11. Artificial neural networks and Clustering

Shabana K M
PhD Research Scholar
Computer Science and Engineering

IIT Palakkad

5 December 2021

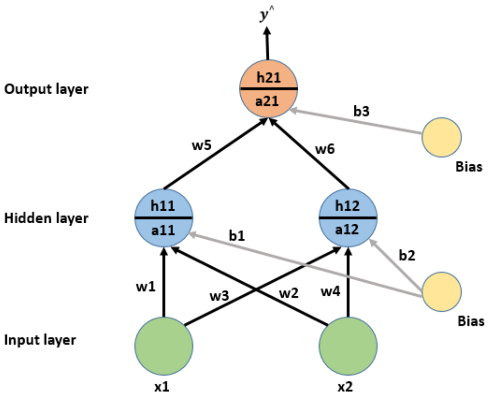
INDIAN INSTITUTE OF TECHNOLOGY PALAKKAD

IEEE STUDENT BRANCH GEC PALAKKAD | IEEE LINK | INTERNZONE | IEEE Kerala Section | IEEE

Recap

- Support vector machines
  - soft margin svm
  - the kernel trick
- Boosting
  - AdaBoost
  - Gradient Boosting
- Artificial neural networks
  - layer organization
  - representation power

Feed-forward computation



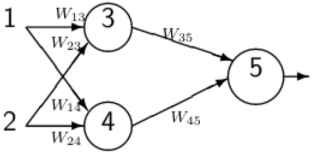
$$a11 = w1 * x1 + w2 * x2 + b1$$
$$a12 = w3 * x1 + w4 * x2 + b2$$

$$h11 = g(a11)$$
$$h12 = g(a12)$$

$$a21 = w5 * h11 + w6 * h12 + b3$$
$$h21 = g(a21)$$

$$\hat{y} = h21$$

Feed-forward computation:Example



$w_{13} = 2$ $w_{23} = -3$	$w_{35} = 2$
$w_{14} = 1$ $w_{24} = 4$	$w_{45} = -1$

$$f(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

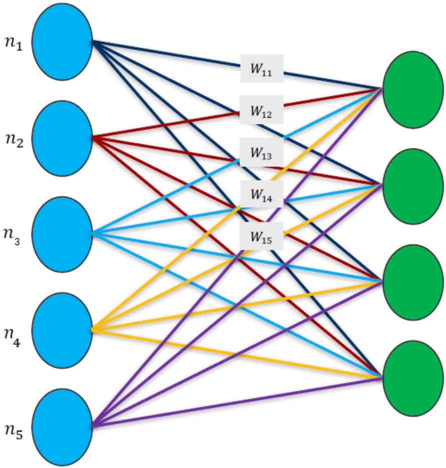
What is the network output if the inputs are $x_1 = 1$ and $x_2 = 0$?

$$a_3 = w_{13} * x1 + w23 * x2$$
$$a_3 = 2 * 1 + -3 * 0 = 2$$
$$a_4 = 1 * 1 + 4 * 0 = 1$$

$$h_3 = f(a_3) = 1$$
$$h_4 = f(a_4) = 1$$

$$a_5 = w_{35} * h_3 + w_{45} * h_4$$
$$a_5 = 2 * 1 + -1 * 1 = 1$$
$$y = f(a_5) = 1$$

Feed-forward computation



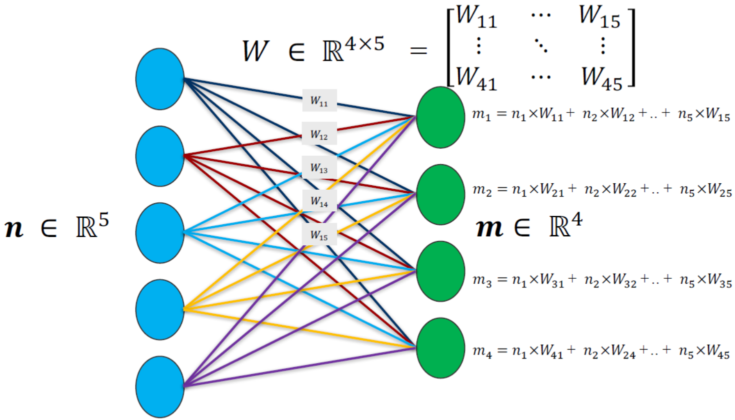
$$m_1 = n_1 \times W_{11} + n_2 \times W_{12} + .. + n_5 \times W_{15}$$

$$m_2 = n_1 \times W_{21} + n_2 \times W_{22} + .. + n_5 \times W_{25}$$

$$m_3 = n_1 \times W_{31} + n_2 \times W_{32} + .. + n_5 \times W_{35}$$

$$m_4 = n_1 \times W_{41} + n_2 \times W_{24} + .. + n_5 \times W_{45}$$

Feed-forward computation

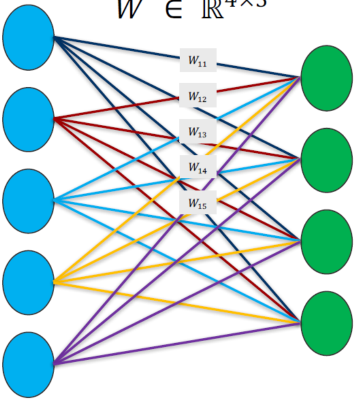


- The size of the `weight` matrix would be m x n

Feed-forward computation

$W \in \mathbb{R}^{4\times 5}$

$\boldsymbol{n} \in \mathbb{R}^5$



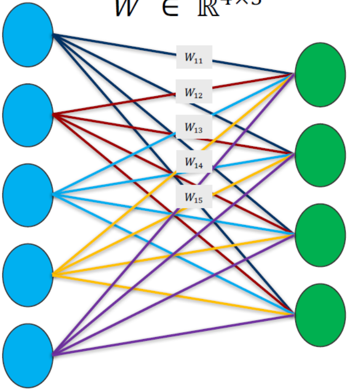
$\boldsymbol{m} \in \mathbb{R}^4$

$$\boxed{\boldsymbol{m} = W^{4\times 5} \cdot \boldsymbol{n}} + \boldsymbol{b}^{4\times 1}$$

Feed-forward computation

$W \in \mathbb{R}^{4\times5}$

$\boldsymbol{n} \in \mathbb{R}^5$



$\boldsymbol{m} \in \mathbb{R}^4$

$$\boldsymbol{m} = W^{4\times5} \cdot \boldsymbol{n} + \boldsymbol{b}^{4\times1}$$

$$\boxed{\boldsymbol{h} = \mathrm{g}(\boldsymbol{m})}$$

Training neural networks: Key Idea

- Find weights:

$$w^* = \arg\min_{w} \sum_{i=1}^{N} loss(\hat{y}_i, y_i)$$

where $\hat{y}_i = f(x; w)$ is the output of the neural network

- Define a loss function, such as:

Squared loss $\sum_{i=1}^{N} \frac{1}{2}(\hat{y}_i - y_i)^2$ [Regression]
Cross-entropy loss $-\sum_{i=1}^{N} y_i log(\hat{y}_i)$ [Classification]

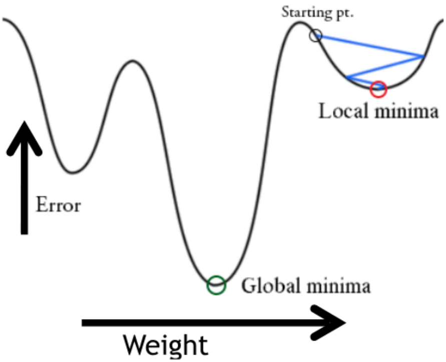
- Use gradient descent

$$w_{t+1} = w_t - \eta \frac{\partial E}{\partial w_t}$$

where η is the `learning rate` and E is the `error/loss`

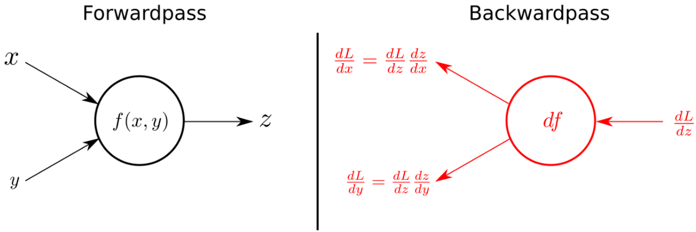
Training neural networks

- Training neural networks is a **non-convex** optimization problem
  - could run into local minima during training



Training neural networks

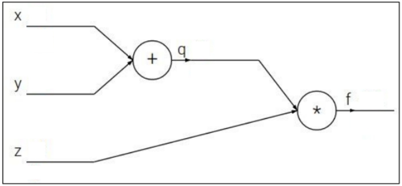
- First perform a **forward pass**
- Update weights with a **backward pass**



Backpropagation: a simple example

Backpropagation: a simple example

$$f(x, y, z) = (x + y)z$$

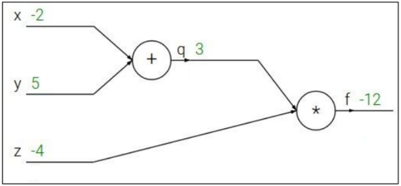


Backpropagation: a simple example

Backpropagation: a simple example

$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4



Backpropagation: a simple example

Backpropagation: a simple example

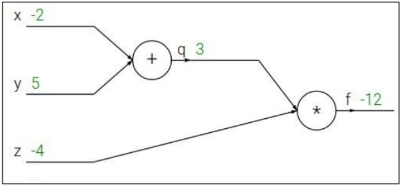
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$\boxed{q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1}$$

$$\boxed{f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q}$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: a simple example

Backpropagation: a simple example

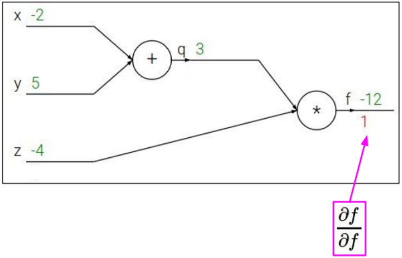
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: a simple example

Backpropagation: a simple example

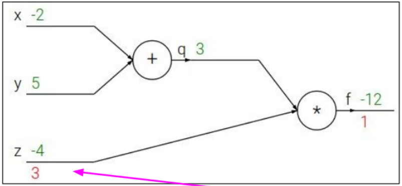
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$\frac{\partial f}{\partial z}$

Backpropagation: a simple example

Backpropagation: a simple example

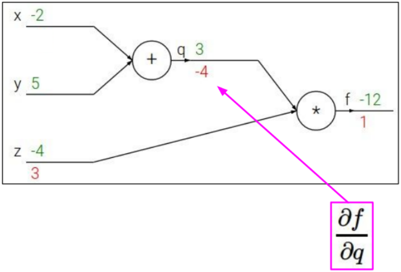
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: a simple example

Backpropagation: a simple example

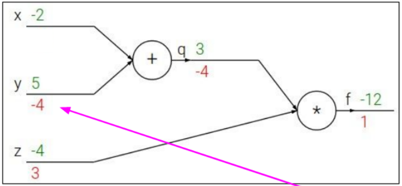
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$\boxed{q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1}$$

$$\boxed{f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q}$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\boxed{\frac{\partial f}{\partial y}}$$

Chain rule:

$$\boxed{\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial y}}$$

Upstream gradient

Local gradient

Backpropagation: a simple example

Backpropagation: a simple example

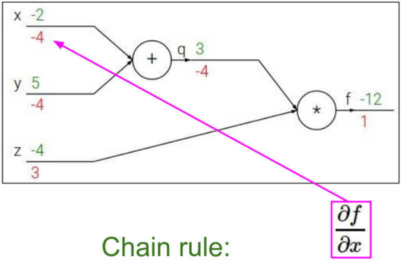
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



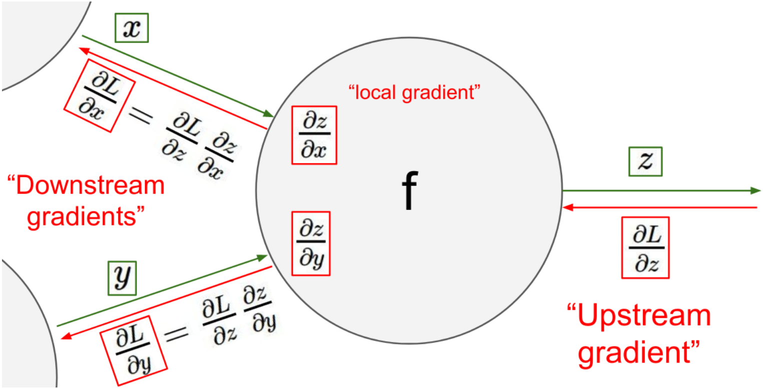
Chain rule:

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$$

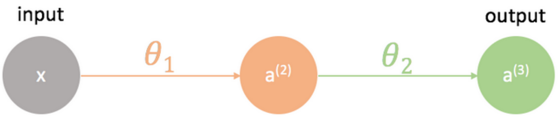
Upstream gradient

Local gradient

Backpropagation: a simple example



Backpropagation for neural networks



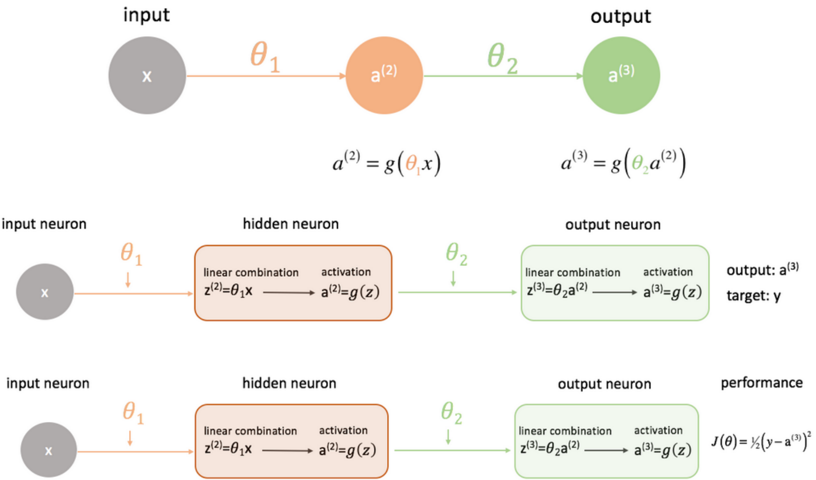
$a^{(2)} = g(\theta_1 x)$ $\quad$ $a^{(3)} = g(\theta_2 a^{(2)})$

Goal: minimize the difference between neural network's output and the target output – defined by the loss function $J(\theta)$ $\quad \theta = [\theta_1 \ \theta_2]$

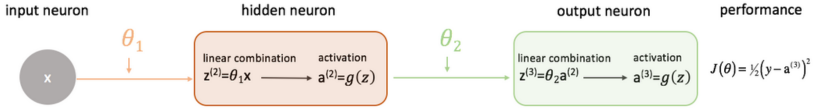
- define the relationship between the cost function and each weight (given by `partial derivative`)
  - represents the performance change with respect to each parameter
- update these weights in an iterative process using gradient descent

• $\frac{\partial J(\theta)}{\partial \theta_1} = ?$ $\quad$ • $\frac{\partial J(\theta)}{\partial \theta_2} = ?$

Backpropagation for neural networks



Backpropagation for neural networks



Chain rule: $\frac{\partial}{\partial z} f(g(z)) = \frac{\partial f}{\partial g} \frac{\partial g}{\partial z}$

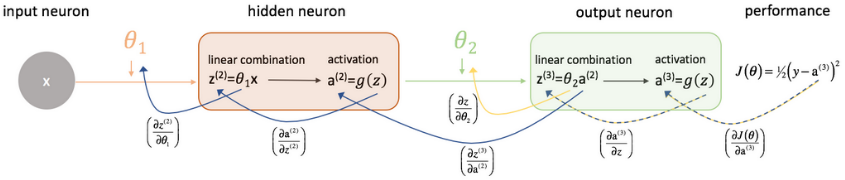
Applying chain rule to solve for $\frac{\partial J(\theta)}{\partial \theta_2}$

$$\frac{\partial J(\theta)}{\partial \theta_2} = \left(\frac{\partial J(\theta)}{\partial a^{(3)}}\right)\left(\frac{\partial a^{(3)}}{\partial z^{(3)}}\right)\left(\frac{\partial z^{(3)}}{\partial \theta_2}\right)$$

Using a similar logic, we have

$$\frac{\partial J(\theta)}{\partial \theta_1} = \left(\frac{\partial J(\theta)}{\partial a^{(3)}}\right)\left(\frac{\partial a^{(3)}}{\partial z^{(3)}}\right)\left(\frac{\partial z^{(3)}}{\partial a^{(2)}}\right)\left(\frac{\partial a^{(2)}}{\partial z^{(2)}}\right)\left(\frac{\partial z^{(2)}}{\partial \theta_1}\right)$$

Backpropagation for neural networks



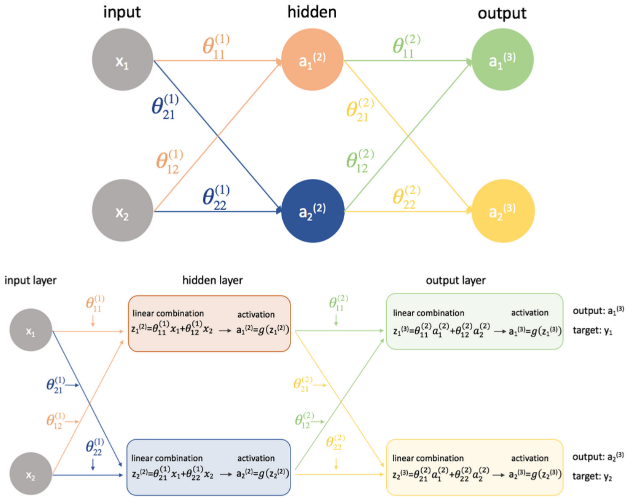
$$\frac{\partial J(\theta)}{\partial \theta_2} = \left(\frac{\partial J(\theta)}{\partial a^{(3)}}\right)\left(\frac{\partial a^{(3)}}{\partial z^{(3)}}\right)\left(\frac{\partial z^{(3)}}{\partial \theta_2}\right)$$

$$\frac{\partial J(\theta)}{\partial a^{(3)}} = (y - a^{(3)})$$

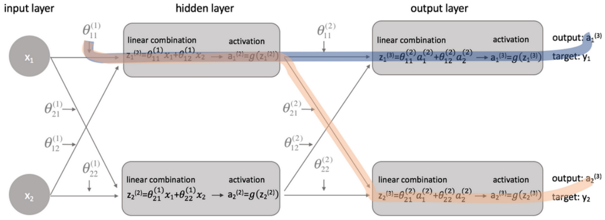
If $g(z) = \frac{1}{1+e^{-z}}$ then $\frac{\partial a^{(3)}}{\partial z^{(3)}} = \frac{e^{-z^{(3)}}}{(1+e^{-z^{(3)}})^2} = g(z^{(3)})(1 - g(z^{(3)}))$

$$\frac{\partial z^{(3)}}{\partial \theta_2} = a^{(2)}$$

Backpropagation for neural networks



Backpropagation for neural networks



$$J(\theta) = \frac{1}{2}((y_1 - a_1^{(3)})^2 + (y_2 - a_2^{(3)})^2)$$

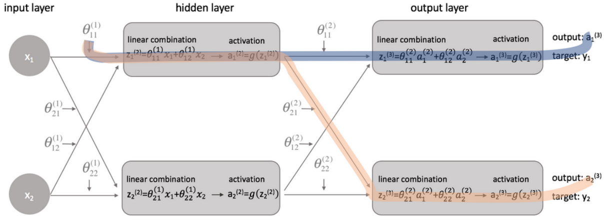
The derivative chain for the blue path is:

$$\left(\frac{\partial J(\theta)}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right)$$

The derivative chain for the orange path is:

$$\left(\frac{\partial J(\theta)}{\partial a_2^{(3)}}\right)\left(\frac{\partial a_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right)$$

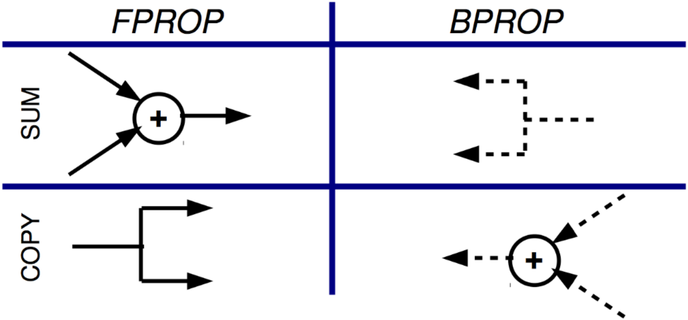
Backpropagation for neural networks



Combining these two, we get the total expression for $\frac{\partial J(\theta)}{\partial \theta_{11}^{(1)}}$

$$\frac{\partial J(\theta)}{\partial \theta_{11}^{(1)}} = \left(\frac{\partial J(\theta)}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right) + \left(\frac{\partial J(\theta)}{\partial a_2^{(3)}}\right)\left(\frac{\partial a_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right)$$

Backpropagation for neural networks



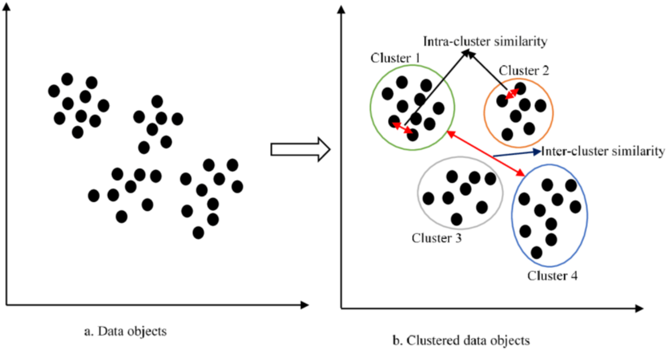
Limitations of neural networks

- requires large training dataset
- prone to overfitting
- model is not interpretable
- size and structure chosen by trial and error

Clustering

- the organization of unlabeled data into similarity groups called clusters
  - points within each cluster are similar to each other
  - points from different clusters are dissimilar
- unsupervised learning technique
  - no target labels available
- data points are usually in a high-dimensional space and similarity defined using a distance measure
  - Euclidean, cosine, Jaccard, edit distance, etc.
- helps to find intrinsic structures within data

Clustering: Example



Clustering: Applications

- cluster customers based on their purchase histories
- group documents based on their content
- **image segmentation** - partitioning an image into multiple segments

Original Image

Segmented Image when K = 3

Types of clustering

Hierarchical find successive clusters using previously established clusters — can be `top-down` or `bottom-up`

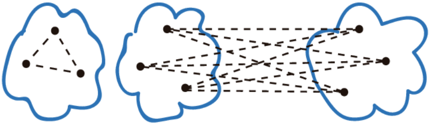
1. **Agglomerative:** begin with each element as a separate cluster and merge them into successively larger clusters
2. **Divisive:** begin with the whole dataset and successively break it down into smaller clusters

Partitional decomposes a data set into a set of disjoint clusters — all clusters are determined at once

Bayesian generate a `posteriori distribution` over the collection of all partitions of data

What do we need for clustering

- a **distance measure:** similarity/dissimilarity measure
- criterion function to evaluate a clustering
  1. **Intra-cluster cohesion(compactness):** measures the closeness of the data points to the cluster centroid
  2. **Inter-cluster separation(isolation):** different cluster centroids should be far away from one another



- number of clusters
  - fixed by the user
  - determined from the dataset based on some criterion
- clustering algorithm

References

1. https://cs231n.github.io/neural-networks-1/
2. http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture04.pdf
3. https://www.cs.toronto.edu/~jlucas/teaching/csc411/lectures/tut5_handout.pdf
4. http://www.mit.edu/~9.54/fall14/slides/Class13.pdf

Thanks Google for the pictures!