Introduction to Machine Learning

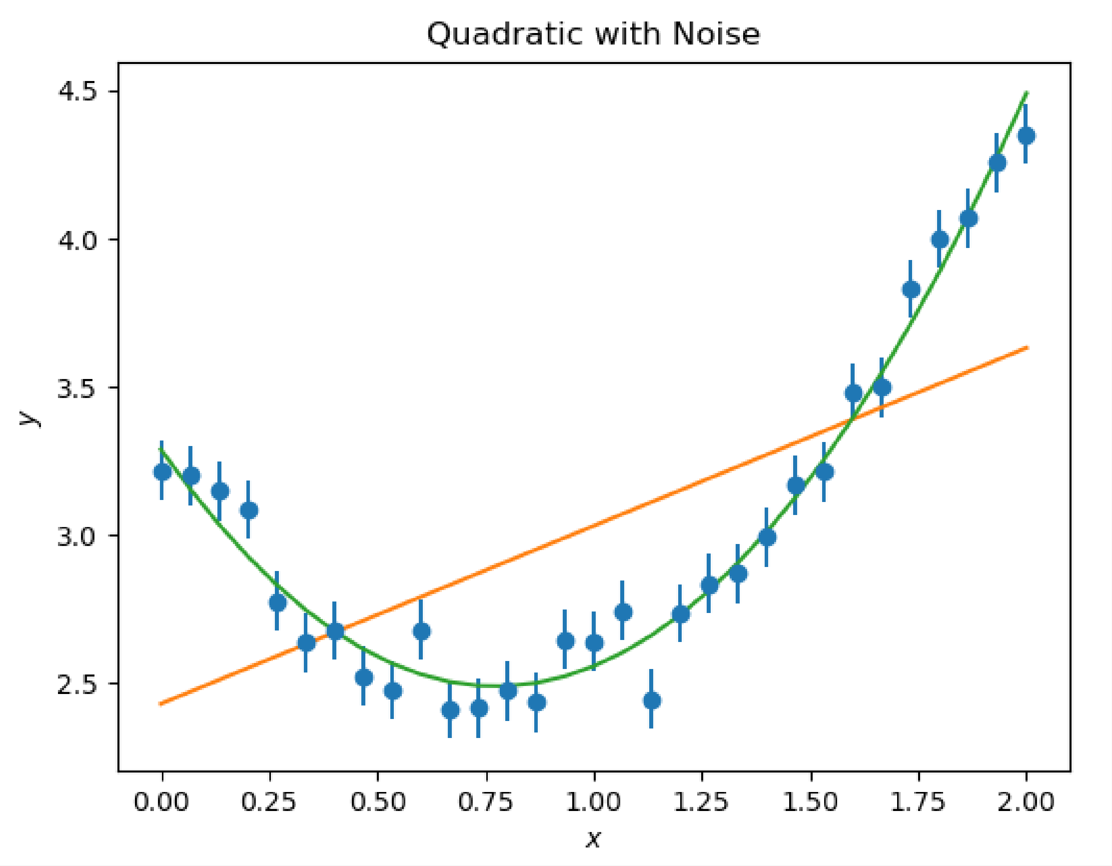
AST 5765
Michael Himes
November 19, 2020

Overview

- Review model fitting
- What is machine learning?
- Types of ML
- ML details
- Demo: visual neural network
- ML applications
- Demo: basic dense neural networks

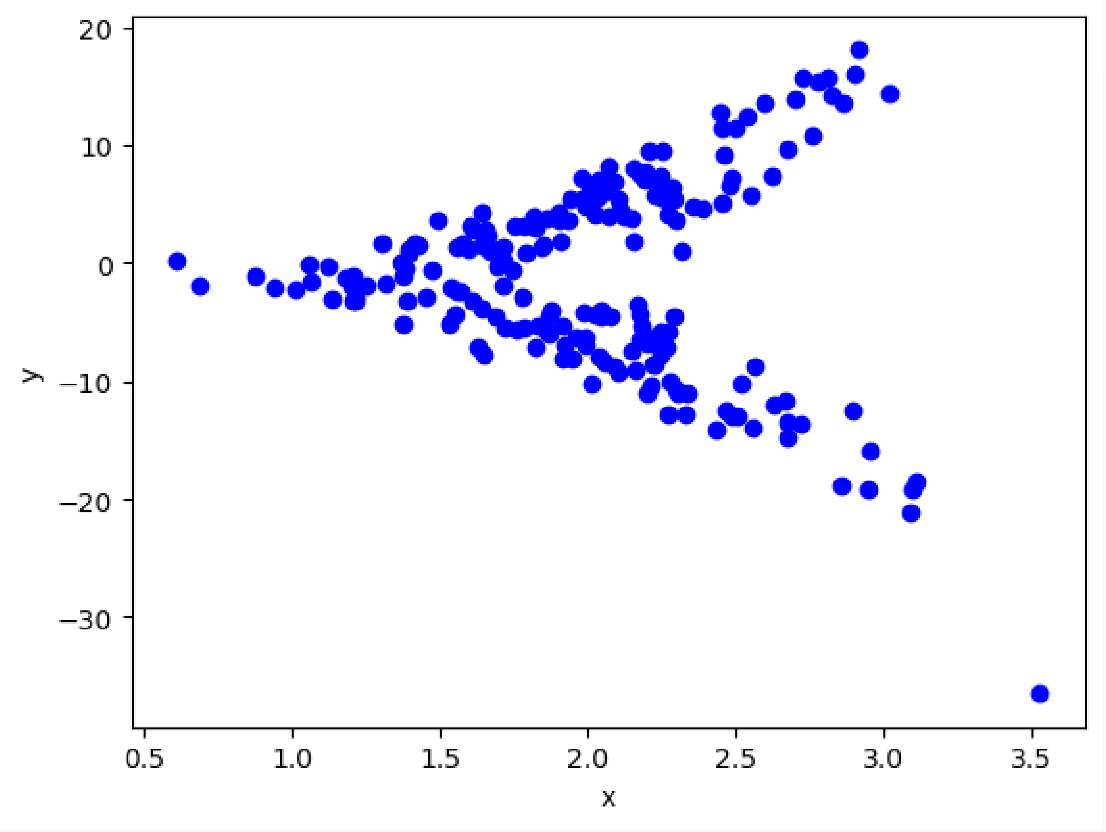
Model fitting

- If we have a data set of (x, y) pairs, we can fit the parameters of some model
- Least squares
  - $y_i = p_1 x_i + p_0$
  - $y_i = p_2 x_i^2 + p_1 x_i + p_0$
- Modeling a complicated function?
  - $y_i = \Sigma_n p_n x_i^n$



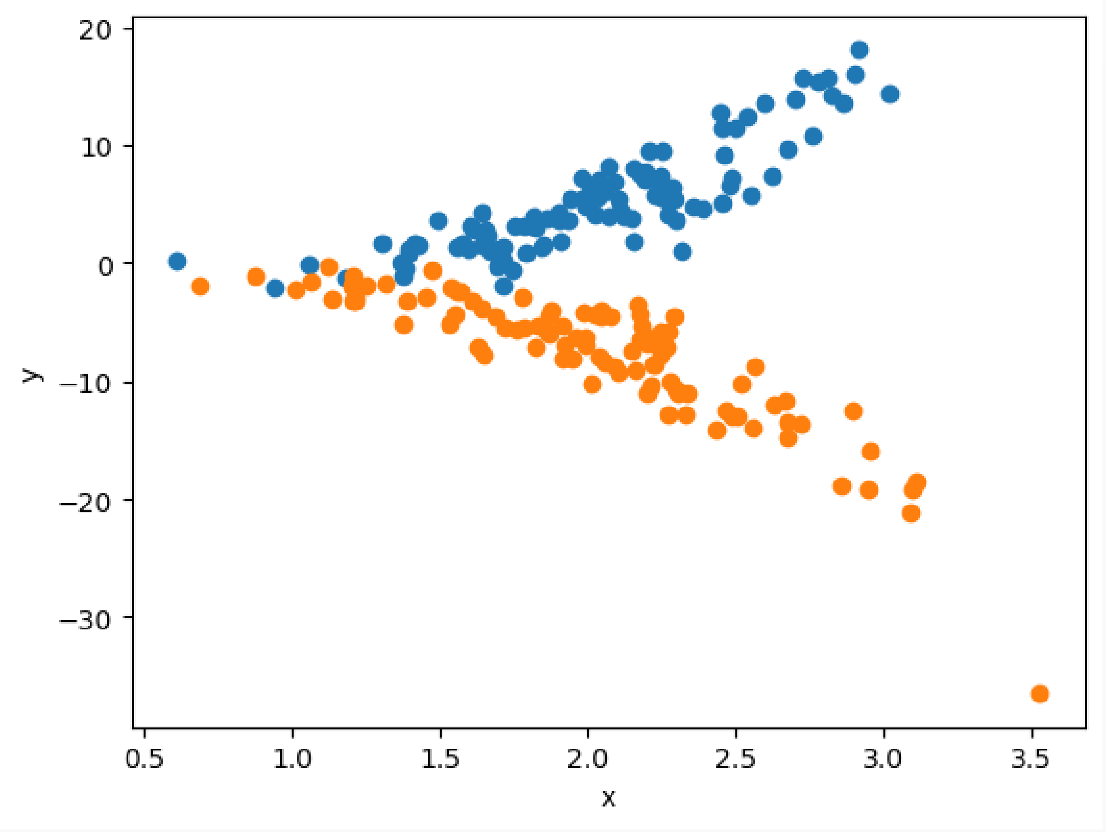
Model fitting

- What if we have a data set of inputs ($x_{1,i}$, $x_{2,i}$, …, $x_{n,i}$) and outputs ($y_{1,i}$, $y_{2,i}$, …, $y_{m,i}$)?
- Least squares not tractable
- Data may not appear to be a function
  - Experimental variability, or measuring multiple effects?



Model fitting

- What if we have a data set of inputs ($x_{1,i}$, $x_{2,i}$, …, $x_{n,i}$) and outputs ($y_{1,i}$, $y_{2,i}$, …, $y_{m,i}$)?
- Least squares not tractable
- Data may not appear to be a function
  - Experimental variability, or measuring multiple effects?



Machine learning (ML)

- “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.” – Mitchell (1997)
- An algorithm that learns from data, without prior knowledge about the data
  - ML models are blank slates
  - ML is not magic that solves all of our problems

Types of ML

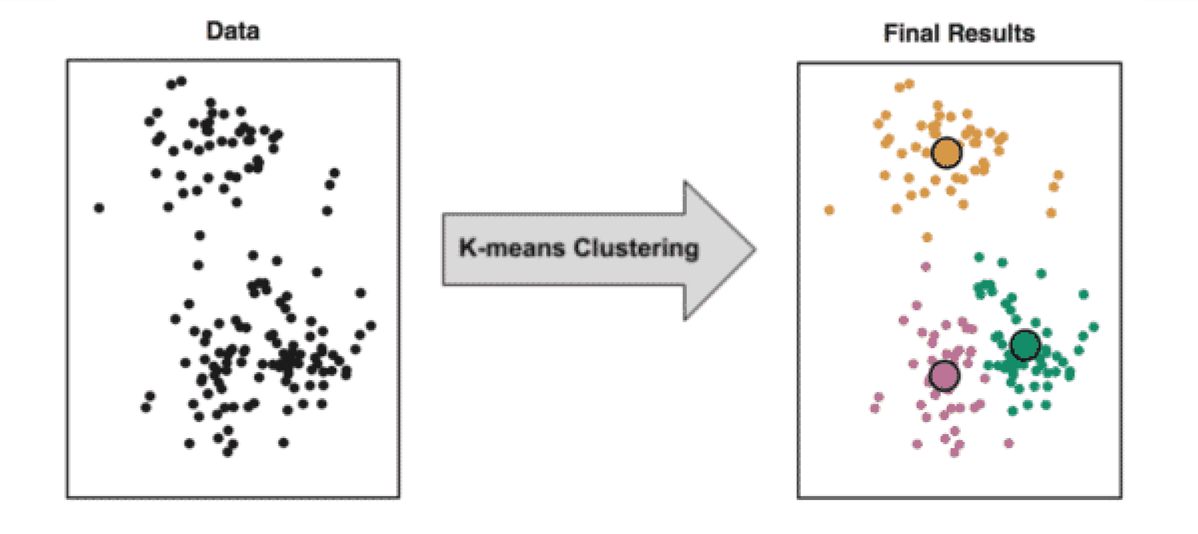
- Supervised learning
  - Models labeled data sets of input-output pairs
- Unsupervised learning
  - Models unlabeled data sets of outputs
- Semi-supervised learning
  - Combination of labeled and unlabeled data
- Reinforcement learning
  - Seeks to maximize reward or minimize risk

Supervised ML

- Training data is used to teach the model what the output should be for a given input
  - The data is fed to the model in batches
  - Model makes predictions on each batch
  - Small adjustments to parameters are made each time based on the error of its output as measured by the loss function (backpropagation)
- Fits model parameters to approximate a function, similar to least squares

Unsupervised ML

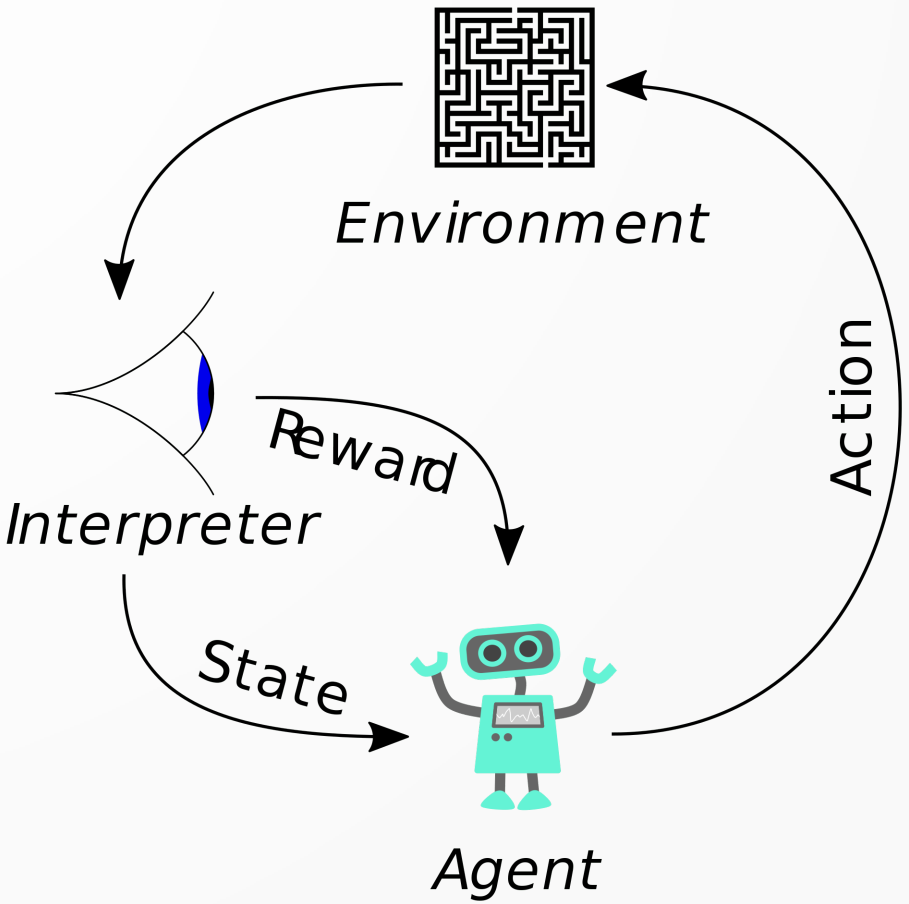
- Finds patterns in an unlabeled data set (outputs with no corresponding inputs)
- Common uses: clustering, anomaly detection



“An Introduction to Statistical Learning” by James, Witten, Hastie, and Tibshirani

Reinforcement ML

- Agent seeks to accomplish some task
- For each action the agent takes, the interpreter evaluates the reward function
- Agent’s future actions depend on rewards of past states

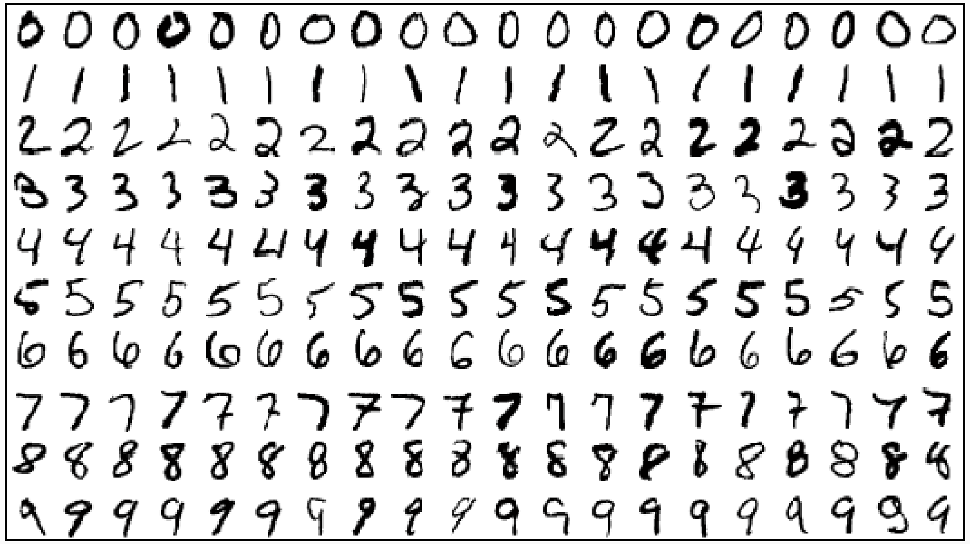


https://i2.wp.com/webindream.com/wp-content/uploads/2017/12/Reinforcement_learning_diagram.svg_.png

ML Task Examples

- Classification
- Regression (predict a value given some input)
- Transcription
- Translation

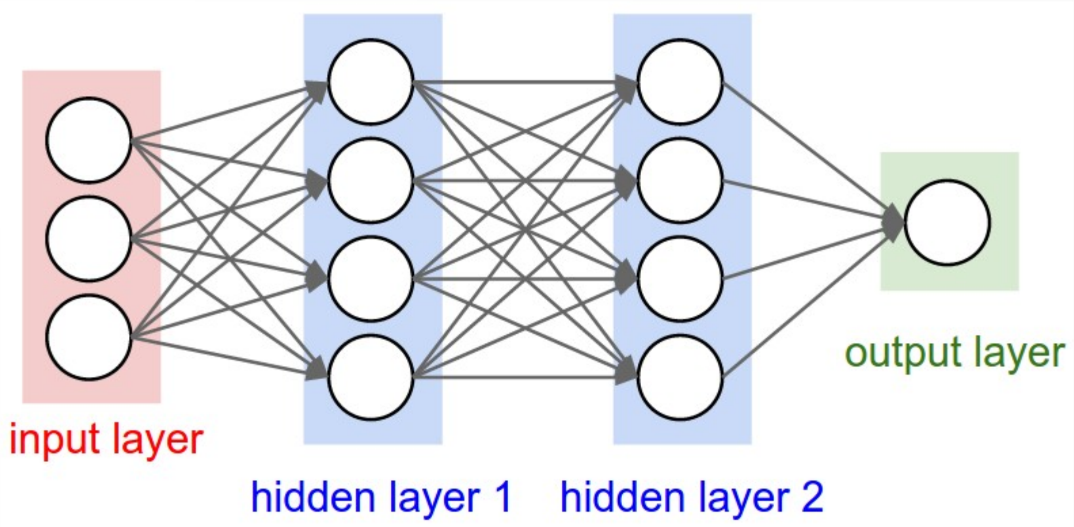
https://www.shutterstock.com/image-photo/dog-cat-179786294



Mnist database of handwritten digits

Neural networks (NNs)

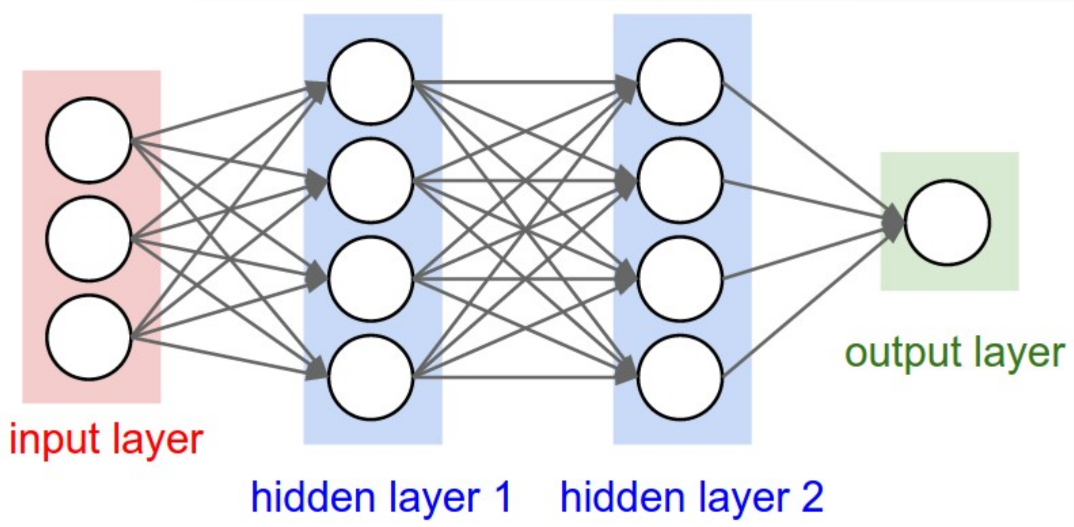
- Type of ML model
- Uses sequential layers of nodes, where the outputs of nodes from one layer are the inputs for the next layer's nodes



Andrej Karpathy

Neural networks (NNs)

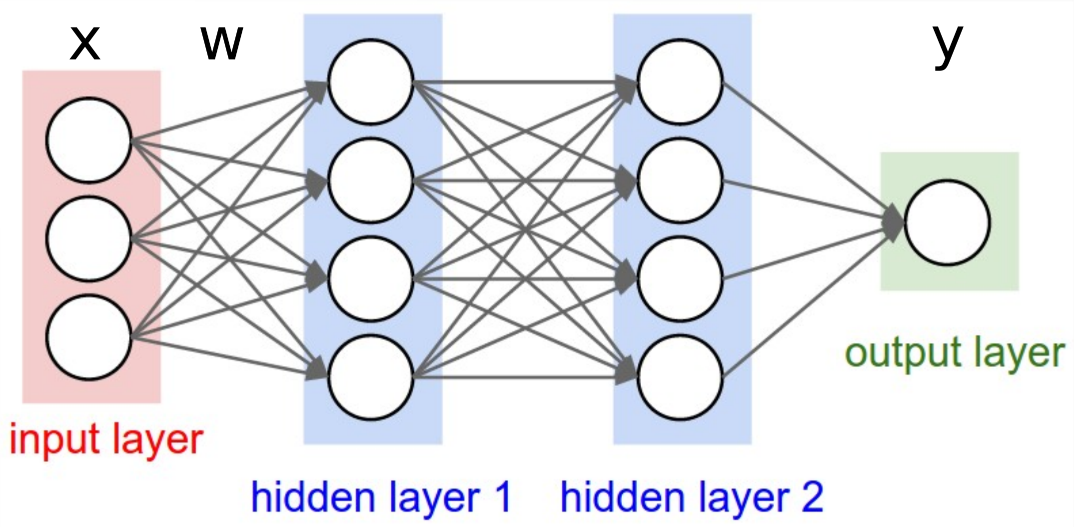
- Each node has an activation function to transform the inputs (model parameters)
- Each connection between nodes has an associated weight (model parameters)



Andrej Karpathy

Neural networks (NNs)

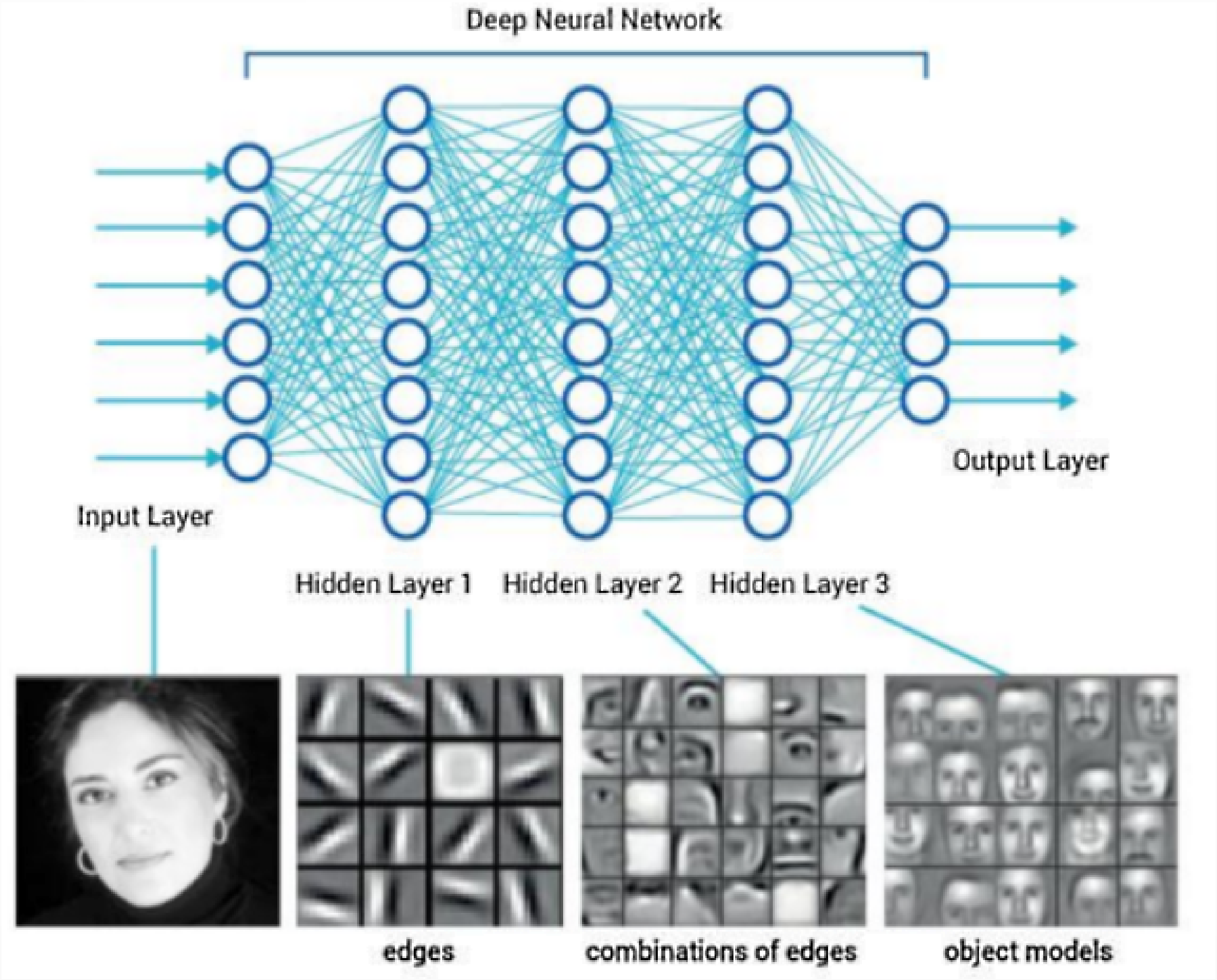
- Nodes work similar to Boolean logic gates
- "Hidden layers" work on higher-order features in the data



Andrej Karpathy

- No hidden layers and linear activation functions? Linear regression!
  - $\mathbf{y}_i = \mathbf{w} \cdot \mathbf{x}_i$
  - $\mathbf{y}_i = w_1 x_1 + w_2 x_2 + \ldots$

What are hidden layers doing?

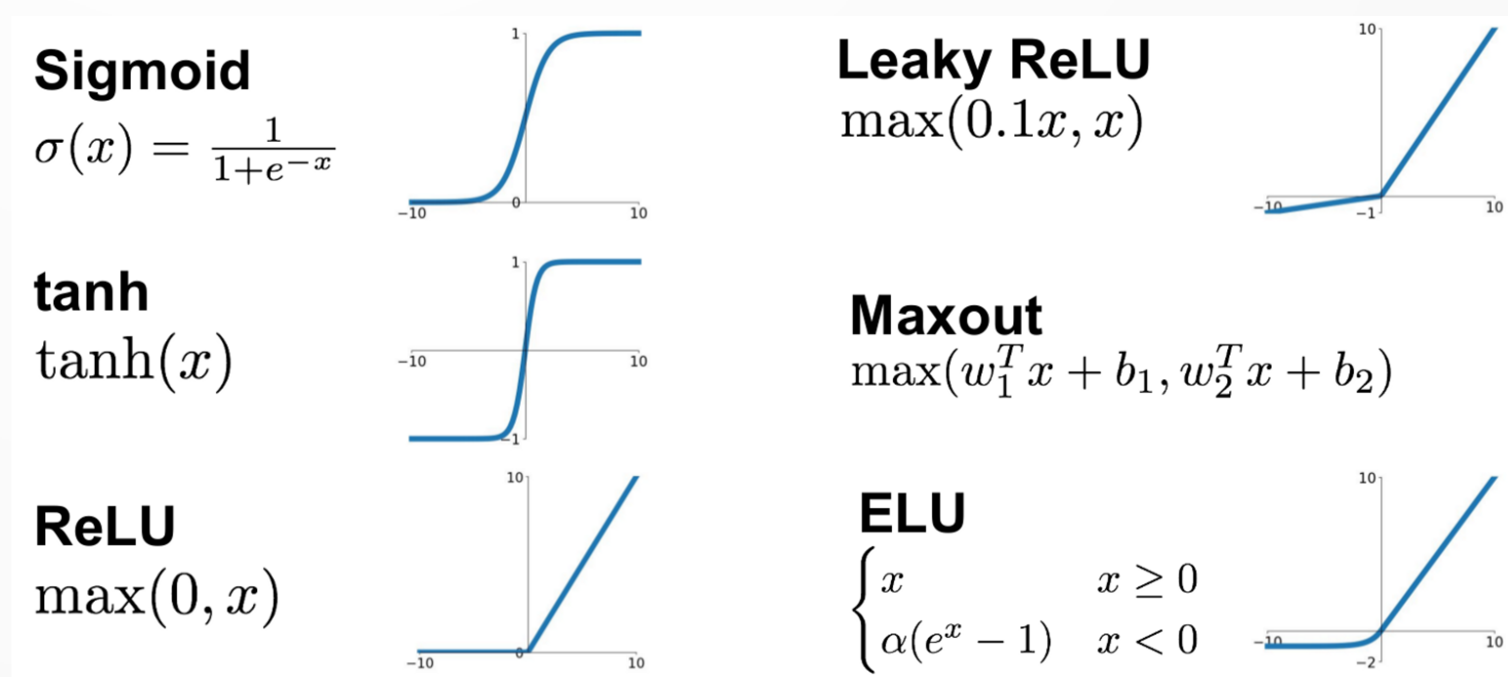


https://iq.opengenus.org/hidden-layers/

Activation Functions

- Controls how node inputs are mapped to outputs
  - Can be any function, as long as it is defined for (-∞,∞)
- Common activations:
  - Linear (identity)
  - Exponential
  - Softmax
  - Softplus
  - Scaled ELU

https://miro.medium.com/max/1192/1*4ZEDRpFuClpUjNgjDdT2Lg.png

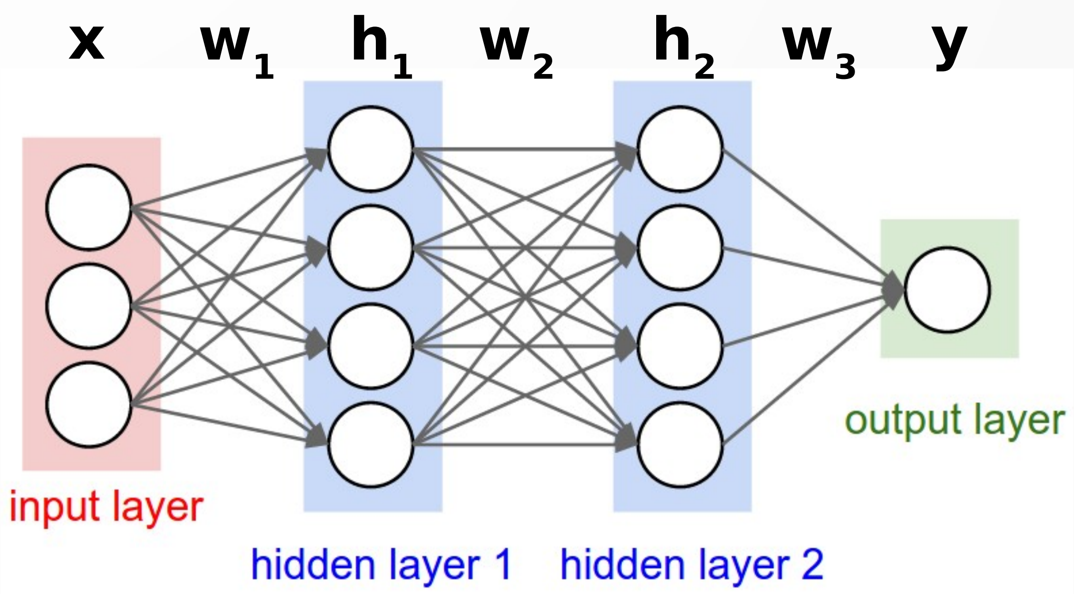


Rectified Linear Unit

Exponential Linear Unit

NN Math Example

- NNs are just doing weird math… Let's see it
- Assume linear activation for $\mathbf{y}$,
  sigmoid activation for $\mathbf{h_2}$,
  and tanh acivation for $\mathbf{h_1}$



Andrej Karpathy

- $\mathbf{y} = \mathbf{w_3} \cdot \mathbf{h_2}$
- $\mathbf{h_2} = \{1 + \exp(-\mathbf{w_2} \cdot \mathbf{h_1})\}^{-1}$
- $\mathbf{h_1} = \tanh(\mathbf{w_1} \cdot \mathbf{x})$
- $\mathbf{y} = \mathbf{w_3} \cdot \{1+\exp(-\mathbf{w_2} \cdot \tanh(\mathbf{w_1} \cdot \mathbf{x}))\}^{-1}$

Demo: visual NN

- Before we get into more details...
- https://playground.tensorflow.org

How to train an NN

- A data set is split into training, validation, and test sets
  - Training set: trains the network and updates the weights
  - Validation set: monitors training and prevents overfitting
  - Test set: evaluates model performance/generalization

How to train an NN

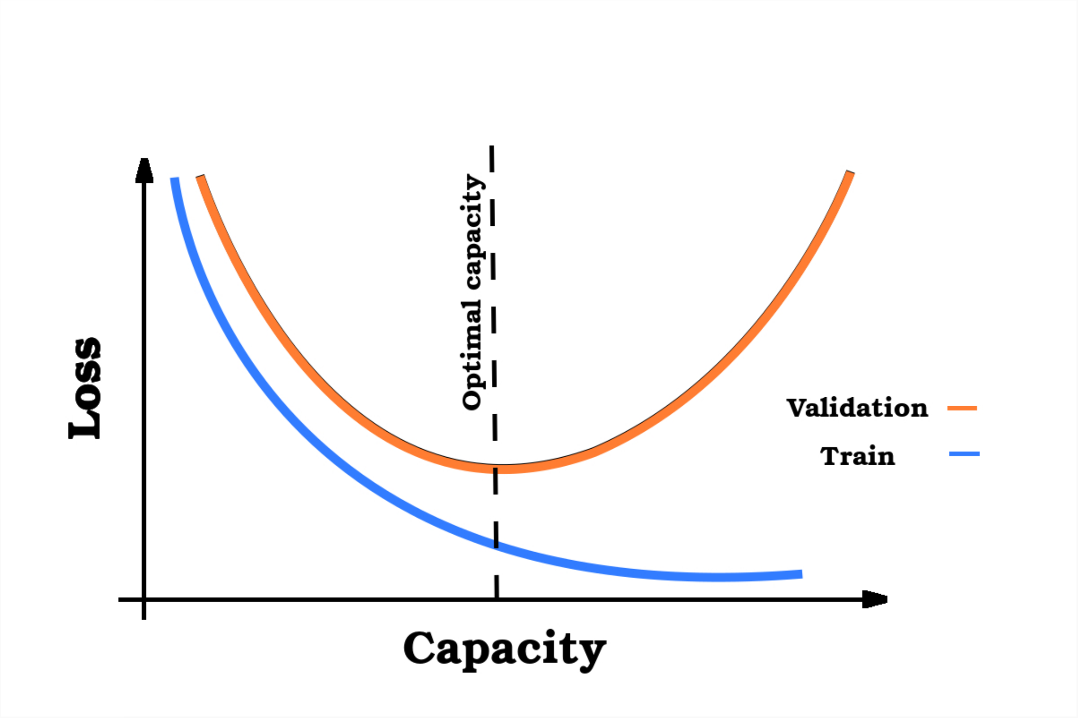
- Loss functions govern how an NN learns to minimize the error in predictions
- Common losses:
  - Mean squared error
  - Mean absolute error
  - Binary crossentropy (for binary classification)
  - Categorical crossentropy (for multiclassification)
- You can also define your own!

How to train an NN

- Training steps
  - Make a prediction for some input
  - Calculate the error according to loss function
  - Update the weights via backpropagation
- Validation steps
  - Tests the generality of the model during training
  - Halts training when overfitting detected (early stopping)

How to train an NN

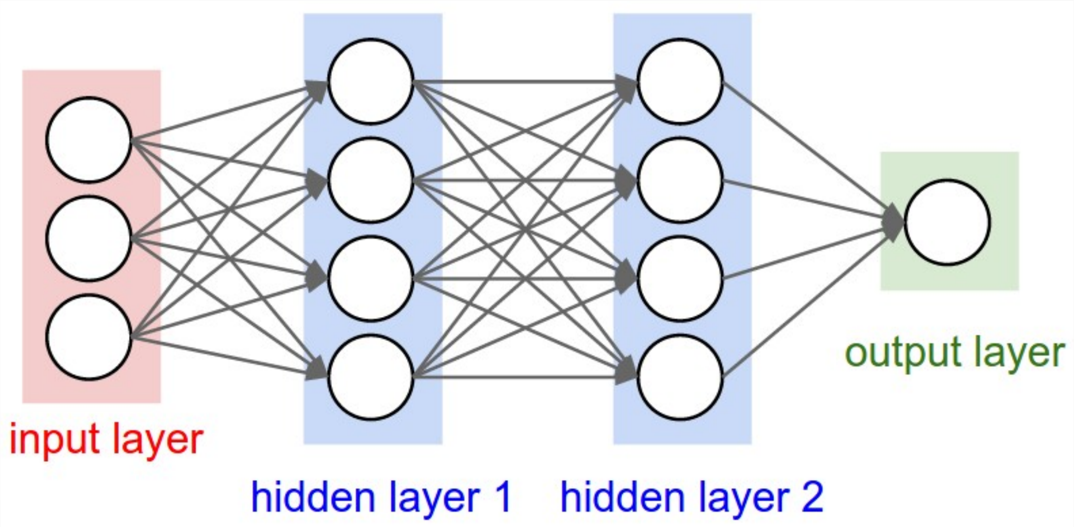
- Every N training steps, make 1 validation step
- Repeat for many epochs (iterations through the training set) to learn everything that it can, without overfitting
- Then, the model can be applied to the test set, which evaluates the performance of the model



https://media.mljar.com/blog/validation-learning-not-memorizing/head/validation-plot.jpg

Feedforward/Dense NNs

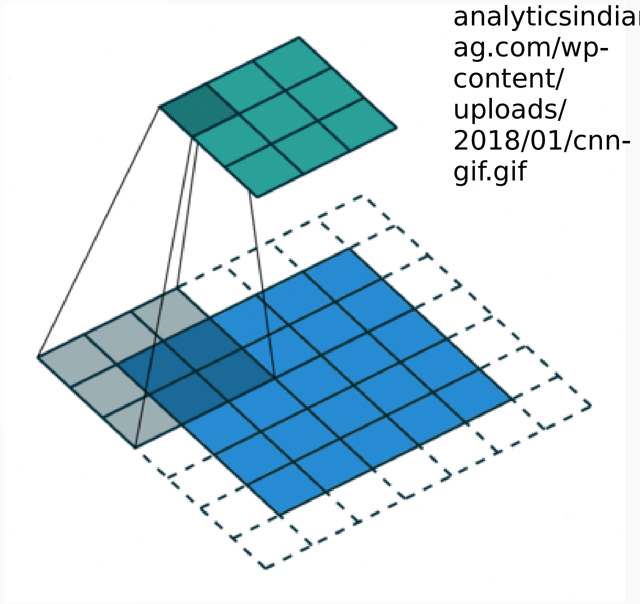
- Simplest NN, where nodes are connected to all nodes in adjacent layers
- Useful in tasks like image classification and regression
- In theory, they can model anything!
  - In practice, other approaches are better for certain tasks



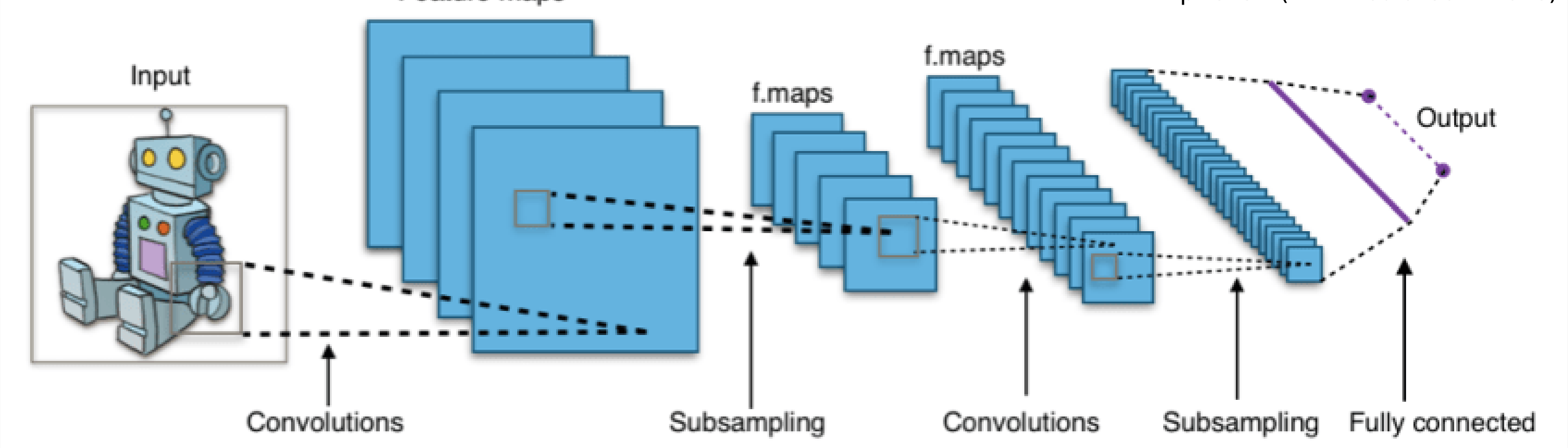
Andrej Karpathy

Convolutional NNs

- Nodes are connected to only a few nodes in adjacent layers
  - Like collection of dense NNs, where each NN is highly specialized (feature maps)
- Useful in classification, regression, computer vision, signal/image processing



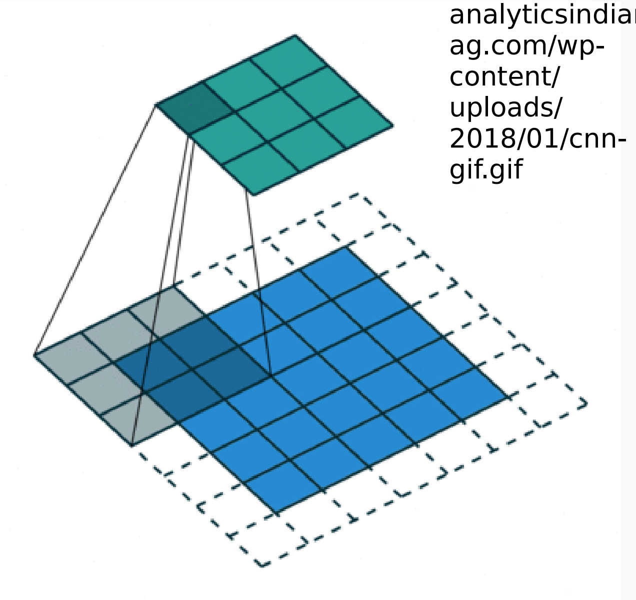
https://analyticsindiamag.com/wp-content/uploads/2018/01/cnn-gif.gif



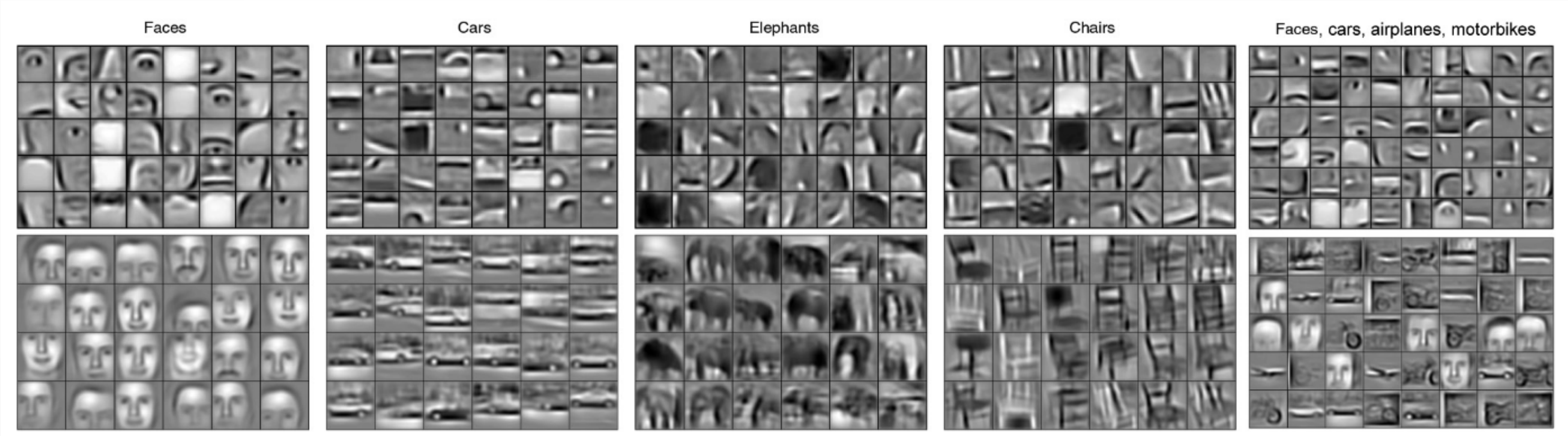
Aphex34 (Wikimedia Commons)

Convolutional NNs

- Nodes are connected to only a few nodes in adjacent layers
  - Like collection of dense NNs, where each NN is highly specialized (feature maps)
- Useful in classification, regression, computer vision, signal/image processing



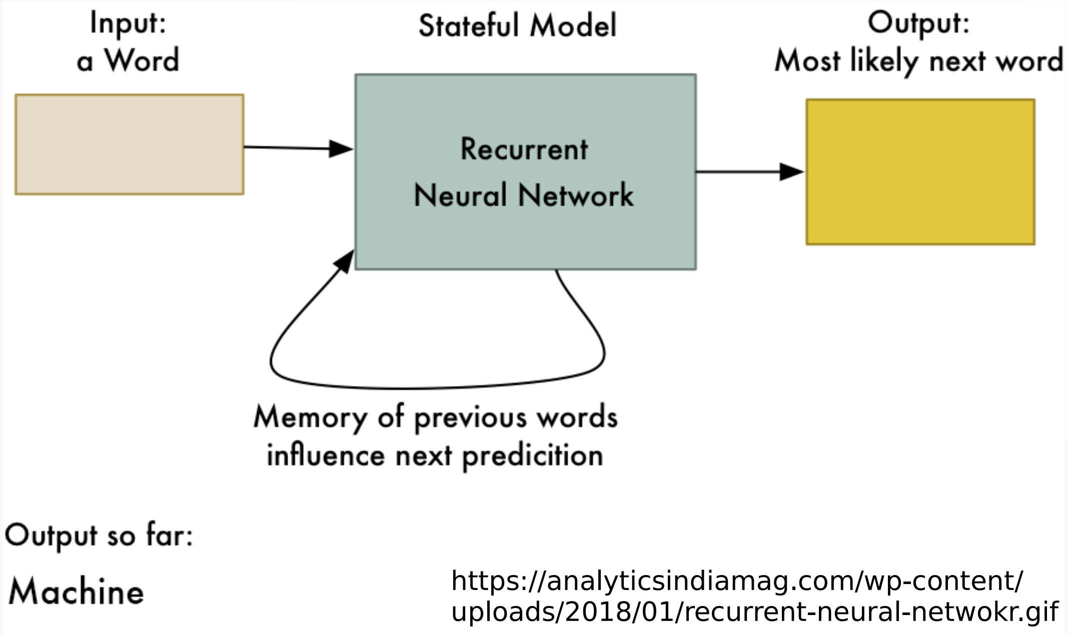
https://analyticsindiamag.com/wp-content/uploads/2018/01/cnn-gif.gif



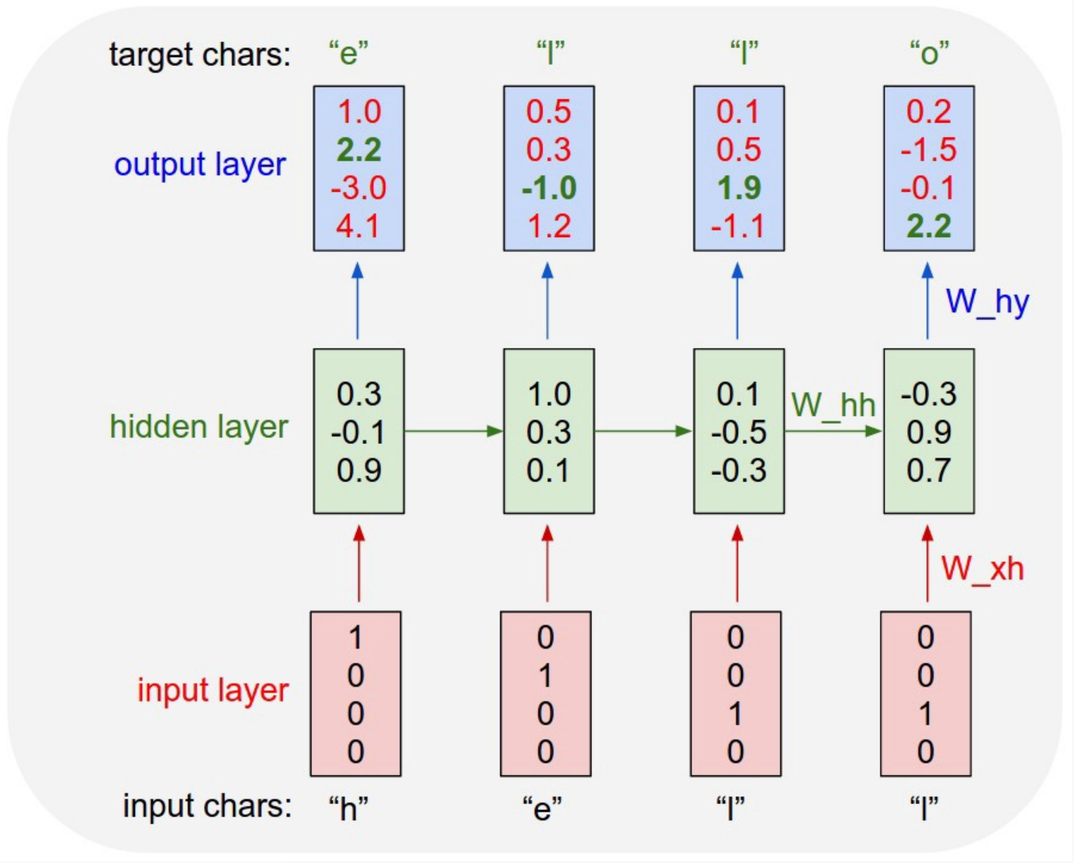
Lee et al. (2011) “Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks” DOI:10.1145/2001269.2001295

Recurrent NNs

- Stores information about previous inputs to inform future predictions
- Useful for time-series data



https://analyticsindiamag.com/wp-content/uploads/2018/01/recurrent-neural-netwokr.gif



https://miro.medium.com/max/902/1*IMaIbwl6uj3nlqxixZYFvA.jpeg

Applications

- Automated driving gets a lot of attention
- But, there are a lot of other useful applications…
- https://www.youtube.com/watch?v=D5VN56jQMWM
- https://www.youtube.com/watch?v=gn4nRCC9TwQ
- https://www.youtube.com/watch?v=fUyU3lKzoio
- https://www.youtube.com/watch?v=aFuA50H9uek

- If interested, some more applications:
- https://www.youtube.com/watch?v=HKcO3-6TYr0
- https://www.youtube.com/watch?v=Bui3DWs02h4

Further resources

- Online
  - Image classification: https://distill.pub/2018/building-blocks/
  - Build NN with Numpy: https://towardsdatascience.com/neural-net-from-scratch-using-numpy-71a31f6e3675
- Books
  - *Deep Learning* by Goodfellow, Bengio, and Courville
  - *Pattern Recognition and Machine Learning* by Bishop
- Internet searches! There are many great tutorials available

Demo: basic dense NNs

- Model a unit circle with parameter theta
- Model a Gaussian with parameters position, mean, and standard deviation
- Of course, neither of these actually need ML to solve – we're just demonstrating how to use ML
  - Most problems require graphics processing units (GPUs) in order to train in a reasonable amount of time