MANCHESTER
1824
The University of Manchester

COMP24112: Machine Learning

Chapter 7: Artificial Neural Network III

Dr. Tingting Mu

Email: tingting.mu@manchester.ac.uk

Content

- Backpropagation in artificial neural network.

Backpropagation

- Technically, backpropagation is a method of calculating the gradient of the loss function with respect to layers of the neural network weights.

- It uses the chain rule to iteratively compute the gradient for each layer.

$$\text{Given } z\left(y\left(x\right)\right), \text{ chain rule: } \frac{dz}{dx} = \frac{dz}{dy} \times \frac{dy}{dx}$$

- It can be viewed as a process of calculating the error contribution of each neuron after processing a batch of training data.

Loss and Regularisation

- After choosing a loss function ***Loss*(*W_{NN}*)**, a regularisation term, e.g., the sum of the all the squared weights, is added to the final optimisation objective function for training:

$$O\left(\mathbf{W}_{NN}\right) = loss\left(\mathbf{W}_{NN}\right) + \lambda \frac{1}{2}\left\|\mathbf{W}_{NN}\right\|_2^2$$

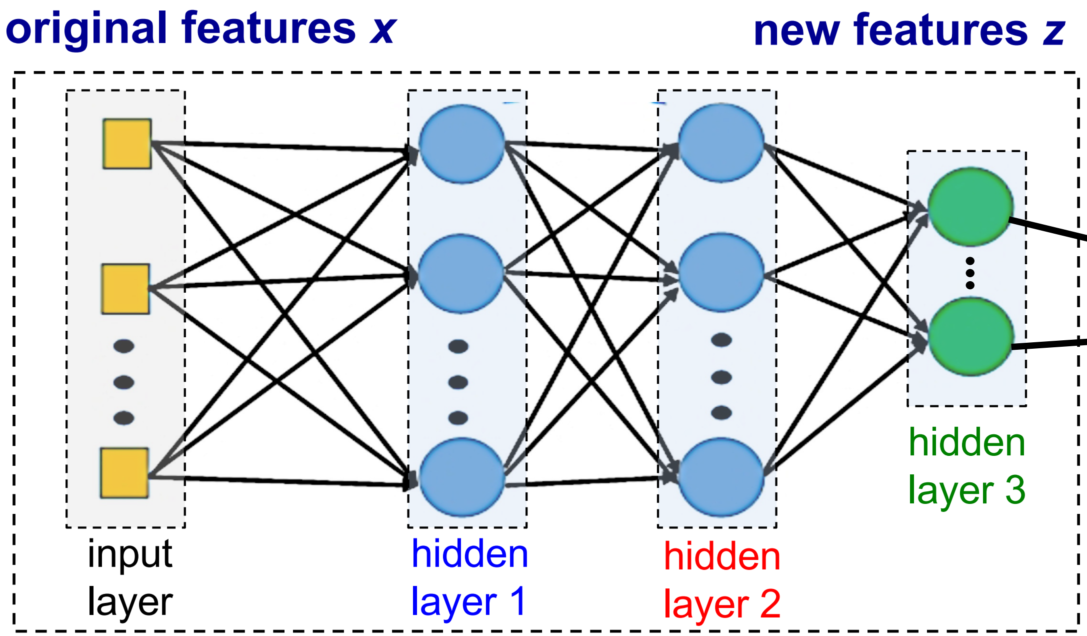
- This is called l_2-regularisation. The regularisation term plays the same role as in the regularised least squares model.

- The regularisation parameter $\lambda \geq 0$ is a hyper-parameter.

Backpropagation Illustration

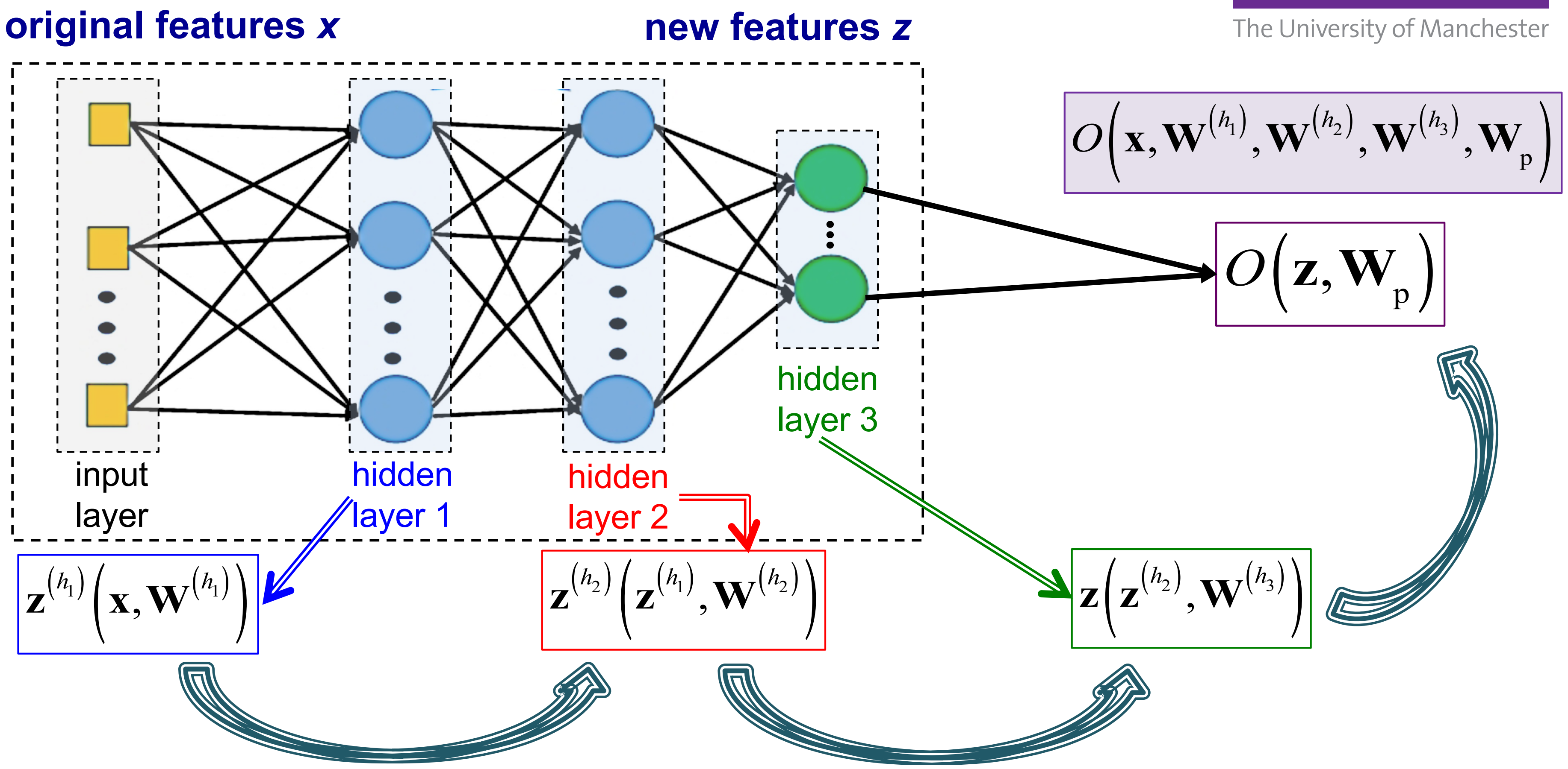
$$O\left(\mathbf{x}, \mathbf{W}^{(h_1)}, \mathbf{W}^{(h_2)}, \mathbf{W}^{(h_3)}, \mathbf{W}_{\mathrm{p}}\right)$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_1)}}, \frac{\partial O}{\partial \mathbf{W}^{(h_2)}}, \frac{\partial O}{\partial \mathbf{W}^{(h_3)}}, \frac{\partial O}{\partial \mathbf{W}_{\mathrm{p}}} = ?$$



$$O\left(\mathbf{z}, \mathbf{W}_{\mathrm{p}}\right)$$

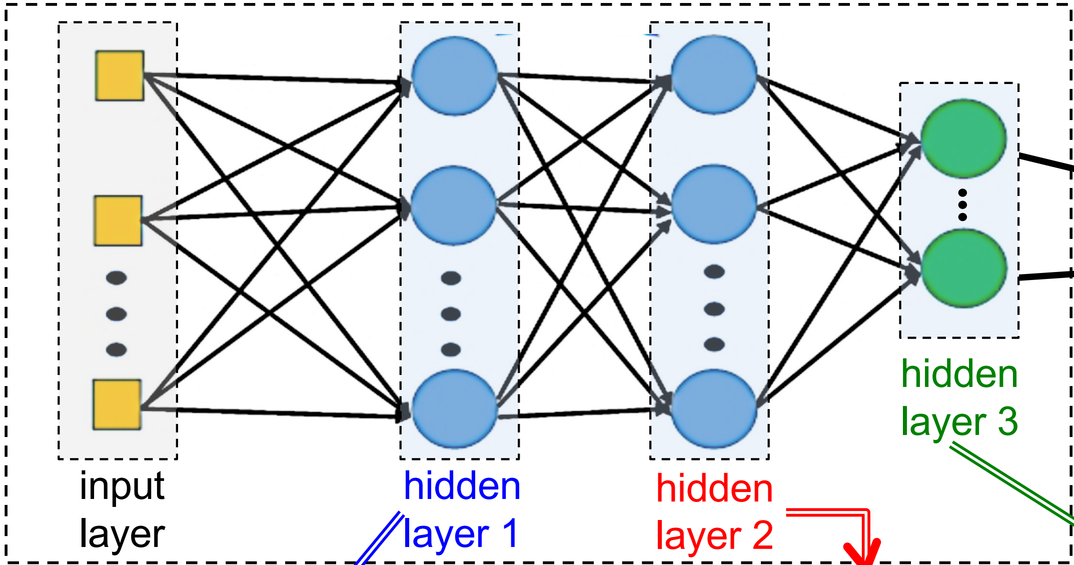
$$\mathbf{z} = \varphi\left(\mathbf{x}, \mathbf{W}^{(h_1)}, \mathbf{W}^{(h_2)}, \mathbf{W}^{(h_3)}\right)$$



Feedforward loss calculation!

MANCHESTER 1824
The University of Manchester

original features x **new features z**



$$\frac{\partial O}{\partial \mathbf{W}^{(h_1)}}, \frac{\partial O}{\partial \mathbf{W}^{(h_2)}}, \frac{\partial O}{\partial \mathbf{W}^{(h_3)}}, \frac{\partial O}{\partial \mathbf{W}_{\mathrm{p}}} = ?$$

$$O\left(\mathbf{z}, \mathbf{W}_{\mathrm{p}}\right)$$

$$\frac{\partial O}{\partial \mathbf{W}_{\mathrm{p}}}$$

$$\mathbf{z}^{(h_1)}\left(\mathbf{x}, \mathbf{W}^{(h_1)}\right)$$

$$\mathbf{z}^{(h_2)}\left(\mathbf{z}^{(h_1)}, \mathbf{W}^{(h_2)}\right)$$

$$\mathbf{z}\left(\mathbf{z}^{(h_2)}, \mathbf{W}^{(h_3)}\right)$$

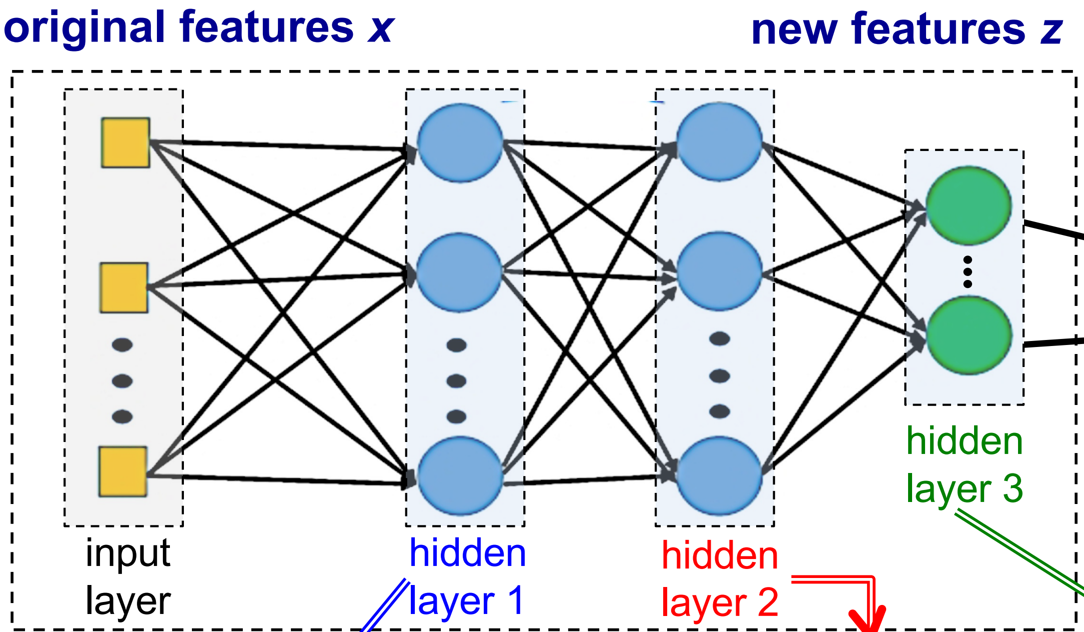
$$\frac{\partial O}{\partial \mathbf{W}^{(h_1)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{z}^{(h_1)}} \times \frac{\partial \mathbf{z}^{(h_1)}}{\partial \mathbf{W}^{(h_1)}}$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_2)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{W}^{(h_2)}}$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_3)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(h_3)}}$$

Backward gradient calculation!

MANCHESTER 1824
The University of Manchester



$$O\left(\mathbf{z},\mathbf{W}_{\mathrm{p}}\right)$$

$$\frac{\partial O}{\partial \mathbf{z}} \qquad \frac{\partial O}{\partial \mathbf{W}_{\mathrm{p}}}$$

$$\mathbf{z}^{(h_1)}\left(\mathbf{x},\mathbf{W}^{(h_1)}\right) \qquad \mathbf{z}^{(h_2)}\left(\mathbf{z}^{(h_1)},\mathbf{W}^{(h_2)}\right) \qquad \mathbf{z}\left(\mathbf{z}^{(h_2)},\mathbf{W}^{(h_3)}\right)$$

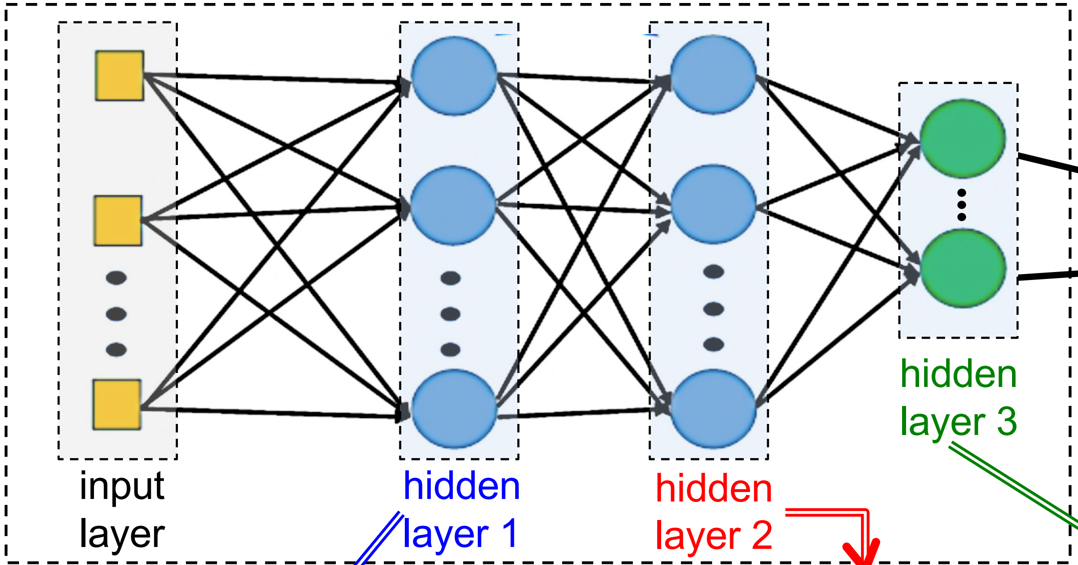
$$\frac{\partial O}{\partial \mathbf{W}^{(h_1)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{z}^{(h_1)}} \times \frac{\partial \mathbf{z}^{(h_1)}}{\partial \mathbf{W}^{(h_1)}}$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_2)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{W}^{(h_2)}}$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_3)}} = \boxed{\frac{\partial O}{\partial \mathbf{z}}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(h_3)}}$$

Backward gradient calculation!

original features *x* **new features *z***



$$O\left(\mathbf{z}, \mathbf{W}_{\mathrm{p}}\right)$$

$$\boxed{\frac{\partial O}{\partial \mathbf{z}}} \qquad \boxed{\frac{\partial O}{\partial \mathbf{W}_{\mathrm{p}}}}$$

$$\mathbf{z}^{(h_1)}\left(\mathbf{x}, \mathbf{W}^{(h_1)}\right) \qquad \mathbf{z}^{(h_2)}\left(\mathbf{z}^{(h_1)}, \mathbf{W}^{(h_2)}\right) \qquad \mathbf{z}\left(\mathbf{z}^{(h_2)}, \mathbf{W}^{(h_3)}\right)$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_1)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{z}^{(h_1)}} \times \frac{\partial \mathbf{z}^{(h_1)}}{\partial \mathbf{W}^{(h_1)}}$$

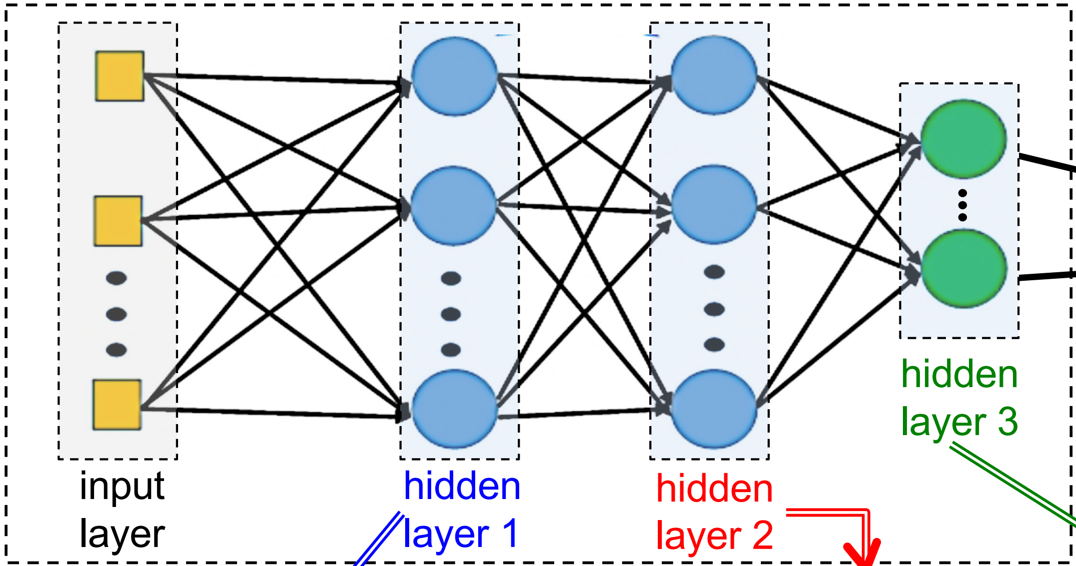
$$\frac{\partial O}{\partial \mathbf{W}^{(h_2)}} = \boxed{\frac{\partial O}{\partial \mathbf{z}}} \times \boxed{\frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{W}^{(h_2)}}$$

$$\boxed{\frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}}}$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_3)}} = \boxed{\frac{\partial O}{\partial \mathbf{z}}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(h_3)}}$$

Backward gradient calculation!

original features x → **new features z**



$$O\left(\mathbf{z}, \mathbf{W}_{\mathrm{p}}\right)$$

$$\frac{\partial O}{\partial \mathbf{z}} \qquad \frac{\partial O}{\partial \mathbf{W}_{\mathrm{p}}}$$

$$\mathbf{z}\left(\mathbf{z}^{(h_2)}, \mathbf{W}^{(h_3)}\right)$$

$$\frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \qquad \frac{\partial O}{\partial \mathbf{W}^{(h_3)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(h_3)}}$$

$$\mathbf{z}^{(h_2)}\left(\mathbf{z}^{(h_1)}, \mathbf{W}^{(h_2)}\right)$$

$$\frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{z}^{(h_1)}} \qquad \frac{\partial O}{\partial \mathbf{W}^{(h_2)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{W}^{(h_2)}}$$

$$\mathbf{z}^{(h_1)}\left(\mathbf{x}, \mathbf{W}^{(h_1)}\right)$$

$$\frac{\partial O}{\partial \mathbf{W}^{(h_1)}} = \frac{\partial O}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{z}^{(h_2)}} \times \frac{\partial \mathbf{z}^{(h_2)}}{\partial \mathbf{z}^{(h_1)}} \times \frac{\partial \mathbf{z}^{(h_1)}}{\partial \mathbf{W}^{(h_1)}}$$

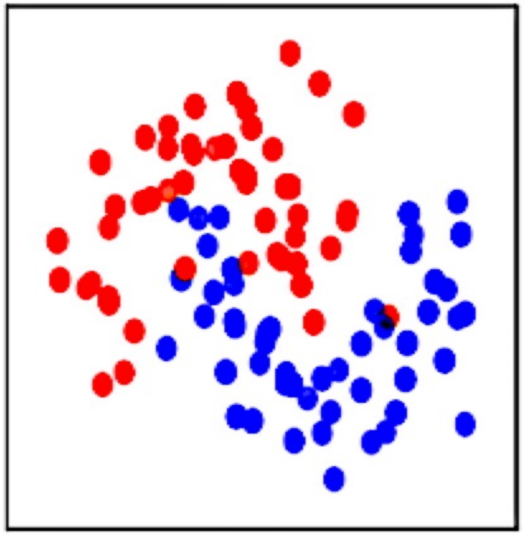
Backward gradient calculation!

backpropagation

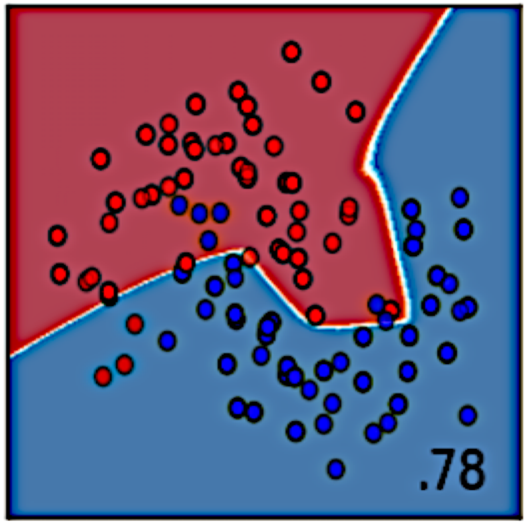
Example: Binary Classification by MLP

- Train an MLP for binary classification using cross-entropy loss with l_2 regularisation and stochastic gradient descent.
- The demonstrated examples are from scikit-learn https://scikit-learn.org/stable/modules/neural_networks_supervised.html#classification

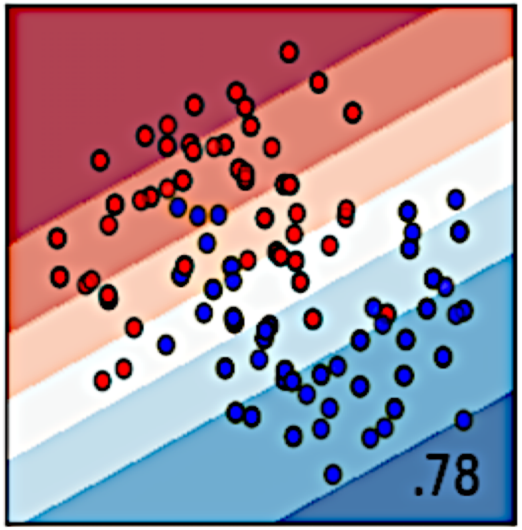
Data



MLP, $\lambda = 0.001$



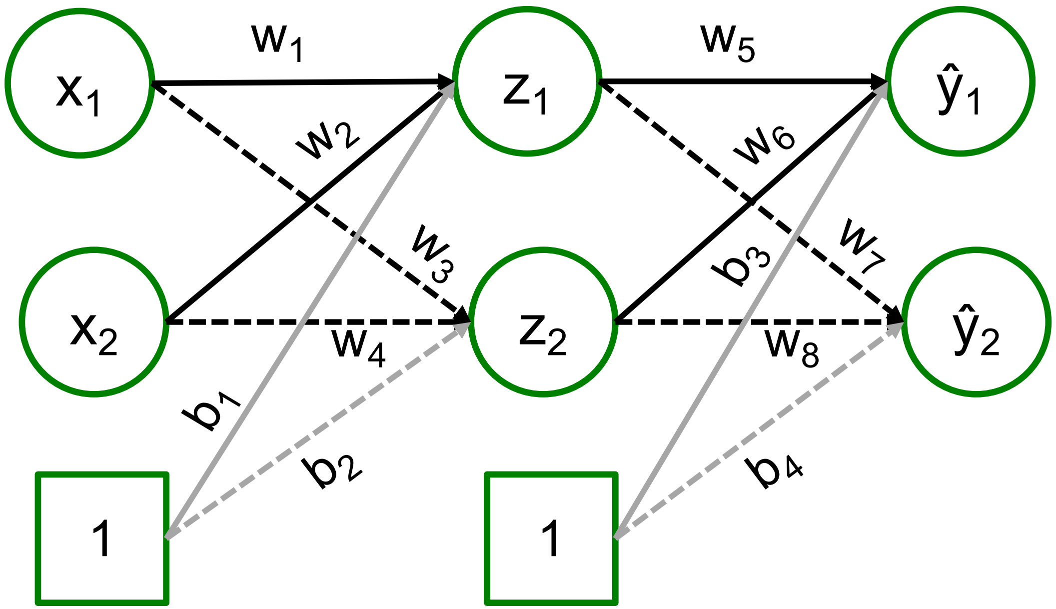
MLP, $\lambda = 10$

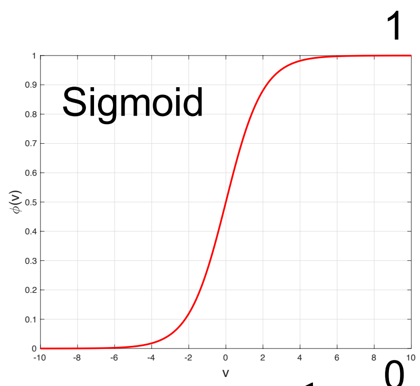


Too high regularisation parameter, under-fitting.

Example

- Focus on an MLP with 2 input and 2 output, 1 hidden layer with 2 hidden neurons, with sigmoid activation.

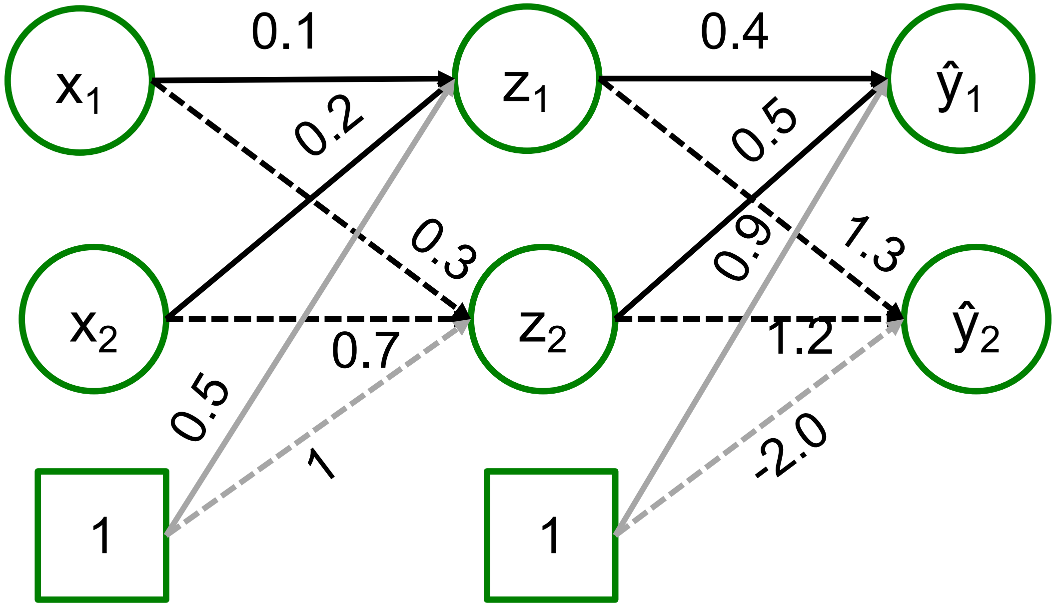


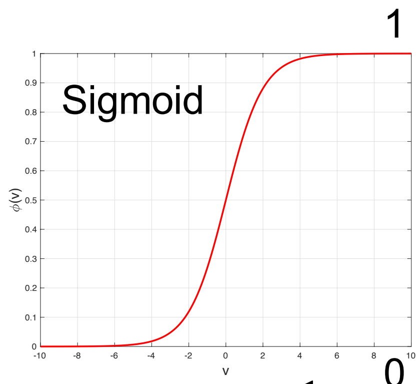


$$\varphi(v) = \frac{1}{1+\exp(-v)}$$

Example

- Initialise the network with some random weights, and train it using stochastic gradient descent.





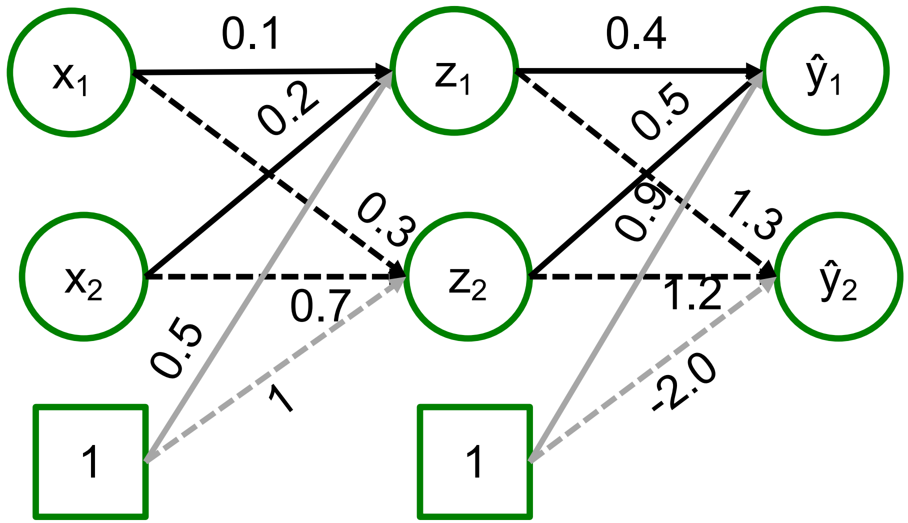
$$\varphi(v) = \frac{1}{1+\exp(-v)}$$

Example

- Update w_2 using one training sample:

$$\frac{\partial O}{\partial w_2} = \left[\hat{y}_1 w_5 \left(\hat{y}_1 - y_1\right) \left(1 - \hat{y}_1\right) + \hat{y}_2 w_7 \left(\hat{y}_2 - y_2\right) \left(1 - \hat{y}_2\right)\right] z_1(1 - z_1) x_2$$

$$w_2^{(new)} = w_2^{(old)} - \eta \frac{\partial O}{\partial w_2}$$



For this example neural network, practice the following:

I. For a training example with input (x_1=1, x_2=2) and output (y_1 =0.1, y_2=0.8), calculate its squared error.

II. Figure out the update equation for w_2 using this training example.

See Chapter 7 Practice and its solution on weight updating using a specific training sample.

Summary

- Neural network architecture
  - Single neuron model
  - Single layer perceptron
  - Multi layer perceptron
- Neural network training
  - Hebbian learning rules
  - Gradient based training
- Neural network regularisation
- Example training algorithms
  - The perceptron algorithm
  - Backpropagation