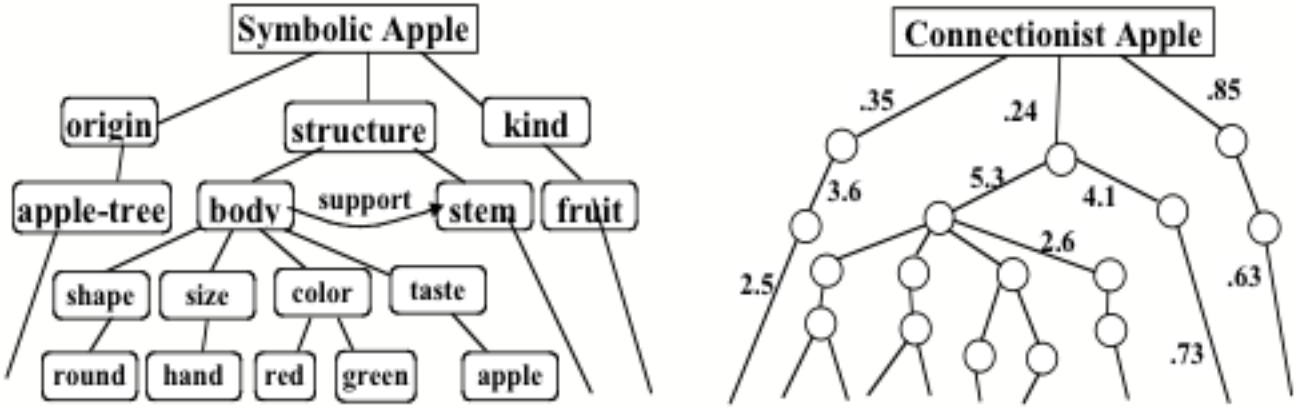
MC4

History of Artificial Intelligence, Neural Networks and Applications

Artificial Intelligence: A Brief History

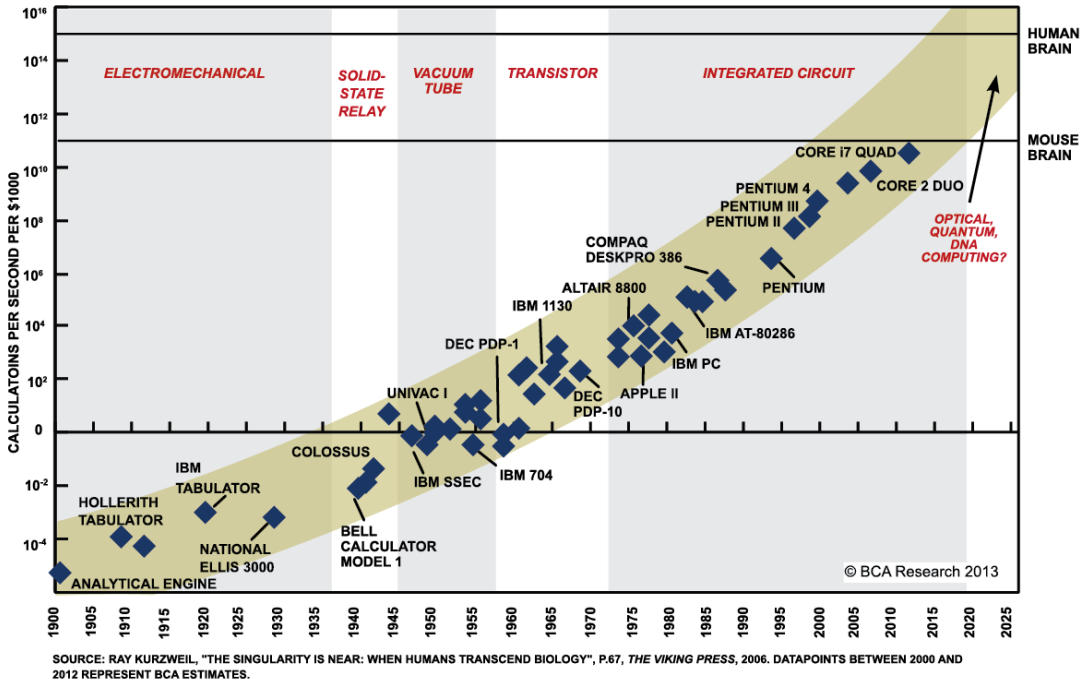
- Artificial intelligence: *An artificial system developed in computer software, physical hardware, or other context that solves tasks requiring human-like perception, cognition, planning, learning, communication, or physical action* (NDAA, 2019)
- AI was split into two approaches: symbolic and connectionist
  - Symbolic = computation on human-readable representations by logical statements
  - Connectionist = computation on numerical representations by networked connections



Steve Daniels, Medium

Artificial Intelligence: A Brief History

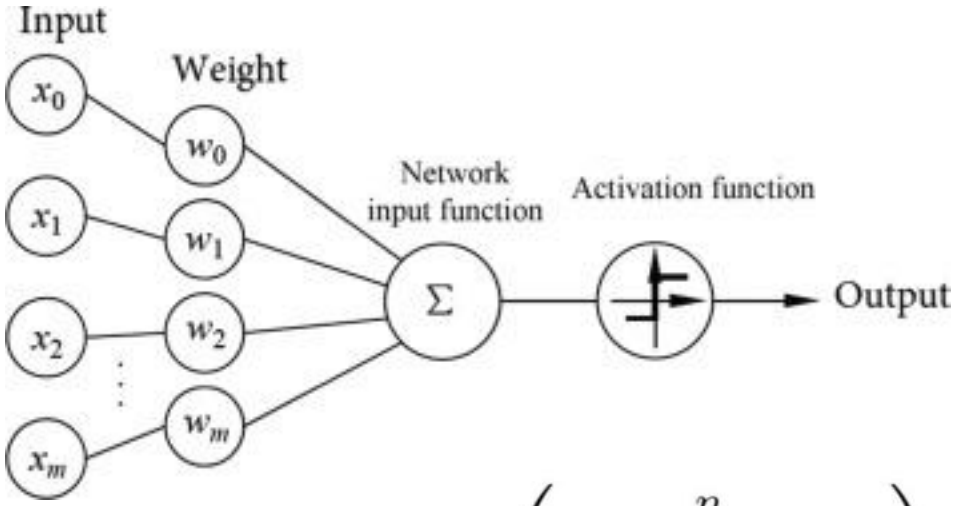
- Artificial intelligence: *An artificial system developed in computer software, physical hardware, or other context that solves tasks requiring human-like perception, cognition, planning, learning, communication, or physical action* (NDAA, 2019)
- AI was split into two approaches: symbolic and connectionist
  - Symbolic = computation on human-readable representations by logical statements
  - Connectionist = computation on numerical representations by networked connections
- In the mid-to-late 20th century, symbolic AI dominated the field
- AI Winters: periods of stalling due to computational or otherwise (70s-80s, 90s)
- In the 2010s, larger connectionist models became more feasible as computing power increased exponentially
- More recently, rise of transformers, some neurosymbolic models



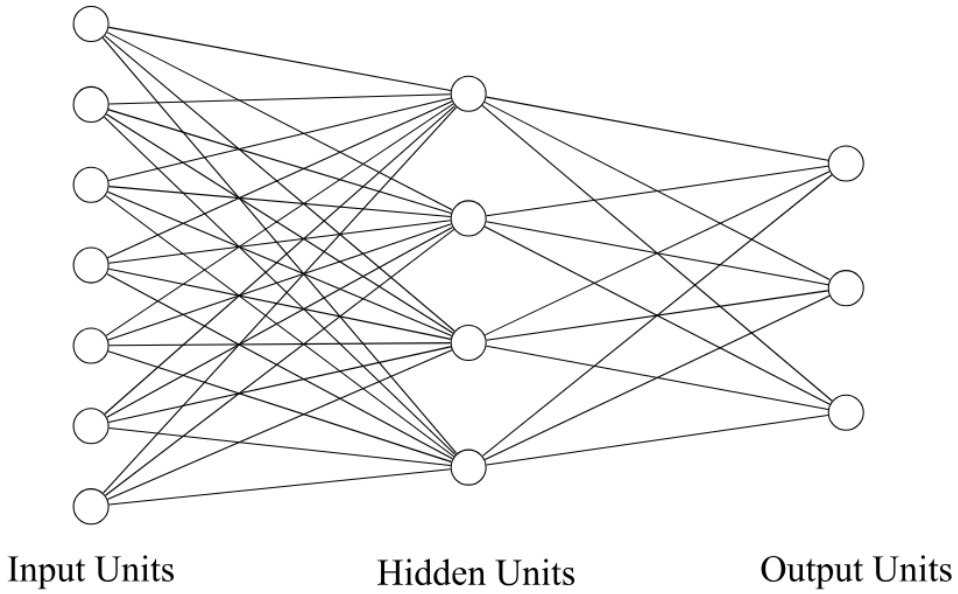
via ExtremeTech

Connectionist Models

- Connectionism uses abstractions of the brain's circuitry known as **Artificial Neural Networks (ANNs)**
- **Perceptron**: early connectionist model, simple architecture
  - **Input layer** receives input values
  - **Weights** applied to each neuron's input
  - **Activation function** allows network to learn nonlinear relationships
  - **Output layer** returns weighted sum of neurons + activation function
- Later, **multilayer perceptron** was introduced to solve more complex problems (see more on this here)
  - The layers in between input and output are known as the **hidden layers**
  - Typically feedforward

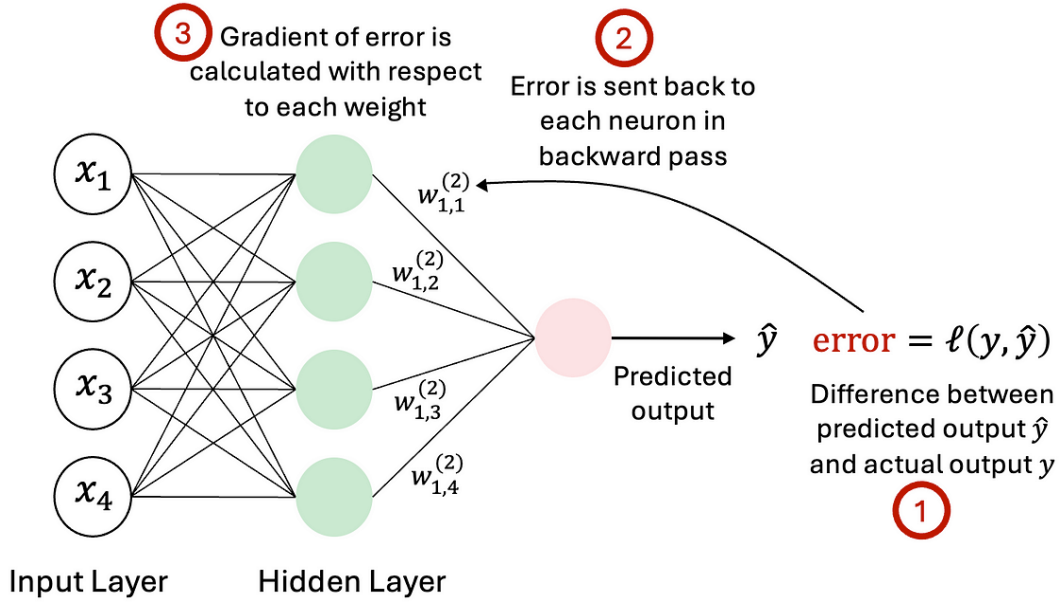


$$f\left(b + \sum_{i=1}^{n} x_i w_i\right)$$



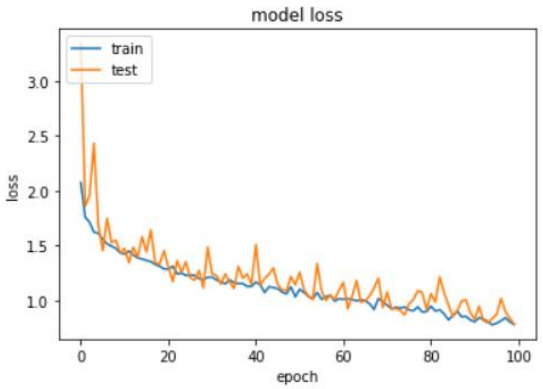
Learning in ANNs vs. Classical ML Models

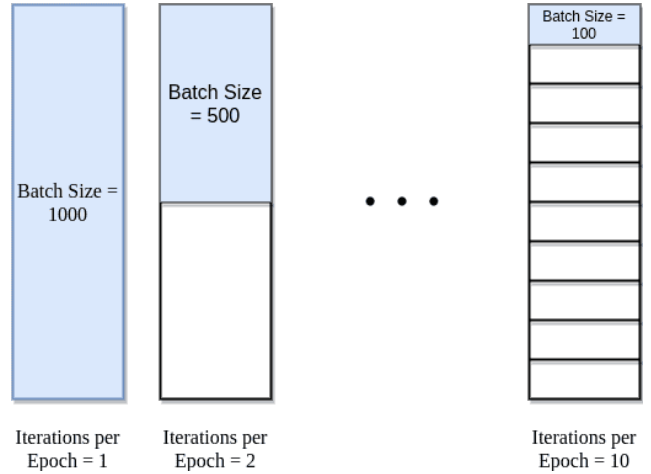
- Consider the regression problem of fitting weight matrix **m** for data approximating **y = mx + b**
- Linear regression typically solves this via approximating m using least squares
- ANNs learn via:
  - **Propagation:** inputs are passed through the network to obtain an output, and then
  - **Loss Function:** Difference between the output and expected value (error/loss) is calculated
  - **Backpropagation:** Loss used to update the weights of previous layers
- Instead of an exact equation to fit, the model updates its internal weights over the course of several exposures to data

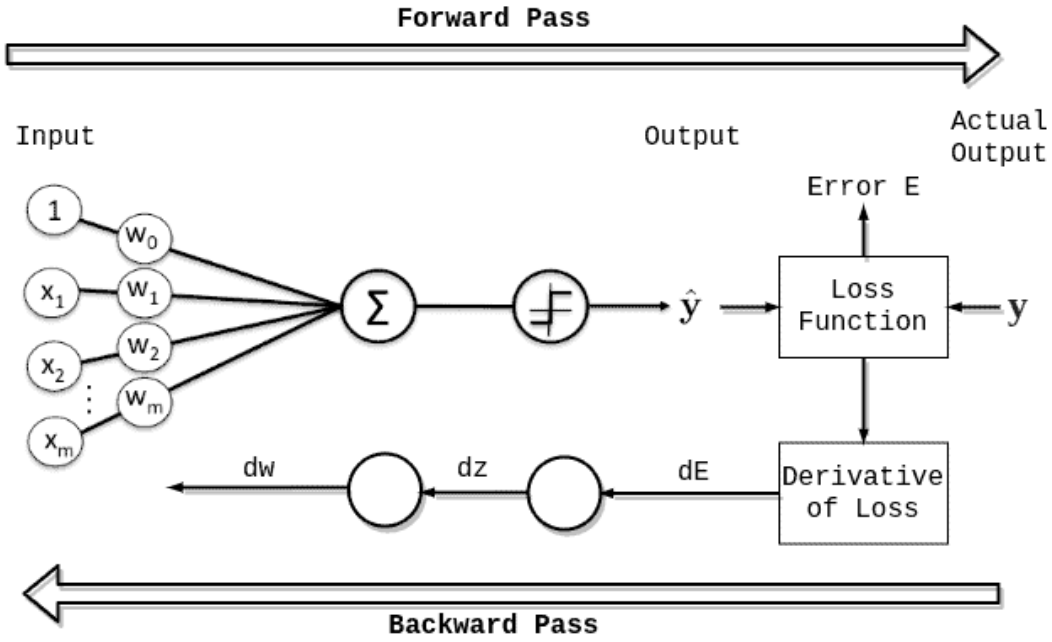


Essential Neural Network Model Training Pipeline

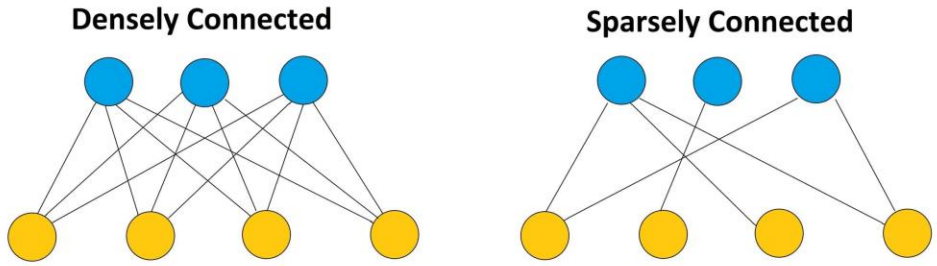
- ANN training typically follows the scheme:
  - Split dataset into **batches**
  - Train neural network on each batch, updating weights on each
  - Do a full pass of all batches N times (each pass is an **epoch**)
- For training evaluation, it is customary to plot the average loss for each epoch







Layers and Operations in ANNs



- **Fully connected layer**
  - All neurons connected
  - Function: full propagation of information between layers
- **Sparsely connected / dropout layer**
  - Few connections between neurons
  - Function: eliminating redundant information, preventing overfitting
- **Convolution layer**
  - Combines local mappings of data via windowed filtering
  - Function: can detect specific features in data (e.g., edges, textures, patterns)
- **Pooling layer**
  - Each kxk window in data is given a single value
  - Function: Downsamples/aggregates information between layers
- **Flatten layer**
  - Put data into a single 1-dimensional vector
  - Function: Reshape data, combine features
- **Softmax layer**
  - A nonlinear function typically used for mapping output to probabilities between 0-1
  - Typically used in the final layer

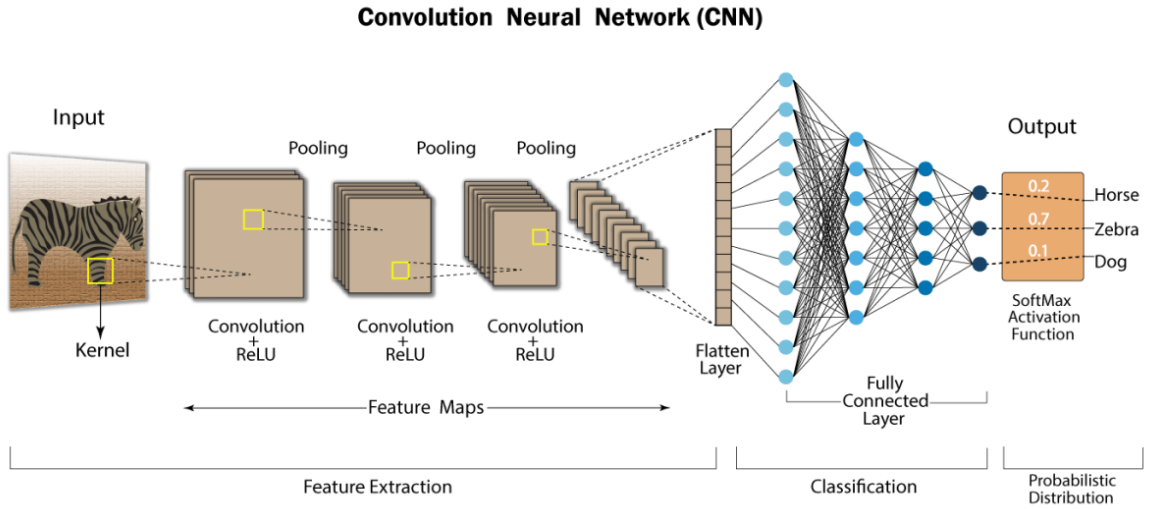


Diagram of a CNN, often used for image processing

Current Hot Topics

- Neural-Representational Alignment
  - How can we build models that align with the human brain?
- Behavioral Alignment
  - How can we build models that perform as well as humans in behavioral tasks (control, perception, memory, etc.)?
- Ethical AI
  - How can we build AI that are aligned with human morals?
- Efficiency
  - How can we build AI that uses fewer computational resources?
- Model Interpretability
  - How can we better understand the hidden layers (internal representations of ANNs)?

Jupyter Notebook Time!

- Today, we will return to our regression problem, but solve with an ANN!
- If you're returning, run **git pull** to update your local repository
- If you're new, clone the git repository for the course: https://github.com/orbitalhybridization/STARS_ML_MiniCourses