Digital Data

Types of Digital Data

Structured

- Data Organized in the form of rows and columns.
- Examples : RDBMS, Microsoft Excel

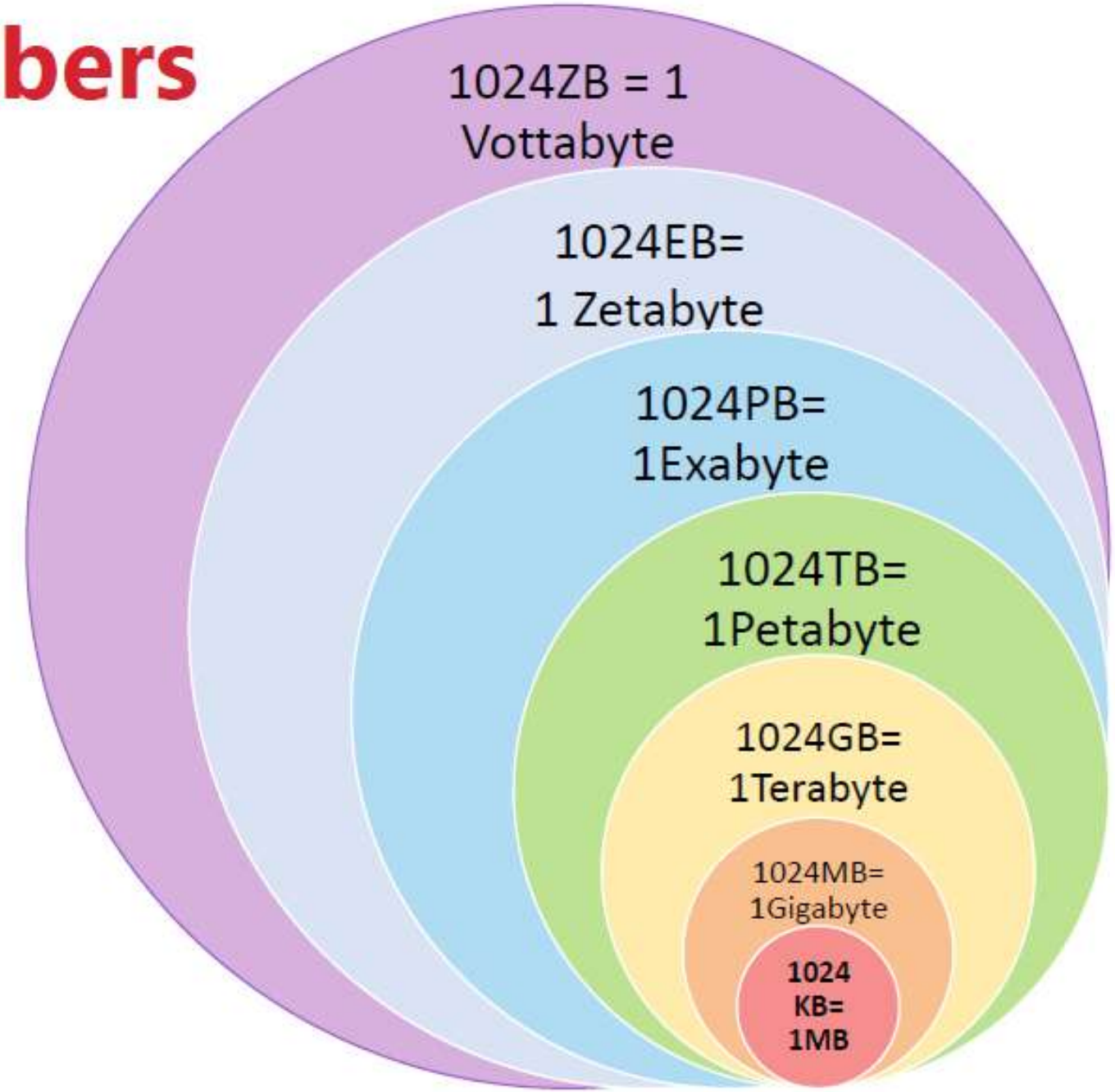
Semi structured

- Has some structure, but does not conform to any data model.
- Examples : All markup languages, HTML,XML etc

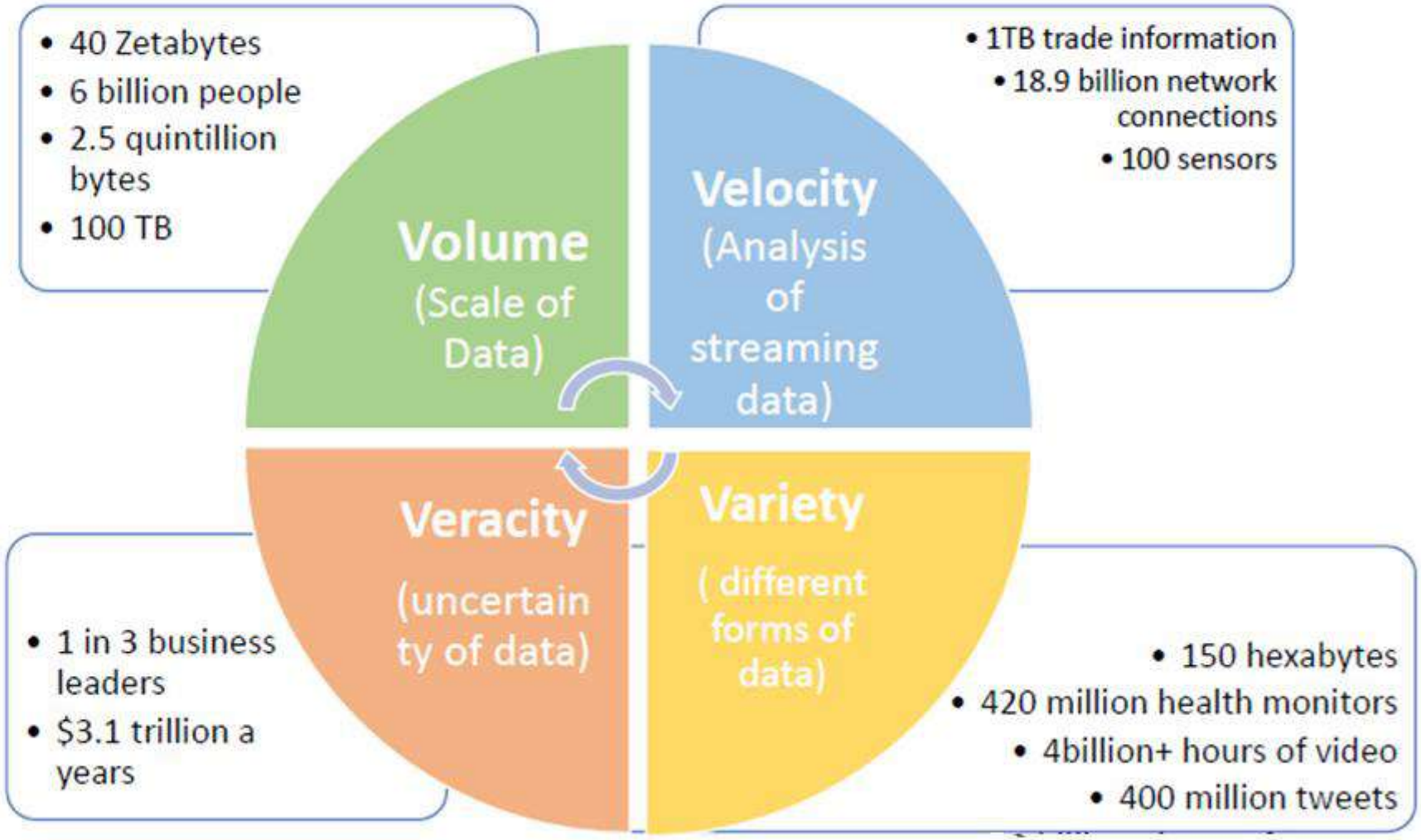
Unstructured

- Does not confirm to any data model.
- Examples : Media files, data from sensors and physical devices.

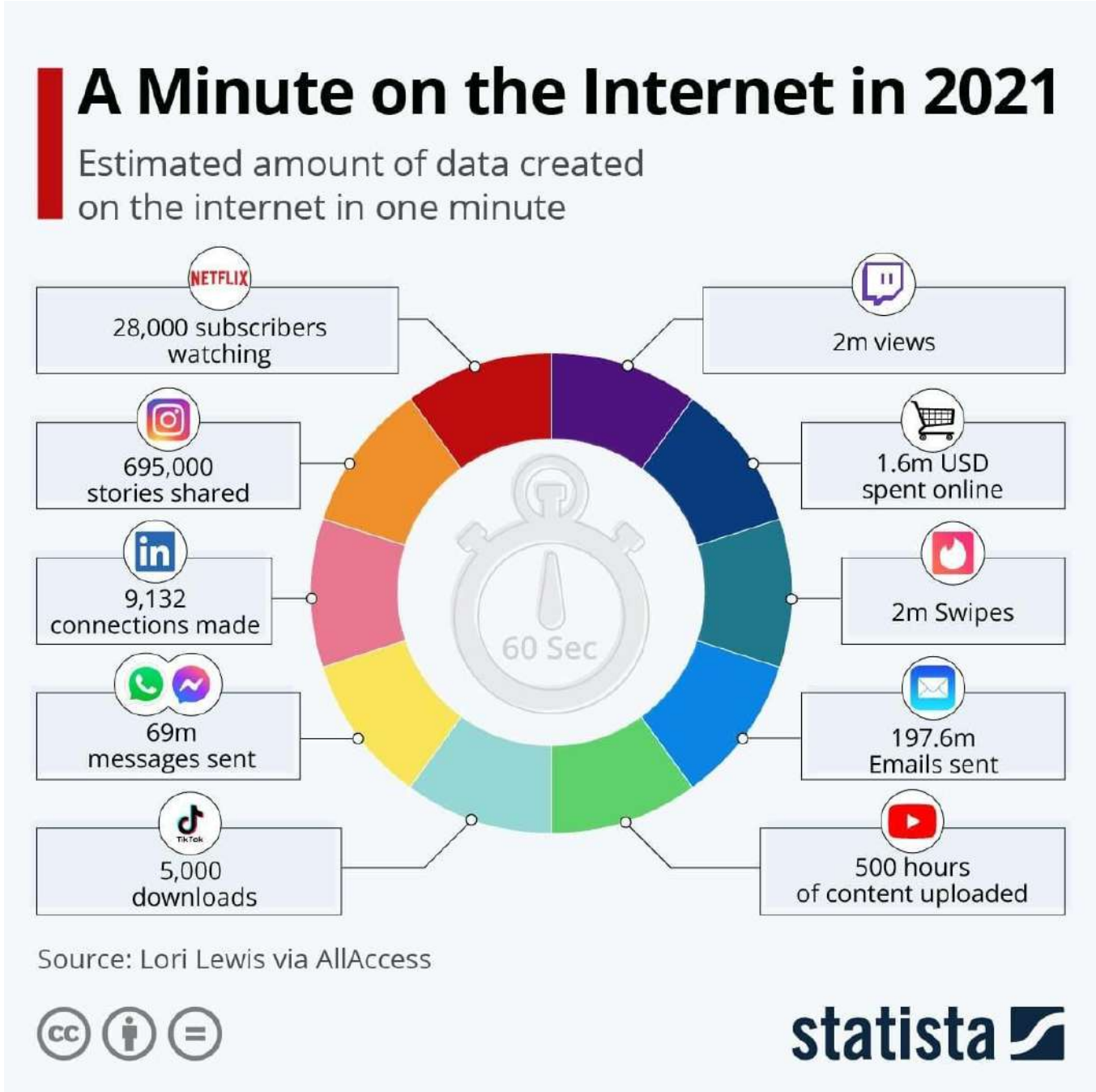
Big Data Numbers



What is Big Data?



What Happens in 60 seconds over internet in 2023?



Link: https://www.weforum.org/agenda/2021/08/one-minute-internet-web-social-media-technology-online/

Machine Learning

SL No	No. of Action Scene	No. of Comedy Scene	Class/ Label/ Category
1	100	15	Action
2	20	95	Comedy
3	90	5	Action
4	10	85	Comedy
5	50	50	

What is the category of the movie?

How many groups are there?

k nearest neighbour

Train Set

SL No	No. of Action Scene	No. of Comedy Scene	Class/ Labels/ Category
T1	100	15	Action
T2	20	95	Comedy
T3	90	5	Action
T4	10	85	Comedy

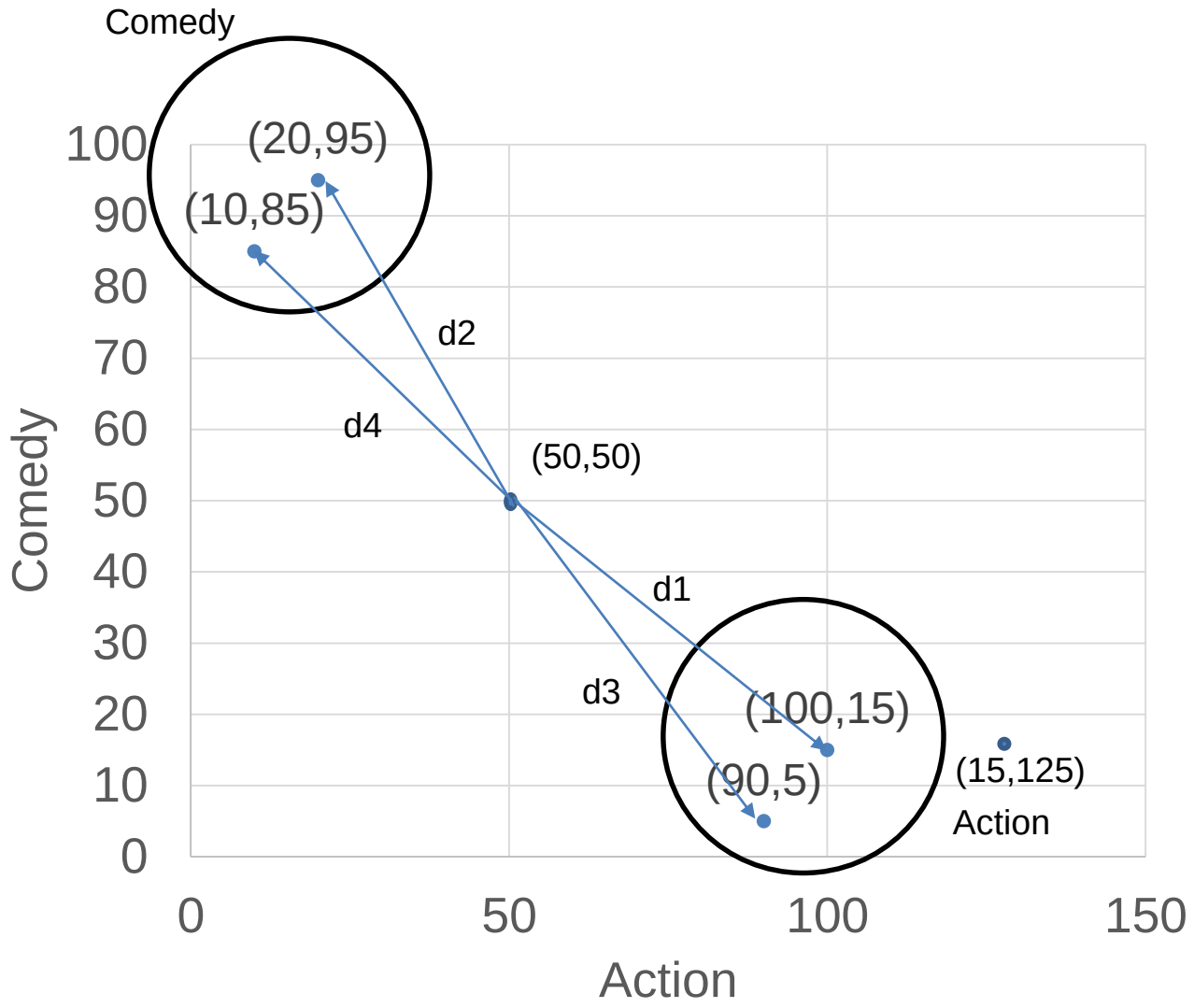
Test Set

T5	50	50	?

Steps to be followed

1. Compute Euclidian Distance from val/test to all the training data points
2. Sort the data in the ascending order
3. Compute the value of k
    a. Use Validation data to determine the value of k. Voting scheme may be used for taking the majority voting.
    b. Compute accuracy on different values of k and choose the one with the highest accuracy
    c. Value of k should be odd in case of binary classification

- Use the value of k to compute the classification on test data and compute the accuracy



Step 1: Compute Euclidean Distance

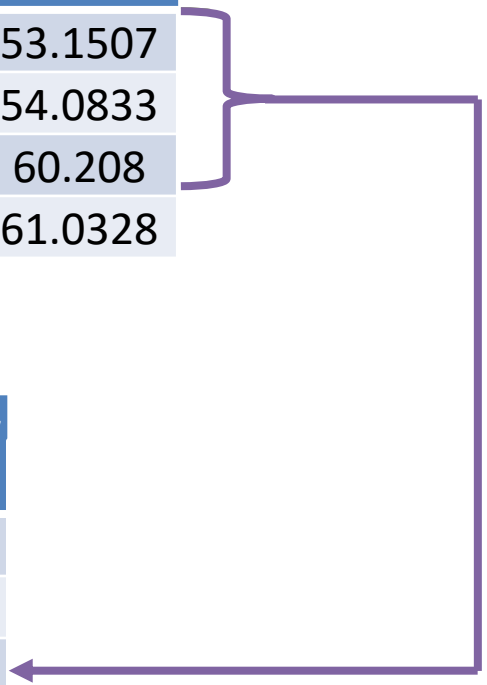
Sl. No.	Distance	Train	Test	Class of Train	Distance
1	d1	T1	T5	Action	61.03277808
2	d1	T2	T5	Comedy	54.08326913
3	d1	T3	T5	Action	60.20797289
4	d1	T4	T5	Comedy	53.15072906

Step 2: Sort the data on the basis of Distance

Sl. No.	Distance	Train	Test	Class of Train	Distance
4	d1	T4	T5	Comedy	53.1507
2	d1	T2	T5	Comedy	54.0833
3	d1	T3	T5	Action	60.208
1	d1	T1	T5	Action	61.0328

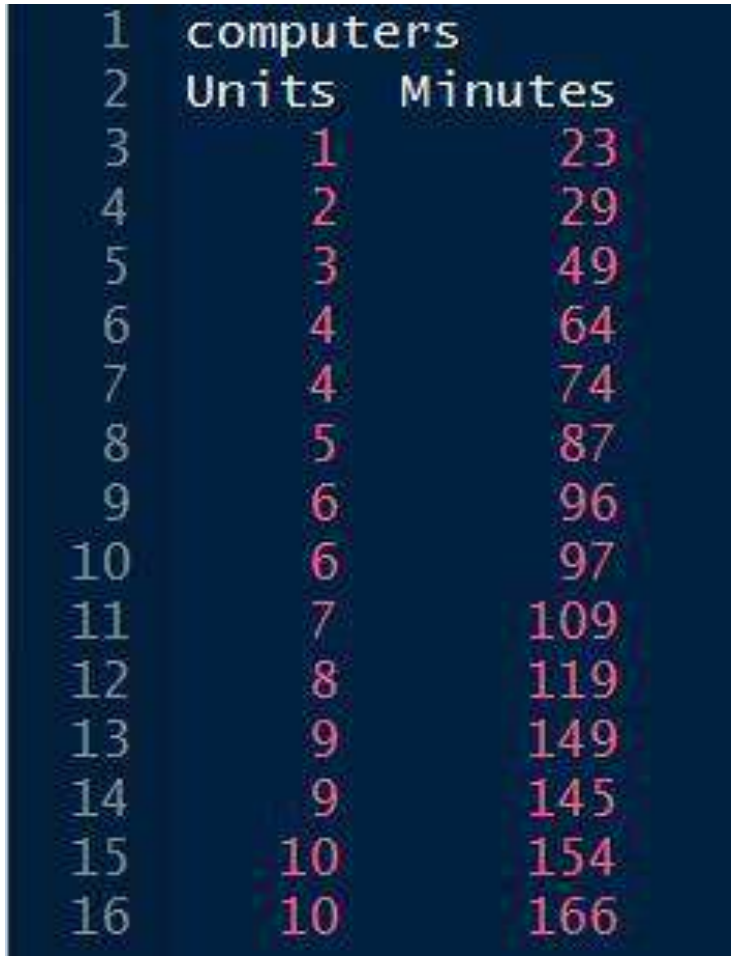
Step 2: Determine the value of k

Different Values of k	Classes	Majority Vote
1	{Comedy}	Comedy
2	{Comedy}	Comedy
3	{Comedy, Comedy, Action}	Comedy



Linear Regression

The following data* represents the number of units (Units) being replaced in a computer and the corresponding time taken to repair the computer in minutes (Minutes)



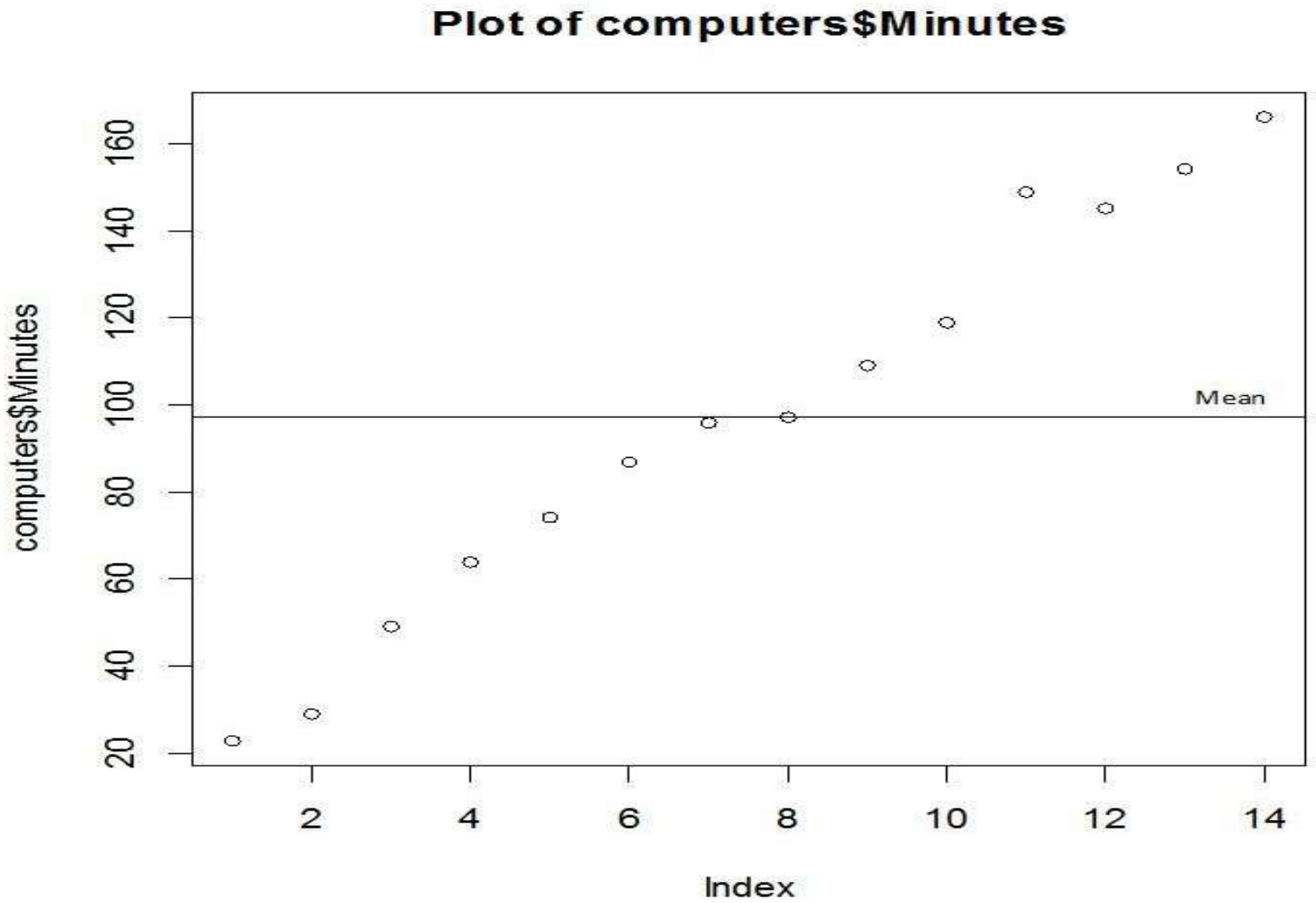
	computers	
2	Units	Minutes
3	1	23
4	2	29
5	3	49
6	4	64
7	4	74
8	5	87
9	6	96
10	6	97
11	7	109
12	8	119
13	9	149
14	9	145
15	10	154
16	10	166

If we were to estimate the typical time taken by the computer shop to repair a computer, which of the following would be appropriate?

a) Arithmetic mean = 97.21 minutes

b) Median = 96.50 minutes

Answer: Either Mean or Median can be used when trying to predict the expected value of a single variable.



[Data: 23 29 49 64 74 87 96 97 109 119 149 145 154 166, Mean: 97.21]

Regression Analysis

Regression analysis is a statistical process for estimating the relationships among variables. It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors').

Regression analysis focuses on building a model that can be used to predict the value of the dependent variable based on the predictor variables.

The regression model is represented using a mathematical model of form $y= f(X)$, where y is the dependent variable and X is the set of predictor variables (x_1, x_2… x_n).

Regression Analysis

Linear form: $f(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon$

Non-linear form: $f(X) = \beta_0 + \beta_1 {x_1}^{p_1} + \beta_2 {x_2}^{p_2} + \ldots + \beta_n {x_n}^{p_n} + \epsilon$

Regression Analysis

Some commonly used types of linear forms are:

Simple linear form – Here there is one predictor and one dependent variable : $f(X) = \beta_0 + \beta_1 x_1 + \epsilon$

Multiple linear form – Here there are multiple predictor variables and one dependent variable:

$f(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon$

Some commonly used types of non-linear forms are:

Polynomial form: $f(X) = \beta_0 + \beta_1 {x_1}^{p_1} + \beta_2 {x_2}^{p_2} + \ldots + \beta_n {x_n}^{p_n} + \epsilon$

Quadratic form: $f(X) = \beta_0 + \beta_1 {x_1}^2 + \beta_2 x_1 + \epsilon$

Logistic form: $f(x) = \frac{1}{1+e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n)}} + \epsilon$

Where β_0, β_1, $\beta_2 \ldots \beta_n$ are said to be the regression coefficients and ϵ accounts for the error in prediction. The regression coefficients and the error in prediction are real numbers.

Regression Analysis

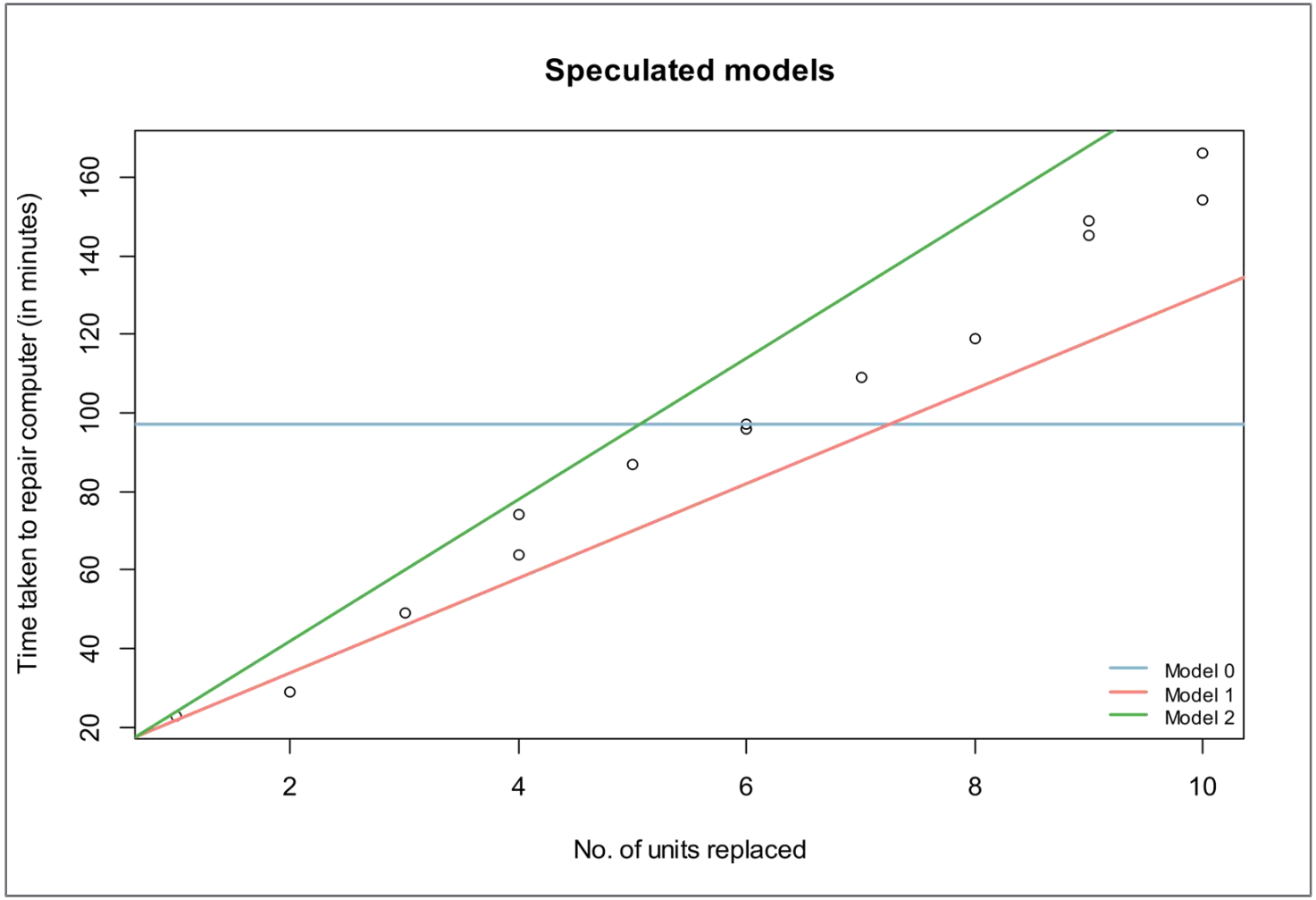
Time taken to repair the computer = $\beta_0 + \beta_1 *$ No. of units replaced

Model 0: Time taken to repair the computer = *97.21 (mean)*

Model 1:Time taken to repair the computer = 10 + 12 * No. of units replaced

Model 2: Time taken to repair the computer = 6 + 18 * No. of units replaced

Regression Analysis



Regression Analysis

```
model0
Units replaced Observed time taken Expected value Expected - Observed value
             1                  23       97.21429                74.2142857
             2                  29       97.21429                68.2142857
             3                  49       97.21429                48.2142857
             4                  64       97.21429                33.2142857
             4                  74       97.21429                23.2142857
             5                  87       97.21429                10.2142857
             6                  96       97.21429                 1.2142857
             6                  97       97.21429                 0.2142857
             7                 109       97.21429               -11.7857143
             8                 119       97.21429               -21.7857143
             9                 149       97.21429               -51.7857143
             9                 145       97.21429               -47.7857143
            10                 154       97.21429               -56.7857143
            10                 166       97.21429               -68.7857143
```

Error = Sum(minutes mean - computer$minutes)
= -8.25e -14

Error = Sum((minutes mean - computer$minutes)^2)
= 27768.36

Regression Analysis

model1

Units replaced	Observed time	Expected time	Expected - Observed value
1	23	22	-1
2	29	34	5
3	49	46	-3
4	64	58	-6
4	74	58	-16
5	87	70	-17
6	96	82	-14
6	97	82	-15
7	109	94	-15
8	119	106	-13
9	149	118	-31
9	145	118	-27
10	154	130	-24
10	166	130	-36

Error = Sum((minutesmean - computer$minutes)^2)
= 4993

Regression Analysis

model2

Units replaced	Observed time	Expected time	Expected - Observed value
1	23	24	1
2	29	42	13
3	49	60	11
4	64	78	14
4	74	78	4
5	87	96	9
6	96	114	18
6	97	114	17
7	109	132	23
8	119	150	31
9	149	168	19
9	145	168	23
10	154	186	32
10	166	186	20

Error = Sum((minutesmean - computer$minutes)^2)
= 5001

Regression Analysis

The best fit model is obtained by solving the linear regression model $y = \beta_0 + \beta_1 x + \varepsilon$

To determine the coefficients β_0 and β_1 such that the error (as shown below) is minimum:

Differentiating the equation $\sum_{i=1}^{n} (y_i - b_0 - b_1 x_i)^2$ w.r.t b_0 and equating to 0 we get

$$b_1 = \frac{[\ \sum_{i=1}^{n} (x_i y_i)\ ] - n\bar{x}\bar{y}}{[\ \sum_{i=1}^{n} {x_i}^2\ ] - n\bar{x}^2}$$

Similarly, differentiating the equation $\sum_{i=1}^{n} (y_i - b_0 - b_1 x_i)^2$ w.r.t b_1 and equating to 0 we get

$$b_0 = \bar{y} - b_1\bar{x}$$

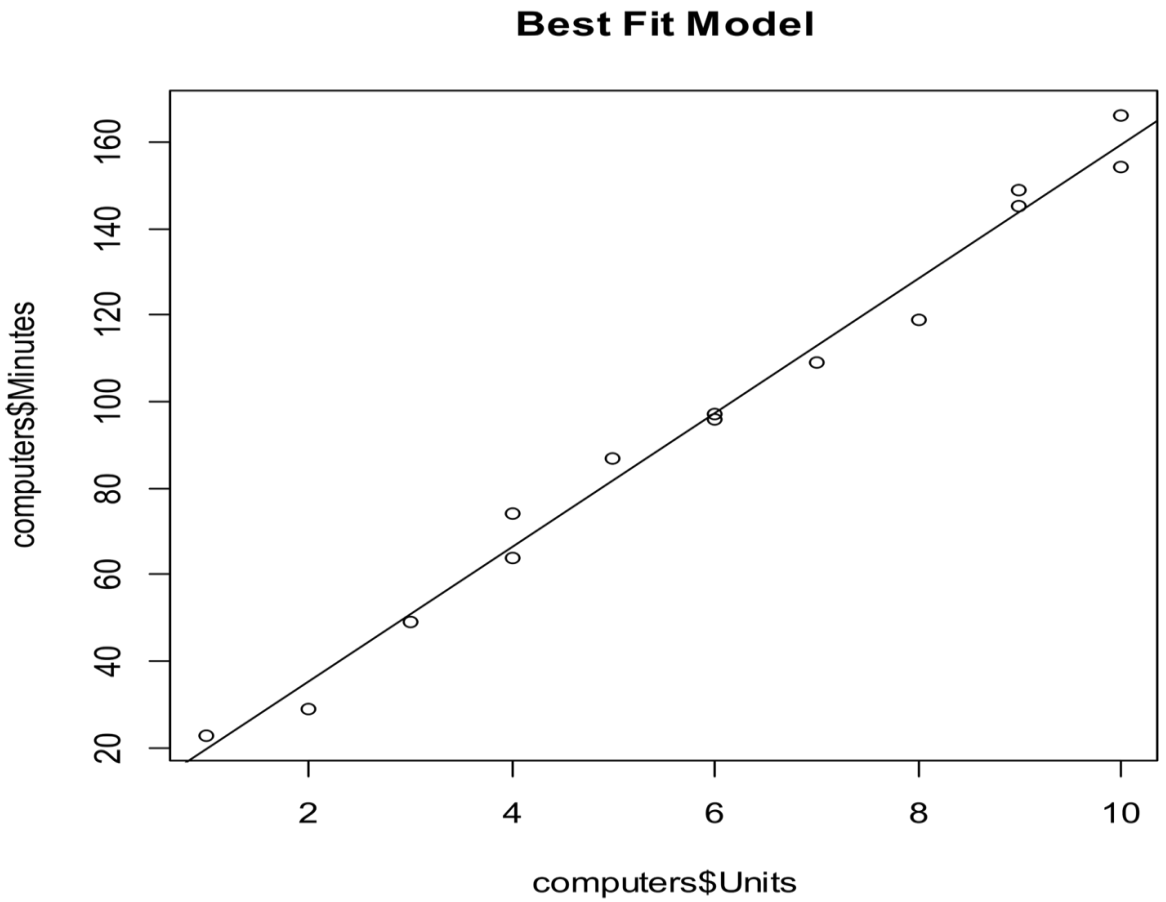
Here, b_0 and b_1 are the regression coefficients for the sample, $\bar{x}$ and $\bar{y}$ are the sample means of the variables X and Y respectively.

Best fit Model

Using the previously obtained formula, b_0 and b_1 for the given sample dataset is determined as 4.162 (approximately) and 15.509 (approximately) respectively as shown.

```
#Computing values for b0 and b1
x <- computers$Units
y <- computers$Minutes
xiyi<- x*y
n <- nrow(computers)
xmean <- mean(computers$Units)
ymean <- mean(computers$Minutes)
numerator <- sum(xiyi)-n*xmean*ymean
denominator <- sum(x^2)-n*(xmean^2)
b1 <- numerator/denominator
b0 <- ymean-b1*xmean
#values of b0 and b1
b0
[1] 4.161654
b1
[1] 15.50877
```

The plot of this model along with the given data is as shown below




```
#Plot of Best fit Model: Minutes = 4.161654 + 15.50877*units to be replaced
plot(computers$Units,computers$Minutes, main="Model 1")
abline(b0,b1)
```

Best fit Model

The following code snippet shows the number of units replaced, observed time taken, expected time taken (based on the model) and the difference between predicted and observed values for the best fit model. The total sum of squared errors for the best fit model is 348.8484.

```
#Best fit model Units, Observed time, Expected time, Difference between observed and expected time
best_fit <- data.frame(matrix(data=c(computers$Units,computers$Minutes,(b0+b1*computers$Units),
                                     ((b0+b1*computers$Units)-computers$Minutes)),ncol=4))
colnames(best_fit) <- c("Units replaced", "Observed time", "Expected time",
                        "Expected - Observed value")
best_fit
```

Units replaced	Observed time	Expected time	Expected - Observed value
1	23	19.67043	-3.3295739
2	29	35.17920	6.1791980
3	49	50.68797	1.6879699
4	64	66.19674	2.1967419
4	74	66.19674	-7.8032581
5	87	81.70551	-5.2944862
6	96	97.21429	1.2142857
6	97	97.21429	0.2142857
7	109	112.72306	3.7230576
8	119	128.23183	9.2318296
9	149	143.74060	-5.2593985
9	145	143.74060	-1.2593985
10	154	159.24937	5.2493734
10	166	159.24937	-6.7506266

```
#Sum of squared errors for Best fit model
sum((((b0+b1*computers$Units)-computers$Minutes))^2)
[1] 348.8484
```

Best Fit Model

The best fit linear regression model can also be obtained in R using the lm() command as shown below:

Syntax for the lm() function : lm(dependent variable ~ predictor variable)

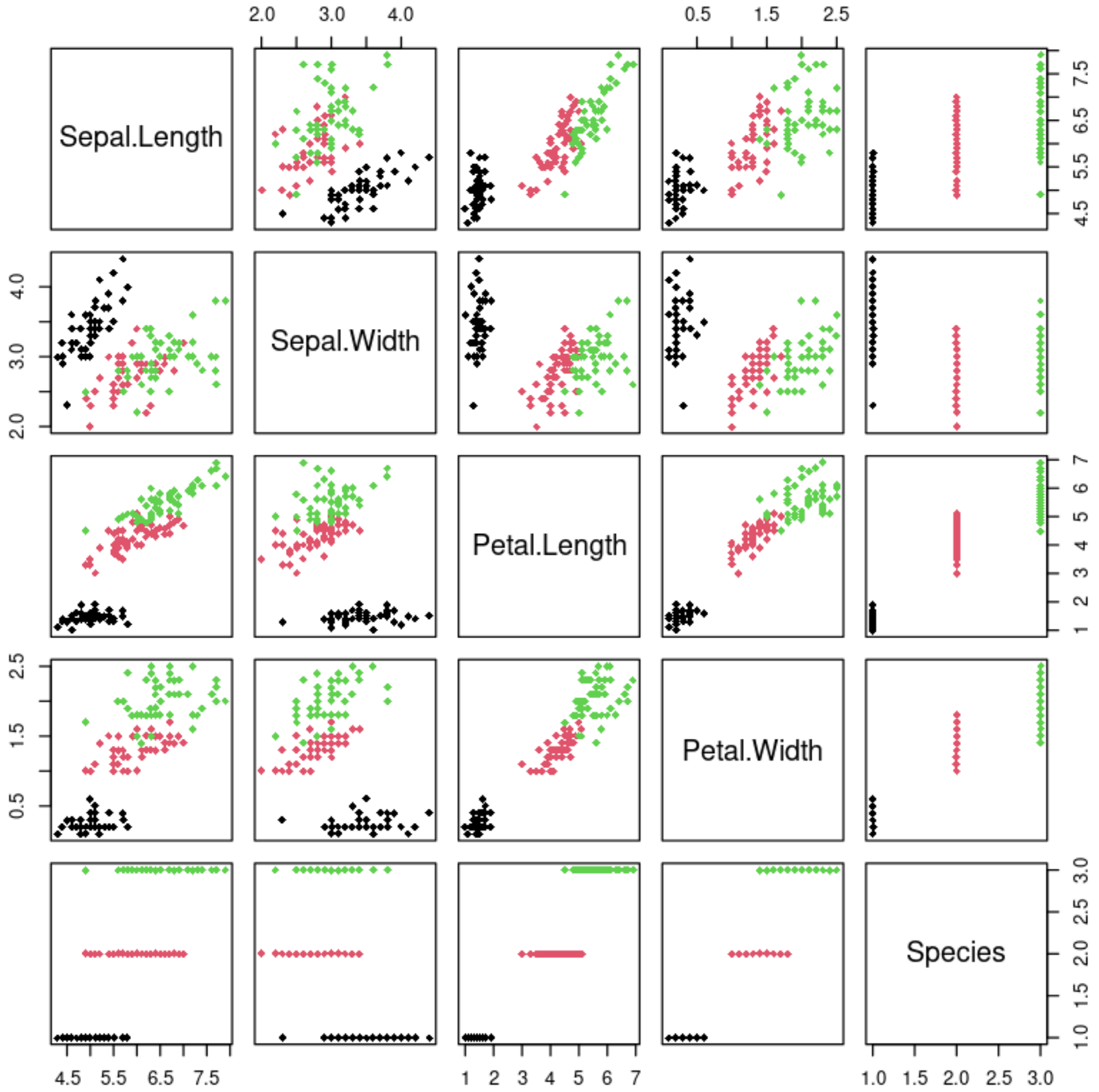
```
1 lm.model <- lm(computers$Minutes~computers$Units)
2 lm.model
3
4 Call:
5   lm(formula = computers$Minutes ~ computers$Units)
6
7 Coefficients:
8   (Intercept)  computers$Units
9 4.162          15.509
```

The value of coefficients b_0 and b_1 indicate that it takes approximately 15.509 minutes to replace a unit and a fixed time of approximately 4.162 minutes to understand a given repair.

Logistic Regression

Iris Dataset

Sepal Lengh	Sepal Width	Petal Length	Petal Width	Class
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3	1.4	0.2	Iris-setosa
5.6	2.5	3.9	1.1	Iris-versicolor
5.9	3.2	4.8	1.8	Iris-versicolor
6.3	2.9	5.6	1.8	Iris-virginica
6.5	3	5.8	2.2	Iris-virginica



Learning Function

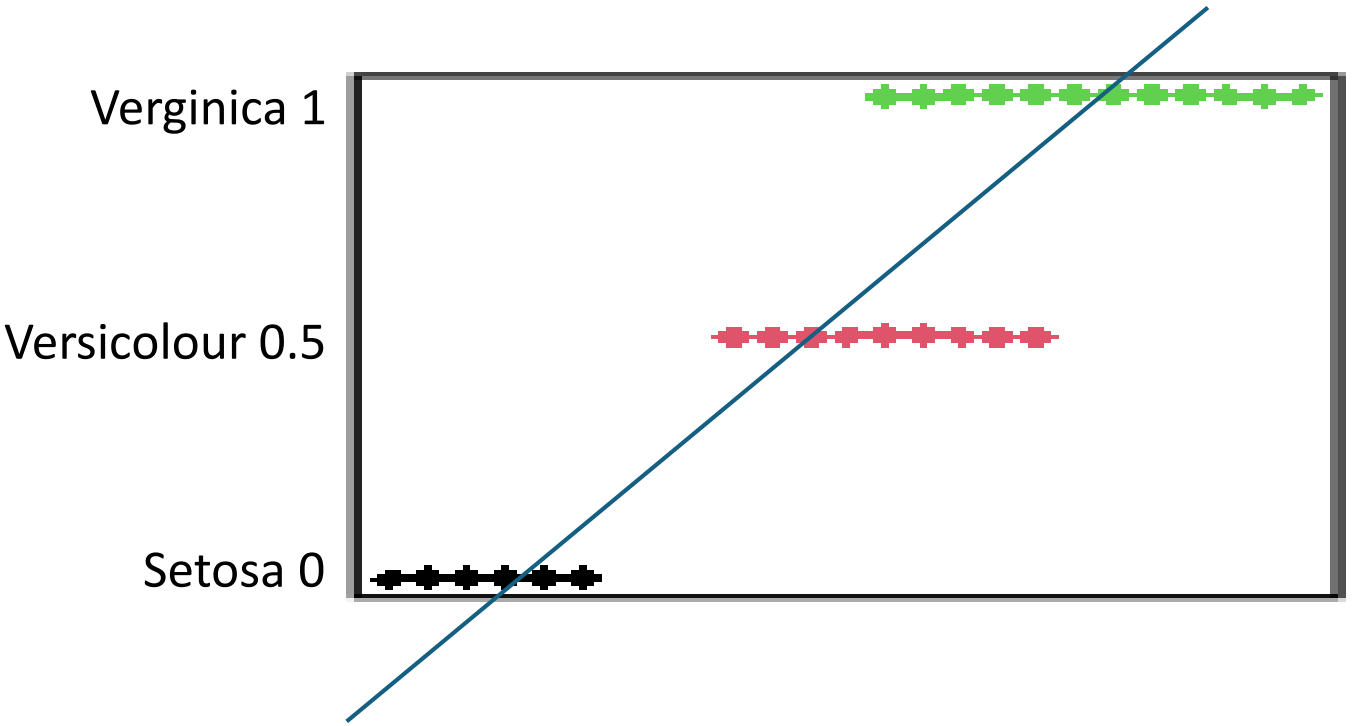
Multiple Linear Regression

$$F(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4$$

Simple Linear Regression

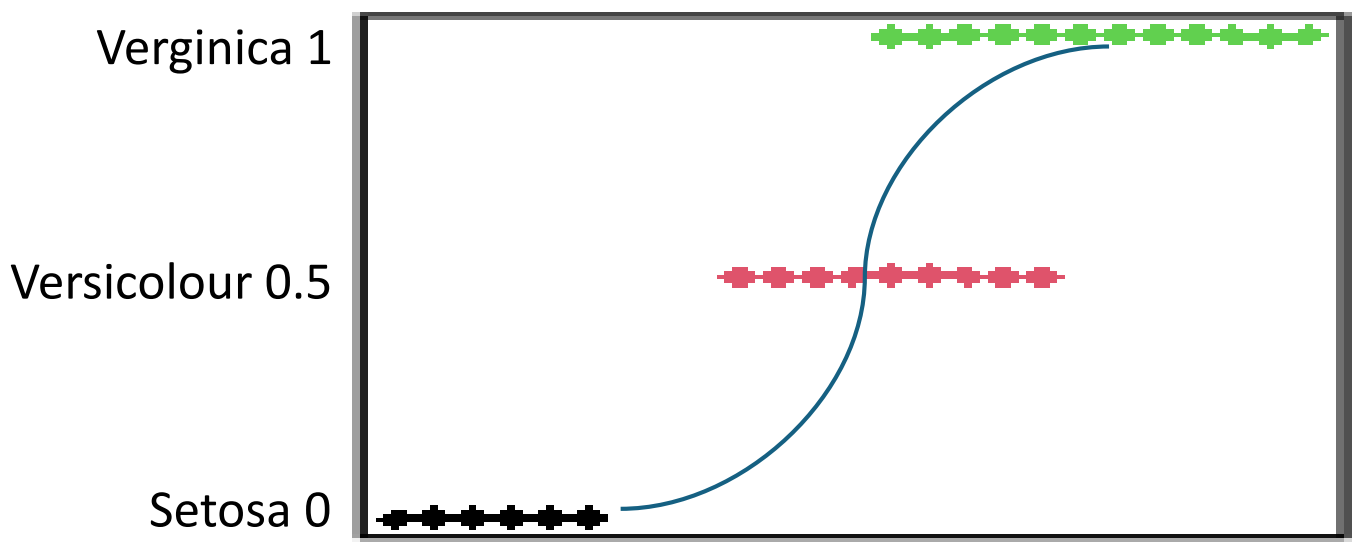
$$F(x) = \beta_0 + \beta_1 x_1$$

Range: (-∞ , ∞)



Learning Function

Need a function whose range is (0,1)



Odds

Odds of my winning to loosing is 2 is to 8.

- **Odds –** Something happening to something not happening

$$Odds = \frac{Something\ happening}{Something\ not\ happening} = \frac{Win}{Loose}$$

- **Probability** – Something happening to all the set of events

$$P_{win} = \frac{Something\ happening}{(Something\ happening) + (Something\ not\ happening)} = \frac{Win}{Win + Loose}$$

$$P_{Loose} = \frac{Something\ not\ happening}{(Something\ happening) + (Something\ not\ happening)} = \frac{Loose}{Win + Loose}$$

$$Odds = \frac{Something\ happening}{Something\ not\ happening} = \frac{Win}{Loose} = \frac{Win\ /(Win+Loose)}{Loose\ /(Win+Loose)} = \frac{P_{win}}{P_{Loose}}$$

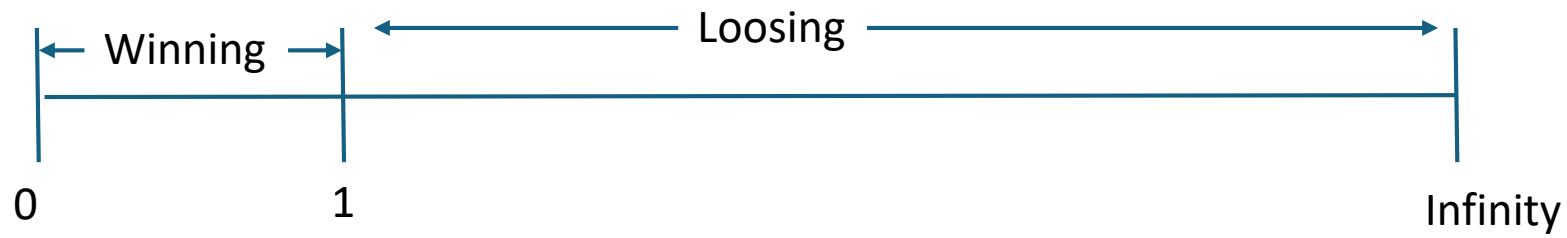
Odds

Winning

- Odd(1:2) = $\frac{1}{2}$ =0.5
- Odd(1:4) = $\frac{1}{4}$ =0.25
- Odd(1:8) = $\frac{1}{8}$ =0.125
- Odd(1:16) = $\frac{1}{16}$ =0.0625
- Odd(1:32) = $\frac{1}{32}$ =0.03125

Loosing

- Odd(2:1) = $\frac{2}{1}$ = 2
- Odd(4:1) = $\frac{4}{1}$ = 4
- Odd(8:1) = $\frac{8}{1}$ = 8
- Odd(16:1) = $\frac{16}{1}$ = 16
- Odd(32:1) = $\frac{32}{1}$ = 32



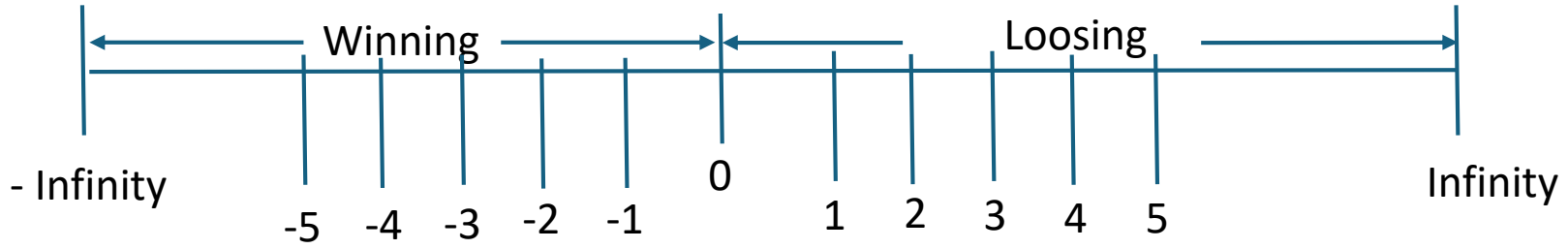
Log of Odds

Winning

- Odd(1:2) = $\frac{1}{2}$ =0.5 = log (0.5) = -1
- Odd(1:4) = $\frac{1}{4}$ =0.25 = log (0.25) = -2
- Odd(1:8) = $\frac{1}{8}$ =0.125 = log (0.125) = -3
- Odd(1:16) = $\frac{1}{16}$ =0.0625 = log (0.0625) = -4
- Odd(1:32) = $\frac{1}{32}$ =0.03125 = log (0.03125) = -5

Loosing

- Odd(2:1) = $\frac{2}{1}$ = 2 = log (2) = 1
- Odd(4:1) = $\frac{4}{1}$ = 4 = log (4) = 2
- Odd(8:1) = $\frac{8}{1}$ = 8 = log (8) = 3
- Odd(16:1) = $\frac{16}{1}$ = 16 = log (16) = 4
- Odd(32:1) = $\frac{32}{1}$ = 32 = log (32) = 5



Log of Odds

$$Odds = \frac{P_{win}}{P_{Loose}}$$

$$\log(Odds) = \log\left(\frac{P_{win}}{P_{Loose}}\right)$$

$$\log(Odds) = \log\left(\frac{P_{win}}{1 - P_{win}}\right)$$

$$\log\left(\frac{P_{win}}{1 - P_{win}}\right) = \beta_0 + \beta_1 x$$

$$\frac{P_{win}}{1 - P_{win}} = e^{\beta_0 + \beta_1 x}$$

$$P_{win} = (1 - P_{win})e^{\beta_0 + \beta_1 x}$$

$$P_{win} = 1e^{\beta_0 + \beta_1 x} - P_{win}e^{\beta_0 + \beta_1 x}$$

$$P_{win} + P_{win}e^{\beta_0 + \beta_1 x} = 1e^{\beta_0 + \beta_1 x}$$

$$P_{win}(1 + e^{\beta_0 + \beta_1 x}) = 1e^{\beta_0 + \beta_1 x}$$

$$P_{win} = \frac{e^{\beta_0 + \beta_1 x}}{(1 + e^{\beta_0 + \beta_1 x})}$$

Divide numerator and Denominator by $e^{\beta_0 + \beta_1 x}$

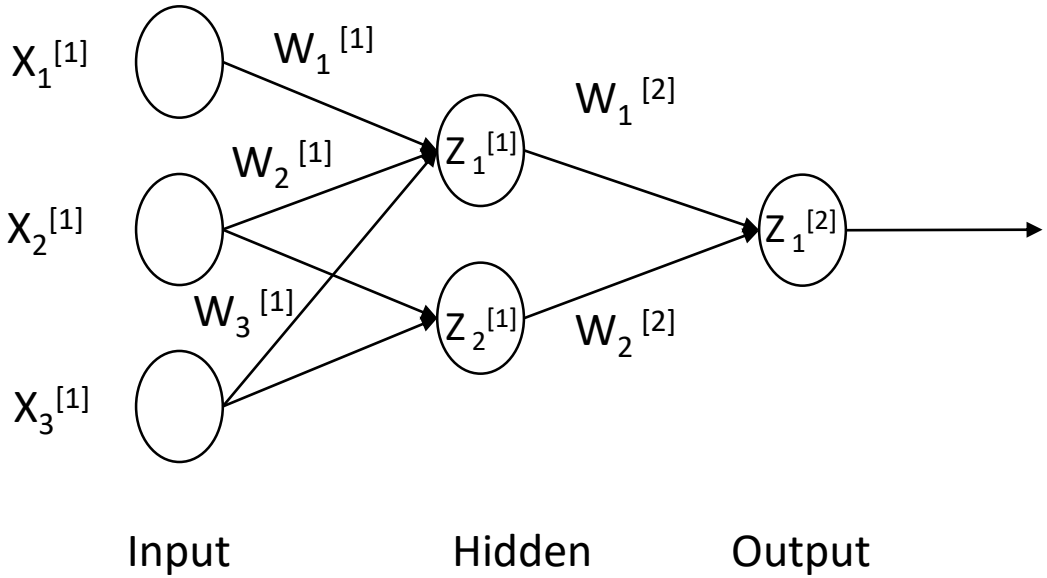
$$P_{win} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Logistic Function/Sigmoid Function

Activation Function

Nomenclature & Linear Activation Function

- **Superscipt** represents the layer
- **Subscript** the sequence in that layer



$$Z^{[1]} = W^{[1]T} X^{[1]} + B^{[1]}$$

$$a^{[1]} = g(Z^{[1]})$$

$$a^{[1]} = C^1 * Z^{[1]}$$

$$a^{[1]} = C^1 * (W^{[1]T} X^{[1]} + B^{[1]})$$

$$a^{[1]} = C^1 * W^{[1]T} X^{[1]} + C^1 * B^{[1]}$$

$$Z^{[2]} = W^{[2]T} a^{[1]} + B^{[2]}$$

$$Z^{[2]} = W^{[2]T} (C^1 * W^{[1]T} X^{[1]} + C^1 * B^{[1]}) + B^{[2]}$$

$$a^{[2]} = g(Z^{[2]})$$

$$a^{[2]} = C^2(Z^{[2]})$$

$$a^{[2]} = C^2(W^{[2]T} (C^1 * W^{[1]T} X^{[1]} + C^1 * B^{[1]}) + B^{[2]})$$

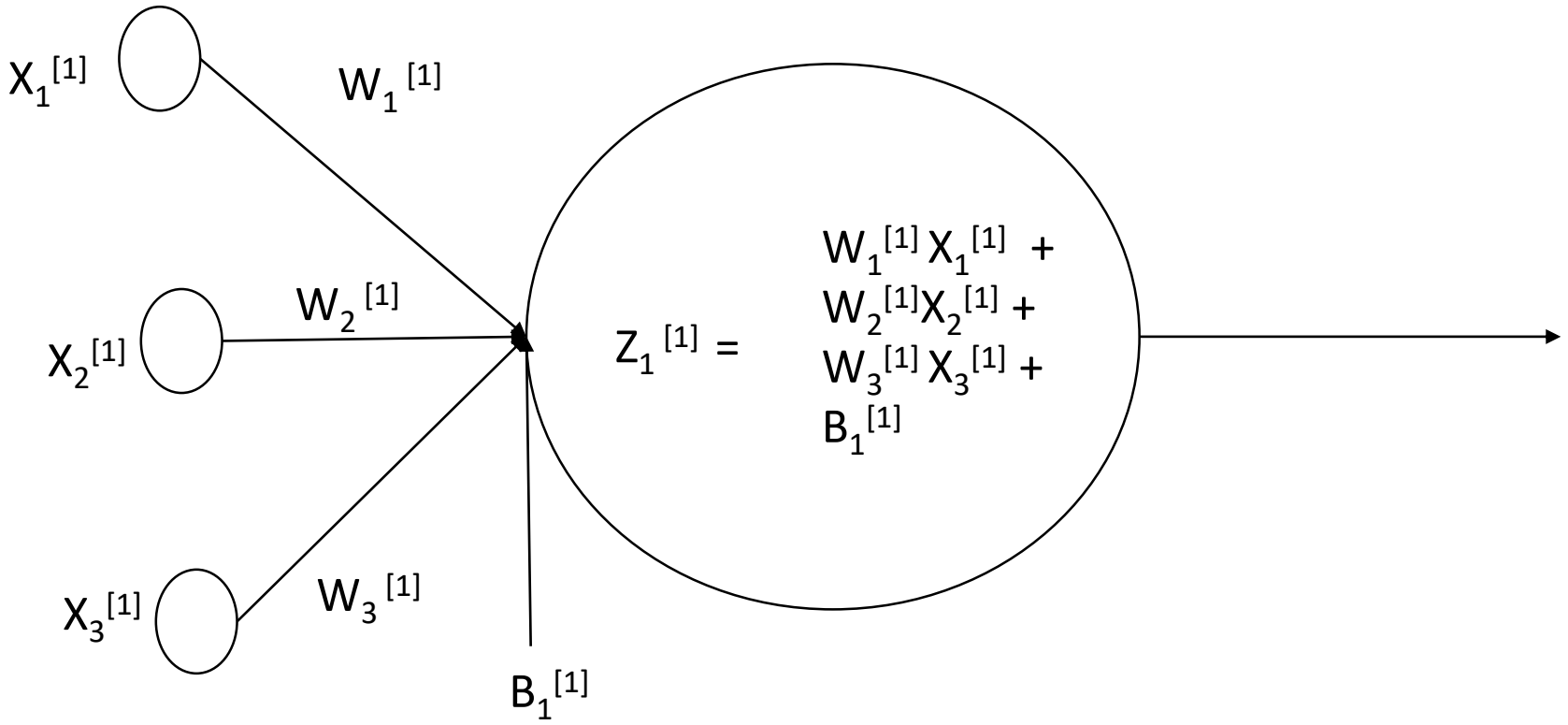
$$a^{[2]} = C^2* W^{[2]T} *C^1 * W^{[1]T} X^{[1]} + C^2* W^{[2]T} *C^1 * B^{[1]} + C^2B^{[2]})$$

$$a^{[2]} =C^2* W^{[2]T} *C^1 * B^{[1]} + C^2B^{[2]}) + C^2* W^{[2]T} *C^1 * W^{[1]T} X^{[1]}$$

$$a^{[2]} =B^* + W^* X^{[1]}$$

Linear Regression

$$Z^{[1]} = W^{[1]T} X^{[1]} + B^{[1]}$$

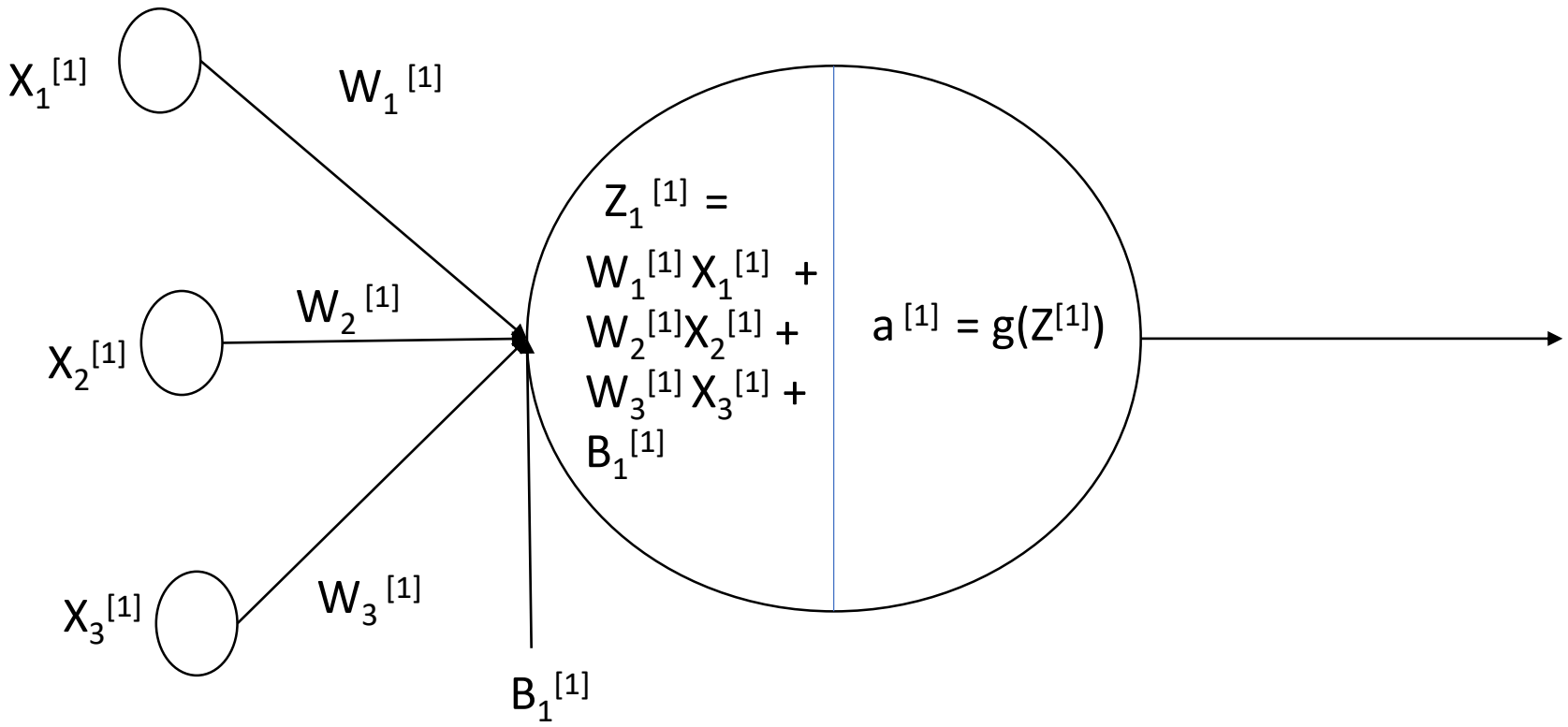


Input Hidden Output

Activation Function

$$Z^{[1]} = W^{[1]T} X^{[1]} + B^{[1]}$$

$$a^{[1]} = g(Z^{[1]})$$



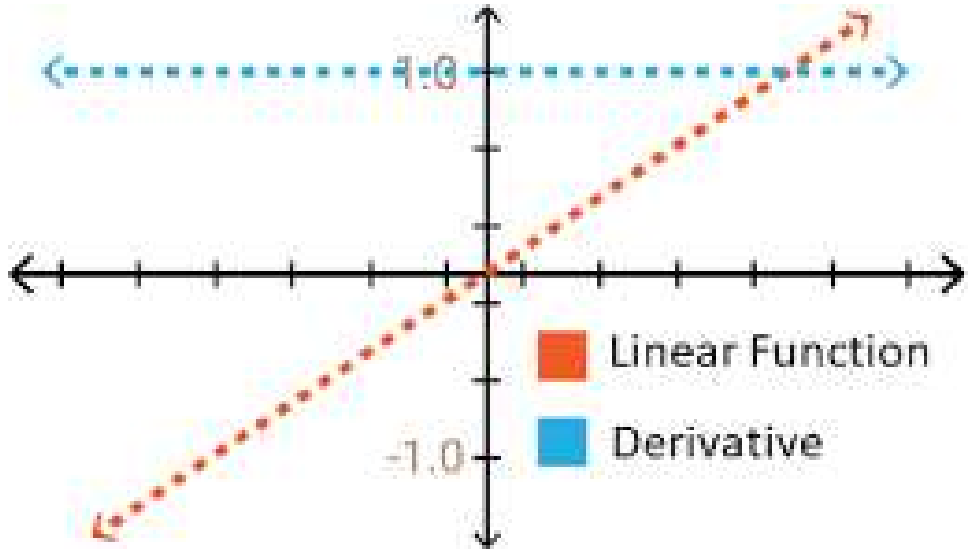
Note:

- **Superscript** Represents the Layer
- **Subscript** Represents the elements(Weight, activation fn, etc) of each layer

Linear

$$f(x) = ax + b$$

$$\frac{df(x)}{dx} = a$$



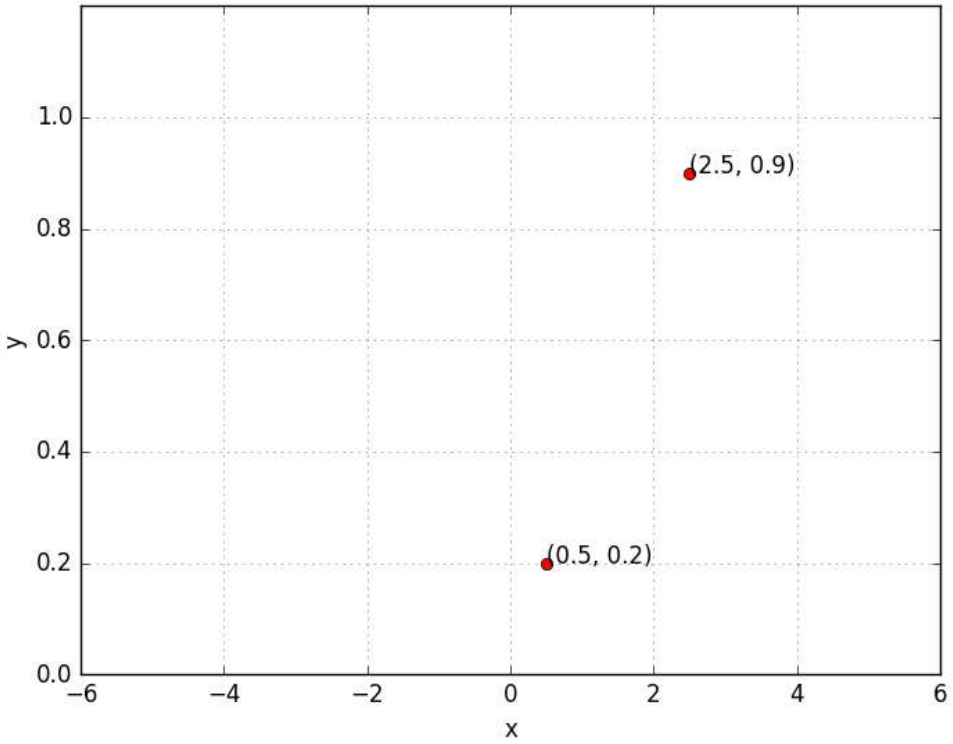
Observations:

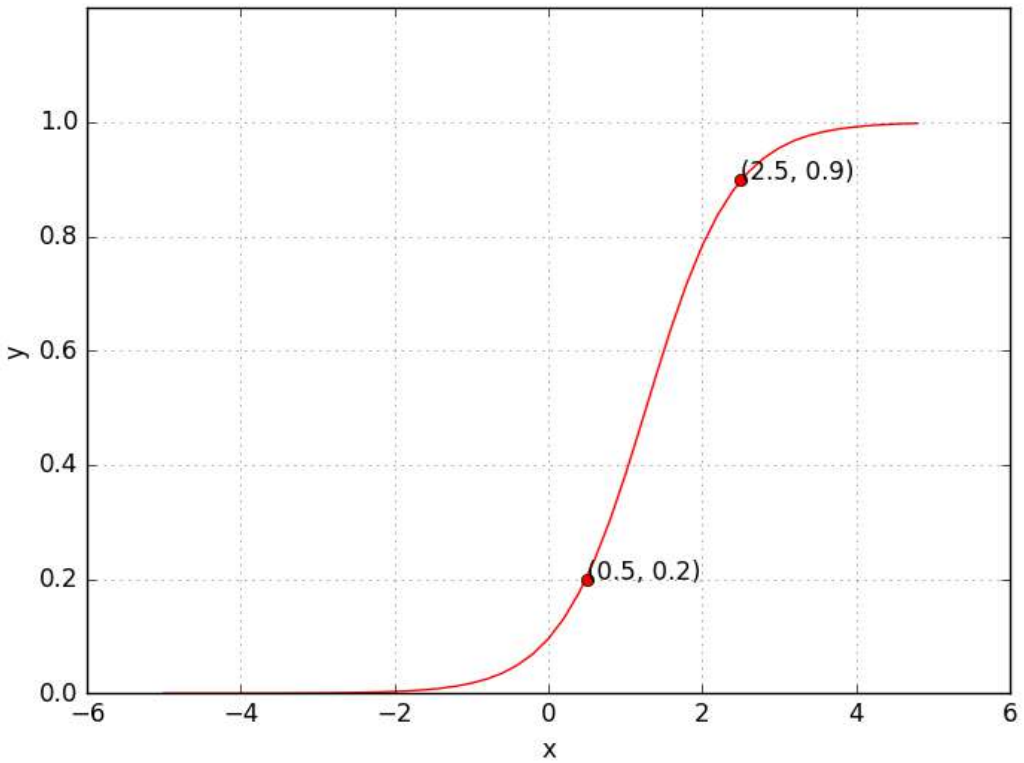
- Constant gradient
- Gradient does not depend on the change in the input

Non-Linear

- Sigmoid (Logistic)
- Hyperbolic Tangent (Tanh)
- Rectified Linear Unit (ReLU)
  - *Leaky Relu*
  - *Parametric Relu*
- Exponential Linear Unit (ELU)

Sigmoid Activation Functions (Logistics)





Suppose we train the network with $(x, y) = (0.5, 0.2)$ and $(2.5, 0.9)$

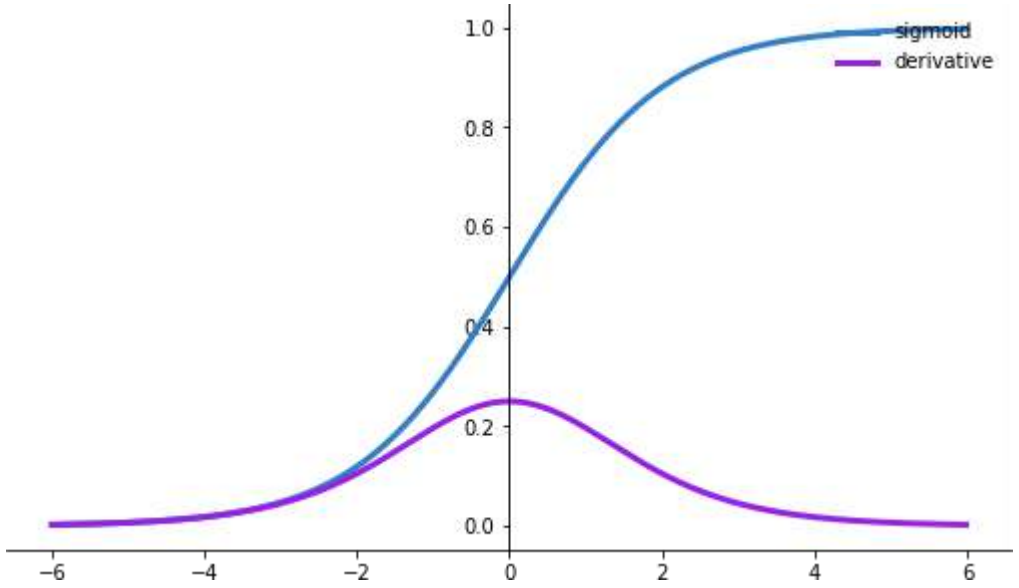
At the end of training we expect to find w^*, b^* such that:

$f(0.5) \rightarrow 0.2$ and $f(2.5) \rightarrow 0.9$

Sigmoid Activation Functions (Logistics)

$$f(x) = \frac{1}{1 + e^{-x}}$$

$$\frac{df(x)}{dx} = f(x)(1 - f(x))$$



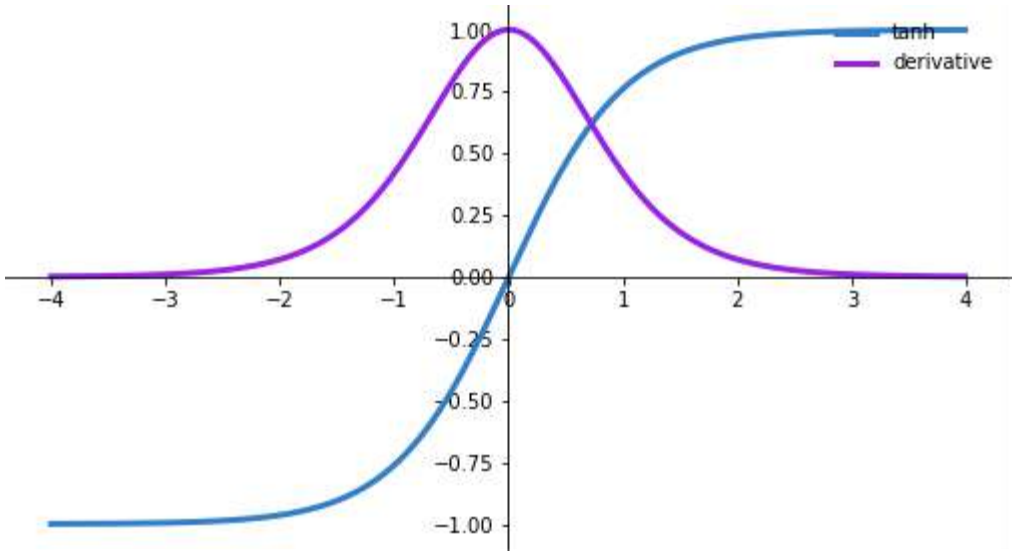
Observations:

- Output: 0 to 1
- Outputs are not zero-centered
- Can saturate and kill (vanish) gradients
- Derivative ranges from 0 to 0.25
- Can saturate and kill (vanish) gradients
- Large or very small inputs, the sigmoid function approaches either 0 or 1. The derivative of the function might become 0.

Tanh Activation Function

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

$$\frac{df(x)}{dx} = 1 - f(x)^2$$



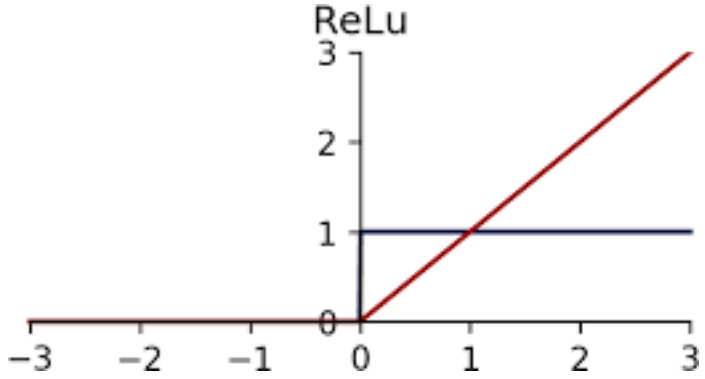
Observations:

- Output: -1 to +1
- Outputs are zero-centered
- Derivative ranges from 0 to 1
- Can Saturate and kill (vanish) gradients
- Gradient is more steeped than Sigmoid, resulting in faster convergence

Rectified Linear Unit(ReLU)

$$f(x) = \max(0, x)$$

$$\frac{df(x)}{dx} = \begin{matrix} 0 & x < 0 \\ 1, & x \geq 0 \end{matrix}$$



Observations:

- Its of linear nature
- Greatly increase training speed compared to tanh and sigmoid
- Reduces likelihood of killing(vanishing) gradient
- Might Lead to Exploding Gradient
- Dead nodes (Does not update their weights during training)
- Ranges from 0 to infinity

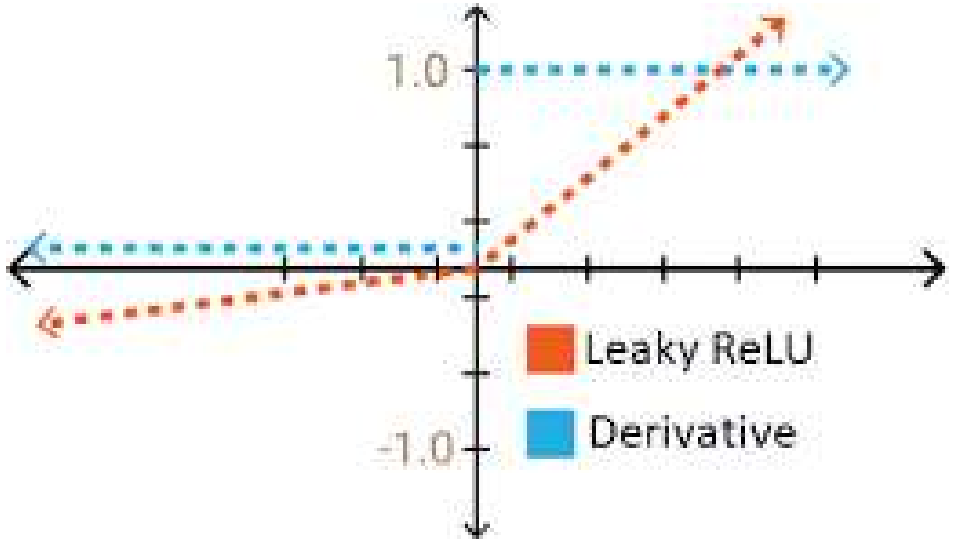
Some Variants of ReLU:

- Leaky ReLU-
- Parameterized ReLU (PReLU)-
- Exponential Linear Unit (ELU)-

Leaky-ReLU

$$f(x) = \max(0.01x, x)$$

$$\frac{df(x)}{dx} = \begin{cases} 0.01, & x < 0 \\ 1, & x \geq 0 \end{cases}$$



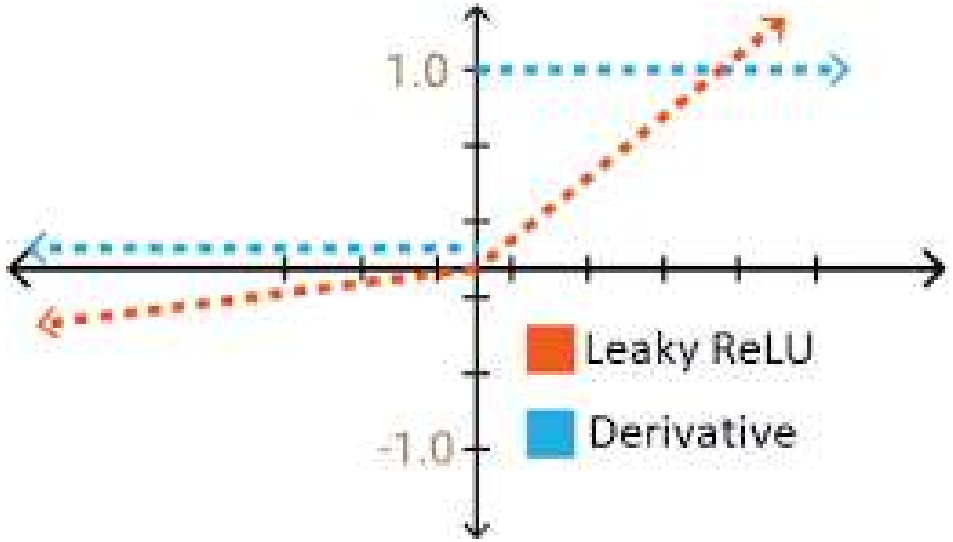
Observations:

- It prevents the neuron from dying
- More useful for Deep Networks
- Coefficient is fixed 0.01
- Slope for the negative values are fixed

Parameterized-ReLU

$$f(x) = \max(\alpha x, x)$$

$$\frac{df(x)}{dx} = \begin{matrix} \alpha, & x < 0 \\ 1, & x \geq 0 \end{matrix}$$



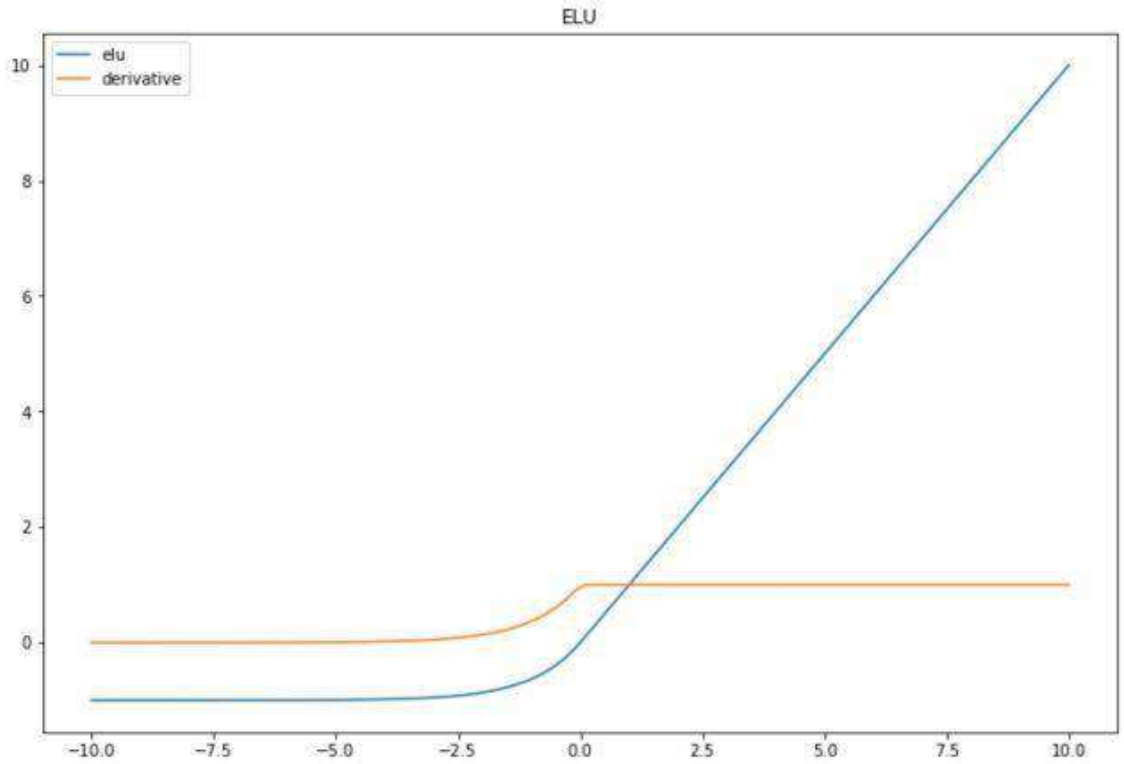
Observations:

- Another Hyperparameter, Learns the slope for the negative values
- Convergence speed is better as compared to Leaky ReLU

Exponential Linear Unit (ELU)

$$f(x) = \begin{cases} \alpha(e^x - 1), & x < 0 \\ x & x \geq 0 \end{cases}$$

$$\frac{df(x)}{dx} = \begin{cases} f(x) + \alpha, & x < 0 \\ 1, & x \geq 0 \end{cases}$$



Observations:

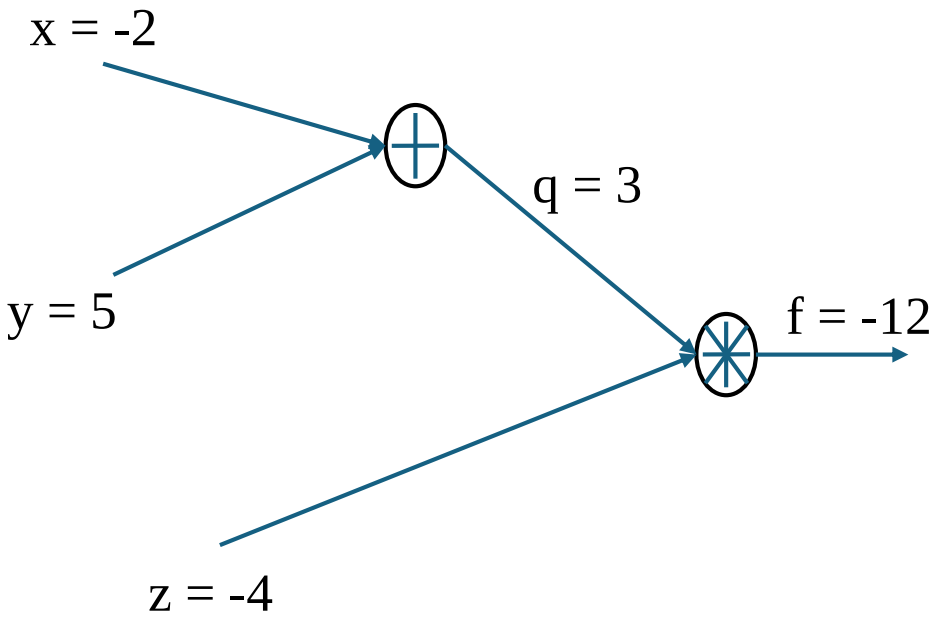
- It can produce –ve output
- It can blow up activation function
- α should be positive
- Derivative ranges from 0 to infinity

Summary

- We learn characteristics of different Activation Functions and their gradient
- The choice of activation function depend on the nature of the problem, nature of the target output and the deepness of the network.

Forward Pass

Forward Pass

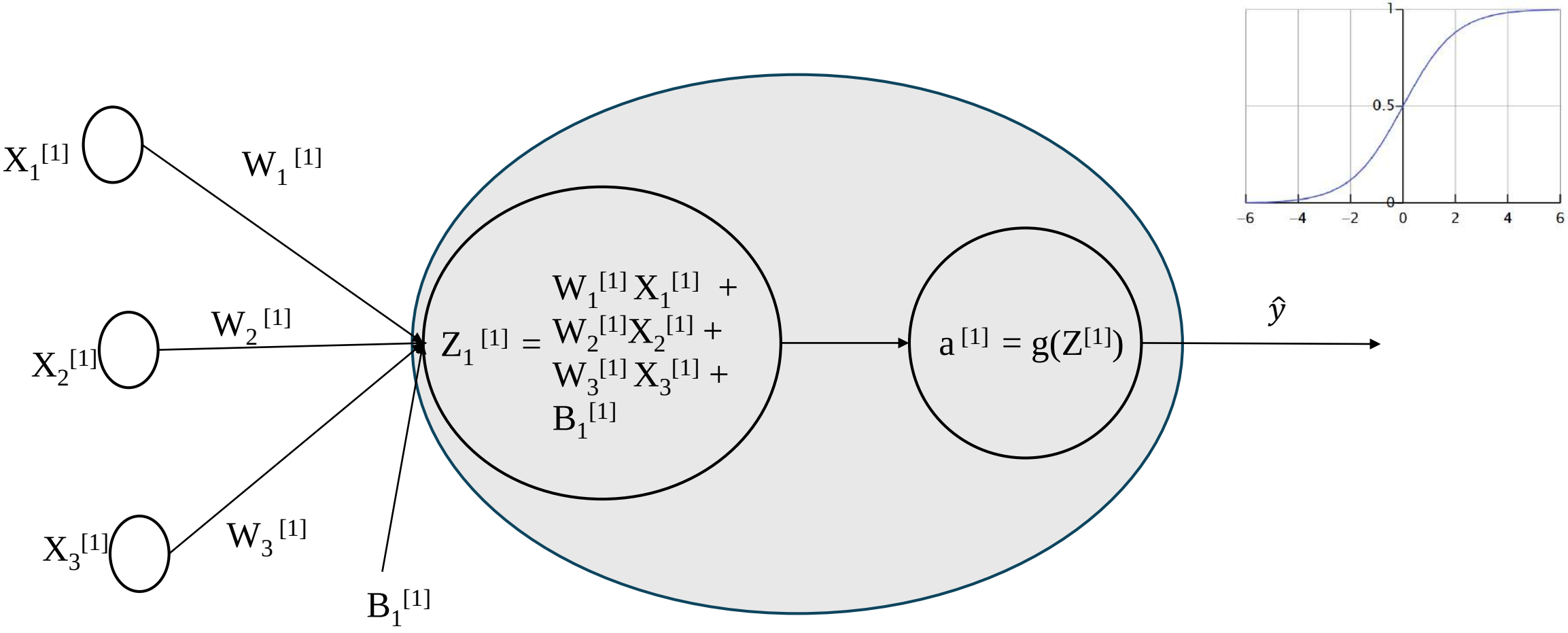


$$f(x, y, z) = (x + y)z$$

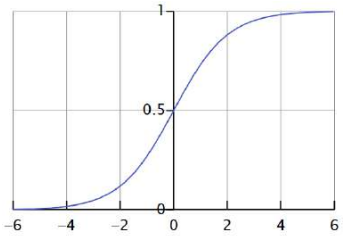
$$q = x + y$$

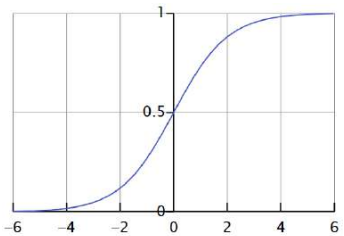
$$f = q \times y$$

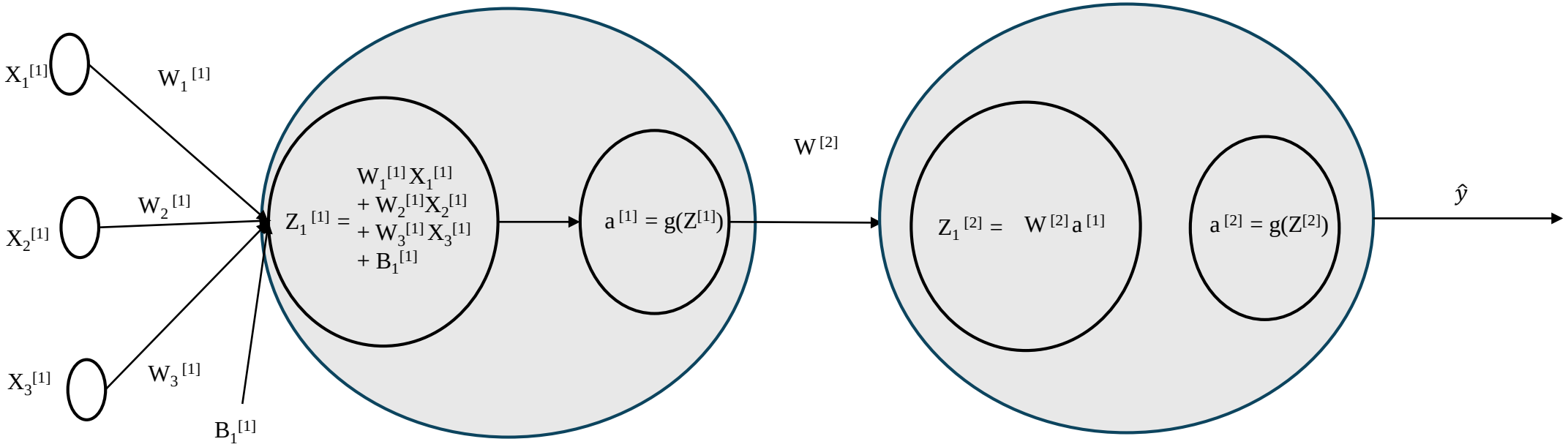
Activation Function



Forward Pass





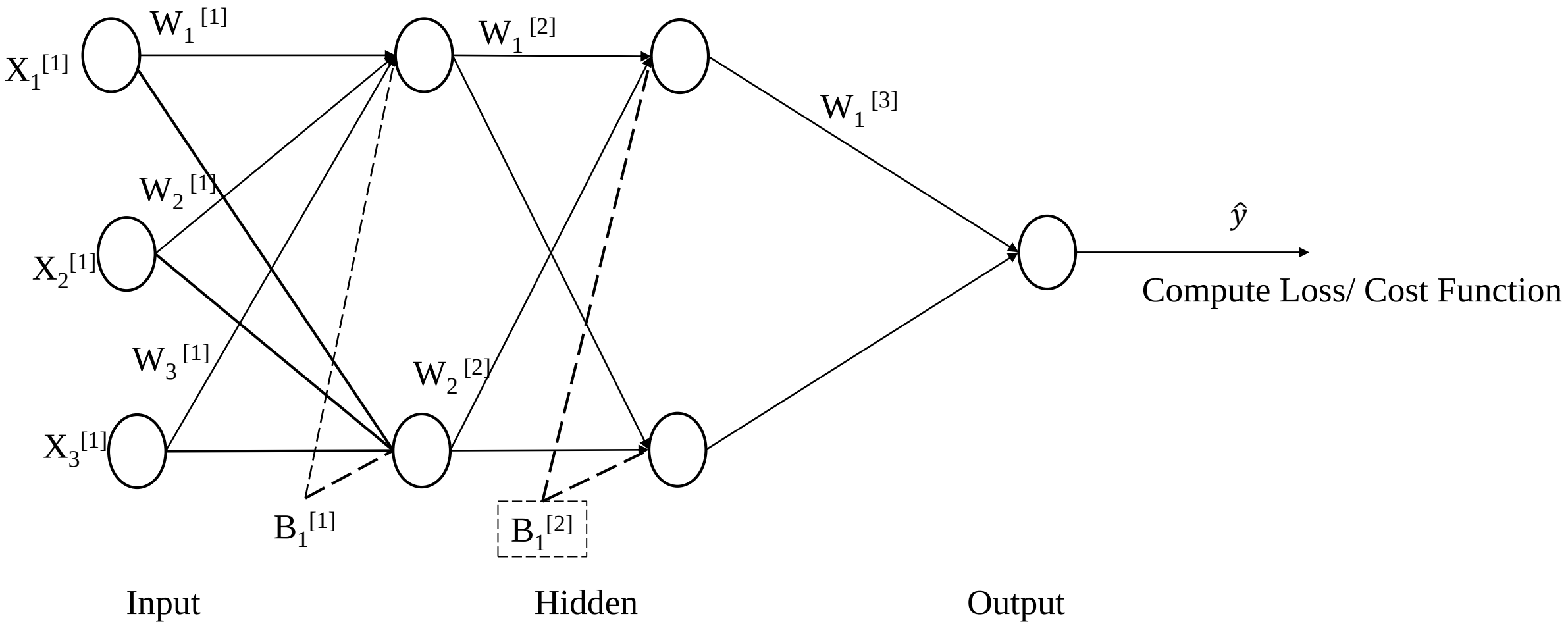


Input

2 Hidden Unit

Output

Deep Network



Loss Vs Cost Function

Loss Vs Cost Function

- **Loss function**
  - A **loss function** (also known as an **error function** or **objective function**) quantifies how well a machine learning model's predictions match the actual target values (ground truth)
  - Helps us to understand the difference between the predicted value and the actual value
  - Error for a single training example

$$AE = |y - \hat{y}|$$

- **Cost function**
  - Average of loss function on the entire training dataset
  - Types of cost function
    - Regression
    - Binary Classification
    - Multi-Class Classification

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y - \hat{y}|$$

Cost Function

Mean Absolute Error	$MAE = \frac{1}{n}\sum_{i=1}^{n} \lvert y_i - \hat{y}_i \rvert$
Mean Squared Error	$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
Root Mean Square Error	$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$
Cross Entropy	$CE = -\frac{1}{n}\sum_{i=1}^{n} (y_i) log(\hat{y}_i)$
Binary Cross Entropy	$BCE = -\frac{1}{n}\sum_{i=1}^{n} [(y_i) log(\hat{y}_i) + [(1 - y_i) log(1 - \hat{y}_i)]$

Cost Function Regression

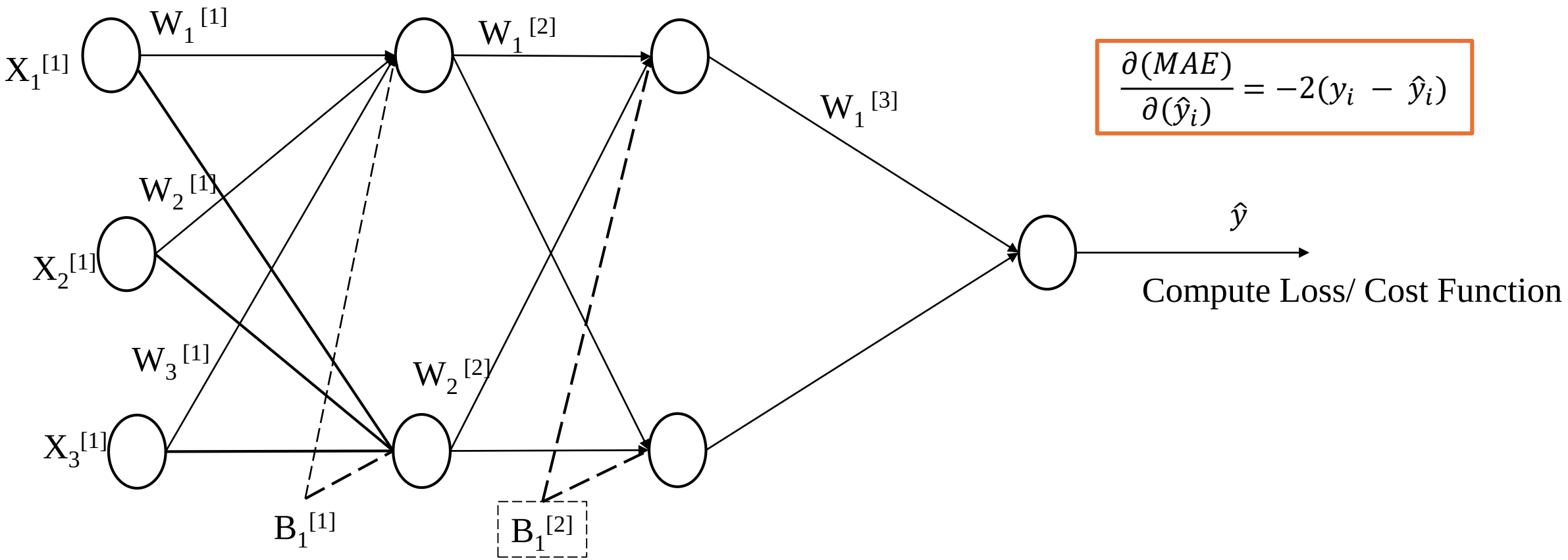
Cost Functions	Type	• Pros	Cons
Mean Absolute Error (MAE)	Regression	• Robust to outliers • Intuitive interpretation • Represents the average absolute error	• Not differentiable at zero. • Ignores error direction (positive or negative).
Mean Squared Error (MSE)	Regression	• Differentiable and widely used • Penalizes large errors more than small errors	• Sensitive to outliers • Units are squared
Root Mean Squared Error (RMSE)	Regression	• Same unit as the target variable • Penalizes large errors but not excessively.	• Sensitive to outliers • Less robust for skewed distributions

Cost Function

Cost Functions	Type	• Pros	Cons
Cross-Entropy	Classification	• Suitable for multi-class classification. • Clear probabilistic interpretation.	• Requires calibrated probabilities to work effectively
Binary Cross-Entropy	Classification	• Suitable for binary classification tasks • Encourages well-calibrated probabilities	• Can suffer from vanishing gradients • Not ideal for multi-class problems
Softmax Cross-Entropy	Classification	• Appropriate for multi-class classification • Encourages class separation	• Requires one-hot encoded labels • Sensitive to outliers. • May not work well with imbalanced data.

Gradient Descent

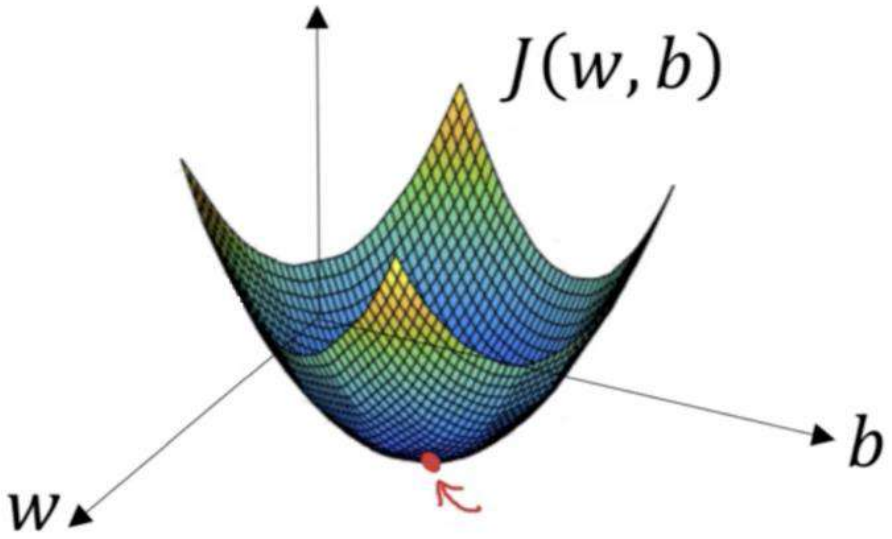
Deep Network

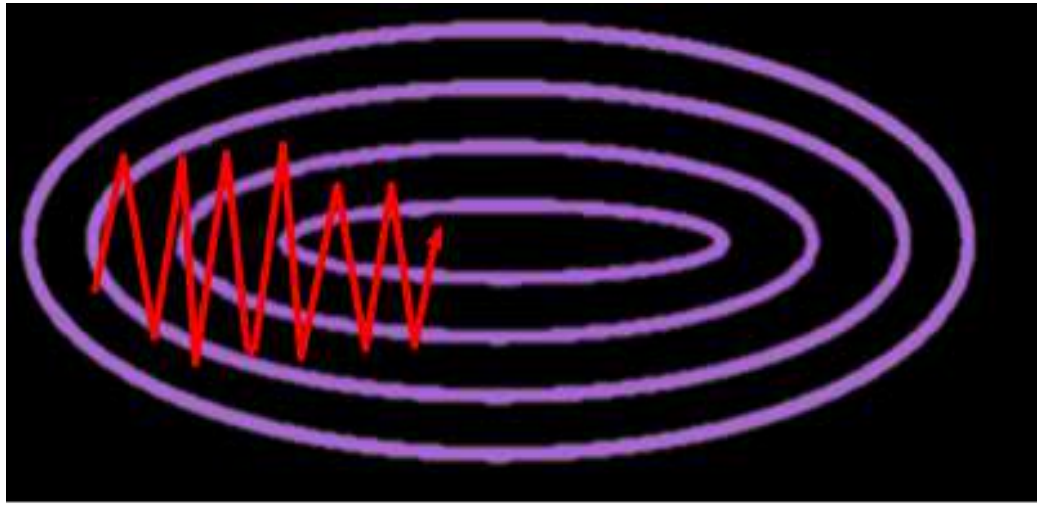


$$\frac{\partial(MAE)}{\partial(\hat{y}_i)} = -2(y_i - \hat{y}_i)$$

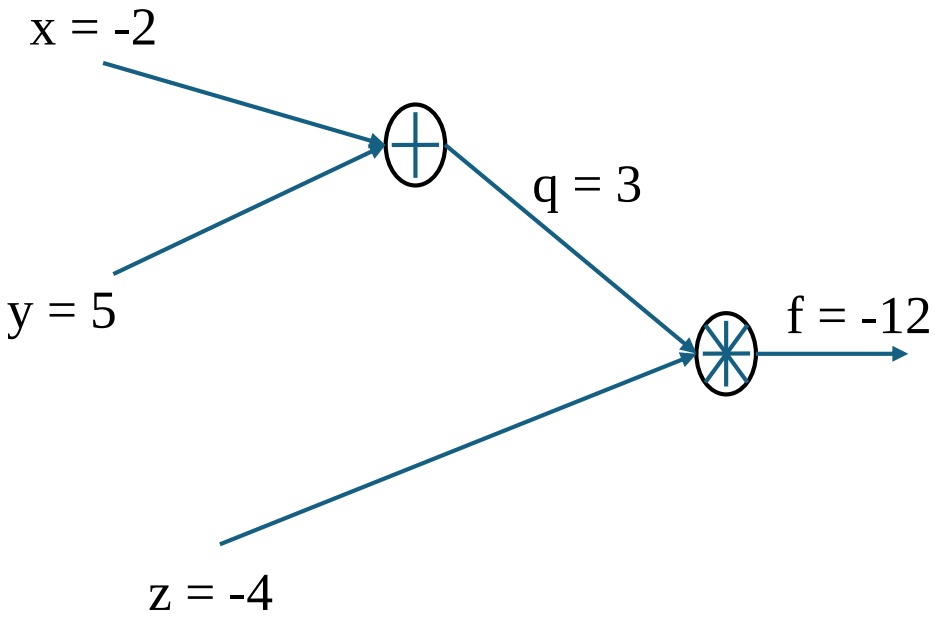
Gradient Descent

- It is an optimization algorithm
- Minimizes a given function to its local minima
- It is a measure of change in weights with respect to change in error (Slope function)
- If slope/gradient is zero then model stops learning





Gradient



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

$$q = x + y$$

$$f = q \times y$$

Gradient from Previous Layer

Local Gradient

Chain Rule

$$\frac{\partial(f)}{\partial(x)} = \frac{\partial(f)}{\partial(q)} \; x \; \frac{\partial(q)}{\partial(x)} = -4 \text{ x } 1 = -4$$

$$\frac{\partial(f)}{\partial(y)} = \frac{\partial(f)}{\partial(q)} \; x \; \frac{\partial(q)}{\partial(y)} = -4 \text{ x } 1 = -4$$

$$\frac{\partial(f)}{\partial(f)} = 1$$

$$\frac{\partial(q)}{\partial(x)} = 1$$

$$\frac{\partial(q)}{\partial(y)} = 1$$

$$\frac{\partial(f)}{\partial(q)} = z = -4$$

$$\frac{\partial(f)}{\partial(z)} = q = 3$$

Gradient Descent Variants

Batch GD	$\theta = \theta - \eta \cdot \nabla J(\theta)$	• Guarantees convergence to the global minimum. • Stable and smooth updates.	• Slow for large datasets. • Requires loading the entire dataset into memory.
Stochastic GD	$\theta = \theta - \eta \cdot \nabla J(\theta, x^i y^i)$	• Faster iteration speed. • Can escape local minima (due to noisy updates). • Suitable for online learning.	• Convergence can be noisy and less stable. • Requires careful tuning of the learning rate.
Mini – Batch GD	$\theta = \theta - \eta \cdot \frac{1}{m}\sum_{i=1}^{m} \nabla J(\theta, x^i y^i)$	• Balance between Batch GD and SGD: faster than Batch GD and more stable than SGD. • Efficient computation using vectorization.	• Still needs tuning of learning rate and batch size. • Less stable than full Batch GD.
Momentum	$v_t = \gamma v_{t-1} - \eta \cdot \nabla J(\theta)$ $\theta = \theta - v_t$	• Accelerates convergence for high-curvature, small, and noisy gradients. • Reduces oscillation.	• Requires additional hyperparameter (momentum coefficient γ). • Can overshoot minima.

Backpropagation

Backpropogation

- **Following are the steps:**
  - Initialize weights randomly
  - Compute the Cost Function
  - Compute the gradient
  - Update the weights

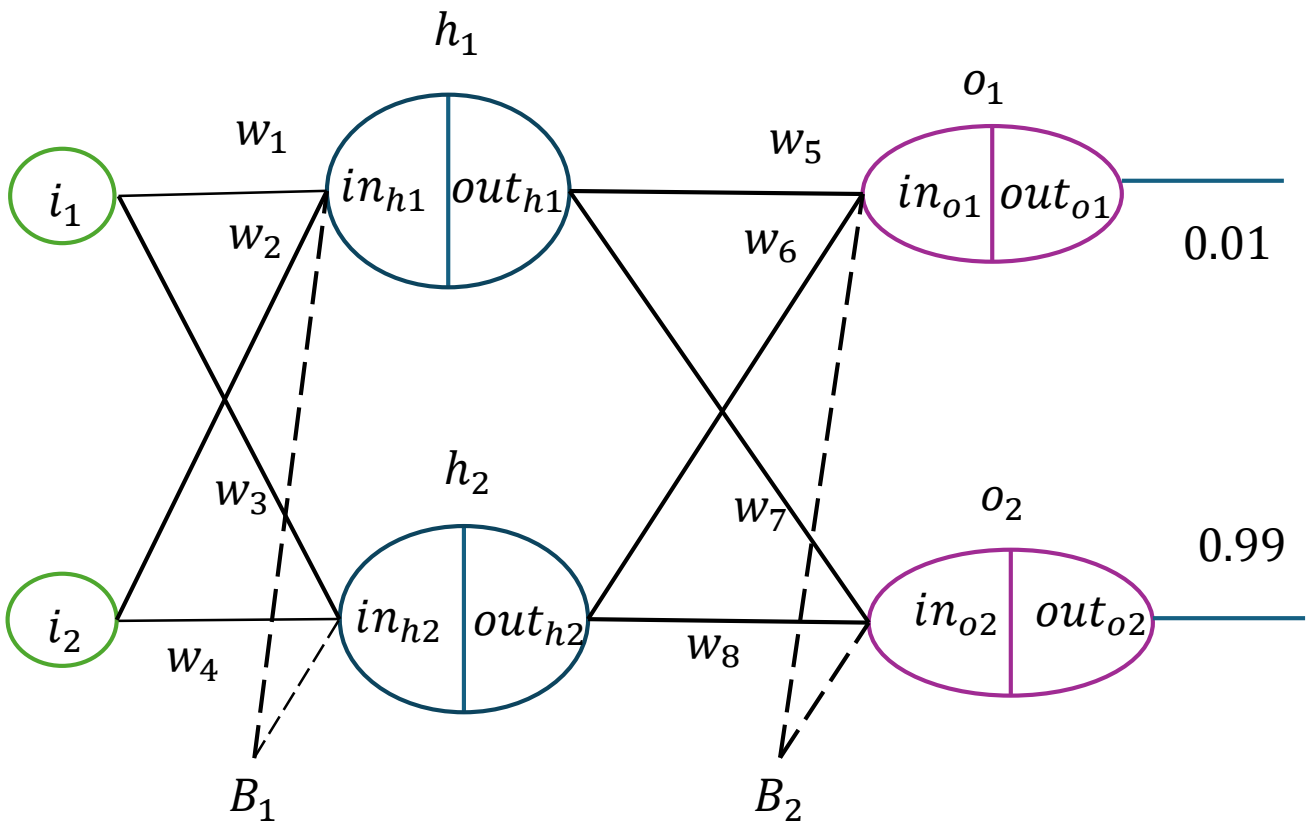
$$W = W - \alpha \frac{\partial(J(W,B))}{\partial(W)}$$

$$B = B - \alpha \frac{\partial(J(W,B))}{\partial(B)}$$

Derivatives of Cost Function

Mean Absolute Error	$MAE = \frac{1}{n}\sum_{i=1}^{n} \lvert y_i - \hat{y}_i \rvert$	$\frac{\partial(MAE)}{\partial(\hat{y}_i)} = \begin{matrix}\{1 \lvert y_i < \hat{y}_i\} \\ \{-1 \lvert y_i > \hat{y}_i\}\end{matrix}$
Mean Squared Error	$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$	$\frac{\partial(MAE)}{\partial(\hat{y}_i)} = -2(y_i - \hat{y}_i)$
Root Mean Square Error	$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$	$\frac{\partial(RMSE)}{\partial(\hat{y}_i)} = \frac{1}{RMSE} \cdot \frac{1}{n}(y_i - \hat{y}_i)$
Cross Entropy	$CE = -\frac{1}{n}\sum_{i=1}^{n} (y_i) log(\hat{y}_i)$	$\frac{\partial(CE)}{\partial(\hat{y}_i)} = \begin{matrix}\left\{-\frac{1}{\hat{y}_i} \middle\vert \hat{y}_i = 1\right\} \\ \{0 \lvert\, \hat{y}_i = 0\}\end{matrix}$
Binary Cross Entropy	$BCE = -\frac{1}{n}\sum_{i=1}^{n} [(y_i) log(\hat{y}_i) + [(1 - y_i) log(1 - \hat{y}_i)]$	$\frac{\partial(BCE)}{\partial(\hat{y}_i)} = -\frac{y_i}{\hat{y}_i} + \frac{1 - y_i}{1 - \hat{y}_i}$

Forward Pass Layer 1



$$in_{h1} = w_1 i_1 + w_2 i_2 + b_1 = 0.3775$$

$$in_{h2} = w_3 i_1 + w_4 i_2 + b_1 = 0.3925$$

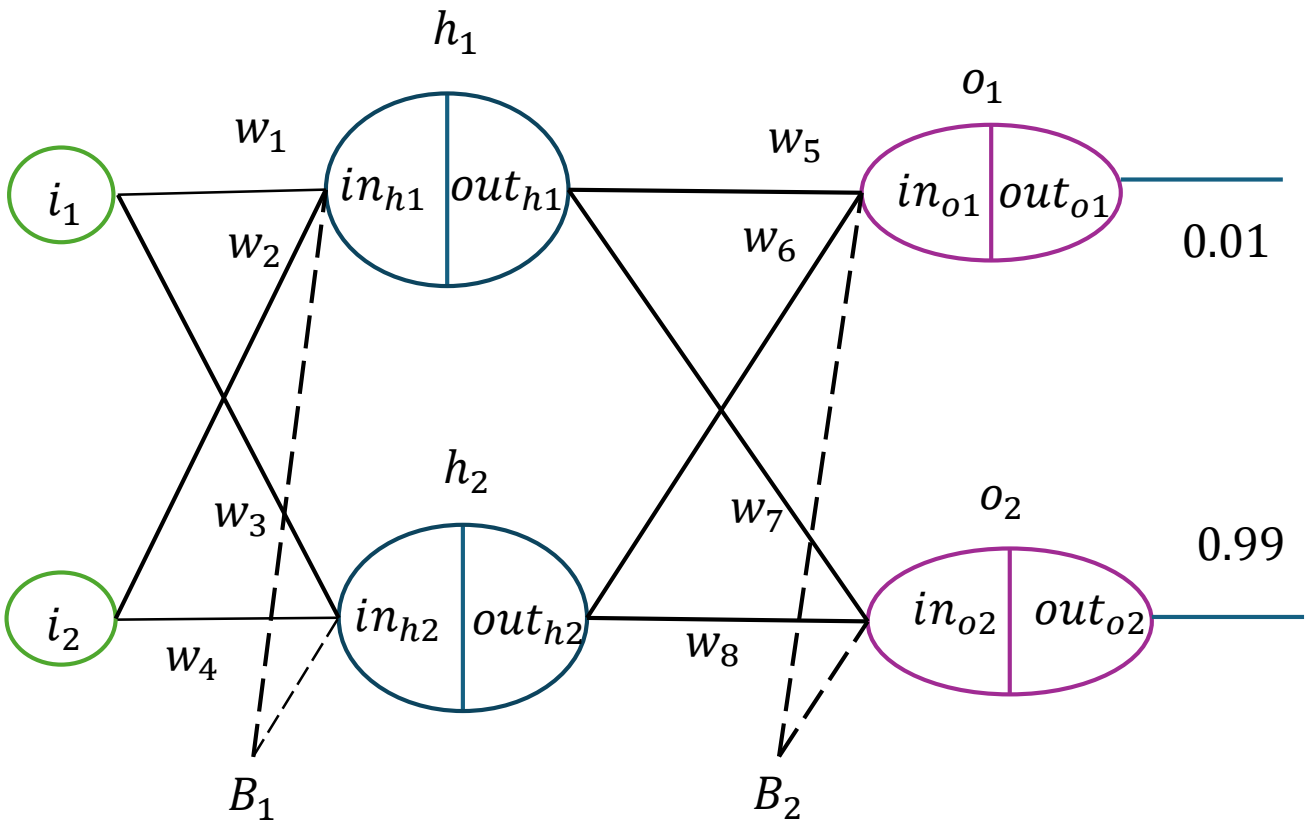
$$out_{h1} = \frac{1}{1 + e^{-in_{h1}}} = 0.593269992$$

$$out_{h2} = \frac{1}{1 + e^{-in_{h2}}} = 0.596884378$$

$i_1 = 0.05$ $w_1 = 0.15$ $w_2 = 0.20$ $w_3 = 0.25$ $w_4 = 0.30$ $B_1 = 0.35$

$i_2 = 0.10$ $w_5 = 0.40$ $w_6 = 0.45$ $w_7 = 0.50$ $w_8 = 0.55$ $B_2 = 0.60$

Forward Pass Layer 2



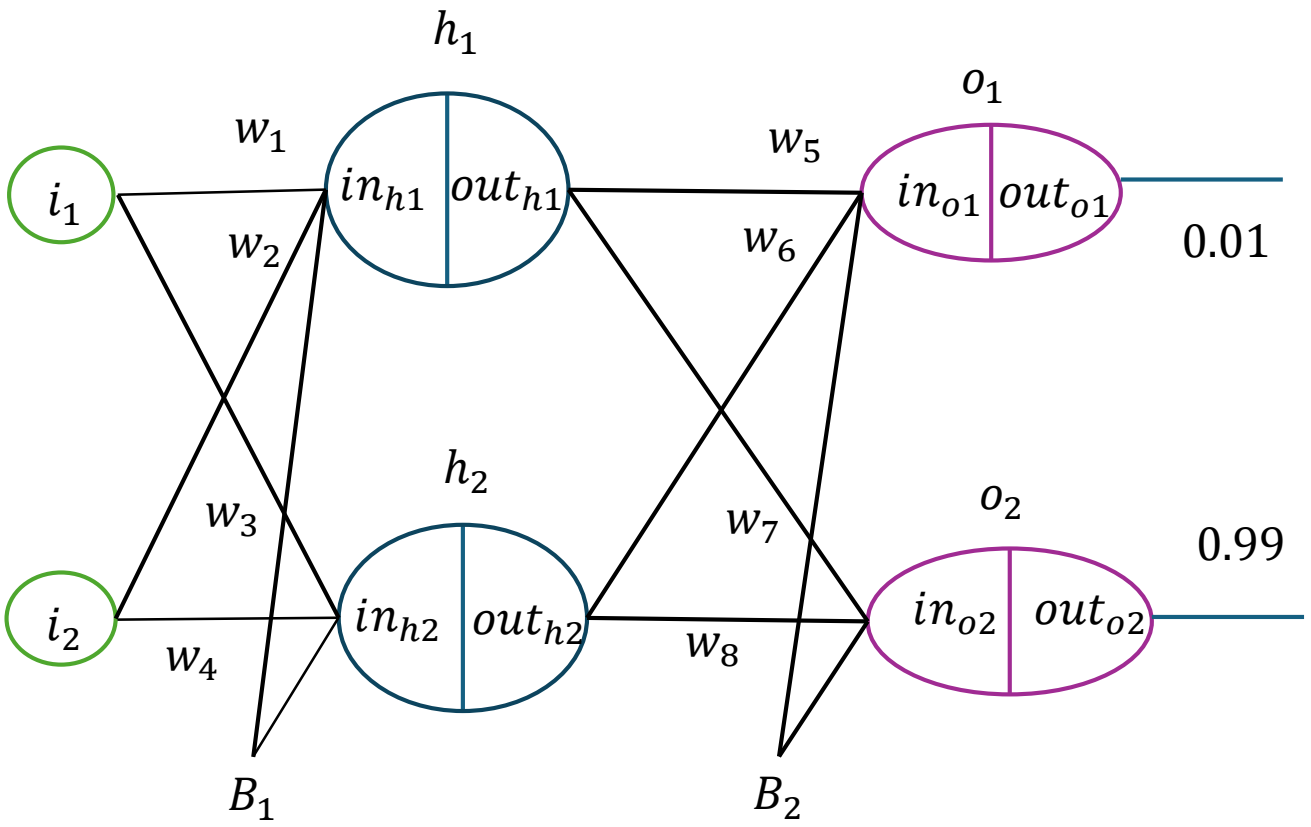
$$in_{o1} = w_5 out_{h1} + w_6 out_{h2} + b_2 = 1.105905967$$

$$in_{o2} = w_7 out_{h1} + w_8 out_{h2} + b_2 = 1.224921404$$

$$out_{o1} = \frac{1}{1 + e^{-in_{o1}}} = 0.75136507$$

$$out_{o2} = \frac{1}{1 + e^{-in_{o2}}} = 0.772928465$$

Forward Pass Error (Cost Function)

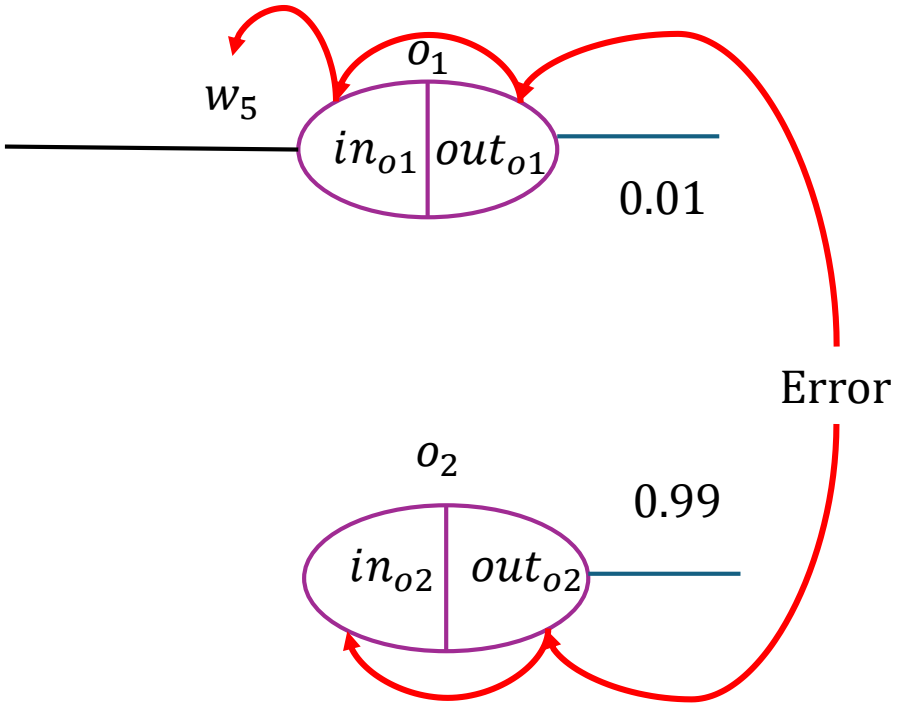


$$Error_{o1} = \frac{1}{2}(out_{o1} - out1)^2 = 0.274811083$$

$$Error_{o2} = \frac{1}{2}(out_{o2} - out2)^2 = 0.023560026$$

$$Total\ Error = Error_{o1} + Error_{o2} = 0.298371109$$

Backpropagation



$$\frac{\partial Error_{o1}}{\partial w_5} = \frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}} \text{ x } \frac{\partial in_{o1}}{\partial w_5}$$

$$\frac{\partial Error_{o1}}{\partial out_{o1}} = -(out1 - out_{o1}) = 0.74136507$$

$$\frac{\partial out_{o1}}{\partial in_{o1}} = out_{o1}(1 - out_{o1}) = 0.186815602$$

$$\frac{\partial in_{o1}}{\partial w_5} = out_{h1} = 0.593269992$$

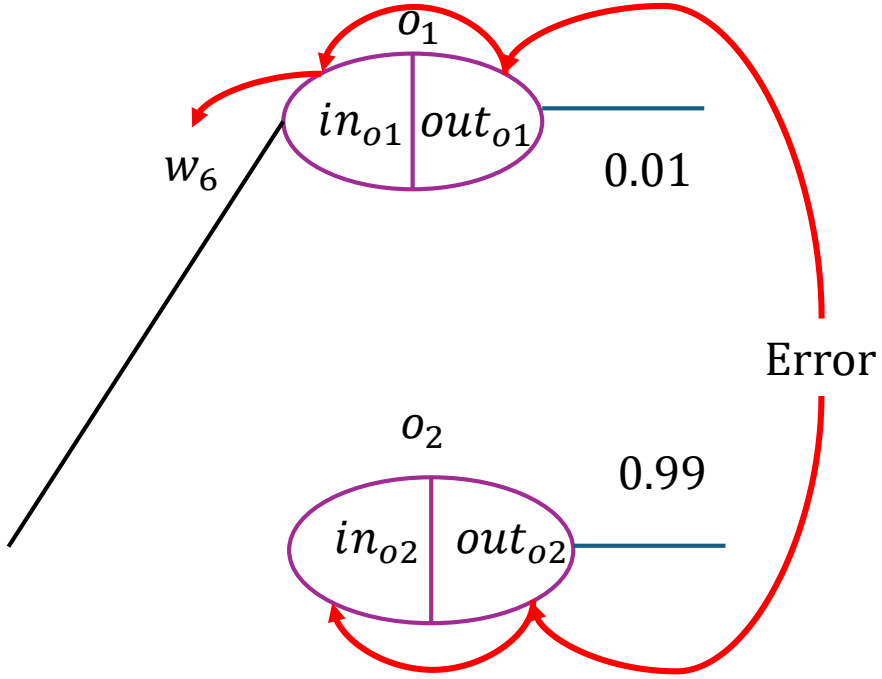
$$\frac{\partial Error}{\partial w_5} = \frac{\partial Error_{o1}}{\partial w_5} + \frac{\partial Error_{o2}}{\partial w_5}$$

$$\frac{\partial Error}{\partial w_5} = 0.74136507 \text{ x } 0.186815602 \text{ x } 0.593269992$$

$$\frac{\partial Error}{\partial w_5} = 0.082167041$$

BackPropagation

$$\frac{\partial Error_{o1}}{\partial w_6} = \frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}} \text{ x } \frac{\partial in_{o1}}{\partial w_6}$$



$$\frac{\partial Error_{o1}}{\partial out_{o1}} = -(out1 - out_{o1}) = 0.74136507$$

$$\frac{\partial out_{o1}}{\partial in_{o1}} = out_{o1}(1 - out_{o1}) = 0.186815602$$

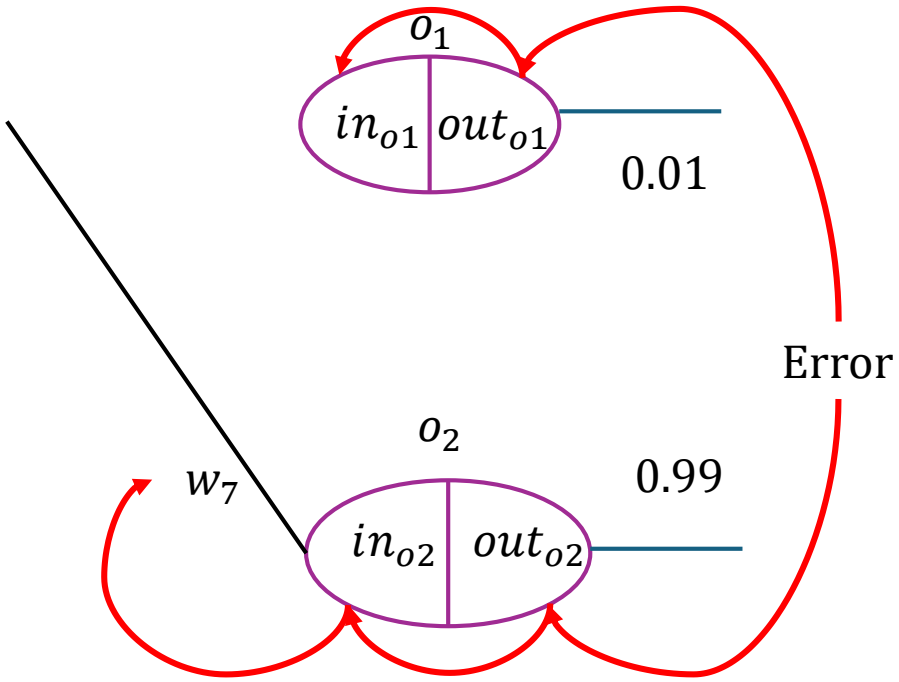
$$\frac{\partial in_{o1}}{\partial w_6} = \text{out}_{\text{h2}} = 0.596884378$$

$$\frac{\partial Error}{\partial w_6} = \frac{\partial Error_{o1}}{\partial w_6} + \frac{\partial Error_{o2}}{\partial w_6}$$

$$\frac{\partial Error_{o1}}{\partial w_6} = 0.7413650 \text{ x } 0.186815602 \text{ x } 0.596884378$$

$$\frac{\partial Error}{\partial 6} = 0.082667628$$

BackPropagation



$$\frac{\partial Error}{\partial w_7} = \frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}} \text{ x } \frac{\partial in_{o2}}{\partial w_7}$$

$$\frac{\partial Error_{o2}}{\partial out_{o2}} = -(out2 - out_{o2}) = \text{-0.217071535}$$

$$\frac{\partial out_{o2}}{\partial in_{o2}} = out_{o2}(1 - out_{o1}) = 0.175510053$$

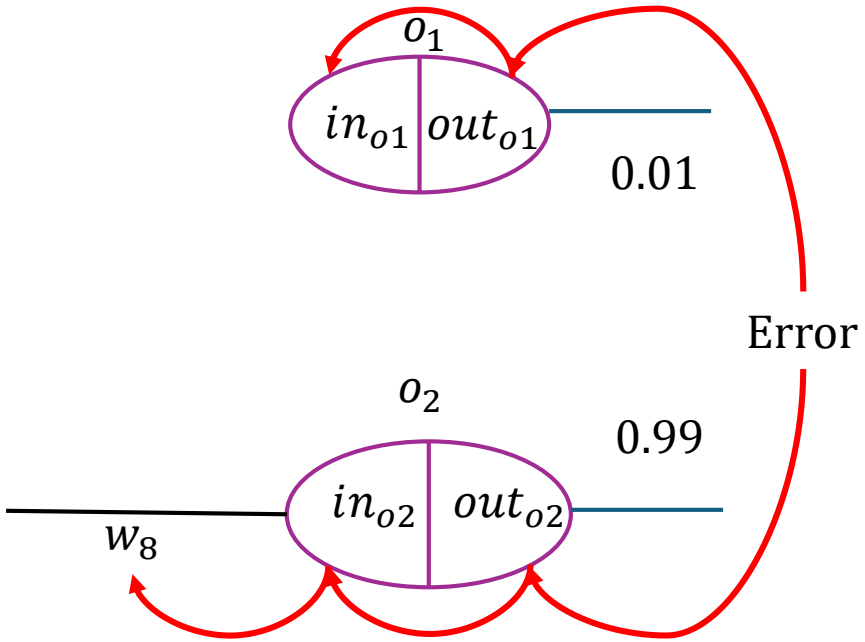
$$\frac{\partial in_{o2}}{\partial 7} = out_{h1} = 0.593269992$$

$$\frac{\partial Error_{o2}}{\partial w_7} = \text{-0.217071535 x0.175510053 x 0.593269992}$$

$$\frac{\partial Error}{\partial w_7} = \frac{\partial Error_{o1}}{\partial w_7} + \frac{\partial Error_{o2}}{\partial w_7}$$

$$\frac{\partial Error}{\partial w_7} = \text{-0.02260254}$$

BackPropagation



$$\frac{\partial Error}{\partial w_8} = \frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}} \text{ x } \frac{\partial in_{o2}}{\partial w_8}$$

$$\frac{\partial Error_{o2}}{\partial out_{o2}} = -(out2 - out_{o2}) = \text{-0.217071535}$$

$$\frac{\partial out_{o2}}{\partial in_{o2}} = out_{o2}(1 - out_{o2}) = 0.175510053$$

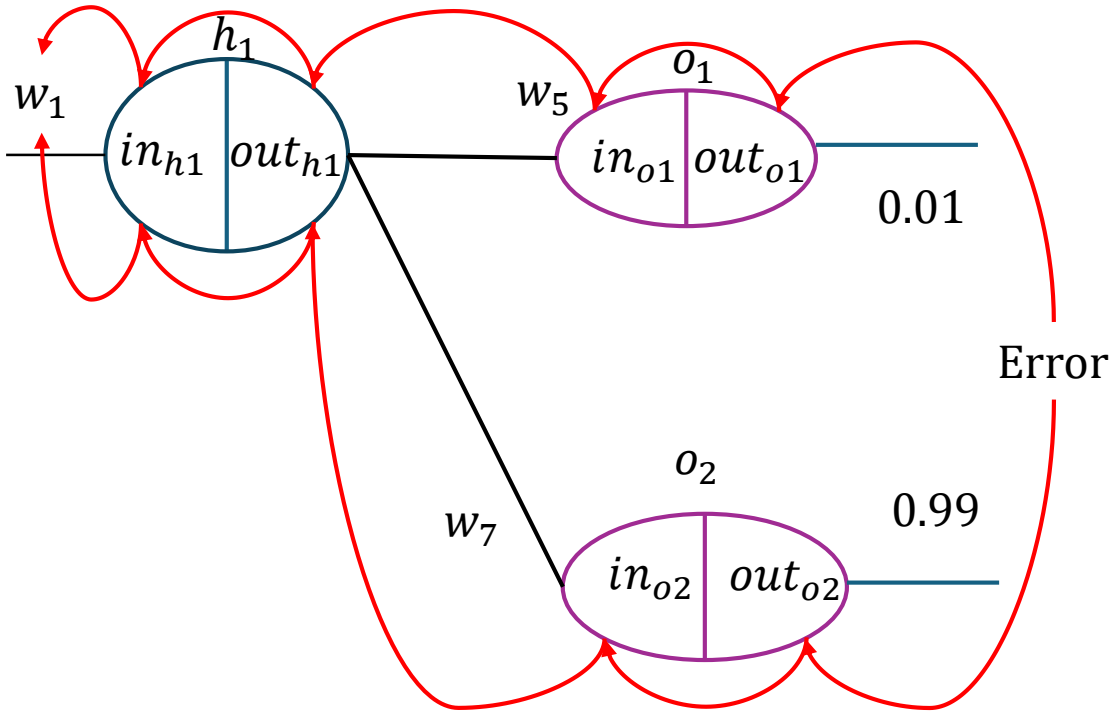
$$\frac{\partial in_{o2}}{\partial w_8} = out_{h2} = 0.593269992$$

$$\frac{\partial Error_{o2}}{\partial w_8} = \text{-0.217071535 x 0.175510053 x 0.596884378}$$

$$\frac{\partial Error}{\partial w_8} = \frac{\partial Error_{o1}}{\partial w_8} + \frac{\partial Error_{o2}}{\partial w_8}$$

$$\frac{\partial Error}{\partial w_8} = \text{-0.022740242}$$

BackPropagation



Already Computed

$$\frac{\partial Error_{o1}}{\partial w_1} = \boxed{\frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}}} \text{ x } \frac{\partial in_{o1}}{\partial out_{h1}} \text{ x } \frac{\partial out_{h1}}{\partial in_{h1}} \text{ x } \frac{\partial in_{h1}}{\partial w_1}$$

$$\frac{\partial Error_{o2}}{\partial w_1} = \boxed{\frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}}} \text{ x } \frac{\partial in_{o2}}{\partial out_{h1}} \text{ x } \frac{\partial out_{h1}}{\partial in_{h1}} \text{ x } \frac{\partial in_{h1}}{\partial w_1}$$

$$\frac{\partial in_{o1}}{\partial out_{h1}} = \frac{\partial (w_5 out_{h1} + w_6 out_{h2} + b_2 1)}{\partial out_{h1}} = w_5 = 0.4$$

$$\frac{\partial out_{h1}}{\partial in_{h1}} = \frac{\partial \left(\frac{1}{1 + e^{-in_{h1}}}\right)}{\partial in_{h1}} = out_{h1}(1 - out_{h1}) = 0.241300709$$

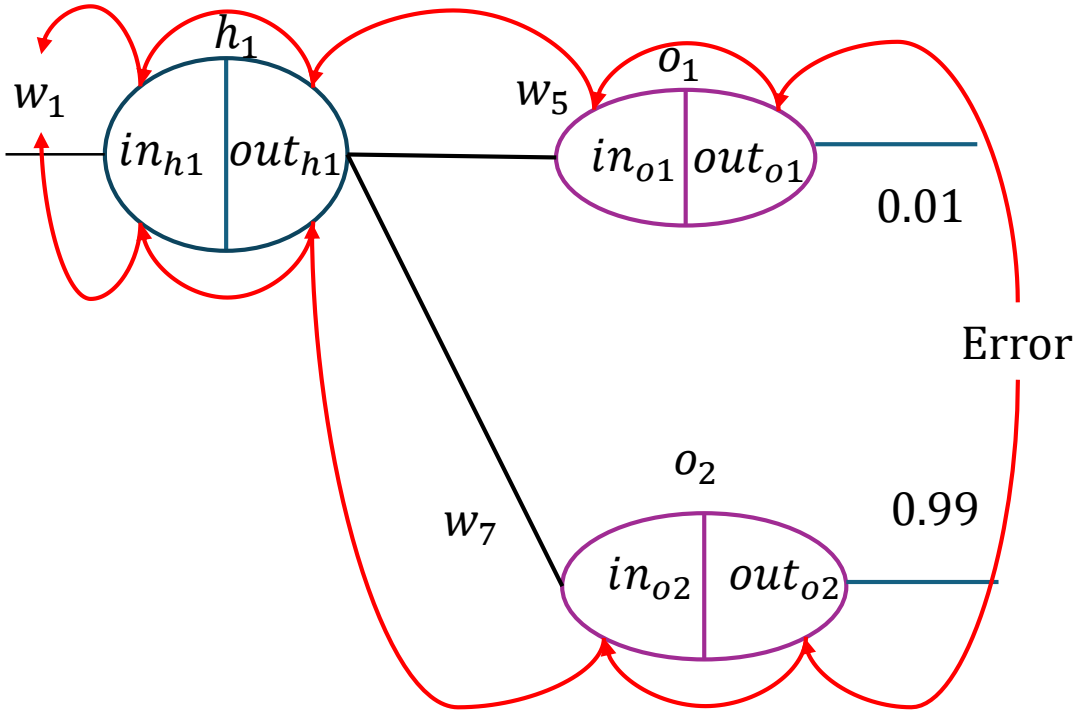
$$\frac{\partial \text{Error}}{\partial w_1} = \frac{\partial Error_{o1}}{\partial w_1} + \frac{\partial Error_{o2}}{\partial w_1}$$

$$\frac{\partial in_{h1}}{\partial w_1} = \frac{\partial (w_1 i_1 + w_2 i_2 + b_1)}{\partial w_1} = i_1 = 0.05$$

$$\frac{\partial Error_{o1}}{\partial w_1} = 0.74136507 \text{ x } 0.186815602 \text{ x } 0.4 \text{ x } 0.241300709 \text{ x } 0.05 = 0.000668$$

BackPropagation

$$\frac{\partial Error}{\partial w_1} = \frac{\partial Error_{o1}}{\partial w_1} + \frac{\partial Error_{o2}}{\partial w_1}$$



Already Computed

$$\frac{\partial Error_{o1}}{\partial w_1} = \boxed{\frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}}} \text{ x } \frac{\partial in_{o1}}{\partial out_{h1}} \text{ x } \boxed{\frac{\partial out_{h1}}{\partial in_{h1}} \text{ x } \frac{\partial in_{h1}}{\partial w_1}}$$

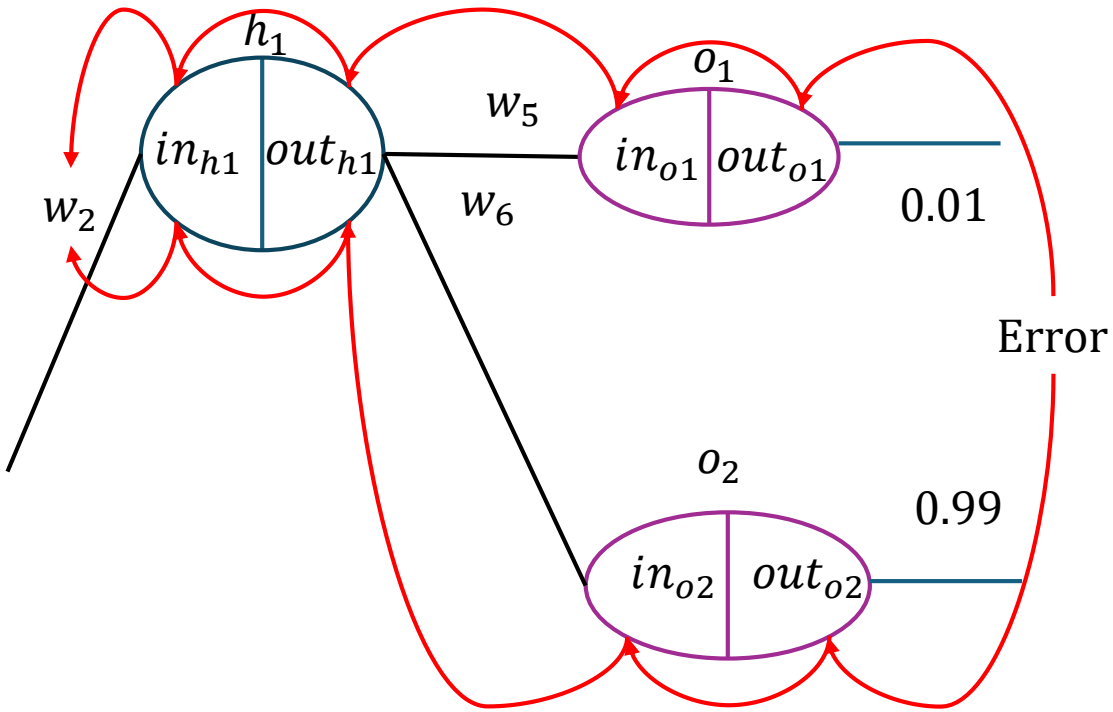
$$\frac{\partial Error_{o2}}{\partial w_1} = \boxed{\frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}}} \text{ x } \frac{\partial in_{o2}}{\partial out_{h1}} \text{ x } \boxed{\frac{\partial out_{h1}}{\partial in_{h1}} \text{ x } \frac{\partial in_{h1}}{\partial w_1}}$$

$$\frac{\partial in_{o2}}{\partial out_{h1}} = \frac{\partial (w_7 out_{h1} + w_8 out_{h2} + b_2)}{\partial out_{h1}} = w_7 = 0.5$$

$$\frac{\partial Error_{o2}}{\partial w_1} = \text{-0.217071535 x 0.175510053 x 0.5 x 0.241300709 x 0.05} = -0.000229828$$

$$\frac{\partial Error}{\partial w_1} = \frac{\partial Error_{o1}}{\partial w_1} + \frac{\partial Error_{o2}}{\partial w_1} = 0.000438568 + (-0.000229828) = 0.000438568$$

BackPropagation



$$\frac{\partial \text{Error}}{\partial w_2} = \frac{\partial Error_{o1}}{\partial w_2} + \frac{\partial Error_{o2}}{\partial w_2}$$

Already Computed

$$\frac{\partial Error_{o1}}{\partial w_2} = \boxed{\frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}} \text{ x } \frac{\partial in_{o1}}{\partial out_{h1}} \text{ x } \frac{\partial out_{h1}}{\partial in_{h1}}} \text{ x } \frac{\partial in_{h1}}{\partial w_2}$$

$$\frac{\partial Error_{o2}}{\partial w_2} = \boxed{\frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}} \text{ x } \frac{\partial in_{o2}}{\partial out_{h1}} \text{ x } \frac{\partial out_{h1}}{\partial in_{h1}}} \text{ x } \frac{\partial in_{h1}}{\partial w_2}$$

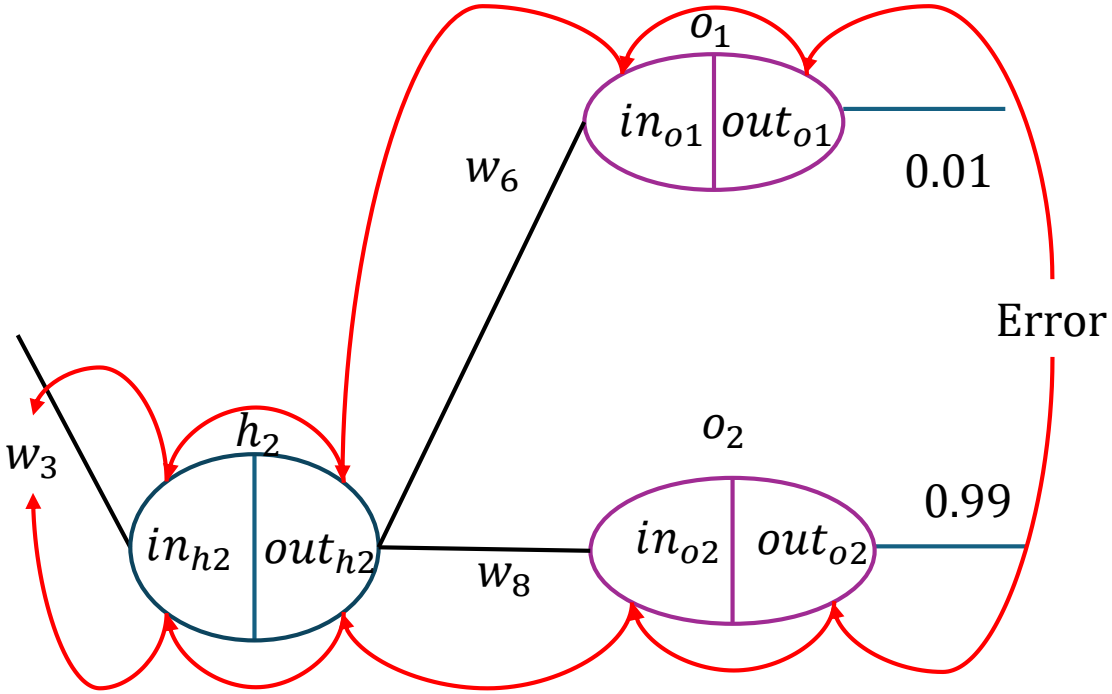
$$\frac{\partial in_{h1}}{\partial w_2} = \frac{\partial (w_1 i_1 + w_2 i_2 + b_1)}{\partial w_2} = i_2 = 0.1$$

$$\frac{\partial Error_{o1}}{\partial w_2} = 0.74136507 \text{ x } 0.186815602 \text{ x } 0.4 \text{ x } 0.241300709 \text{ x } 0.1 = 0.001336792$$

$$\frac{\partial Error_{o2}}{\partial w_2} = \text{-}0.217071535 \text{ x } 0.175510053 \text{ x } 0.5 \text{ x } 0.241300709 \text{ x } 0.1 = -0.000367725$$

$$\frac{\partial \text{Error}}{\partial w_2} = \frac{\partial Error_{o1}}{\partial w_2} + \frac{\partial Error_{o2}}{\partial w_2} = 0.001336792 + (-0.000367725) = 0.000969067$$

BackPropagation



$$\frac{\partial Error}{\partial w_3} = \frac{\partial Error_{o1}}{\partial w_3} + \frac{\partial Error_{o2}}{\partial w_3}$$

Already Computed

$$\frac{\partial Error_{o1}}{\partial w_3} = \boxed{\frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}}} \text{ x } \frac{\partial in_{o1}}{\partial out_{h2}} \text{ x } \frac{\partial out_{h2}}{\partial in_{h2}} \text{ x } \frac{\partial in_{h2}}{\partial w_3}$$

$$\frac{\partial Error_{o2}}{\partial w_3} = \boxed{\frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}}} \text{ x } \frac{\partial in_{o2}}{\partial out_{h2}} \text{ x } \frac{\partial out_{h2}}{\partial in_{h2}} \text{ x } \frac{\partial in_{h2}}{\partial w_3}$$

$$\frac{\partial in_{o1}}{\partial out_{h2}} = \frac{\partial (w_5 out_{h1} + w_6 out_{h2} + b_2 1)}{\partial out_{h2}} = w_6 = 0.45$$

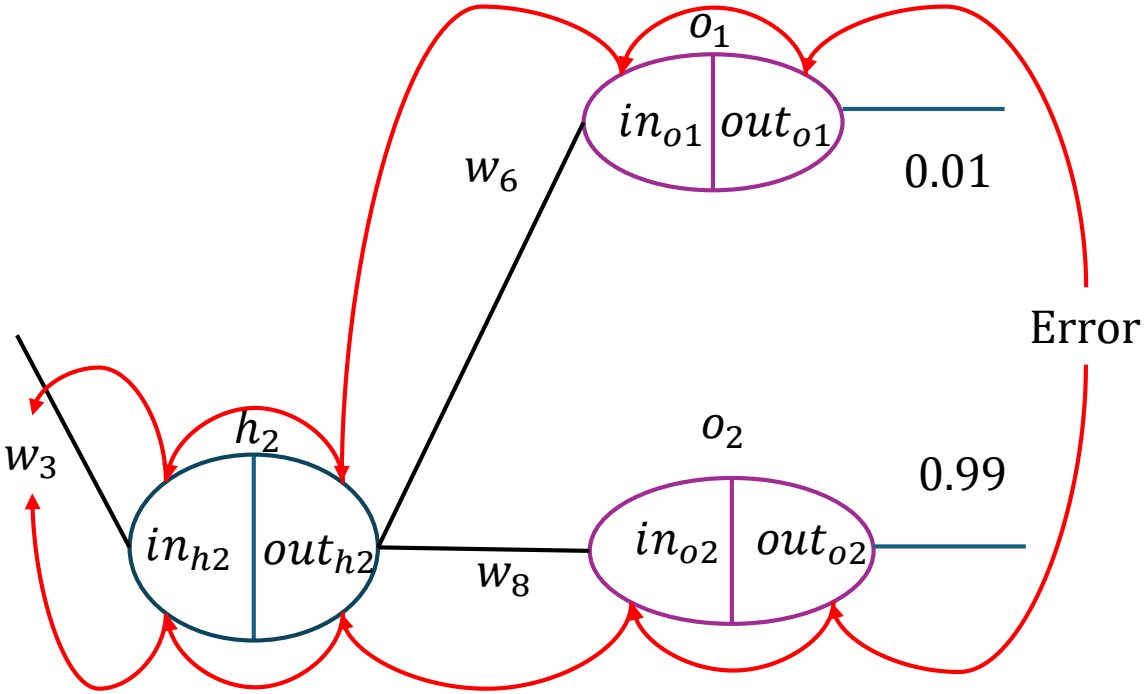
$$\frac{\partial out_{h2}}{\partial in_{h2}} = \frac{\partial \left(\frac{1}{1 + e^{-in_{h2}}}\right)}{\partial in_{h2}} = out_{h2}(1 - out_{h2}) = 0.240613417$$

$$\frac{\partial in_{h2}}{\partial w_3} = \frac{\partial (w_3 i_1 + w_4 i_2 + b_1)}{\partial w_1} = i_1 = 0.05$$

$$\frac{\partial Error_{o1}}{\partial w_3} = 0.74136507 \text{ x } 0.186815602 \text{ x } 0.45 \text{ x } 0.240613417 \text{ x } 0.05 = 0.000749804$$

BackPropagation

$$\frac{\partial Error}{\partial w_3} = \frac{\partial Error_{o1}}{\partial w_3} + \frac{\partial Error_{o2}}{\partial w_3}$$



Already Computed

$$\frac{\partial Error_{o1}}{\partial w_3} = \frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}} \text{ x } \frac{\partial in_{o1}}{\partial out_{h2}} \text{ x } \frac{\partial out_{h2}}{\partial in_{h2}} \text{ x } \frac{\partial in_{h2}}{\partial w_3}$$

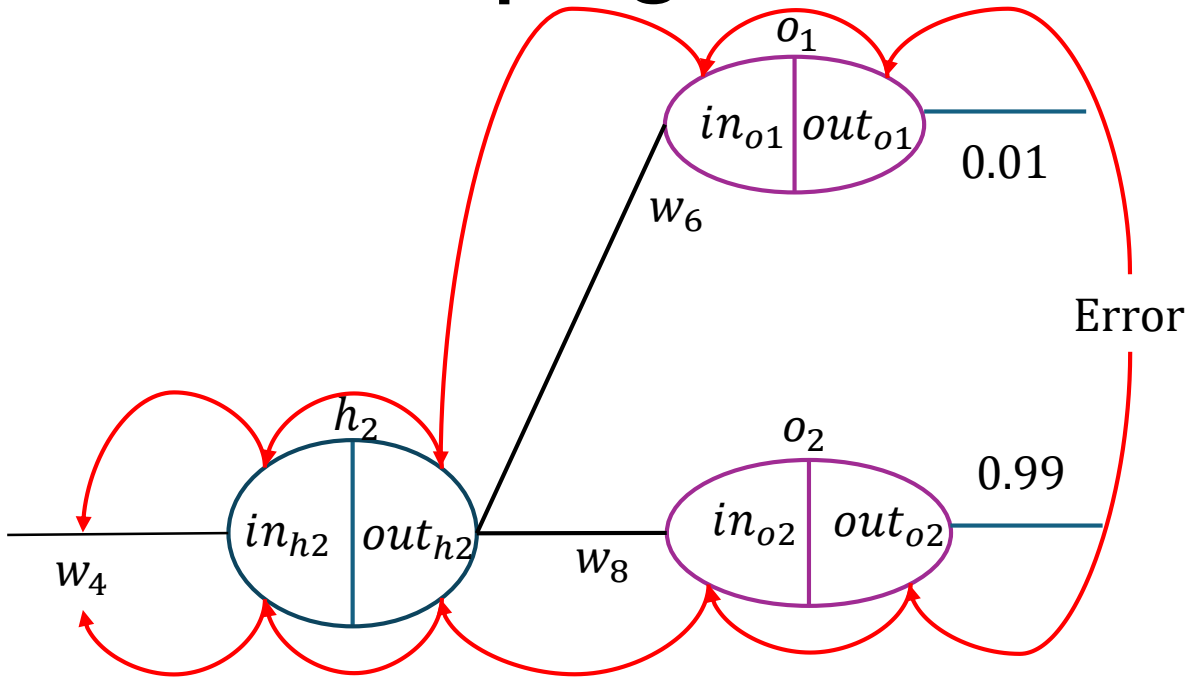
$$\frac{\partial Error_{o2}}{\partial w_3} = \frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}} \text{ x } \frac{\partial in_{o2}}{\partial out_{h2}} \text{ x } \frac{\partial out_{h2}}{\partial in_{h2}} \text{ x } \frac{\partial in_{h2}}{\partial w_3}$$

$$\frac{\partial in_{o2}}{\partial out_{h2}} = \frac{\partial (w_7 out_{h1} + w_8 out_{h2} + b_2)}{\partial out_{h2}} = w_8 = 0.55$$

$$\frac{\partial Error_{o2}}{\partial w_3} = \text{-0.217071535 x 0.175510053 x 0.55 x 0.240613417 x 0.05} = -0.000252091$$

$$\frac{\partial Error}{\partial w_3} = \frac{\partial Error_{o1}}{\partial w_3} + \frac{\partial Error_{o2}}{\partial w_3} = 0.000749804 + (-0.000252091) = 0.000497713$$

BackPropagation



$$\frac{\partial Error}{\partial w_4} = \frac{\partial Error_{o1}}{\partial w_4} + \frac{\partial Error_{o2}}{\partial w_4}$$

Already Computed

$$\frac{\partial Error_{o1}}{\partial w_4} = \boxed{\frac{\partial Error_{o1}}{\partial out_{o1}} \text{ x } \frac{\partial out_{o1}}{\partial in_{o1}} \text{ x } \frac{\partial in_{o1}}{\partial out_{h2}} \text{ x } \frac{\partial out_{h2}}{\partial in_{h2}}} \text{ x } \frac{\partial in_{h2}}{\partial w_4}$$

$$\frac{\partial Error_{o2}}{\partial w_4} = \boxed{\frac{\partial Error_{o2}}{\partial out_{o2}} \text{ x } \frac{\partial out_{o2}}{\partial in_{o2}} \text{ x } \frac{\partial in_{o2}}{\partial out_{h2}} \text{ x } \frac{\partial out_{h2}}{\partial in_{h2}}} \text{ x } \frac{\partial in_{h2}}{\partial w_4}$$

$$\frac{\partial in_{h2}}{\partial w_4} = \frac{\partial (w_3 i_1 + w_4 i_2 + b_1)}{\partial w_2} = i_2 = 0.1$$

$$\frac{\partial Error_{o2}}{\partial w_4} \text{ -0.217071535 x 0.175510053 x 0.55 x 0.240613417 x 0.1} = -0.000504182$$

$$\frac{\partial Error_{o1}}{\partial w_4} = 0.74136507 \text{ x 0.186815602 x 0.45 x 0.240613417 x 0.1} = 0.001499608$$

$$\frac{\partial Error}{\partial w_4} = \frac{\partial Error_{o1}}{\partial w_4} + \frac{\partial Error_{o2}}{\partial w_4} = 0.001499608 + (-0.000504182) = 0.000995425$$

$$\frac{\partial \mathrm{E}rror}{\partial w_5} = 0.082167041$$

$$\frac{\partial \mathrm{E}rror}{\partial w_1} = 0.000438568$$

$$\frac{\partial \mathrm{E}rror}{\partial 6} = 0.082667628$$

$$\frac{\partial \mathrm{Error}}{\partial w_2} = \ 0.000969067$$

$$\frac{\partial \mathrm{E}rror}{\partial w_7} = \text{-}0.02260254$$

$$\frac{\partial \mathrm{E}rror}{\partial w_3} = 0.000497713$$

$$\frac{\partial \mathrm{E}rror}{\partial w_8} = \text{-}0.022740242$$

$$\frac{\partial \mathrm{E}rror}{\partial w_4} = 0.000995425$$

BackPropagation - Weight Update

Learning Rate

$$w_i = w_i - \alpha \, x \frac{\partial Error}{\partial w_i}$$

$w_1 = 0.15 - 0.5 * 0.0.000438568$

$w_2 = 0.20 - 0.5 * 0.00096906$

$w_3 = 0.25 - 0.5 * 0.000497713$

$w_4 = 0.30 - 0.5 * 0.000995425$

$w_5 = 0.40 - 0.5 * 0.082167041$

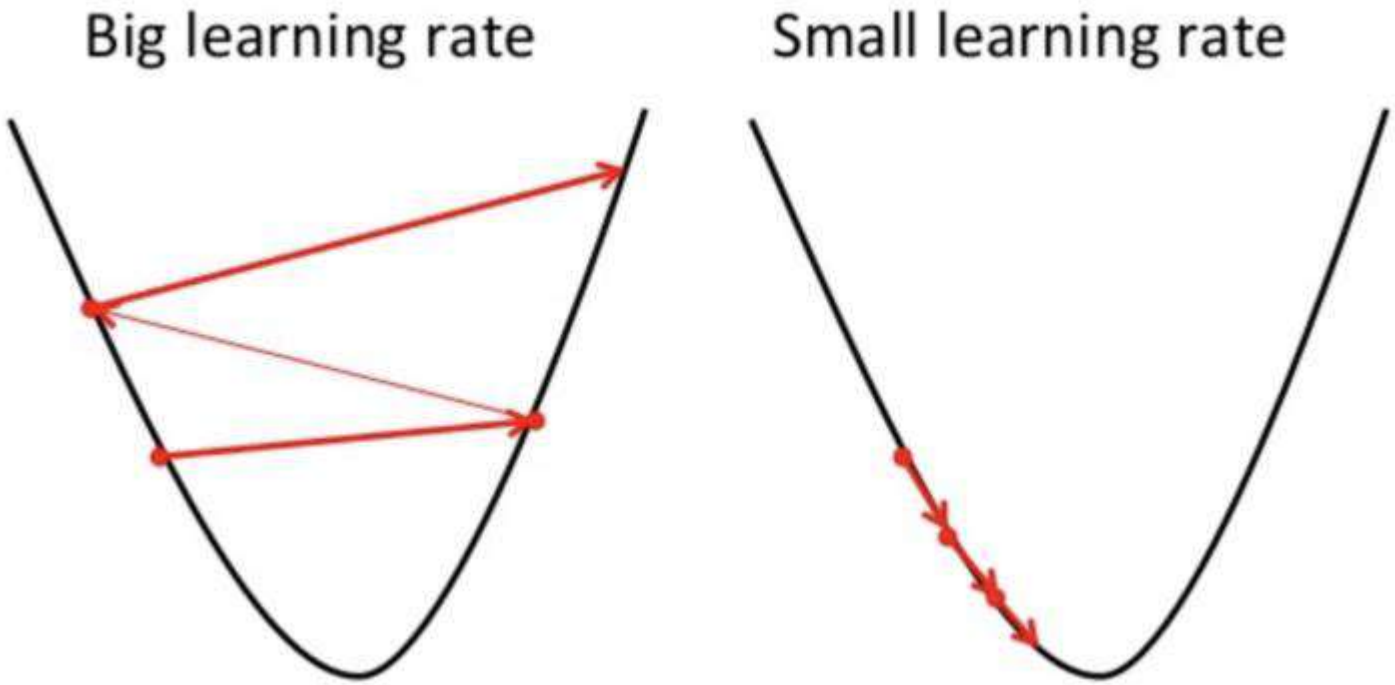
$w_6 = 0.45 - 0.5 * 0.082667628$

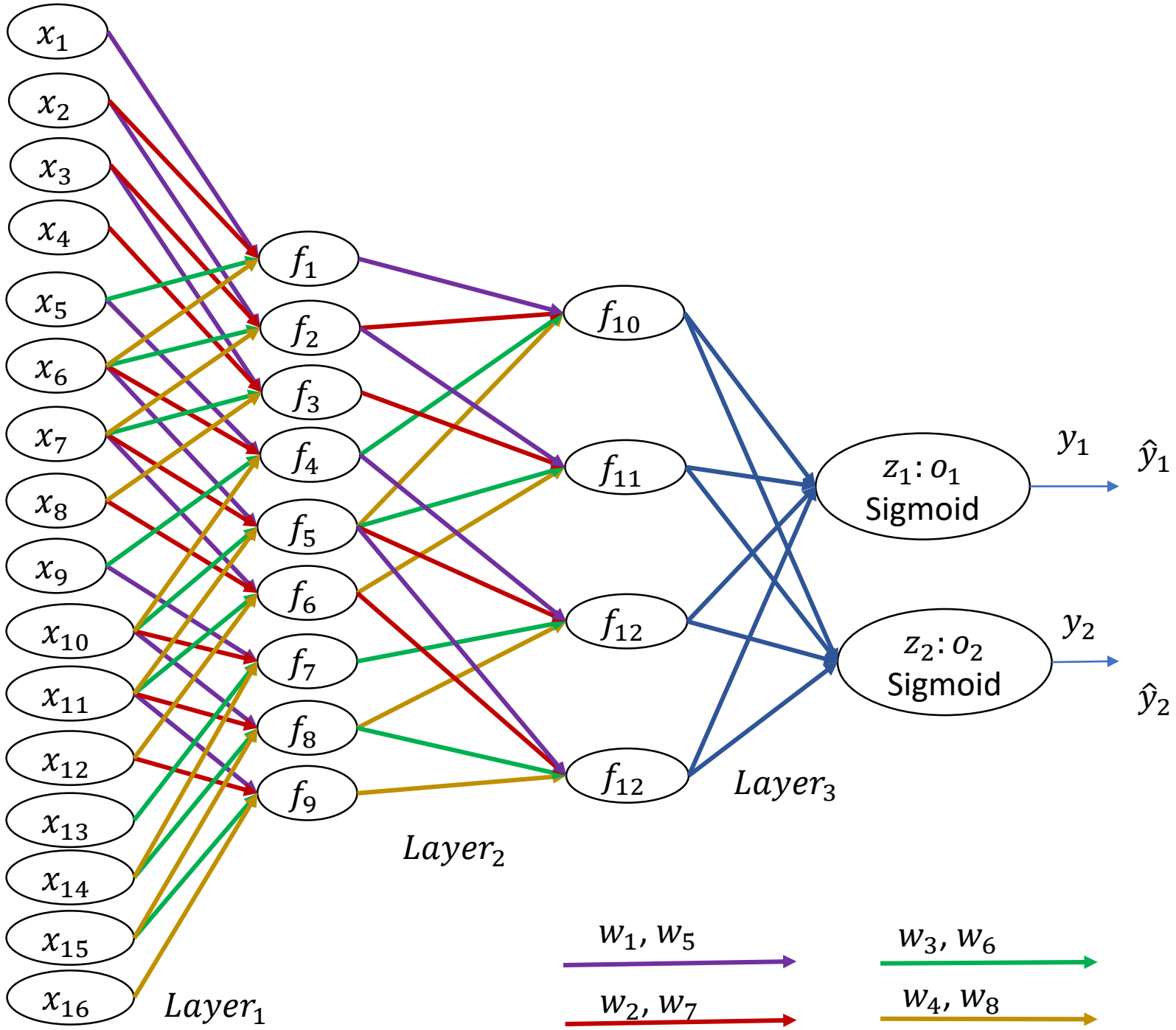
$w_7 = 0.50 - 0.5 * -0.02260254$

$w_8 = 0.55 - 0.5 * -0.022740242$

w1	0.149780716
w2	0.199515467
w3	0.249751144
w4	0.299502287
w5	0.35891648
w6	0.408666186
w7	0.51130127
w8	0.561370121

Learning Rate





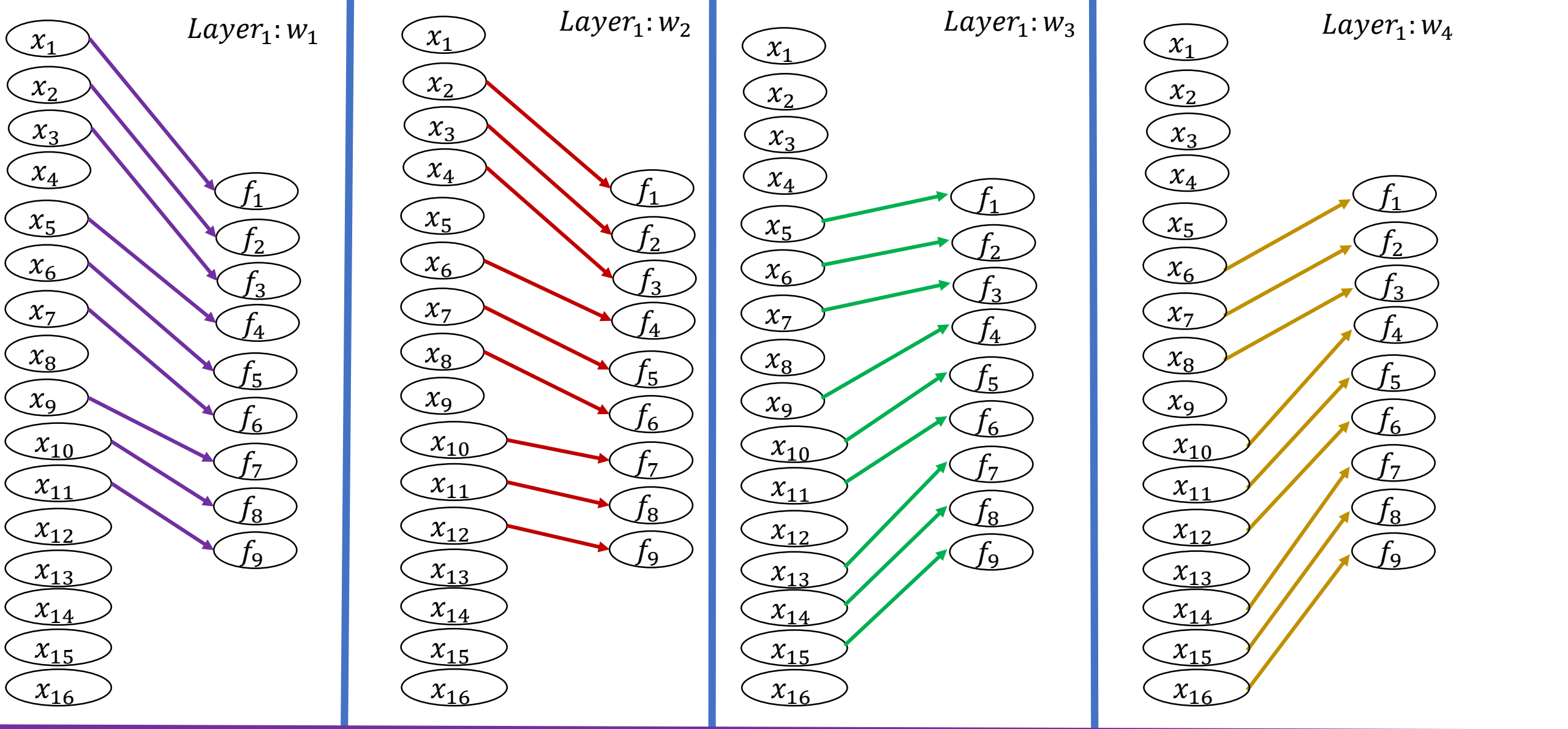
Shared Weights

$Layer_2: w_1, w_2, w_3, w_4$

$Layer_2: w_5, w_6, w_7, w_8$

Weights

$Layer_3: w_9, w_{10}, w_{11}, w_{12}, w_{13}, w_{14}, w_{15}, w_{16}$



$$f_1 = w_1 * x_1 + w_2 * x_2 + w_3 * x_5 + w_4 * x_6$$

$$f_2 = w_1 * x_2 + w_2 * x_3 + w_3 * x_6 + w_4 * x_7$$

$$f_3 = w_1 * x_3 + w_2 * x_4 + w_3 * x_7 + w_4 * x_8$$

$$f_4 = w_1 * x_5 + w_2 * x_6 + w_3 * x_9 + w_4 * x_{10}$$

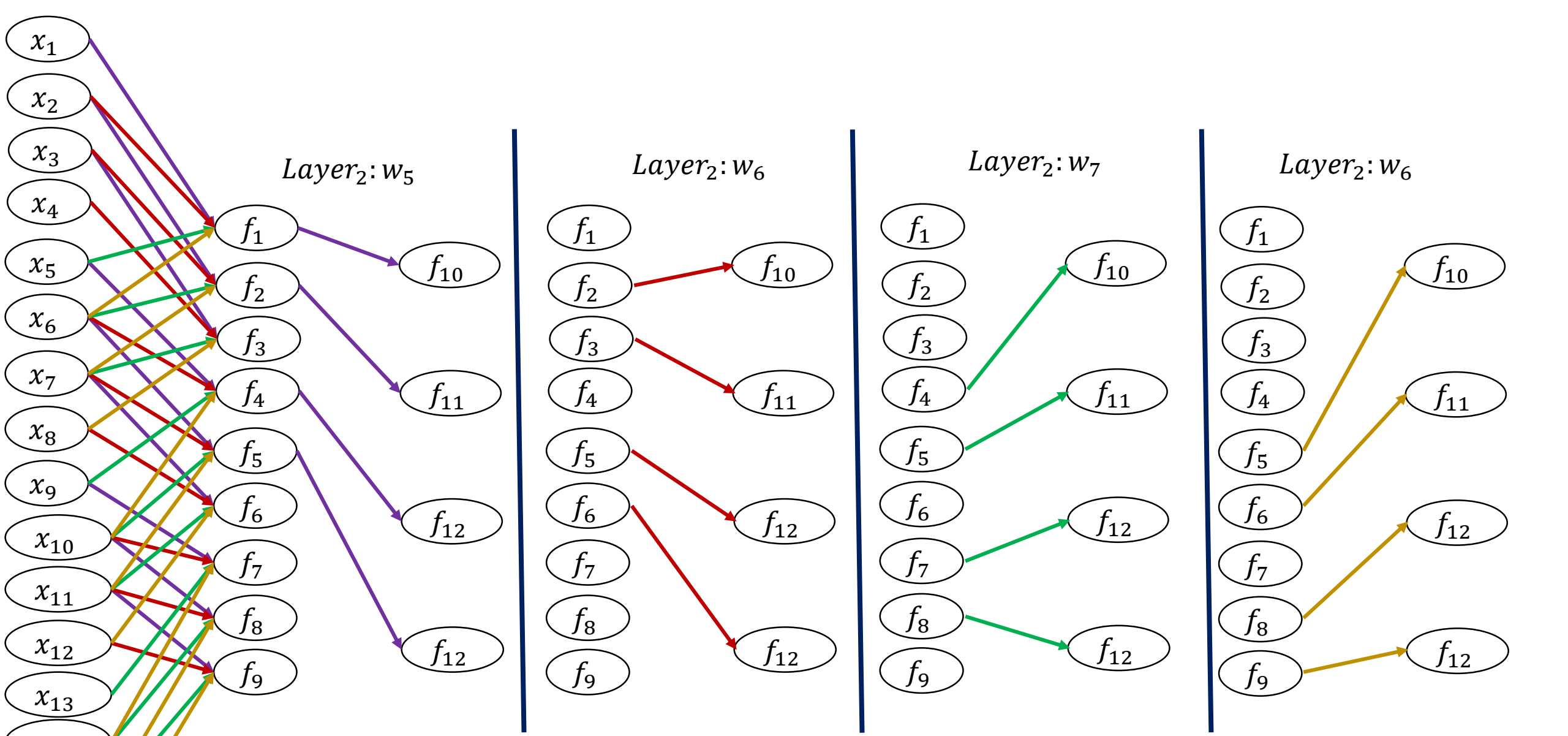
$$f_5 = w_1 * x_6 + w_2 * x_7 + w_3 * x_{10} + w_4 * x_{11}$$

$$f_6 = w_1 * x_7 + w_2 * x_8 + w_3 * x_{11} + w_4 * x_{12}$$

$$f_7 = w_1 * x_9 + w_2 * x_{10} + w_3 * x_{13} + w_4 * x_{14}$$

$$f_8 = w_1 * x_{10} + w_2 * x_{11} + w_3 * x_{14} + w_4 * x_{15}$$

$$f_9 = w_1 * x_{11} + w_2 * x_{12} + w_3 * x_{15} + w_4 * x_{16}$$



$$f_{10} = w_5 * f_1 + w_6 * f_2 + w_7 * f_4 + w_8 * f_5$$

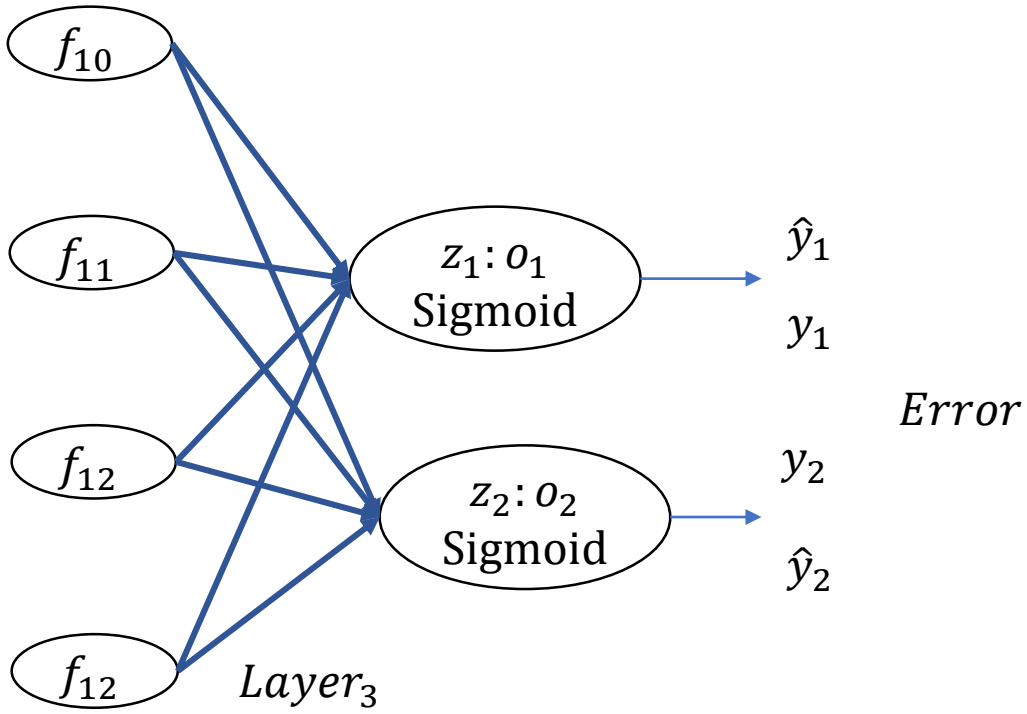
$$f_{11} = w_5 * f_2 + w_6 * f_3 + w_7 * f_5 + w_8 * f_6$$

$$f_{12} = w_5 * f_4 + w_6 * f_5 + w_7 * f_7 + w_8 * f_8$$

$$f_{13} = w_5 * f_5 + w_6 * f_6 + w_7 * f_8 + w_8 * f_9$$

Weights

$Layer_3: w_9, w_{10}, w_{11}, w_{12}, w_{13}, w_{14}, w_{15}, w_{16}$



$$Error_1 = -y_1 \log(\hat{y}_1) - (1 - y_1) \log(1 - \hat{y}_1)$$

$$Error_2 = -y_2 \log(\hat{y}_2) - (1 - y_2) \log(1 - \hat{y}_2)$$

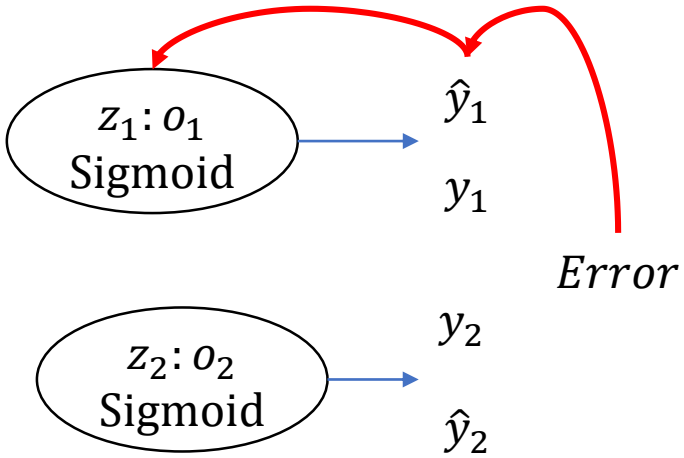
$$Error = Error_1 + Error_2$$

$$z_1 = w_9 * f_{10} + w_{10} * f_{11} + w_{11} * f_{12} + w_{12} * f_{13}$$

$$z_2 = w_{13} * f_{10} + w_{14} * f_{11} + w_{15} * f_{12} + w_{16} * f_{13}$$

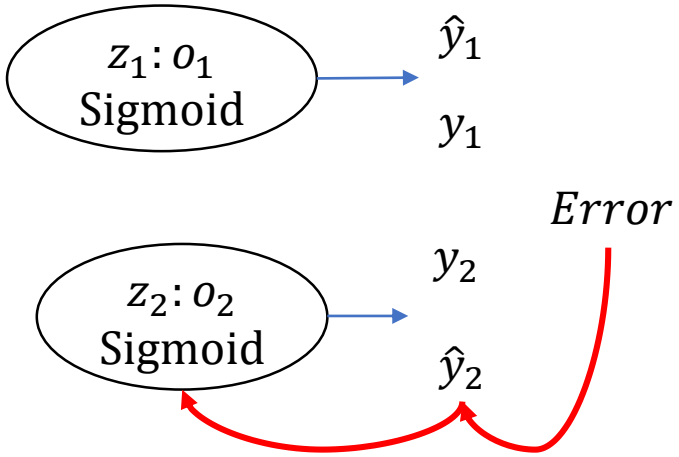
$$o_1 = \frac{1}{(1 + e^{-z_1})} = \hat{y}_1 =$$

$$o_2 = \frac{1}{(1 + e^{-z_2})} = \hat{y}_2 =$$



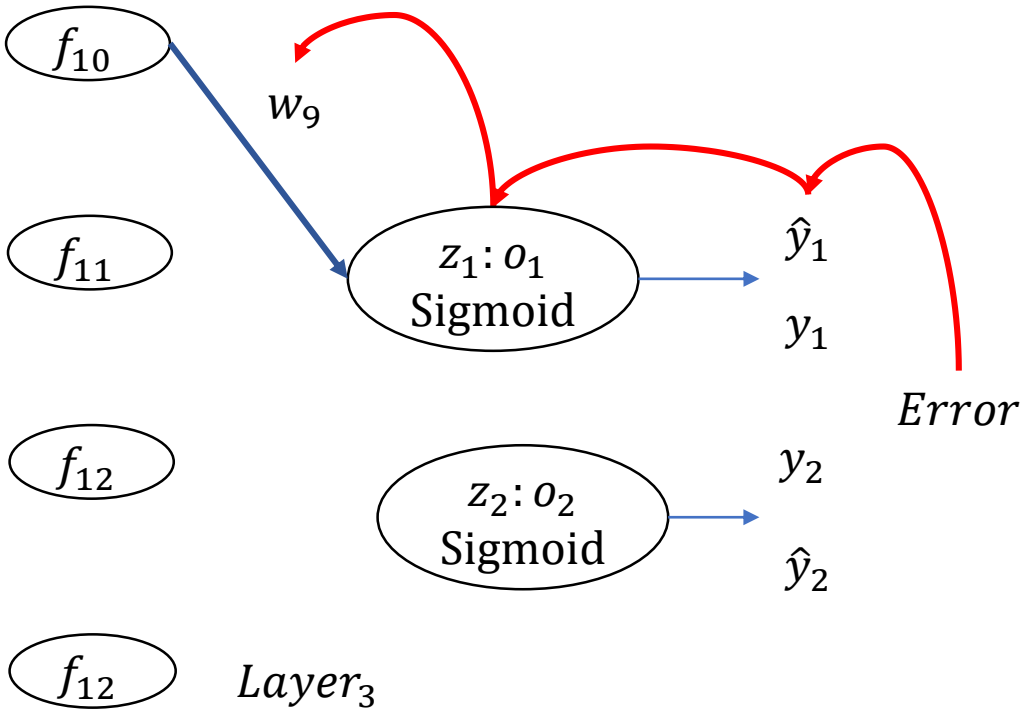
$$\frac{\partial \mathrm{E}rror}{\partial o_1} = \frac{\partial Error_{o1}}{\partial o_1} + \frac{\partial Error_{o2}}{\partial o_1}$$

$$\frac{\partial Error_{o1}}{\partial o_1} = -\left(\frac{y_1}{\hat{y}_1}\right) - \frac{(1-y_1)}{(1-\hat{y}_1)}$$



$$\frac{\partial \mathrm{E}rror}{\partial o_2} = \frac{\partial Error_{o1}}{\partial o_2} + \frac{\partial Error_{o2}}{\partial o_2}$$

$$\frac{\partial Error_{o1}}{\partial o_2} = -\left(\frac{y_2}{\hat{y}_2}\right) - \frac{(1-y_2)}{(1-\hat{y}_2)}$$



$$\frac{\partial o_1}{\partial z_1} = \frac{\partial(\frac{1}{1+e^{-z_1}})}{\partial z_1} = o_1(1-o_1)$$

$$\frac{\partial z_1}{\partial w_9} = \frac{\partial(w_9 * f_{10} + w_{10} * f_{11} + w_{11} * f_{12} + w_{12} * f_{13})}{\partial w_9}$$

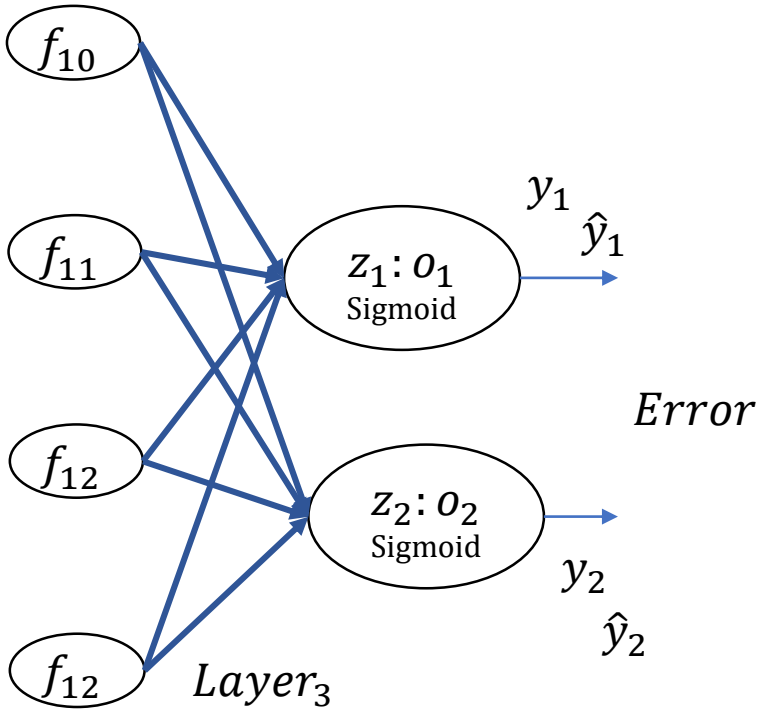
$$\frac{\partial Error_{o1}}{\partial w_9} = \frac{\partial Error_{o1}}{\partial o_1} \text{ x } \frac{\partial o_1}{\partial z_1} \text{ x } \frac{\partial z_1}{\partial w_9}$$

$$\frac{\partial \mathrm{E}rror}{\partial w_9} = \frac{\partial Error_{o1}}{\partial w_9} + \frac{\partial Error_{o2}}{\partial w_9}$$

$$\frac{\partial Error_{o1}}{\partial w_9} = \frac{\partial Error_{o1}}{\partial o_1} \text{ x } \frac{\partial o_1}{\partial z_1} \text{ x } \frac{\partial z_1}{\partial w_9}$$

Weights

$Layer_3: w_9, w_{10}, w_{11}, w_{12}, w_{13}, w_{14}, w_{15}, w_{16}$



$$\frac{\partial Error_{o1}}{\partial w_9} = \frac{\partial Error_{o1}}{\partial o_1} \text{ x } \frac{\partial o_1}{\partial z_1} \text{ x } \frac{\partial z_1}{\partial w_9}$$

$$\frac{\partial Error_{o1}}{\partial w_{10}} = \frac{\partial Error_{o1}}{\partial o_1} \text{ x } \frac{\partial o_1}{\partial z_1} \text{ x } \frac{\partial z_1}{\partial w_{10}}$$

$$\frac{\partial Error_{o1}}{\partial w_{11}} = \frac{\partial Error_{o1}}{\partial o_1} \text{ x } \frac{\partial o_1}{\partial z_1} \text{ x } \frac{\partial z_1}{\partial w_{11}}$$

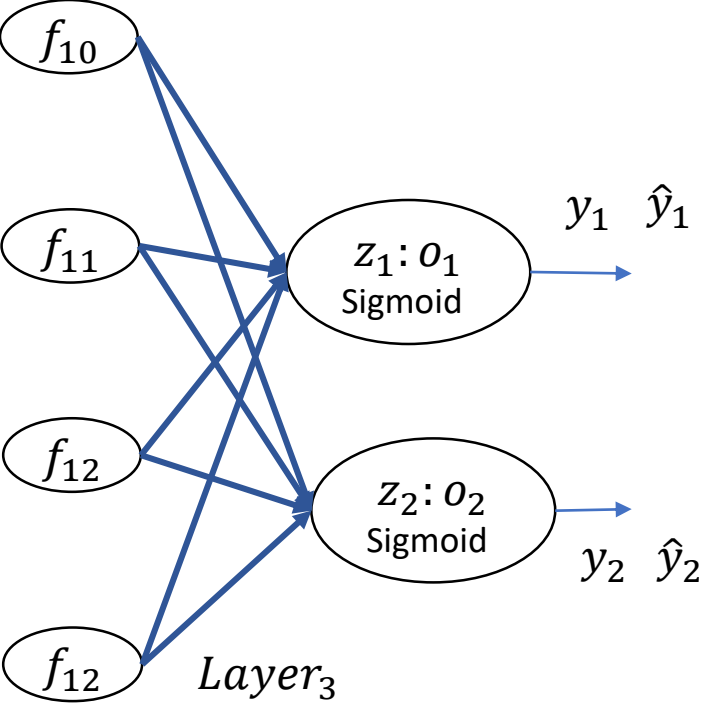
$$\frac{\partial Error_{o1}}{\partial w_{12}} = \frac{\partial Error_{o1}}{\partial o_1} \text{ x } \frac{\partial o_1}{\partial z_1} \text{ x } \frac{\partial z_1}{\partial w_{12}}$$

$$\frac{\partial Error_{o2}}{\partial w_{13}} = \frac{\partial Error_{o2}}{\partial o_2} \text{ x } \frac{\partial o_2}{\partial z_2} \text{ x } \frac{\partial z_2}{\partial w_{13}}$$

$$\frac{\partial Error_{o2}}{\partial w_{14}} = \frac{\partial Error_{o2}}{\partial o_2} \text{ x } \frac{\partial o_2}{\partial z_2} \text{ x } \frac{\partial z_2}{\partial w_{14}}$$

$$\frac{\partial Error_{o2}}{\partial w_{15}} = \frac{\partial Error_{o2}}{\partial o_2} \text{ x } \frac{\partial o_2}{\partial z_2} \text{ x } \frac{\partial z_2}{\partial w_{15}}$$

$$\frac{\partial Error_{o2}}{\partial w_{16}} = \frac{\partial Error_{o2}}{\partial o_2} \text{ x } \frac{\partial o_2}{\partial z_2} \text{ x } \frac{\partial z_2}{\partial w_{16}}$$



$$\frac{\partial Error}{\partial f_{10}} = \frac{\partial Error_{o1}}{\partial f_{10}} + \frac{\partial Error_{o2}}{\partial f_{10}}$$

$$\frac{\partial Error_{o1}}{\partial f_{10}} = \frac{\partial Error_{o1}}{\partial o_1} X \frac{\partial o_1}{\partial z_1} X \frac{\partial z_1}{\partial f_{10}}$$

$$\frac{\partial Error_{o2}}{\partial f_{10}} = \frac{\partial Error_{o2}}{\partial o_2} X \frac{\partial o_2}{\partial z_2} X \frac{\partial z_2}{\partial f_{10}}$$

$$\frac{\partial z_1}{\partial f_{10}} = \frac{\partial (w_9 * f_{10} + w_{10} * f_{11} + w_{11} * f_{12} + w_{12} * f_{13})}{\partial f_{10}} = w_9$$

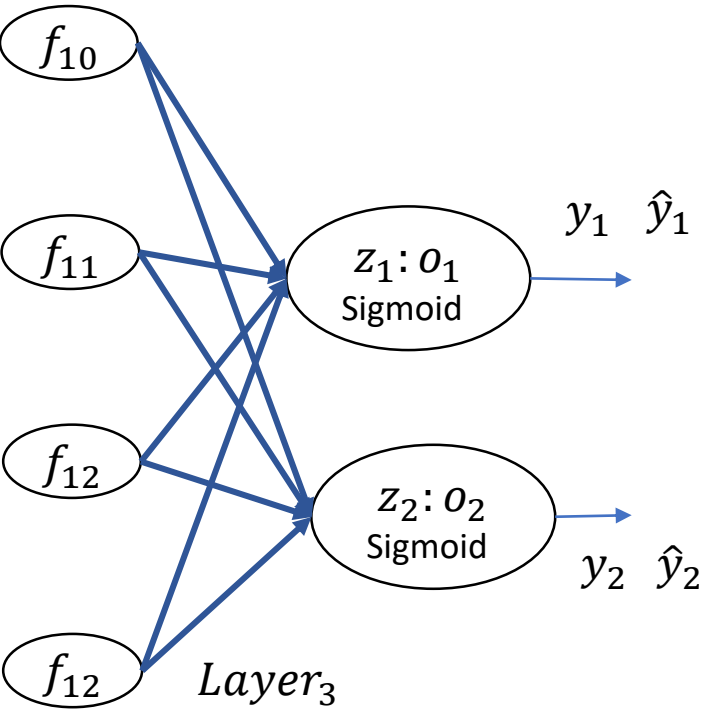
$$\frac{\partial z_2}{\partial f_{10}} = \frac{\partial (w_{13} * f_{10} + w_{14} * f_{11} + w_{15} * f_{12} + w_{16} * f_{13})}{\partial f_{10}} = w_{13}$$

$$\frac{\partial Error_{o1}}{\partial f_{10}} = \frac{\partial Error_{o1}}{\partial o_1} X \frac{\partial o_1}{\partial z_1} X \frac{\partial z_1}{\partial f_{10}}$$

$$\frac{\partial Error_{o2}}{\partial f_{10}} = \frac{\partial Error_{o2}}{\partial o_2} X \frac{\partial o_2}{\partial z_2} X \frac{\partial z_2}{\partial f_{10}}$$

$$\frac{\partial Error}{\partial f_{10}} = \frac{\partial Error_{o1}}{\partial f_{10}} + \frac{\partial Error_{o2}}{\partial f_{10}}$$

$$\frac{\partial Error}{\partial f_{10}} = \frac{\partial Error_{o1}}{\partial f_{10}} + \frac{\partial Error_{o2}}{\partial f_{10}}$$

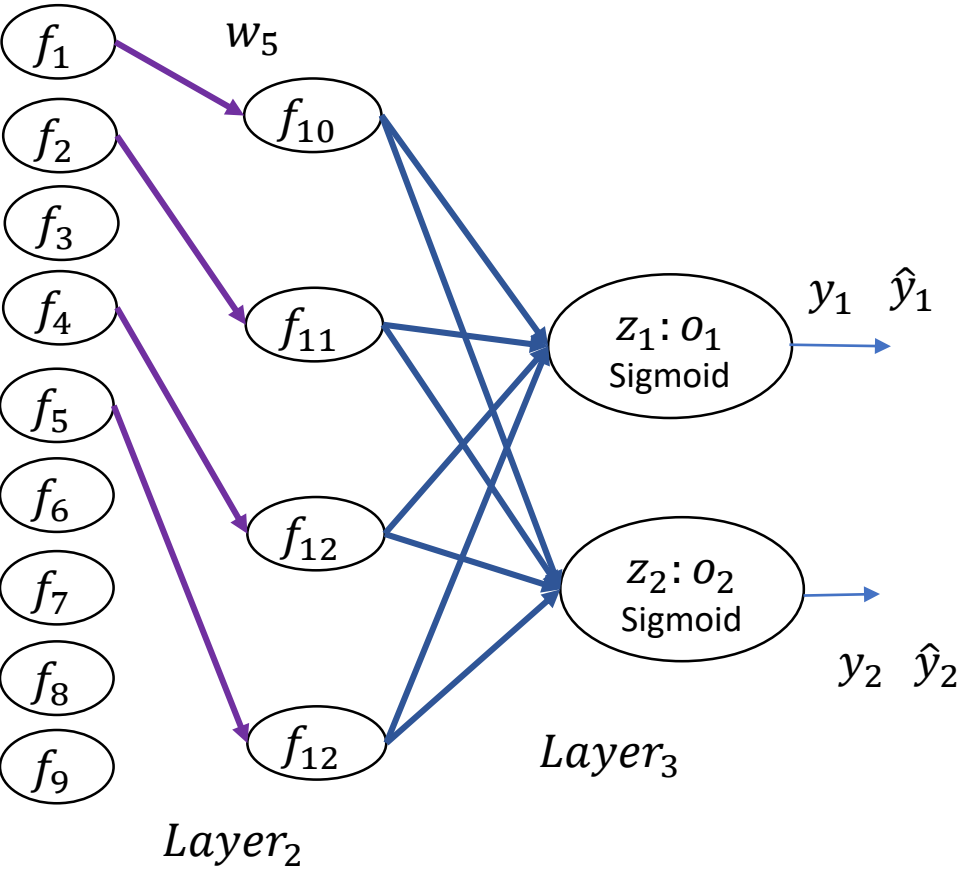


$$\frac{\partial \mathrm{Error}}{\partial f_{10}} = \frac{\partial Error_{o1}}{\partial f_{10}} + \frac{\partial Error_{o2}}{\partial f_{10}} = \frac{\partial Error_{o1}}{\partial o_1} X \frac{\partial o_1}{\partial z_1} X \frac{\partial z_1}{\partial f_{10}} + \frac{\partial Error_{o2}}{\partial o_2} X \frac{\partial o_2}{\partial z_2} X \frac{\partial z_2}{\partial f_{10}} = 0$$

$$\frac{\partial \mathrm{Error}}{\partial f_{11}} = \frac{\partial Error_{o1}}{\partial f_{11}} + \frac{\partial Error_{o2}}{\partial f_{11}} = \frac{\partial Error_{o1}}{\partial o_1} X \frac{\partial o_1}{\partial z_1} X \frac{\partial z_1}{\partial f_{11}} + \frac{\partial Error_{o2}}{\partial o_2} X \frac{\partial o_2}{\partial z_2} X \frac{\partial z_2}{\partial f_{11}} = 0$$

$$\frac{\partial \mathrm{Error}}{\partial f_{12}} = \frac{\partial Error_{o1}}{\partial f_{12}} + \frac{\partial Error_{o2}}{\partial f_{12}} = \frac{\partial Error_{o1}}{\partial o_1} X \frac{\partial o_1}{\partial z_1} X \frac{\partial z_1}{\partial f_{12}} + \frac{\partial Error_{o2}}{\partial o_2} X \frac{\partial o_2}{\partial z_2} X \frac{\partial z_2}{\partial f_{12}} = 0$$

$$\frac{\partial \mathrm{Error}}{\partial f_{13}} = \frac{\partial Error_{o1}}{\partial f_{13}} + \frac{\partial Error_{o2}}{\partial f_{13}} = \frac{\partial Error_{o1}}{\partial o_1} X \frac{\partial o_1}{\partial z_1} X \frac{\partial z_1}{\partial f_{13}} + \frac{\partial Error_{o2}}{\partial o_2} X \frac{\partial o_2}{\partial z_2} X \frac{\partial z_2}{\partial f_{13}} = 0$$



$$\frac{\partial \mathrm{Error}}{\partial w_5} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_5} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_5} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_5} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_5}$$

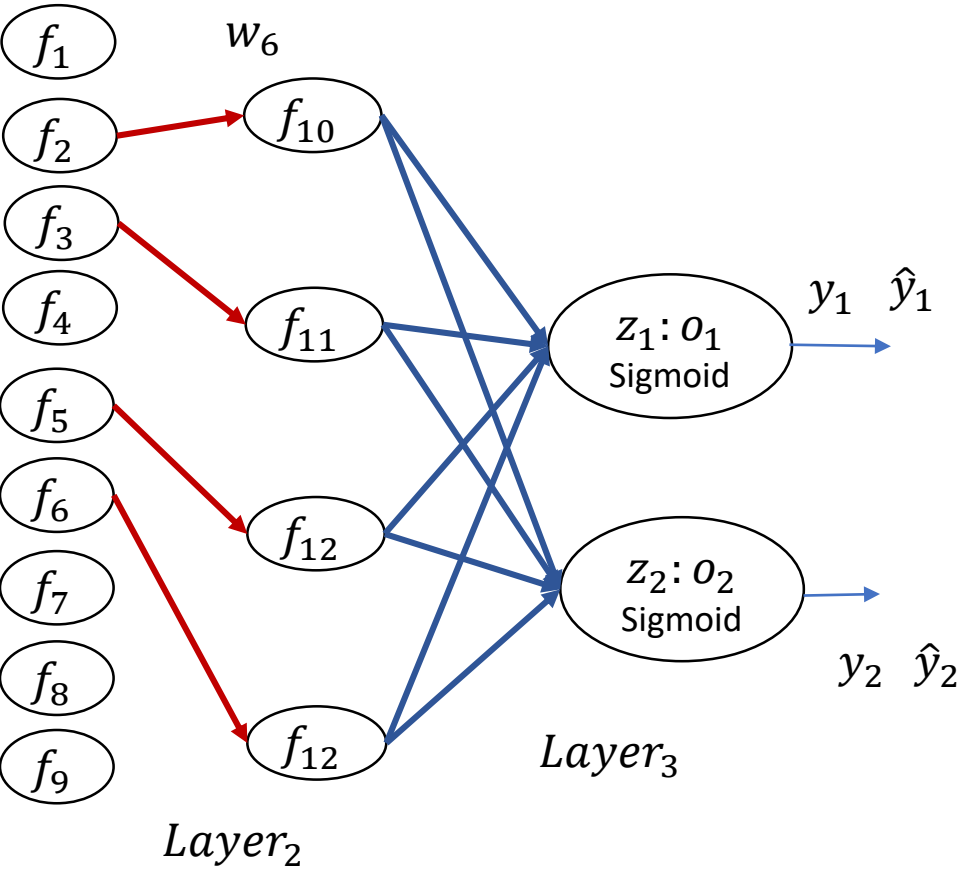
$$\frac{\partial f_{10}}{\partial w_5} = \frac{\partial (w_5 * f_1 + w_6 * f_2 + w_7 * f_4 + w_8 * f_5)}{\partial w_5} = f_1 = 0$$

$$\frac{\partial f_{11}}{\partial w_5} = \frac{\partial (w_5 * f_2 + w_6 * f_3 + w_7 * f_5 + w_8 * f_6)}{\partial w_5} = f_2 = 0$$

$$\frac{\partial f_{12}}{\partial w_5} = \frac{\partial (w_5 * f_4 + w_6 * f_5 + w_7 * f_7 + w_8 * f_8)}{\partial w_5} = f_4 = 0$$

$$\frac{\partial f_{13}}{\partial w_5} = \frac{\partial (w_5 * f_5 + w_6 * f_6 + w_7 * f_8 + w_8 * f_9)}{\partial w_5} = f_5 = 0$$

$$\frac{\partial \mathrm{Error}}{\partial w_5} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_5} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_5} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_5} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_5}$$



$$\frac{\partial \mathrm{Error}}{\partial w_6} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_6} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_6} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_6} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_6}$$

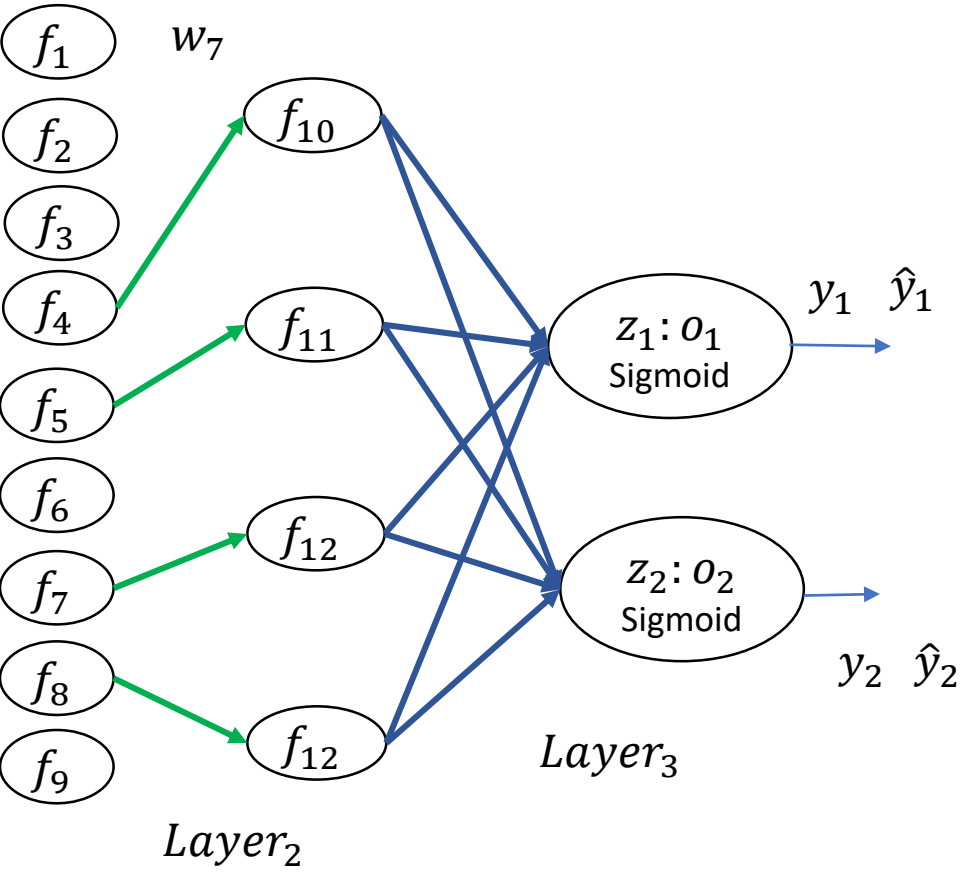
$$\frac{\partial f_{10}}{\partial w_6} = \frac{\partial (w_5 * f_1 + w_6 * f_2 + w_7 * f_4 + w_8 * f_5)}{\partial w_6} = f_2 = 0$$

$$\frac{\partial f_{11}}{\partial w_6} = \frac{\partial (w_5 * f_2 + w_6 * f_3 + w_7 * f_5 + w_8 * f_6)}{\partial w_6} = f_3 = 0$$

$$\frac{\partial f_{12}}{\partial w_6} = \frac{\partial (w_5 * f_4 + w_6 * f_5 + w_7 * f_7 + w_8 * f_8)}{\partial w_6} = f_5 = 0$$

$$\frac{\partial f_{13}}{\partial w_6} = \frac{\partial (w_5 * f_5 + w_6 * f_6 + w_7 * f_8 + w_8 * f_9)}{\partial w_6} = f_6 = 0$$

$$\frac{\partial \mathrm{Error}}{\partial w_6} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_6} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_6} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_6} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_6}$$



$$\frac{\partial \mathrm{Error}}{\partial w_7} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_7} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_7} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_7} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_7}$$

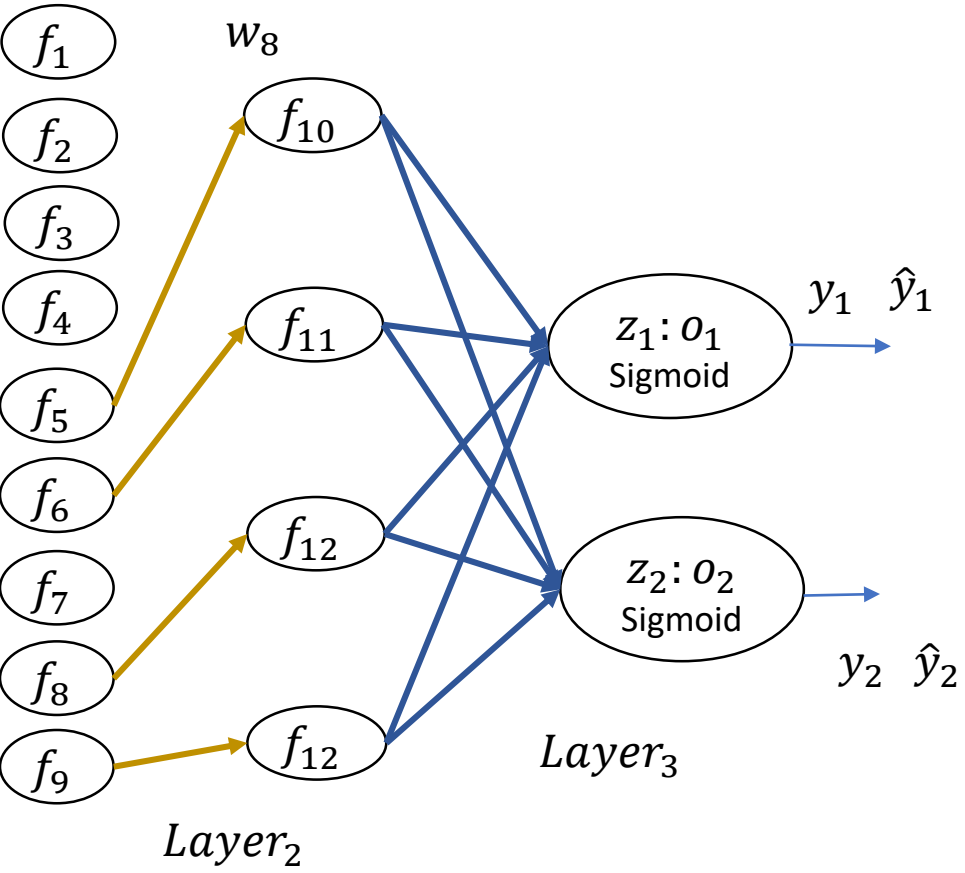
$$\frac{\partial f_{10}}{\partial w_7} = \frac{\partial (w_5 * f_1 + w_6 * f_2 + w_7 * f_4 + w_8 * f_5)}{\partial w_7} = f_4 = 0$$

$$\frac{\partial f_{11}}{\partial w_7} = \frac{\partial (w_5 * f_2 + w_6 * f_3 + w_7 * f_5 + w_8 * f_6)}{\partial w_7} = f_5 = 0$$

$$\frac{\partial f_{12}}{\partial w_7} = \frac{\partial (w_5 * f_4 + w_6 * f_5 + w_7 * f_7 + w_8 * f_8)}{\partial w_7} = f_6 = 0$$

$$\frac{\partial f_{13}}{\partial w_7} = \frac{\partial (w_5 * f_5 + w_6 * f_6 + w_7 * f_8 + w_8 * f_9)}{\partial w_7} = f_8 = 0$$

$$\frac{\partial \mathrm{Error}}{\partial w_7} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_7} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_7} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_7} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_7}$$



$$\frac{\partial \mathrm{Error}}{\partial w_8} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_8} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_8} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_8} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_8}$$

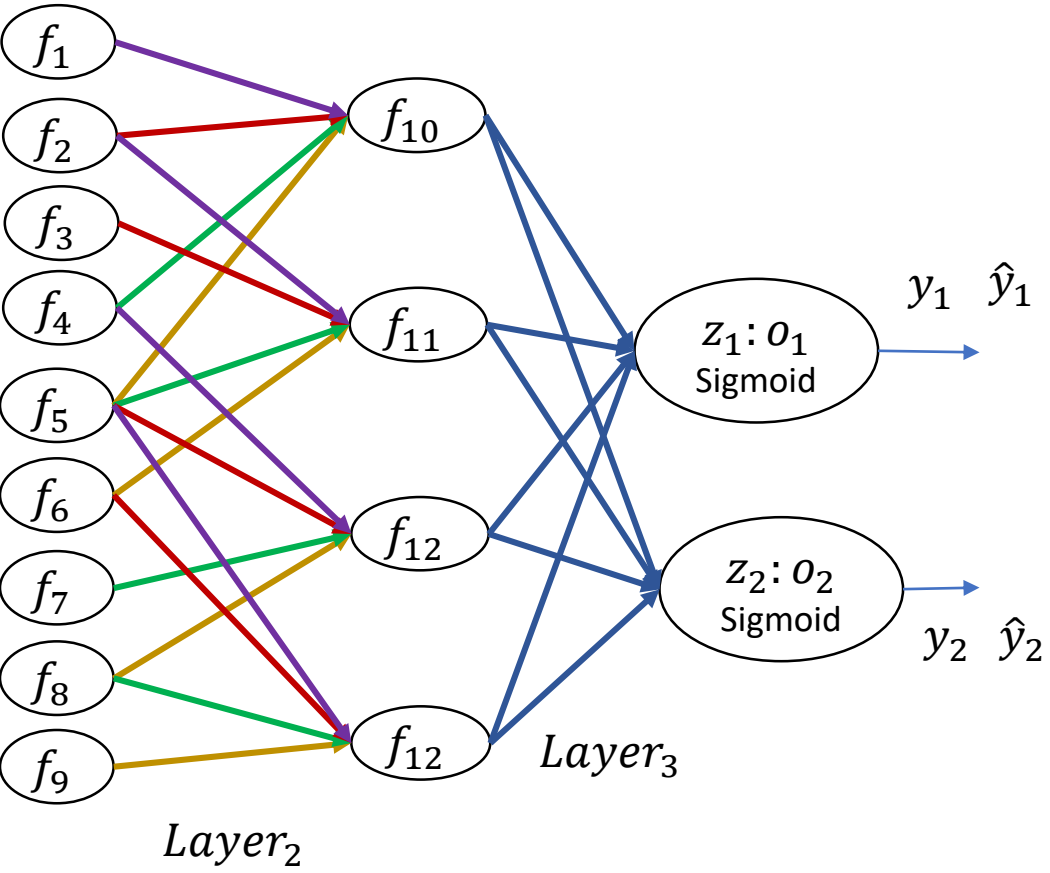
$$\frac{\partial f_{10}}{\partial w_8} = \frac{\partial (w_5 * f_1 + w_6 * f_2 + w_7 * f_4 + w_8 * f_5)}{\partial w_8} = f_5 = 0$$

$$\frac{\partial f_{11}}{\partial w_8} = \frac{\partial (w_5 * f_2 + w_6 * f_3 + w_7 * f_5 + w_8 * f_6)}{\partial w_8} = f_6 = 0$$

$$\frac{\partial f_{12}}{\partial w_8} = \frac{\partial (w_5 * f_4 + w_6 * f_5 + w_7 * f_7 + w_8 * f_8)}{\partial w_8} = f_8 = 0$$

$$\frac{\partial f_{13}}{\partial w_8} = \frac{\partial (w_5 * f_5 + w_6 * f_6 + w_7 * f_8 + w_8 * f_9)}{\partial w_8} = f_9 = 0$$

$$\frac{\partial \mathrm{Error}}{\partial w_8} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial w_8} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial w_8} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial w_8} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial w_8}$$



$$\frac{\partial \mathrm{Error}}{\partial f_1} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_1}$$

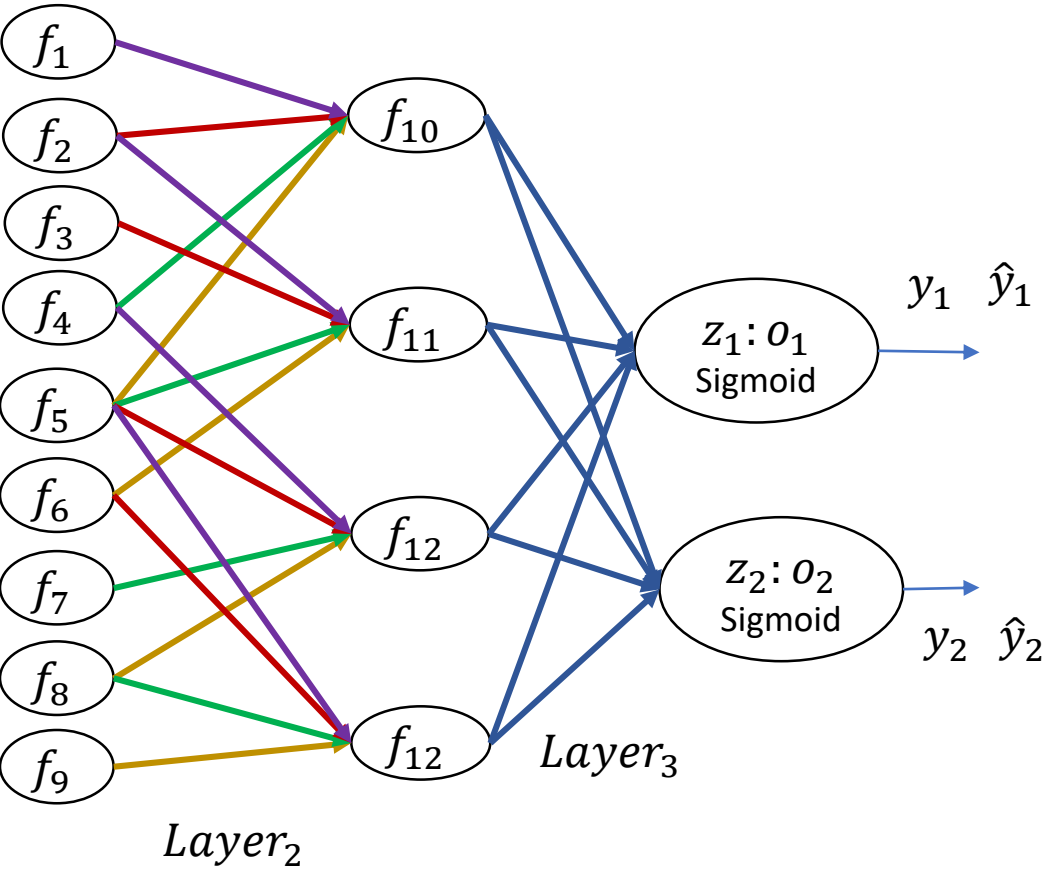
$$\frac{\partial f_{10}}{\partial f_1} = \frac{\partial (w_5 * f_1 + w_6 * f_2 + w_7 * f_4 + w_8 * f_5)}{\partial f_1} = w_5 = 0$$

$$\frac{\partial f_{11}}{\partial f_1} = \frac{\partial (w_5 * f_2 + w_6 * f_3 + w_7 * f_5 + w_8 * f_6)}{\partial f_1} = 0 = 0$$

$$\frac{\partial f_{12}}{\partial f_1} = \frac{\partial (w_5 * f_4 + w_6 * f_5 + w_7 * f_7 + w_8 * f_8)}{\partial f_1} = 0 = 0$$

$$\frac{\partial f_{13}}{\partial f_1} = \frac{\partial (w_5 * f_5 + w_6 * f_6 + w_7 * f_8 + w_8 * f_9)}{\partial f_1} = 0 = 0$$

$$\frac{\partial \mathrm{Error}}{\partial f_1} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_1}$$



$$\frac{\partial \mathrm{Error}}{\partial f_2} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_2}$$

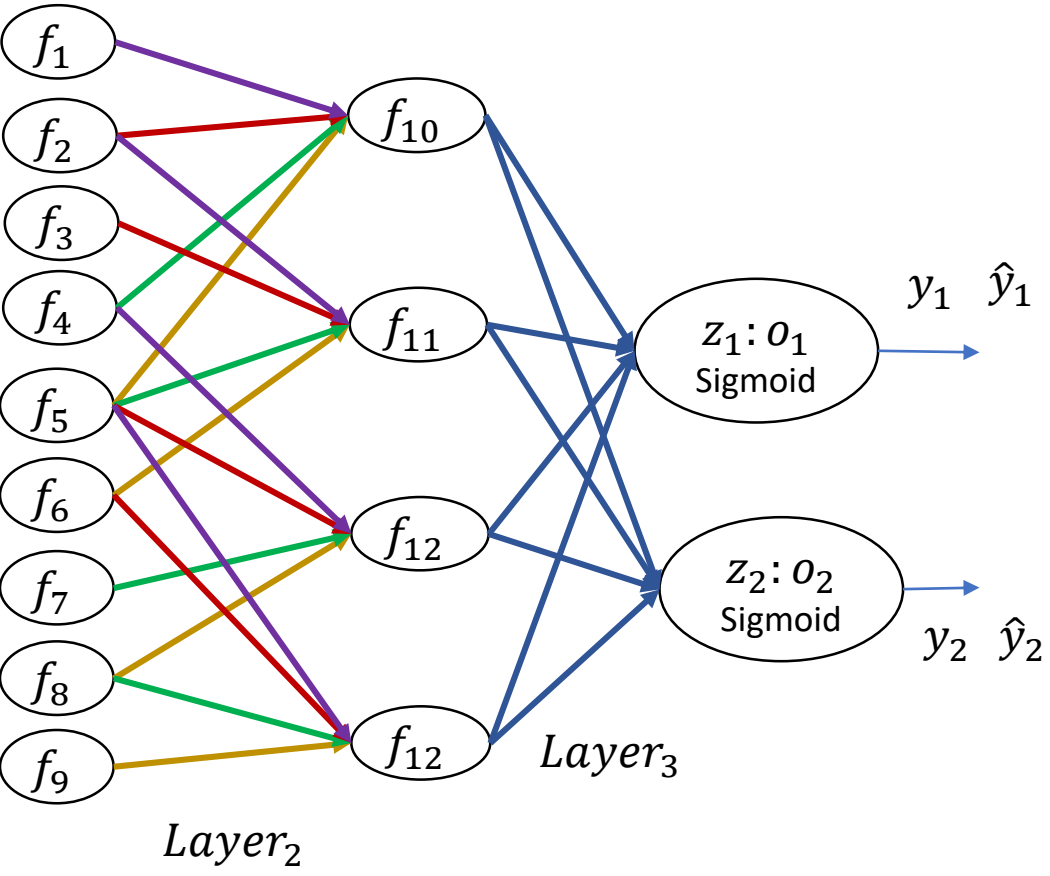
$$\frac{\partial f_{10}}{\partial f_2} = \frac{\partial (w_5 * f_1 + w_6 * f_2 + w_7 * f_4 + w_8 * f_5)}{\partial f_2} = w_6 = 0$$

$$\frac{\partial f_{11}}{\partial f_2} = \frac{\partial (w_5 * f_2 + w_6 * f_3 + w_7 * f_5 + w_8 * f_6)}{\partial f_2} = w_5 = 0$$

$$\frac{\partial f_{12}}{\partial f_2} = \frac{\partial (w_5 * f_4 + w_6 * f_5 + w_7 * f_7 + w_8 * f_8)}{\partial f_2} = 0 = 0$$

$$\frac{\partial f_{13}}{\partial f_2} = \frac{\partial (w_5 * f_5 + w_6 * f_6 + w_7 * f_8 + w_8 * f_9)}{\partial f_2} = 0 = 0$$

$$\frac{\partial \mathrm{Error}}{\partial f_2} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_2}$$



$$\frac{\partial \mathrm{Error}}{\partial f_1} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_1} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_1}$$

$$\frac{\partial \mathrm{Error}}{\partial f_2} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_2} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_2}$$

$$\frac{\partial \mathrm{Error}}{\partial f_3} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_3} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_3} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_3} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_3}$$

$$\frac{\partial \mathrm{Error}}{\partial f_4} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_4} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_4} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_4} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_4}$$

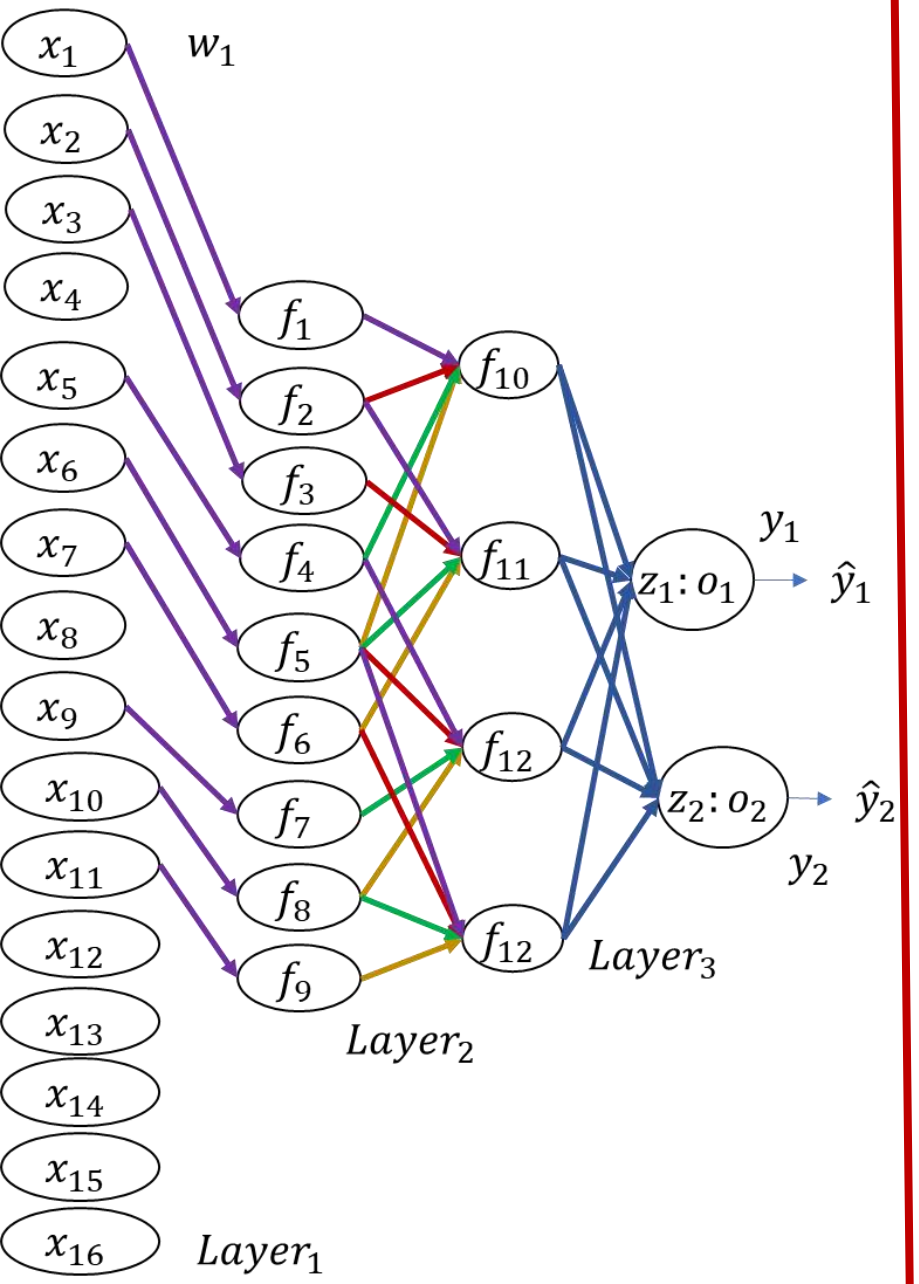
$$\frac{\partial \mathrm{Error}}{\partial f_5} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_5} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_5} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_5} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_5}$$

$$\frac{\partial \mathrm{Error}}{\partial f_6} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_6} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_6} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_6} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_6}$$

$$\frac{\partial \mathrm{Error}}{\partial f_7} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_7} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_7} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_7} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_7}$$

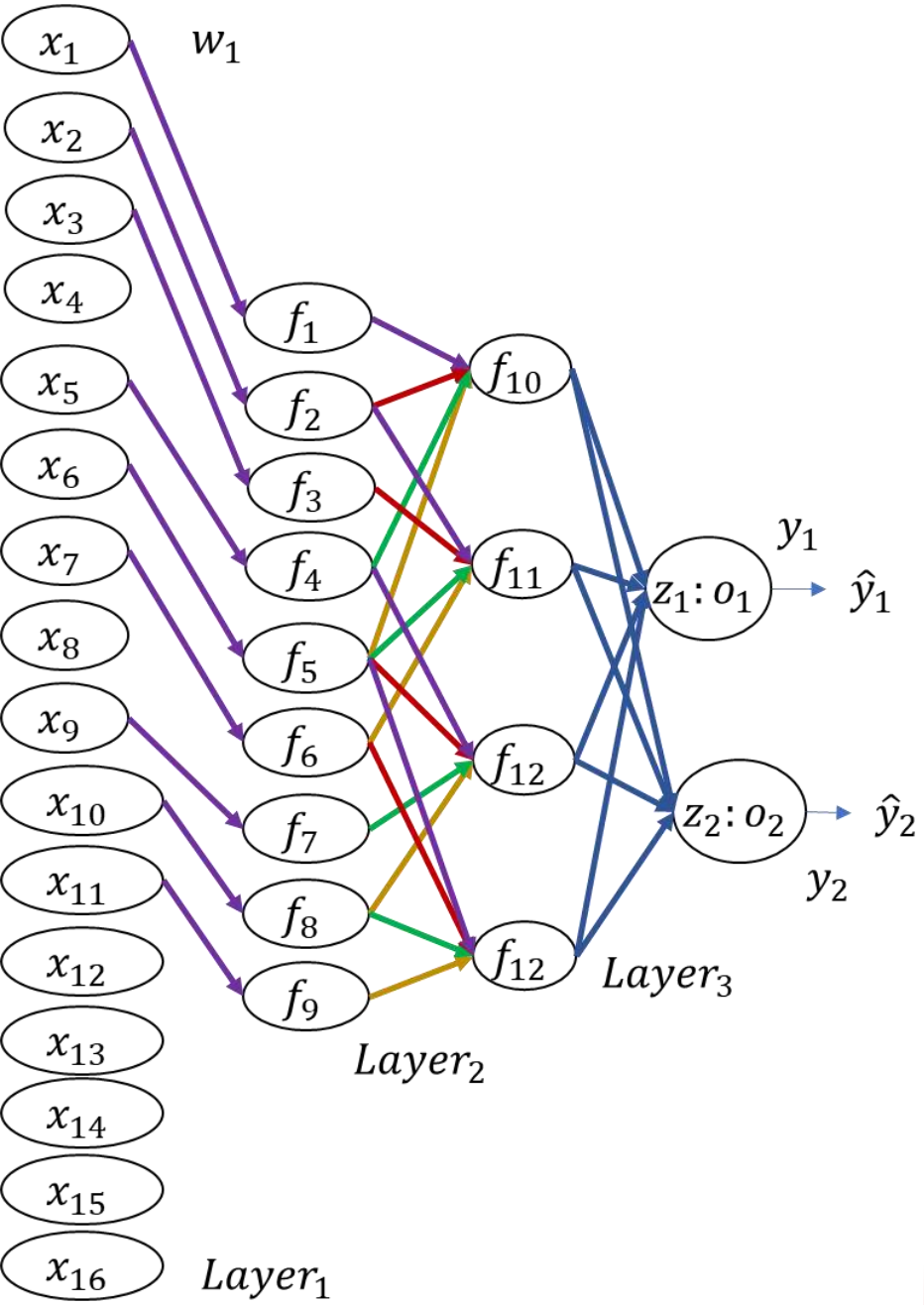
$$\frac{\partial \mathrm{Error}}{\partial f_8} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_8} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_8} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_8} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_8}$$

$$\frac{\partial \mathrm{Error}}{\partial f_9} = \frac{\partial \mathrm{Error}}{\partial f_{10}} X \frac{\partial f_{10}}{\partial f_9} + \frac{\partial \mathrm{Error}}{\partial f_{11}} X \frac{\partial f_{11}}{\partial f_9} + \frac{\partial \mathrm{Error}}{\partial f_{12}} X \frac{\partial f_{12}}{\partial f_9} + \frac{\partial \mathrm{Error}}{\partial f_{13}} X \frac{\partial f_{13}}{\partial f_9}$$



$$\frac{\partial \mathrm{Error}}{\partial w_1} = \frac{\partial \mathrm{Error}}{\partial f_1} X \frac{\partial f_1}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_2} X \frac{\partial f_2}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_3} X \frac{\partial f_3}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_4} X \frac{\partial f_4}{\partial w_1} X$$

$$\frac{\partial \mathrm{Error}}{\partial f_5} X \frac{\partial f_5}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_6} X \frac{\partial f_6}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_7} X \frac{\partial f_7}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_8} X \frac{\partial f_8}{\partial w_1} X \frac{\partial \mathrm{Error}}{\partial f_9} X \frac{\partial f_9}{\partial w_1}$$



$$\frac{\partial f_1}{\partial w_1} = \frac{\partial(w_1 * x_1 + w_2 * x_2 + w_3 * x_5 + w_4 * x_6)}{\partial w_1} = x_1 = 0$$

$$\frac{\partial f_2}{\partial w_1} = \frac{\partial(w_1 * x_2 + w_2 * x_3 + w_3 * x_6 + w_4 * x_7)}{\partial w_1} = x_2 = 0$$

$$\frac{\partial f_3}{\partial w_1} = \frac{\partial(w_1 * x_3 + w_2 * x_4 + w_3 * x_7 + w_4 * x_8)}{\partial w_1} = x_3 = 0$$

$$\frac{\partial f_4}{\partial w_1} = \frac{\partial(w_1 * x_5 + w_2 * x_6 + w_3 * x_9 + w_4 * x_{10})}{\partial w_1} = x_5 = 0$$

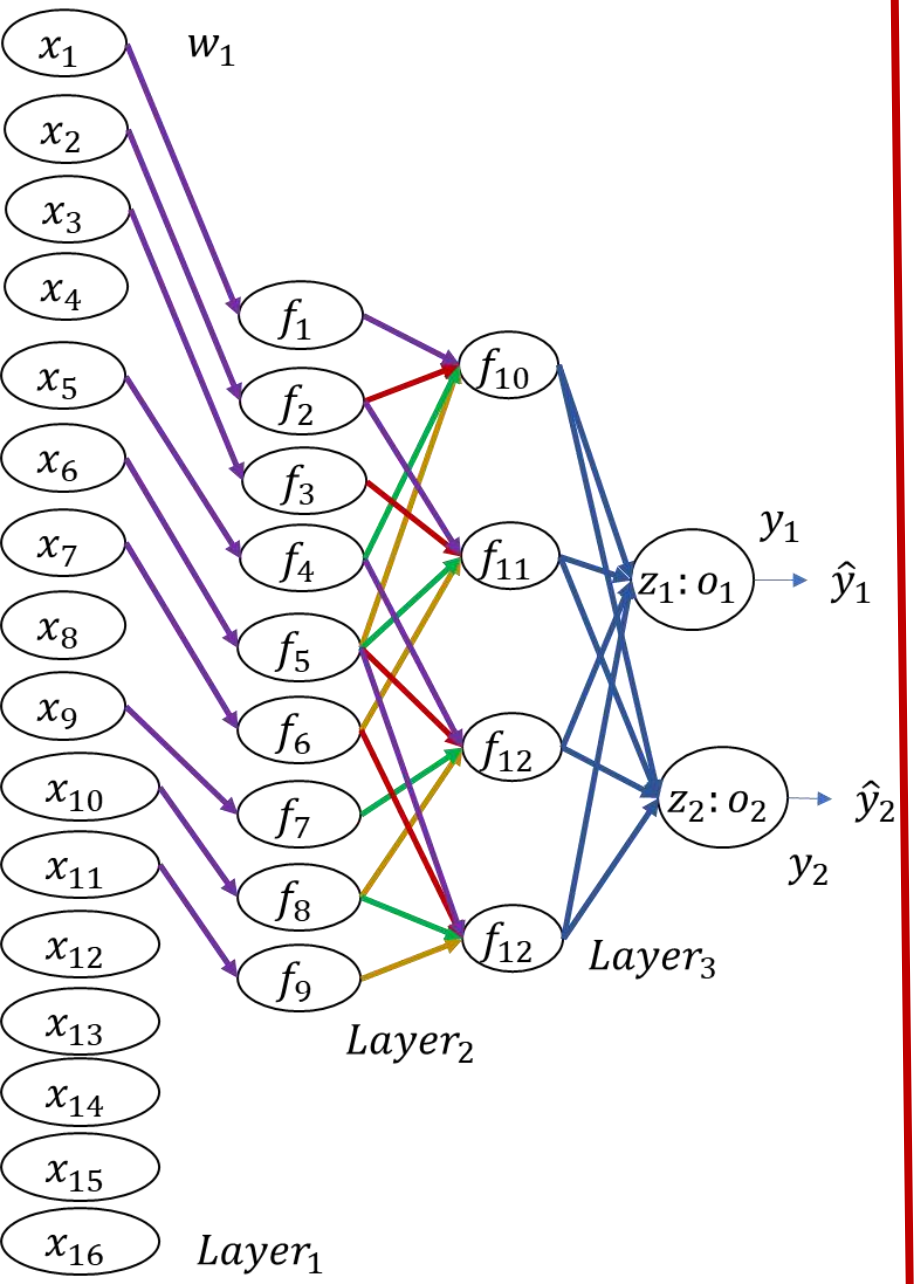
$$\frac{\partial f_5}{\partial w_1} = \frac{\partial(w_1 * x_6 + w_2 * x_7 + w_3 * x_{10} + w_4 * x_{11})}{\partial w_1} = x_6 = 0$$

$$\frac{\partial f_6}{\partial w_1} = \frac{\partial(w_1 * x_7 + w_2 * x_8 + w_3 * x_{11} + w_4 * x_{12})}{\partial w_1} = x_7 = 0$$

$$\frac{\partial f_7}{\partial w_1} = \frac{\partial(w_1 * x_9 + w_2 * x_{10} + w_3 * x_{13} + w_4 * x_{14})}{\partial w_1} = x_9 = 0$$

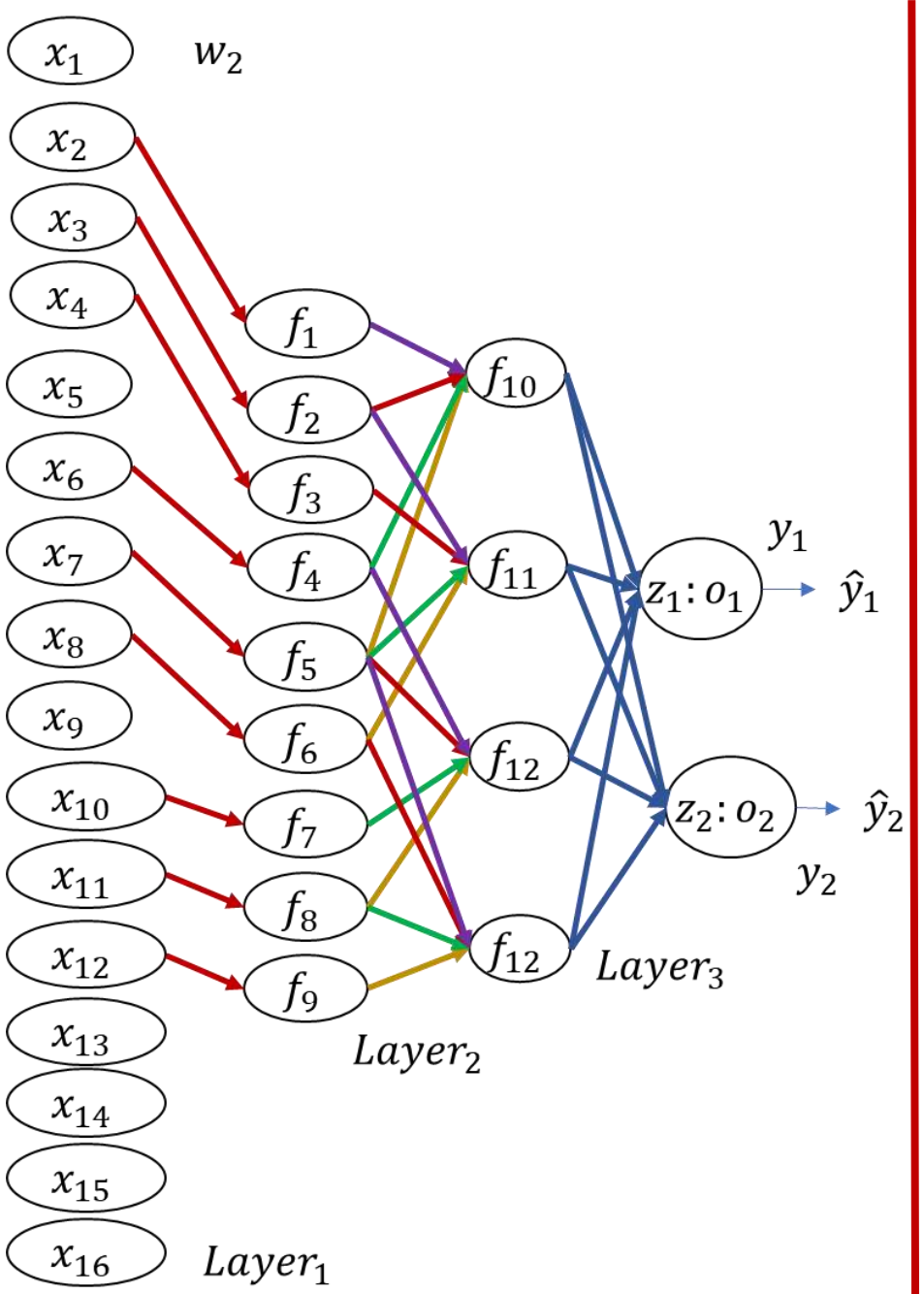
$$\frac{\partial f_8}{\partial w_1} = \frac{\partial(w_1 * x_{10} + w_2 * x_{11} + w_3 * x_{14} + w_4 * x_{15})}{\partial w_1} = x_{10} = 0$$

$$\frac{\partial f_9}{\partial w_1} = \frac{\partial(w_1 * x_{11} + w_2 * x_{12} + w_3 * x_{15} + w_4 * x_{16})}{\partial w_1} = x_{11} = 0$$



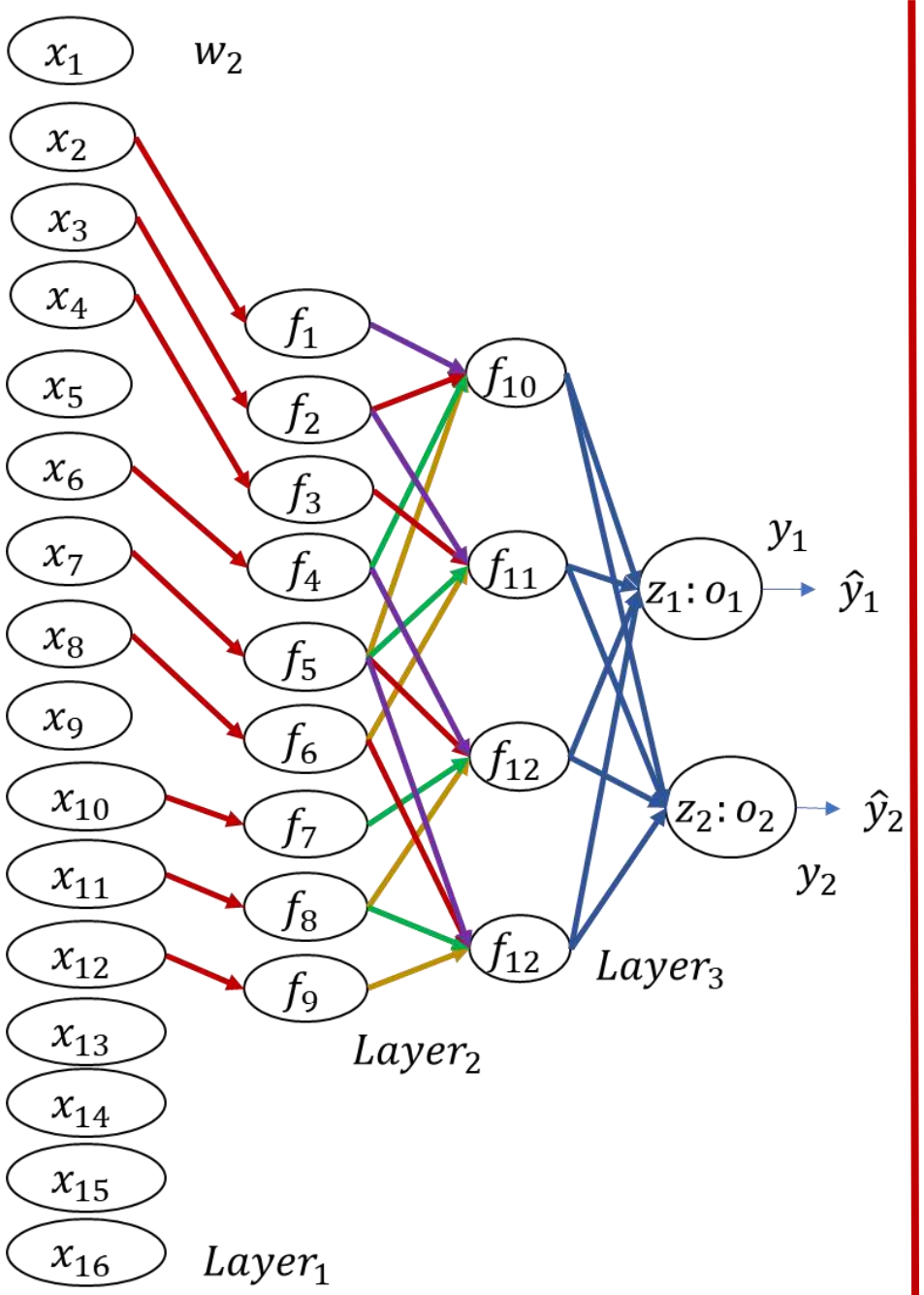
$$\frac{\partial \mathrm{Error}}{\partial w_1} = \frac{\partial \mathrm{Error}}{\partial f_1} X \frac{\partial f_1}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_2} X \frac{\partial f_2}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_3} X \frac{\partial f_3}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_4} X \frac{\partial f_4}{\partial w_1} X$$

$$\frac{\partial \mathrm{Error}}{\partial f_5} X \frac{\partial f_5}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_6} X \frac{\partial f_6}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_7} X \frac{\partial f_7}{\partial w_1} + \frac{\partial \mathrm{Error}}{\partial f_8} X \frac{\partial f_8}{\partial w_1} X \frac{\partial \mathrm{Error}}{\partial f_9} X \frac{\partial f_9}{\partial w_1}$$



$$\frac{\partial \mathrm{Error}}{\partial w_2} = \frac{\partial \mathrm{Error}}{\partial f_1} X \frac{\partial f_1}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_2} X \frac{\partial f_2}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_3} X \frac{\partial f_3}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_4} X \frac{\partial f_4}{\partial w_2} X$$

$$\frac{\partial \mathrm{Error}}{\partial f_5} X \frac{\partial f_5}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_6} X \frac{\partial f_6}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_7} X \frac{\partial f_7}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_8} X \frac{\partial f_8}{\partial w_2} X \frac{\partial \mathrm{Error}}{\partial f_9} X \frac{\partial f_9}{\partial w_2}$$



$$\frac{\partial f_1}{\partial w_2} = \frac{\partial(w_1 * x_1 + w_2 * x_2 + w_3 * x_5 + w_4 * x_6)}{\partial w_2} = x_2 = 0$$

$$\frac{\partial f_2}{\partial w_2} = \frac{\partial(w_1 * x_2 + w_2 * x_3 + w_3 * x_6 + w_4 * x_7)}{\partial w_2} = x_3 = 0$$

$$\frac{\partial f_3}{\partial w_2} = \frac{\partial(w_1 * x_3 + w_2 * x_4 + w_3 * x_7 + w_4 * x_8)}{\partial w_2} = x_4 = 0$$

$$\frac{\partial f_4}{\partial w_2} = \frac{\partial(w_1 * x_5 + w_2 * x_6 + w_3 * x_9 + w_4 * x_{10})}{\partial w_2} = x_6 = 0$$

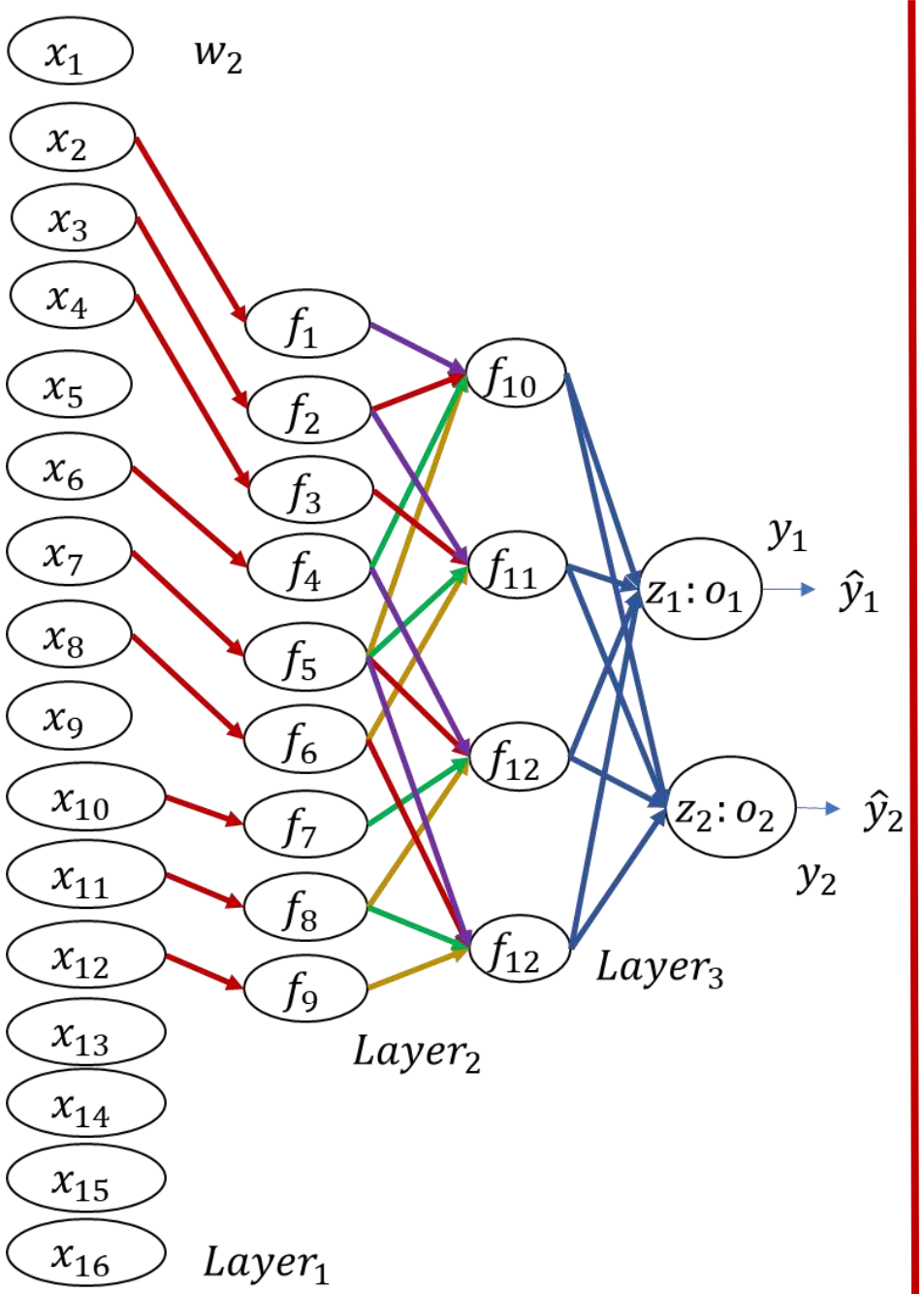
$$\frac{\partial f_5}{\partial w_2} = \frac{\partial(w_1 * x_6 + w_2 * x_7 + w_3 * x_{10} + w_4 * x_{11})}{\partial w_2} = x_7 = 0$$

$$\frac{\partial f_6}{\partial w_2} = \frac{\partial(w_1 * x_7 + w_2 * x_8 + w_3 * x_{11} + w_4 * x_{12})}{\partial w_2} = x_8 = 0$$

$$\frac{\partial f_7}{\partial w_2} = \frac{\partial(w_1 * x_9 + w_2 * x_{10} + w_3 * x_{13} + w_4 * x_{14})}{\partial w_2} = x_{10} = 0$$

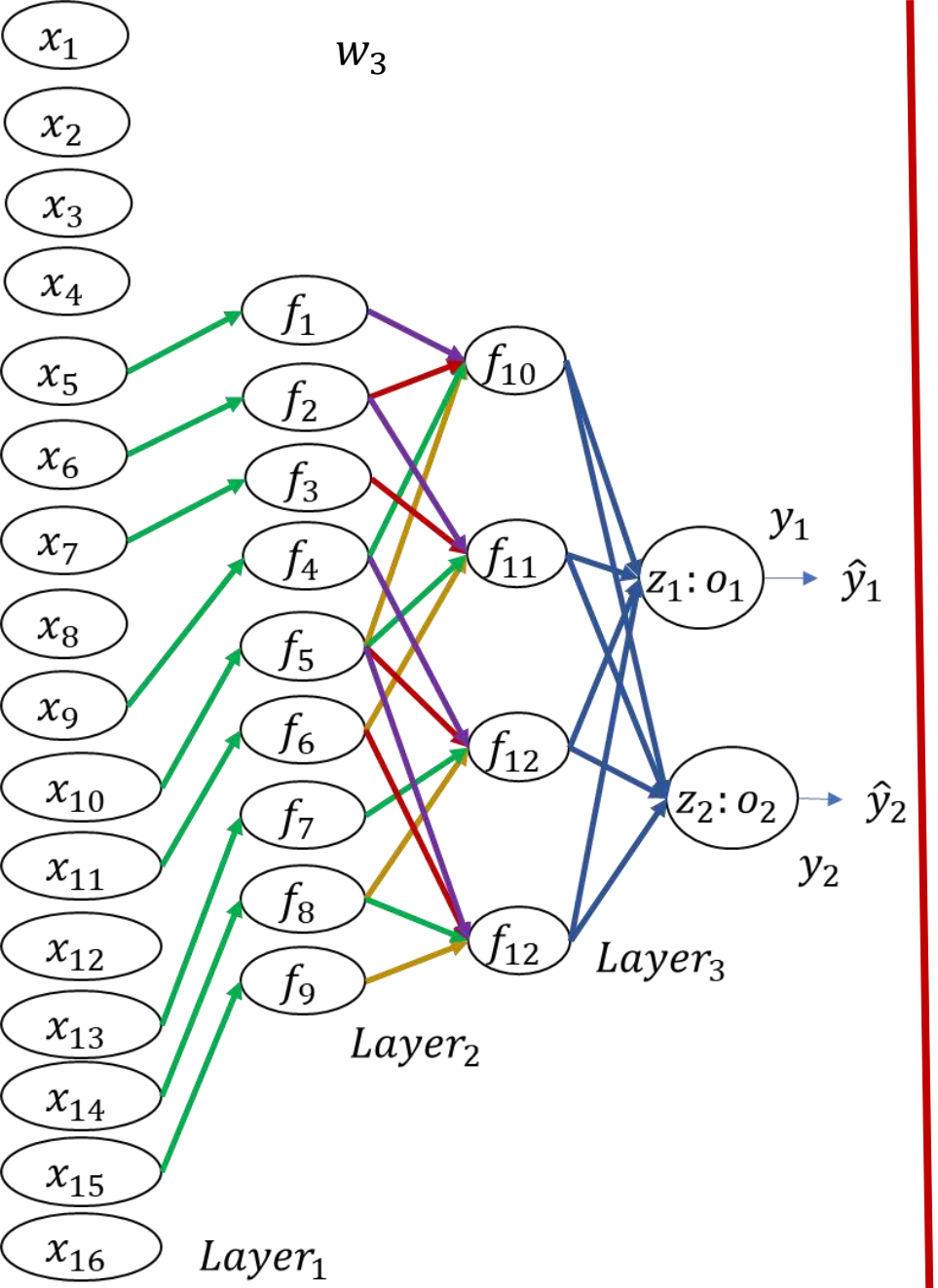
$$\frac{\partial f_8}{\partial w_2} = \frac{\partial(w_1 * x_{10} + w_2 * x_{11} + w_3 * x_{14} + w_4 * x_{15})}{\partial w_2} = x_{11} = 0$$

$$\frac{\partial f_9}{\partial w_2} = \frac{\partial(w_1 * x_{11} + w_2 * x_{12} + w_3 * x_{15} + w_4 * x_{16})}{\partial w_2} = x_{12} = 0$$



$$\frac{\partial \mathrm{Error}}{\partial w_2} = \frac{\partial \mathrm{Error}}{\partial f_1} X \frac{\partial f_1}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_2} X \frac{\partial f_2}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_3} X \frac{\partial f_3}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_4} X \frac{\partial f_4}{\partial w_2} X$$

$$\frac{\partial \mathrm{Error}}{\partial f_5} X \frac{\partial f_5}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_6} X \frac{\partial f_6}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_7} X \frac{\partial f_7}{\partial w_2} + \frac{\partial \mathrm{Error}}{\partial f_8} X \frac{\partial f_8}{\partial w_2} X \frac{\partial \mathrm{Error}}{\partial f_9} X \frac{\partial f_9}{\partial w_2}$$



$$\frac{\partial f_1}{\partial w_3} = \frac{\partial(w_1 * x_1 + w_2 * x_2 + w_3 * x_5 + w_4 * x_6)}{\partial w_3} = x_2 = 0$$

$$\frac{\partial f_2}{\partial w_3} = \frac{\partial(w_1 * x_2 + w_2 * x_3 + w_3 * x_6 + w_4 * x_7)}{\partial w_3} = x_3 = 0$$

$$\frac{\partial f_3}{\partial w_3} = \frac{\partial(w_1 * x_3 + w_2 * x_4 + w_3 * x_7 + w_4 * x_8)}{\partial w_3} = x_4 = 0$$

$$\frac{\partial f_4}{\partial w_3} = \frac{\partial(w_1 * x_5 + w_2 * x_6 + w_3 * x_9 + w_4 * x_{10})}{\partial w_3} = x_6 = 0$$

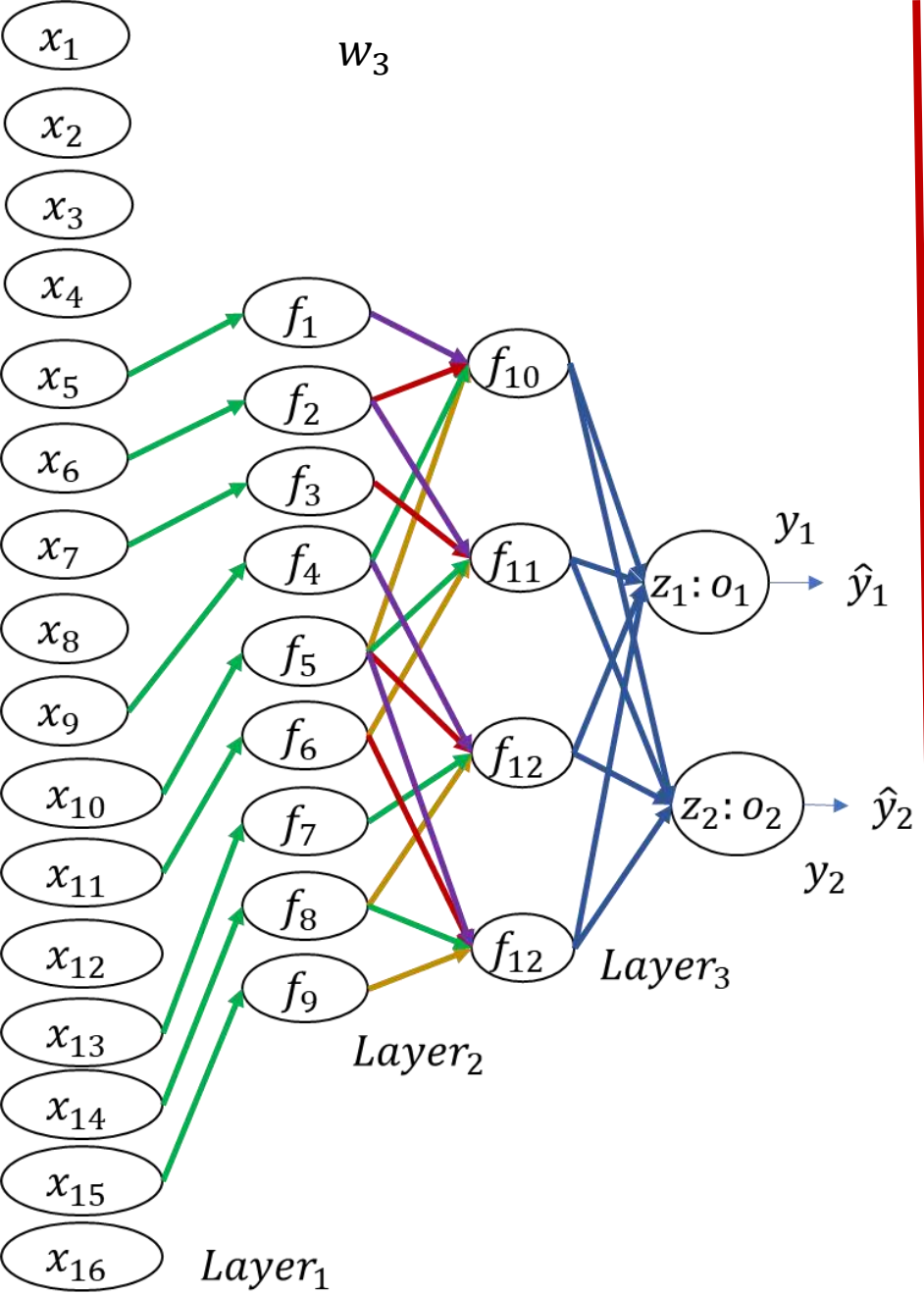
$$\frac{\partial f_5}{\partial w_3} = \frac{\partial(w_1 * x_6 + w_2 * x_7 + w_3 * x_{10} + w_4 * x_{11})}{\partial w_3} = x_7 = 0$$

$$\frac{\partial f_6}{\partial w_3} = \frac{\partial(w_1 * x_7 + w_2 * x_8 + w_3 * x_{11} + w_4 * x_{12})}{\partial w_3} = x_8 = 0$$

$$\frac{\partial f_7}{\partial w_3} = \frac{\partial(w_1 * x_9 + w_2 * x_{10} + w_3 * x_{13} + w_4 * x_{14})}{\partial w_3} = x_{10} = 0$$

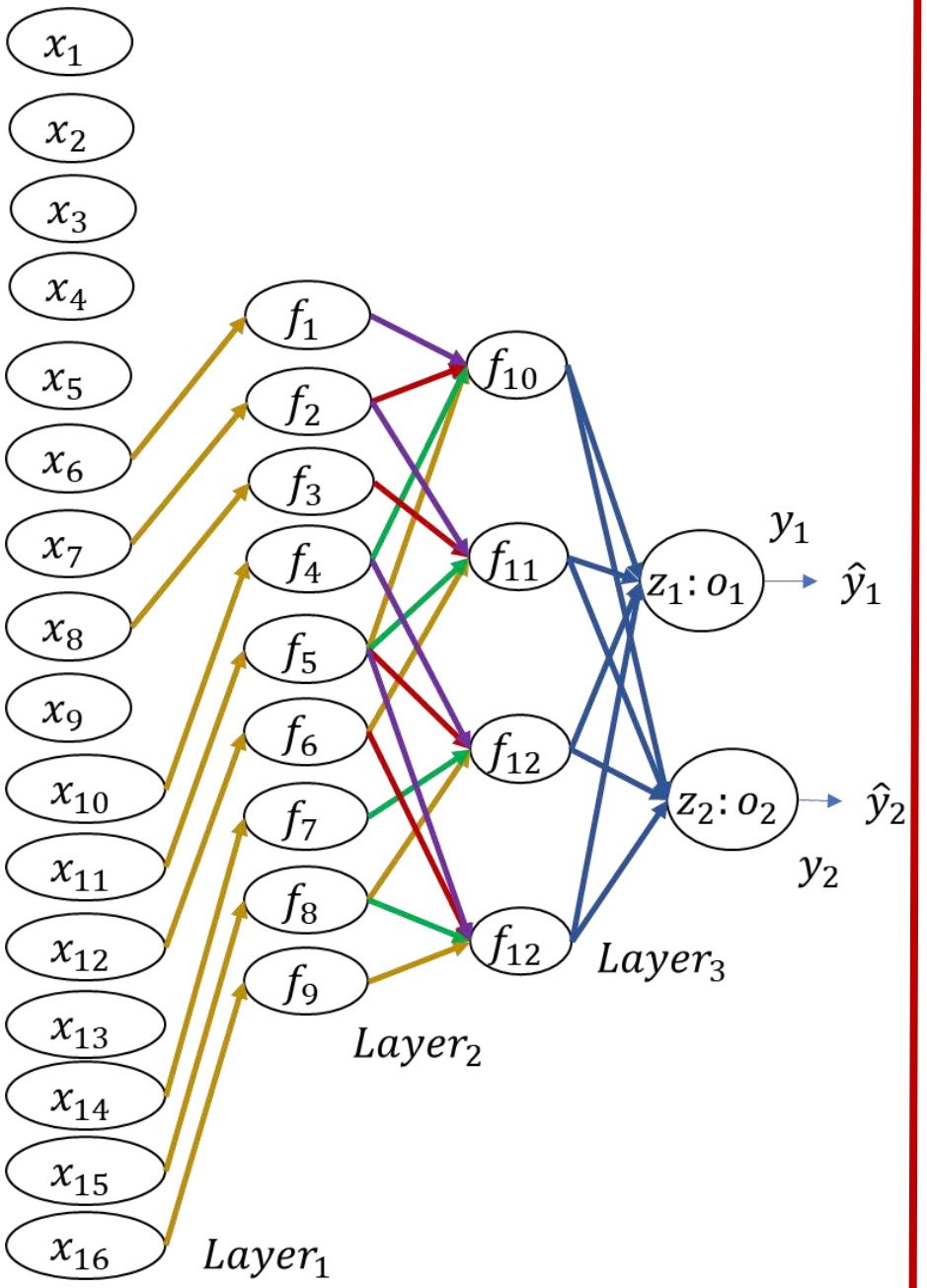
$$\frac{\partial f_8}{\partial w_3} = \frac{\partial(w_1 * x_{10} + w_2 * x_{11} + w_3 * x_{14} + w_4 * x_{15})}{\partial w_3} = x_{11} = 0$$

$$\frac{\partial f_9}{\partial w_3} = \frac{\partial(w_1 * x_{11} + w_2 * x_{12} + w_3 * x_{15} + w_4 * x_{16})}{\partial w_3} = x_{12} = 0$$



$$\frac{\partial \mathrm{Error}}{\partial w_3} = \frac{\partial \mathrm{Error}}{\partial f_1} X \frac{\partial f_1}{\partial w_3} + \frac{\partial \mathrm{Error}}{\partial f_2} X \frac{\partial f_2}{\partial w_3} + \frac{\partial \mathrm{Error}}{\partial f_3} X \frac{\partial f_3}{\partial w_3} + \frac{\partial \mathrm{Error}}{\partial f_4} X \frac{\partial f_4}{\partial w_3} X$$

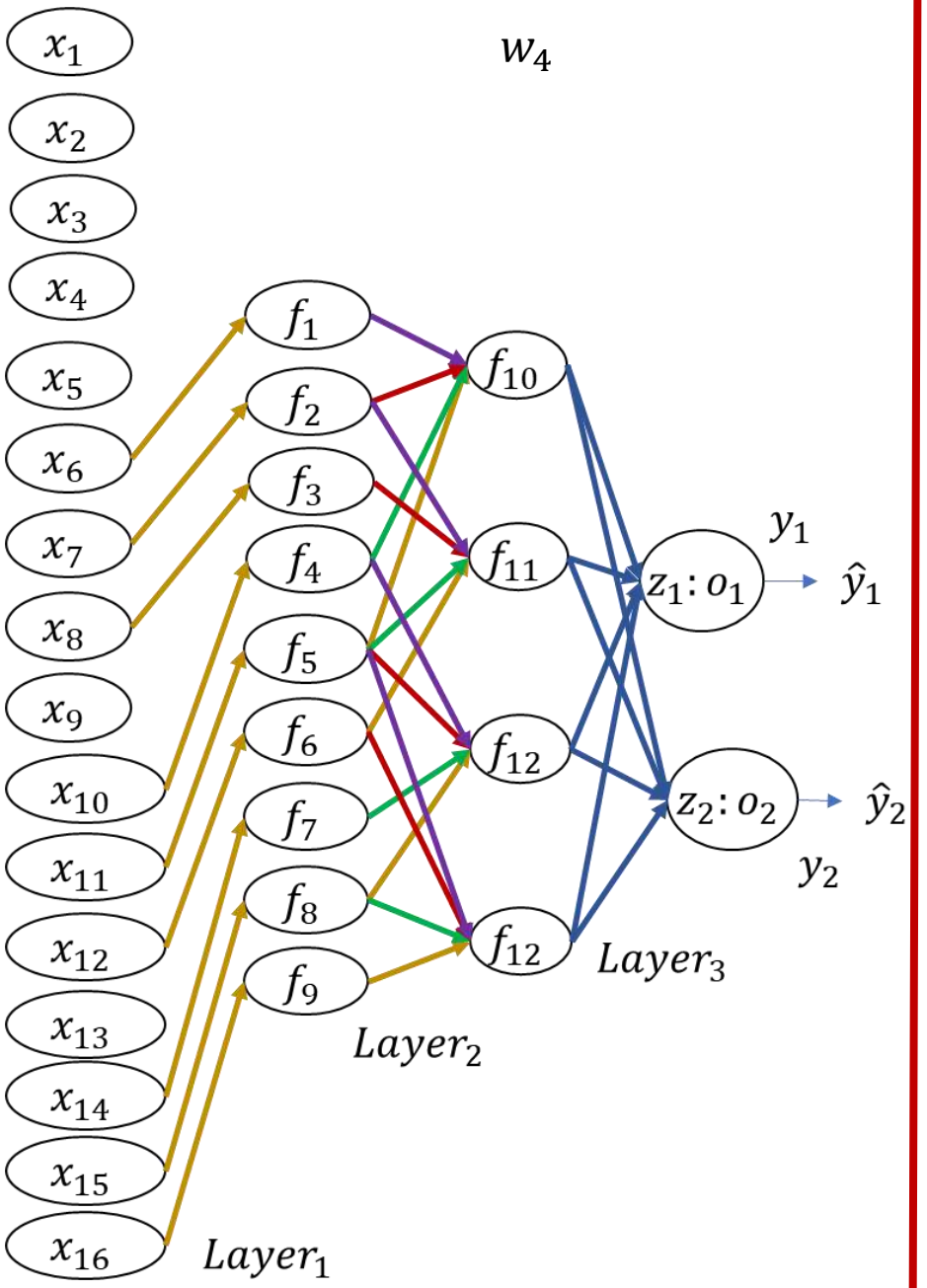
$$\frac{\partial \mathrm{Error}}{\partial f_5} X \frac{\partial f_5}{\partial w_3} + \frac{\partial \mathrm{Error}}{\partial f_6} X \frac{\partial f_6}{\partial w_3} + \frac{\partial \mathrm{Error}}{\partial f_7} X \frac{\partial f_7}{\partial w_3} + \frac{\partial \mathrm{Error}}{\partial f_8} X \frac{\partial f_8}{\partial w_3} X \frac{\partial \mathrm{Error}}{\partial f_9} X \frac{\partial f_9}{\partial w_3}$$



w_4

$$\frac{\partial \mathrm{Error}}{\partial w_4} = \frac{\partial \mathrm{Error}}{\partial f_1} X \frac{\partial f_1}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_2} X \frac{\partial f_2}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_3} X \frac{\partial f_3}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_4} X \frac{\partial f_4}{\partial w_4} X$$

$$\frac{\partial \mathrm{Error}}{\partial f_5} X \frac{\partial f_5}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_6} X \frac{\partial f_6}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_7} X \frac{\partial f_7}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_8} X \frac{\partial f_8}{\partial w_4} X \frac{\partial \mathrm{Error}}{\partial f_9} X \frac{\partial f_9}{\partial w_4}$$



$$\frac{\partial f_1}{\partial w_4} = \frac{\partial(w_1 * x_1 + w_2 * x_2 + w_3 * x_5 + w_4 * x_6)}{\partial w_4} = x_2 = 0$$

$$\frac{\partial f_2}{\partial w_4} = \frac{\partial(w_1 * x_2 + w_2 * x_3 + w_3 * x_6 + w_4 * x_7)}{\partial w_4} = x_3 = 0$$

$$\frac{\partial f_3}{\partial w_4} = \frac{\partial(w_1 * x_3 + w_2 * x_4 + w_3 * x_7 + w_4 * x_8)}{\partial w_4} = x_4 = 0$$

$$\frac{\partial f_4}{\partial w_4} = \frac{\partial(w_1 * x_5 + w_2 * x_6 + w_3 * x_9 + w_4 * x_{10})}{\partial w_4} = x_6 = 0$$

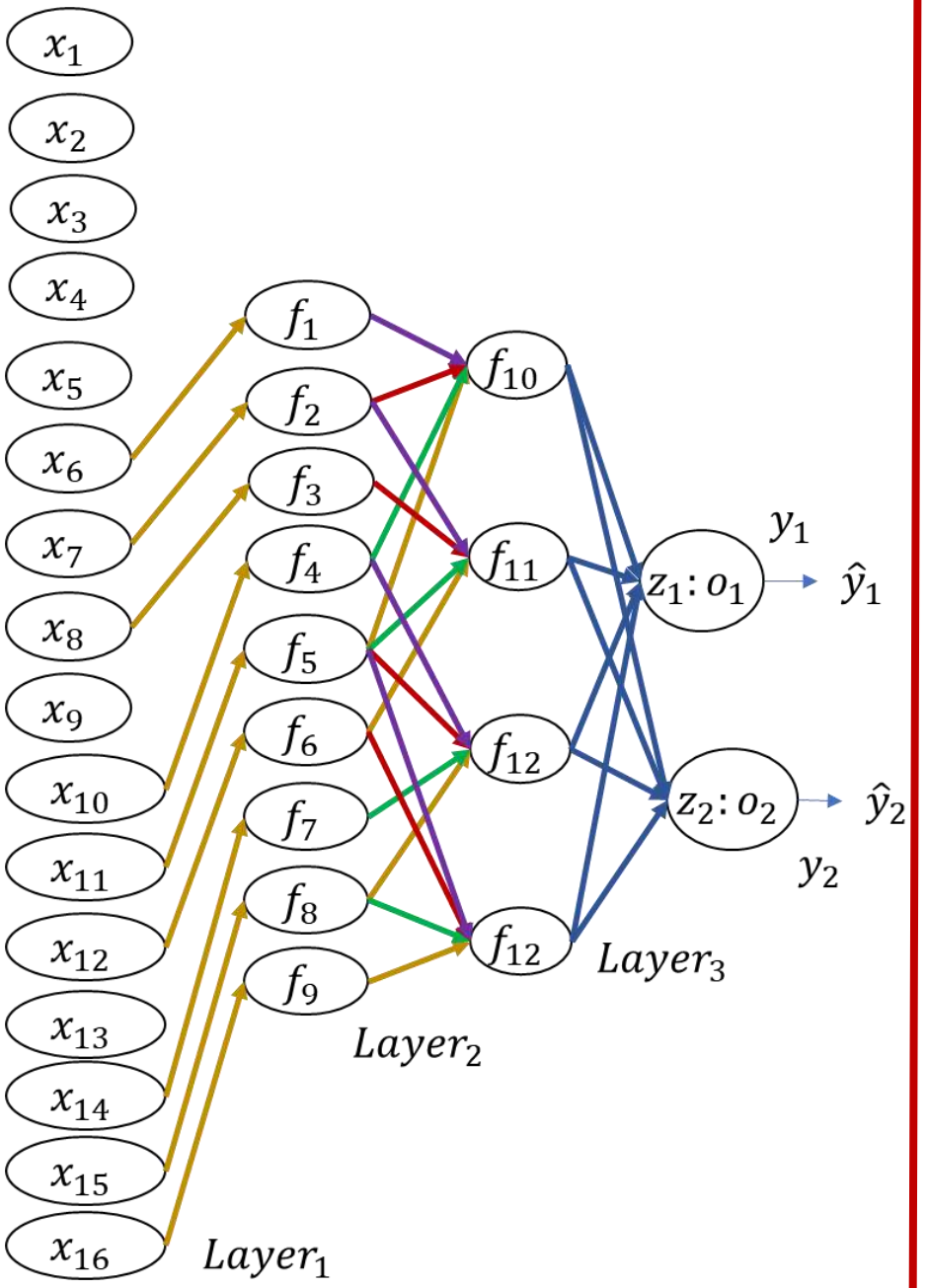
$$\frac{\partial f_5}{\partial w_4} = \frac{\partial(w_1 * x_6 + w_2 * x_7 + w_3 * x_{10} + w_4 * x_{11})}{\partial w_4} = x_7 = 0$$

$$\frac{\partial f_6}{\partial w_4} = \frac{\partial(w_1 * x_7 + w_2 * x_8 + w_3 * x_{11} + w_4 * x_{12})}{\partial w_4} = x_8 = 0$$

$$\frac{\partial f_7}{\partial w_4} = \frac{\partial(w_1 * x_9 + w_2 * x_{10} + w_3 * x_{13} + w_4 * x_{14})}{\partial w_4} = x_{10} = 0$$

$$\frac{\partial f_8}{\partial w_4} = \frac{\partial(w_1 * x_{10} + w_2 * x_{11} + w_3 * x_{14} + w_4 * x_{15})}{\partial w_4} = x_{11} = 0$$

$$\frac{\partial f_9}{\partial w_4} = \frac{\partial(w_1 * x_{11} + w_2 * x_{12} + w_3 * x_{15} + w_4 * x_{16})}{\partial w_4} = x_{12} = 0$$



w_4

$$\frac{\partial \mathrm{Error}}{\partial w_4} = \frac{\partial \mathrm{Error}}{\partial f_1} X \frac{\partial f_1}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_2} X \frac{\partial f_2}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_3} X \frac{\partial f_3}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_4} X \frac{\partial f_4}{\partial w_4} X$$

$$\frac{\partial \mathrm{Error}}{\partial f_5} X \frac{\partial f_5}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_6} X \frac{\partial f_6}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_7} X \frac{\partial f_7}{\partial w_4} + \frac{\partial \mathrm{Error}}{\partial f_8} X \frac{\partial f_8}{\partial w_4} X \frac{\partial \mathrm{Error}}{\partial f_9} X \frac{\partial f_9}{\partial w_4}$$

How To Improve the Model

Scope of Improvement

- More data may be required
- Data needs to have more diversity
- Algorithm needs longer training
- More hidden layers or hidden units are required
- Add Regularization
- Change the Neural network architecture like activation function etc.
- There are many other considerations you can think of.

Dataset Splitting

Training set	Which you run your learning algorithm on.
Dev (development) set or Validation Set	Which you use to tune parameters, select features, and make other decisions regarding the learning algorithm. Sometimes also called the **hold-out cross validation set** .
Test set	which you use to evaluate the performance of the algorithm, but not to make any decisions regarding what learning algorithm or parameters to use.

Dataset Splitting

Training Set	Dev Test or Hold Out Cross Validation Set	Test Set

Traditional Style partitioning 70/30 or 60/20/20

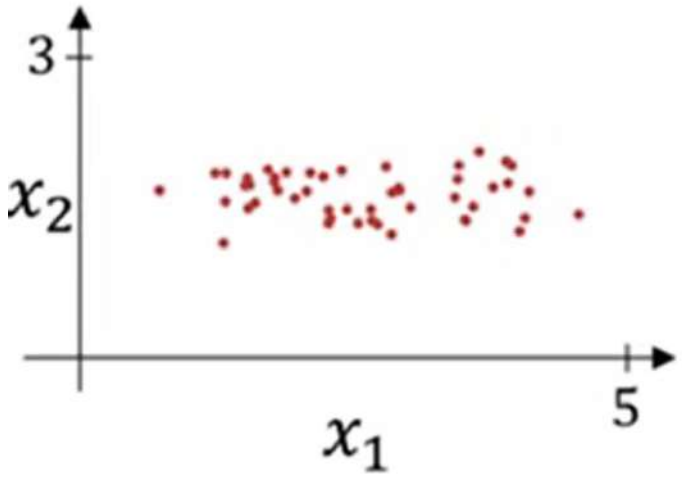
But in the era of Deep Learning may even go down to 99 0.5 0.5

If the data size is 1,00,0000 then 5000 5000 data size will still be there in dev and test sets

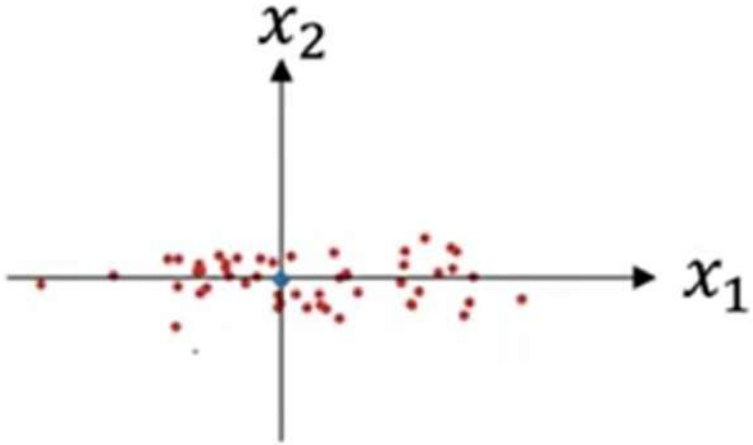
Val & Test Set

- The purpose of the dev and test sets are to direct your team toward the most important changes to make to the machine learning system
- Very important that dev and test set reflect data you expect to get in the future
- Bad distribution will severely restrict analysis to guess that why test data is not giving good results
- It is naturally good to have the data in all the sets from the same distribution.
    - For example Housing data coming from Mumbai and we are trying to find the house prices in Chandigarh.
    - Else wasting a lot of time in improving the performance of dev set and then finding out that it is not working well for the test set.

Normalizing Data Sets



Original data



Subtract mean

$$\mu = \frac{1}{m}\Sigma_{l=1}^{m} x^{(i)}$$

$$x = x - \mu$$

Normalization

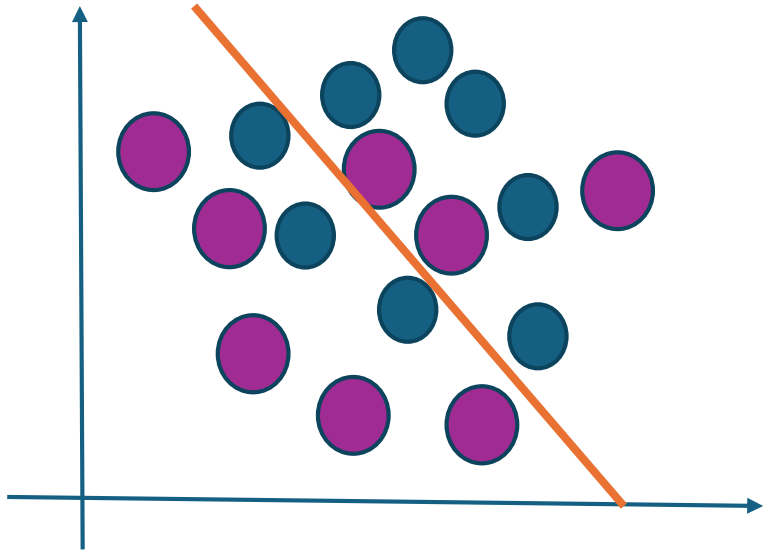
- **Stabilizes Training:**
  - Stabilizing the gradient descent & ensures gradients do not become too large or too small.
  - In deep networks where small changes in input can lead to large changes in output.
- **Speeds Up Convergence:**
  - Model may converge faster.
  - Optimizer can navigate the loss landscape more efficiently when the features are on a similar scale

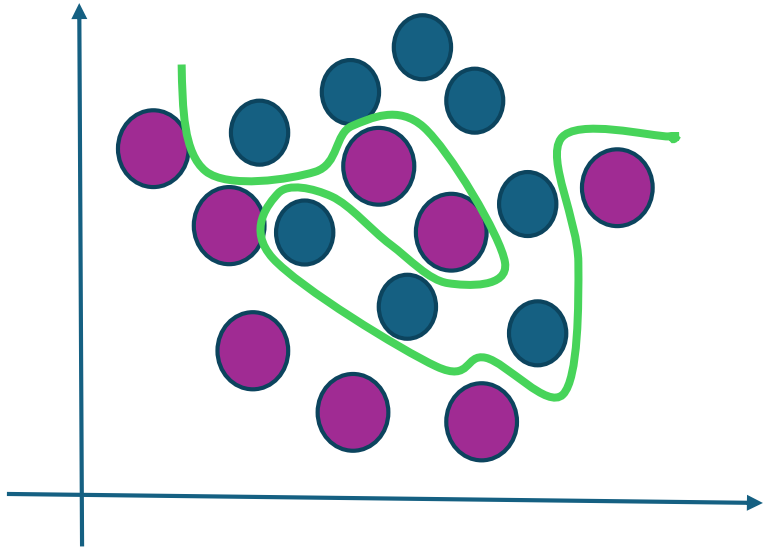
Batch GD, Mini-Batch GD & Stochastic GD

Aspect	Batch Gradient Descent	Mini-Batch Gradient Descent	Stochastic Gradient Descent (SGD)
Definition	Uses the entire training dataset to compute the gradient and update the model parameters.	Uses a subset of the training data (mini-batch) to compute the gradient and update the model parameters.	Uses a single training example to compute the gradient and update the model parameters.
Pros	- Stable and accurate estimate of the gradient.	- Balances stability and speed Allows for more frequent updates - Can lead to faster convergence.	- Very fast updates - Can handle large datasets efficiently - Introduces noise that can help escape local minima.
Cons	- Slow and computationally expensive with large datasets.	- Choice of mini-batch size affects performance - Too small: noisy updates - Too large: slow training.	- High variance in updates can lead to noisy gradients - May require more iterations to converge.
Update Frequency	Once per epoch (entire dataset).	Multiple times per epoch (based on mini-batch size).	Once per training example.
Memory Usage	High, as it requires loading the entire dataset into memory.	Lower, as only a subset of the data is loaded at a time.	Very low, as only one training example is loaded at a time.
Convergence	Can be slow due to the large amount of data processed at once.	Typically faster due to more frequent updates.	Can be very fast but may be noisy and require more iterations.

Regularization

Why

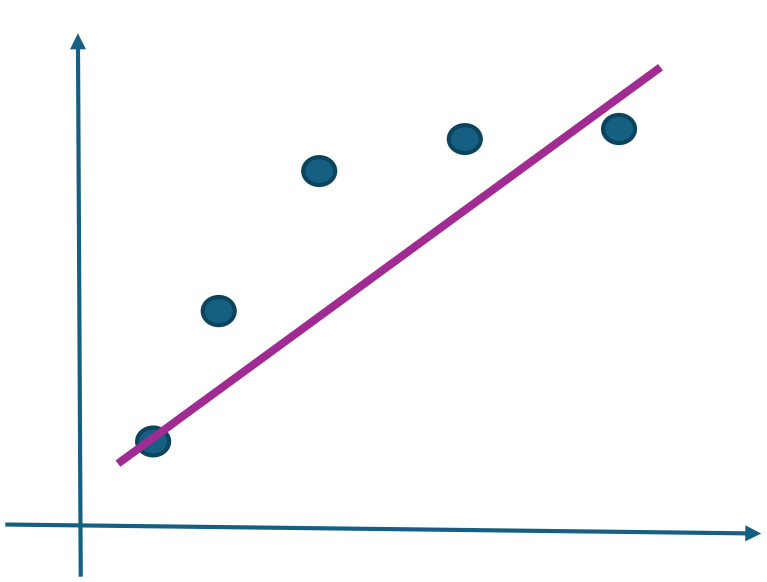




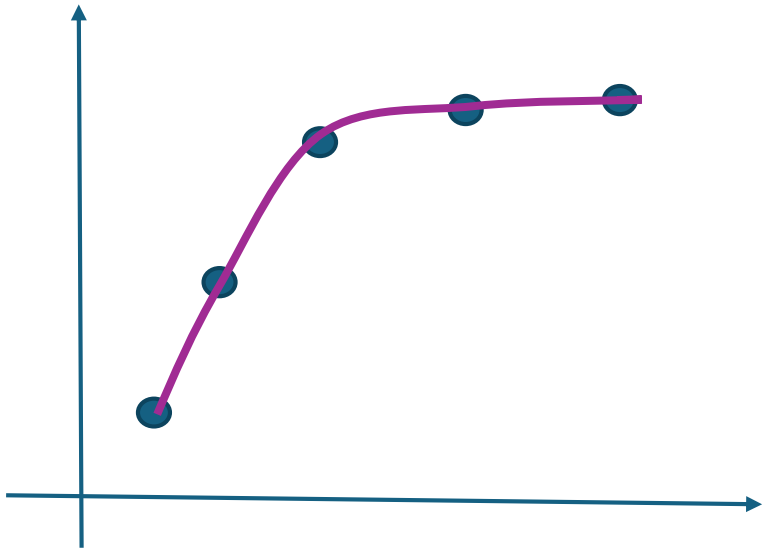
- Model learns to perform extremely well on training data but fails to generalize to new, unseen data (Test Data)
- Overfitting happens when the model is too complex, allowing it to capture noise and irrelevant patterns in the training data.

- Regularization is a technique used to prevent overfitting by adding a penalty to the model's complexity.
- Helps models generalize better to new, unseen data.

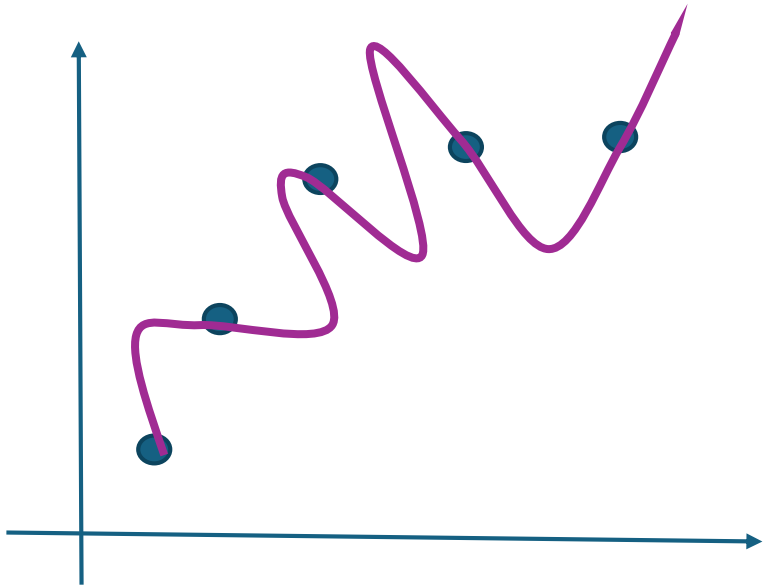
Why



Underfit High Bias



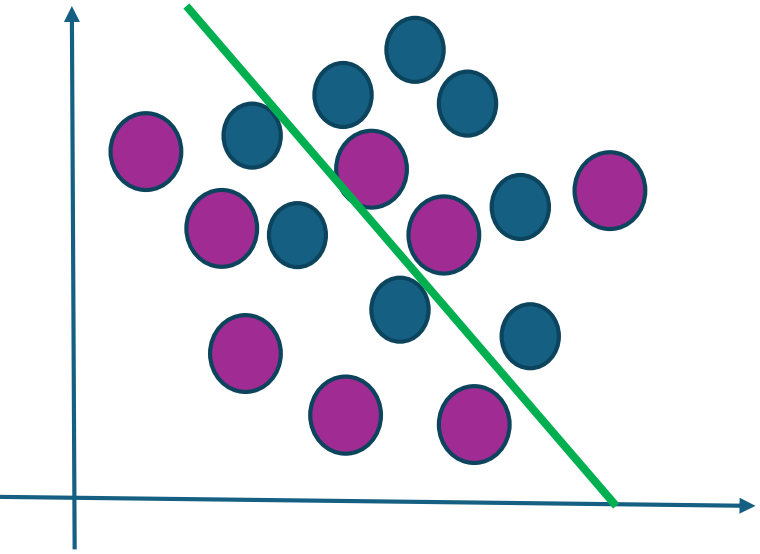
Balanced



Overfit High Variance

- **High Bias** – Underfit (Training error is high). Model is too simple to capture the patterns in the data
- **High Variance –** Overfit (Model fails on test/unseen data)

Why

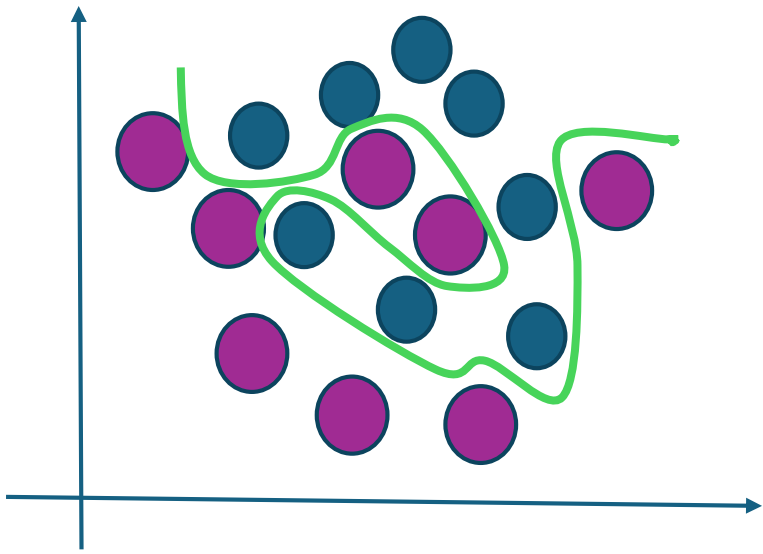


Underfit

$f(x) = g(\beta_0 + \beta_1 x_1)$
$f(x) = g(\beta_0 + \beta_1 x_1 + \beta_2 x_2)$

Balanced

$f(x) = g(\beta_0 + \beta_1 x_1^2 + \beta_2 x_2^2 + \cdots)$



Overfit

$f(x) = g(\beta_0 + \beta_1 x_1 + \beta_2 x_2^2 + \beta_3 x_1^2 x_2^2 + \beta_3 x_1^2 x_2^3 \ldots)$

When

- **Overfitting**: High training accuracy but low validation accuracy.
- **High Model Complexity**: Learn complex patterns, including noise in the training data.
- **Small or Noisy Dataset**: If the dataset is small or contains a significant amount of noise, the model might overfit, learning patterns that do not generalize well to new data.
  - **Insufficient Variety**: Lack the diversity, full range of possible inputs
  - **Poor Generalization**: Pick up on irrelevant features.
- **Multicollinearity**: When features are highly correlated.
- **Complex Models**: When using models with many parameters or features.

How

- **Penalty Term:** Added to the loss function to discourage large coefficients.
- **Loss Function:** Regularization = Loss Function + Penalty
- **Objective:** Minimize the loss function while keeping the model simple.

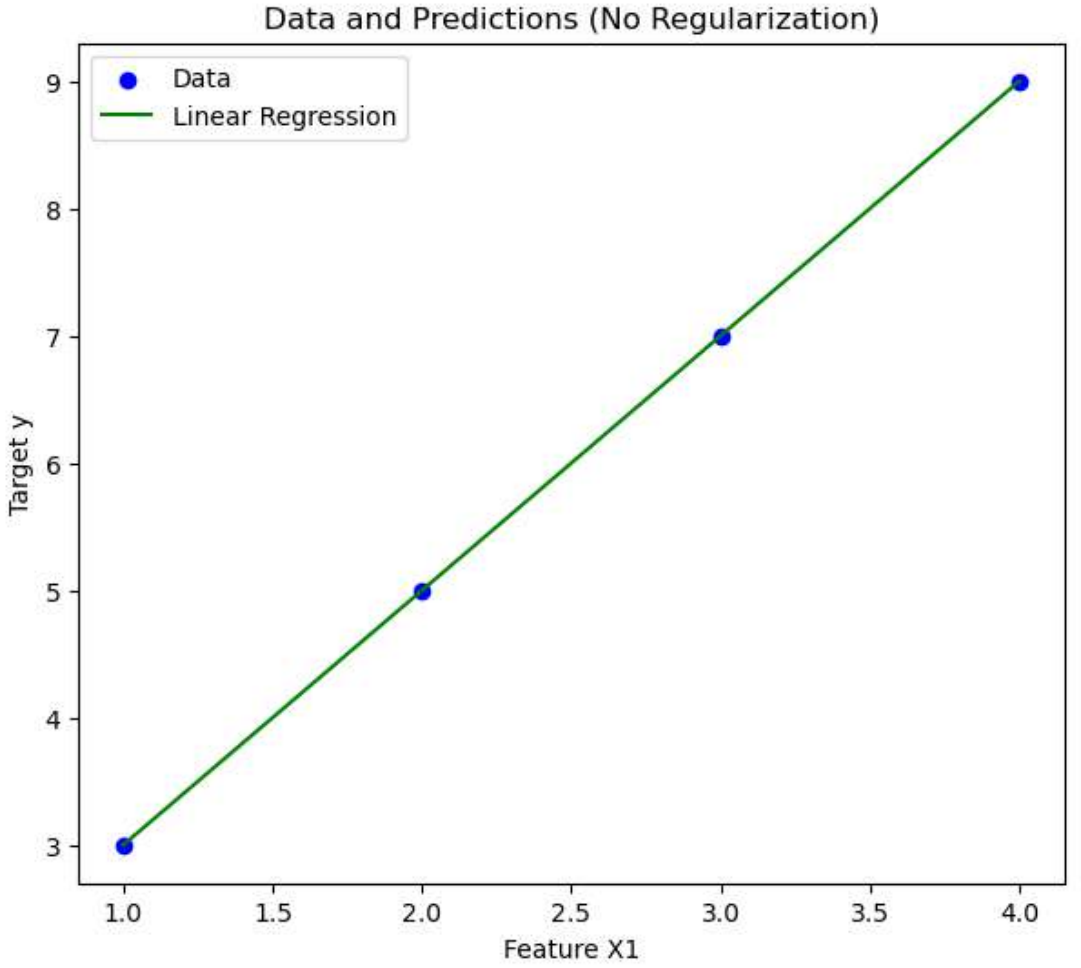
Without Regularization

(x_1)	(x_2)	(y)
1	2	3
2	3	5
3	4	7
4	5	9

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

$$Loss\ Function = \sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_1 + \beta_2 x_2)\right)^2$$

$$\hat{y} = 1 + x_1 + x_2$$



Types of Regularization

- L1 Regularization (Lasso)
- L2 Regularization (Ridge)
- Dropout
- Early Stopping
- Data Augmentation

L1 Regularization (Lasso)

(x_1)	(x_2)	(y)
1	2	3
2	3	5
3	4	7
4	5	9

- Adds the absolute value of the coefficients as a penalty term to the loss function

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

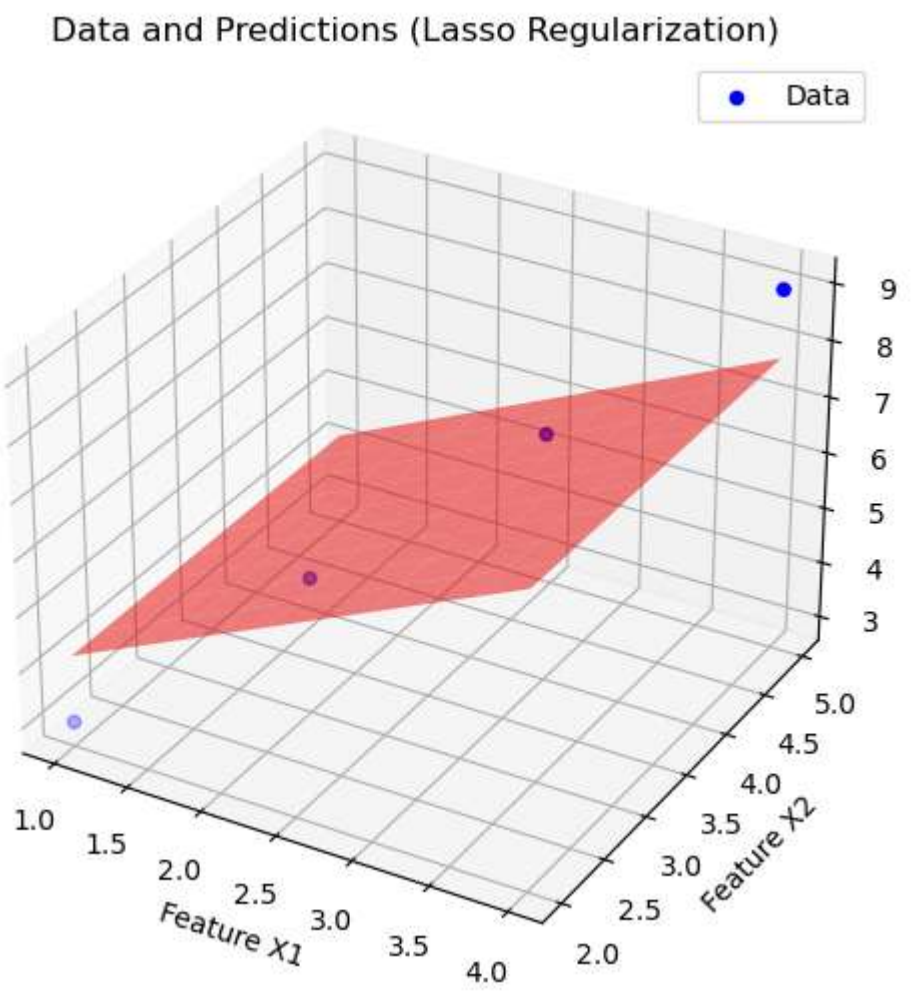
$$Loss\ Function = Loss\ Function + \lambda \sum |\beta_i|$$

If λ is large, it might set one of the weights to zero, effectively removing that feature from the model

$$Loss\ Function = \sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_1 + \beta_2 x_2)\right)^2 + \lambda(|\beta_1| + |\beta_2|)$$

$$\hat{y} = 1 + 0.x_1 + x_2$$

- Here x_1 is removed from the model
- It can be used as a feature selection



L2 Regularization (Ridge)

(x_1)	(x_2)	(y)
1	2	3
2	3	5
3	4	7
4	5	9

- Adds the squared value of the coefficients as a penalty term to the loss function.
- It tends to shrink the coefficients but does not set any of them to zero

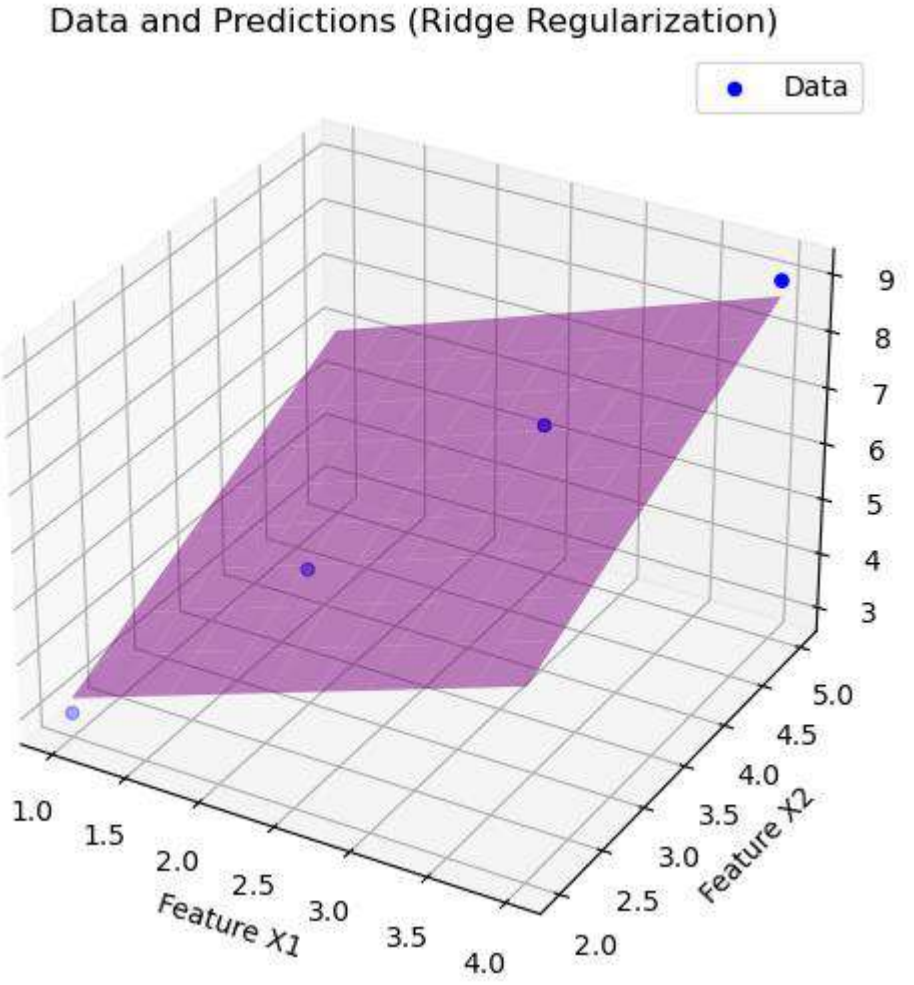
$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

$$Loss\ Function = Loss\ Function + \lambda \sum {\beta_i}^2$$

$$Loss\ Function = \sum_{i=1}^{n} \left(y_i - (\beta_0 + \beta_1 x_1 + \beta_2 x_2)\right)^2 + \lambda({\beta_1}^2 + {\beta_2}^2)$$

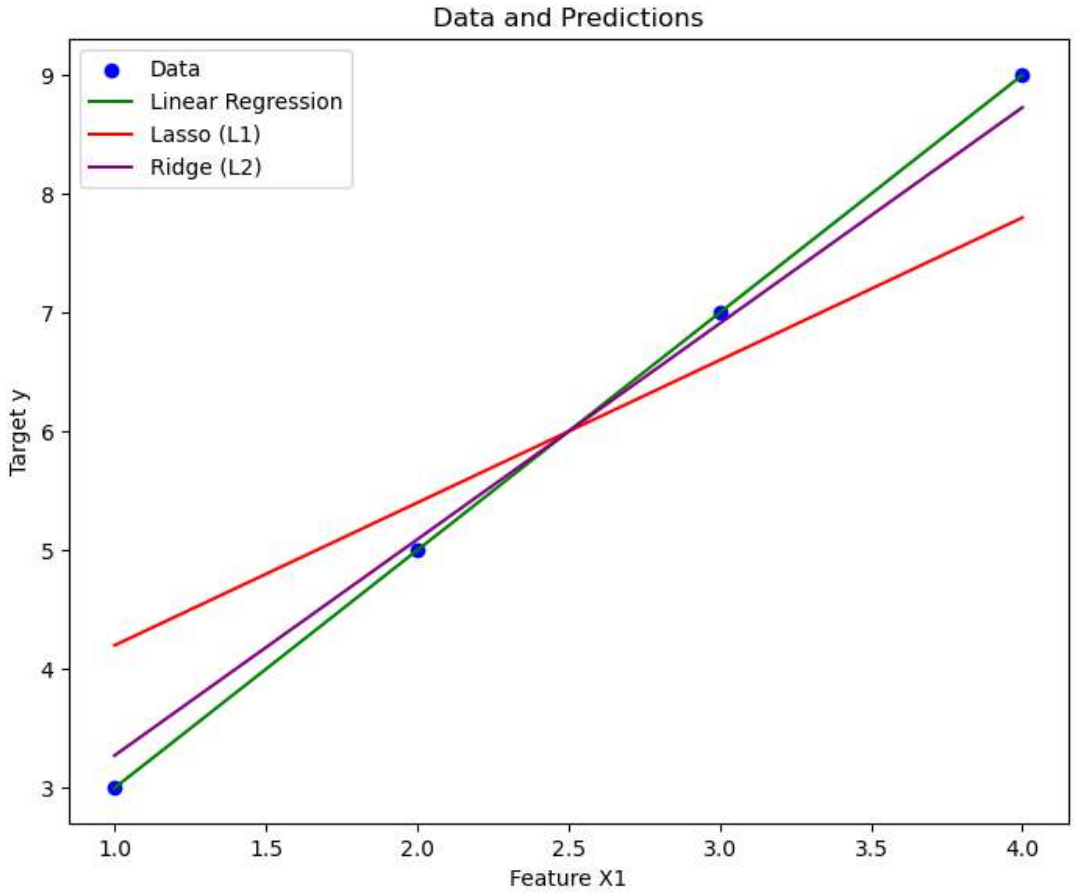
$$\hat{y} = 1 + 08.\, x_1 + 0.8 x_2$$

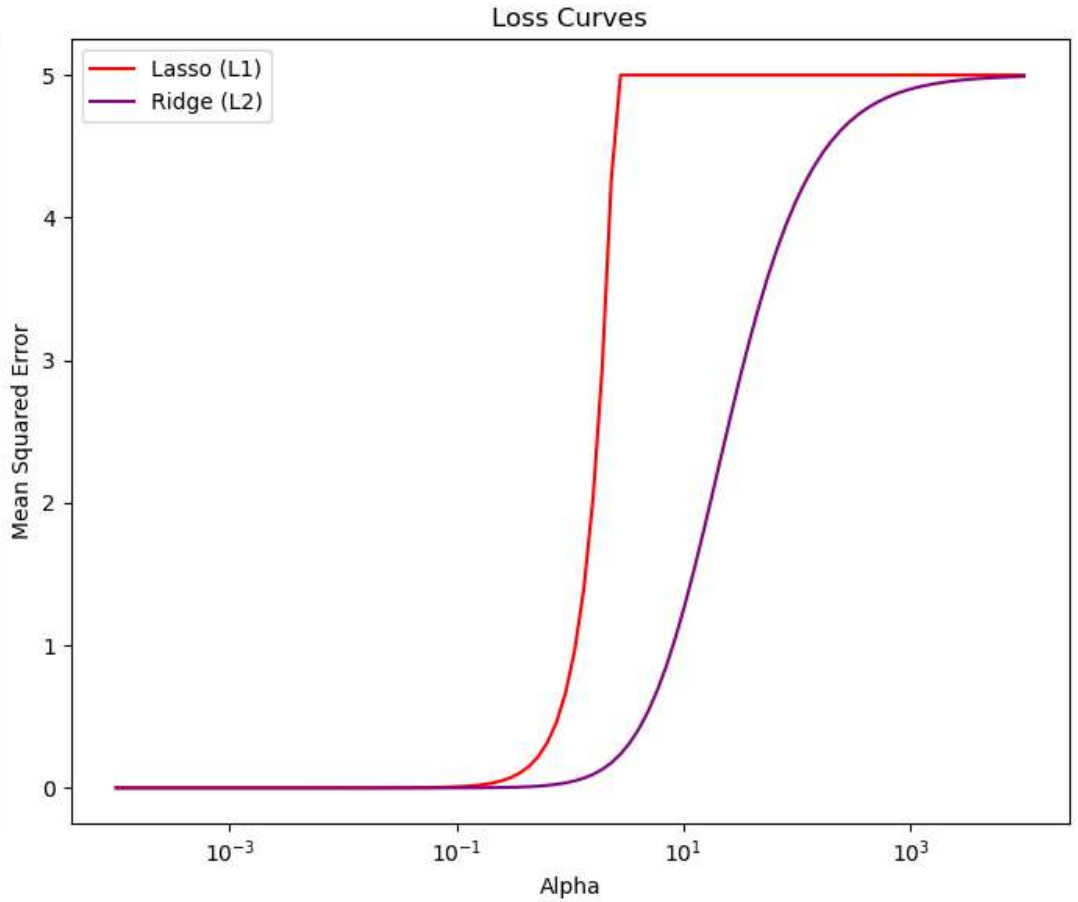
The weights are shrunk but not set to zero



Comparison

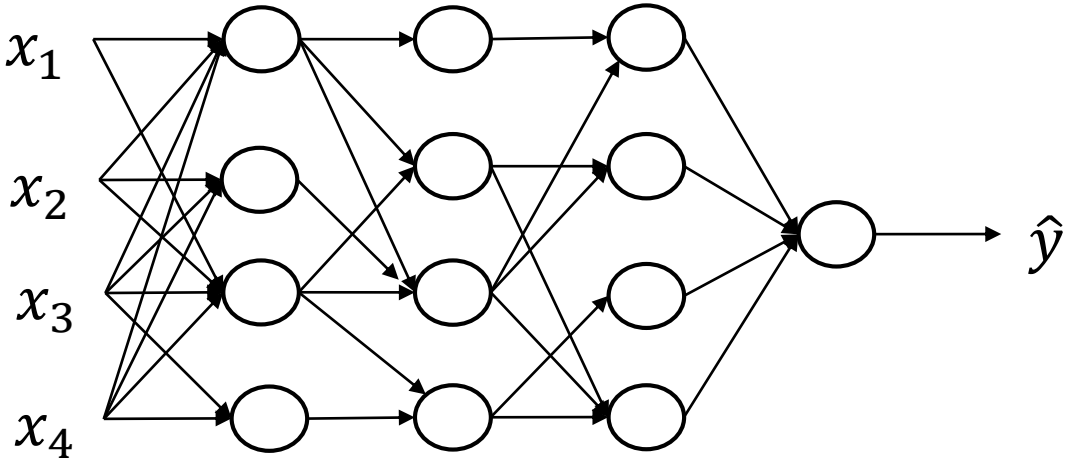
(x_1)	(x_2)	(y)
1	2	3
2	3	5
3	4	7
4	5	9

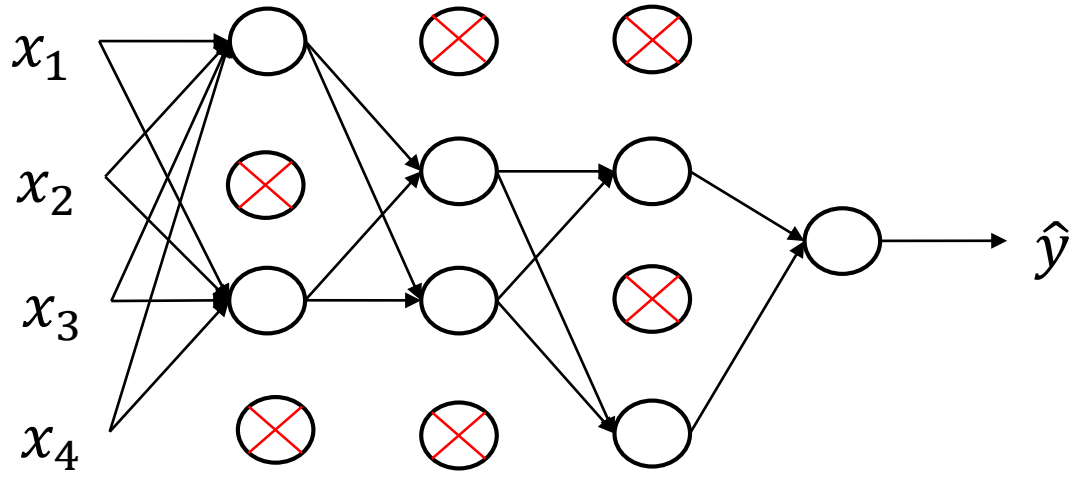




Dropout

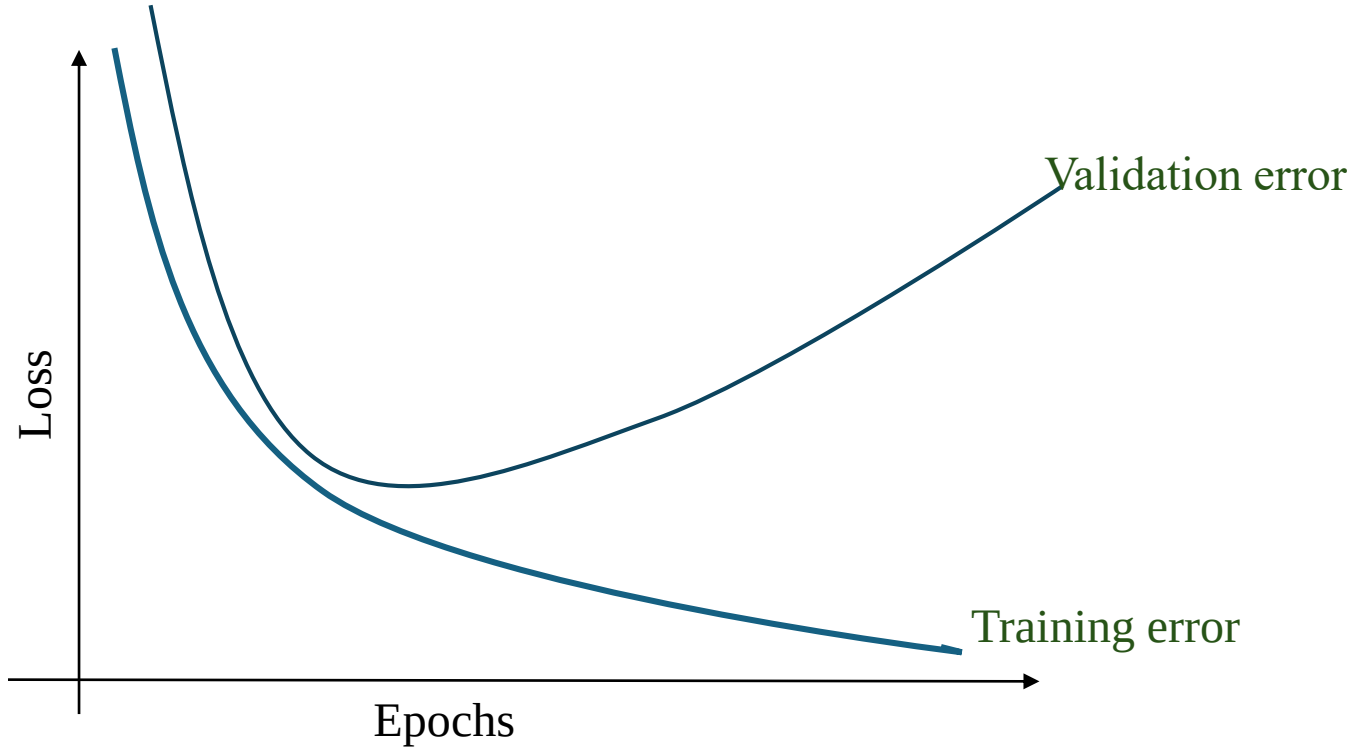
- Randomly “dropping out” (i.e., setting to zero) a fraction of the neurons during training
- Forces the network to learn more robust features that are not reliant on any single neuron.
- Dropout can be different for different layers
- Helps in shrinking weights
- Finding the optimal dropout rate can be time-consuming and may require extensive experimentation.
- Finding the optimal dropout rate can be time-consuming and may require extensive experimentation.
- Training may be slow as it needs more epochs to converge
- Model may get **underfitted** if dropout rate is high





Early stopping

- Monitors the model's performance on a validation set during training
- Stops the training process when the performance starts to degrade/overfit
- Training is stopped when the validation loss stops decreasing and starts to increase, indicating overfitting.



Data Augmentation

- Improves generalization ability of the model
- Provides variations in the data to the model during the training phase and prevents overfitting
  - **Geometric Transformations**: Rotation, translation, scaling, flipping, and cropping.
  - **Color Transformations**: Adjusting brightness, contrast, saturation, and hue.
  - **Noise Addition**: Adding random noise to the images.
  - **Other Techniques**: Random erasing, cutout, mixup, cut mix, etx.

Exponentially weighted averages

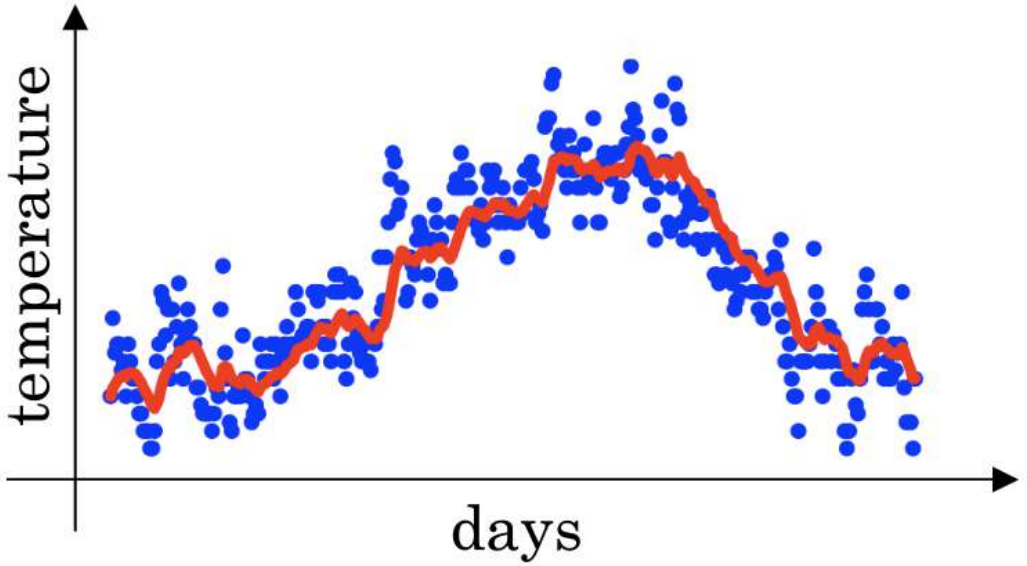
Temperature for a one year period in Delhi

$V_{100}= 0.9V_{99}+0.1\theta_{100}$
$V_{99}= 0.9V_{98}+0.1\theta_{99}$
$V_{98}= 0.9V_{97}+0.1\theta_{98}$
$V_t= \beta V_{t-1}+(1-\beta)\theta_t$

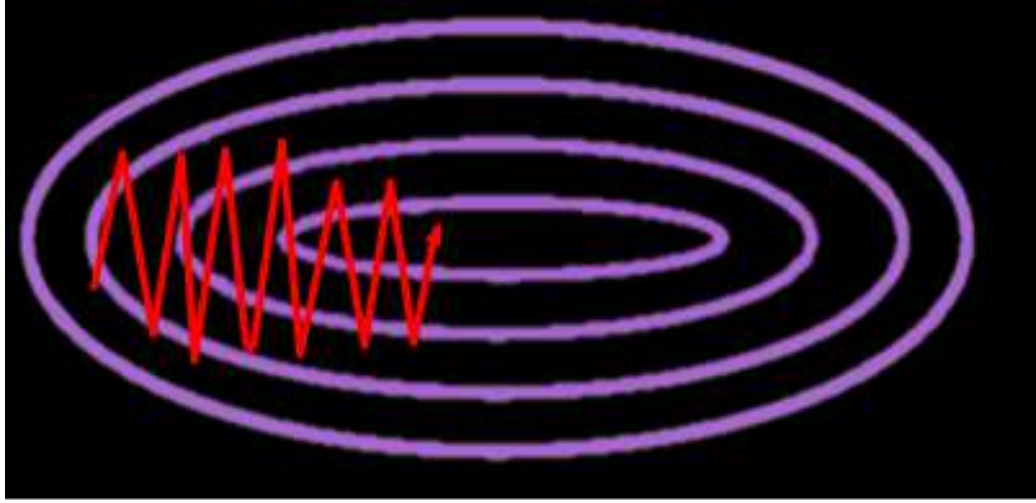
$\beta=0.9$

V_t is average over $\frac{1}{1-\beta}$ days eg $\frac{1}{1-0.9}$ = 10 days

Cold start-up in case the initial values are low



Momentum



Compute $\partial\beta_i$ on current mini-batch

$V_{\partial\beta_i} = \gamma V_{\partial\beta_i} + (1-\gamma)\partial\beta_i$
$\beta_i = \beta_i - \alpha V_{\partial\beta_i}$

$\gamma$is new hyperparameter and the recommended value is 0.9

- It helps accelerate gradient vectors in the right directions, that helps in faster convergence
- It will moderate the movements in the vertical direction and will help us move faster in the horizontal direction
- Instead of calculating gradients independently we use exponentially weighted averages to calculate the gradient

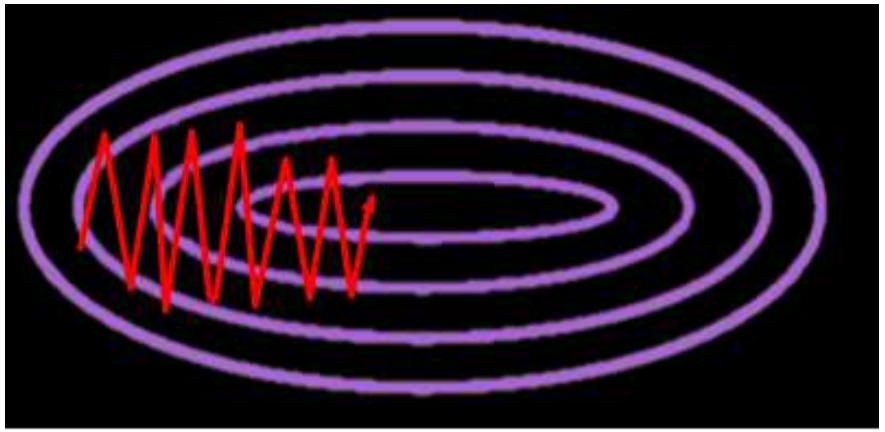
RMSProp (Root Mean Square Prop)

Compute $\partial\beta_i$ on current mini-batch

$$S_{\partial\beta_i} = \gamma S_{\partial\beta_i} + (1-\gamma)\partial\beta_i{}^2$$

$$\beta_i = \beta_i - \alpha \frac{\partial\beta_i}{\sqrt{S_{\partial\beta_i}} + \epsilon}$$

$$\epsilon = 10^{-8}$$



Adam optimization Algorithm

Compute $\partial\beta_i$ on current mini-batch

$$S_{\partial\beta_i} = \gamma_1 S_{\partial\beta_i} + (1-\gamma_1)\partial\beta_i{}^2$$

$$V_{\partial\beta_i} = \gamma_2 V_{\partial\beta_i} + (1-\gamma_2)\partial\beta_i$$

$$\beta_i = \beta_i - \alpha \frac{V_{\partial\beta_i}}{\sqrt{S_{\partial\beta_i}} + \epsilon}$$

γ_1=0.9 $(\partial\beta_i)$ also referred to as moment 1

γ_2=0.999 $(\partial\beta_i{}^2)$ also referred to as moment 2

$\epsilon = 10^{-8}$

Adaptive Moment estimation

Optimizers

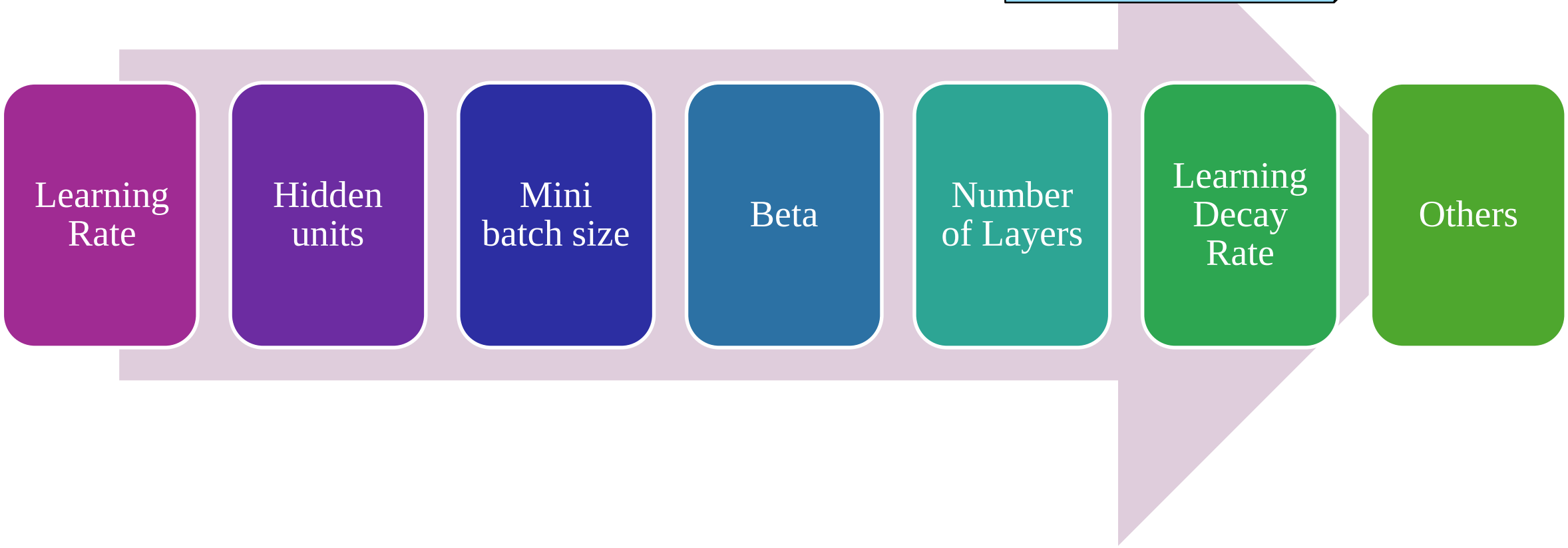
Batch GD	$\beta_i = \beta_i - \alpha \cdot \nabla J(\theta)$	• Guarantees convergence to the global minimum. • Stable and smooth updates.	• Slow for large datasets. • Requires loading the entire dataset into memory.
Stochastic GD	$\beta_i = \beta_i - \alpha \cdot \nabla J(\theta, x^i y^i)$	• Faster iteration speed. • Can escape local minima (due to noisy updates). • Suitable for online learning.	• Convergence can be noisy and less stable. • Requires careful tuning of the learning rate.
Mini – Batch GD	$\beta_i = \beta_i - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla J(\theta, x^i y^i)$	• Balance between Batch GD and SGD: faster than Batch GD and more stable than SGD. • Efficient computation using vectorization.	• Still needs tuning of learning rate and batch size. • Less stable than full Batch GD.
Momentum	$v_t = \gamma v_{t-1} - \alpha \cdot \nabla J(\theta)$ $\beta_i = \beta_i - v_t$	• Accelerates convergence for high-curvature, small, and noisy gradients. • Reduces oscillation.	• Requires additional hyperparameter (momentum coefficient γ). • Can overshoot minima.

Optimizers

RMS Prop	$S_{\partial\beta_i} = \gamma S_{\partial\beta_i} + (1-\gamma)\partial\beta_i^2$ $\beta_i = \beta_i - \alpha \frac{\partial\beta_i}{\sqrt{S_{\partial\beta_i}} + \in}$	- Adapts learning rate for each parameter - Prevents learning rate from becoming too small	- Requires careful tuning of hyperparameters - Can be sensitive to the choice of learning rate
Adam	$S_{\partial\beta_i} = \gamma_1 S_{\partial\beta_i} + (1-\gamma_1)\partial\beta_i^2$ $V_{\partial\beta_i} = \gamma_2 V_{\partial\beta_i} + (1-\gamma_2)\partial\beta_i$ $\beta_i = \beta_i - \alpha \frac{V_{\partial\beta_i}}{\sqrt{S_{\partial\beta_i}} + \in}$	- Combines the benefits of RMSProp and Momentum - Adapts learning rates for each parameter - Works well with sparse gradients	- Computationally more expensive due to additional calculations - Requires tuning of multiple hyperparameters

How to try Hyperparameters

First focus on Most important ones and then the lesser ones in the sequence



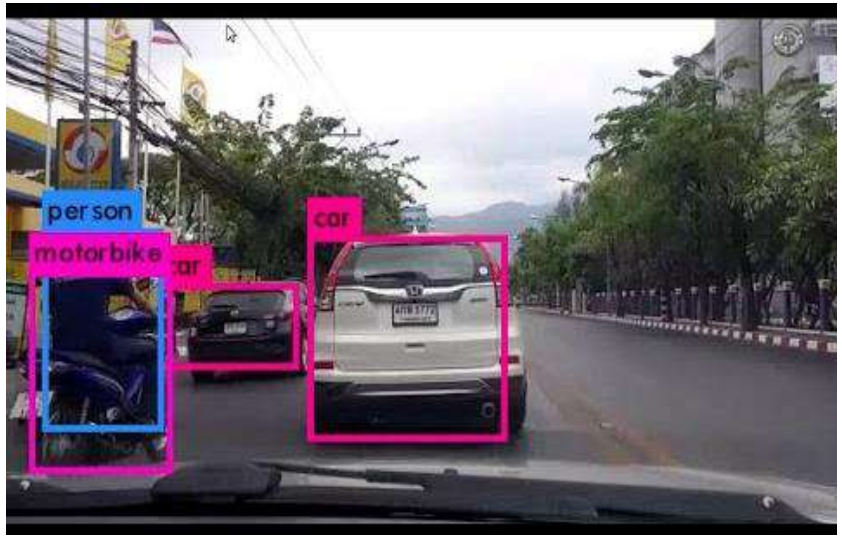
Thank You

Convolutional Neural Networks

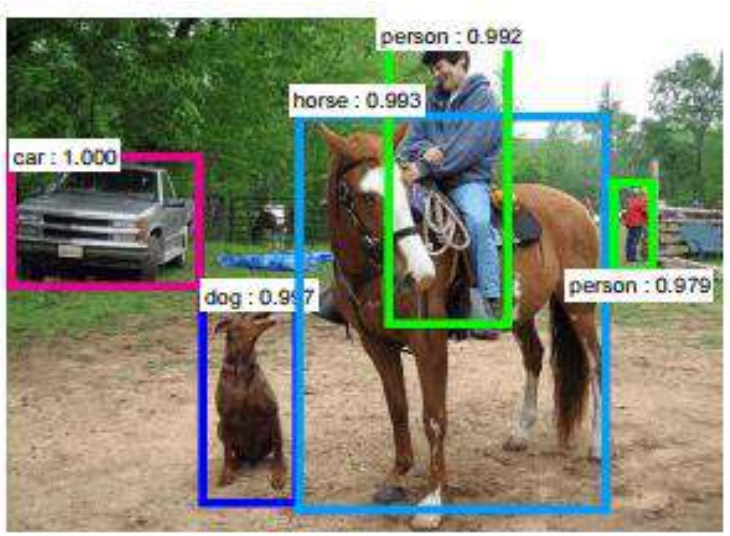
Face Recognition

Style Transferring

Image quality enhancement
Beautification



Object detection (Self driving car)



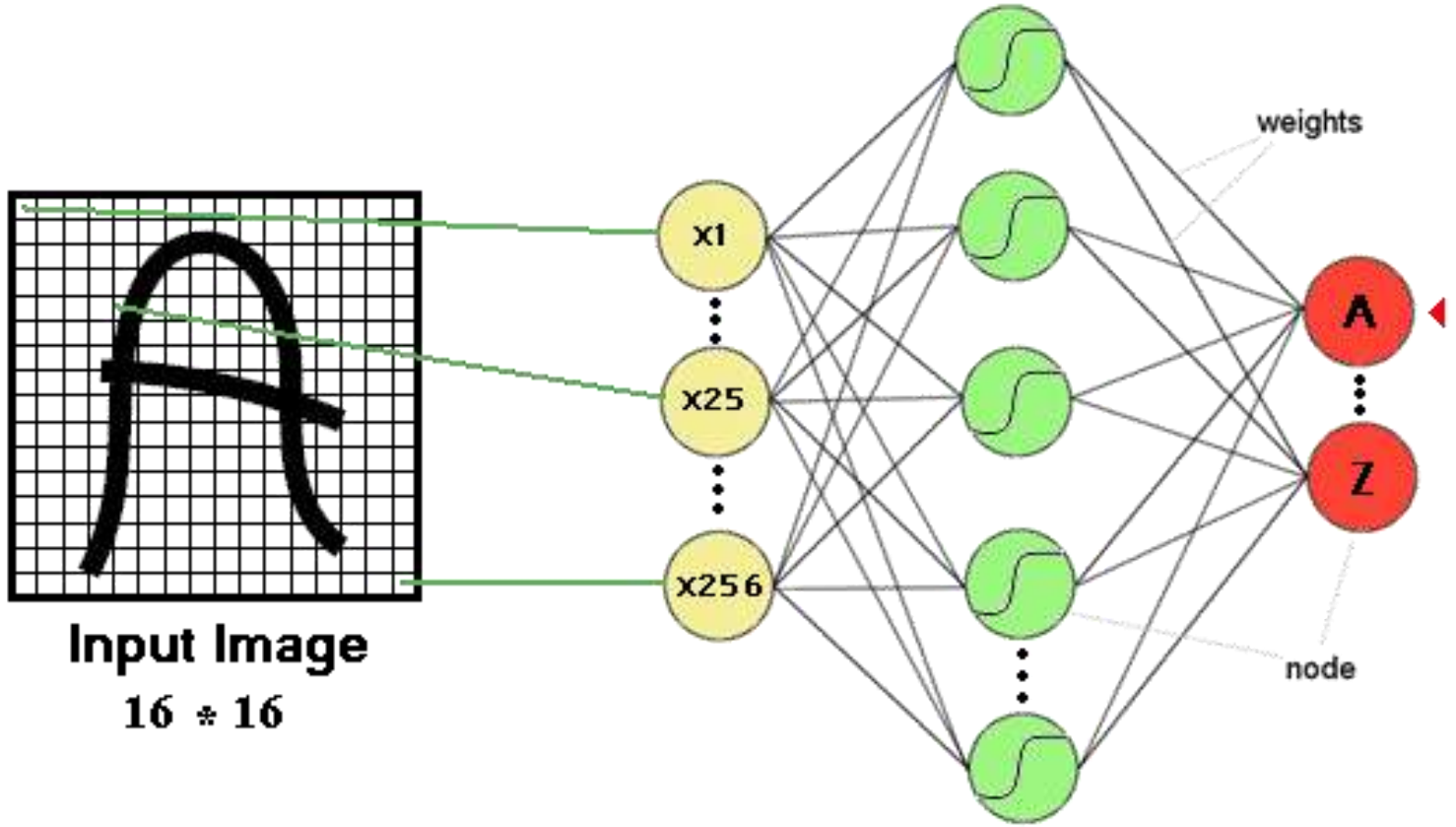
Classification

Gesture Recognition

Motivation

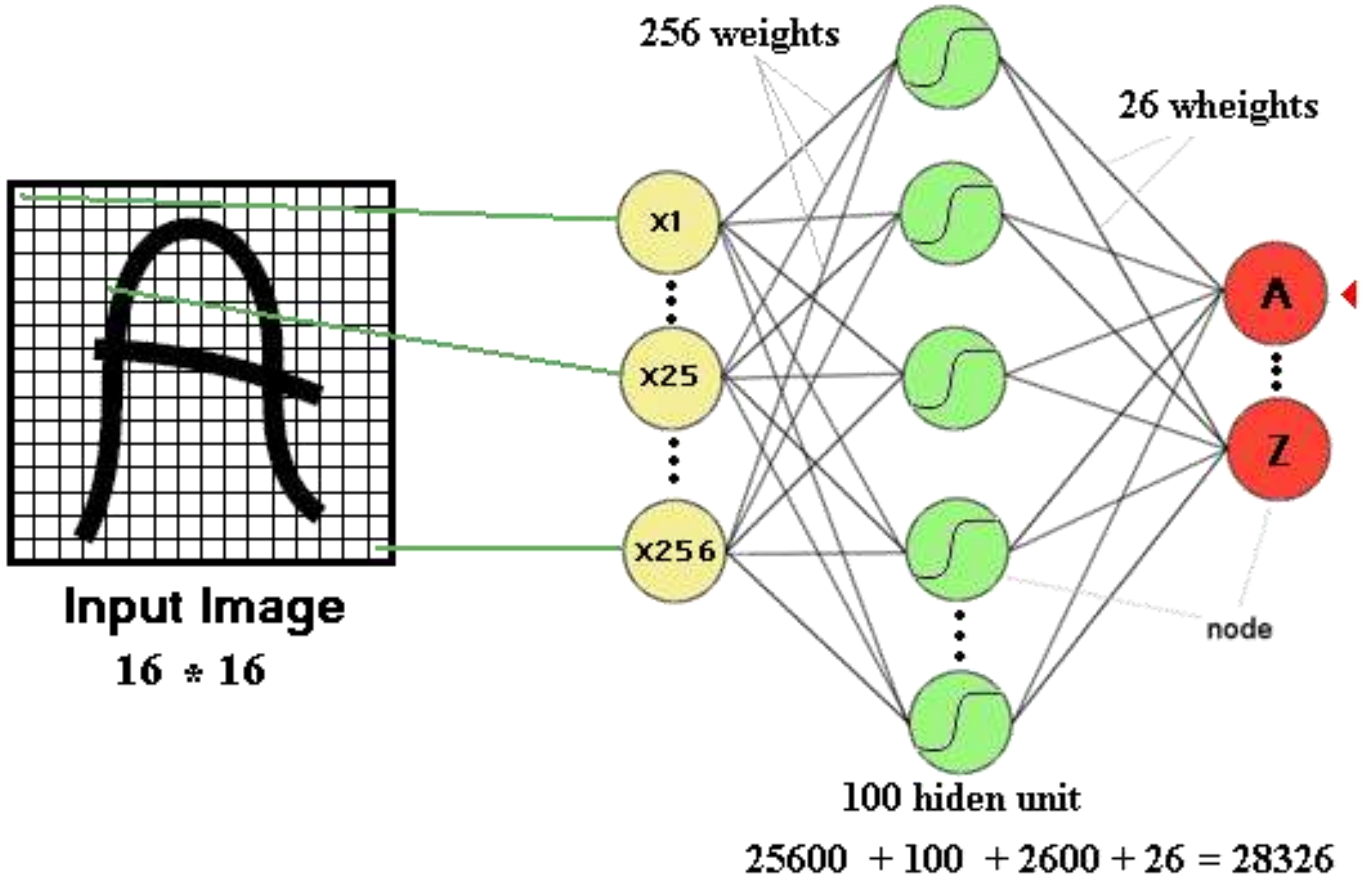
WHY WE NEED CONVOLUTIONAL NEURAL NETWROKS?

Multi-layer perceptron and image processing



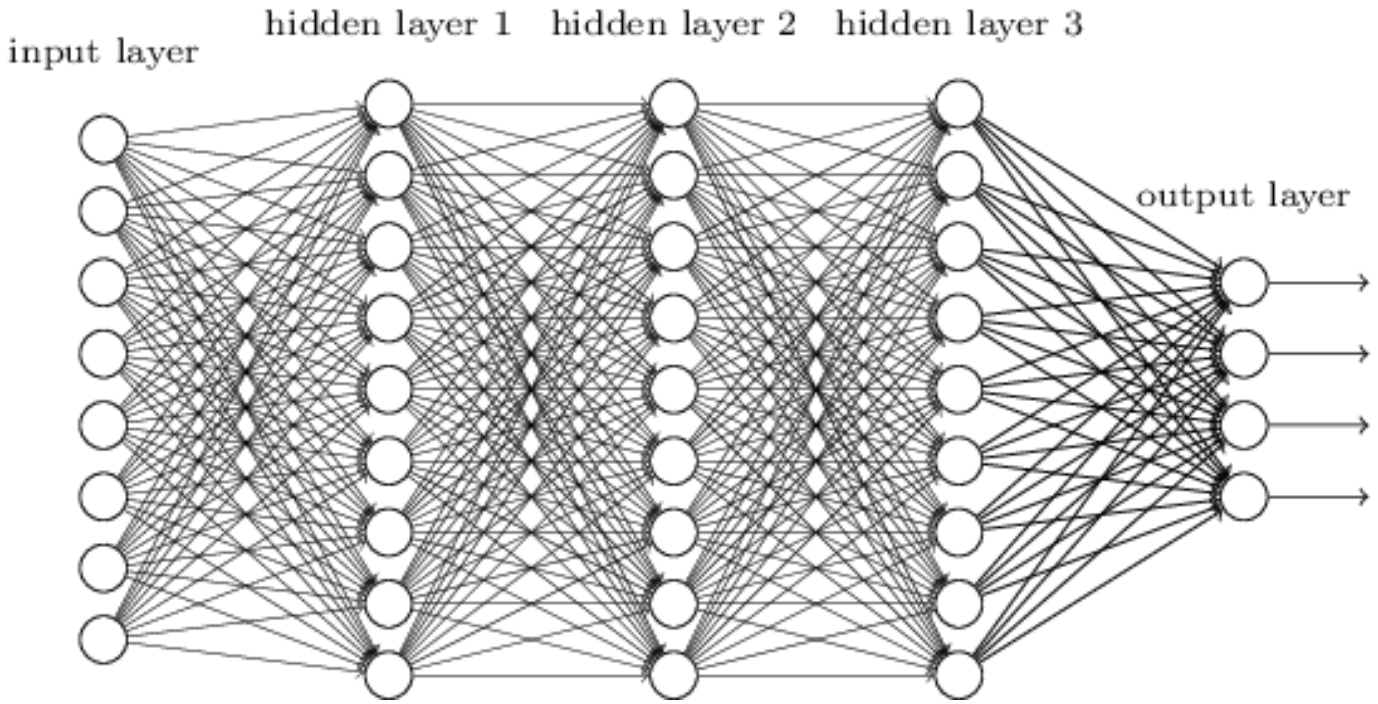
Even a simple 16x16 image

- **256x100 + 100 bias + 100x26 output neurons + 26 bias = 28236**
- the number of trainable parameters becomes extremely large



ANN: Too many parameters

- **We know it is good to learn a small model.**
- **From this fully connected model, do really need all the edges?**
- **Can some of these be shared?**



Identify

Can we do with less information?

- **How much information we can throw away and still recognize the object?**

10%

20%

Can we do with less information?

- **How much information we can throw away and still recognize the object?**

10%

20%

50%

75%

- **Subsampling pixels will not change the object**

bird

Subsampling

bird

We can subsample the pixels to make image smaller

fewer parameters to characterize the image


TIPO EXCELSIUS

CNNs Vs. ANNs

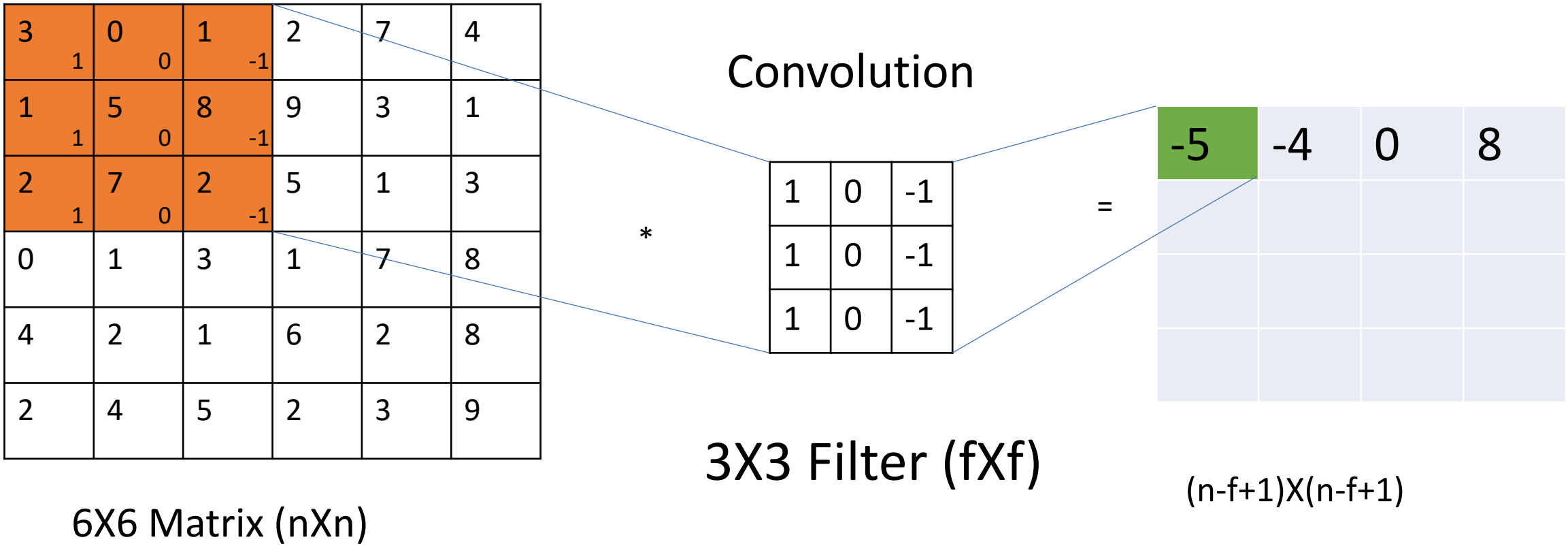
- ANNs suffer from curse of dimensionality when it comes to high resolution images
- We use filters (receptive fields) to exploit spatial locality by enforcing a local connectivity pattern between neurons of adjacent layers

- Parameter Sharing
- Sparsity of connection

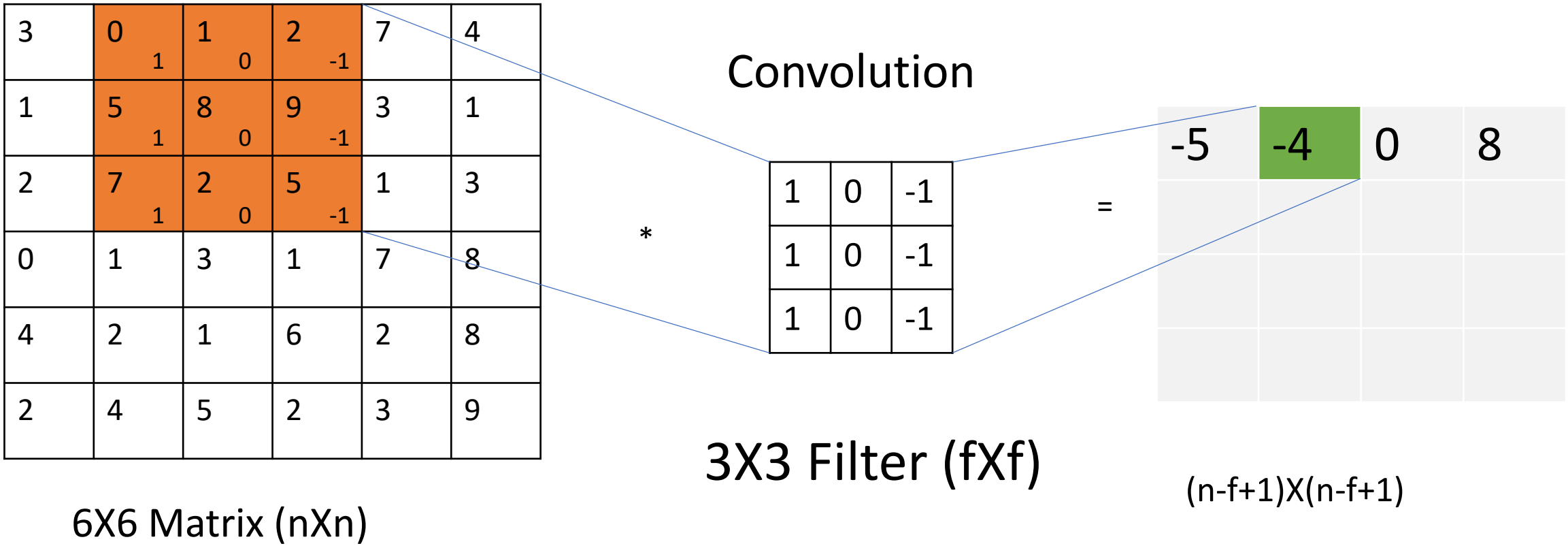
Convolution

- Convolution is a pointwise multiplication of two functions to produce a third function.
- Primary purpose of convolution in CNN is to extract features from the input image.
- Matrix formed by sliding the filter over the image and computing the dot product is called the ‘Convolved Feature’ or ‘Activation Map’ or the ‘Feature Map‘.

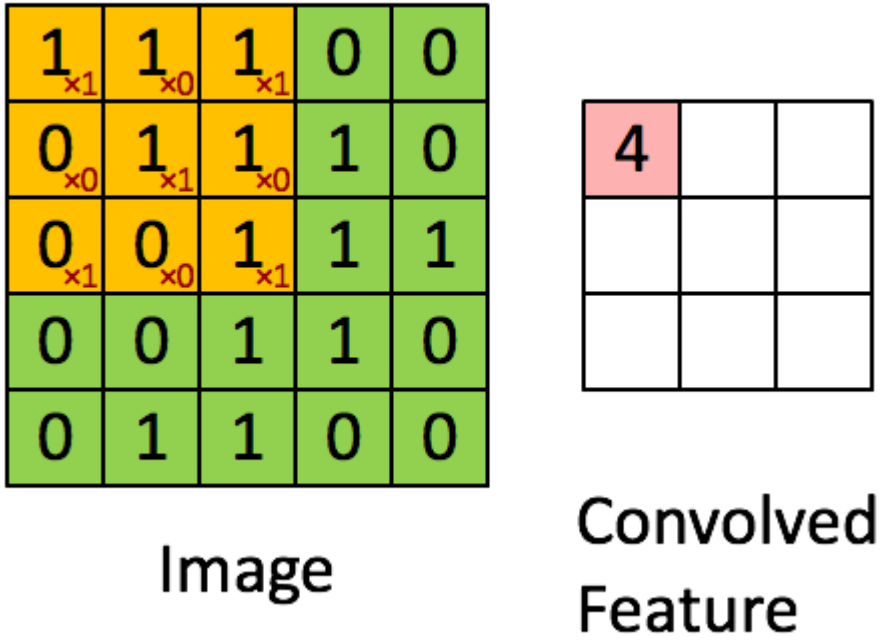
Convolution Example



Convolution Example



Convolution Example



*http://ufldl.stanford.edu/tutorial/supervised/FeatureExtractionUsingConvolution/

Detecting Vertical edges

6*6 = 36

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

1	0	-1
1	0	-1
1	0	-1

4*4=16

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

- In case of ANN # parameter to train = 36*16 = 576
- In case of CNN # parameter to train = 9

Filter Weights

1	1	1
0	0	0
-1	-1	-1

Horizontal Filter

1	0	-1
2	0	-2
1	0	-1

Sobel Filter

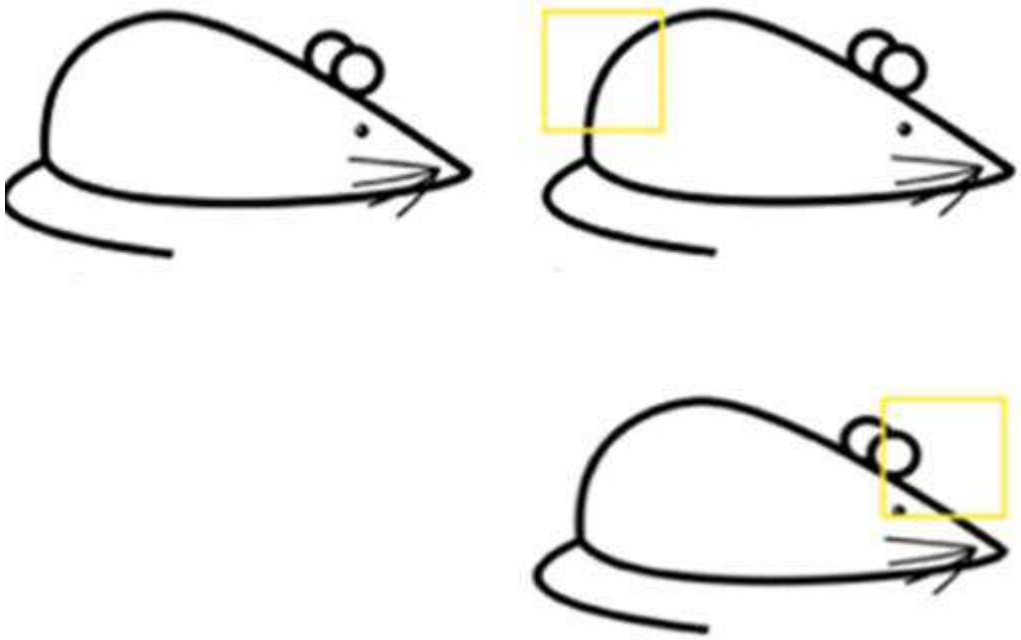
3	0	-3
10	0	-10
3	0	-3

Schorr Filter

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

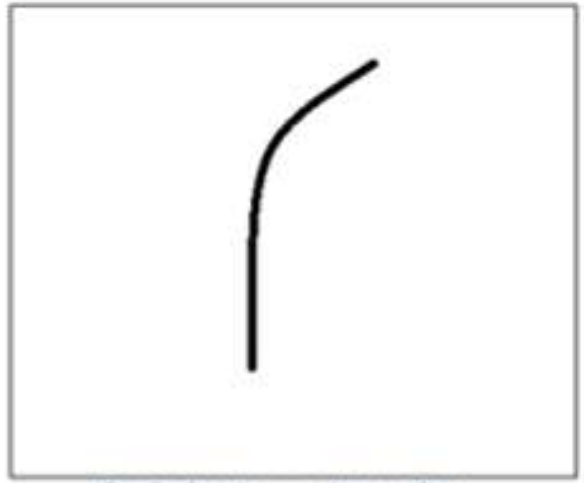
Convolutional Neural Networks automatically estimates the weights of the filter

More Intuition



0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

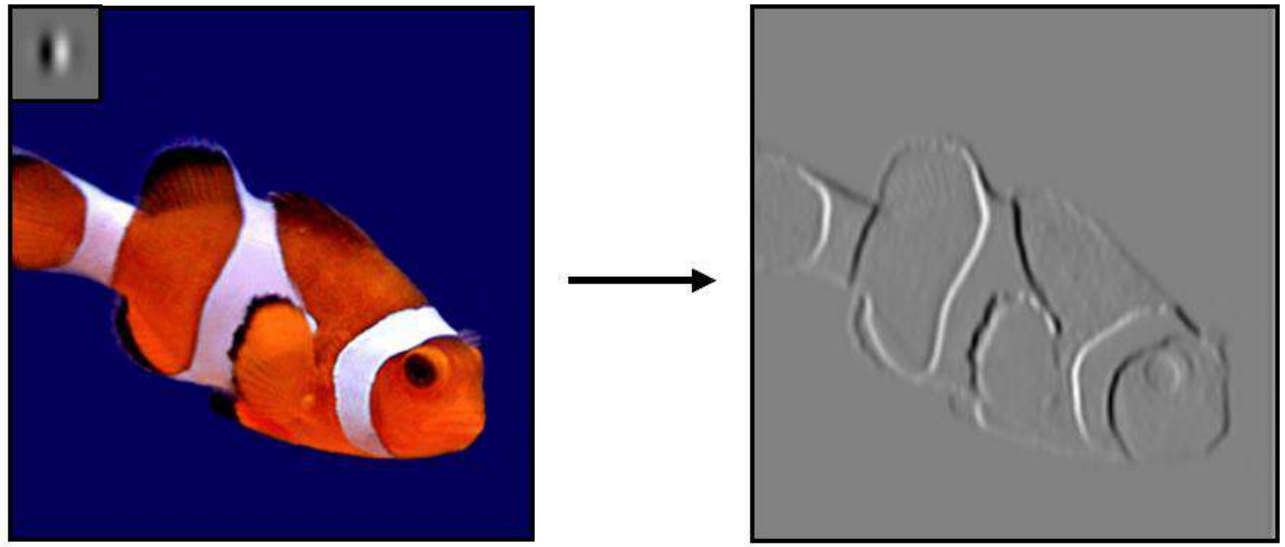


Visualization of a curve detector filter

Interpretation

Convolution is just another way of computing $W^T X$

In CNN, **input** is **image**, **kernel** is **convolution filter** to be learned, **response** is the **feature map**



Padding

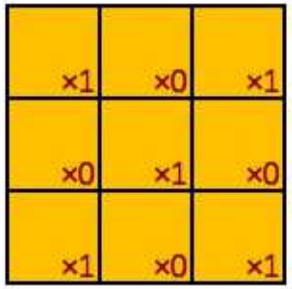
Padding is used to preserve the original dimensions of the input

Zeros are added to outside of the input

Number of zero layers depend upon the size of the kernel

0	0	0	0	0	0	0
0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	0	1	1	1	0
0	0	1	1	1	0	0
0	0	1	1	0	0	0
0	0	0	0	0	0	0

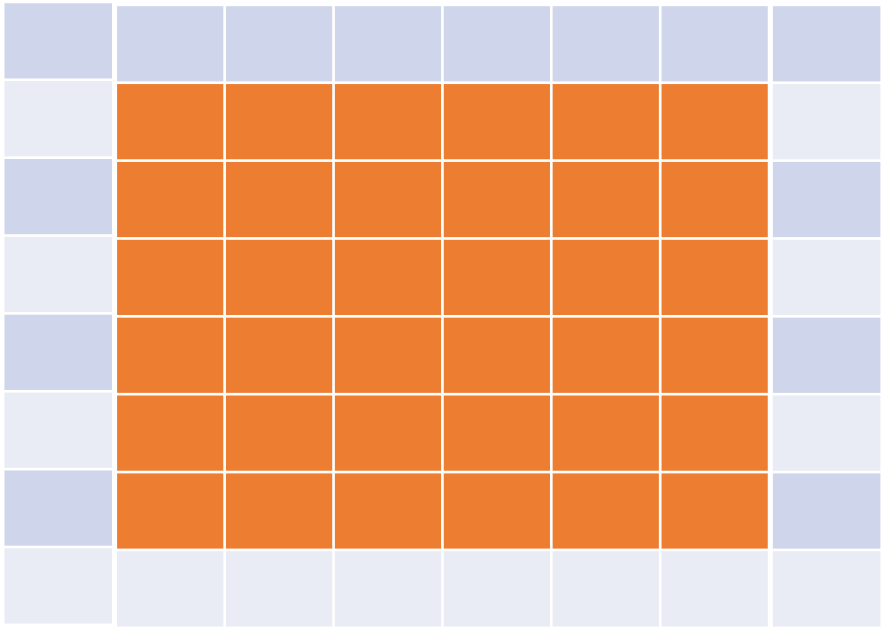
5X5 (with padding)



2	2	3	1	1
1	4	3	4	1
2	2	4	3	3
1	2	3	4	1
1	2	3	1	1

5X5

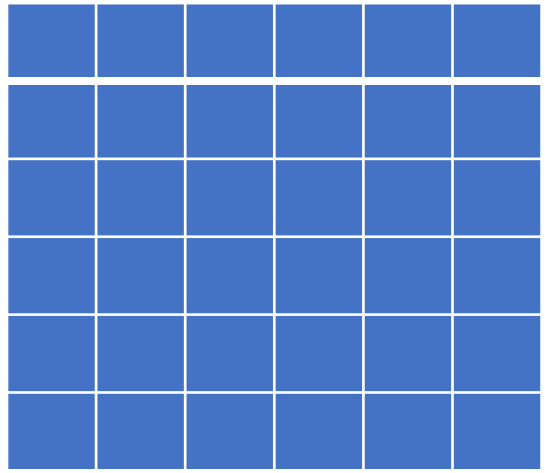
Padding



*

f 3X3

=



nXn 6X6 to 8X8 Padding=1

(n-f+1)X(n-f+1) to (n+2p-f+1)X(n+2p-f+1)

Valid to same

Stride=s

Floor($\frac{n+2p-f}{s}+1$) XFloor($\frac{n+2p-f}{s}+1$)

Stride

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6X6 Matrix

1	0	-1
1	0	-1
1	0	-1

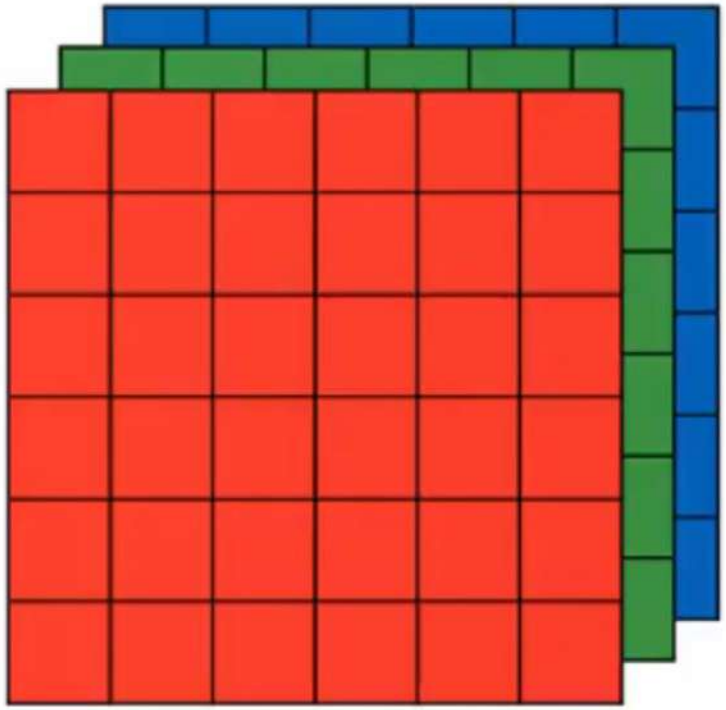
3X3 Filter

-5	8
-3	-16

2X2

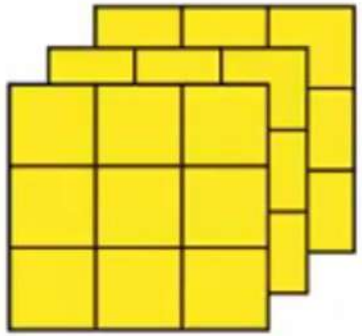
Stride=s (3 Here) Floor($\frac{n+2p-f}{s}+1$) XFloor($\frac{n+2p-f}{s}+1$)

Channels



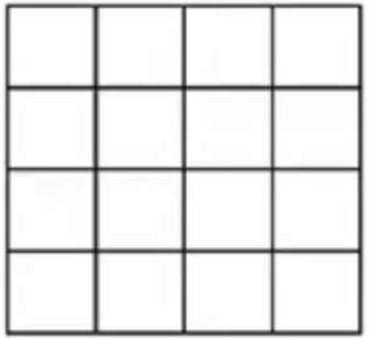
6X6 Matrix

*



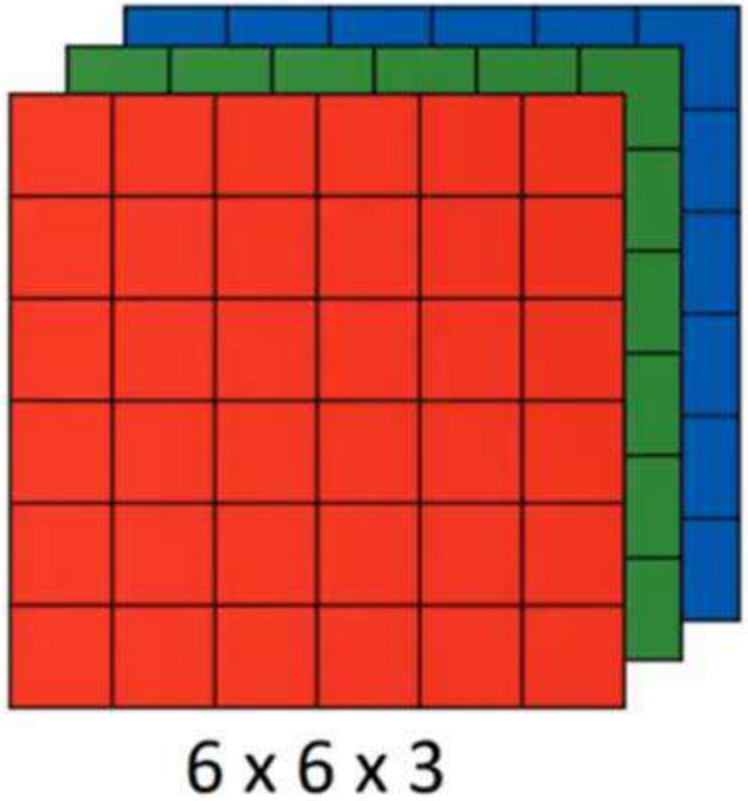
3X3 Filter

=

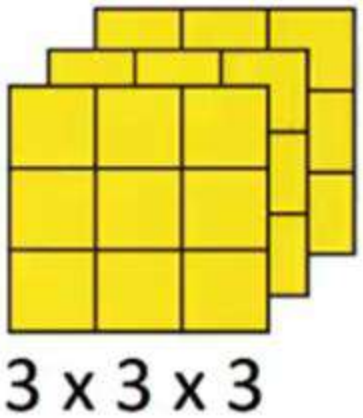


4 x 4

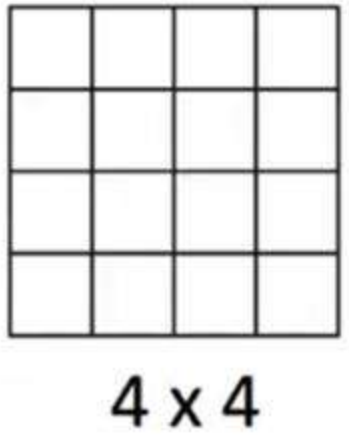
Padding =0 Stride=1



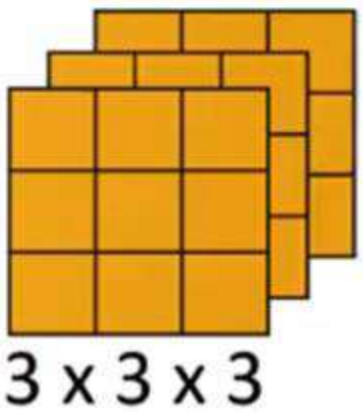
*



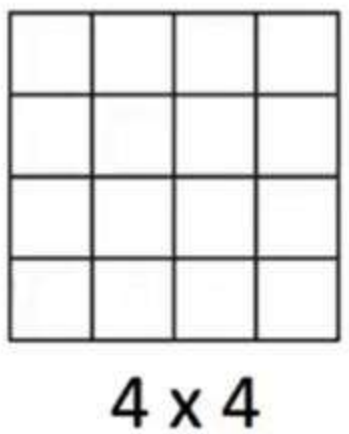
=



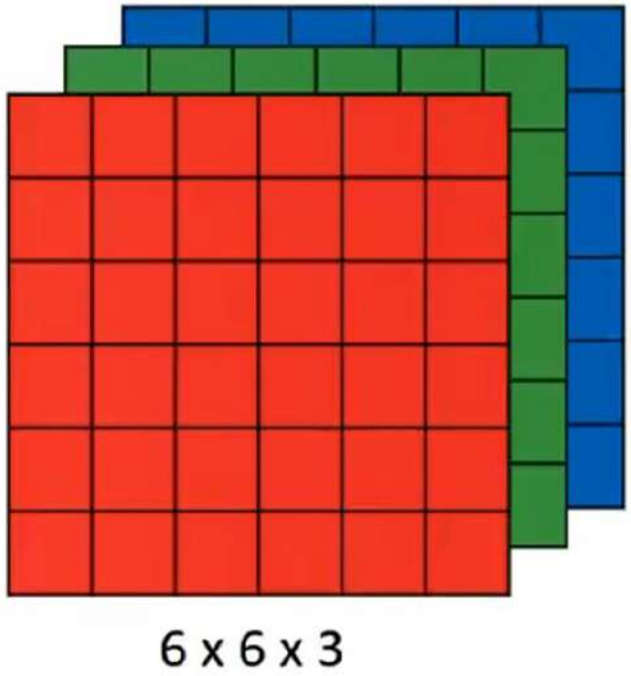
*



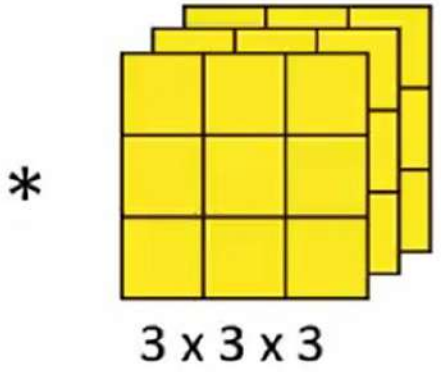
=

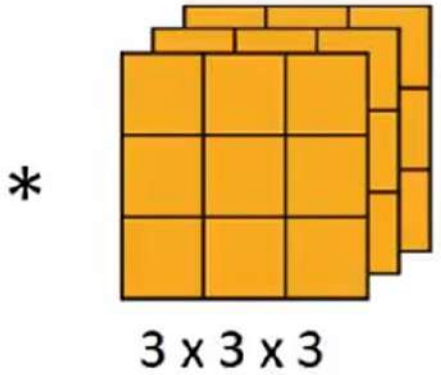


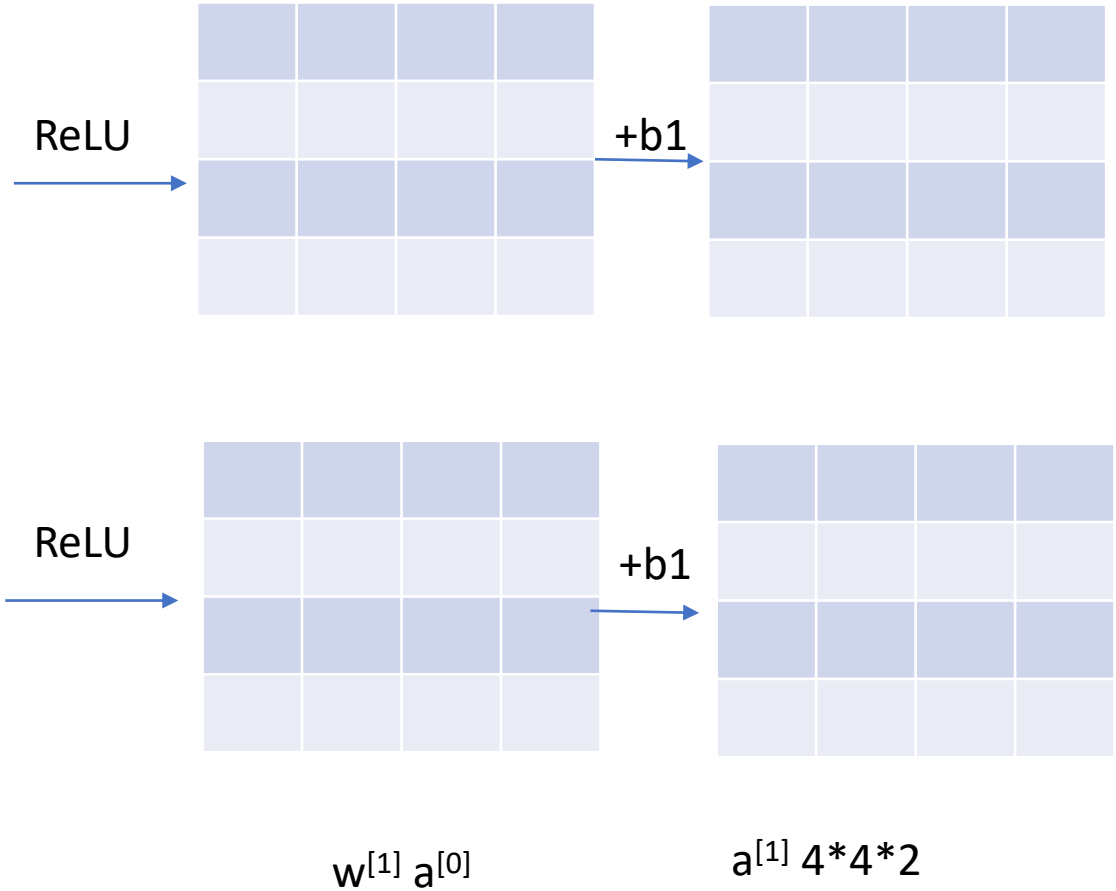
$nXnXn_c$ $fXfXn_c$ $n-f+1Xn-f+1Xn_{c'}$ c'=no of filters



$a^{[0]}$







$w^{[1]}$ $a^{[0]}$

$a^{[1]}$ 4*4*2

$w^{[1]}$ 2 filters means two units here

Pooling

1	4	6	3
1	8	9	7
2	9	1	2
3	4	4	3

Max Pooling : One example of pooling layer

8	9
9	4

f=2

s=2 4X4 converted to 2X2

Function of Pooling is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network.

Pooling

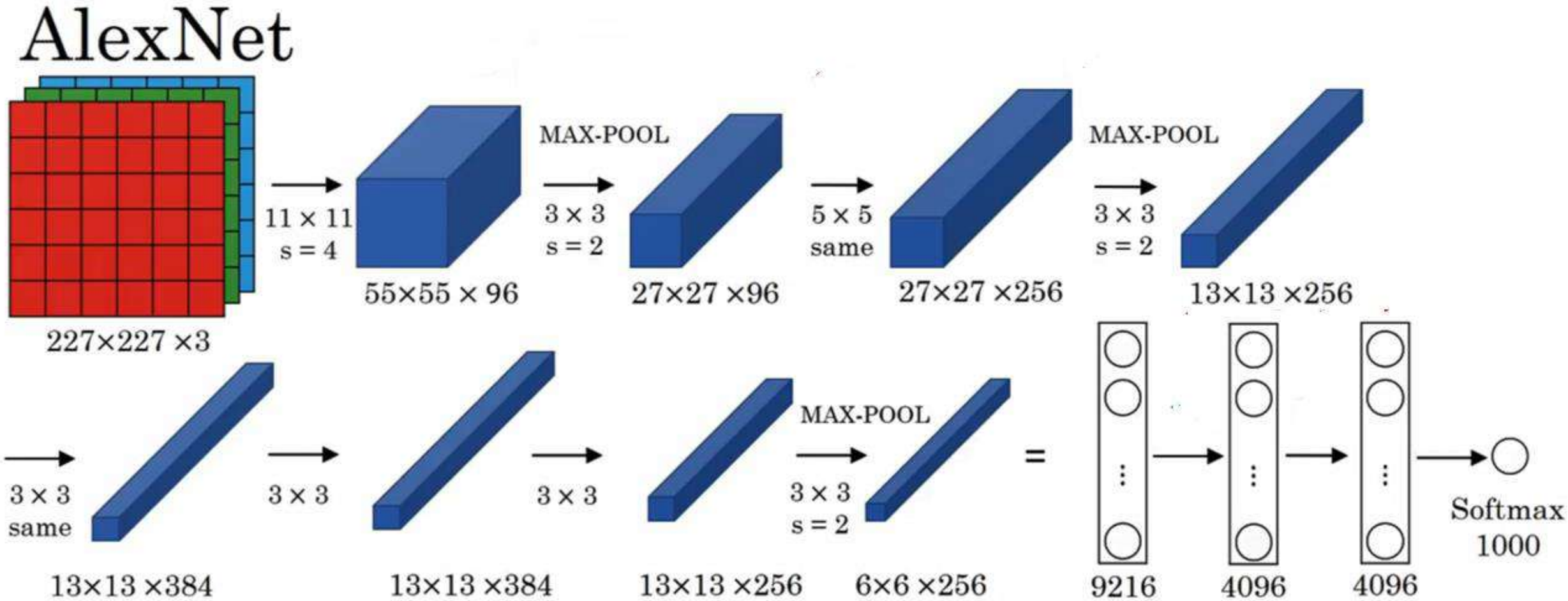
1	4	6	3
1	8	9	7
2	9	1	2
3	4	4	3

3.5	6.25
4.5	2.5

Average Pooling : Another example of pooling layer

f=2

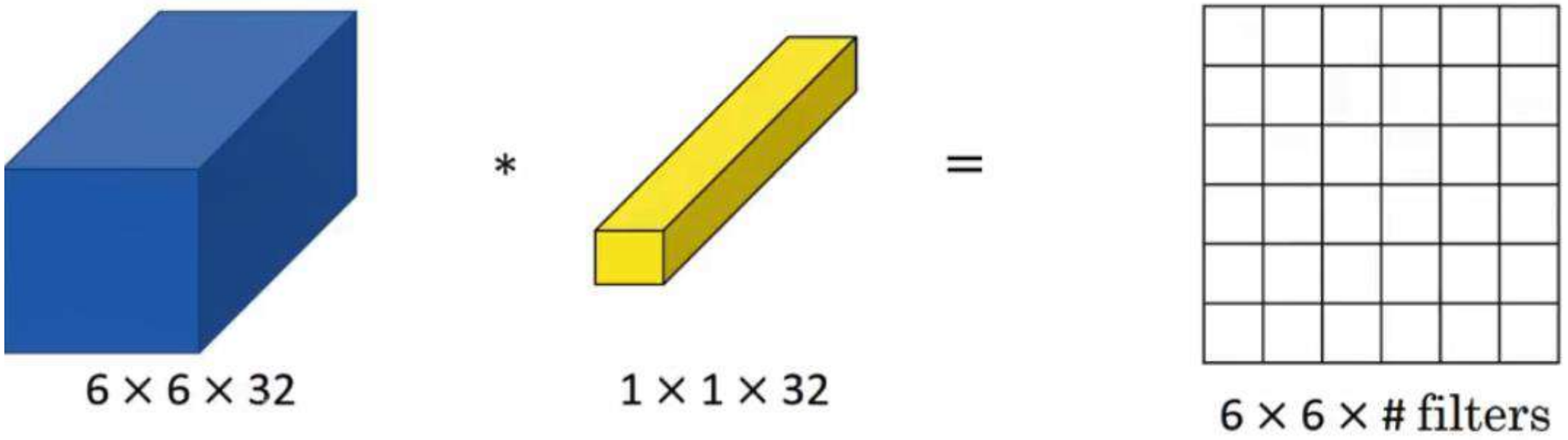
s=2 4X4 converted to 2X2


AlexNet
227×227 ×3
11 × 11
s = 4
55×55 × 96
MAX-POOL
3 × 3
s = 2
27×27 ×96
5 × 5
same
27×27 ×256
MAX-POOL
3 × 3
s = 2
13×13 ×256
3 × 3
same
13×13 ×384
3 × 3
13×13 ×384
3 × 3
13×13 ×256
MAX-POOL
3 × 3
s = 2
6×6 ×256
=
9216
4096
4096
Softmax
1000

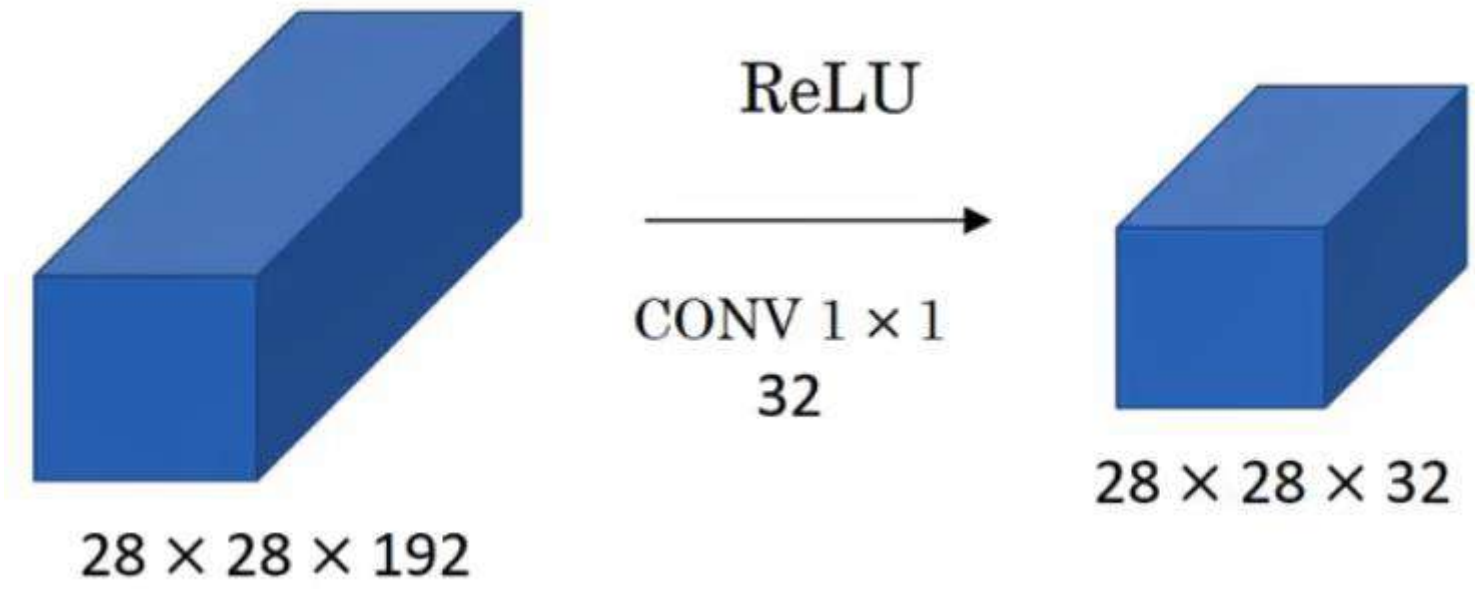
Famous CNN Models

- LeNet - 1990
- AlexNet - 2012
- ZFNet - 2013
- GoogLeNet - 2014
- VGGNet - 2014
- ResNet - 2015
- Inception v3 - 2016
- MobileNet – 2017
- Squeezenet - 2017

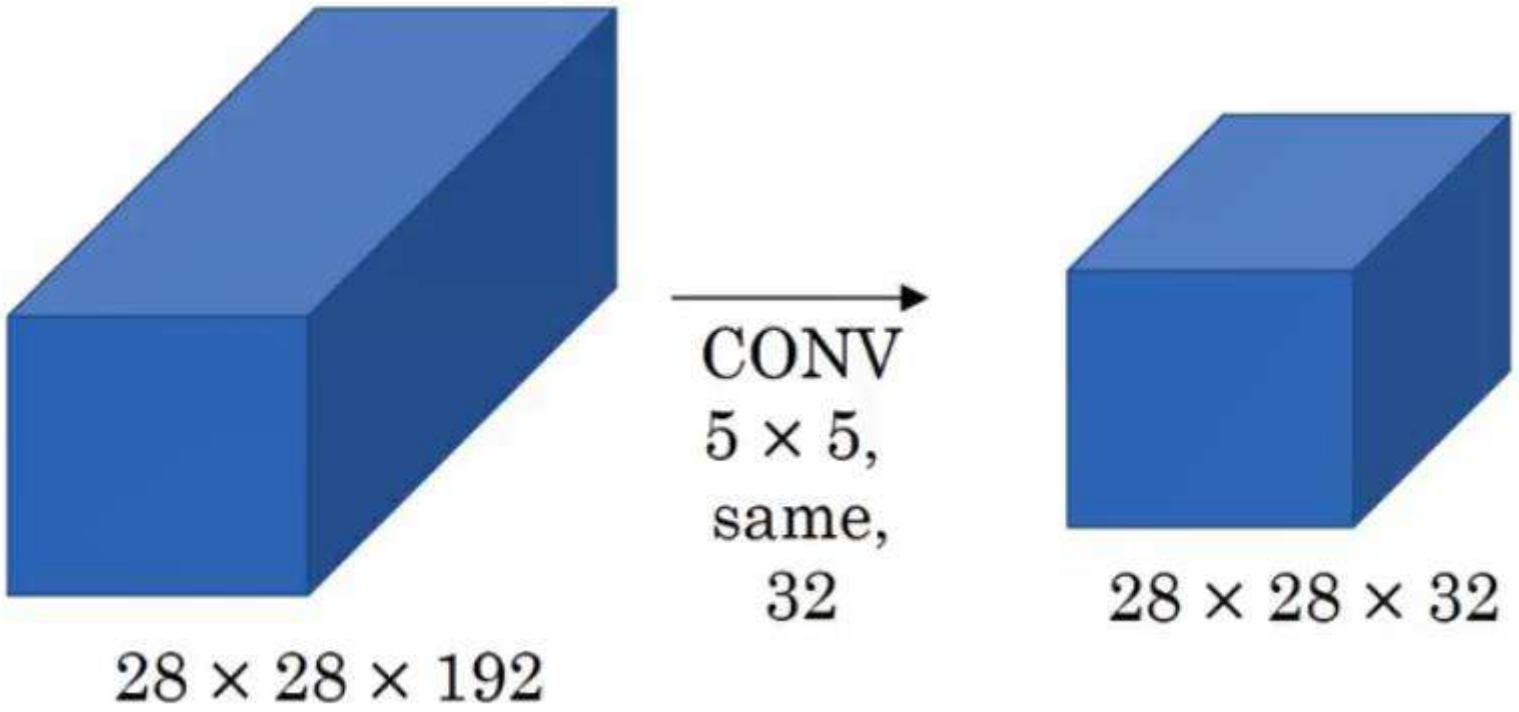
Innovative use of 1X1 Convolution Filter



Shrinking of Channels

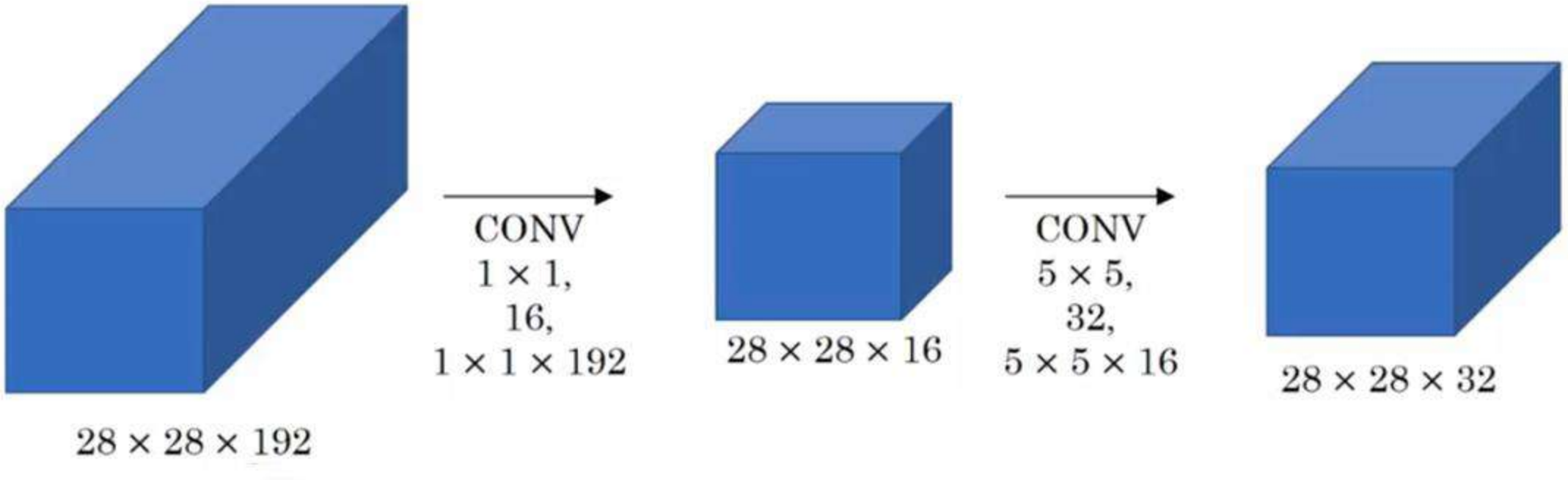


Problem of Computational Cost



28X28X32X5X5X192=120 Million Calculations

Using 1X1 Convolution to reduce the cost



28X28X16X192 + 28X28X32X5X5X16 = 12.4 Million Calculations

Open Source Implementations

-Developers has spent months in developing some good open source networks

-Many of them have access to high-speed GPUs and have the capacity to do lot of experimentation for months

-It is always good to start with a known open source implementation

-Search <name of the network> Github

-you will get few links pointing you to Github profiles of individuals who have put these network models along with the learned weights online

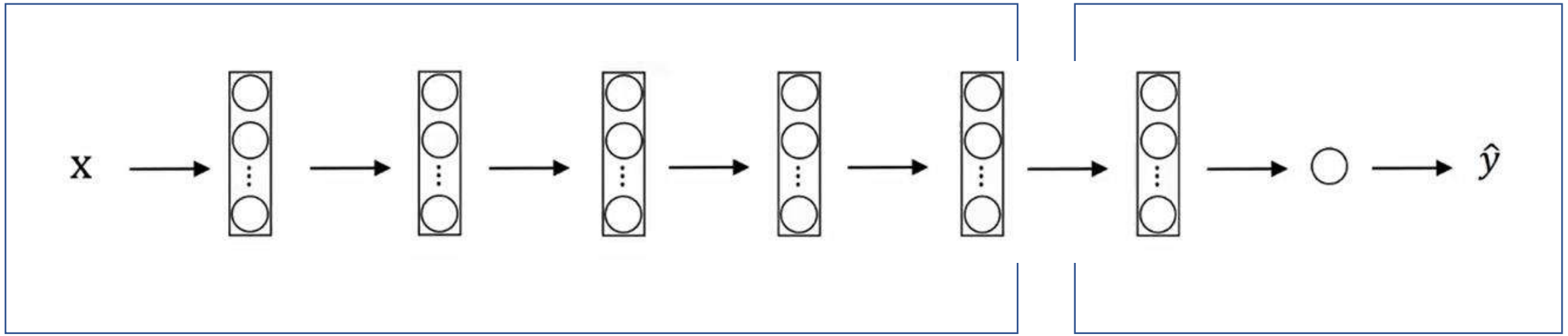
-Download or clone these network depending upon whether you are working on the your machine or remotely on a notebook

-Use this trained model fully or partially depending upon your requirements

Transfer Learning

- Sharing the Knowledge gained solving one problem and applying to a different but related problem
- Transfer Learning is next popular driver of deep learning after supervised learning
- The feature spaces of the source and target domain are different, e.g. the documents are written in two different languages.
- The marginal probability distributions of source and target domain are different, e.g. the documents discuss different topics.
- Simulation training is becoming a hot area within the sphere of Deep Learning. Few labs have also started using AR/VR Technologies to be integrated for making advance learning models for some of the critical problems area

Transfer Learning – Small Data



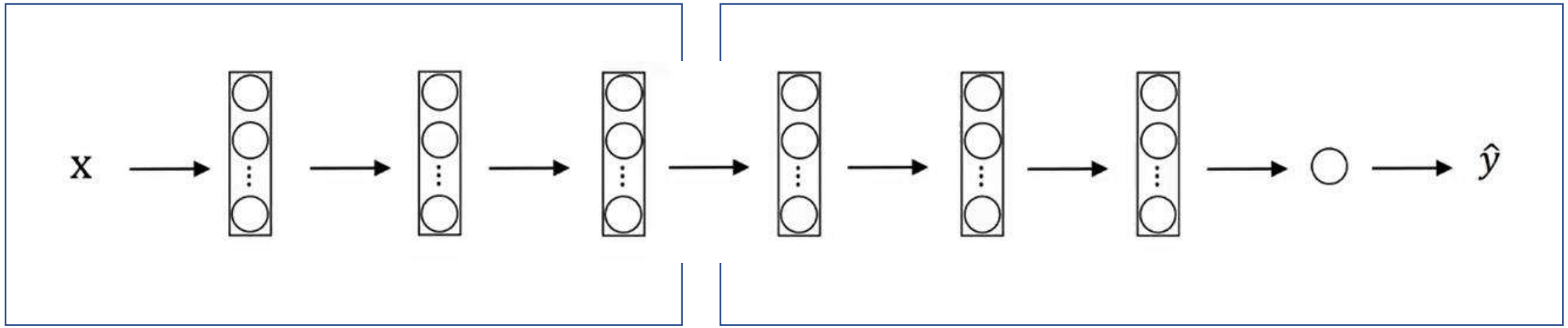
*Finding a Bengal Cat or British Cat where you have small dataset of these categories of cats

*You can download a cat classifier and train last few layers on your data specifically

*In Popular Deep Learning platforms, now you have good support for transfer learning

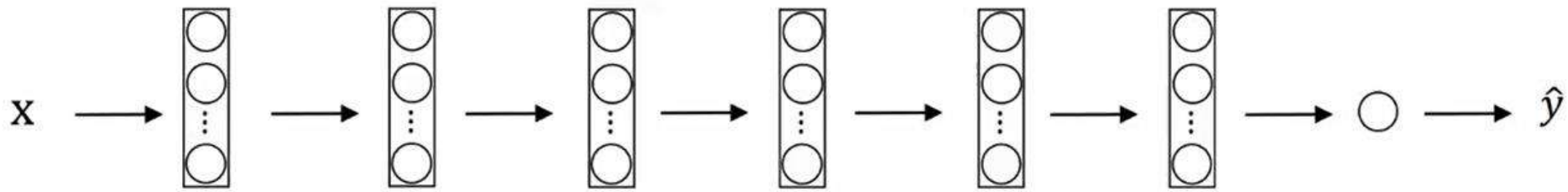
* Functions like Trainable_Parameters and Freezing specific layers are available

Transfer Learning-Mid Size Data



In this category we use initial set of layers from the open source model and use the trained weight values. For the remaining layers there are two ways to handle: a) either we can train the layers from start or b) we can start the weight optimization from the existing set of weights that we have received from the open source model

Transfer Learning- Enough Data



In scenarios, where we have enough data to train our new model, open source model may still be useful. We can use the pre-trained weights of the model from the same domain and consider that as a starting point for our model. It may be much easier and faster to adapt these model for new set of data that we want to train upon.

Object Localization and detection

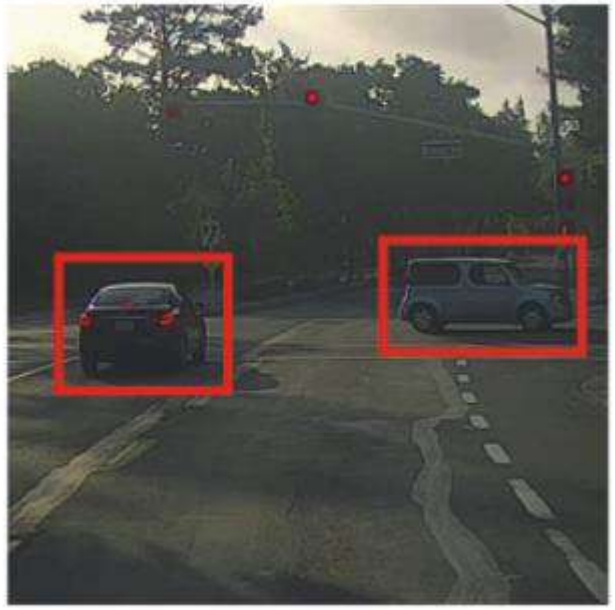
Image classification

Car or Not

Classification with localization

Usually Single Object

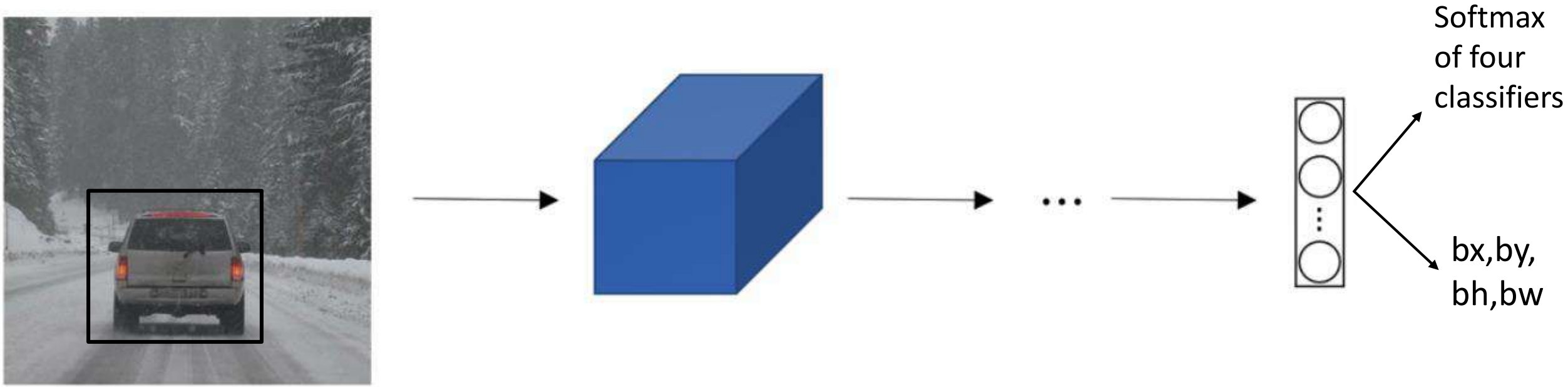
Detection



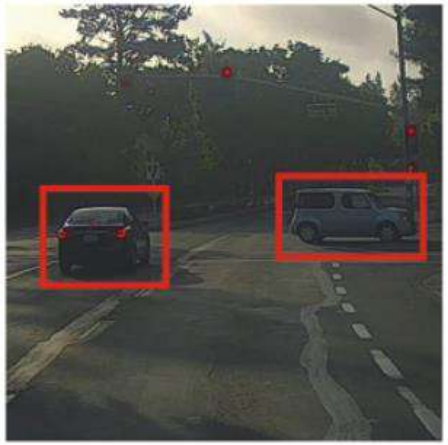
Multiple Objects

Classification with localization

Pedestrian
Car
Motorcycle
background



Car Detection



x	y
	1
	1
	1
	0
	0

Conv Net → Y

Having Closely Cropped Images to train the System

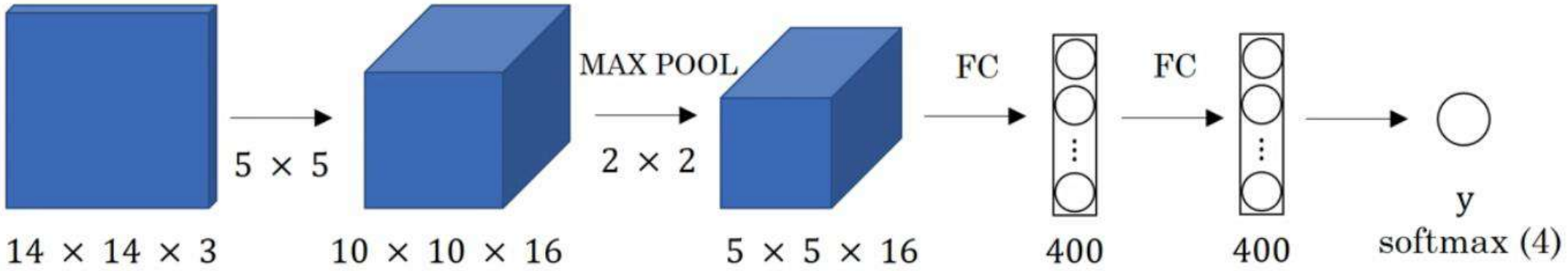
Sliding Window Detection

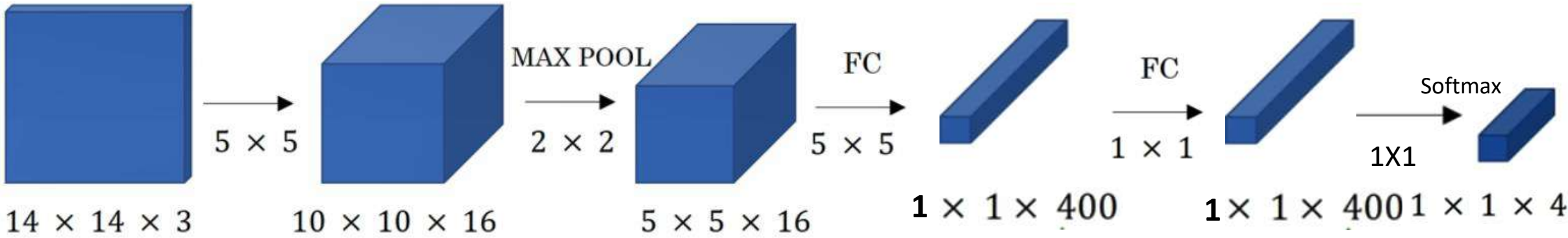
Each Iteration for each of the Sliding Window Box needs a convnet to classify and localize the object.

Computation Cost is Huge.

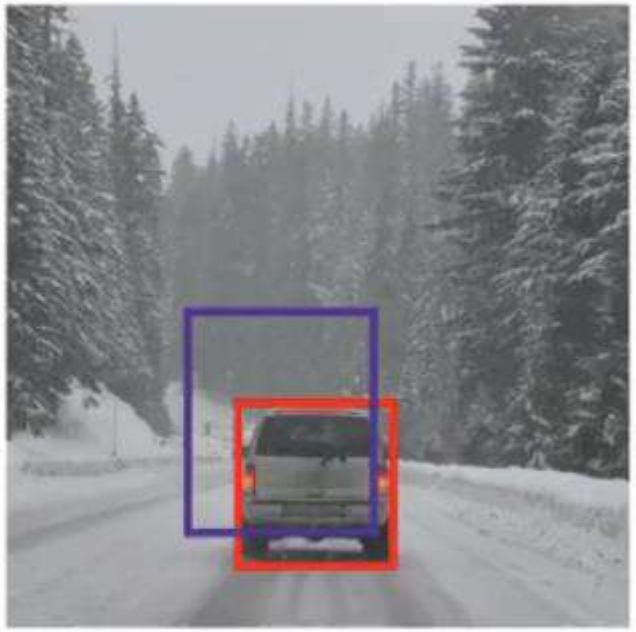
We have a solution

Turning Fully Connected Layer to Convolution Layers





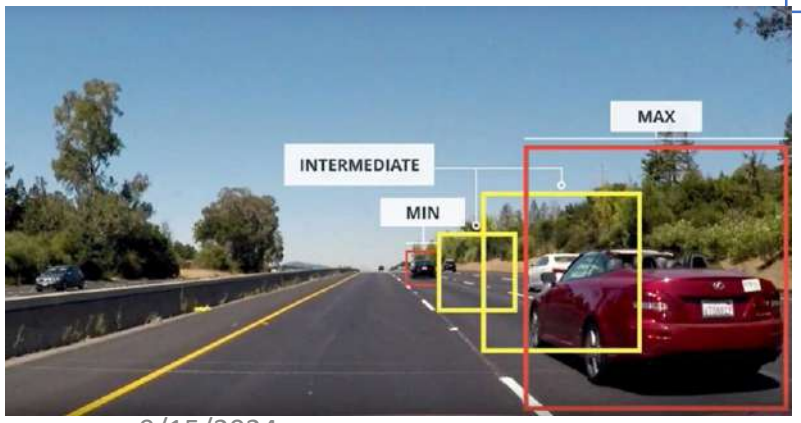
Intersection Over Union



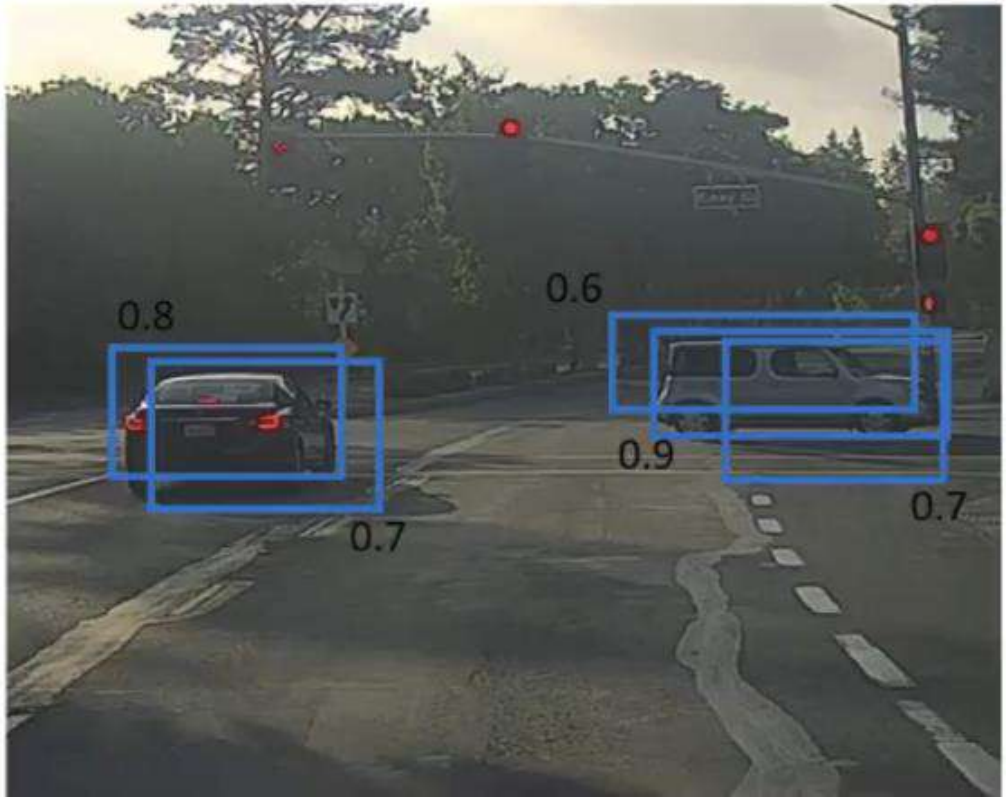
Intersection Over Union

Size of Intersect Area/Size of Union Area
May be Correct if IoU ≥ 0.5

So, it is a measure of the overlap between two bounding boxes



Non-Max Suppression



It Cleans up multiple bounding boxes around one object and gives us one bounding box per object.

Each Output (Probabilities associated with multiple bounding boxes is Pc)

Discard all boxes with Pc ≤ 0.6
While There are any remaining boxes
    Pick the box with the largest Pc
    Discard any remaining box
For Multiple objects we can repeat Non-max suppression algorithm for each of the object we are trying to find out like pedestrian, motorcycle.

Face Identification Vs. Verification

Verification

Input an image and also the Name/ID of the person

Output whether the input image is of the claimed person

Identification

Input an Image

Output whether the image is any of the K persons in the database

RNN

Recurrent Neural Networks

Sequence Application Variation

- Audio Signal to Sequence - Speech Recognition
- Nothing to Sequence or Single Parameter to Sequence - Music Generation
- Sequence to Single Output - Sentiment Classification
- Sequence to Sequence - Machine Translation
- Video Frame Sequence to Output - Activity Recognition
- Sub-Sequence from a Sequence - Finding Specific Protein from a DNA Sequence
- Outlining Specific parts of a sequence - Name Entity Recognition

Notation Understanding

X: Rama Conquered Ravana to install the virtue of dharma

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ …… $x^{<t>}$ …….. $x^{<9>}$

$T_x = 9$ (Length of training sequence: 9)

$x^{i<t>}$: t^{th} word of i^{th} training sequence

Y: 1 0 1 0 …… 0 0 0 0

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ …… $y^{<t>}$ …….. $y^{<9>}$

$T_y = 9$ (Length of output sequence: 9)

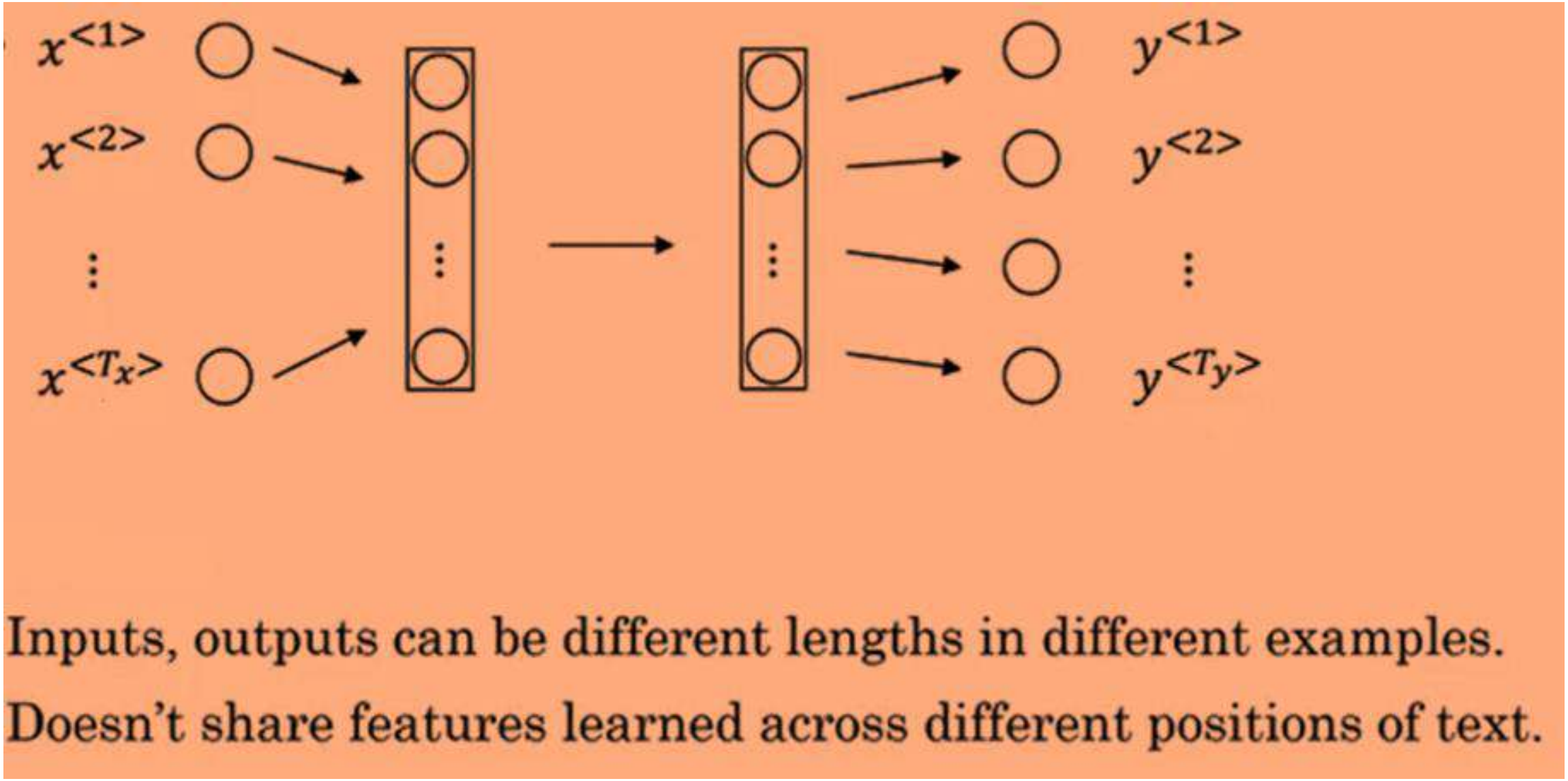
$y^{i<t>}$: t^{th} word of i^{th} output sequence

Representing words and one-hot encoding

X: Rama Conquered Ravana to install the virtue of dharma

A	1	Rama	Ravana
:		0	0
:		0	0
Conquered	329	0	0
:		0	0
:		0	0
Install	4521	:	:
:		:	:
:		:	:
Rama	7689	1 -7689	:
:		:	1-7900
Ravana	7900	:	:
:		0	0
ZZZ	10000	0	0

Standard Neural Network Does not works out to give a good application for sequence models



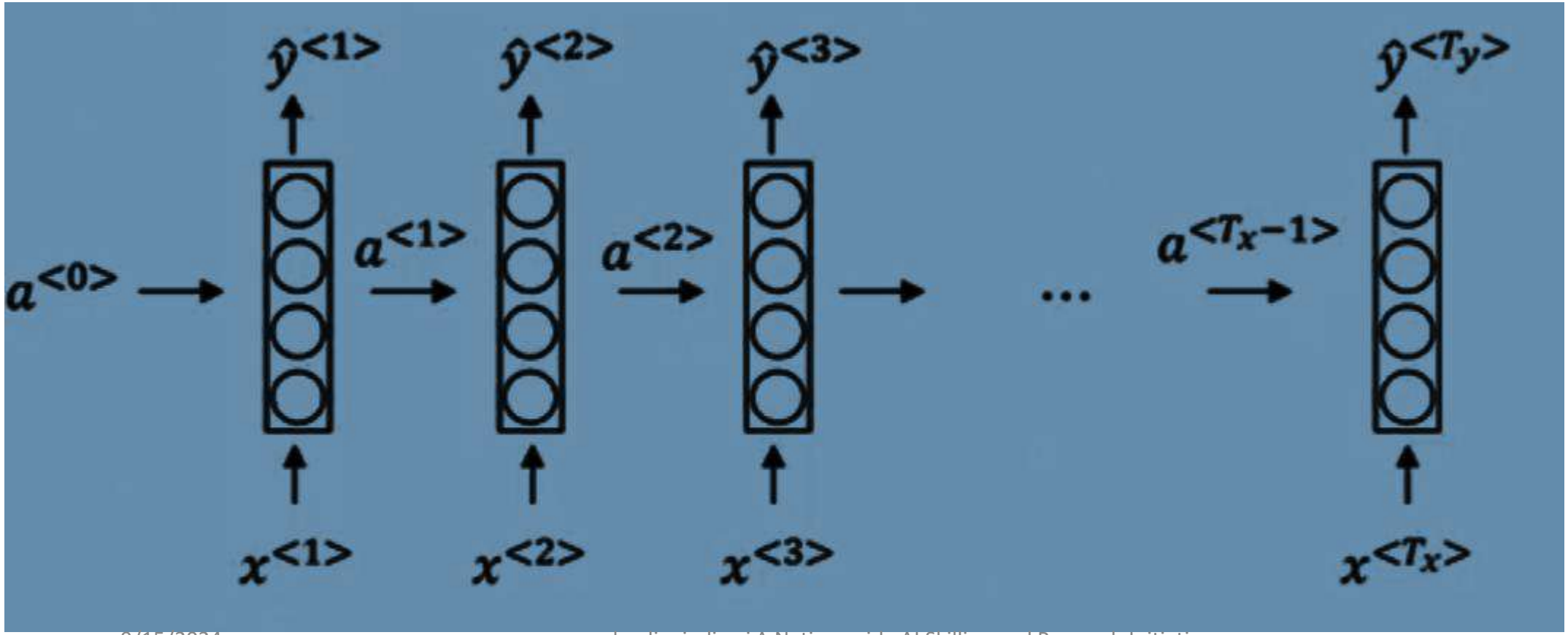
Inputs, outputs can be different lengths in different examples.

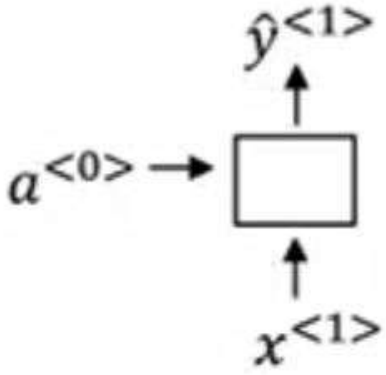
Doesn't share features learned across different positions of text.

Forward Propagation

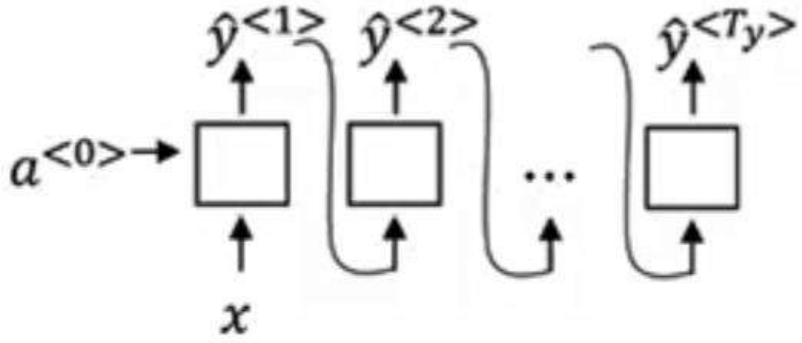
$$a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

$$\hat{y}^{<t>} = g(W_{ya}a^{<t>} + b_y)$$

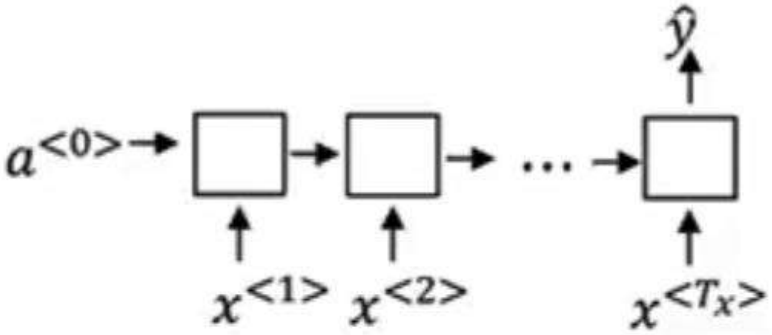




One to one

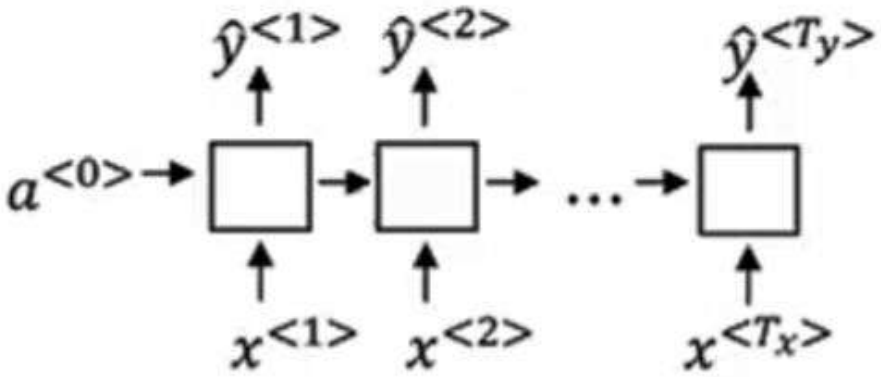


One to many

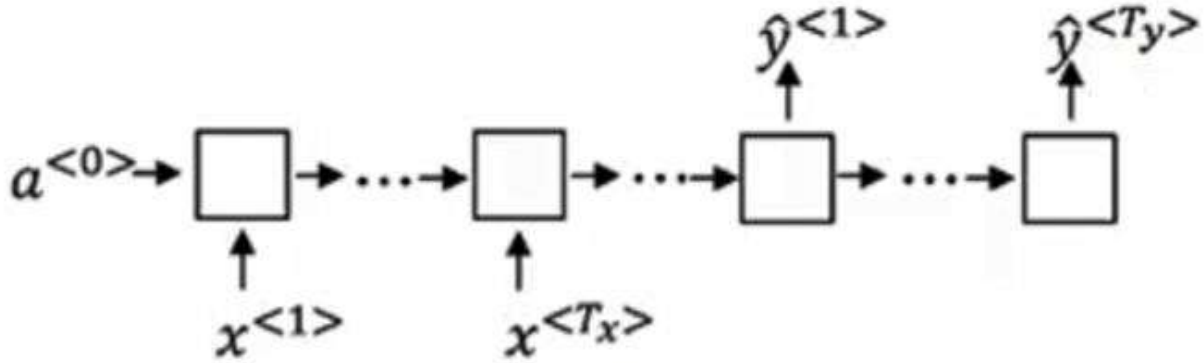


Many to one

Different Type of RNN Architectures



Many to Many



Many to Many

Word Level Language Model

Train your Language model on a large data.

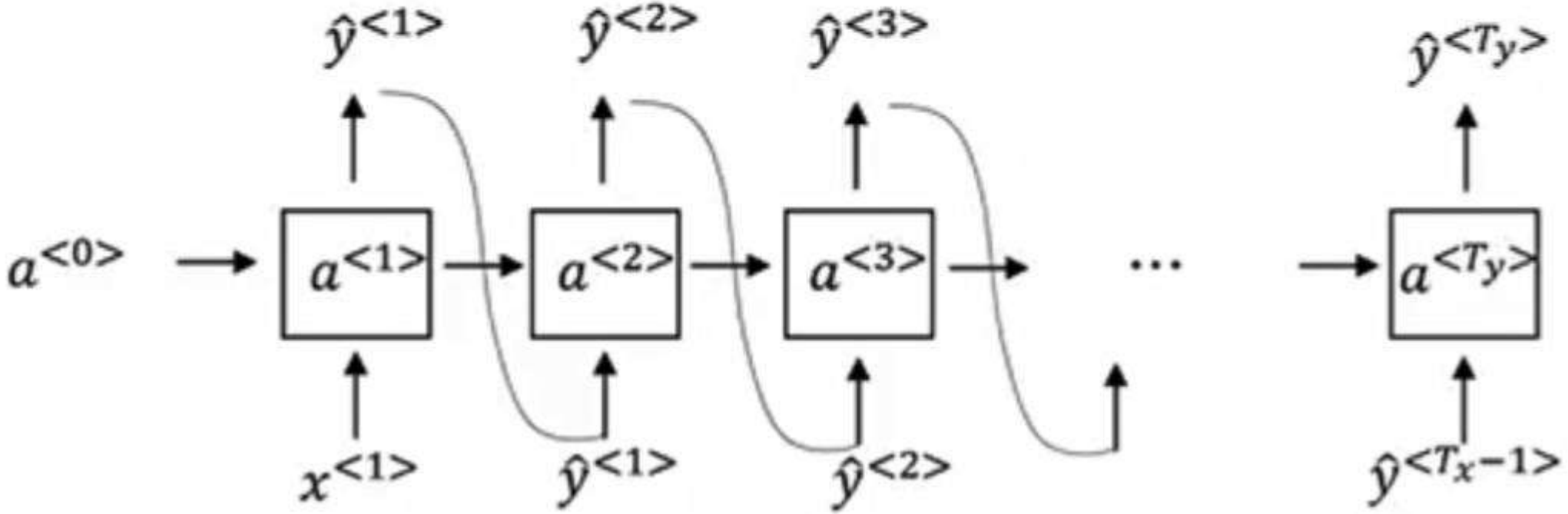
Then You can build on it different kinds of NLP applications as discussed.

Also, You will be able to generate new sentences or paragraphs etc as per the requirements of your application.

Difference between Word Level and Character Level Language Model

- Word Level Language models are more common due to their better performance as of now.
- Word level language models have <EOS> and <UNK> also as tokens in the corpus.
- Character level corpus is very small as compared to word level corpus
- In most cases word level corpus are of size 30-50k but in some cases can be upto 1 million
- In character level language model it becomes hard to predict and relate the relationship between far off characters as the distance becomes very large as compared to word language models.

Word level and character level language model



Sampling a novel sequence

Once you have a trained model on a corpus you can also have a RNN that can sample new sequences for you.

In that case you initialize with a zero and your first output gives a probability in terms of softmax function of the size of the no of categories equal to the size of your corpus.

You Choose a random word as the first output and then that word acts as the input for the second input and so on.

If you get a <UNK> then you can reject that token and continue with the next guess. It can go on until you get a <EOS> token.

Vanishing Gradients Issue

Regular RNNs are mostly influenced by local variations.

The cat , which already ate a lot while enjoying the party, was full.

The cats , which already ate a lot while enjoying the party, were full.

If the RNNs are not able to take care of long term dependencies, then the performance will be below expectations.

For Feed forward RNNs and also for back propagation, it is very difficult to reflect/relay the long term dependencies.

Gated Recurrence Unit

- Helps a Lot in Long Term Connection and also helps a lot in vanishing gradient issue .
- Main difference is that bring in the change in the hidden unit calculations and we introduce a memory cell
- We introduce an update gate u which will have value 0 or 1
- When the value of Gate is 0 that means it will keep the previous value and will not update else it will update the previous value with the new value
- We also introduce a gate r which means relevance of previous memory cell with the current cell

GRU

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[\, c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[\, c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

$$a^{<t>} = c^{<t>}$$

LSTM (Long Short Term Memory)

- In this we don’t have memory Cell value equivalent to activation value.
- We also introduce an additional gate called forget gate. It gives us the option to keep the previous values and also to add/update this gate with additional value from update gate.
- LSTM is most popular now for dealing with long term dependencies
- We also have a output gate.
- LSTM is more robust than GRU, but people use both of them based on applications.

LSTM

$$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * \tanh c^{<t>}$$

Featurized Representation: Word Embeddings

	Man 5391	Woman 9853	King 4914	Queen 7157	Apple 456	Orange 6257
Gender	1	1	0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0
Age	0.03	0.02	0.7	0.69	0.03	0.02
Food	0.04	0.01	0.02	0.01	0.95	0.97
Size						
Cost						
Alive						

If we have 300 such properties and 10000 Words then it will be a 300x10000 Matrix and is denoted as Embedding Matrix (E) and O_{man} is One hot Vector for Man Word and e_{man} can be embedding vector for Man word.

I Want A glass of orange __Juice
I want a glass of Apple

Transfer Learning and Word Embeddings

- Learn Word Embeddings from Large Text Corpus (1-100 Billion Words)
- Pre-trained embeddings are available online
- Transfer Embedding to a new task with smaller training set
- Continue to finetune word embeddings with new data

Analogies using Word Vectors

As Man-> Woman King->?

As INR->India Dollar->?

As Delhi->India Kathmandu->?

As Tall->taller Big->?

As Man->Woman Boy->?

……

$e_{man}-e_{woman} \approx e_{king}-e_{w}$

Find a word w : Maximize similarity(e_w , $e_{king}-e_{man}+e_{woman}$)

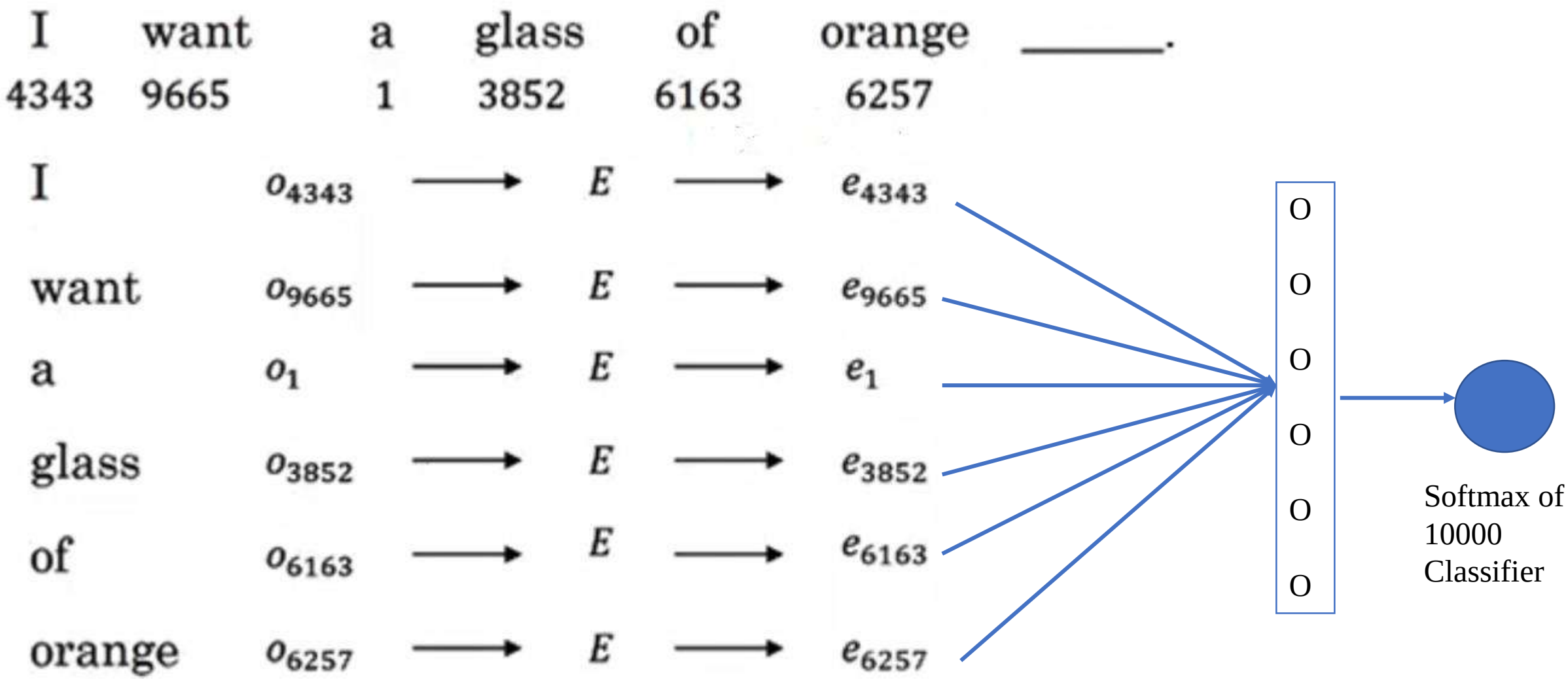
Cosine similarity

Sim(u,v)= $u^T v$ / ||u|| ||v||

Contexts which can help to learn

- Last 4 words
- Last x words and next x words
- Last one word
- One nearby word
- Randomly pick a word to be a context word and randomly pick a target word with in a window of x of the context word – Skip Gram Model

Neural Language Model



Skip Gram Model

I want a glass of orange juice to go along with my cereal

Context: Orange Target: Juice

Context: Orange Target: Glass

Context: Orange Target: my

O_c->E->e_c------→Softmax Classifier→ŷ

Θ_t is the parameter associated with output t i.e chance of being the label

$$p(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10,000} e^{\theta_j^T e_c}}$$

Negative Sampling Algorithm

Defining a new learning problem

I want a glass of orange juice to go along with my cereal

We will use of pair of words and then find out whether the second word can be a target word for the first context word

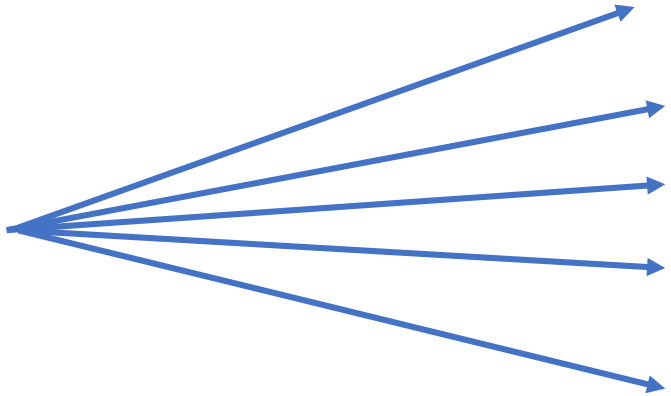
Context	Word	Target	
Orange	Juice	1	(We choose First example as positive example)
Orange	King	0	
Orange	Table	0	
Orange	to	0	
Orange	Pencil	0	

Generally we choose 5-20 such pairs for smaller datasets and 2-4 words for bigger data sets

Converting Softmax to Logistic Classifier

- O_c->E->e_c------→10000 Binary Logistic Classifier→$\hat{y}$

- O_c->E->e_c



- We have 10000 binary classifiers but we only train 5 random pair words in every iteration

Removing Biases in NLP

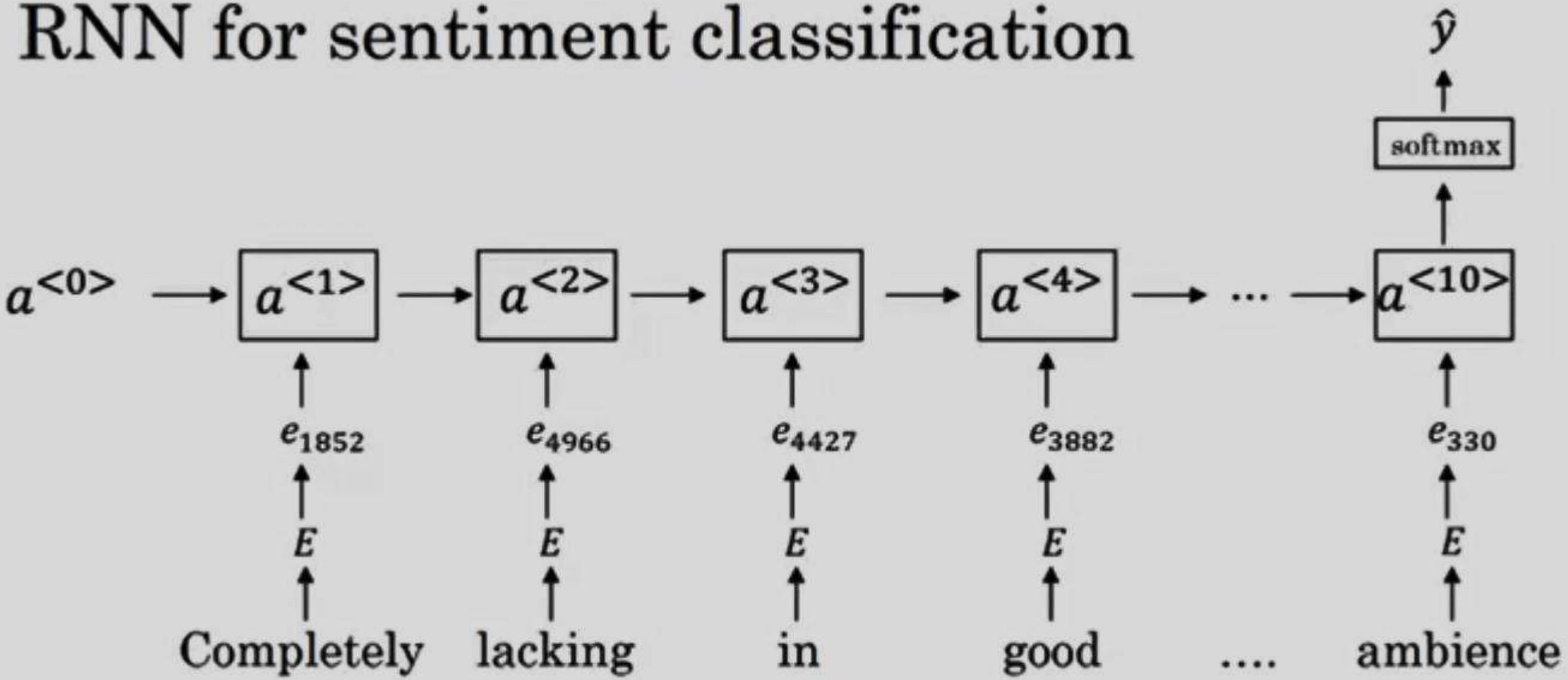
- It related to Gender and ethnicity biases and we need to be very careful about this
- Man: Computer_Programmer as Women Homemaker
- Father: Doctor Mother: Nurse

- Biases will be picked from the text it has been trained upon
- First step is to identify Bias Direction e.g. male to female
- Next is to Neutralize the bias for all the non-definitional word for example Father, Mother, He, She are definitional word for Gender and should not get changes due to this. However, Non-definitional word like soldier, doctor, Manager, Programmer etc should be neutralized for bias
- Last step is to equalize pairs like niece, nephew; grandmother, grandfather and they should be equidistant from words like babysitter etc.

Simple Sentiment Classification Model

Counter Example :Completely Lacking in Good Ambience, Good Taste, Good Service

RNN for sentiment classification



Sequence to Sequence Model

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ $x^{<4>}$ $x^{<5>}$

Jane visite l'Afrique en septembre

⟶ Jane is visiting Africa in September.

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$

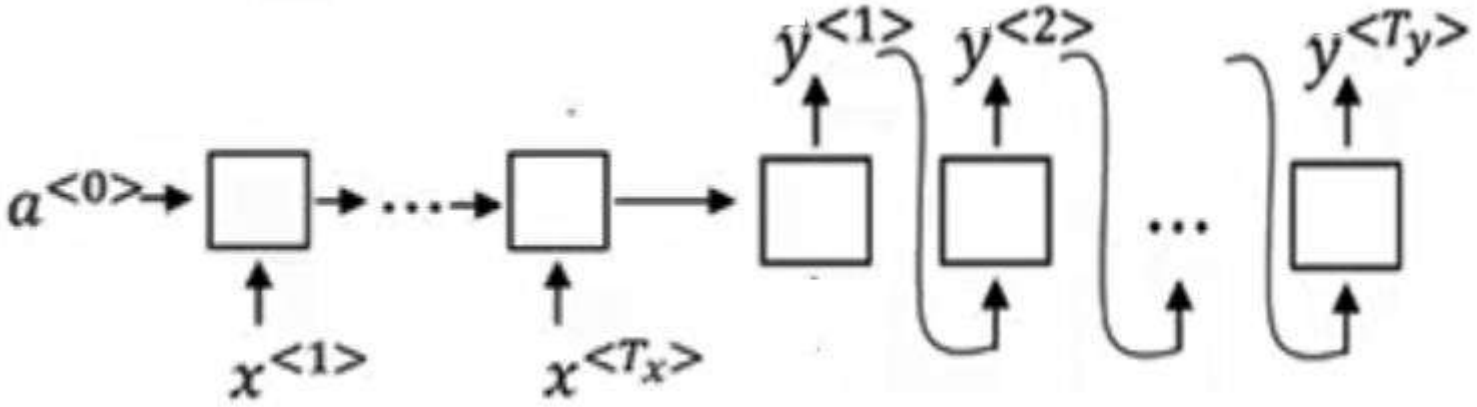
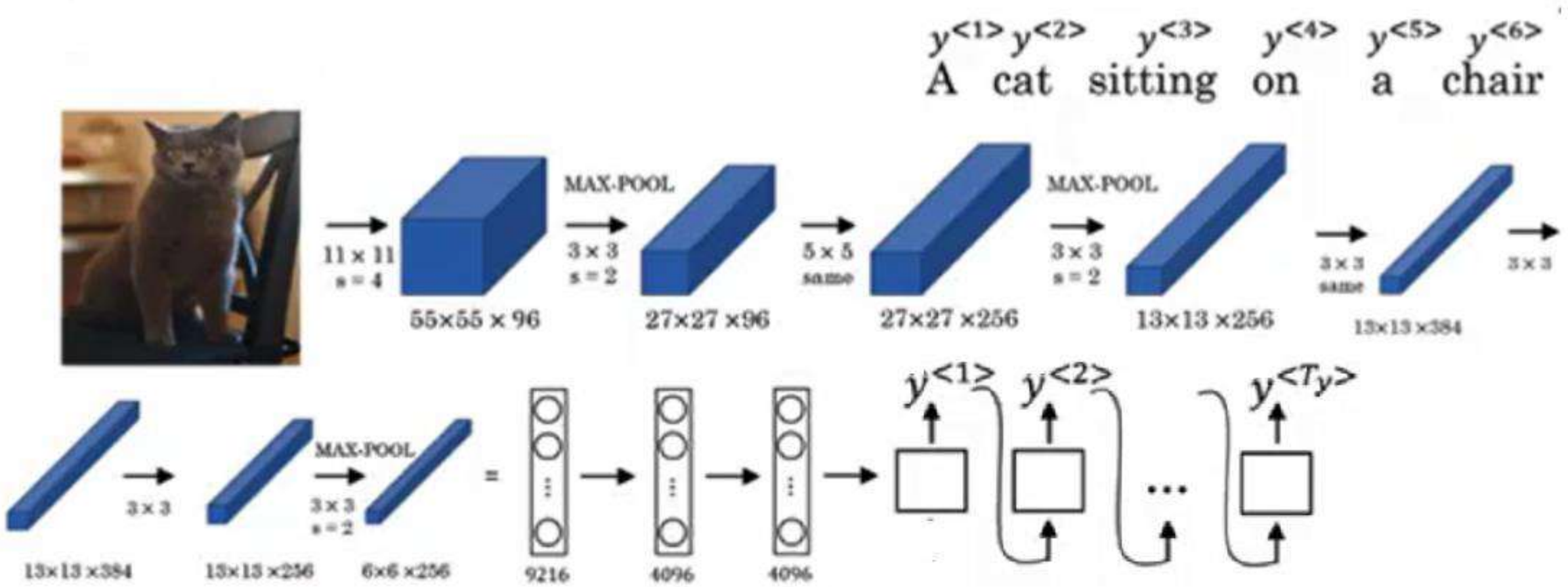
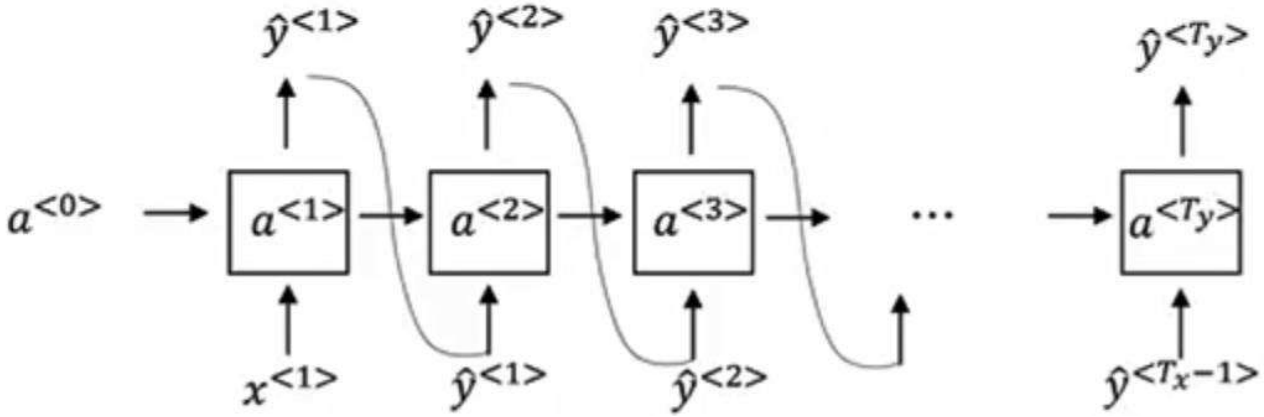


Image captioning model

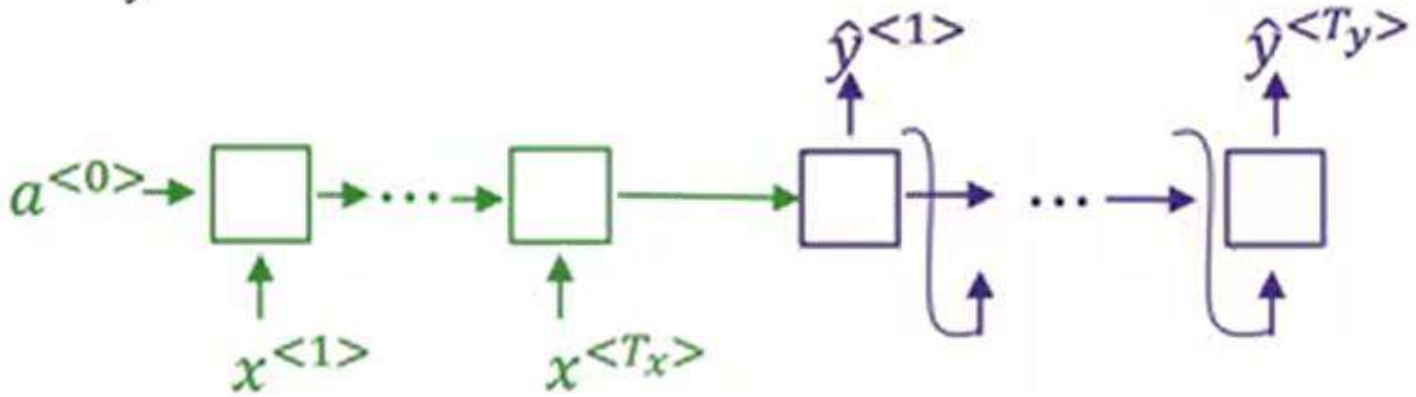


Machine translation as a conditional Language Model



Language Model

$$P(y^{<1>}, \ldots, y^{<T_y>} \mid x)$$



Translation Model

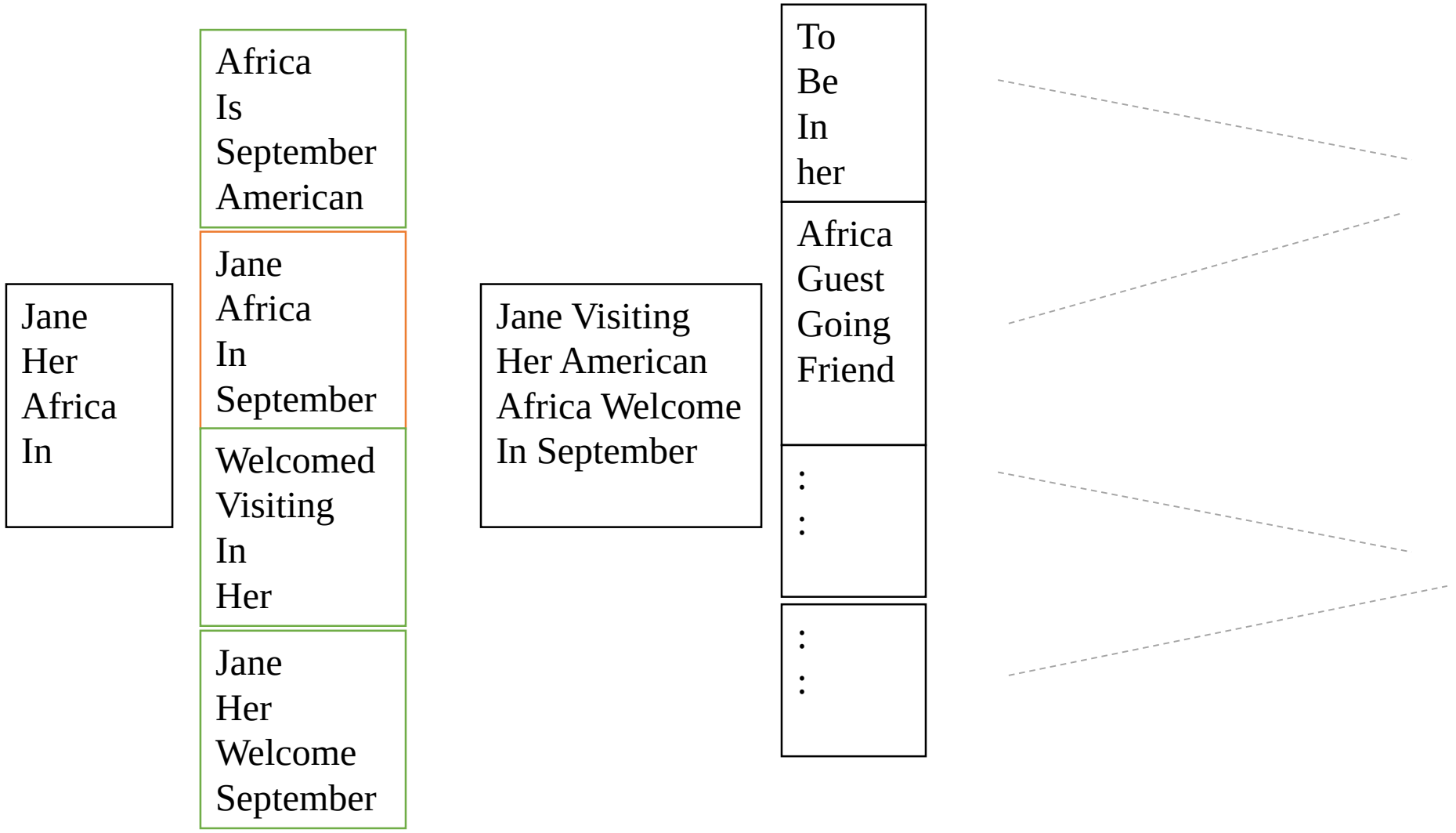
Finding the Most likely translation

Jane Visite l'afrique en septembre

- Jane is visiting Africa in September
- Jane is going to be visiting Africa in September
- In September, Jane will visit Africa
- Her American Friend Welcomed Jane in September

- If the length of the output sequence is 10 and the size of the corpus is 10000 then the possible number of permutations are 10000^{10}
- In a Greedy search algorithm we can choose the first word which is most likely to come (with maximum probability) and then the next output word of the translation with the maximum probability and so on. But, in practice this does not works and does not gives us a good translation.

Beam Search


Jane
Her
Africa
In
Africa
Is
September
American
Jane
Africa
In
September
Welcomed
Visiting
In
Her
Jane
Her
Welcome
September
Jane Visiting
Her American
Africa Welcome
In September
To
Be
In
her
Africa
Guest
Going
Friend
:
:
:
:

Beam Search

In this technique we will choose width of the Beam (Lets say 4)

It means that now we will select 4 words with highest probability given by the output of the softmax function in the last layer

Now corresponding to the each word in the first output we will choose 4 words who have max probability to come as a second word given first word.

Out of these 16 combinations of first two words, we will only choose 4 pairs with highest total probability

Now we will choose the third word corresponding to these four pairs of first two words and so on.

Beam Width

- Large beam width will result in slow performance but better results
- Smaller beam width will result in good performance with compromise in accuracy,
- Generally the acceptable length of the beam width in production systems will be in the range of 3-10 based on the applications, while in case of researchers it can even go to 100.

Bleu Score

- Bilingual Evaluation Understudy **Score**, or **BLEU**for short, is a metric for evaluating a generated sentence to a reference sentence. A perfect match results in a **score** of 1.0, whereas a perfect mismatch results in a **score** of 0.0
- Le Chat est sur le tapis
- The cat is on the mat –Ref 1
- There is a cat on the mat – Ref 2
- The the the the the the the
- Precision 7/7
- But in case of bleu score we use the clip count, which considers the maximum no of times the word appears in reference sequences. In this the appears twice in the Ref1 so clip count will be 2/7

Bleu score

It is important because there can be multiple right translations of a sentence, in that case also we need to choose one of them and it also helps in correctly assessing the error distance between the right translations and machine translation.

The cat is on the mat –Ref 1

There is a cat on the mat – Ref 2

The cat the cat on the mat – Machine Translation

The cat 2 1

Cat the 1 0

Cat on 1 1

On the 1 1

The mat 1 1

4/6 is the precision on the bigram

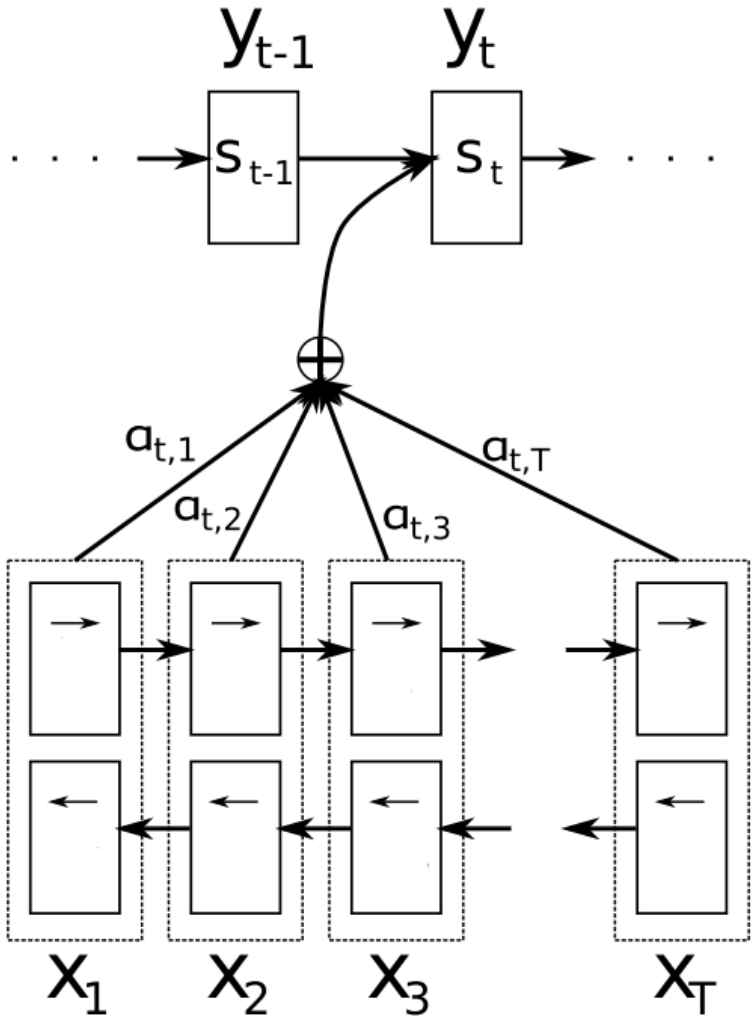
Similarly we will do on trigram and so on

It will help us to choose the right sequence

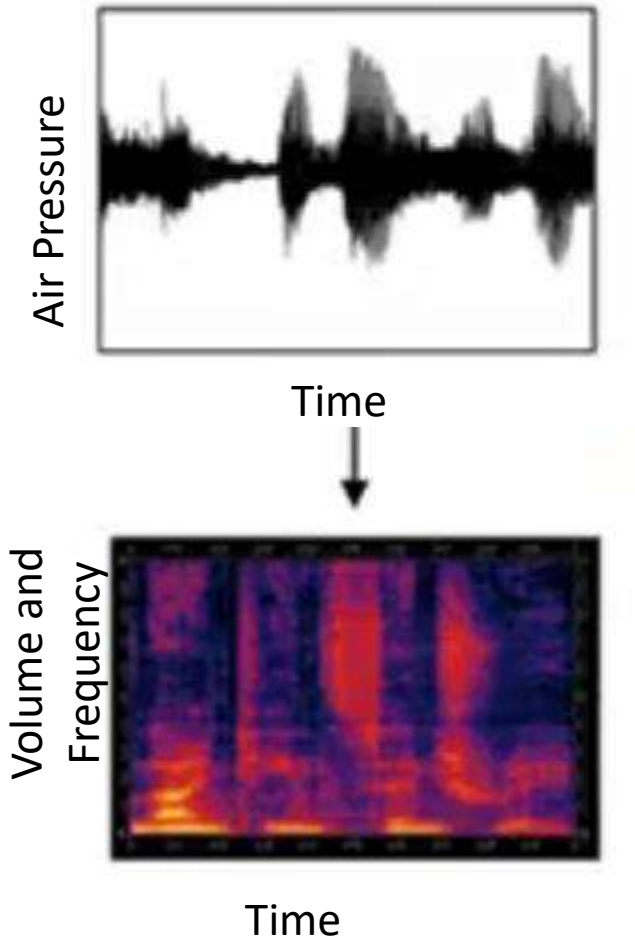
Problem of long sequences: Attention model

- When the sequences grow beyond a certain length (20 or more), then the bleu score goes down considerably and generally does not has good mapping with the goodness of the translation.
- To handle this recently Researchers came out with a new model called attention mechanism.
- It basically tells that how much attention needs to paid to each word of the source sequence for every position of the translated sequence.
- Total of attention values for any particular word should be equal to 1, so we can think of it as a softmax classifier with a small neural network for determining the probabilities of each attention value vector.

Attention Model



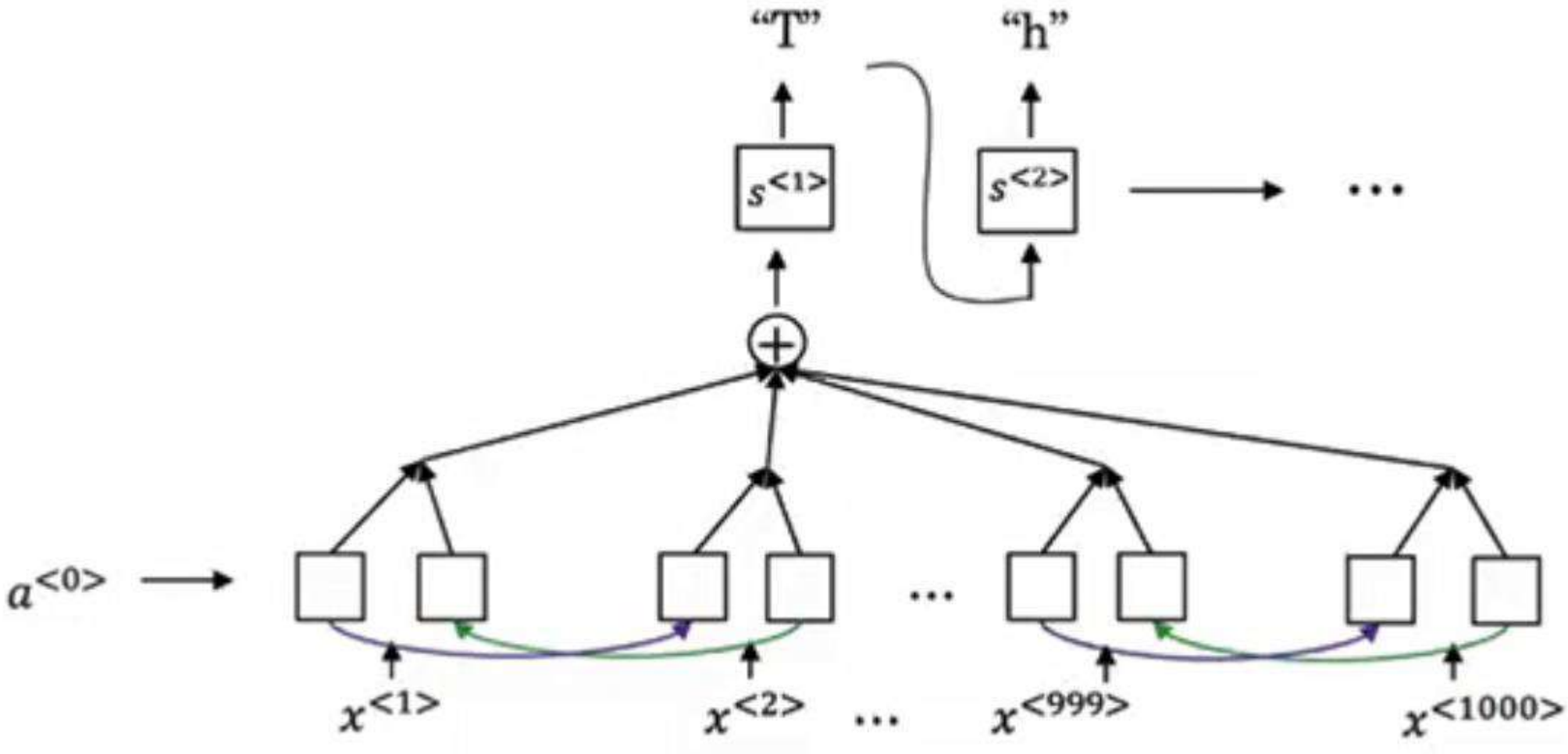
Speech Recognition



Previously we used to have Hand-engineered features consisting of different phonemes

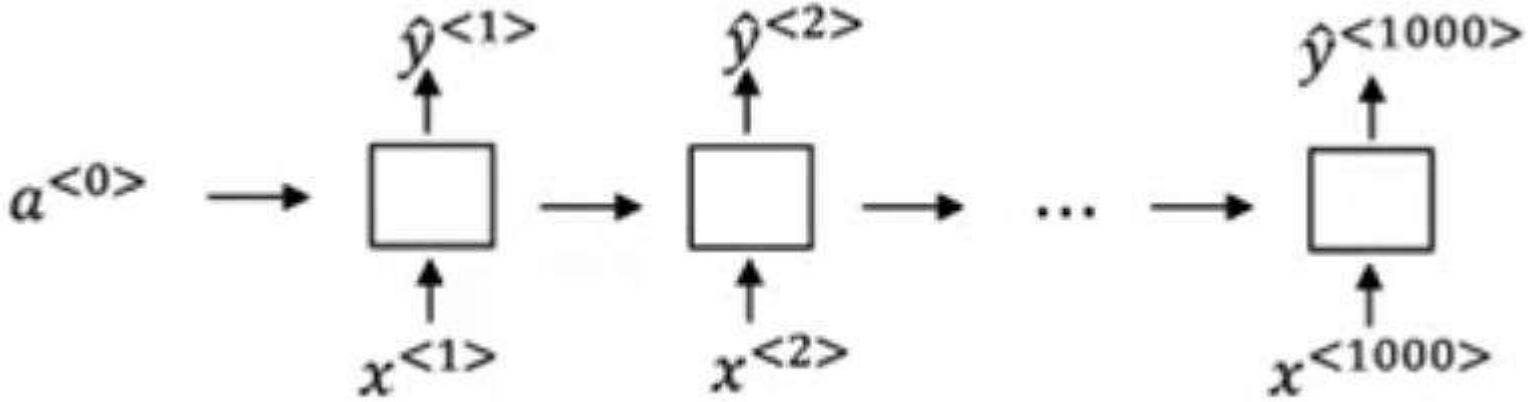
Thousands of hours of speech/audio data is used to train the data depending upon the application

Attention model for speech Systems



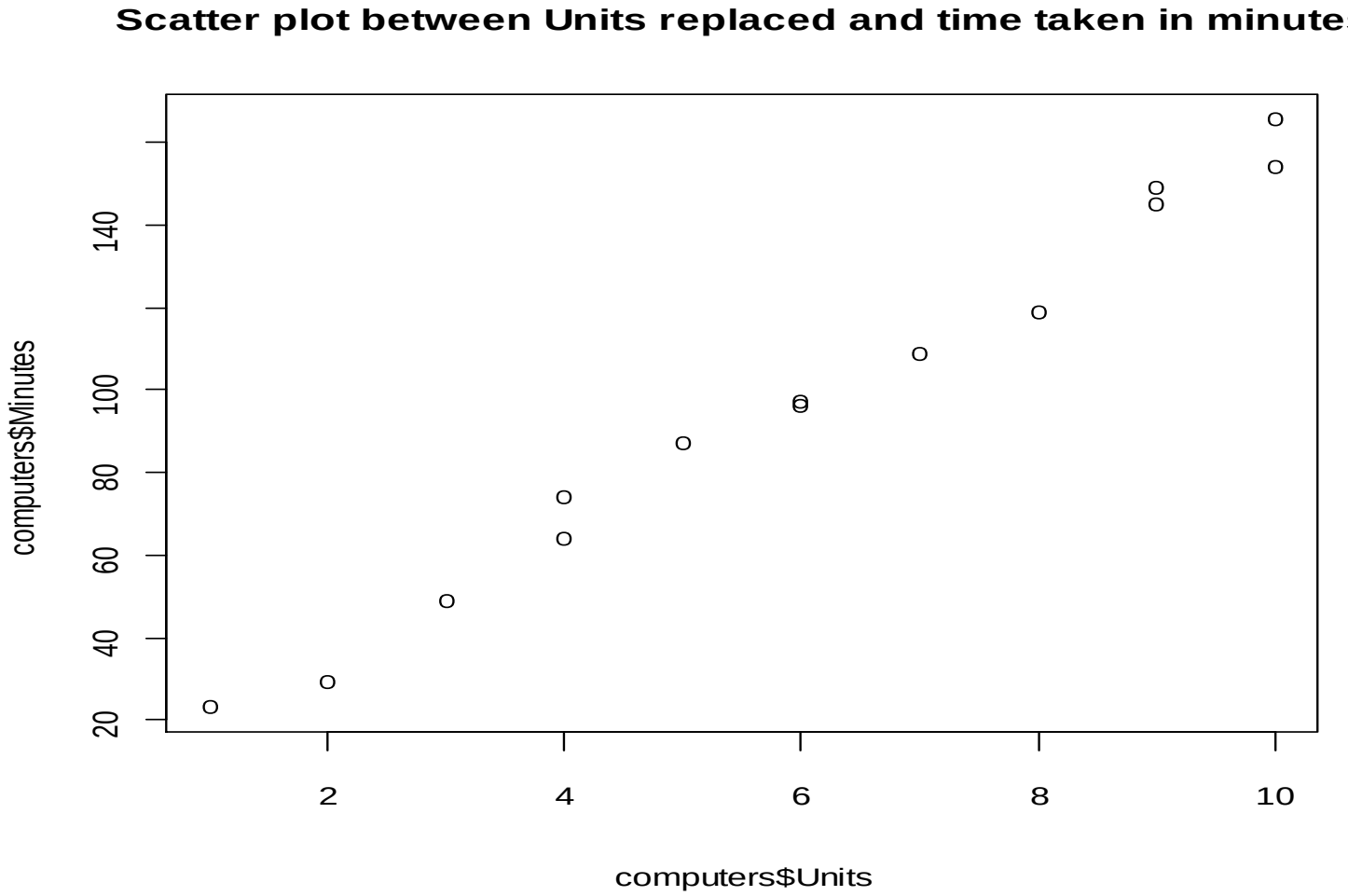
CTC Cost for Speech recognition

- Connectionist Temporal classification
- The Quick Brown Fox



- One of the issues in Speech Systems is that for 1000s of inputs your speech translation may actually produce only a few words. The reason is that the time frame we process to consider a individual sound unit is small as compared to the actual spoken words. So it is important to map the 1000s of outputs to a few words.
- E.g. ttt_h_eee____ ___ ␣_ _ _ _qqqq _ _ _ _
- ␣represents space and _ are blanks

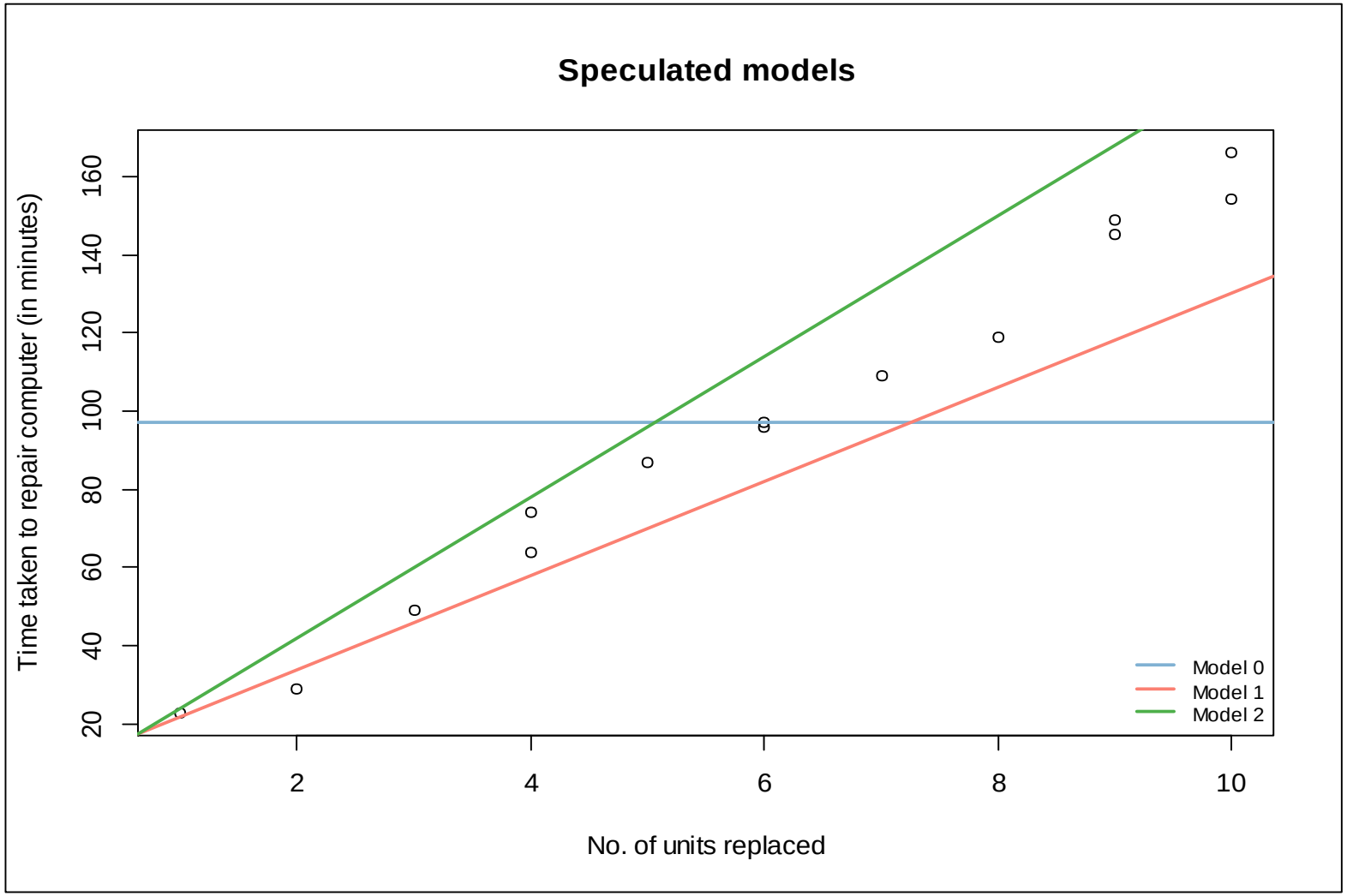
Validating the linearity assumption



It can be observed that the time taken to repair a computer (in minutes) exhibits a linear relationship with the number of units being replaced.

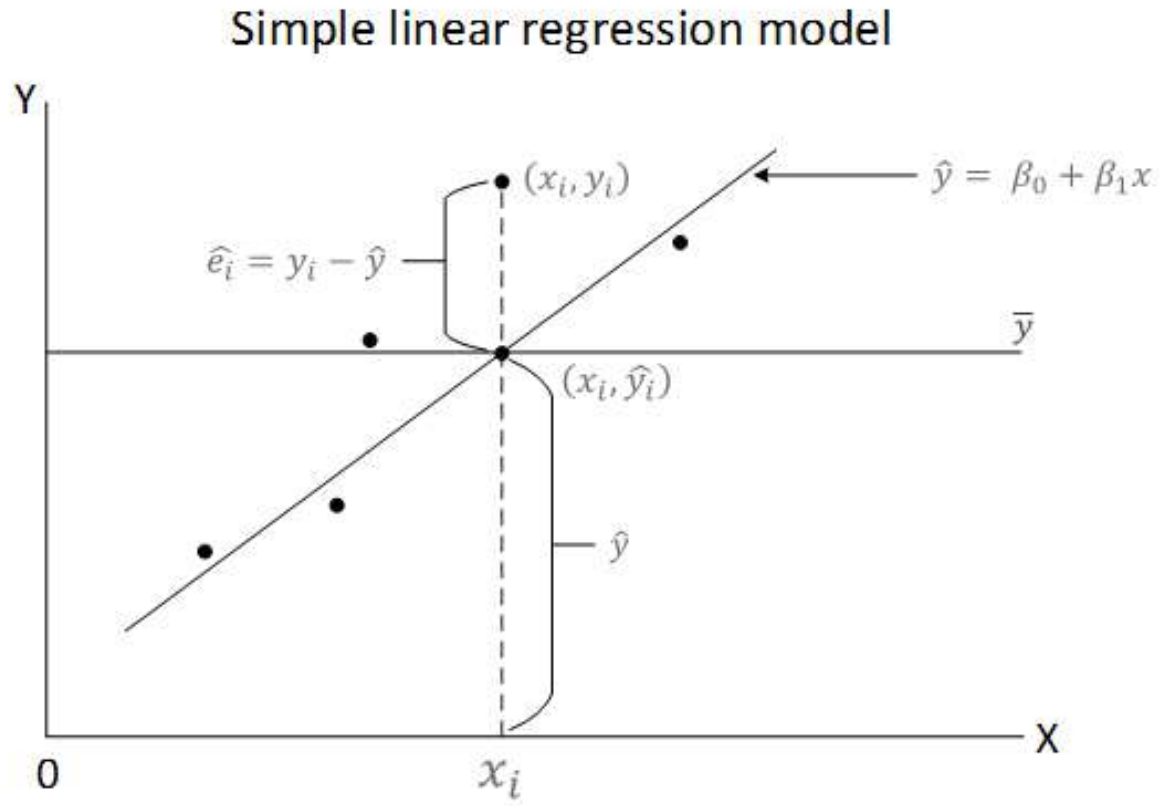
Creating regression models

Plots of model 0, model 1 and model 2 along with the given data are as shown below.

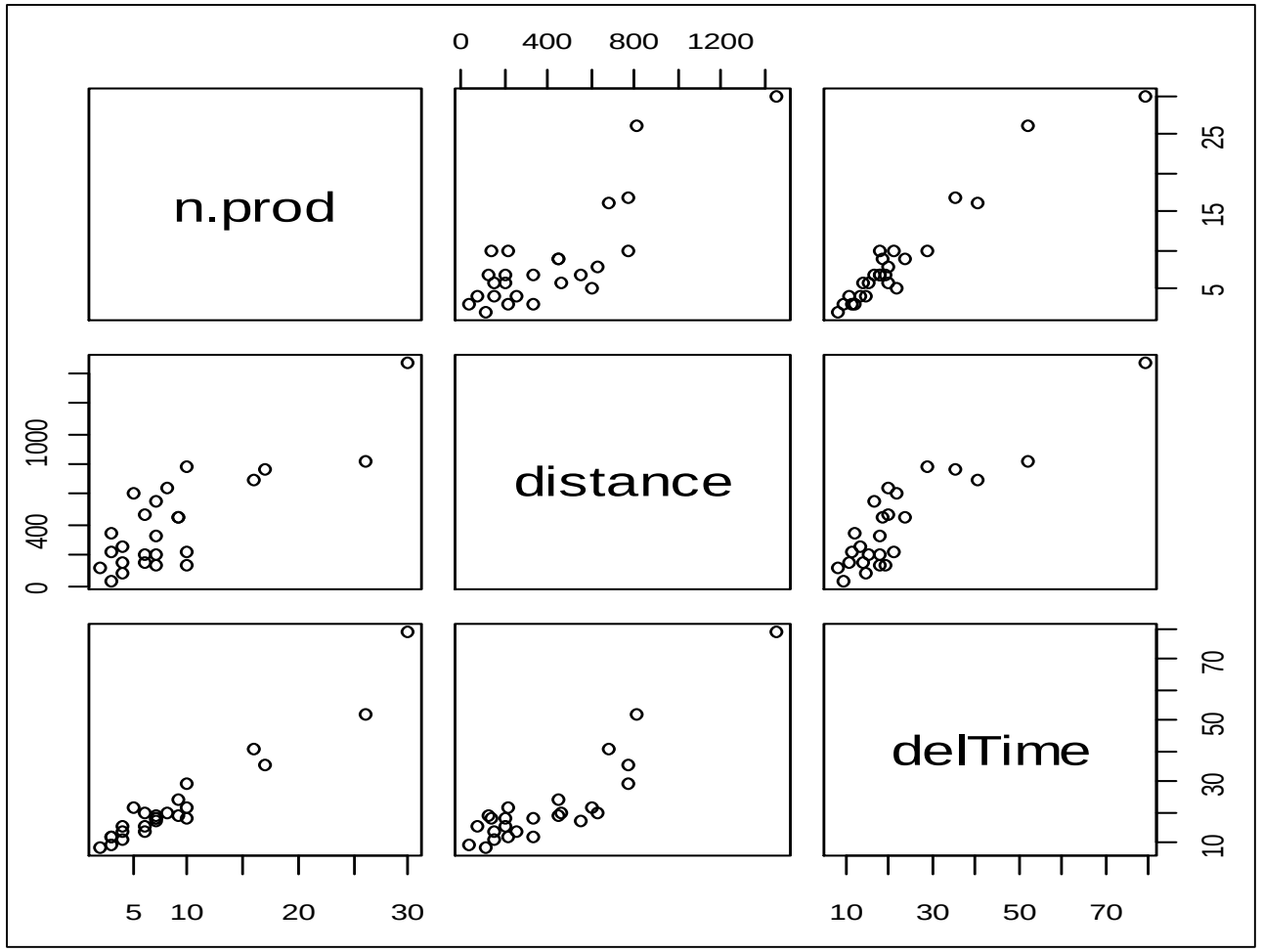


Coefficient of determination

In general, the residuals of a simple linear regression model can be visualized as shown below:

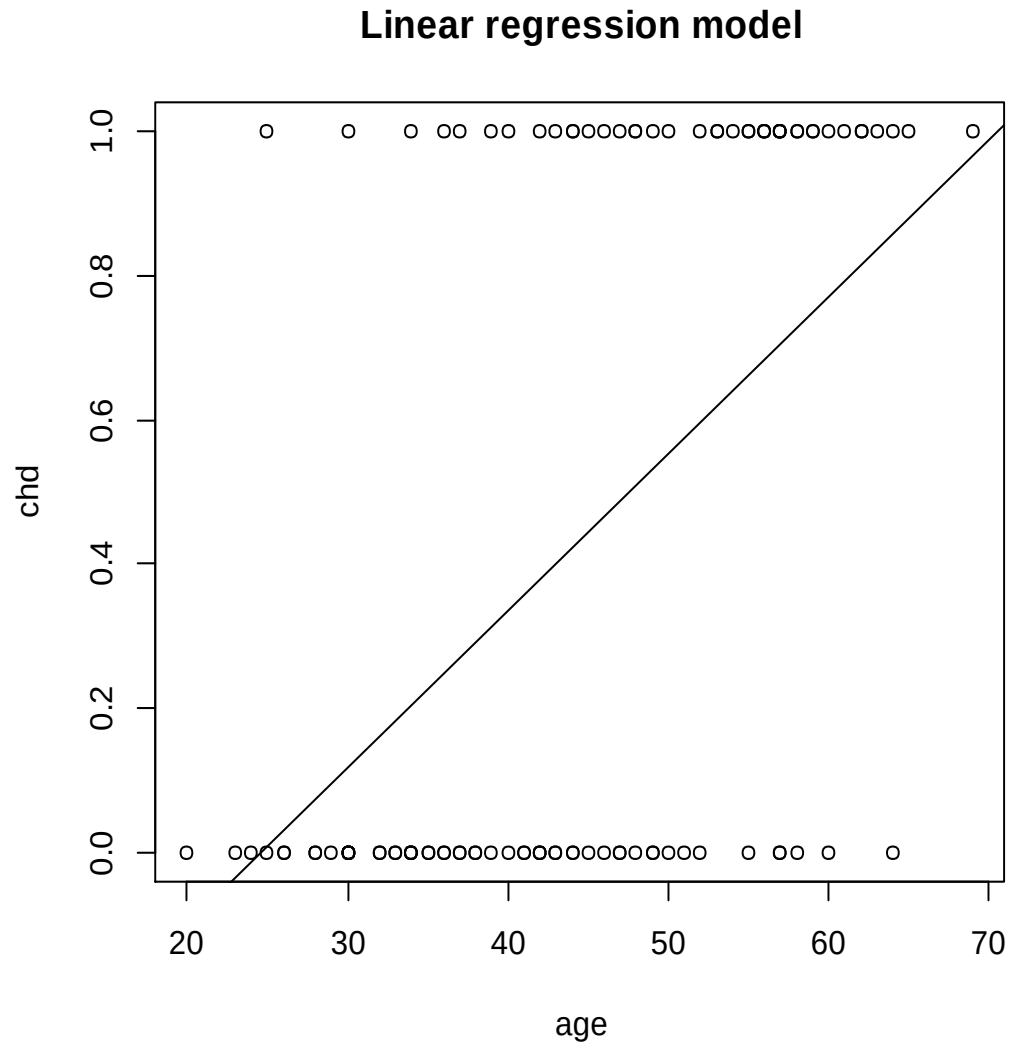


Visualizing the delivery time dataset



The scatter plot matrix for the delivery time dataset shown above suggests a linear relationship among its variables – n.prod, distance and delTime.

Logistic regression



Questions ?

Thanks