Neural Representations II

AI 109

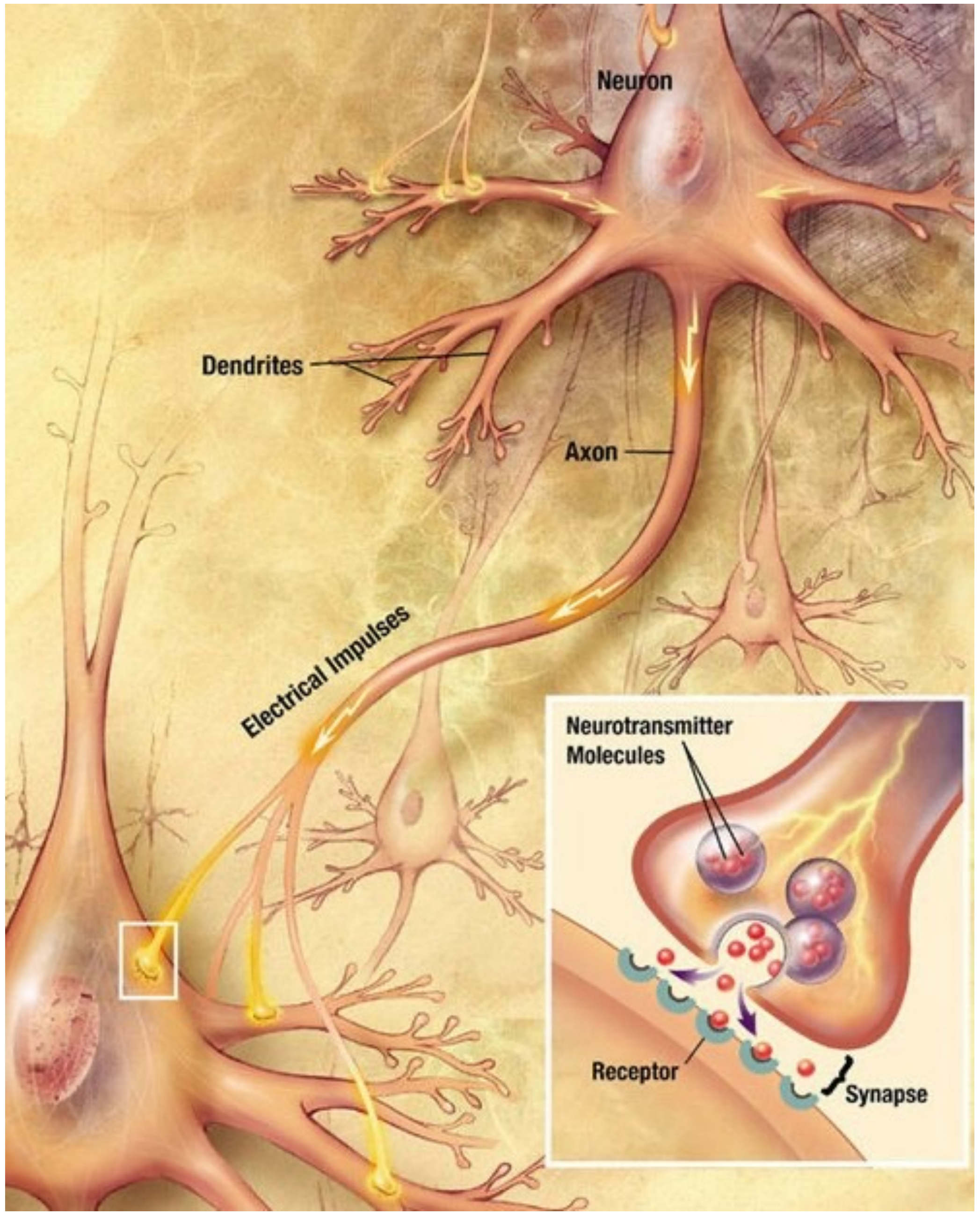
Richard Kelley

Last Time

- Symbolic Representation and its limits.
- Intro to Neuroscience.
- **Perceptrons**

Neurons

- Axons
  - Carry signals away from cell body
- Dendrites
  - Receive signals
- Impulses
  - The signals a neuron sends. These seem to "fire" in pulses.
- Thresholds
  - It seems like some threshold needs to be met before a neuron "fires" and sends signals to its neighbors.



https://en.wikipedia.org/wiki/Neuron#/media/File:Chemical_synapse_schema_cropped.jpg

Rosenblatt’s Perceptron

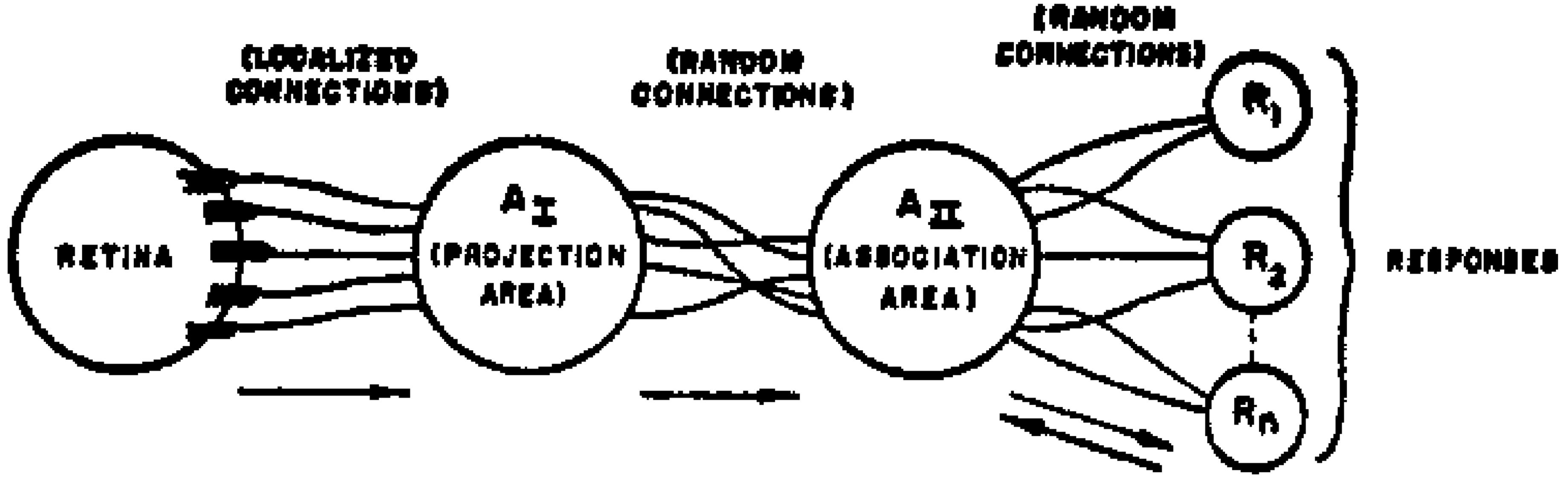
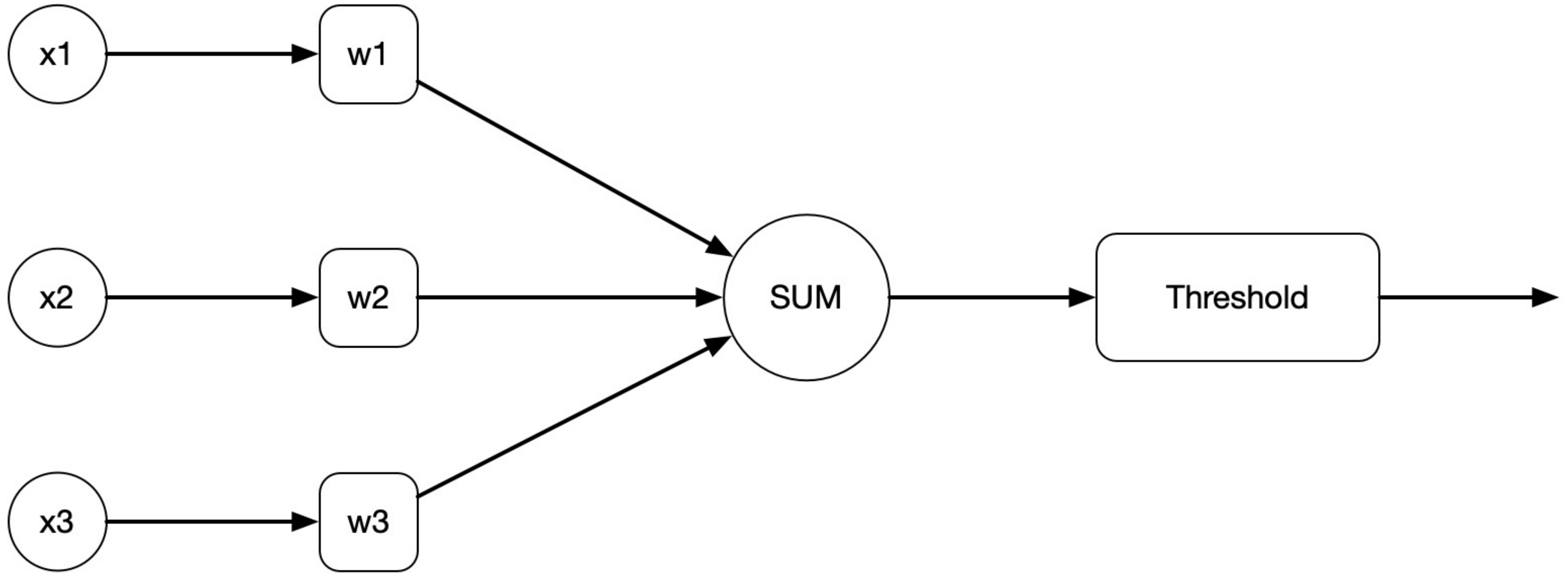


FIG. 1. Organization of a perceptron.

Rosenblatt, F. (1958). *The perceptron: A probabilistic model for information storage and organization in the brain*. **Psychological Review, 65**(6), 386–408.

A Simpler Model

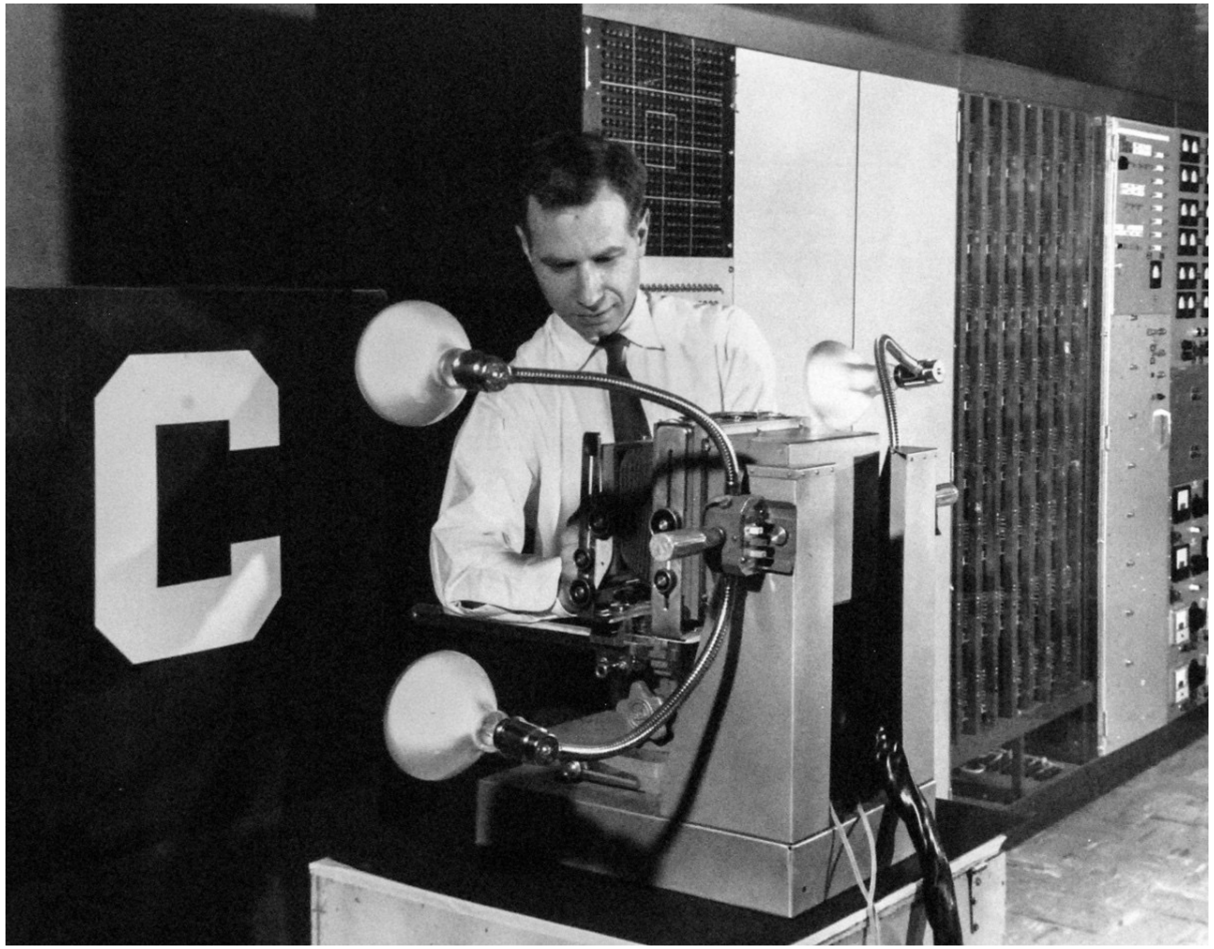
- The spiking model is too complicated (and wouldn’t have worked on a 1950s computer.
- There’s a simpler alternative, called the ***perceptron***:



Perceptron Demo

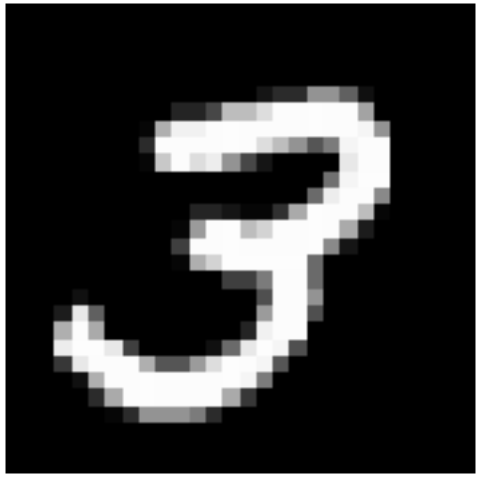
What can we do with this?

- **Classification**
  - Recognize Images. The network outputs a 1 when the input takes a certain shape.
  - Each pixel in the image is an input.
  - Requires that the system “learn” to distinguish one thing from another.
    - Adjusting weights and thresholds.



Classification

- The task of assigning labels to things.
- Very fundamental
  - Gn 1:5 “God called the light Day, and the darkness he called Night.”
  - Gn 2:19 “…whatever the man called every living creature, that was its name.”
- Nowadays, we don’t make up new names for things as much as recognize when something is a particular “kind” of thing.



what kind of thing is this?

How do we write programs to classify things?

1. Decide on the kind of inputs our program will accept.
   - Images of a certain size (28x28 pixels, for example).
   - Audio files.
   - Strings of text.
2. Decide on the kinds of outputs your program will produce. These are called the ***class labels***.
   - “The letter c”, “The letter a”, “The letter b”, …
   - one, two, three, four, …
   - “This is the digit 3”, “This is not the digit 3”
3. Figure out some ***features*** of the input that determine what the label should be.
   - Ideally these are easy to represent in a computer…
4. Write a program that, for a given input, uses the features of that input to figure out the correct class label.
   - We call this process ***classification*** or ***prediction***.

Examples of Classification

- Digit classification
- Email spam detection
- Tumor detection from medical images
- Credit approval for loans
- News article topic classification
- ***Sentiment classification***

Terminology for Classification Problems

- ***Binary Classification***
  - A classification problem where you only have two classes.
- ***Multiclass Classification***
  - A classification problem where you have many classes.
- ***Classifier***
  - A program that solves a classification problem.

- Can you have a classification problem with “infinitely many” classes?

Perceptrons and Classification

- The Perceptron is set up for solving binary classification problems.
    - The outputs are always 0 or 1.
- You just take your class labels and decide that one of them corresponds to 0 and the other corresponds to 1.
    - “hot dog” gets the label 0
    - “not hot dog” gets the label 1

Problem: How do we represent inputs?

- Each type of classification problem seems to deal with a different kind of input.
    - A spam classifier seems really different from a “cats vs. dogs” classifier.
- The features of the input that matter seem to change for each problem too.
    - Finding features that are good for prediction requires a lot of practice, and skill, and luck.
- Then there’s the issue that perceptrons require input in the form of a list of numbers.

Vector Embeddings

- It would be nice if we could come up with a single input representation that worked for every kind of input.
  - It would have to be very general to work with audio, images, text, laser scans, etc. ...
- One of the key idea of Artificial Intelligence (in the last 20 years especially) is the idea of a ***vector embedding***.
  - For any classification problem, convert the "natural" input representation into a ***vector***.
  - Build your classifier to have two stages:
    - Convert your input to a vector.
    - Classify the vector.
- The process of converting an input to a vector is called ***embedding***.

Vectors

- A ***vector*** (in the AI sense) is an ordered list of (usually decimal) numbers.
  - Often written in square brackets or parentheses
  - Can be short or long.
  - [0.0, 0.0], [1.0, 3.14], [1.0, 0.0, 2.0, 5.0, 7.1, 3.14, 2.718]
- The length of the list of numbers is called the ***dimension*** of the vector.
  - [0.0, 0.0] is a two-dimensional vector.
  - [1.0, 0.0, 2.0, 5.0, 7.1, 3.14, 2.718] is a 7-dimensional vector.
- These vectors correspond to the vectors you may have seen in a math or physics class.

Vectors and Perceptrons

- Now we see that perceptrons take *vectors* as inputs and assign a label of 0 or 1 to those vectors.
- Our demos have been in 2D because that’s easy to visualize, but you could make a perceptron that takes *N*-dimensional vectors, for any *N*.

Classification and Perceptrons

- Our very general task of "building a classifier" now becomes a sequence of well-defined tasks:
  - Figure out how to embed your input as a vector.
  - Choose the weights and threshold that get the "best" performance.
    - This could be accuracy, or something else.
- Question: how do we figure out the embeddings?
- Question: What are the potential downsides of embeddings?
  - Are there ethical issues?
  - *You can't control what you don't measure.*

Summary

- Trying to model the neuron leads to the idea of a simple model called the *perceptron*.
  - Perceptrons introduce the idea of *weights* that ultimately define a classifier.
- Using a perceptron to perform classification leads to the idea of *vector embeddings.*
- Open Questions (for future lectures)
  - How do we find “good” weights? Can we find the “best” weights?
  - Are there any fundamental limits to how well perceptrons work?