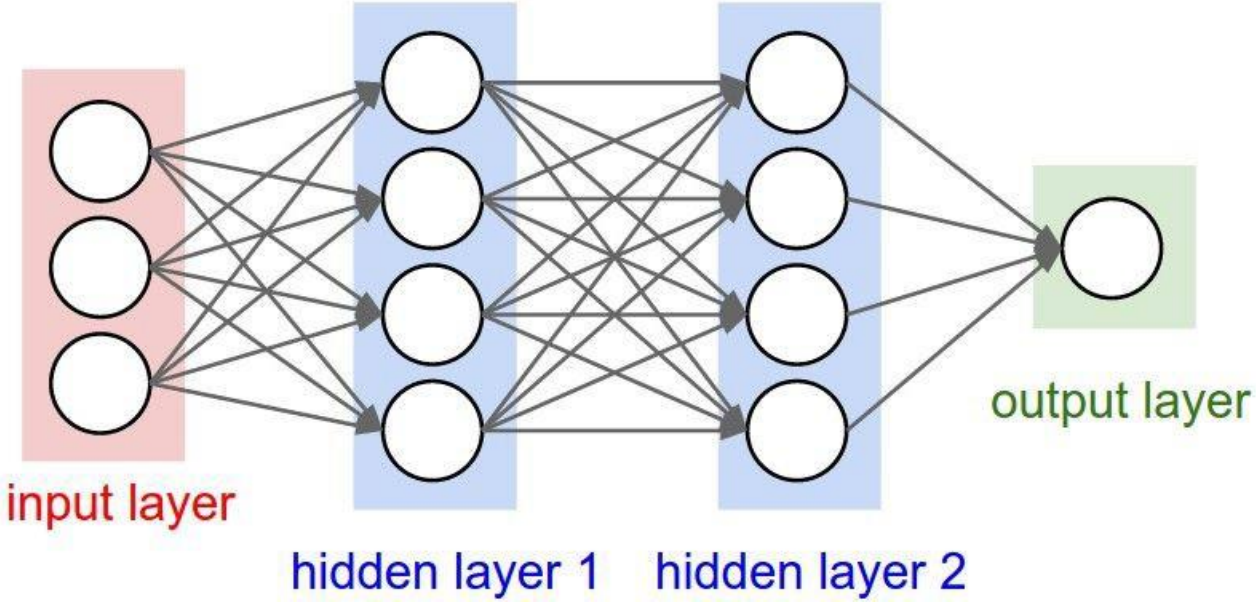
- Traditional Gradient Descent uses the entire dataset to compute the gradient, which can be computationally expensive
- Stochastic Gradient Descent (SGD) updates the parameters using only a single data point (or a small batch)
- In SGD, for each iteration, a data point (or batch) is randomly selected to compute the gradient
- Since only a subset of data is used, the gradient estimation can be noisy, leading to a less smooth path towards the minimum
- However, SGD is much faster than traditional gradient descent

The Metaphor Breaks Down

<u>Biological Neurons:</u>
Complex connectivity patterns

<u>Neurons in a neural network:</u>
Organized into regular layers for computational efficiency





The Metaphor Breaks Down

- Biological neurons are vastly more complex: they use a mixture of electrical and chemical signals, have complex temporal dynamics, and can restructure their own connections.
- The brain is not just a feed-forward network: it has many complex feedback loops, which are not typically found in artificial neural networks.
- The brain isn't easily divided into distinct layers, as we do in artificial neural networks.

The Metaphor Breaks Down

- Over-reliance on the analogy can lead to misunderstandings about how neural networks function and their capabilities.
- This can lead to unrealistic expectations about what neural networks can do, or to overgeneralizations about their functioning.
- For instance, claiming a neural network "thinks" or "understands" like a human brain is misleading.
- To further progress, it's important to view artificial neural networks as mathematical/statistical tools, and not overstate the comparison to the human brain.

Neural Network Playground

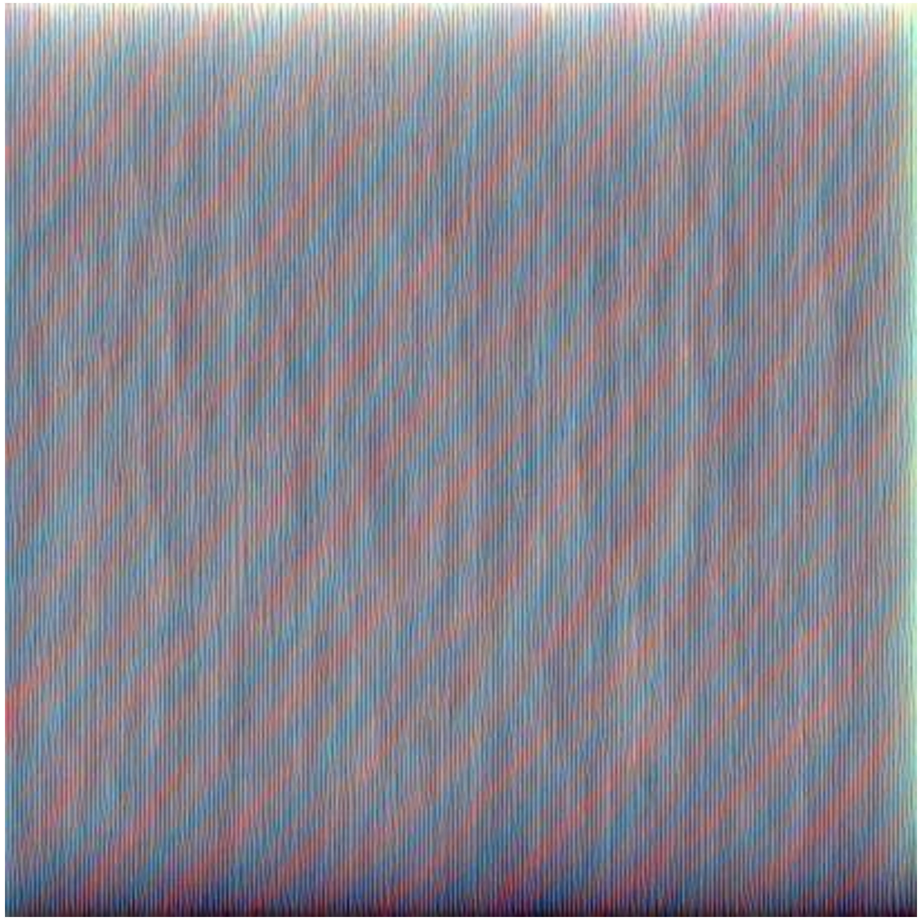
https://playground.tensorflow.org

Going past the fully connected network

- In many image tasks, we want to be able to recognize something regardless of where it is in the image
- For fully-connected networks, the order of the inputs is fixed
- No "shift invariance"

Going past the fully connected network

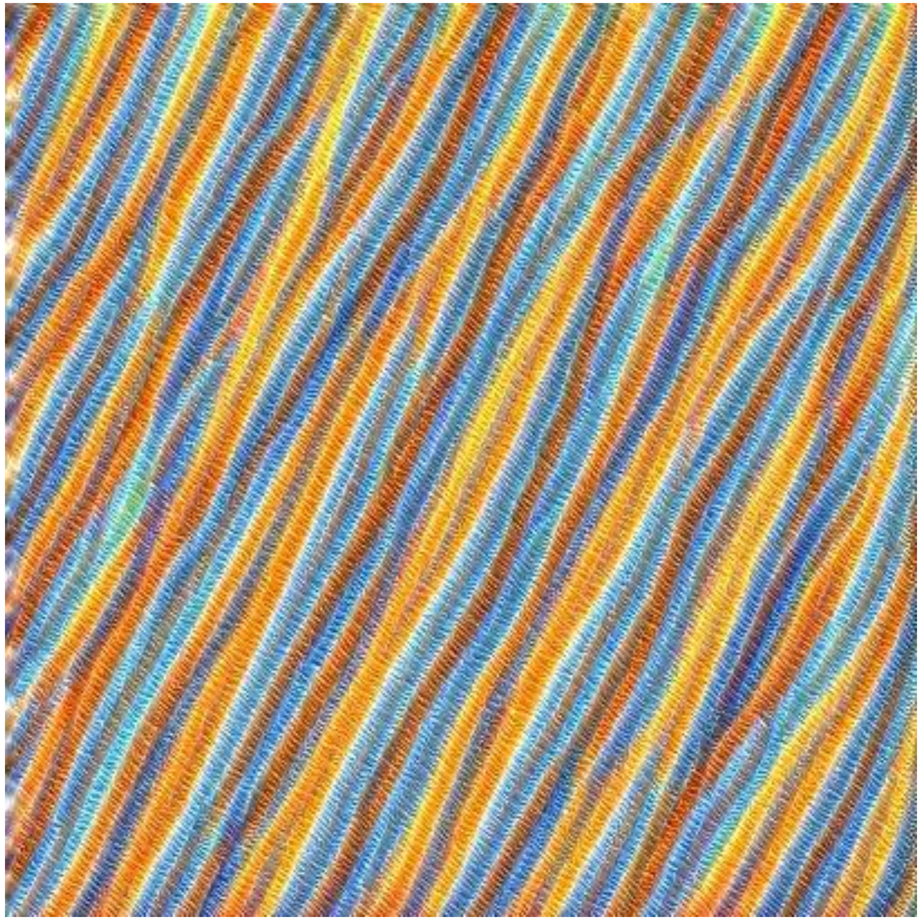
- In the 1950s and 60s, researchers showed that the brain contains neurons which respond to specific patterns, regardless of where they appear
- Combinations of very basic patterns can then be recognized as a more complicated one!



VGG-16, neuron in layer 7

Going past the fully connected network

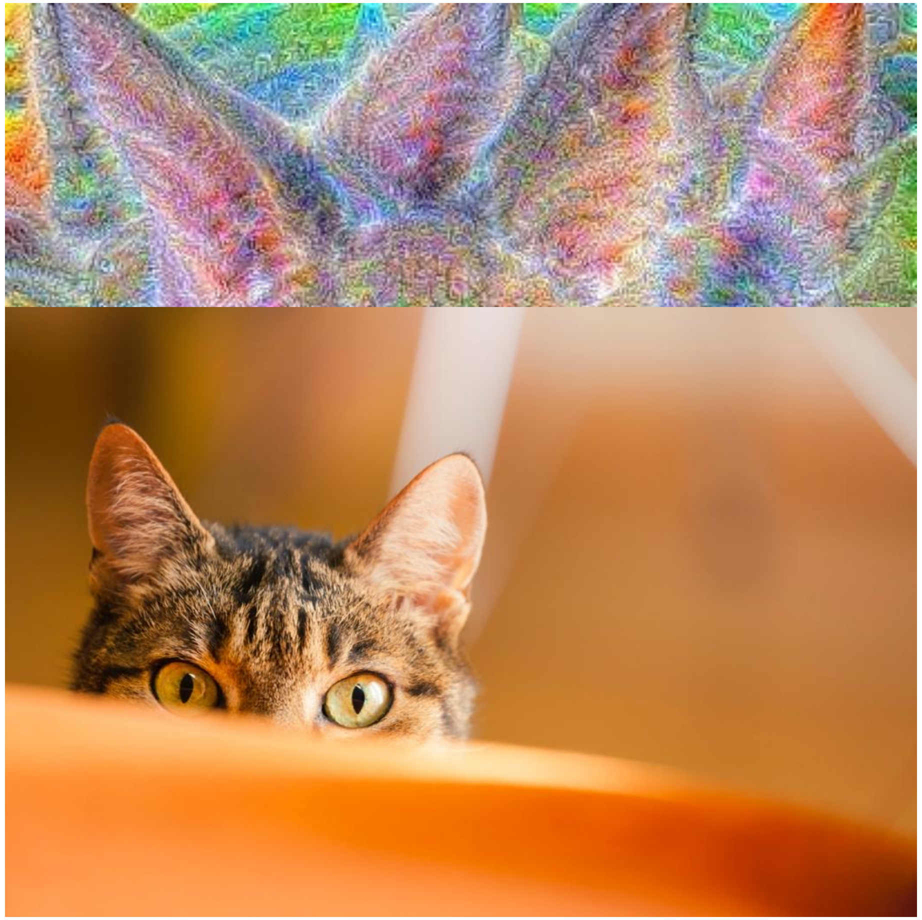
- In the 1950s and 60s, researchers showed that the brain contains neurons which respond to specific patterns, regardless of where they appear
- Combinations of very basic patterns can then be recognized as a more complicated one!



VGG-16, neuron in layer 14

Going past the fully connected network

- In the 1950s and 60s, researchers showed that the brain contains neurons which respond to specific patterns, regardless of where they appear
- Combinations of very basic patterns can then be recognized as a more complicated one!



VGG-16, neuron in layer 40

Interactive CNNs

https://adamharley.com
/nn_vis/cnn/2d.html