Neural networks

Faisal Z. Qureshi

http://vclab.science.ontariotechu.ca

Feed forward neural networks

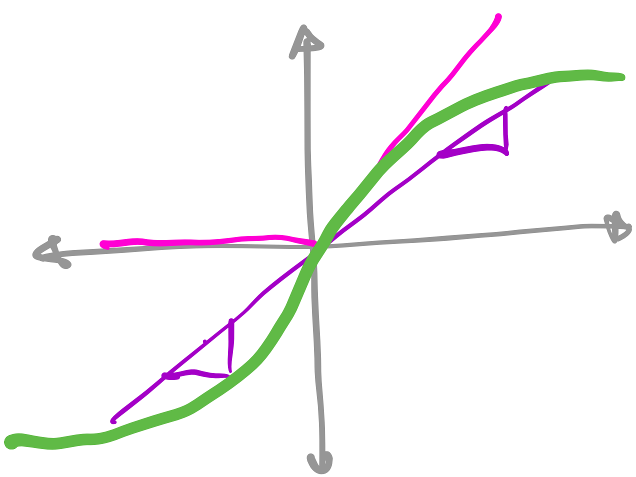
- Approximate some function $y = f^*(\mathbf{x})$ by learning parameters θ s.t. $\tilde{y} = f(\mathbf{x};\theta)$
- Feed forward neural networks can be seen as *directed acyclic graphs*

1

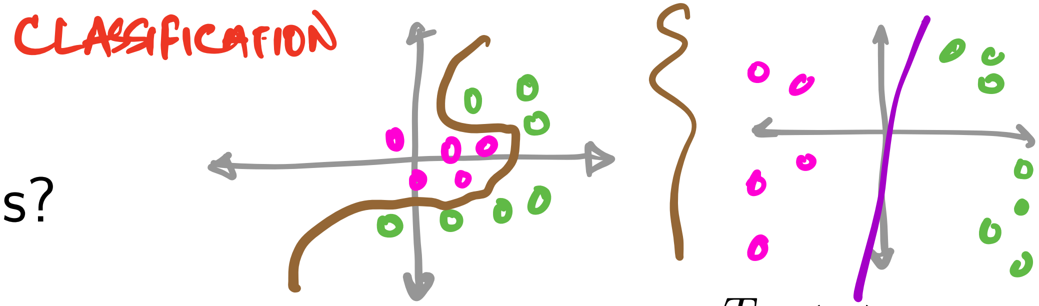
$$y = f(\mathbf{x}) = f^{(L)}(\cdots f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x}))))$$

- Training examples specify the output of the *last* layer
  - Network needs to figure out the inputs/outputs for the *hidden* layers

2

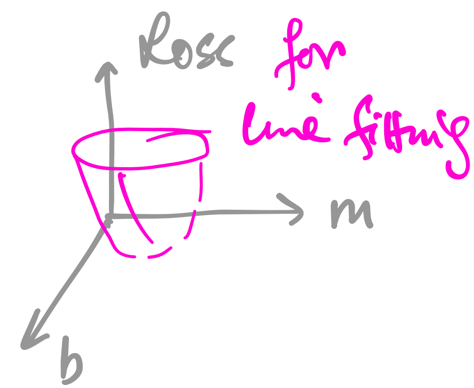


Extending linear models



How can we extend linear models?

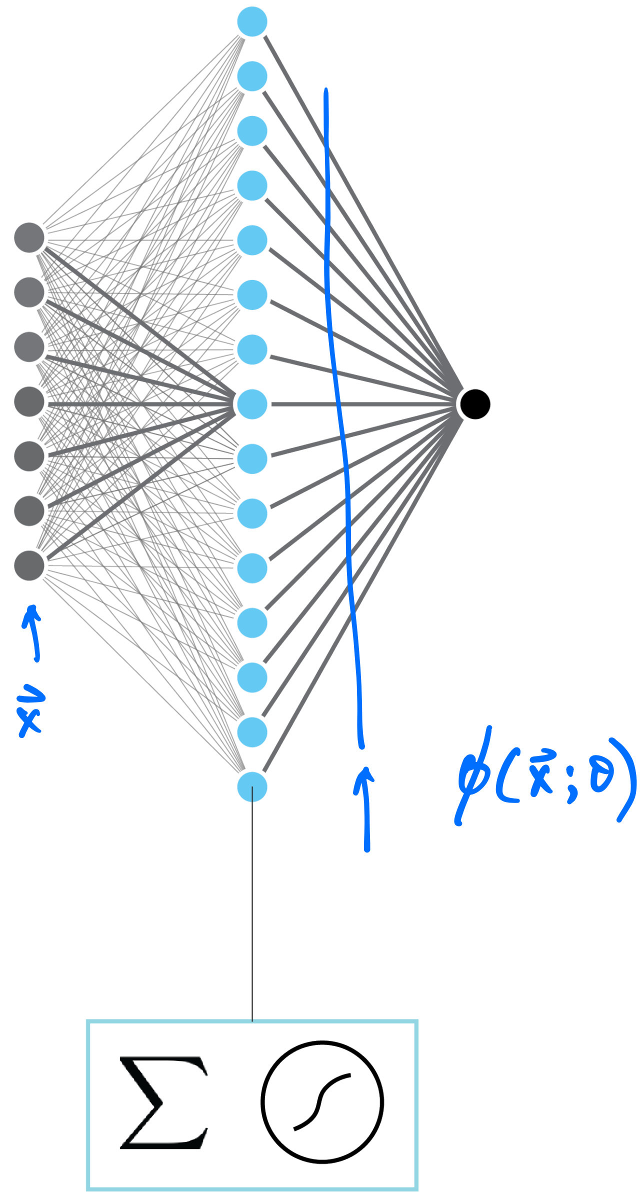
- Specify a very general ϕ s.t. the model becomes $y = \theta^T \phi(\mathbf{x})$
  - Problem with generalization
  - Difficult to encode *prior* information needed to solve AI-level tasks
- Engineer ϕ for the task at hand
  - Tedious
  - Difficult to transfer to new tasks
- Neural networks approaches
  - $y = f(\mathbf{x}; \theta, w) = \phi(\mathbf{x}; \theta)^T w$ i.e. use parameters θ to learn ϕ and use w to map $\phi(\mathbf{x})$ to the desired output y
  - The training problem is non-convex
  - Key advantage: a designer just need to specify the right family of functions and not the exact function ϕ



loss landscape is not quadratic

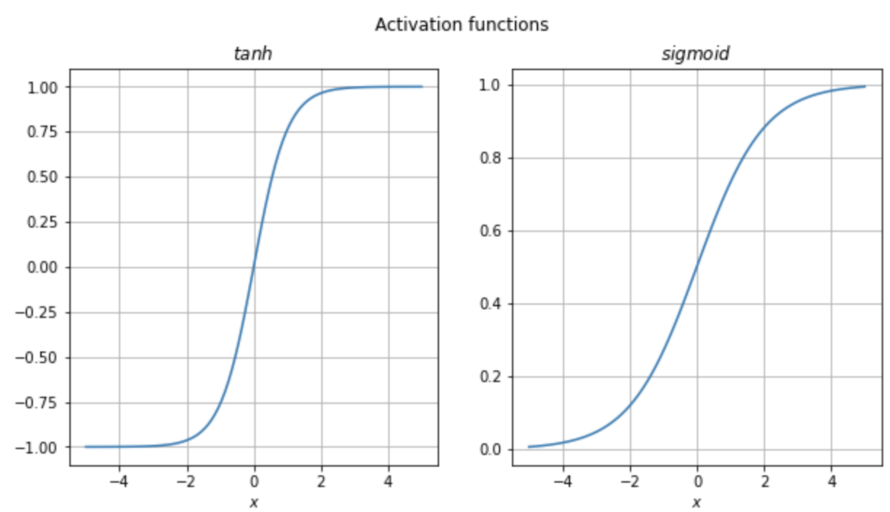
Classical artificial neural networks

- Shallow and wide
- One hidden layer can represent any function
- Focus was on efficient ways to optimize (train)



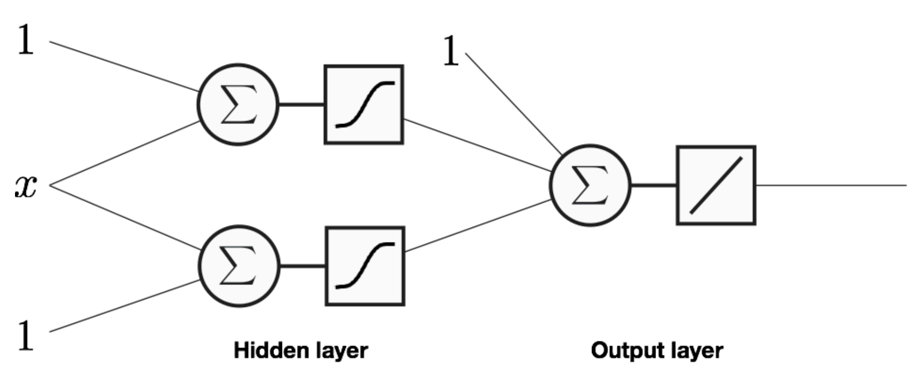
Activation functions (artificial neural networks)

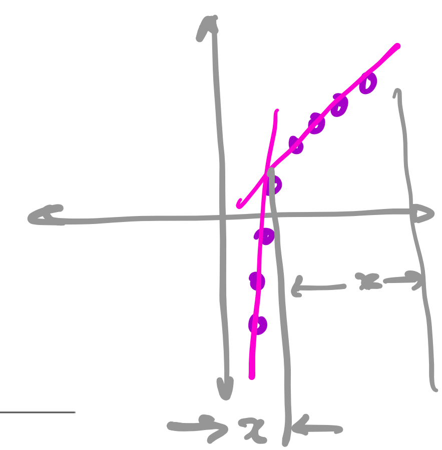
- The activation functions for output:
  - Identity function for regression;
  - Sigmoid for binary classification; and
  - Softmax for multi-class classification.
- The activation functions for hidden layers:
  - tanh (allows for negative output values); and
  - sigmoid.



Example: a regression network

- **Input**: 1D, real numbers
- **Ouput**: 1D, real numbers
- **Hidden layer size**: 2
- **Number of weights**: 7
- **Loss**: MSE
- Probabilistic view of line fitting

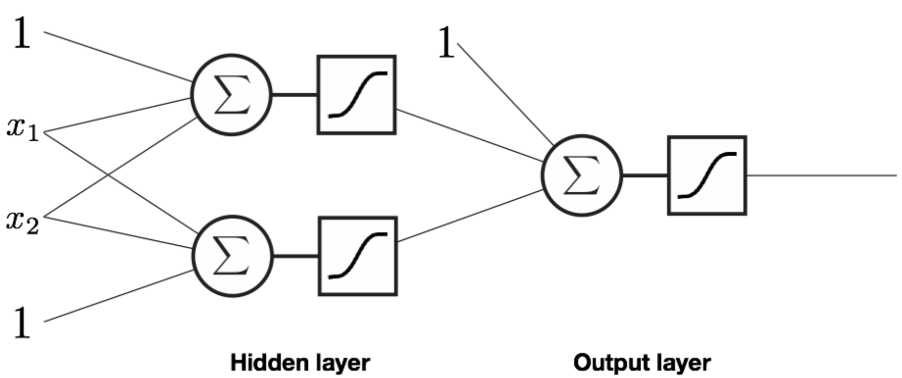




MSE

Example: a classification network

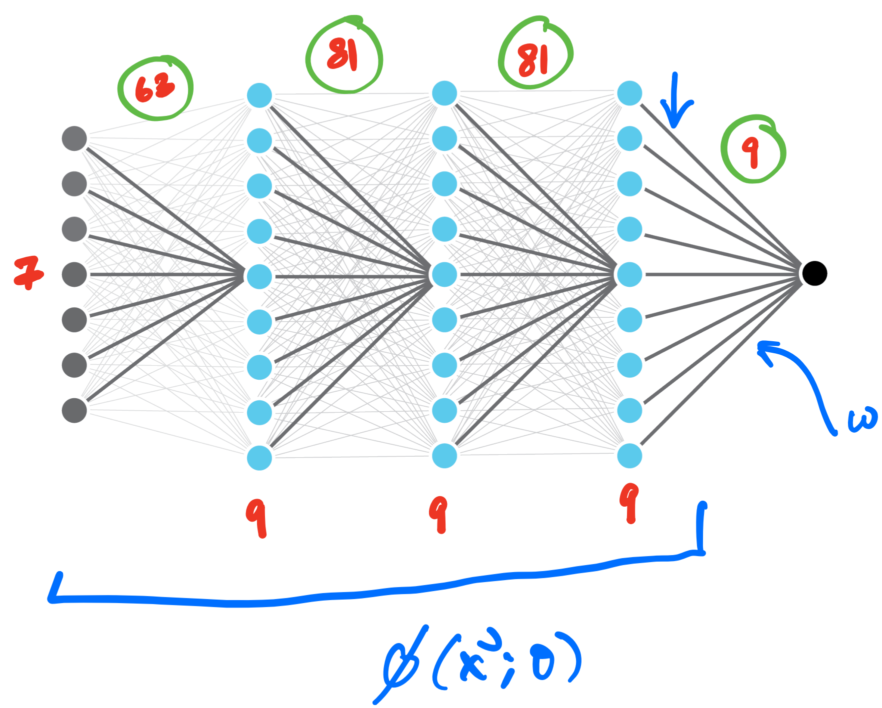
- **Input**: 2D, real numbers
- **Output**: 1D, class labels 0 or 1
- **Hidden layer size**: 2
- **Number of weights**: 9
- **Loss**: Cross-entropy
- Data likelihood under Bernoulli distribution



Cross-entropy

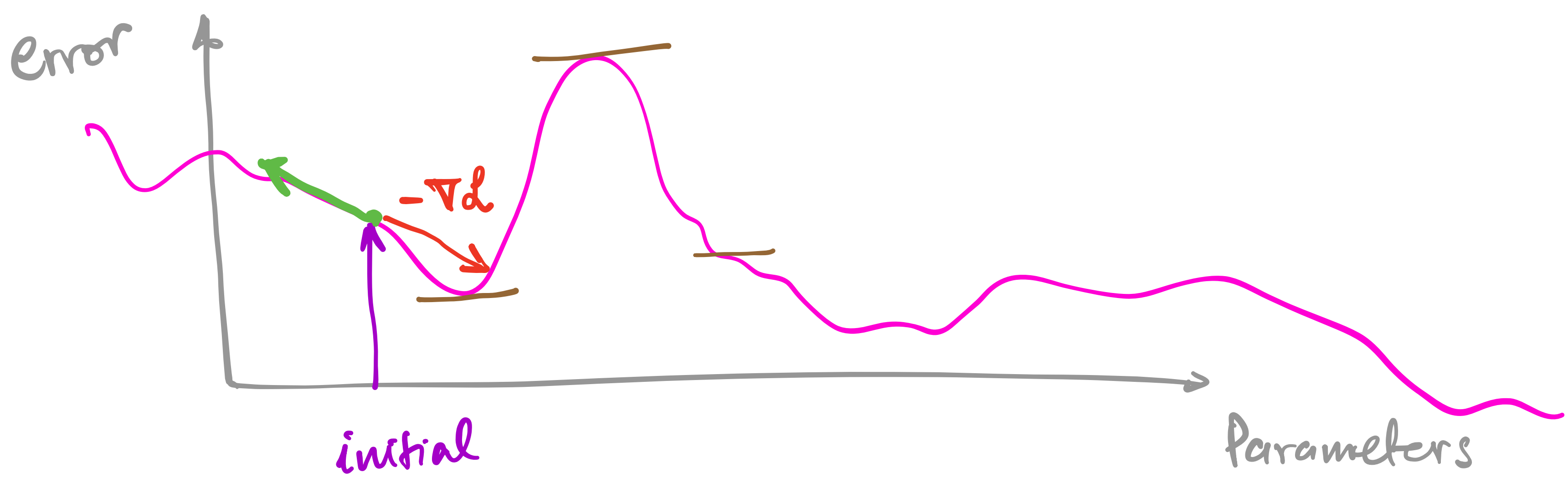
Current view – deep neural networks

- Multi-layer networks
  - These networks are deeper than these are wider
- Hierarchical representation
  - Reduces *semantic gap*
- Deep networks outperform humans on many tasks
- Access to data
- Advances in computer science, physics and engineering



Gradient-based learning in neural networks

- Non-linearities of neural networks render most cost functions non-convex
- Use iterative gradient based optimizers to drive cost function to lower values
- Gradient descent applied to non-convex cost functions has no guarantees is sensitive to initial conditions
  - Initialize weights to small random values
  - Initialize biases to zero or small positive values



Copyright and License