UVA DATA SCIENCE

# DS 6051 Decoding Large Language Models

## Basics of Language Modeling

Chirag Agarwal
Assistant Professor
School of Data Science
University of Virginia

# Recap..

- Course overview
- Introduction to Natural Language Processing
- How to represent words as vectors
- Word2vec

Other popular methods:

- xgboost - group of trees
- lightgbm -

# Supervised Learning (Classification)

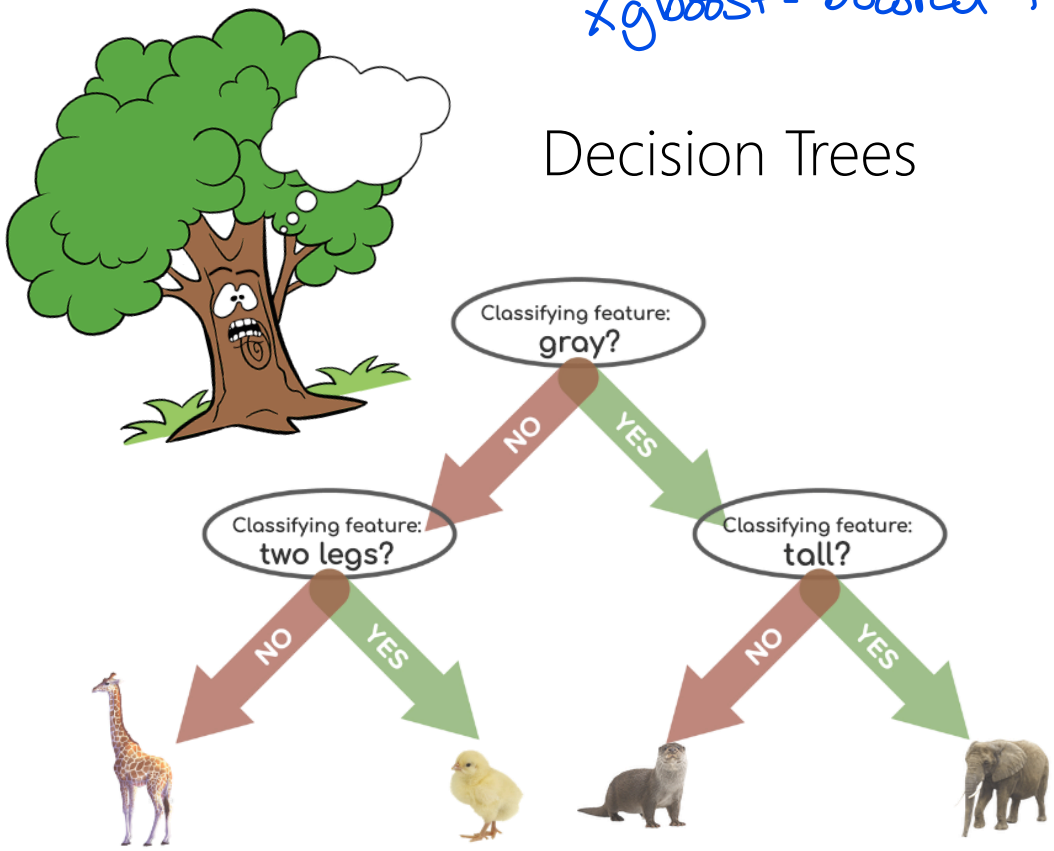
- Inputs $X = \{x_1, x_2, \dots, x_N\}$ and labels $Y = \{y_1, y_2, \dots, y_N\}$
- For NLP, $x_i$ can be words, sub-words, sentences, documents, etc.
- Example tasks:
  - Sentiment classification
  - Topic classification
  - Buy/Sell decision
  - Recommending Products

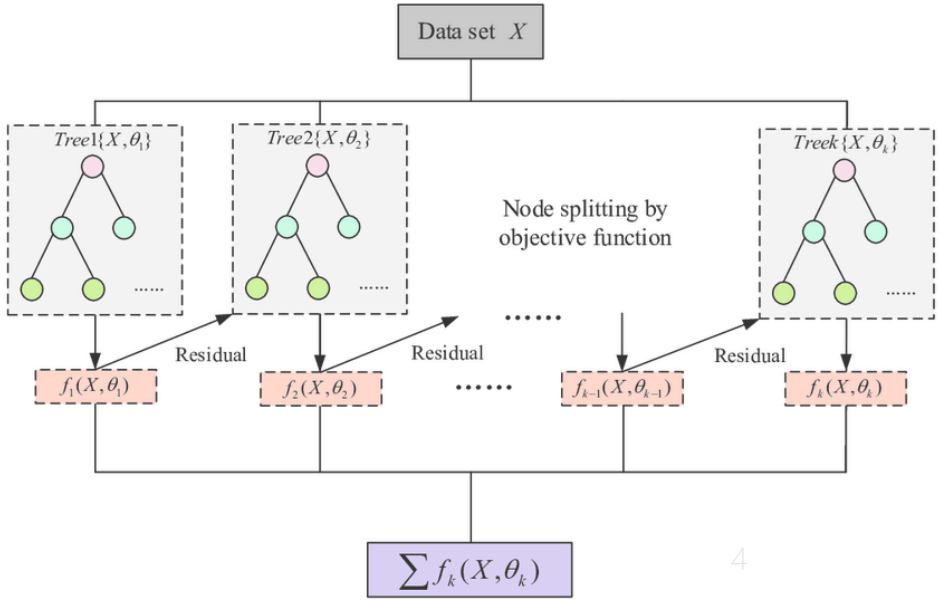
# Supervised Learning Algorithms

Kaggle examples

xgboost - boosted trees

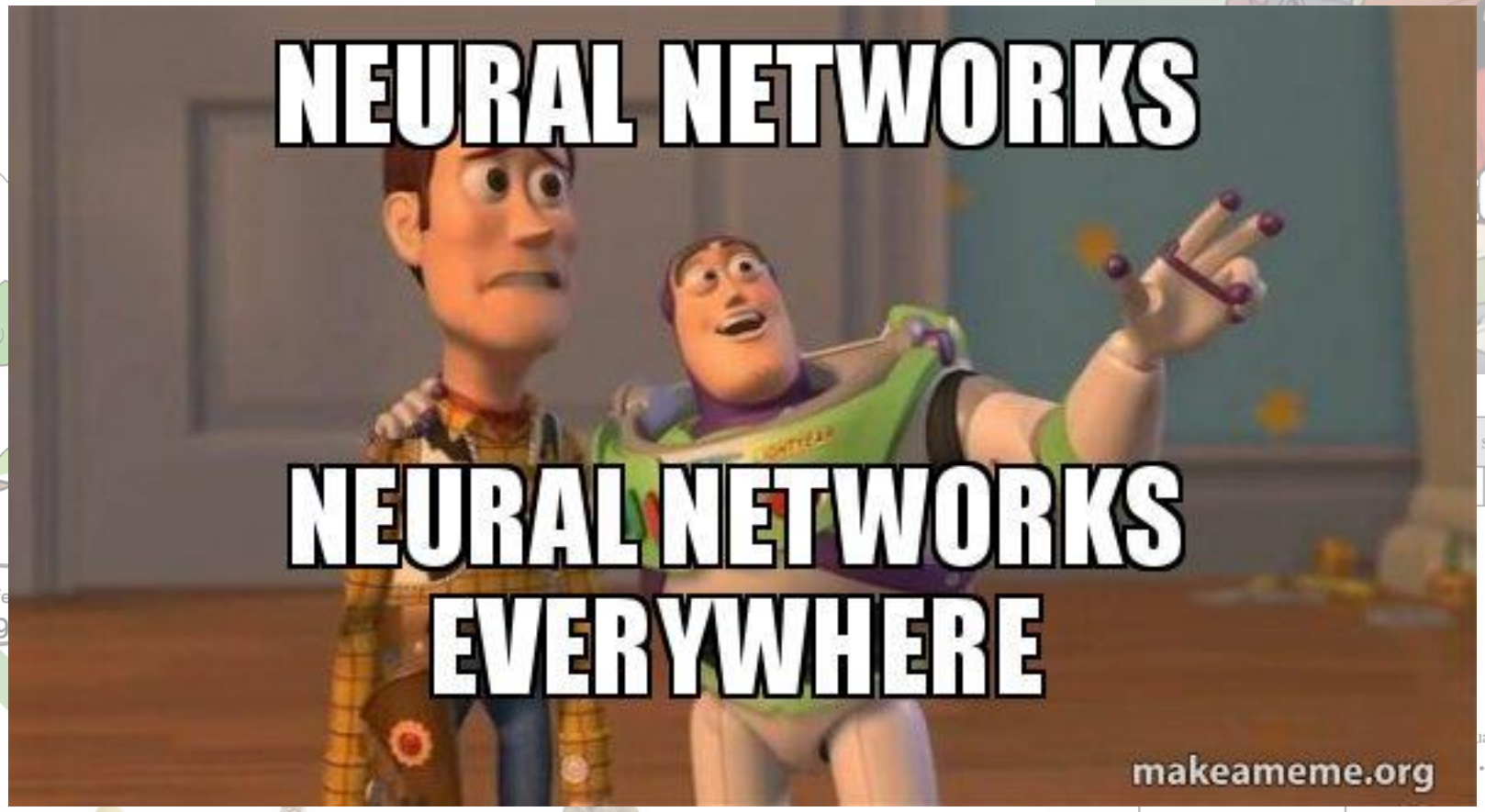
Decision Trees

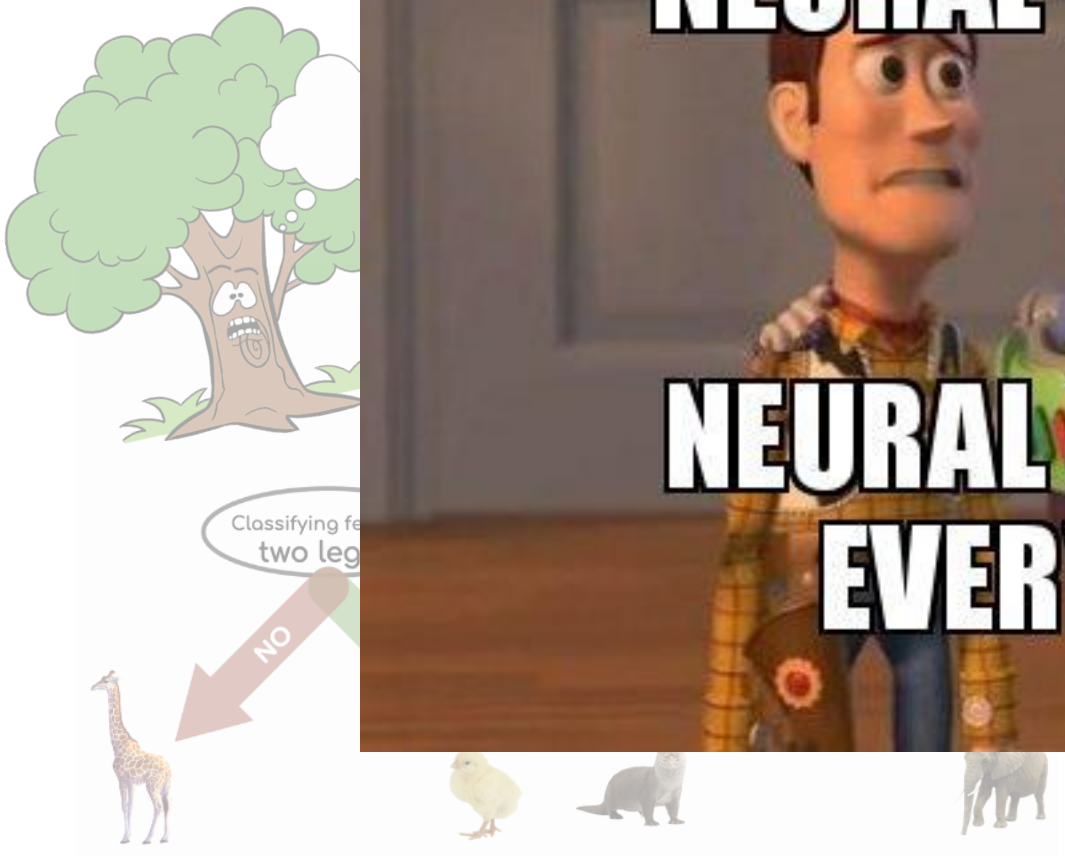


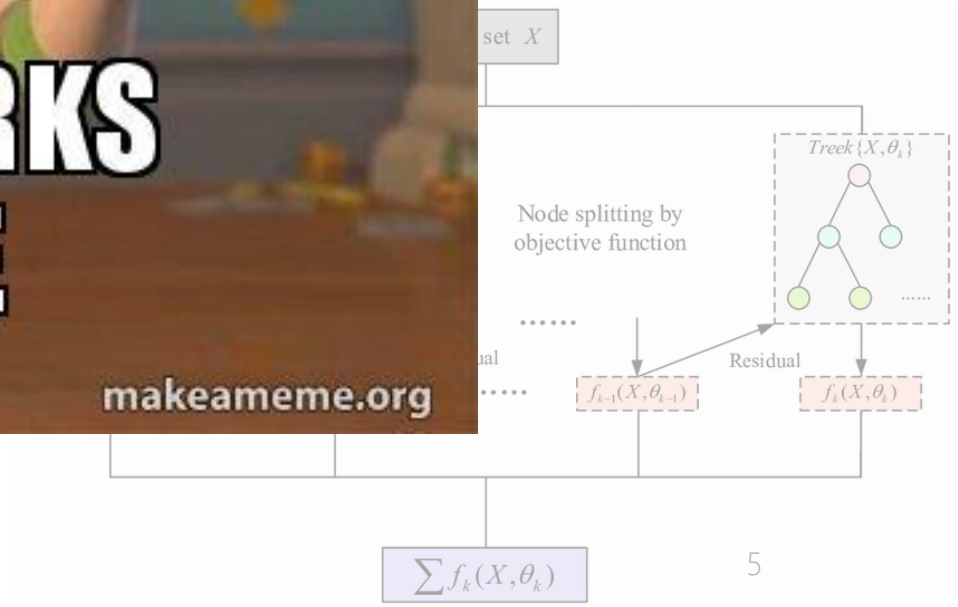




https://raise.mit.edu/daily/neuralnetworks/

# Supervised Learning Algorithms






https://raise.mit.edu/daily/neuralnetworks/

# 1958 New York Times…

## NEW NAVY DEVICE LEARNS BY DOING

### Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

WASHINGTON, July 7 (UPI) —The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

The embryo—the Weather Bureau's $2,000,000 "704" computer—learned to differentiate between right and left after fifty attempts in the Navy's demonstration for newsmen.

The service said it would use this principle to build the first of its Perceptron thinking machines that will be able to read and write. It is expected to be finished in about a year at a cost of $100,000.

Dr. Frank Rosenblatt, designer of the Perceptron, conducted the demonstration. He said the machine would be the first device to think as the human brain. As do human beings, Perceptron will make mistakes at first, but will grow wiser as it gains experience, he said.

Dr. Rosenblatt, a research psychologist at the Cornell Aeronautical Laboratory, Buffalo, said Perceptrons might be fired to the planets as mechanical space explorers.

**Without Human Controls**

The Navy said the perceptron would be the first non-living mechanism "capable of receiving, recognizing and identifying its surroundings without any human training or control."

The "brain" is designed to remember images and information it has perceived itself. Ordinary computers remember only what is fed into them on punch cards or magnetic tape.

Later Perceptrons will be able to recognize people and call out their names and instantly translate speech in one language to speech or writing in another language, it was predicted.

Mr. Rosenblatt said in principle it would be possible to build brains that could reproduce themselves on an assembly line and which would be conscious of their existence.

In today's demonstration, the "704" was fed two cards, one with squares marked on the left side and the other with squares on the right side.

**Learns by Doing**

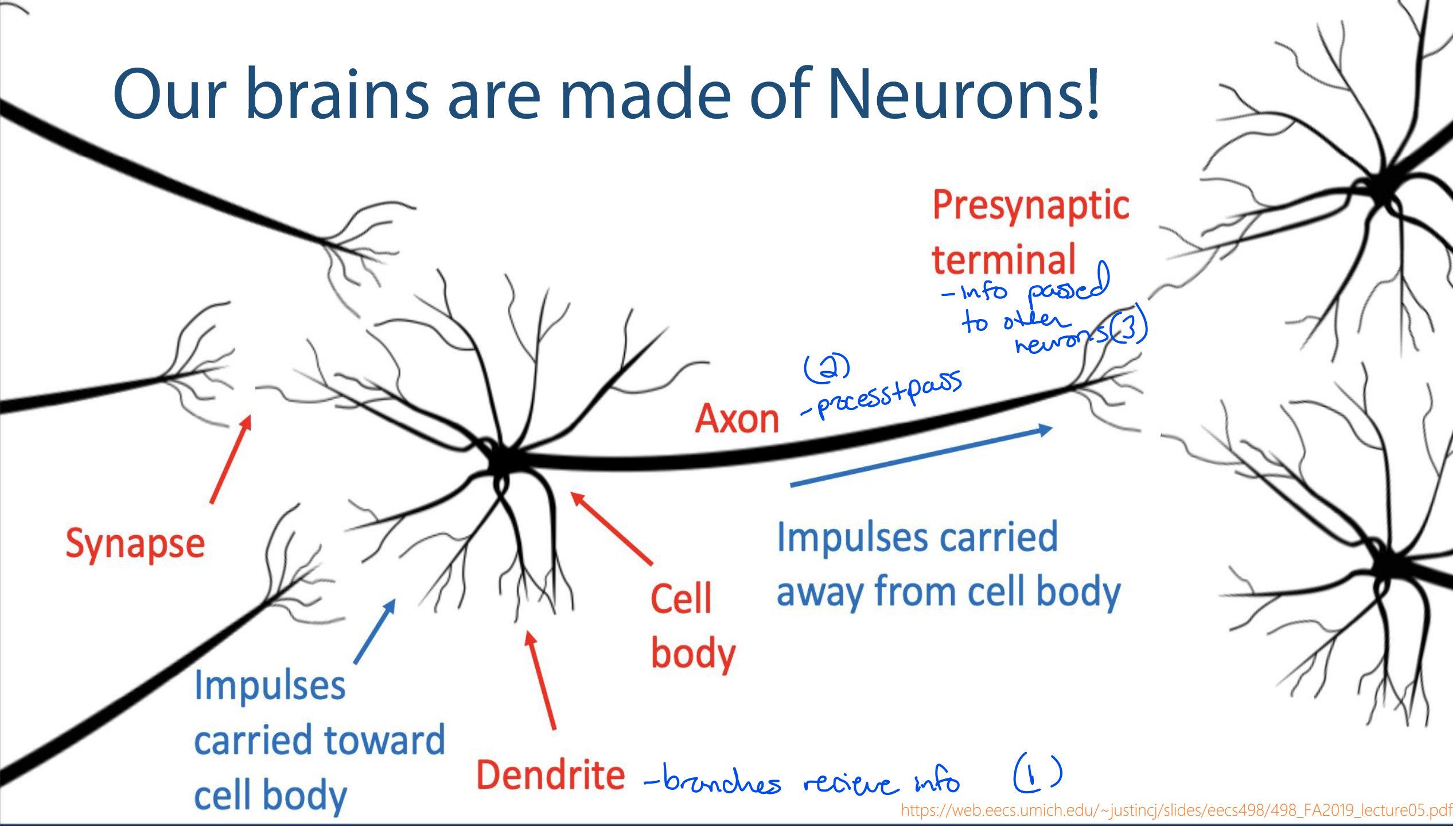
In the first fifty trials, the machine made no distinction between them. It then started registering a "Q" for the left squares and "O" for the right squares.

Dr. Rosenblatt said he could explain why the machine learned only in highly technical terms. But he said the computer had undergone a "self-induced change in the wiring diagram."

The first Perceptron will have about 1,000 electronic "association cells" receiving electrical impulses from an eye-like scanning device with 400 photo-cells. The human brain has 10,000,000,000 responsive cells, including 100,000,000 connections with the eyes.

Artificial way to mimic human brain

# Our brains are made of Neurons!




https://web.eecs.umich.edu/~justincj/slides/eecs498/498_FA2019_lecture05.pdf

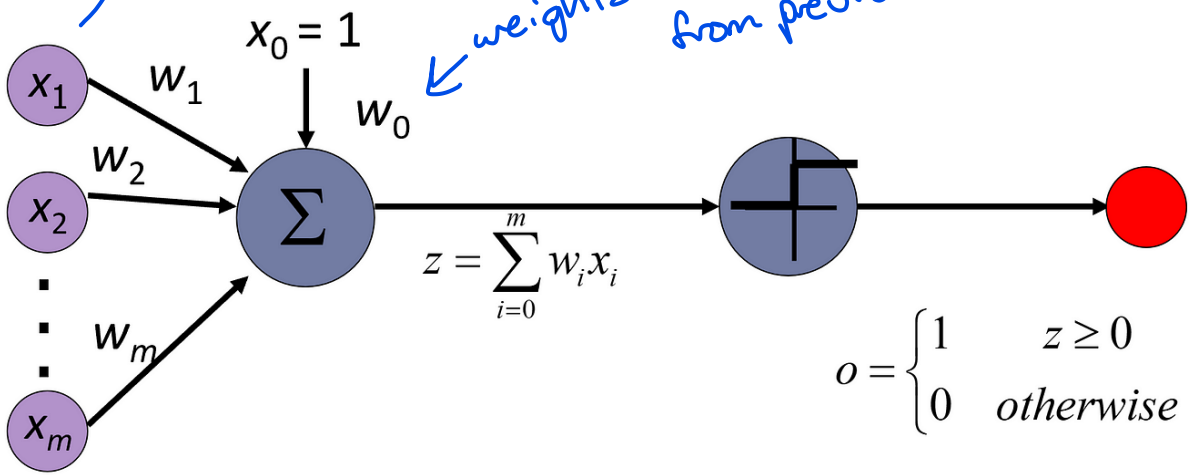
# Biological Neuron

- Prob =: This was only able to solve linear problems!
- Starts w/ random guassian dist. weights before it learns at all

Takes inputs $x_1 \ldots x_m$ and weights them (learned, how do we weight to solve?)

# Artificial Neuron

weights = knowledge from previous




$$z = \sum_{i=0}^{m} w_i x_i$$

$$o = \begin{cases} 1 & z \geq 0 \\ 0 & otherwise \end{cases}$$

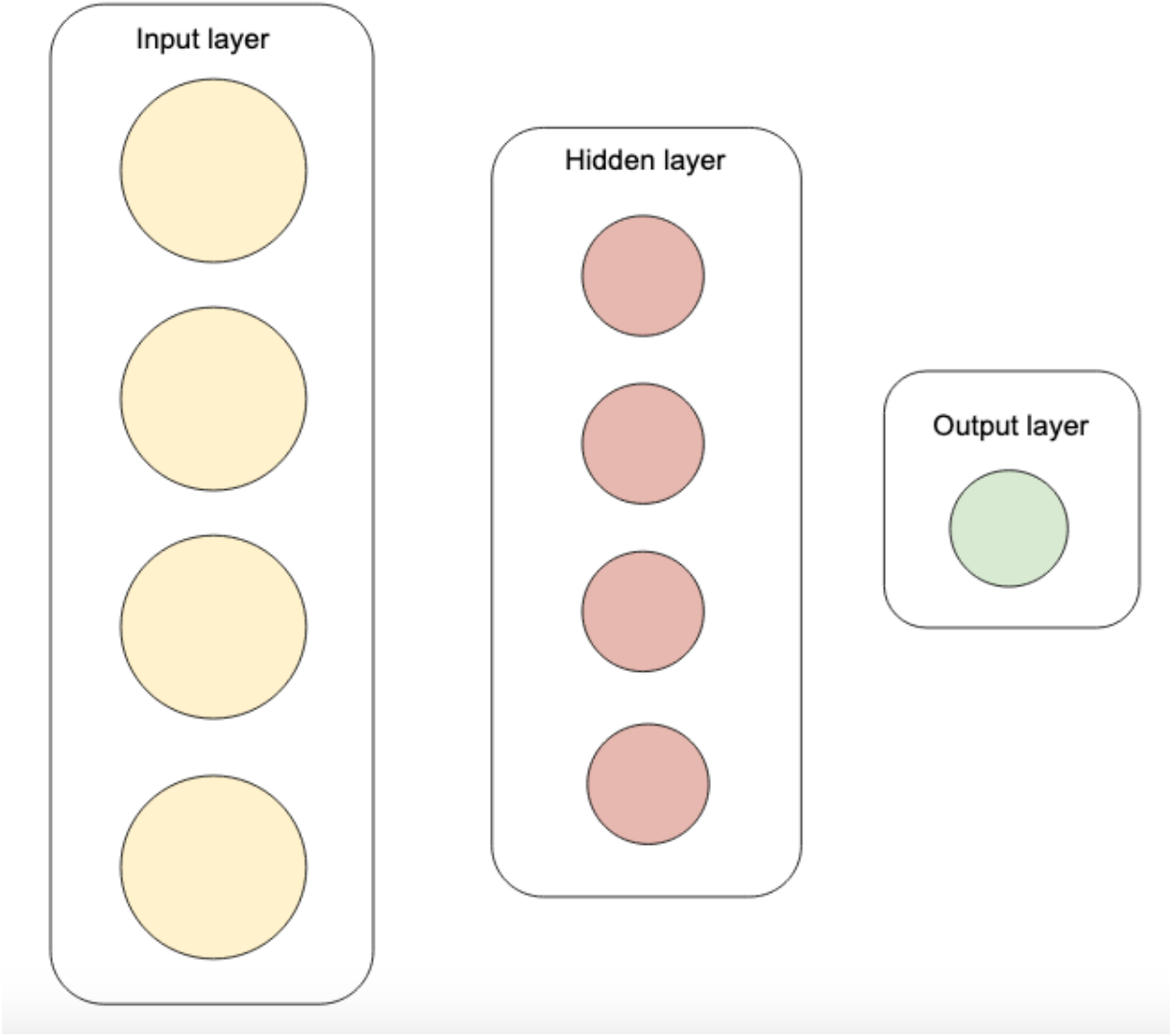
https://www.linkedin.com/pulse/neural-networks-from-biological-artificial-muhammed-nizam-pc
https://www.freepik.com/premium-vector/versus-vs-letters-fight-symbol-flat-comics-style-design-ation-symbol_18874747.htm

# Let’s play a Game!

Steps:

1) Feed Forwarding → from input to output
2) Back Propogation → from output back to input
3) How Does it learn?

- We have 9 players arranged in three layers
- There is an input layer, a hidden layer, and an output layer.

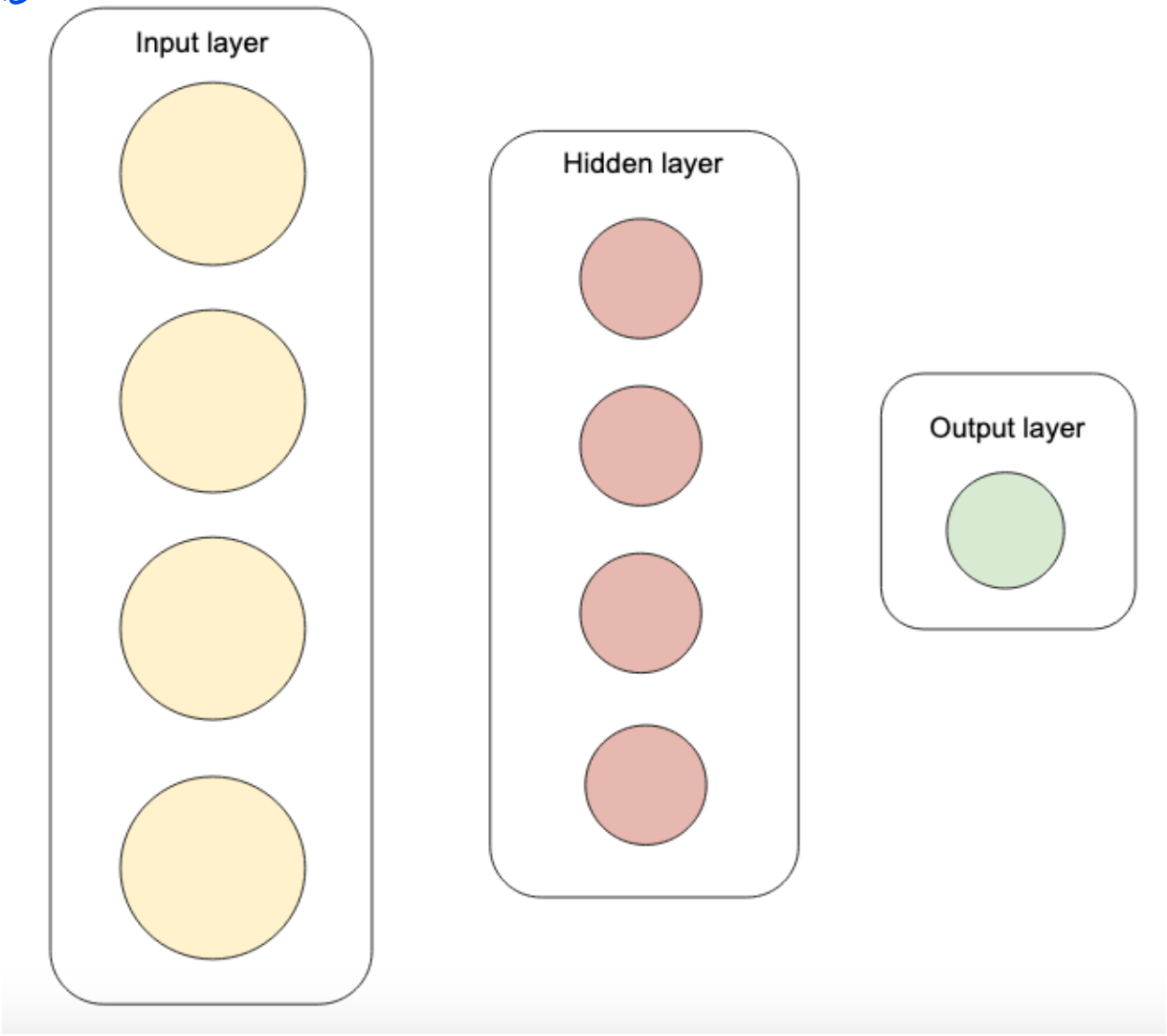





https://raise.mit.edu/daily/neuralnetworks/

# Step 1:

In general, trying to predict caption for many images. Goal is correctly predict captions

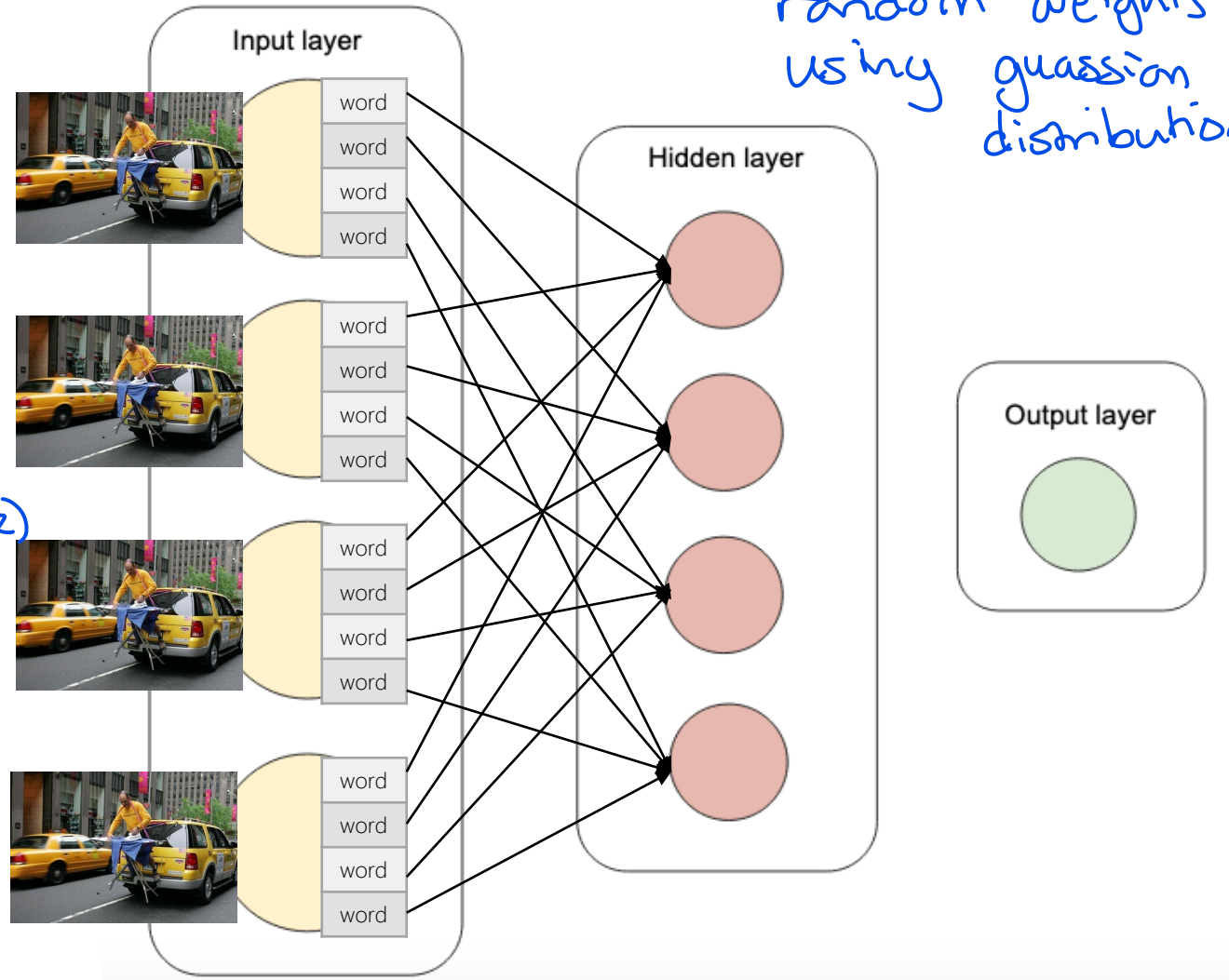



https://raise.mit.edu/daily/neuralnetworks/

# Step 1: Feed Forwarding – from input to output

(Starts w/ random weights using guassian distribution)

- players in the Input layer receives the image and write down 1 word on each slot (4 slots each) (captions) (predicted by current knowledge)
  can be > 4
- send 1 word to each of the 4 players in the hidden layer



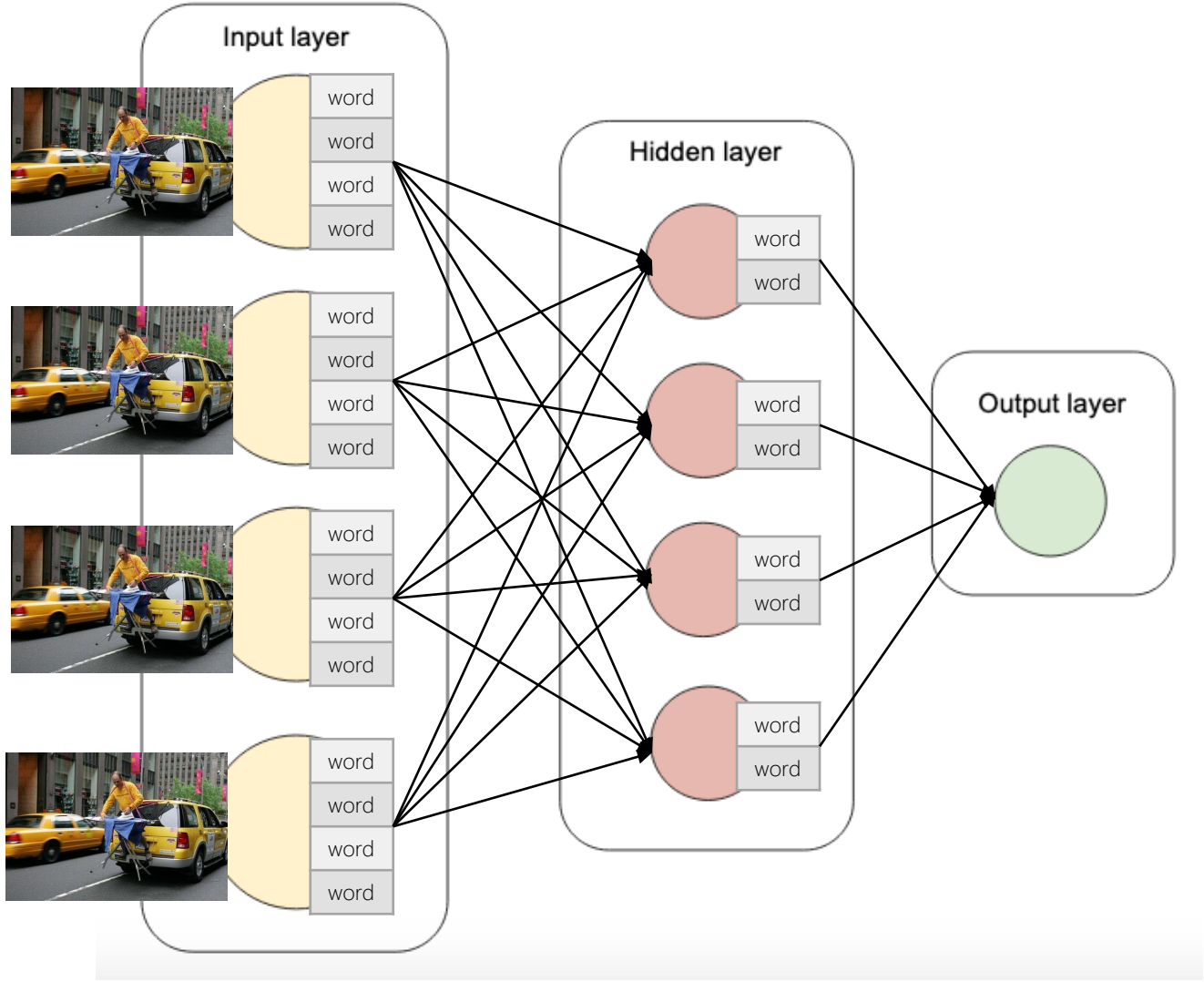



https://raise.mit.edu/daily/neuralnetworks/

# Step 2: Feed Forwarding

Players in the Hidden layer:

- receive 4 words each
- select 2 words to pass to the output node

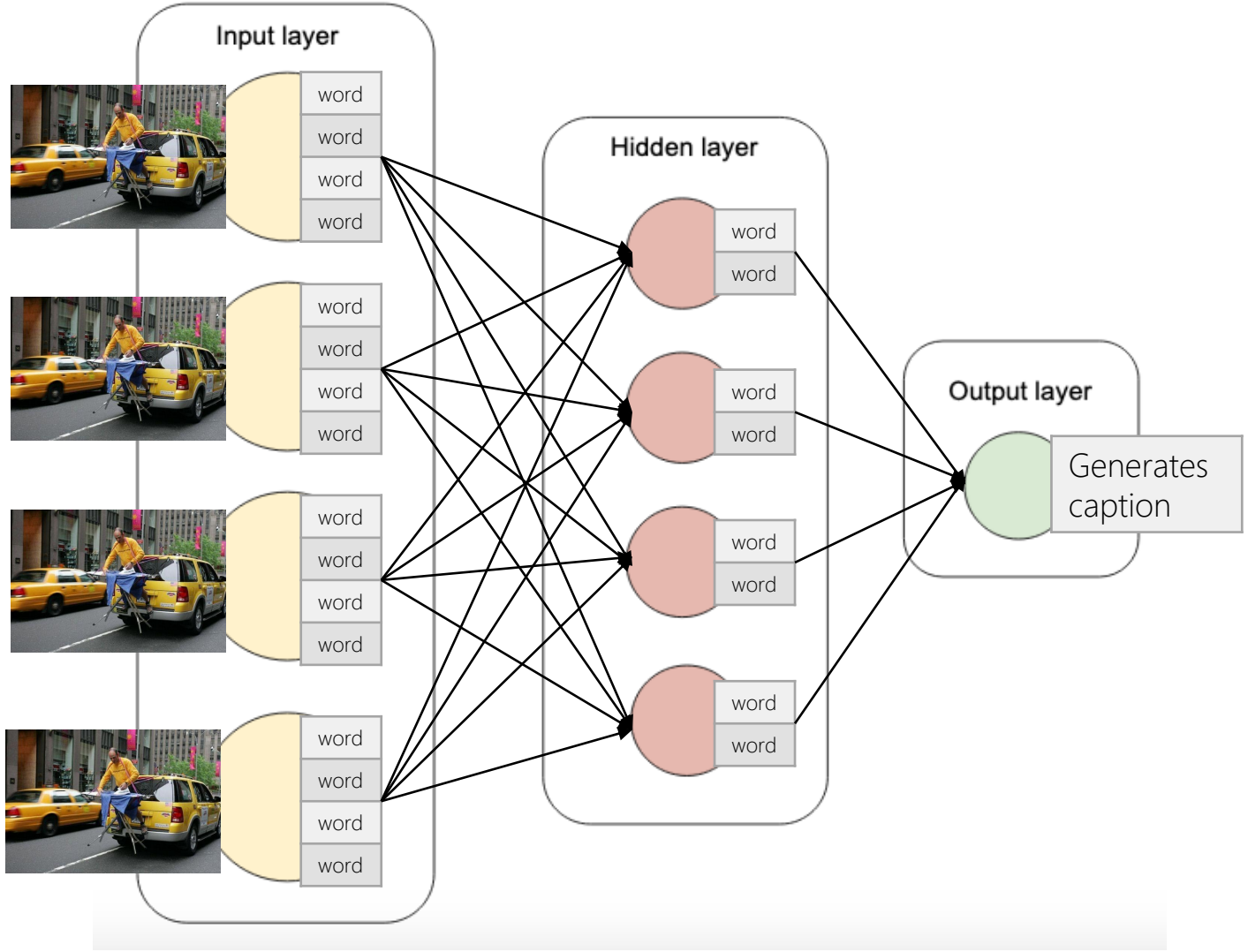



https://raise.mit.edu/daily/neuralnetworks/

# Step 3: Feed Forwading

The output node:

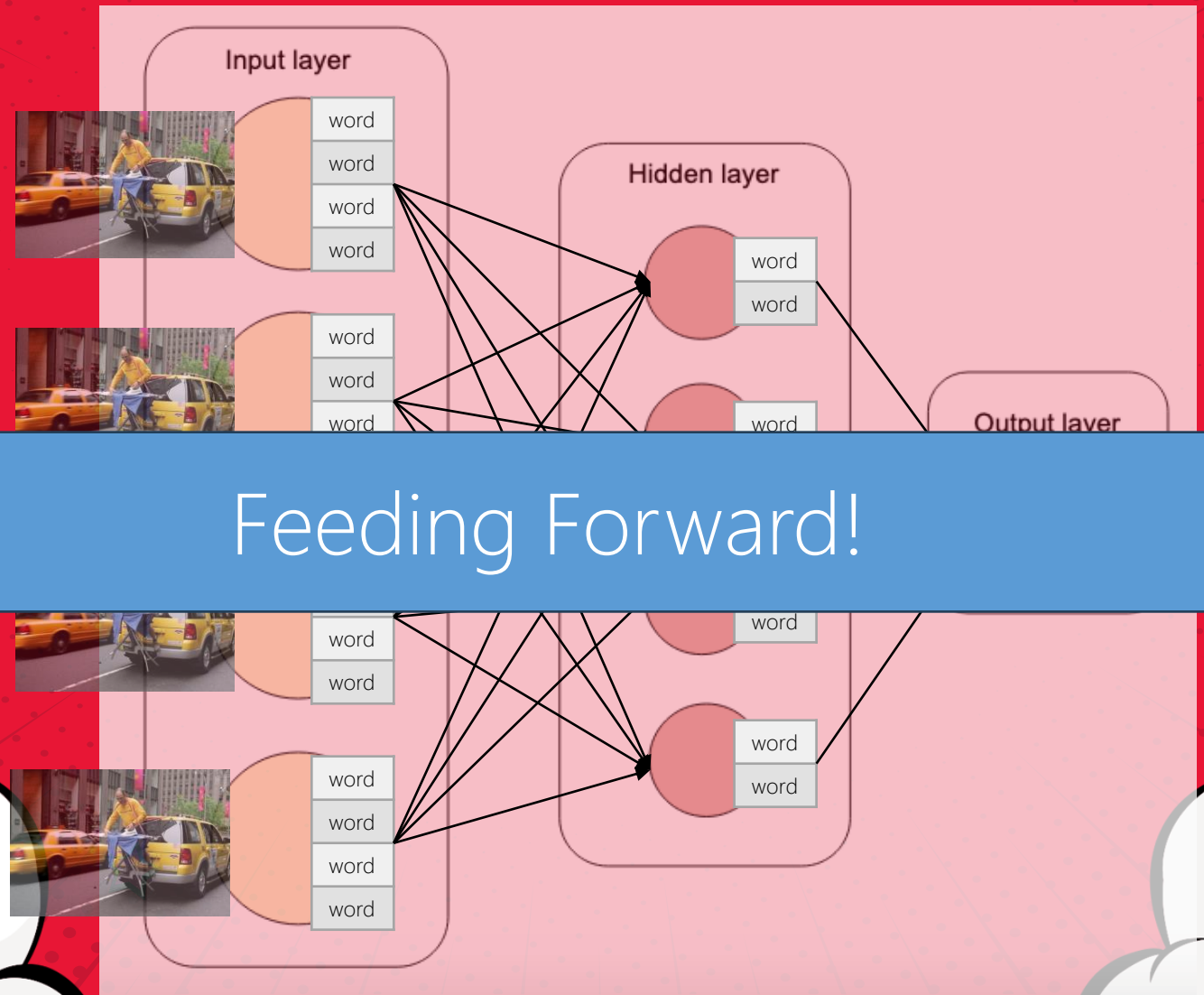
- receive 8 different words
- Picks 4 words to make a caption for an image with some filler words






https://raise.mit.edu/daily/neuralnetworks/

# You just learned what is Feedforward!!




Feeding Forward!

# Step 4: Evaluate

Original Caption

True to image

(Step 2)

Wrinkles? Not in NYC!

Sprinkles? Got in NYC!

Predicted Caption

# But how does the model learn??

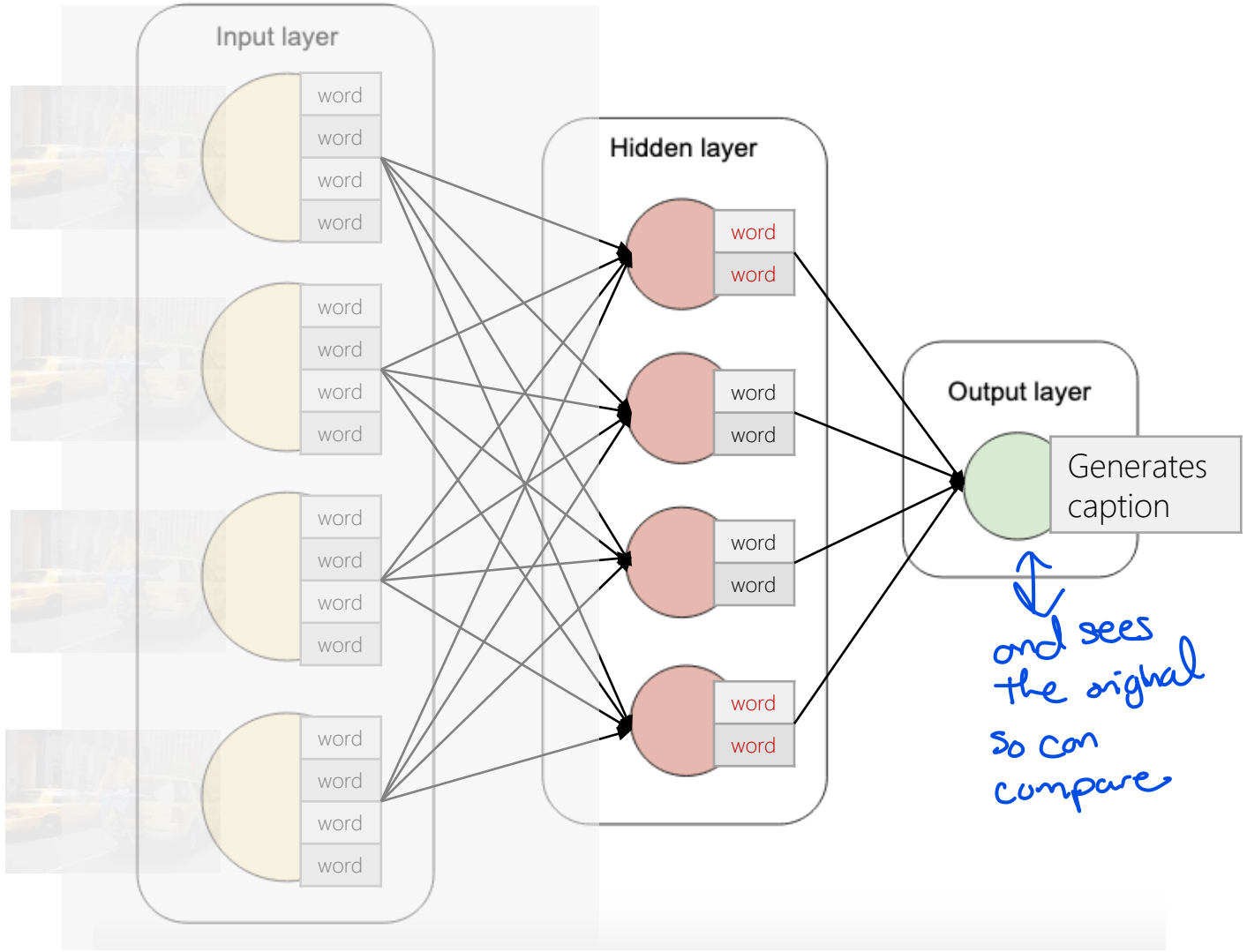
Back Propagation Stage

# Step 5:

Pass corrected info back so can learn from mistakes

The output node:

- Identifies the senders' words that were used in the original caption, i.e., the "correct" words
- Identifies which links gave "better" information




and sees the original so can compare



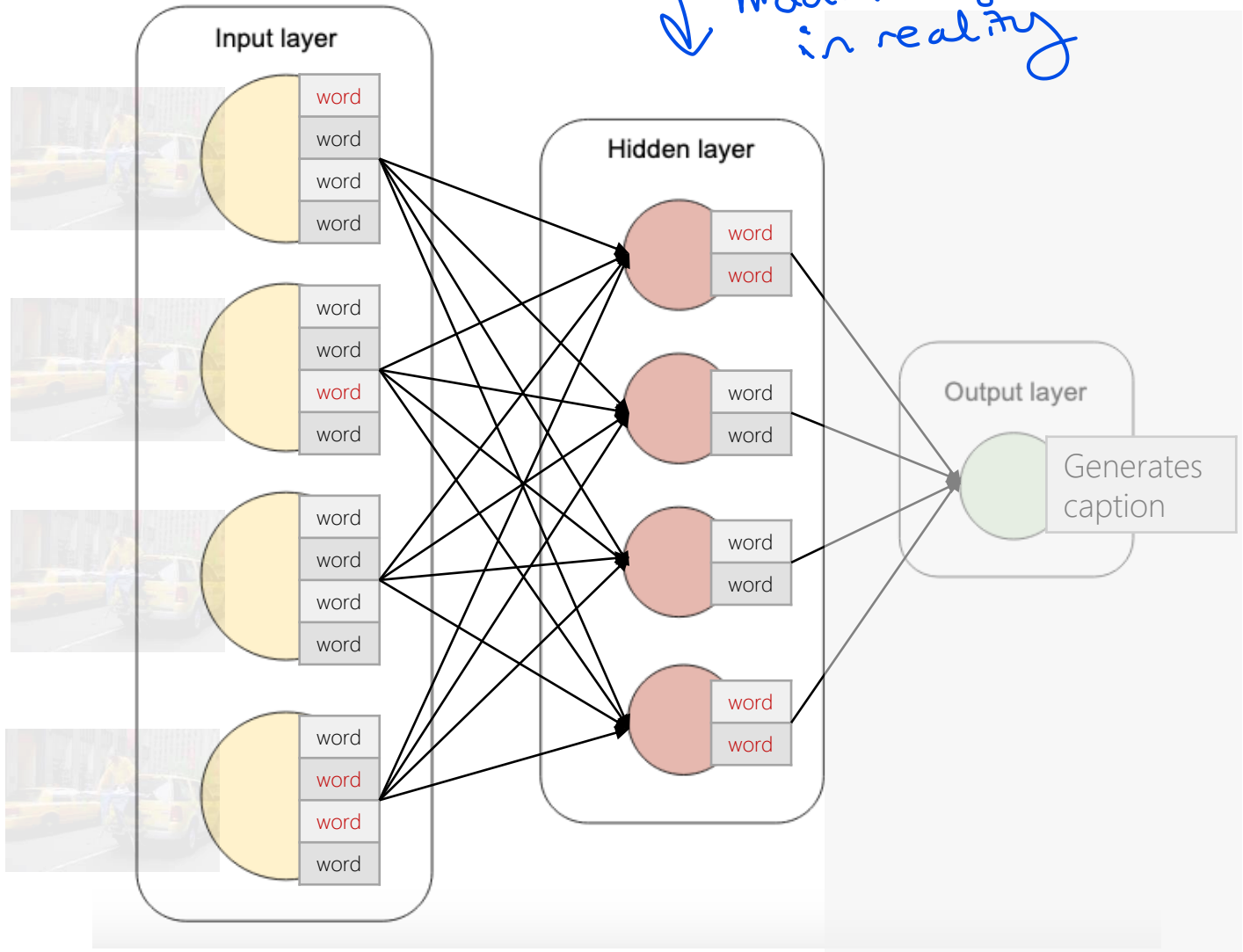
https://raise.mit.edu/daily/neuralnetworks/

# Step 6:

The hidden layer nodes:

- Circles the senders' words that were in the original caption. (these are "correct") (which tokens were right/wrong)
- Identifies which links gave "better" information

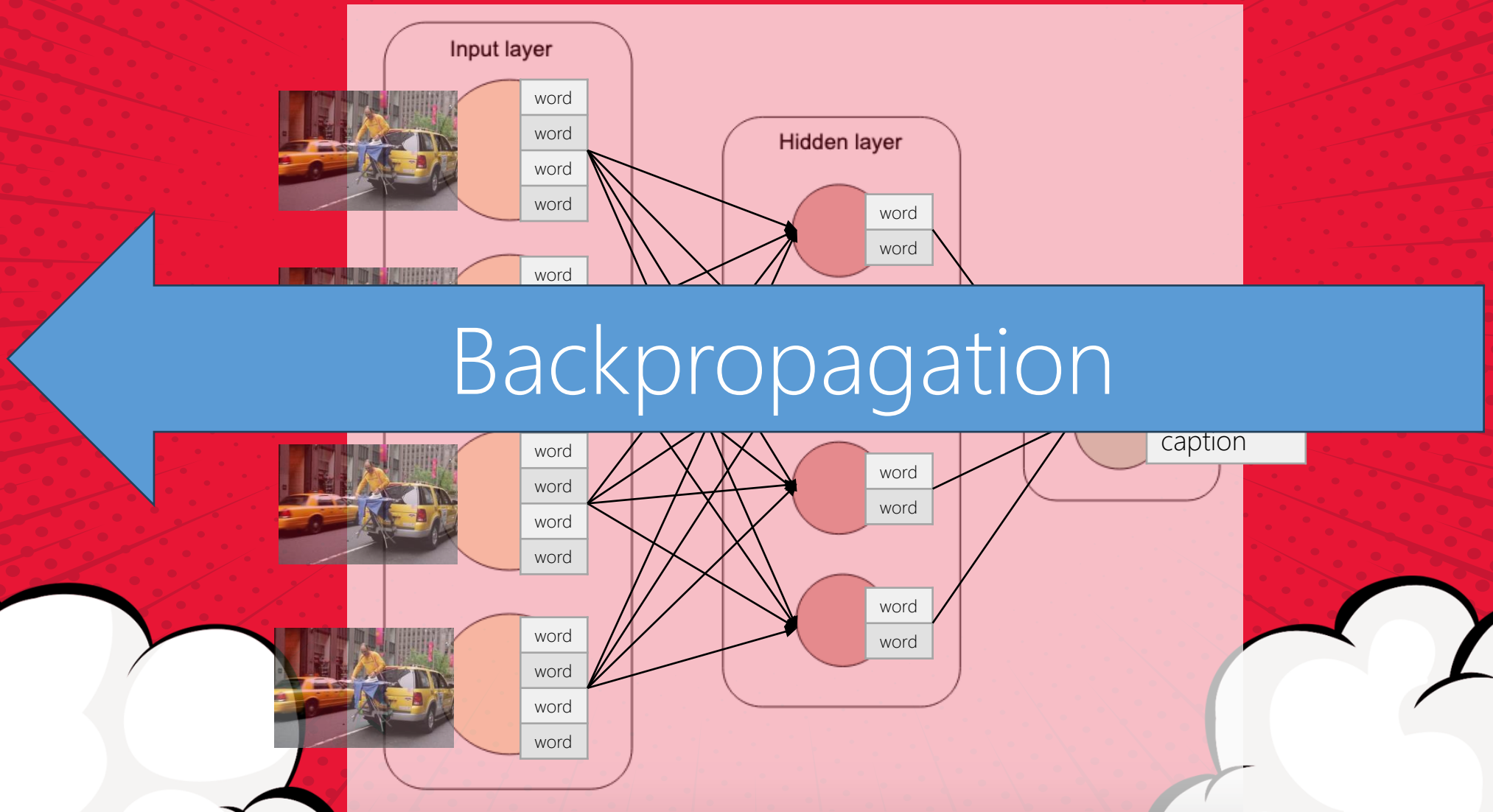
usually many hidden layers in reality






https://raise.mit.edu/daily/neuralnetworks/

# This is essentially Backpropagation!!

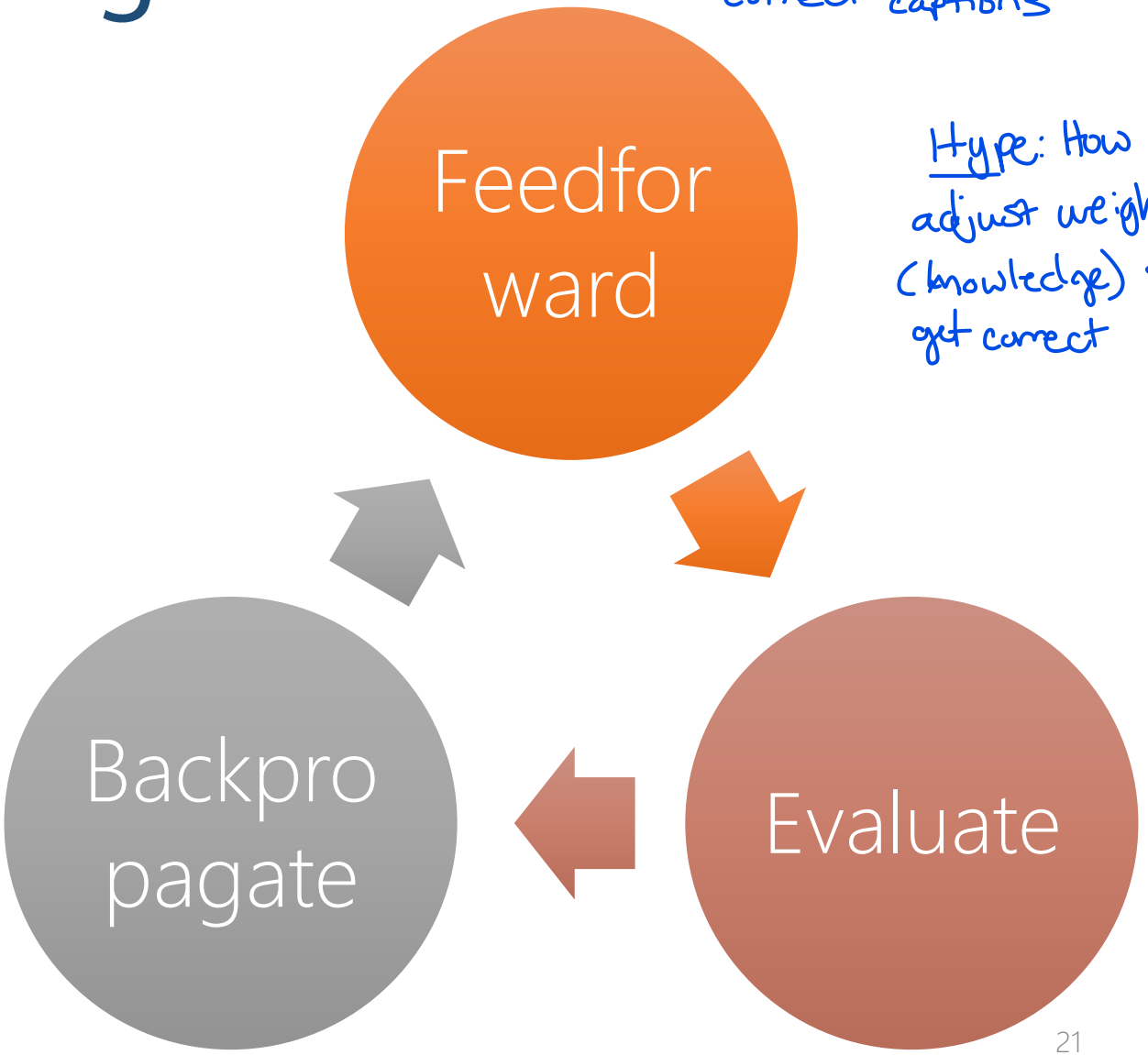


# Neural Network learning

Cycle Repeats!
- Epoch - weights align so generates correct captions

1. Each neuron finds out which words are right/wrong

2. adjusts its <u>weights</u> for how it picks words so the network can do better next time




Hype: How adjust weights (knowledge) so get correct



https://raise.mit.edu/daily/neuralnetworks/

# (Stochastic) Gradient Descent

"Direction"

- How update the weights

- We have a cost function $J(\theta)$ we want to minimize

Loss

→ Ex: Wrinkles? Not in NYC
Sprinkels Cat in NYC
⇒ 0 0 1 1 = loss = 2

- Idea: for current value of $\theta$, calculate $\nabla_\theta J(\theta)$, then take small step in the direction of negative gradient. Repeat.

θ = parameters of neurons = weights

$$\theta_{new}^{n+1} = \theta_{old}^{n} - \alpha \nabla_\theta J(\theta)$$

Gradient (direction)

loss function

learning rate (stepsize)

↑ LR = larger step
↓ LR = smaller step

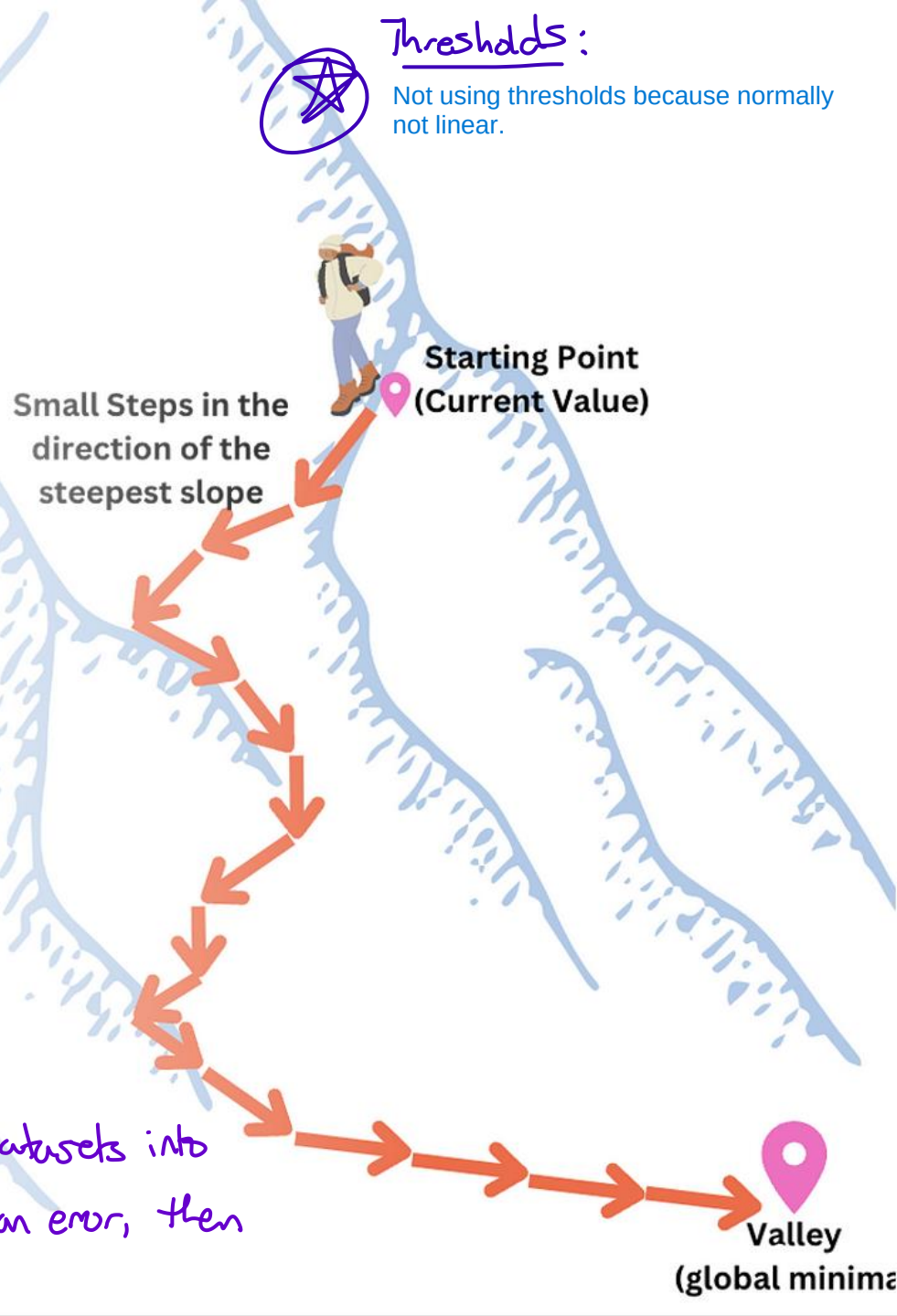
- Why Stochastic?

• Stochastic: randomly pick samples to do the process

• Mini batch - Divide datasets into batches, pass, calc mean error, then pass back that error

Thresholds:

Not using thresholds because normally not linear.




https://pub.towardsai.net/gradient-descent-and-the-melody-of-optimization-algorithms-244830ea2516

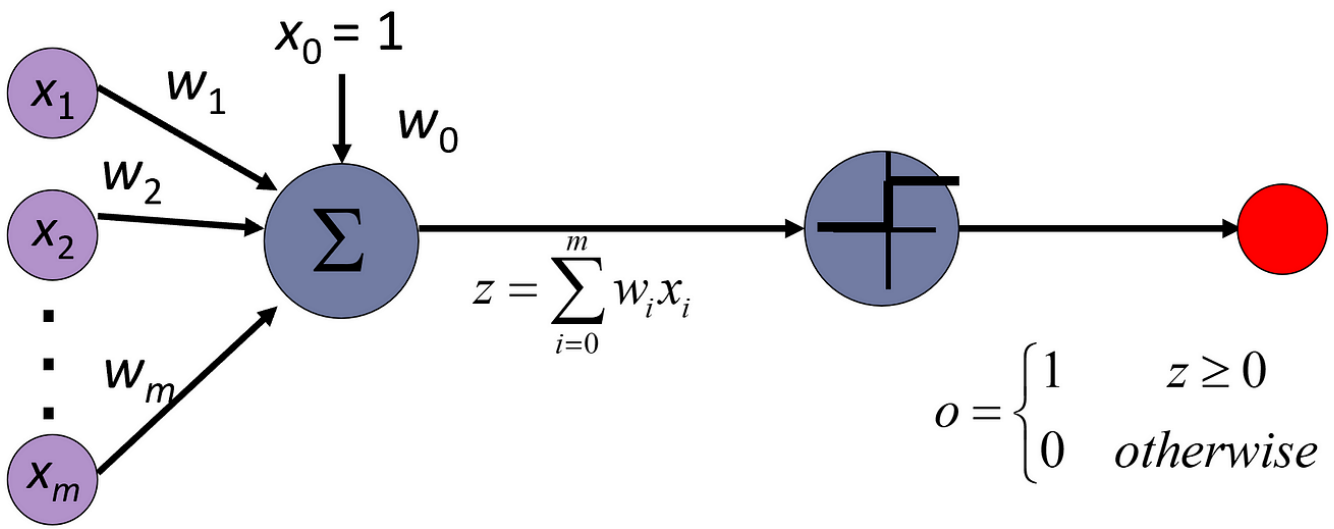
# Why Activation functions?

(Instead of passing using threshold)

↓ Non linear transformation

- Without them, neural networks will only do simple linear transformation

- Different forms:
  - Sigmoid
  - Tanh
  - ReLU
  - SoftPlus
  - Leaky ReLU
  - …




http://playground.tensorflow.org/ ← watch this!



https://www.linkedin.com/pulse/neural-networks-from-biological-artificial-muhammed-nizam-pcxwf/

Diff Types of Activations

## Sigmoid

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

## Tanh

$$y = \tanh(x)$$

## Step Function

$$y = \begin{cases} 0, & x<n \\ 1, & x \geq n \end{cases}$$

## Softplus

$$y = \ln(1+e^{x})$$

## ReLU

$$y = \begin{cases} 0, & x<0 \\ x, & x \geq 0 \end{cases}$$

- not pass
- pass val

pass 0 if $z < 0$ in prev. ex.

## Softsign

$$y = \frac{x}{(1+|x|)}$$

## ELU

$$y = \begin{cases} \alpha(e^{x}-1), & x<0 \\ x, & x \geq 0 \end{cases}$$

## Log of Sigmoid

$$y = \ln\left(\frac{1}{1+e^{-x}}\right)$$

## Swish

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

## Sinc

$$y = \frac{\sin(x)}{x}$$

## Leaky ReLU

$$y = \max(0.1x, x)$$

- Instead of 0
- or pass val

## Mish

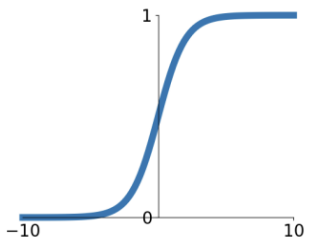
$$y = x(\tanh(\text{softplus}(x)))$$

https://medium.com/@shubhayumajumdar/activation-functions-in-neural-networks-ef7d24531e48

# Why Activation functions?

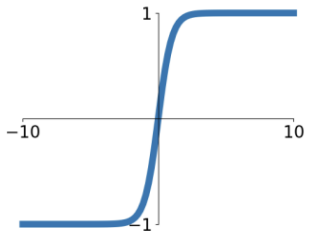
**Sigmoid**

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



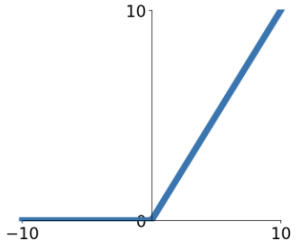

**tanh**

$$\tanh(x)$$



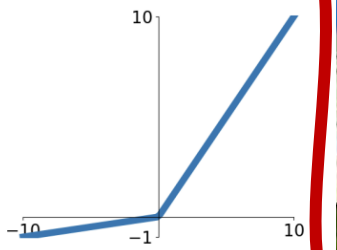

**ReLU**

$$\max(0, x)$$




**Leaky ReLU**

$$\max(0.1x, x)$$

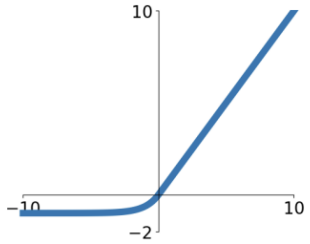


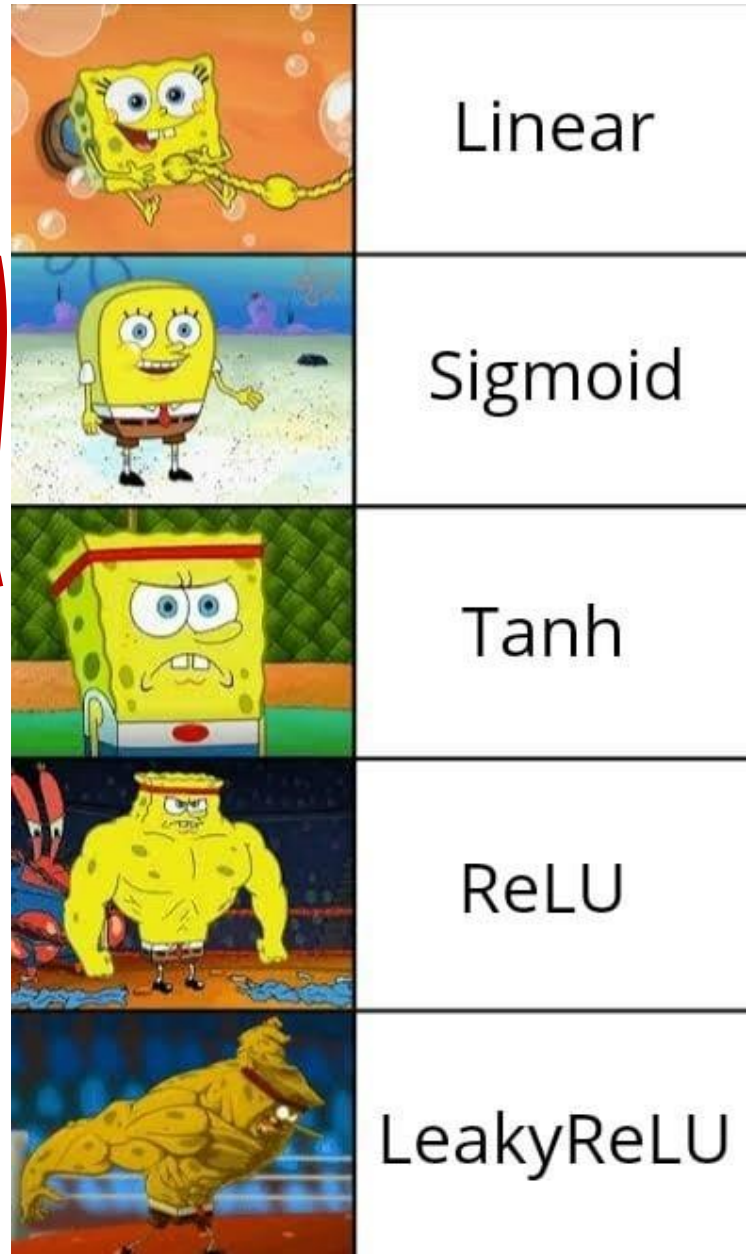

**Maxout**

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

**ELU**

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$






https://web.eecs.umich.edu/~justincj/slides/eecs498/498_FA2019_lecture05.pdf

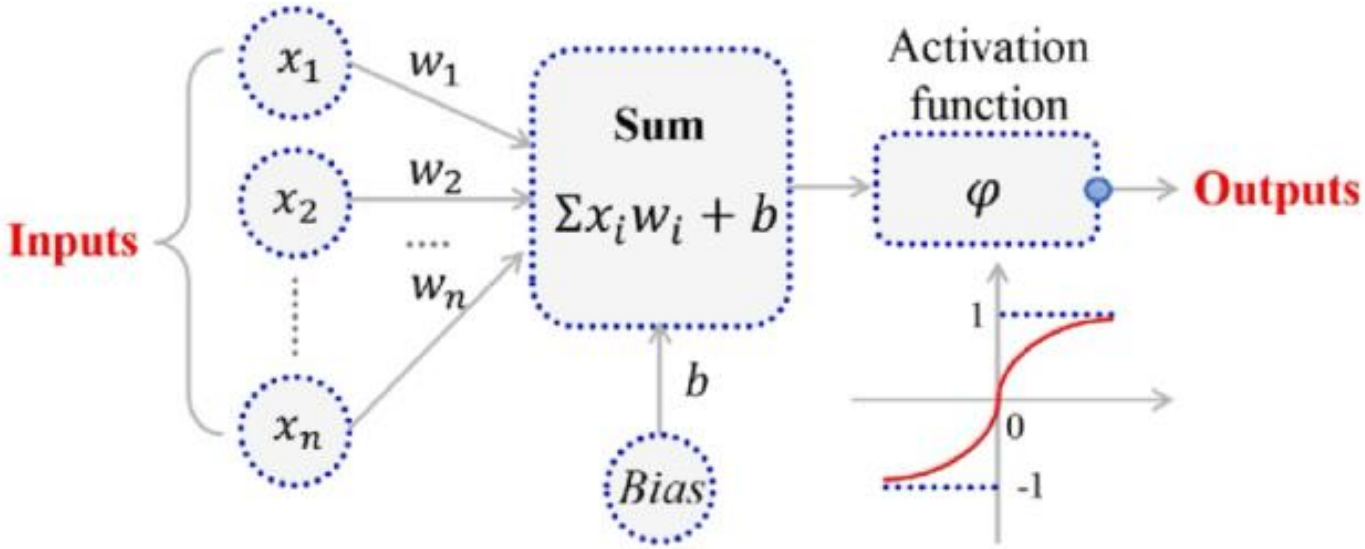
# Another perspective...

- Isn't this similar to logistic regression?

1) Takes input
2) Applies weights
3) Predicts output

Difference?

What logistic reg. Does:

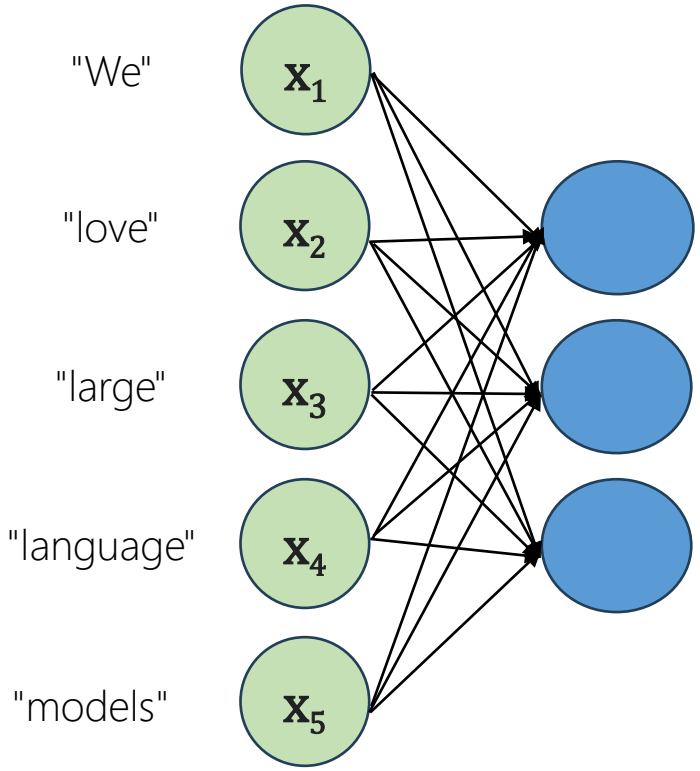





https://www.linkedin.com/pulse/neural-networks-from-biological-artificial-muhammed-nizam-pcxwf/

# Stack different logistic regression ---> You get a Neural Network (voila!)

- Isn't this similar to logistic regression?

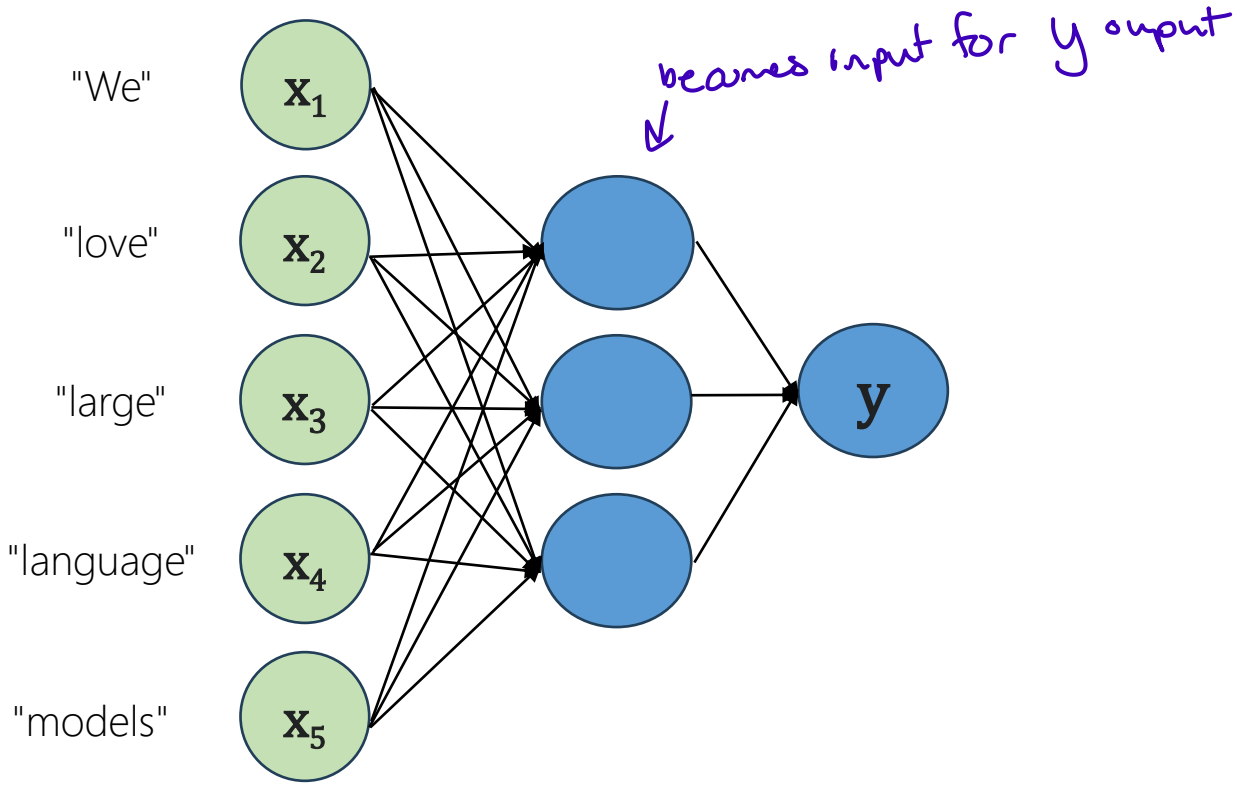





https://www.linkedin.com/pulse/neural-networks-from-biological-artificial-muhammed-nizam-pcxwf/

# Stack different logistic regression ---> You get a Neural Network (voila!)

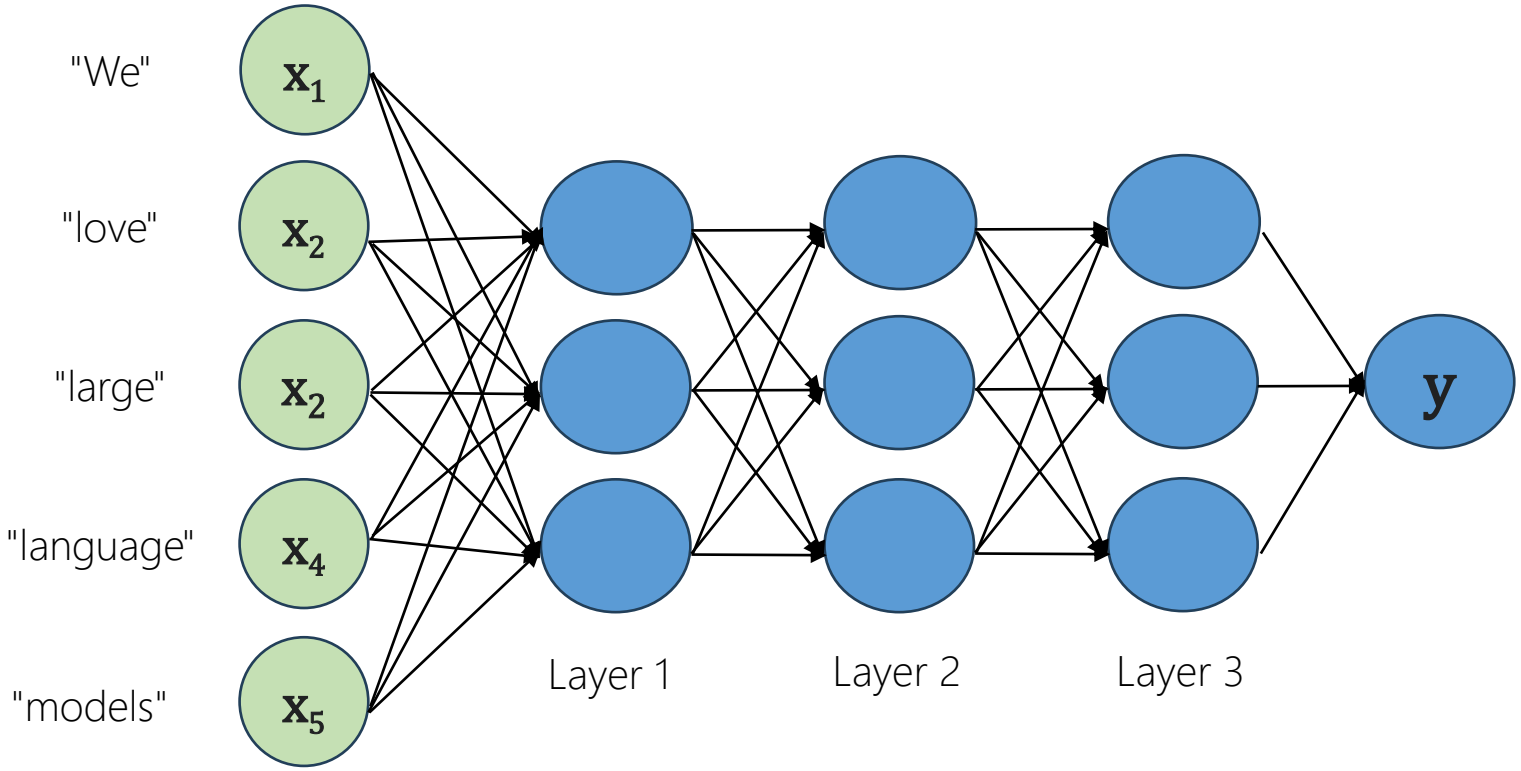
- Isn't this similar to logistic regression?






https://www.linkedin.com/pulse/neural-networks-from-biological-artificial-muhammed-nizam-pcxwf/

# Scale…




Learning a highly non-linear representation of the data and then using it to classify

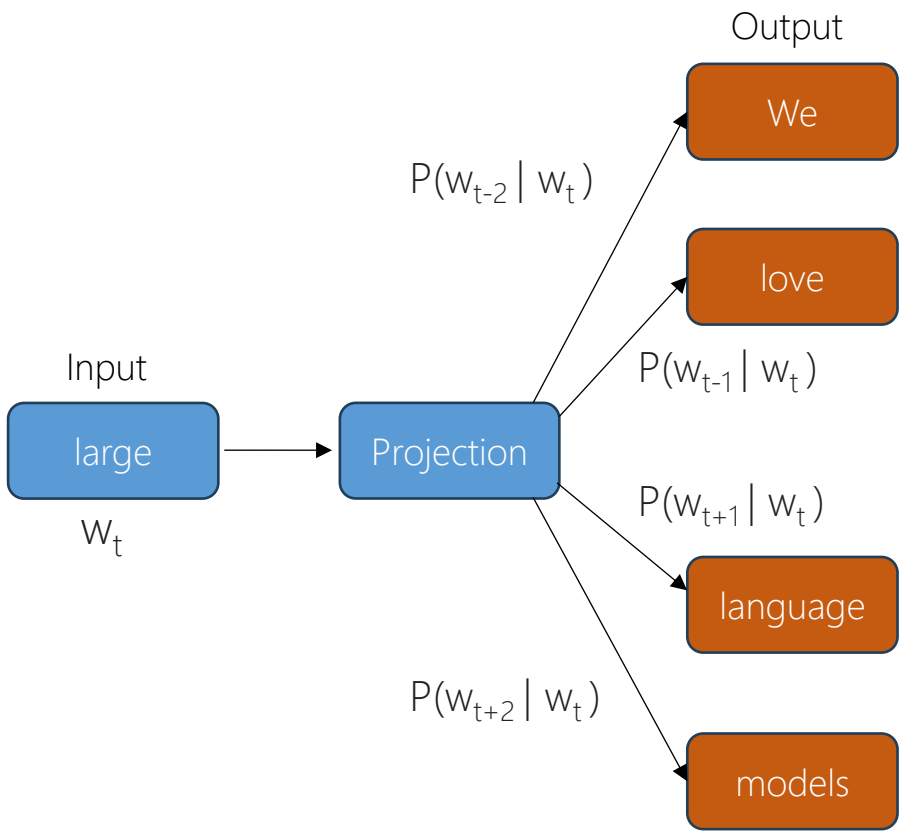


https://www.linkedin.com/pulse/neural-networks-from-biological-artificial-muhammed-nizam-pcxwf/

# Word2Vec

- Start with random word vectors
- Iterate through each word position in the whole corpus
- Try to predict surrounding words using word vectors $P(w_{t+j} \mid w_t)$

- **Learning:** Update vectors so they can predict actual surrounding words better
- Repeating this learns word vectors that capture word similarity and meaningful directions in a word space



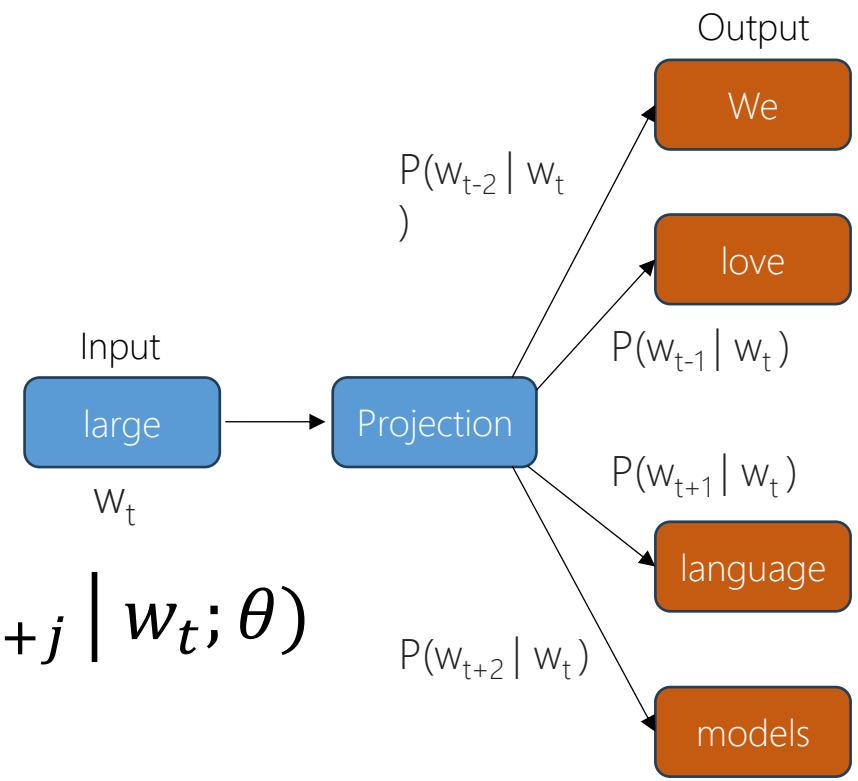
# Skip-gram: Objective function

For each position t = 1, … , T, predict context words within a window of fixed size $W$, given center word $w_t$

$$L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-W \le j \le W \\ j \ne 0}} P\left(w_{t+j} \mid w_t; \theta\right)$$

Average negative log likelihood:

$$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-W \le j \le W \\ j \ne 0}} \log P\left(w_{t+j} \mid w_t; \theta\right)$$



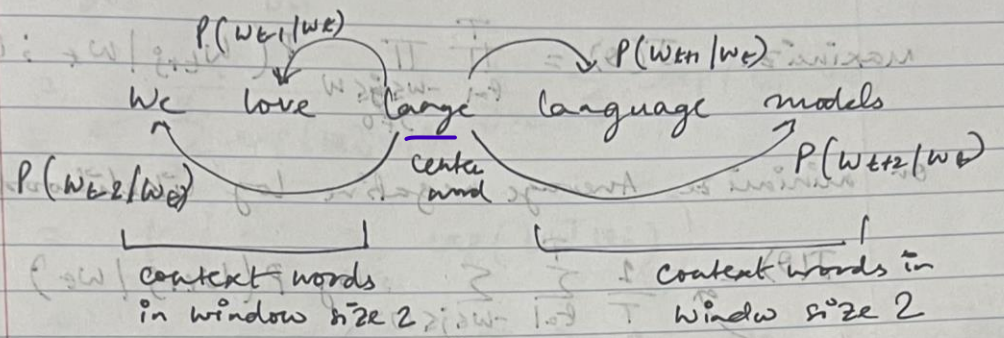
(*) Slide the window so get context for every word

# Skip-gram

Try use center word to predict prob of others

- Soft max function - how predicts prob

Lecture 1: Intro to NLP!




Skip-gram

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-w\le j\le w \\ j\ne 0}} \log P(w_{t+j}|w_t;\theta)$$

Q: How to calculate $P(w_{t+j}|w_t;\theta)$

A: we will use two vectors per word $w$

↳ $v_w$ when 'w' is a center word

↳ $u_w$ when 'w' is a context word

Softmax

There fore, for a center word 'c' & context word 'o'

output center

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w\in V}\exp(u_w^T v_c)}$$

dot product compare similarity of o & c

$u^T v = \sum_{i=1}^{n} u_i v_i$

- across all words

normalize over entire vocab to give prob distribution

Larger dot product = larger probability

very important in deep learning

Normalized btw 0 to 1

- $u_o$ = d dim vect. of output
- $v_c$ = rep of center word

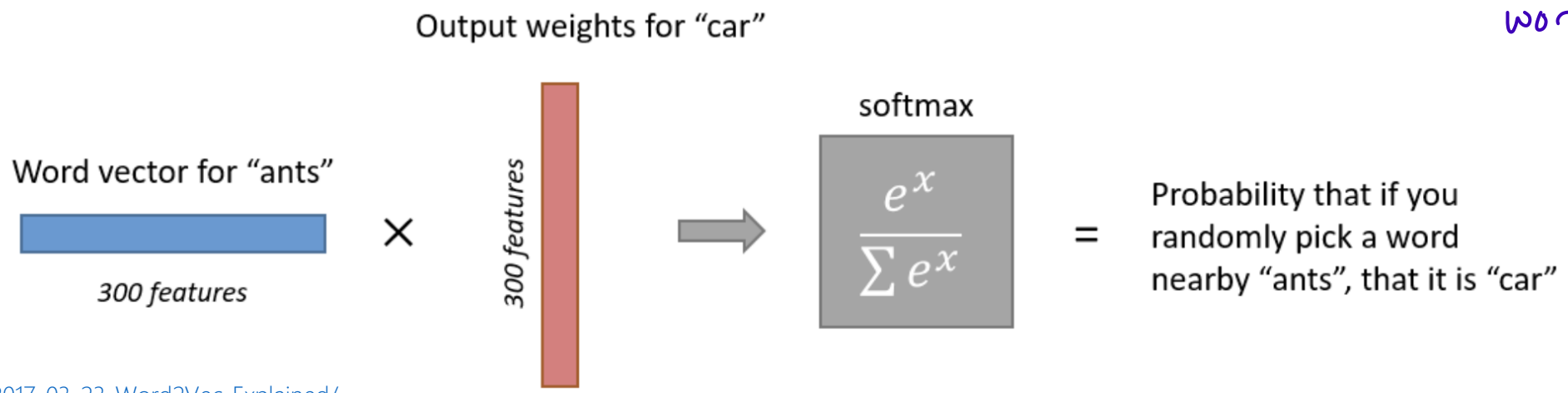
Softmax function maps arbitrary values $x_i$ to a distribution $p_i$

max → amplifies probability of largest $x_i$

soft → assigns some probability to smaller $x_i$

# Training Algorithm for Word2Vec

- SoftMax
- Hierarchical SoftMax - Not work
- Negative sampling - Do this ( Handpicks some negative words - opposites
  Speeds up process
  ↓
  based on center word )




https://israelg99.github.io/2017-03-23-Word2Vec-Explained/

# Why Negative Sampling?

- The denominator of the SoftMax is computationally expensive (when many output classes)

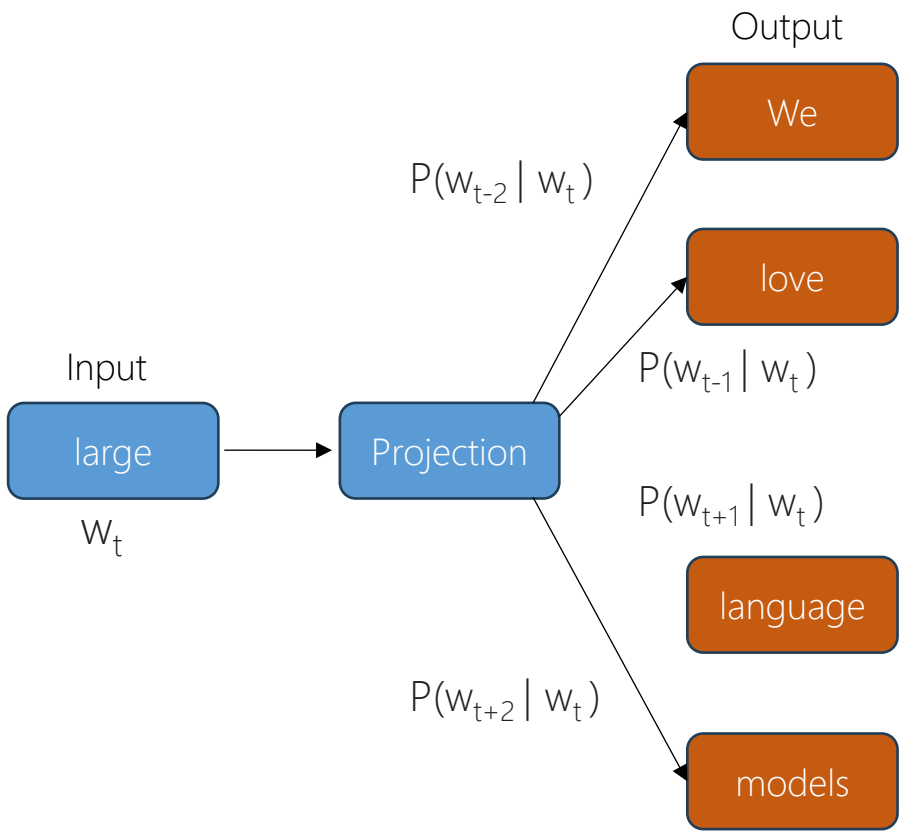
- In practical settings, we implement the skip-gram model with negative sampling



https://israelg99.github.io/2017-03-23-Word2Vec-Explained/

# Word2Vec

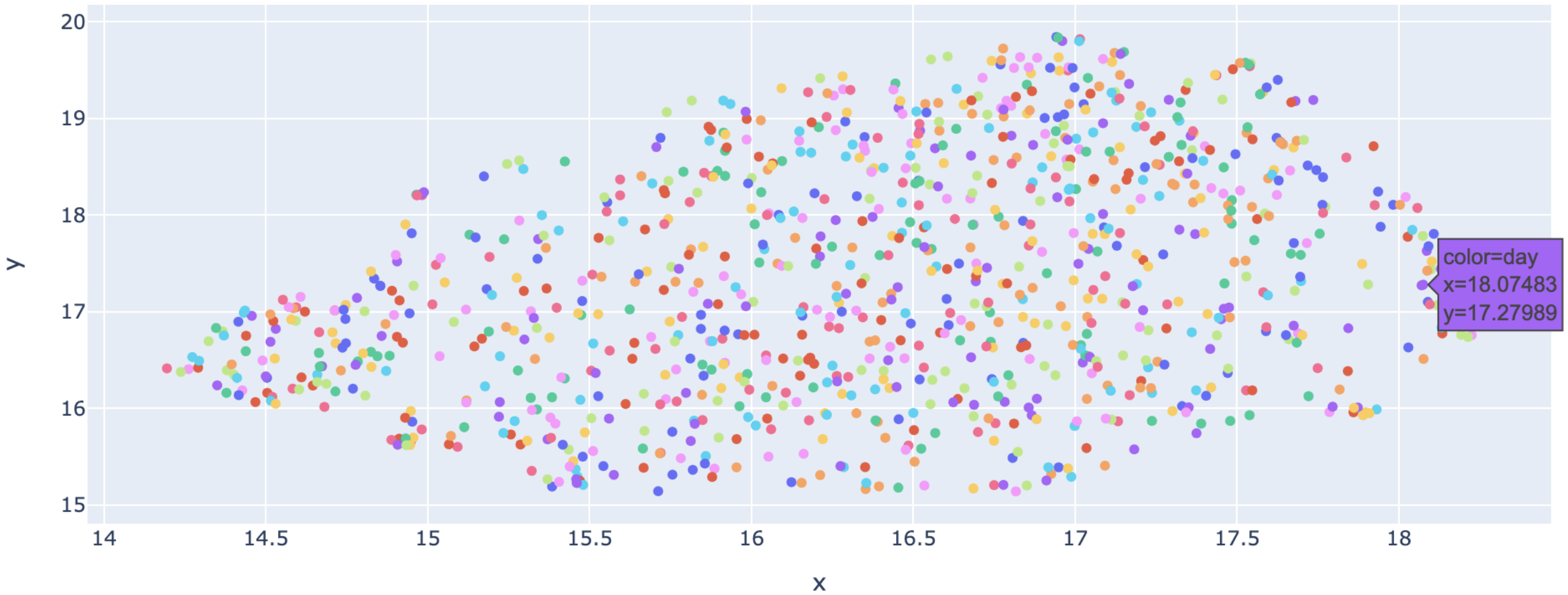
Let's code it..






https://israelg99.github.io/2017-03-23-Word2Vec-Explained/

# Semantic Patterns using Word2Vec



# Other NLP Features

- Term Frequency-Inverse Document Frequency (TF-IDF)
- Word Embeddings (e.g., Word2Vec and Glove)
- Bag of words/ngrams
- Latent Dirichlet Allocation (LDA)
- Part-of-Speech (POS)

# Let's train a sentiment classifier!

# Next Lecture..

- What are Tokenizers?
- Basics of Tokenization