# Deep Learning, Part 1(ish)

Kenneth (Kenny) Joseph

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

# Announcements

- PA3 grades out early next week
- Quiz 10 out, due Tuesday night
- PA5 out tomorrow (due date adjusted accordingly)
- PA 4 – more fun!

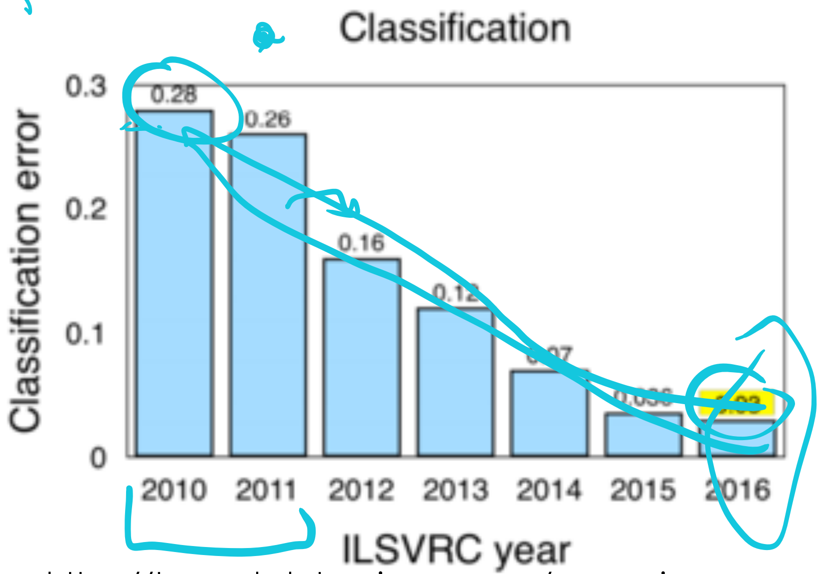
# Quiz 9 Review

# Notes: Rest of the Semester

- Lectures
  - 3ish Deep Learning
  - 2ish Bias and Justice
  - 1ish Choose your Own Adventure
  - 1 Review
- Deliverables
  - 3 Quizzes (10, 11, 12)
  - 1 PA
- Final ~~**(Weds, [illegible] 18th 7:15-10:15PM, Hoch 114)**~~
  - No note sheet
  - Must show your work
  - Randomized seating
  - Exam will be same length, similar format as midterm
  - Exam topics will be released within the next 2 weeks

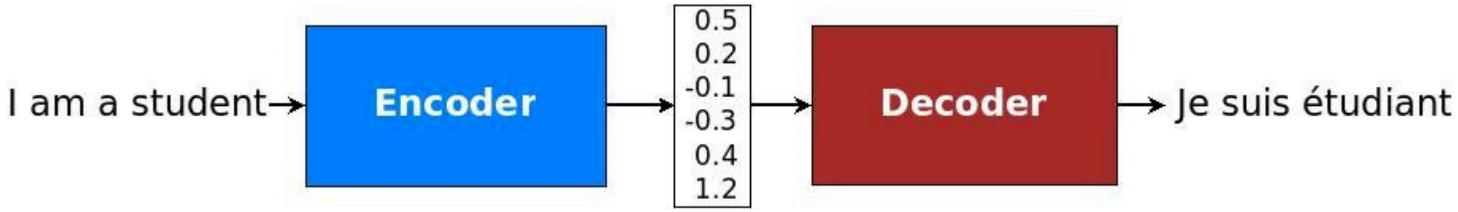
When I say "deep learning," what does that mean to you/what comes to mind?

# Massive performance gains

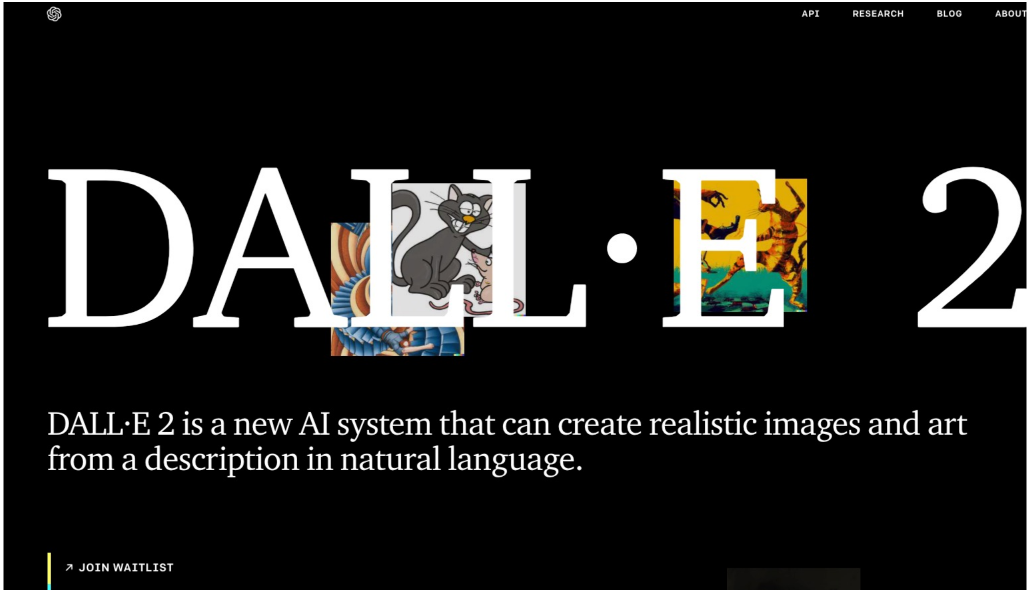



https://towardsdatascience.com/a-year-in-computer-vision-part-1-of-4-eaeb040b6f46

# Potentially Transformative applications



# Fun, creative new uses of ML




https://openai.com/dall-e-2/

GENERATE ANOTHER

## Completion

**Today in CSE 199 we are going to learn about** two very interesting methods of generating a random number in C#: the Random type and the RandomFunc type:

Random type

The Random type is used to generate a random number in an expression context using a System.Byte[] array. We can pass in the Random value directly as a parameter to the random_expression method of an instance of RandomFunc. It supports the following modes:

mode_int32: Generate the same value from the array starting at position i on the array at position j.

Kenny's favorite hobby is to

GENERATE ANOTHER

## Completion

**Kenny's favorite hobby is to** play basketball. This summer, despite his dac absence, he's going to the University of Central Florida for three years and set to take the basketball field this summer. It's a dream come true and a dream he's not going to forget.

Kenny is currently at the family's home in Jacksonville, Florida with his grandmother and two other n

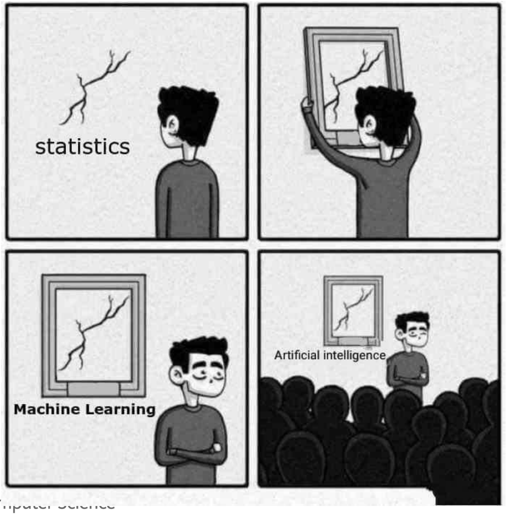
https://talktotransformer.com/

# A massive hype machine

**Ilya Sutskever**
@ilyasut

it may be that today's large neural networks are slightly conscious

6:27 PM · Feb 9





An actual, real, no joke ad I received while making these slides

# Some reasons for skepticism

```
Q: How do you sporgle a morgle?
A: You sporgle a morgle by using a sporgle.

Q: How many bonks are in a quoit?
A: There are three bonks in a quoit.

Q: How many rainbows does it take to jump from Hawaii to seventeen?
A: It takes two rainbows to jump from Hawaii to seventeen.

Q: Which colorless green ideas sleep furiously?
A: Ideas that are colorless, green, and sleep furiously are the ideas of a sleep
furiously.

Q: Do you understand these questions?
A: I understand these questions.
```

https://lacker.io/ai/2020/07/06/giving-gpt-3-a-turing-test.html

# Centralization of ML Power/Research

# Why Now?

| | |
|---|---|
| 1952 | Stochastic Gradient Descent |
| 1958 | Perceptron<br>• Learnable Weights |
| ⋮ | |
| 1986 | Backpropagation<br>• Multi-Layer Perceptron |
| 1995 | Deep Convolutional NN<br>• Digit Recognition |
| ⋮ | |

Neural Networks date back decades, so why the resurgence?

## 1. Big Data

- Larger Datasets
- Easier Collection & Storage

IMAGENET

WIKIPEDIA
The Free Encyclopedia

## 2. Hardware

- Graphics Processing Units (GPUs)
- Massively Parallelizable

## 3. Software

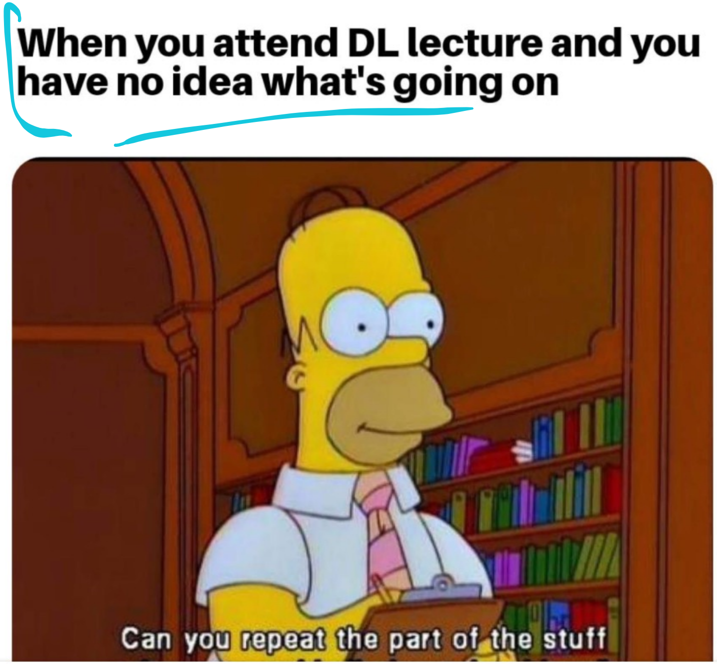
- Improved Techniques
- New Models
- Toolboxes

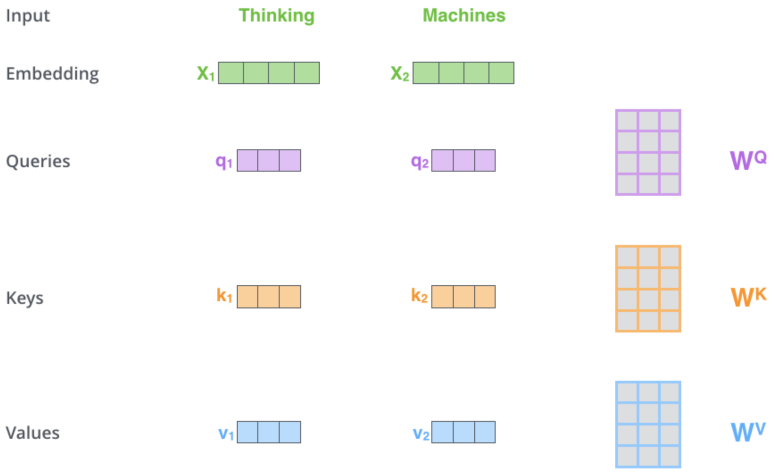
TensorFlow

# What I am aiming for

- I would like if by the end of this you…
  - Appreciate the incredible power of these models
  - Acknowledge their limitations and challenges
  - Understand the basics of these models, and that they are really just very complex matrix algebra
  - Understand how to code to write up your own model, using keras/tensorflow

# What I am *not* aiming for (I know, shocker)



# What I am aiming for (con't)

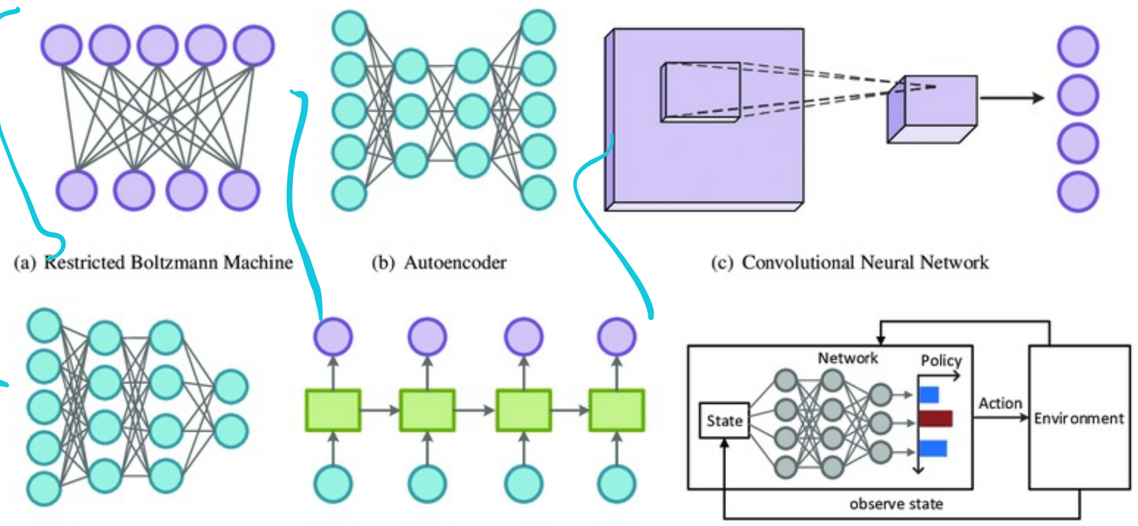
- Today:
  - Have a discussion about what DL is/isn't
  - Link back to our standard approach to ML
  - Understand building blocks of models
  - Understand how predictions are made in MLPs
  - See and run end-to-end code example in keras/tensorflow
- Next Tuesday
  - Understand convolutional neural nets and their implications
  - Start backprop
- Next Thursday
  - Finish backprop
  - See some applications
  - Transition to Bias/Equity

# Revisiting our standard ML view

In Machine Learning, we ask the question of "how would you figure this out" via a **three step process**:

1. We define a **model class** (or synonymously, a **hypotheses class**). We will pick one of those to be our predictive model.
2. We define a **loss function**. This defines the "best" model, i.e. the one we're going to pick.
3. We define an **optimization algorithm**. This is *how we actually find the best model*.

# Model representations of DL

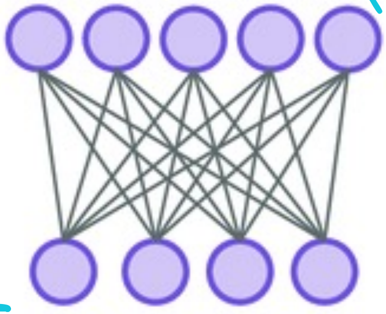



(a) Restricted Boltzmann Machine (b) Autoencoder (c) Convolutional Neural Network

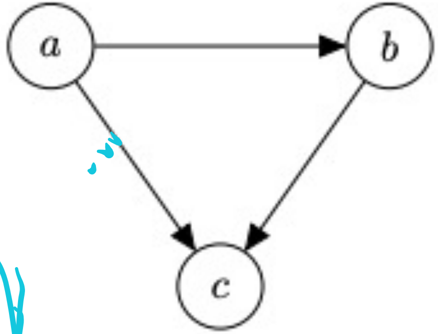
(d) Deep Neural Network (e) Recurrent Neural Network (f) Deep Reinforcement Learning

https://www.researchgate.net/figure/The-structures-of-different-deep-learning-models_fig2_340123883

# Model representations of DL



Vs.



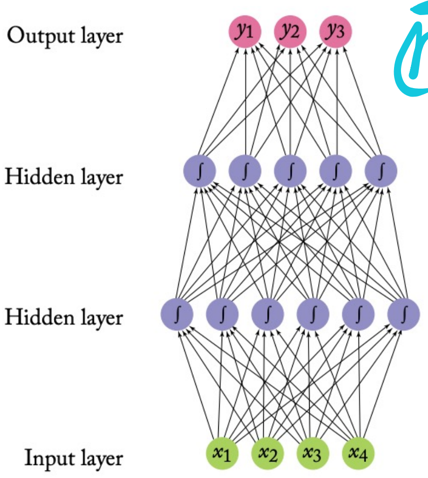
(a) Fully connected.

Not necessarily probabilistic!

≠

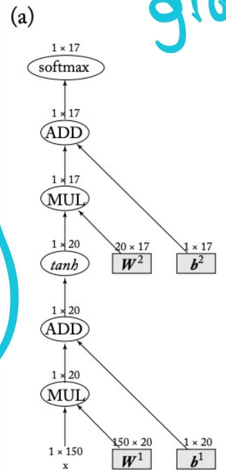
Directed Probabilistic Graphical model

# Equivalent Model Representations




$$
\begin{aligned}
\mathrm{NN}_{\mathrm{MLP2}}(\boldsymbol{x}) &= \boldsymbol{y} \\
\boldsymbol{h}^1 &= g^1(\boldsymbol{x}\boldsymbol{W}^1 + \boldsymbol{b}^1) \\
\boldsymbol{h}^2 &= g^2(\boldsymbol{h}^1\boldsymbol{W}^2 + \boldsymbol{b}^2) \\
\boldsymbol{y} &= \boldsymbol{h}^2\boldsymbol{W}^3.
\end{aligned}
$$

computation graph




https://www.morganclaypool.com/doi/pdf/10.2200/S00762ED1V01Y201703HLT037

# Revisiting our standard ML view

In Machine Learning, we ask the question of "how would you figure this out" via a **three step process**:

1. We define a **model class** (or synonymously, a **hypotheses class**). We will pick one of those to be our predictive model.
2. We define a **loss function**. This defines the "best" model, i.e. the one we're going to pick.
3. We define an **optimization algorithm**. This is *how we actually find the best model*.

# Loss Functions in Deep Learning

stimate.

| symbol | name | equation |
|---|---|---|
| $\mathcal{L}_1$ | $\mathrm{L}_1$ loss | $\lVert \mathbf{y} - \mathbf{o} \rVert_1$ |
| $\mathcal{L}_2$ | $\mathrm{L}_2$ loss | $\lVert \mathbf{y} - \mathbf{o} \rVert_2^2$ |
| $\mathcal{L}_1 \circ \sigma$ | expectation loss | $\lVert \mathbf{y} - \sigma(\mathbf{o}) \rVert_1$ |
| $\mathcal{L}_2 \circ \sigma$ | regularised expectation loss[1] | $\lVert \mathbf{y} - \sigma(\mathbf{o}) \rVert_2^2$ |
| $\mathcal{L}_\infty \circ \sigma$ | Chebyshev loss | $\max_j \lvert \sigma(\mathbf{o})^{(j)} - \mathbf{y}^{(j)} \rvert$ |
| hinge | hinge [13] (margin) loss | $\sum_j \max(0, \frac{1}{2} - \hat{\mathbf{y}}^{(j)}\mathbf{o}^{(j)})$ |
| $\text{hinge}^2$ | squared hinge (margin) loss | $\sum_j \max(0, \frac{1}{2} - \hat{\mathbf{y}}^{(j)}\mathbf{o}^{(j)})^2$ |
| $\text{hinge}^3$ | cubed hinge (margin) loss | $\sum_j \max(0, \frac{1}{2} - \hat{\mathbf{y}}^{(j)}\mathbf{o}^{(j)})^3$ |
| log | log (cross entropy) loss | $-\sum_j \mathbf{y}^{(j)} \log \sigma(\mathbf{o})^{(j)}$ |
| $\log^2$ | squared log loss | $-\sum_j [\mathbf{y}^{(j)} \log \sigma(\mathbf{o})^{(j)}]^2$ |
| tan | Tanimoto loss | $\dfrac{-\sum_j \sigma(\mathbf{o})^{(j)}\mathbf{y}^{(j)}}{\lVert\sigma(\mathbf{o})\rVert_2^2 + \lVert\mathbf{y}\rVert_2^2 - \sum_j \sigma(\mathbf{o})^{(j)}\mathbf{y}^{(j)}}$ |
| $\mathrm{D}_{\mathrm{CS}}$ | Cauchy-Schwarz Divergence [3] | $-\log \dfrac{\sum_j \sigma(\mathbf{o})^{(j)}\mathbf{y}^{(j)}}{\lVert\sigma(\mathbf{o})\rVert_2 \lVert\mathbf{y}\rVert_2}$ |

SSE

SVM

https://arxiv.org/abs/1702.05659

Not much new here!

# Revisiting our standard ML view

In Machine Learning, we ask the question of "how would you figure this out" via a **three step process**:

1. We define a **model class** (or synonymously, a **hypotheses class**). We will pick one of those to be our predictive model.
2. We define a **loss function**. This defines the "best" model, i.e. the one we're going to pick.
3. We define an **optimization algorithm**. This is *how we actually find the best model*.

# Revisiting our standard ML view

In Machine Learning, we ask the question of "how would you figure this out" via a **three step process**:

1. We define a **model class** (or synonymously, a **hypotheses class**). We will pick one of those to be our predictive model.
2. We define a **loss function**. This defines the "best" model, i.e. the one we're going to pick.
3. We define an **optimization algorithm**. This is *how we actually find the best model*.

# Optimization Algorithms

**Leave it to next week!**

Essentially, we need to do some kind of gradient-based optimization, but in a fancy way (backpropagation)

https://emiliendupont.github.io/2018/01/24/optimization-visualization/

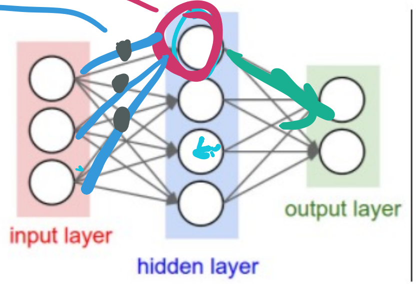
# What I am aiming for (con't)

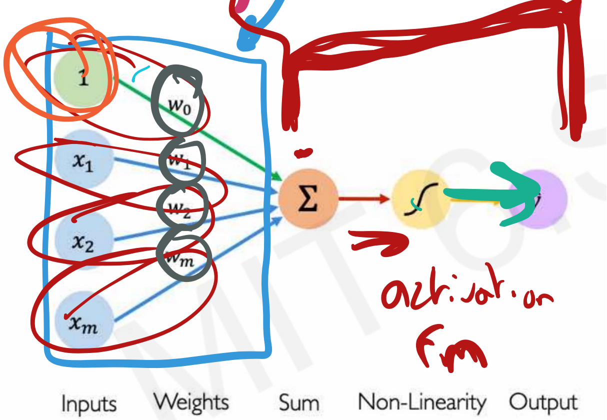
- Today:
  - Have a discussion about what DL is/isn't
  - Link back to our standard approach to ML
  - Understand building blocks of models
  - Understand how predictions are made in MLPs
  - See and run end-to-end code example in keras/tensorflow

# Building Blocks of a Neural Network

- Network structure
  - **Input layer**
  - **Hidden layer**
  - **Output layer**
- Neuron/Unit
  - Input
  - Weights
  - Sum
  - Activation function
  - output

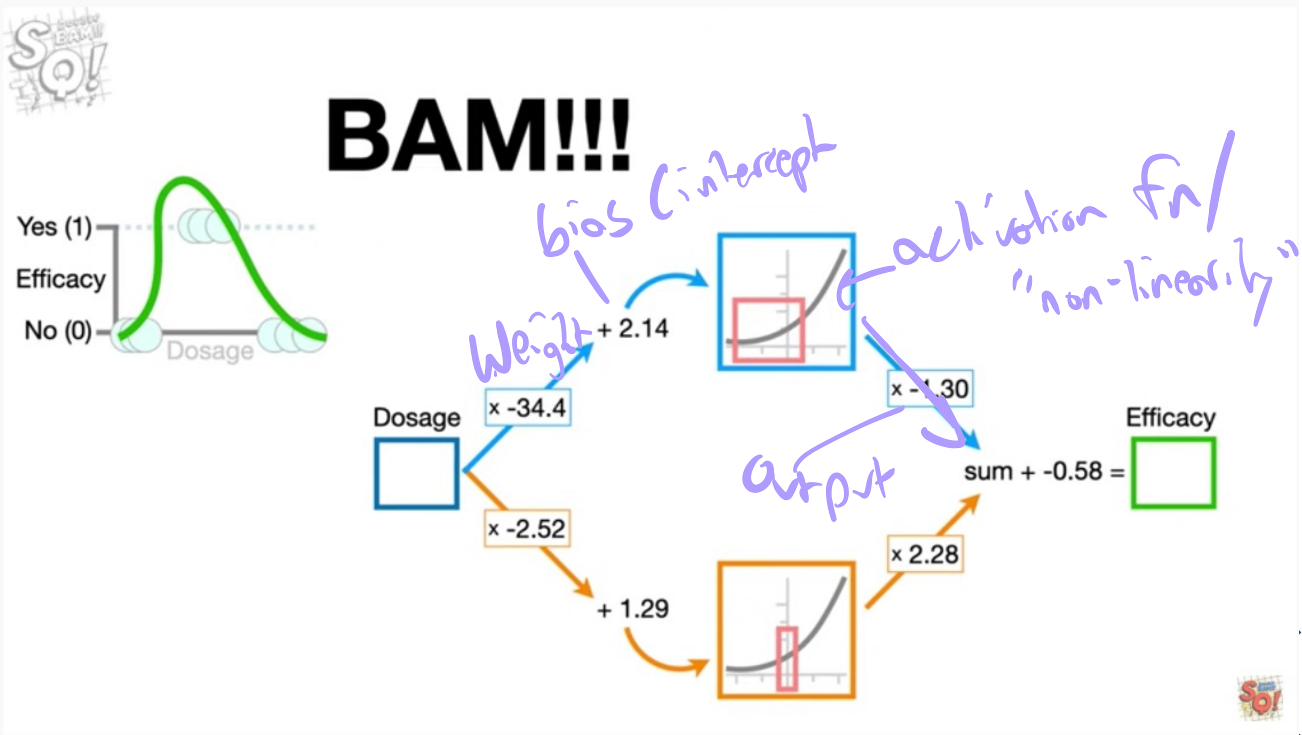
https://cs231n.github.io/neural-networks-1/



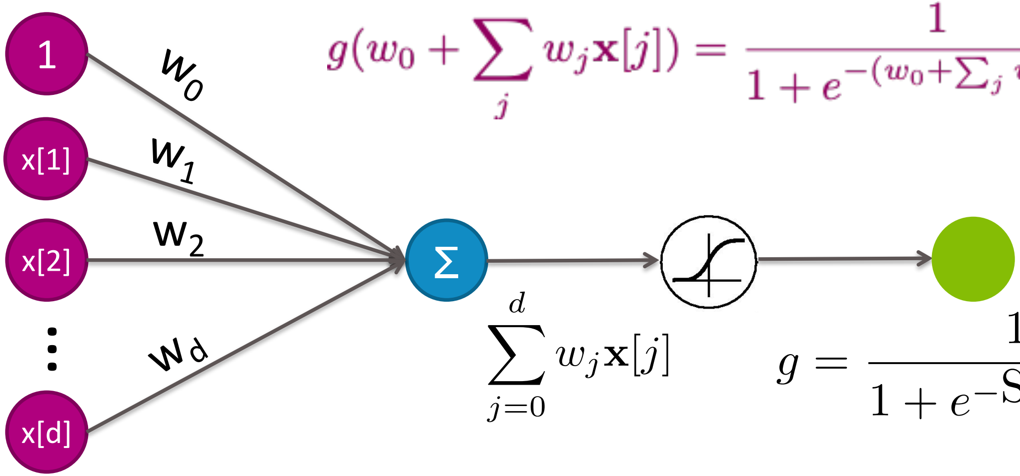




http://introtodeeplearning.com/

# Linking to StatQuest



# One more time! You label!




$$g(w_0 + \sum_j w_j \mathbf{x}[j]) = \frac{1}{1 + e^{-(w_0 + \sum_j w_j \mathbf{x}[j])}}$$

$$\sum_{j=0}^{d} w_j \mathbf{x}[j]$$

$$g = \frac{1}{1 + e^{-\mathrm{Score}(\mathbf{x})}}$$

https://courses.cs.washington.edu/courses/cse416/21sp/

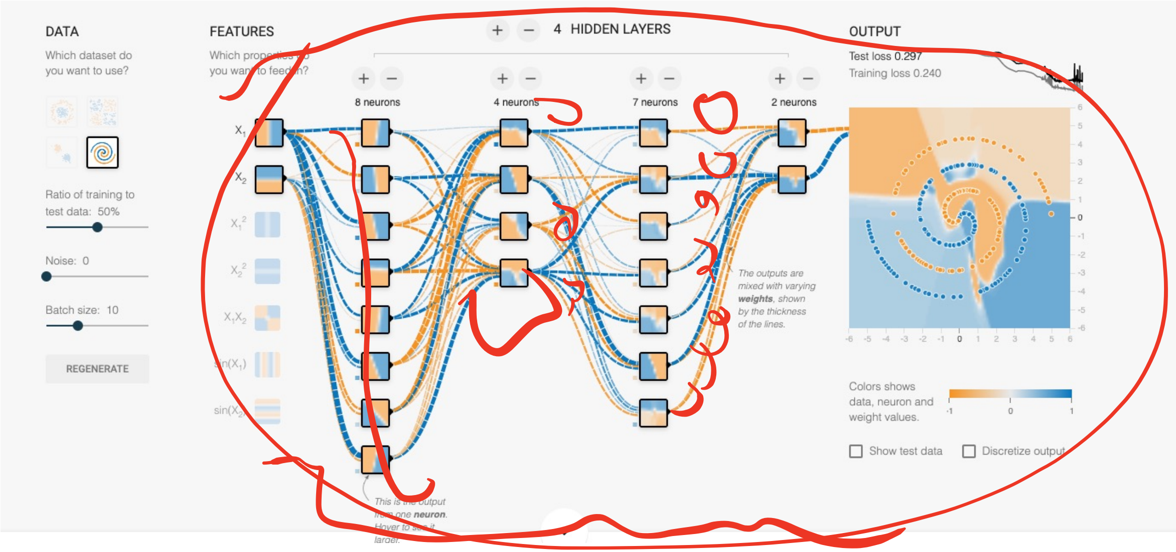
# Main point

- You'll see neural nets drawn in a lot of different ways, but these are the main building blocks
  - … kind of. We'll see other ideas in convolutional neural networks, but this is sufficient for now.

# What I am aiming for (con't)

- Today:
  - Have a discussion about what DL is/isn't
  - Link back to our standard approach to ML
  - Understand building blocks of models
  - Understand how predictions are made in MLPs
  - See and run end-to-end code example in keras/tensorflow

# Playing around with Neural Nets




https://playground.tensorflow.org

# Some questions

$y = \text{Sigmoid}(w^T x)$

1. Can you create a logistic regression model?
2. What are the hyperparameters you can identify? What do you think they do?
3. Create a perfect classifier for the concentric circle dataset using
    1. as few neurons as possible
    2. As few features as possible
    3. As few layers as possible and as few features as possible
4. Create a classifier that fits the really hard dataset

# Some non-obvious things

- A single-layer neural net is a logistic regression (with a sigmoid activation function)
- There are many, many, many hyperparameters, and you can fit the same model with drastically different values of those
- The activation function is critical because it introduces a **nonlinearity**

# Code demo

https://colab.research.google.com/github/keras-team/keras-io/blob/master/examples/vision/ipynb/mnist_convnet.ipynb#scrollTo=PBWCmAxMQSgF