

Neural Networks

Shea Mowry, Meghan Pinter, Maya Bartels,
Felipe Munoz & Han Kahvecioglu

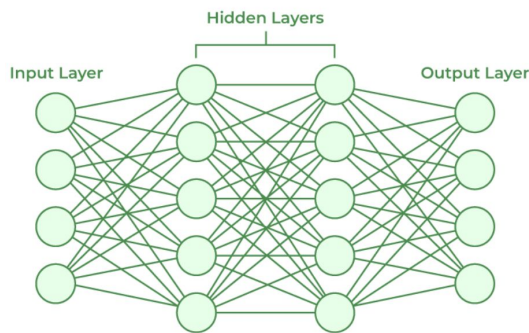
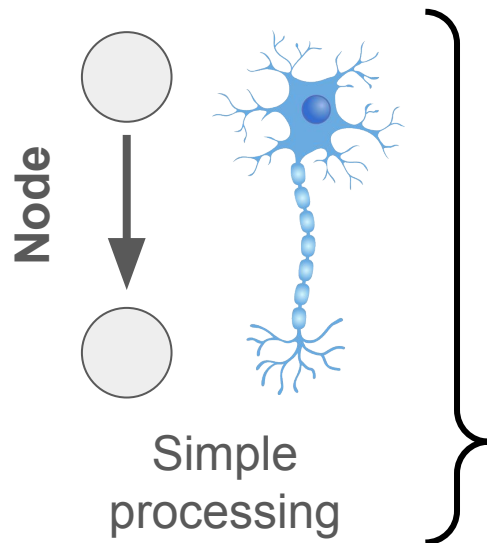
Topic	Who
Intro Neural networks	Han
Training NNs	Felipe
Intro regularization	Maya
Regularization techniques	Shea
Regularization Techniques LASSO and Elastic Net	Maya
MPWD Example	Meghan
Pros and cons (NN, Lasso, EN, Mult. Perceptron)	Shea, Han
Application NN	Felipe
Application Lasso/EN	Maya
Application MPWD	Meghan

Introduction to Neural Networks

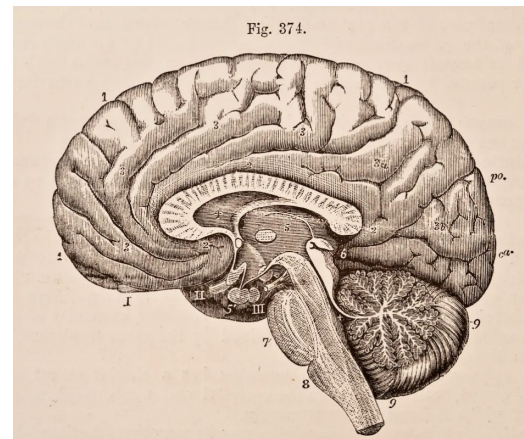
Neural networks

Models that mimic **how biological neurons work together to:**

- identify phenomena
- weigh options
- arrive at conclusions



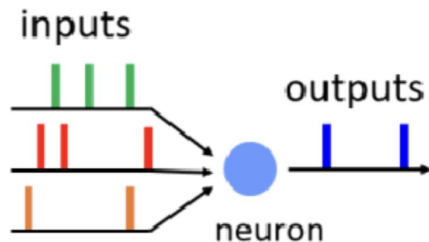
Organized in layers to conduct:



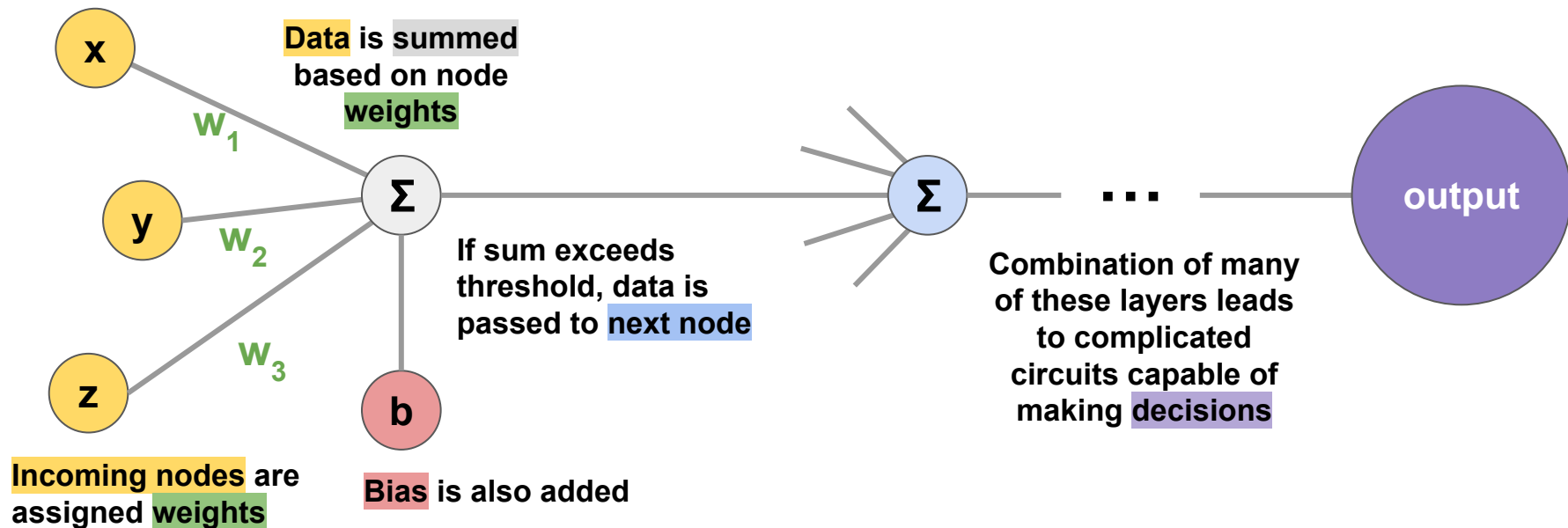
Complex processing

Neural networks: general working principle

Biologically, neurons **integrate** incoming signal and fire if the sum passes a **threshold**.



Neural nets use this principle as well:



Training Neural Networks

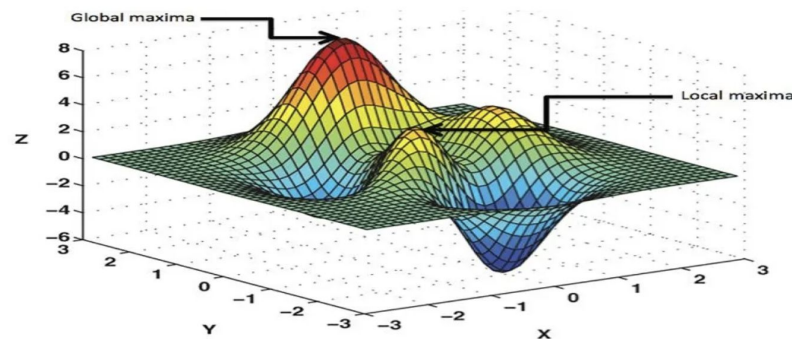
Loss function

- Measures distance between ground truth and model's prediction
- Function of model parameters

Examples of loss functions

- Classification: cross-entropy loss
- Regression: L1, L2, L_n loss

Training Objective → Minimize Loss



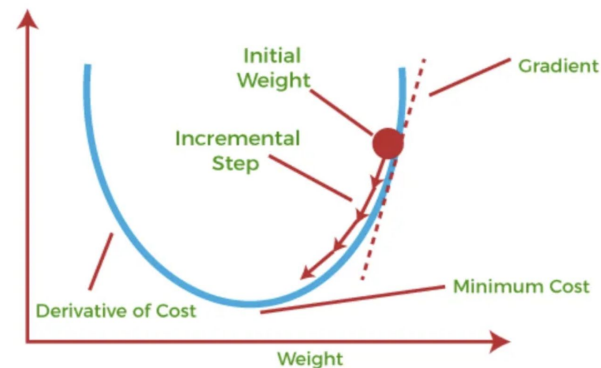
Picture depicting loss graph

source: Medium, 2023

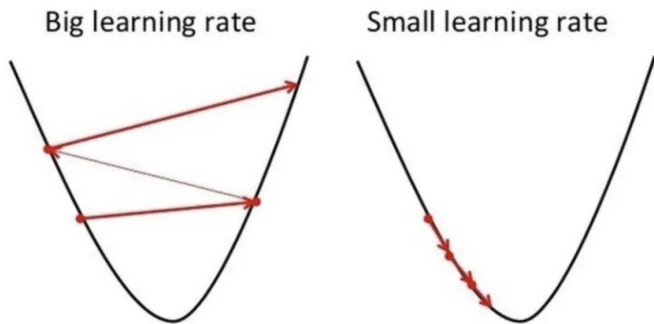
Training Neural Networks

How to minimize loss?

- Update weights: $\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta \mathbf{w}^{(\tau)}$
- Gradient descent: $\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)})$
 - At each time point, take a step in the direction that minimizes error function
 - Hyperparameter η (*eta*) - learning rate



source: Medium, 2024



source: Medium, 2024

η - Learning rate

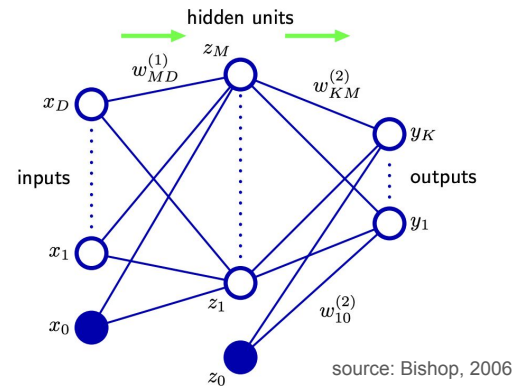
Too big \rightarrow model might miss minima

Too small \rightarrow longer training, model may fail to converge

Training Neural Networks

- Modern neural networks contain billions of parameters
- How to calculate gradient with respect to each one?

Answer: Backpropagation



Forward Pass

- Process example to produce output
- Use loss function to compare output to ground truth

Compute Gradient

- Compute gradient with respect to each parameter using chain rule

Update Weights

- Update weights using gradient information
- Variants of gradient descent
 - SGD
 - Adam

Iterate

- Repeat for n steps
- n is a hyperparameter

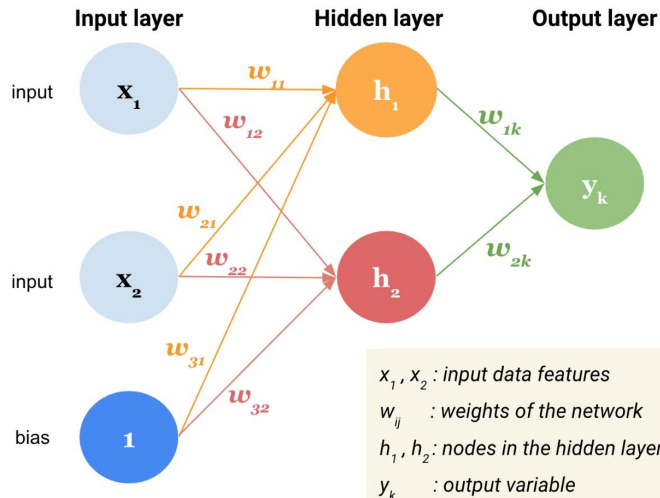
Multilayer Perceptron with Weight Decay - type of NN

Multilayer Perceptron (MLP)

- One input layer
- For each input there is one node
- Output layer has single node for each output
- Can have any number of hidden layers with any number of nodes

Weight Decay

- Regularization technique penalizing large weights
 - Avoids overfitting



$$L_{new}(w) = L_{original}(w) + \lambda w^T w$$

Intro to Regularization

1. Prevent overfitting
2. Make model more generalizable and make accurate predictions about new data

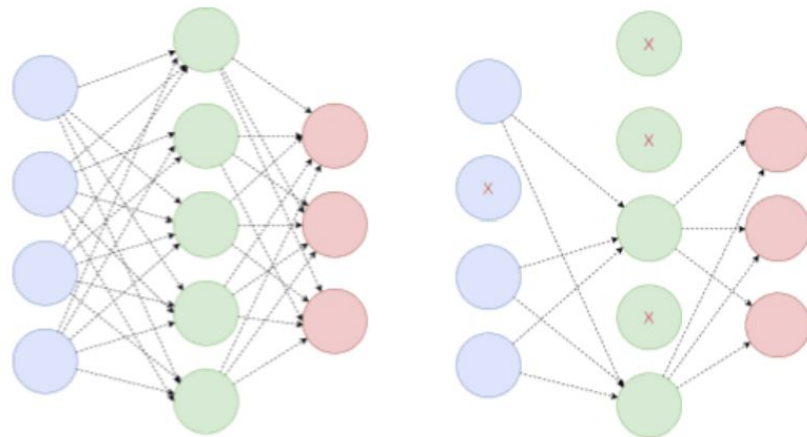
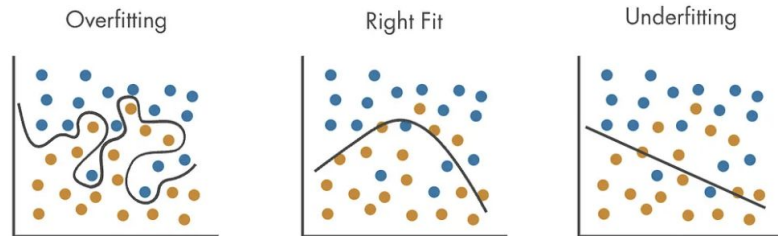
Penalize the cost function helps prevent overfitting by discouraging the model from becoming too complex.

Different regularization techniques have different penalization methods.

$$MAE = \frac{1}{n} \sum \left| y - \hat{y} \right|$$

Annotations for the MAE formula:

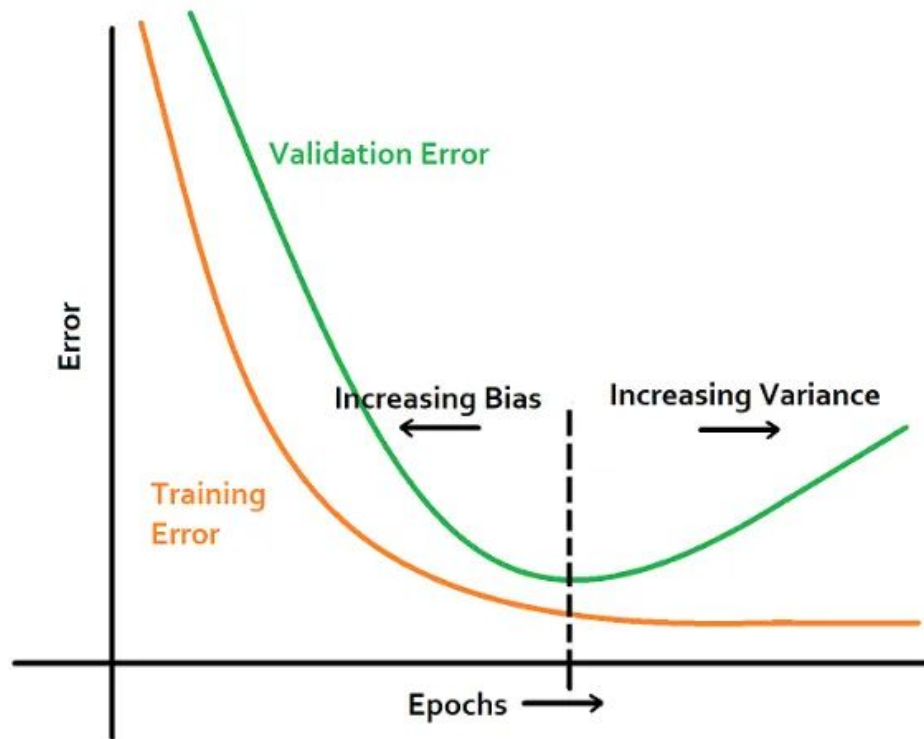
- $\frac{1}{n}$: Divide by the total number of data points
- \sum : Sum of
- y : Actual output value
- \hat{y} : Predicted output value
- $|y - \hat{y}|$: The absolute value of the residual



Regularization Techniques

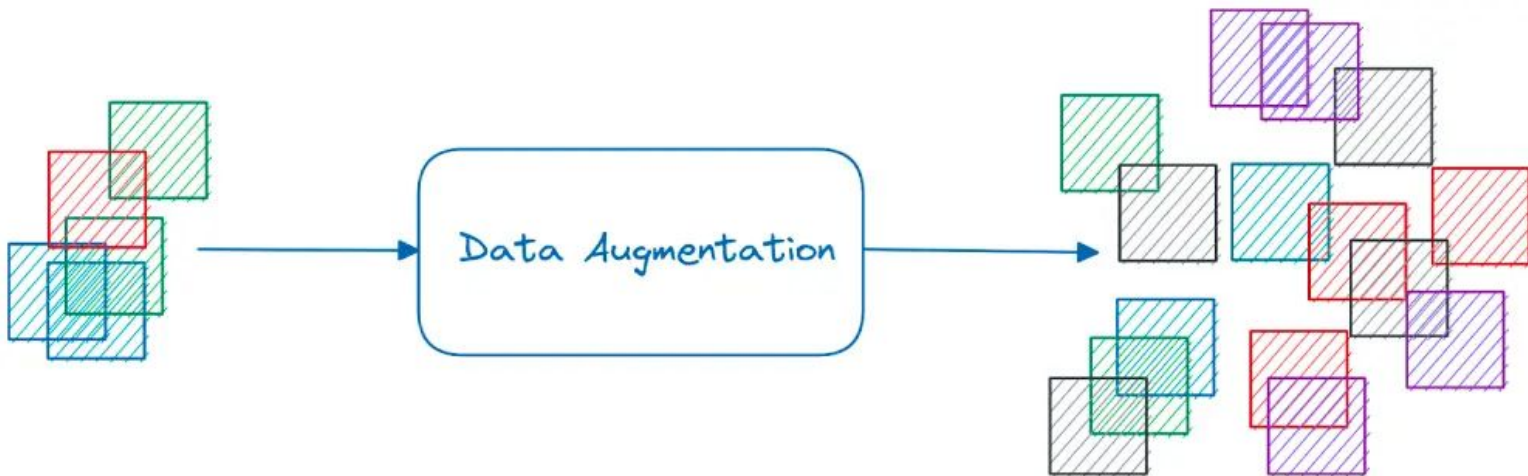
Early stopping

- Stops when the models starts performing poorly to avoid overfitting
- Determined by:
 - Training Error decreasing
 - Validation Error increasing
 - Or both

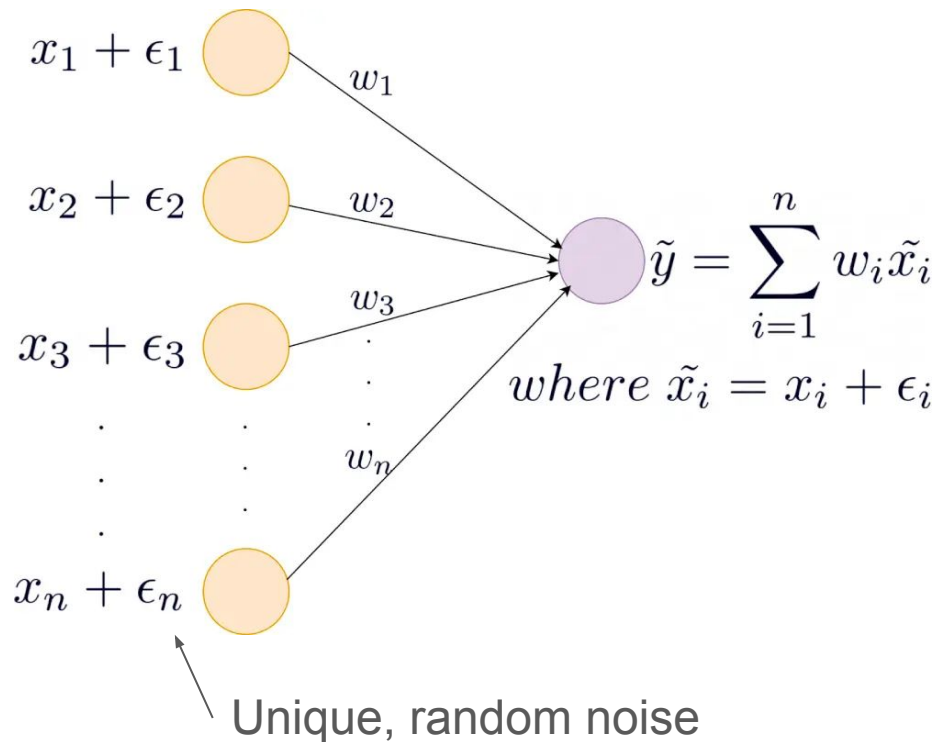


Data augmentation

- Create a more diverse data set from existing data via
 - Color space transformations such as change of pixel intensities
 - Rotation and mirroring
 - Noise injection, distortion, and blurring
 - Cutout, CutMix, and AugMix (removing or changing data)



Addition of Noise

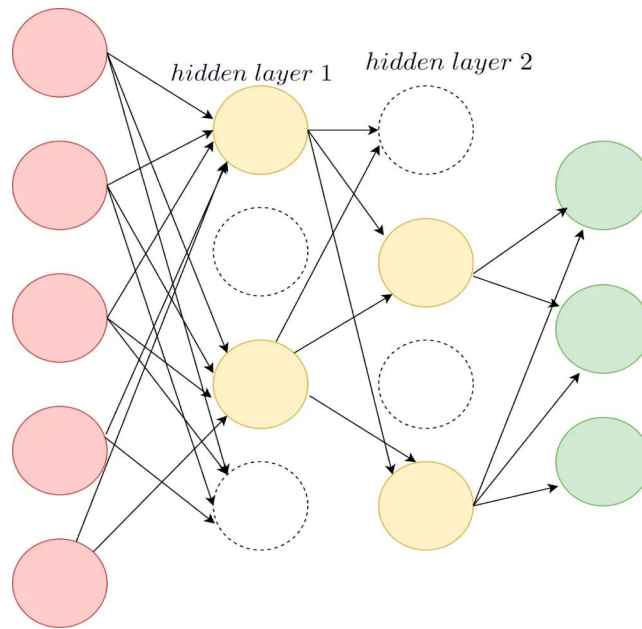
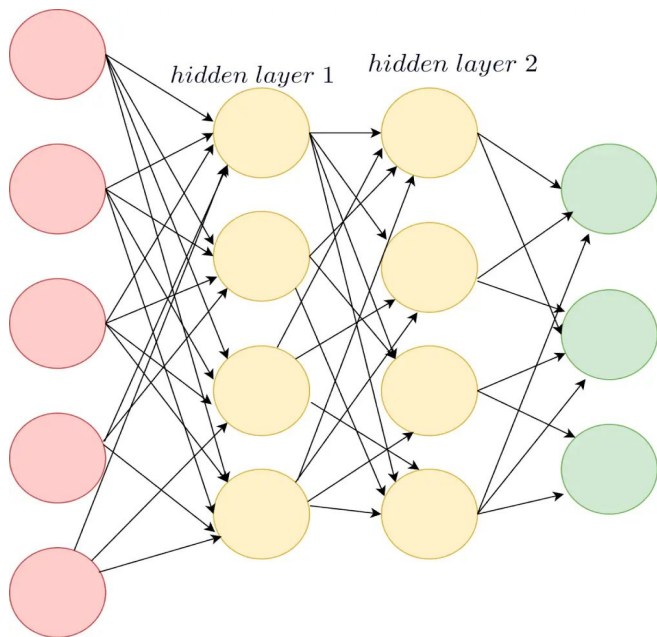


Noise can also be added to:

- Input data
- Output labels
- Gradients
- Weights

Dropout

Creates slightly different network architectures each time

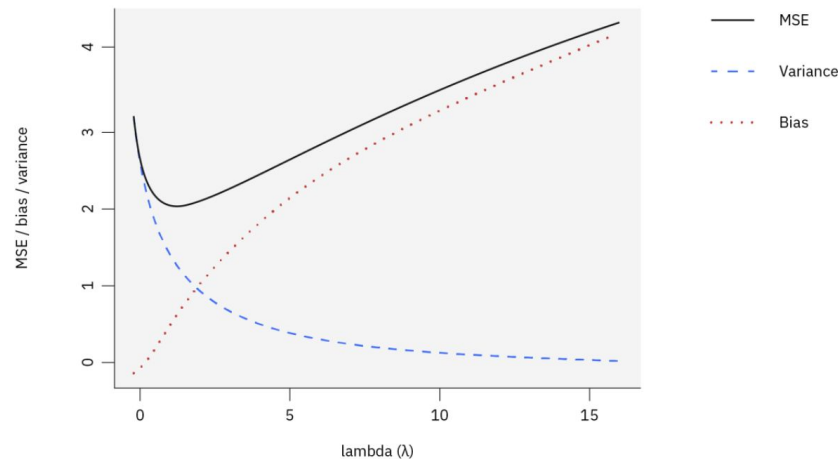


Dropping based on a “drop probability”

LASSO Regularization Technique

- Performs variable selection and regularization to enhance prediction accuracy and interpretability
 - Selects more relevant features from input features
 - shrinks some feature weights to 0 to remove them and only use relevant features
- Uses L1 penalty to reduce coefficients to 0
 - Choose a lambda value that balances bias and variance

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M |W_j|$$



Elastic Net Regularization Technique

- Combines L1 (Lasso) and L2 (Ridge) regularization techniques
 - Controls cost balance and penalizes complexity in the model
 - Uses alpha (regularization parameter)
 - $\alpha = 0 \rightarrow$ no L1 or L2 applied
 - $\alpha = 1 \rightarrow$ L1 and L2 are both applied
 - $0 < \alpha < 1 \rightarrow$ mixture of L1 and L2 are applied

$$ElasticNet = \sum_{i=1}^n (y_i - y(x_i))^2 + \alpha \sum_{j=1}^p |w_j| + \alpha \sum_{j=1}^p (w_j)^2$$

Regularization Terms

Neural Networks Pros & Cons

Strengths:

- + Useful to capture complex patterns (non-linear relationships)
- + Adaptability (can change and improve overtime as fed more data)
- + Tolerant to noise (useful for biological data)
- + Can be generalized to unseen data

Weaknesses:

- Requires large data set
- Computationally intensive
- Black box nature (unclear HOW)
- Prone to overfitting

Multilayer Perceptron with Weight Decay Pros & Cons

Strengths:

- + Reduces overfitting & improves generalization
- + Ability to work with non-linear problems
- + Prevents over-reliance on single data features

Weaknesses:

- Difficult to find right hyperparameter (weight decay coefficient λ)
- Increased computational complexity
- Does not account for bias

Comparison of regularization techniques



LASSO

Suitable for high-dimensional data
(chooses the most important data)
Not well-suited for datasets with
correlated features



Noise Addition

Prepared model for noisy real
world data (audio, image, etc.)
Longer time to converge to
solution



Elastic Net

Can handle highly correlated sets
(even collinearity)
High computational cost



Data augmentation

Good for when not enough data,
especially comp. vision tasks
Risk of generating irrelevant data



Early stopping

Prevents memorization
(better generalization)
Risk of premature stopping



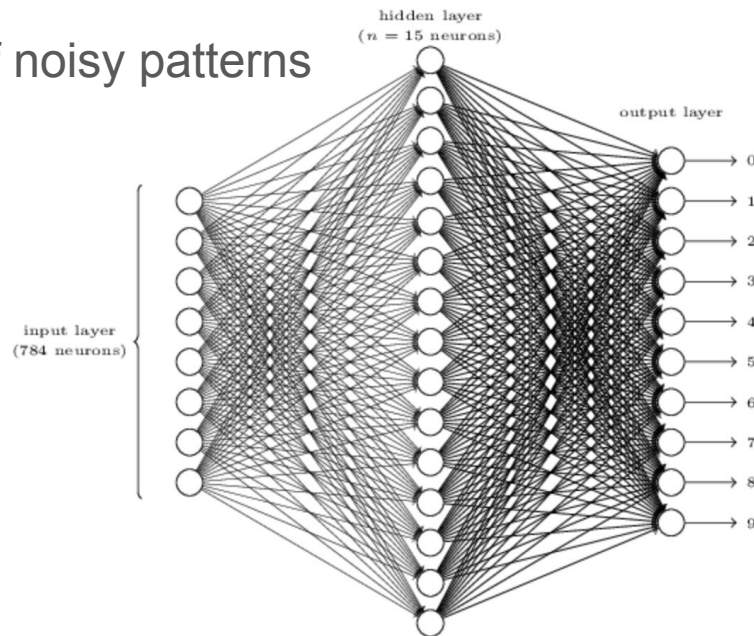
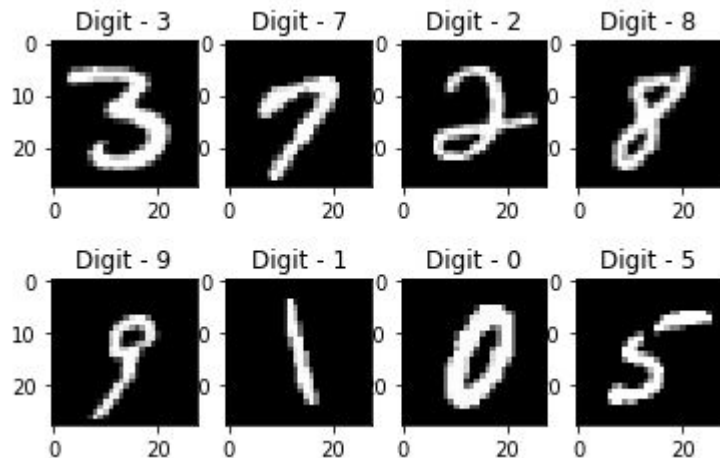
Dropout

Prevents too much relying of
neurons to one another
Incompatible with crucial layers

Application of MLP with Weight Decay

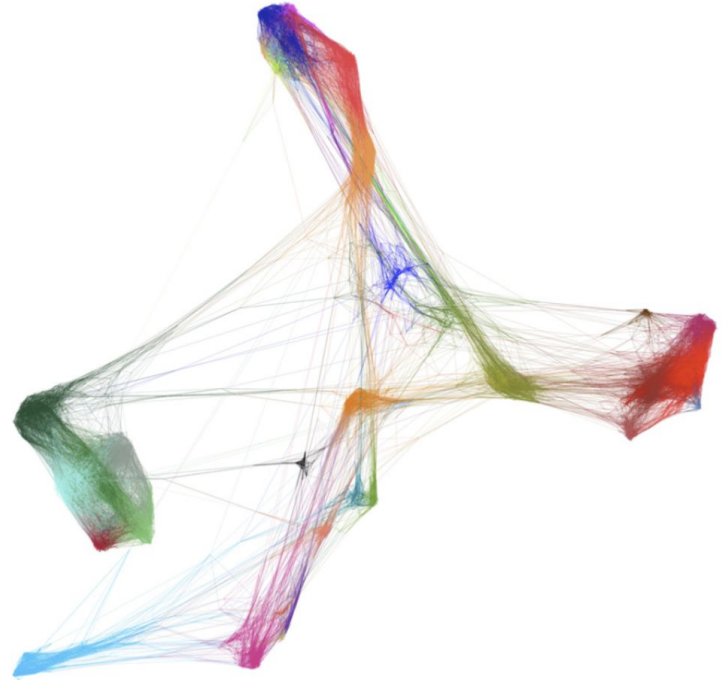
Handwritten digit recognition

- Handwritten digits have complex, non-linear patterns
 - Hidden layers can capture these patterns
- Weight decay helps prevent overfitting of noisy patterns



Applications of LASSO and Elastic Net

- Immunological data has high dimensionality
- Regularization techniques such as LASSO and Elastic Net can help prevent overfitting with high dimensional datasets
 - Penalizes cost function and makes the model more generalizable to new data
 - Removes unnecessary features from the model to reduce complexity
 - Less complex often means a better neural network algorithm
 - Makes model more interpretable for use on future datasets



Applications: Transformer

Transformer

- Captures relationships in sequential data (text)
- Foundation of modern AI models
 - GPT, BERT

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Lukasz Kaiser*
Google Brain
lukaszkaier@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

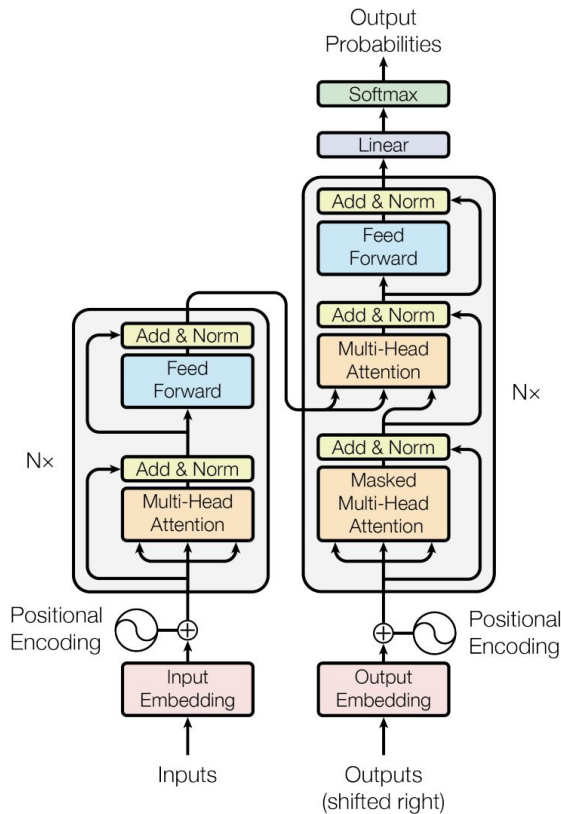


Figure 1: The Transformer - model architecture.



Applications: AlphaFold

Evoformer - inspired by transformer

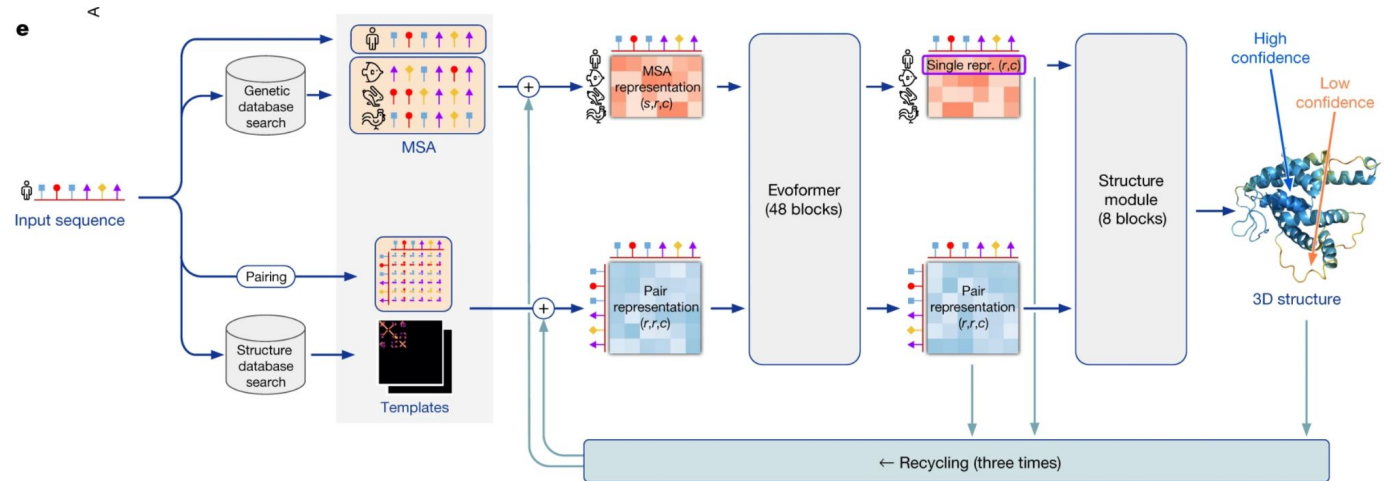
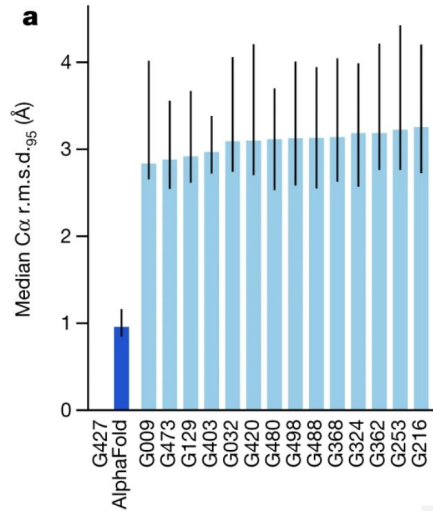
Other structure predictors - RoseTTAFold, ESM

Article | [Open access](#) | Published: 15 July 2021

Highly accurate protein structure prediction with AlphaFold

[John Jumper](#) , [Richard Evans](#), [Alexander Pritzel](#), [Tim Green](#), [Michael Figurnov](#), [Olaf Ronneberger](#), [Kathryn Tunyasuvunakool](#), [Russ Bates](#), [Augustin Žídek](#), [Anna Potapenko](#), [Alex Bridgland](#), [Clemens Meyer](#), [Simon A. A. Kohl](#), [Andrew J. Ballard](#), [Andrew Cowie](#), [Bernardino Romera-Paredes](#), [Stanislav Nikolov](#), [Rishub Jain](#), [Jonas Adler](#), [Trevor Back](#), [Stig Petersen](#), [David Reiman](#), [Ellen Clancy](#), [Michal Zielinski](#), ... [Demis Hassabis](#)  + Show authors

[Nature](#) **596**, 583–589 (2021) | [Cite this article](#)



Resources

Bourzac, K. (2017, April 14). Explained: Neural networks. *MIT News*. <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414>

IBM. (n.d.). Lasso regression. *IBM*.

<https://www.ibm.com/topics/lasso-regression#:~:text=In%20lasso%20regression%2C%20the%20hyperparameter.simpler%20model%20with%20fewer%20parameters>

Jain, A. (2020, May 19). Elastic net regression: Combined features of L1 and L2 regularization. *Medium*.

<https://medium.com/@abhishekjainindore24/elastic-net-regression-combined-features-of-l1-and-l2-regularization-6181a660c3a5>

Singh, P. (2019, October 15). L1 vs L2 regularization: Which is better? *Medium*

<https://medium.com/analytics-vidhya/l1-vs-l2-regularization-which-is-better-d01068e6658c>

AI Mind. (2020, February 18). Regularization techniques in Keras neural networks. *AI Mind*.

<https://pub.aimind.so/regularization-techniques-in-keras-neural-networks-2ec3ae48a764>

The AI Summer. (n.d.). Regularization techniques in machine learning. *The AI Summer*. <https://theaisummer.com/regularization/>

GeeksforGeeks. (2021, November 28). Multi-layer perceptron learning in TensorFlow. *GeeksforGeeks*.

<https://www.geeksforgeeks.org/multi-layer-perceptron-learning-in-tensorflow/>

D2L.ai. (n.d.). Chapter: Multi-Layer Perceptrons, Weight decay. In *Dive into Deep Learning*

https://classic.d2l.ai/chapter_multilayer-perceptrons/weight-decay.html

Brownlee, J. (2019, November 6). How to reduce overfitting in deep learning with weight regularization. *Machine Learning Mastery*.

<https://machinelearningmastery.com/how-to-reduce-overfitting-in-deep-learning-with-weight-regularization/>

Nielsen, M. (n.d.). Chapter 1: Introduction. In *Neural Networks and Deep Learning*. <http://neuralnetworksanddeeplearning.com/chap1.html>

Team 4 - Irun Cohen: Neural Networks and Regularization

- Members: Shea Mowry, Han Kahvecioglu, Maya Bartels, Felipe Munoz, Meghan Pinter
- Focus Algorithms: `nnet` (Neural Network), `mlpWeightDecay` (Multilayer Perceptron with Weight Decay)

What to do:

1. Introduce neural networks and regularization techniques.
2. Explain how Neural Networks and Multilayer Perceptron with Weight Decay work, their strengths, weaknesses, and key differences.