AI Literacy & DS Ethics

The Jackson Laboratory

Module 2: Inside Neural Networks

Day 3 review

What Is one new thing you learned yesterday?

What lingering questions does everyone have about yesterday’s material?

Learning Goals:

What are the different types of AI model and how are they used

You will be able to:

- Break down a model into its basic layers
- Identify what type of layer sets each model apart
- Understand the purpose of each type of layer

Neural Networks: Perceptrons & Fully connected Networks

Not enough regression

Learning Goals:

What are neural networks and how do they work?

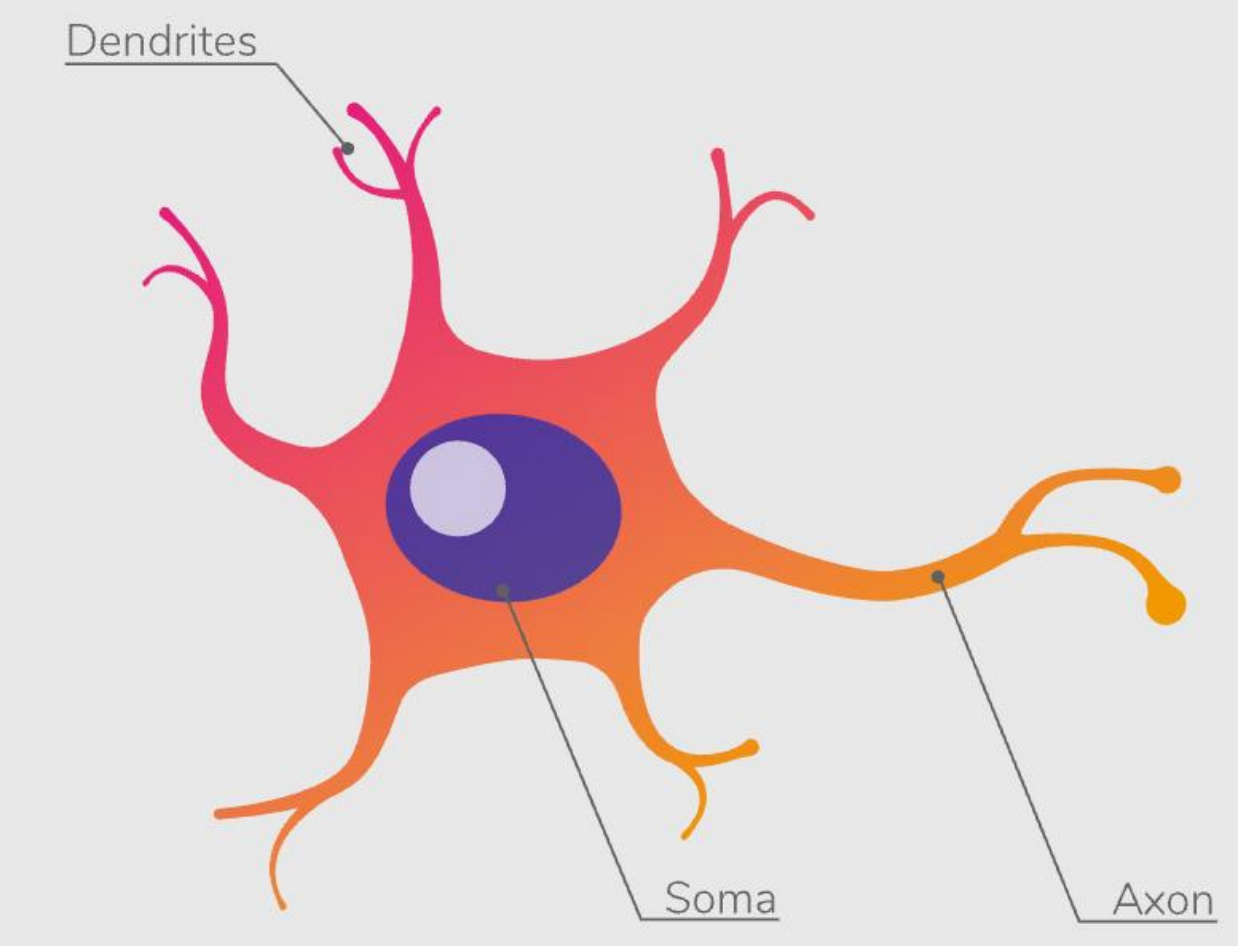
You will be able to:

- Explain what a perceptron is
- Combine perceptrons into a fully connected layer

A brief aside: Biological Neurons

Neurons are the foundational unit of the nervous system.

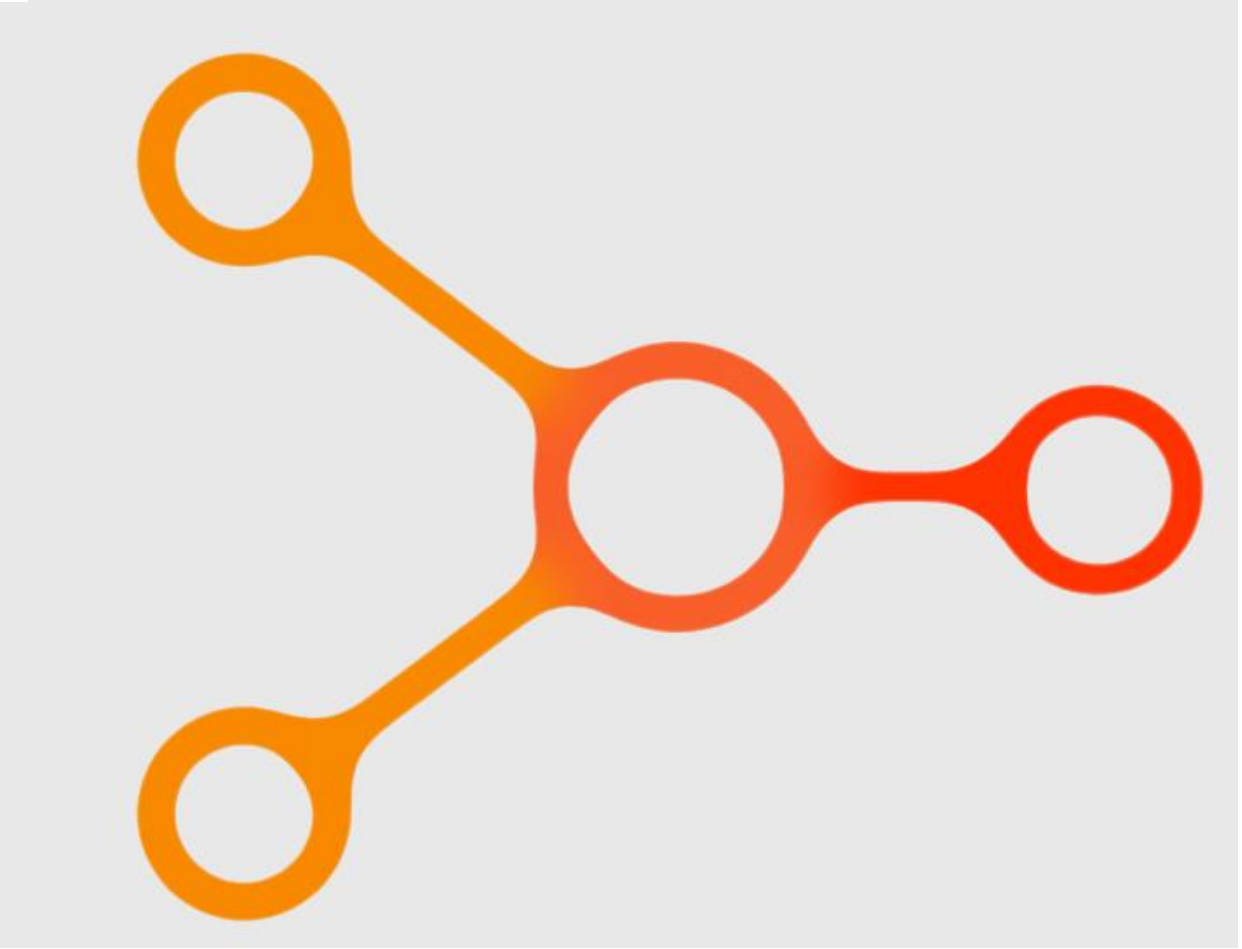
- Dendrites take in signals from other cells
- The cell body/soma processes these signals and decides whether to “fire”
- When the neuron fires a signal is sent through the axon to other cells



Can we create a mathematical model based on neurons?

Perceptrons are the foundational unit of neural networks.

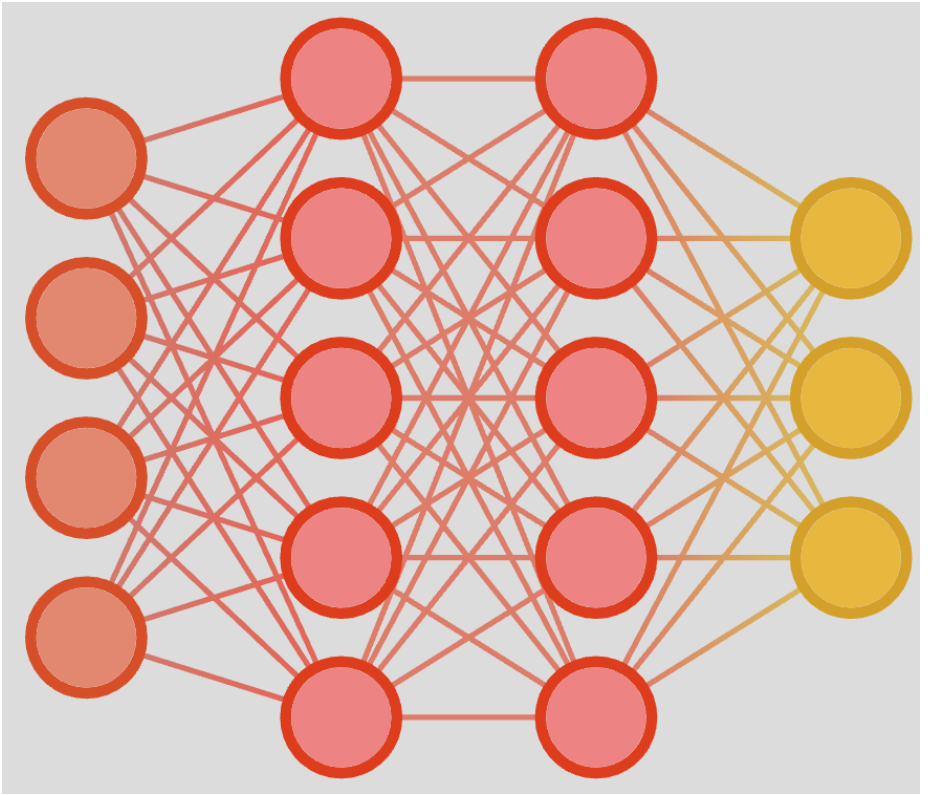
- A perceptron takes one, or (usually) more, inputs.
- These inputs are combined, using a weighted sum, into a single output.
    - Important term: “Linier Algebra”
- This output can then be modified by an activation function
- You may have noticed the behavior of an individual perceptron is basically a regression model like we saw yesterday



We've recreated regression, so what?

The brain is not made up of just one neuron

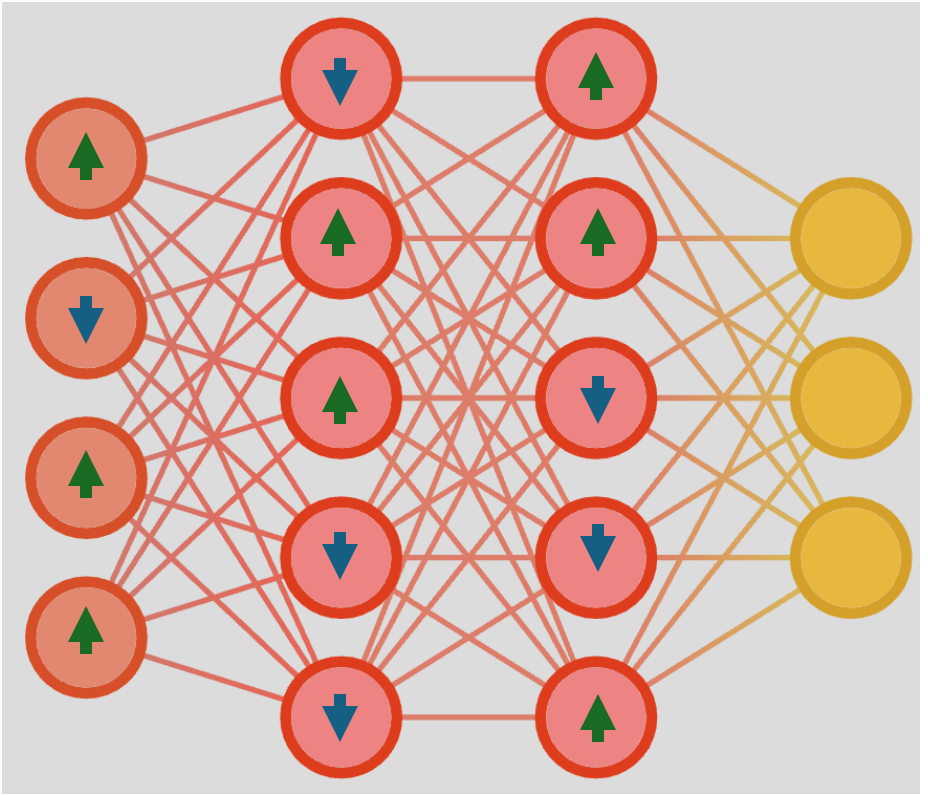
- By connecting multiple perceptrons together we create an artificial neural network
  - Two important terms: Fully connected & Feed forward
- Artificial neural networks can solve much more complex problems than "simple" regression



How do we learn the weights?

Remember gradient decent?

- We now have a much more complex function
- The chain rule from calculus let's us break up a complex derivative
- We can see how each perceptron is contributing to the gradient and determine how to address its weights
- This is called "Back propagation"



To discuss after the activity

This activity demonstrates how a fully connected network operates

Can you think of any ways to improve our HNN’s performance?

Human Neural Network (HNN) Activity

1. Split into three groups, Input, hidden, and output, each will be provided instructions

2. Input will receive a picture of a hand written didgit and “Activate” the input nodes on the board based on the didgit

3. Hidden will then activate their nodes based on the which Input nodes are activated

4. Finaly the output will predict which digit was given to the input based on which hidden nodes were activated

Activity Discussion

This activity demonstrates how a fully connected network operates

Can you think of any ways to improve our HNN’s performance?

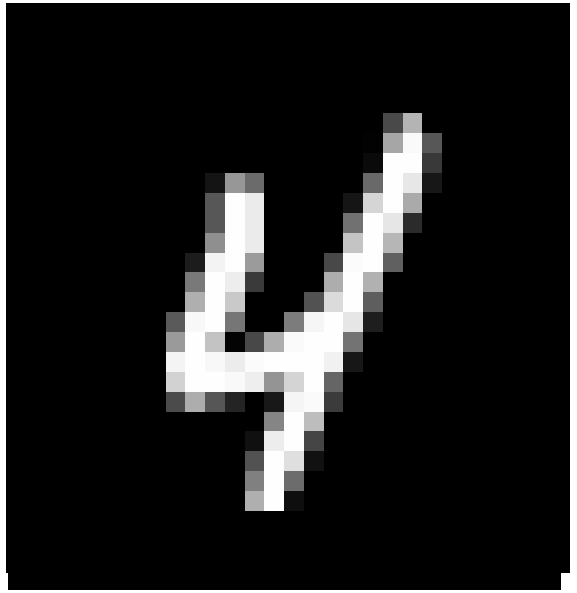
Putting it all together

How do fully connected networks work in practice?

https://okai.brown.edu/chapter6.html

What are some major limitations of Fully Connected Layers?

1. Scalability
2. Scalability!
3. Have I mentioned scalability?
4. Restricted Input



vs

Many networks end in one or more fully connected layer(s).

What we discuss the rest of today and tomorrow will by and large be ways to take input that is too large/amorphous to go directly into a Fully connected network and finding a “feature representation” that accurately represents that input

Review

Fully connected Networks

- Built from multiple regression models combined together
- Can solve complex non-linear problems
- Still used as a fundamental part of many more advanced types of network
- Need to represent input data as a vector of fixed size

Neural Networks: Convolutional Neural Networks

Finding patterns in matrixes

Learning Goals:

What are Convolutional Neural Networks (CNNs) and how do they build on fully connected neural networks ?

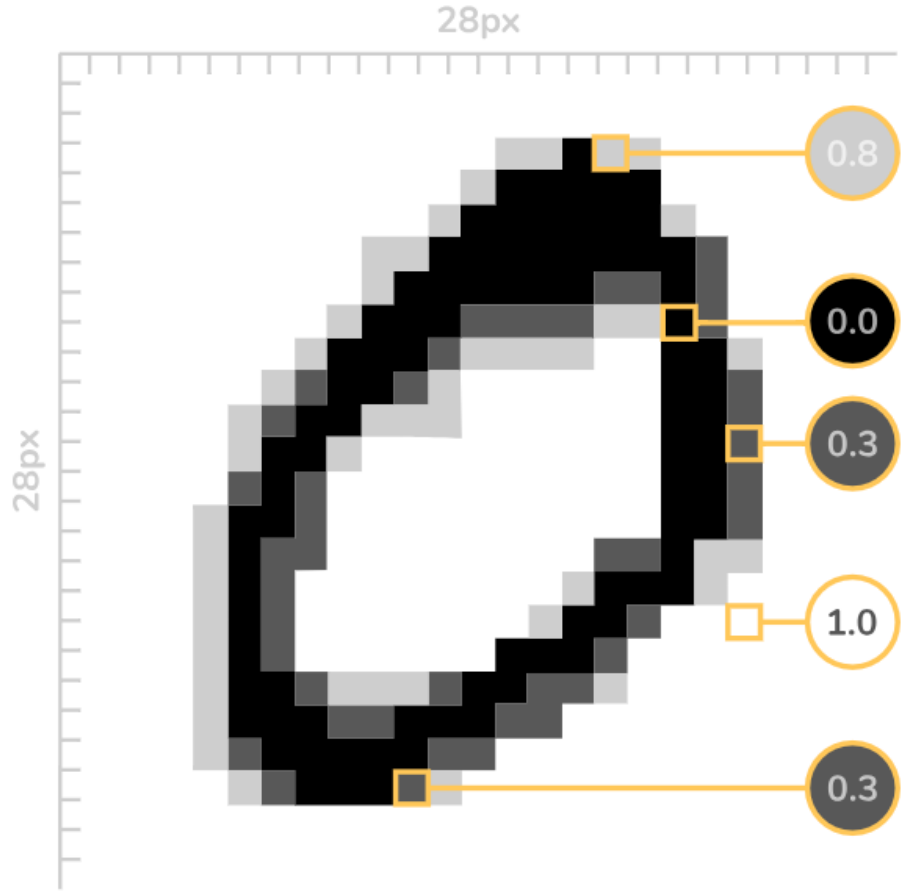
You will be able to:

- Explain what a convolution does
- Contrast a convolutional layer to a fully connected layer
- Determine when to use convolutional layers

A brief aside: An image is worth HxWXC numbers

Think of your computer monitor

- Anything you see on it is thousands of little points of light (pixels)
- Color is just a combination of red green and blue points of light
- We know how bright each point is
- We can represent any image as a rectangle of numbers aka a matrix



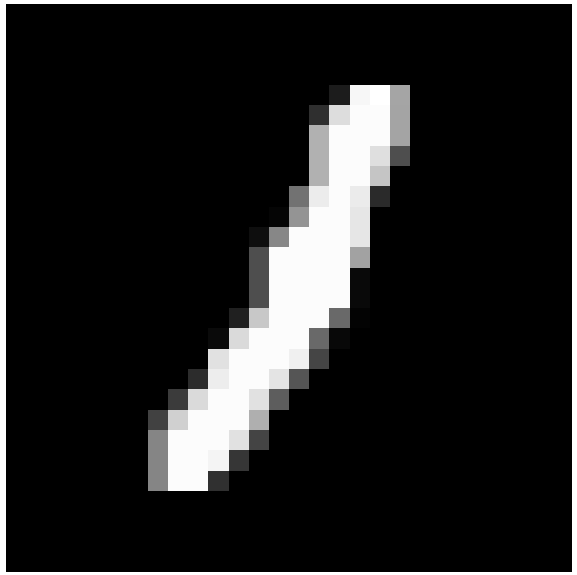
Why do we need convolutions?

MNist has 28pxl X 28 pxl gray scale images

- Each image can be flattened into a 784-element vector

This picture of a dog is 512pxls X 910pxls and in color.

- If we flattened it, it would be nearly five hundred thousand numbers
- We need a way to accurately describe the image with far fewer numbers

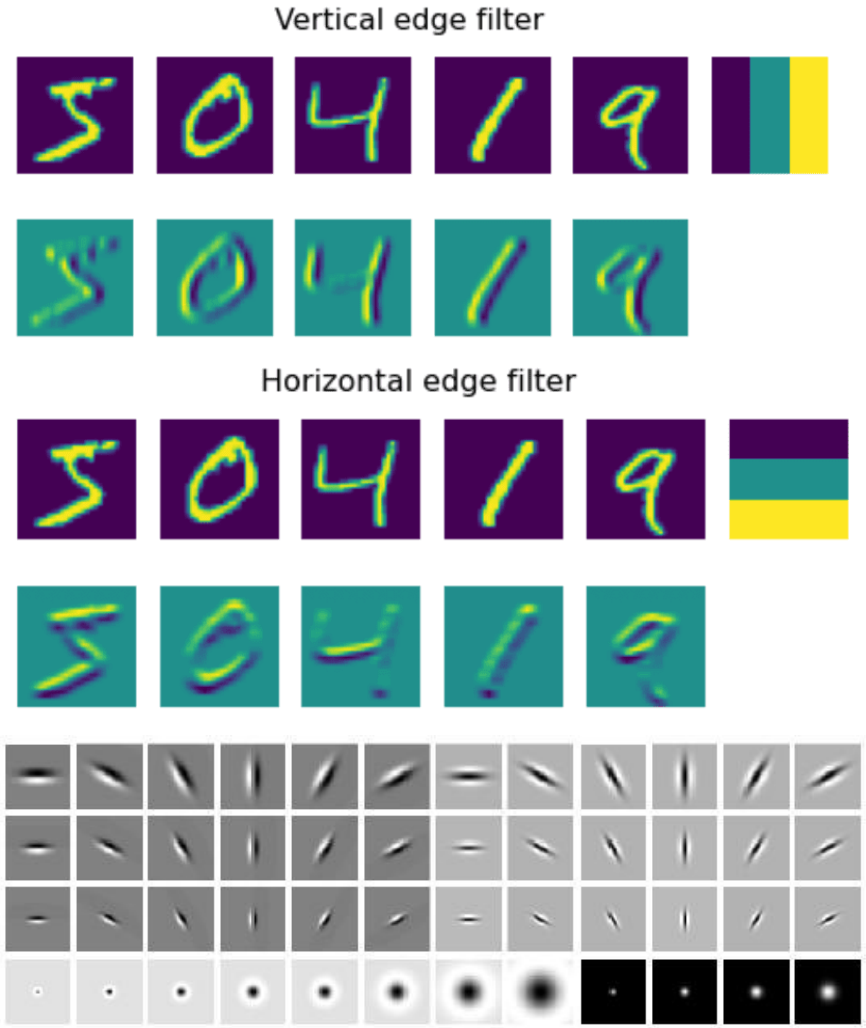


VS

What is a convolution?

Convolutions are filters that look at small patches of an image to find patterns

- We can create filters manually to find specific shapes
    - This is what the input layer was doing in the HNN activity
- Instead of telling the network what to look for, we can let it learn what shapes are important just like we did with the perceptrons



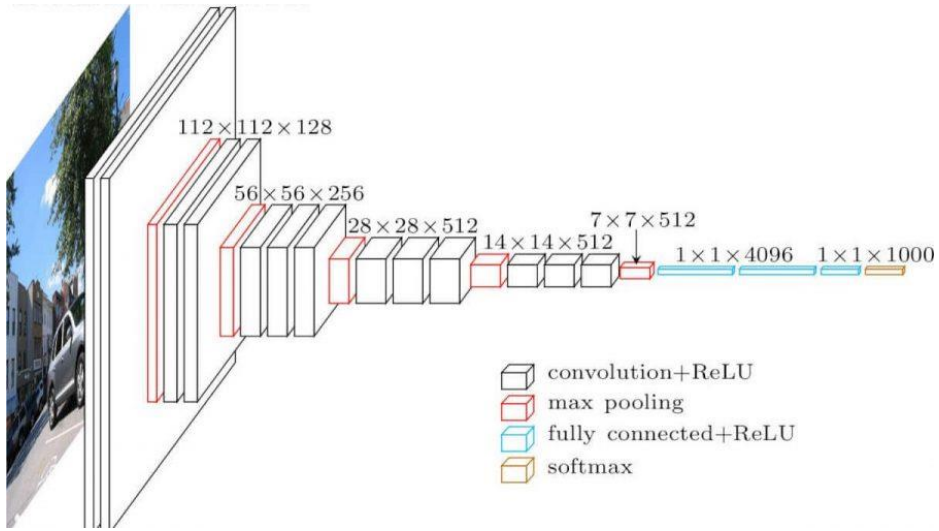
How do we find more complex shapes?

Multiple perceptrons ➔ Fully connected layer

Multiple FCs ➔ Fully connected network

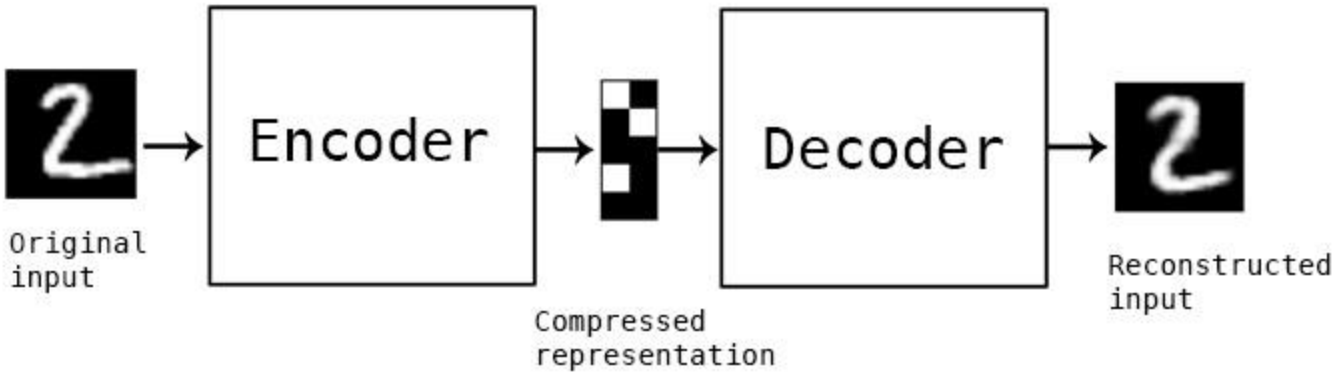
Multiple convolutions ➔ Conv layer

Multiple conv layers ➔ Convolutional NN



What about going in reverse?

Deconvolutions are the inverse of convolutions



- Convolutions take a series of patterns and turn an image into a description of that image
- Deconvolutions take a description and “draw” an image based on it.
- An alternative to ending a network in a fully connected layer when you want a matrix instead of a vector. E.g. image generation

To discuss after the activity

This game demonstrates using simplified patterns to describe complex images

What kinds of shapes were more informative?

Were shapes from certain parts of the image more informative than others?

Contour Classification Activity

1. Split into three groups: 6 inputs, 4 H1, and 2 H2, 1 output
2. Inputs on each of your transparency sheets trace a 1” curve on the provided image, and pass each to a different H1
3. H1 from your 6 sheets and select 2 pairs, pass a pair to each H2
4. H2 from your 4 pairs select a set of 4 sheets (keeping pairs together) and pass to the output
5. Output based on your stack of 8 sheets guess whether the image was an elephant, a walrus or a giraffe.

Activity Discussion

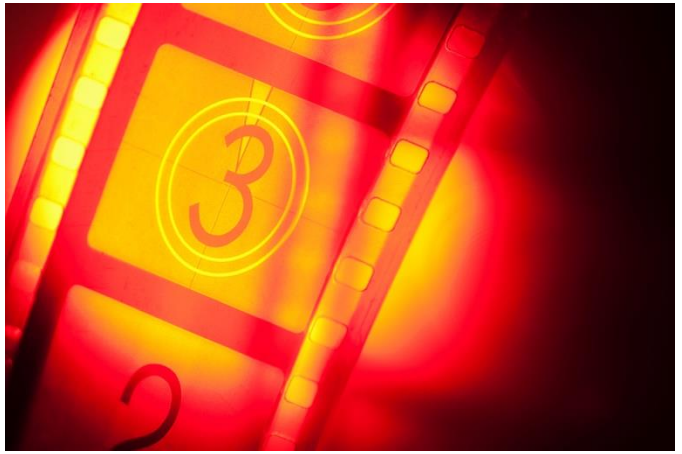
This game demonstrates using simplified patterns to describe complex images

What kinds of shapes were more informative?

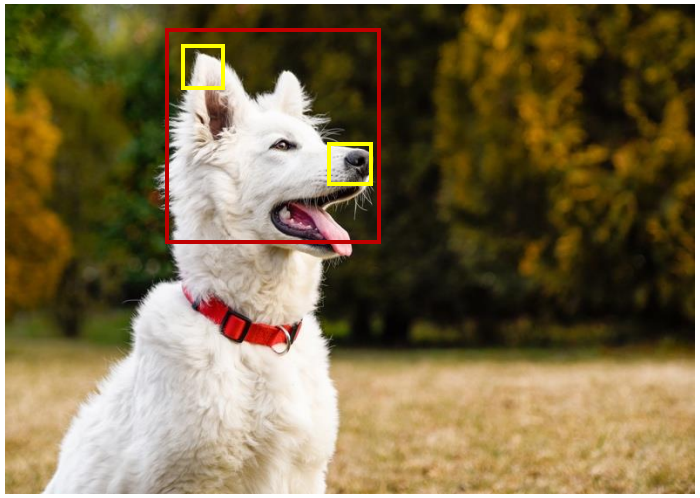
Were shapes from certain parts of the image more informative than others?

What are some major limitations of convolutional layers?

1. No memory
    - Imagine working with a video instead of an image
    - There is temporal context the convolution can not see



2. Limited field of view
    - Because of the limited size of a convolution, you have limited context for fine details



3. Fixed number of channels

Review

Images can be thought of as matrixes of numbers

Convolutional neural networks

Strengths:

- Can efficiently summarize large input matrixes
- Can be reversed to (re)create an image

Limitations:

- No memory
- Limited field of view
- Fixed number of channels

Neural Networks: Recurrent Neural Networks

Working with Sequences

Learning Goals:

What are Recurrent Neural Networks (RNNs) and how do they allow us to work with sequential data such as text?

You will be able to:

- Tokenize input data
- Explain why you would add a loop to a network
- Infer the limitations/downsides of Recurrent layers

A brief aside: Tokenization

Q: How do we perform calculations on data that isn't numbers?

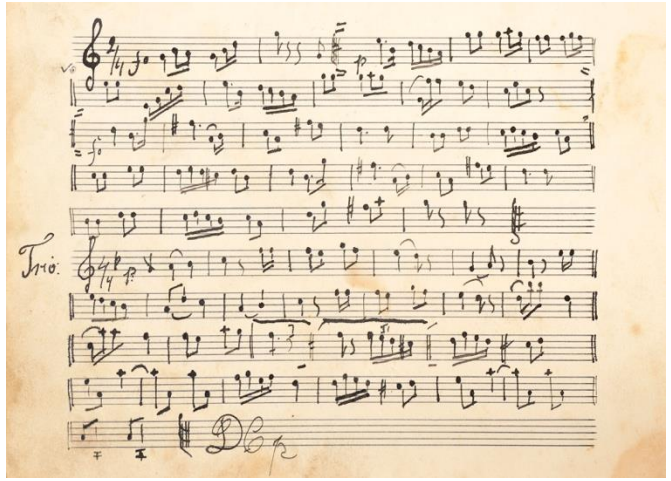
A: A bit of a trick question, because we can't do so directly.

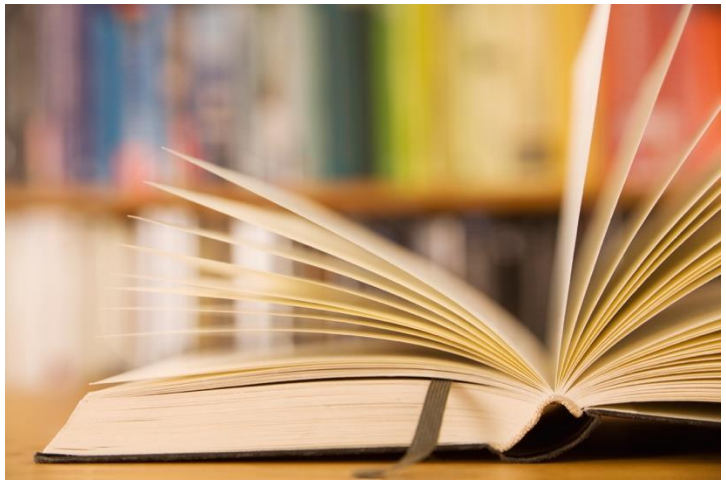
But we can use numbers to represent non-numeric data.

Flow of tokenization used in AI/ML

- Word level
- Character level
- Sub-word level

What sorts of data are sequential?





How is Sequential data different?

Order Matters

- Earlier parts of sequential data provide context for later parts
- Imagine hearing the punchline of a joke before the setup (or without the set up at all)
- Consider how the meaning of this sentence changes if we change the order of two words

Eve spied on Alice

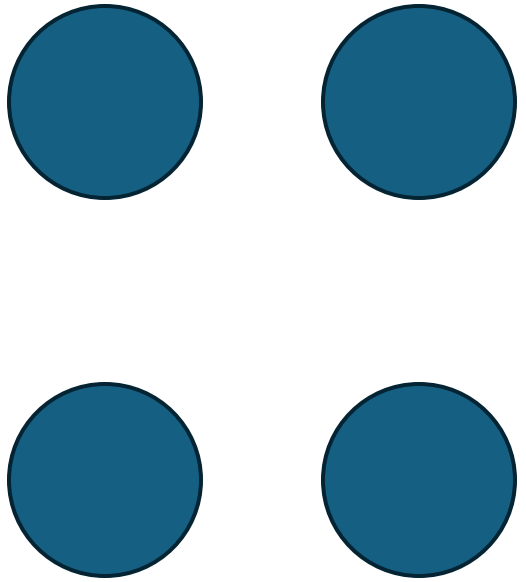
What problems does this create?

We often don’t know the size of our input

- Two sentences can have different numbers of words.
- This means each token must be considered individually

“feed forward” networks only see a snapshot of what is going on

- They possess no memory of previous inputs

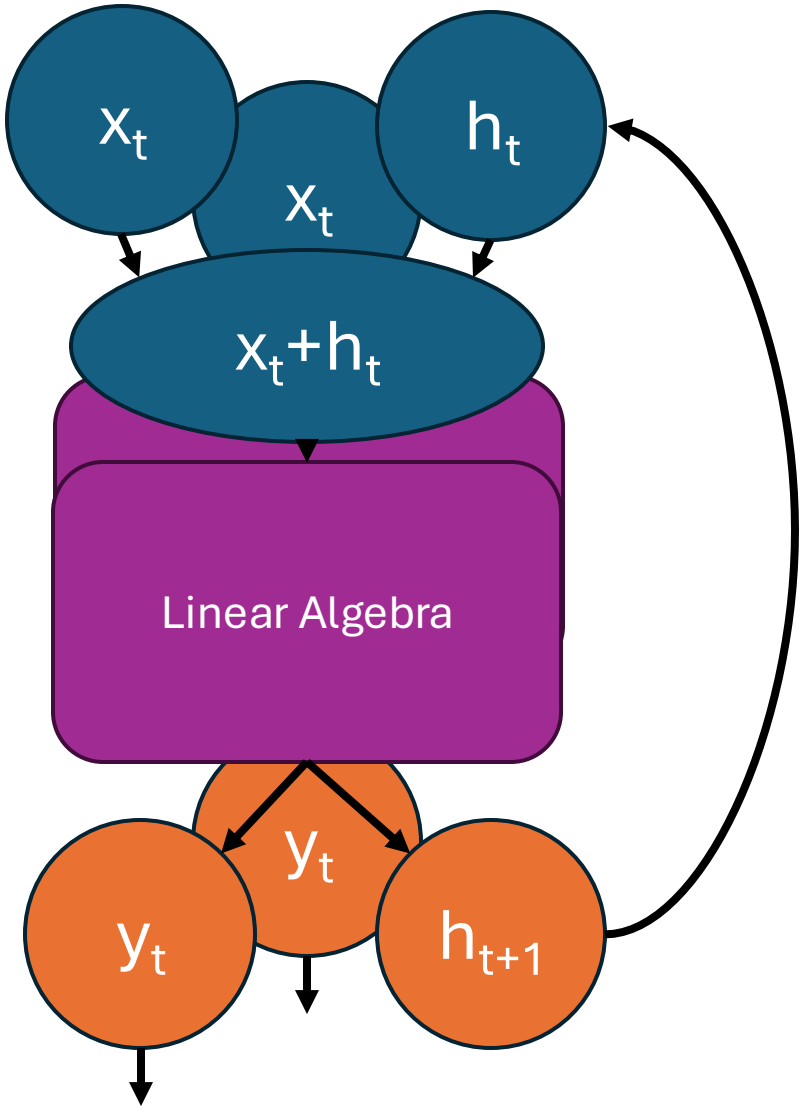


How can we give our network a memory?

Recurrent layers to the rescue

We create a new vector **h** that can store information for the layer

Each input **x** is connected to **h** and fed through the layer to create an output **y** and a new **h** that is used with the next input



To Discuss after the activity

This game demonstrates how considering the sentence as a sequence improves interpretability

Can you think of any limitations with considering the sentence in this way?

Can you think of a method that would improve on this sequential processing?

Sentence Analysis Activity

1. Break into small groups
2. Each group gets a stack of index cards
    - The top card is a random word from the target sentence
    - The rest of the cards are the sentence written out sequentially
3. Try and guess the meaning of the sentence from the top card
4. Flip each of the remaining cards over one by one and try to guess the meaning of the sentence in as few cards as you can
5. Pass your stack to the next group and try again with the stack you receive.

Activity Discussion

This game demonstrates how considering the sentence as a sequence improves interpretability

Can you think of any limitations with considering the sentence in this way?

Can you think of a method that would improve on this sequential processing?

What are some major limitations of recurrent layers?

1. Limited memory
    - There is an upper limit to how much information can be stored in **h**
    - This puts an upper limit on number of tokens the network can consider
2. Recency bias
    - Because **h** changes with each input the networks memory prioritizes more recent tokens
3. No token prioritization
    - The network has no way of learning which tokens are more important
4. Limited parallelizability
    - Whenever you treat something as a sequence you have to process it in order limiting your ability to split the work across multiple processors

What are some major limitations of recurrent layers?

1. Limited memory
    - There is an upper limit to how much information can be stored in **h**
    - This puts an upper limit on number of tokens the network can consider

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

VS

What are some major limitations of recurrent layers?

2. Recency bias
    - Because **h** changes with each input the networks memory prioritizes more recent tokens

The quick brown fox jumps over the lazy dog

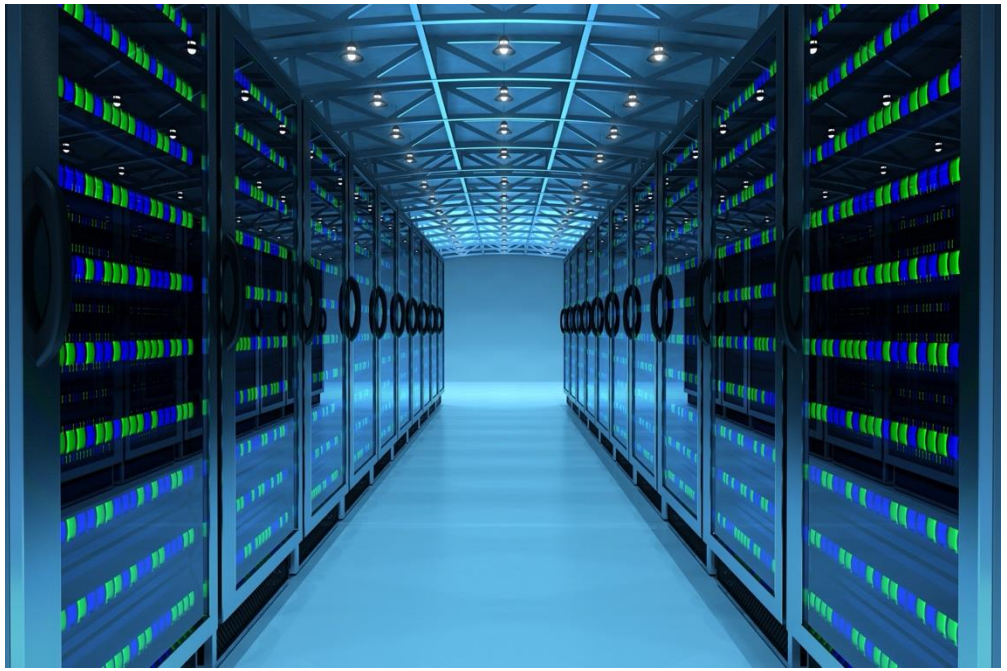
What are some major limitations of recurrent layers?

3. No token prioritization
    - The network has no way of learning which tokens are more important

The quick brown fox jumps over the lazy dog

What are some major limitations of recurrent layers?

4. Limited parallelizability
    - Whenever you treat something as a sequence you have to process it in order limiting your ability to split the work across multiple processors



Review

Tokenization

- By representing things like words with vectors of numbers we can input them into neural networks

Recurrent neural networks

Strengths:

- Adds memory to a neural network
- Incorporates sequential context
- A lot of things are sequential

Limitations:

- Computationally expensive
- Limited memory capacity and range
- Can’t learn relative importance of tokens

Day 4 review

What questions does everyone have about today's material?

Next up: Generative Adversarial Networks & Transformers