Focus for this lecture

Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

- Back Propagation
  - Loss and Objective
  - Chain Rule
  - Momentum
- Vanishing Gradients

Back Propagation

Training Neural Networks

Blank Slide: Recap of Gradient Descent

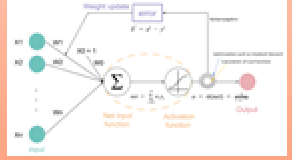
Blank Slide: Recap of Gradient Descent

Plan for the Lecture

- Introduce the “Classical’’ Back propagation algorithm
  - ``Error’’ back propagation
  - The same algorithm for all “deep’’ networks
  - Many practical refinements/tricks in implementation (later lecture)

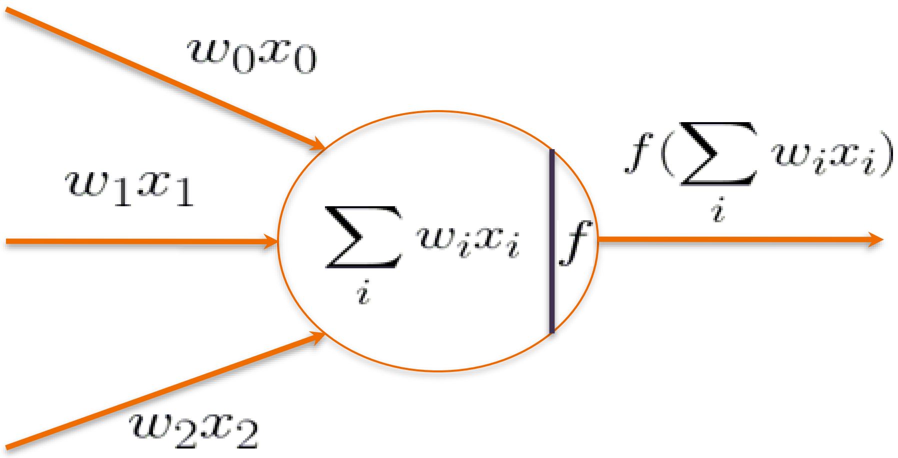
Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

- Back Propagation
  - Loss and Objective

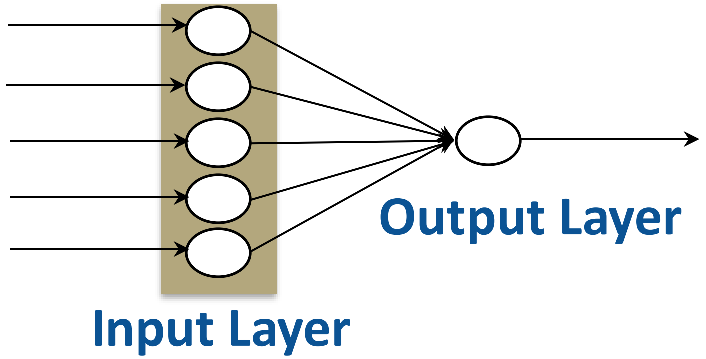


Loss/Objective, Error and Learning

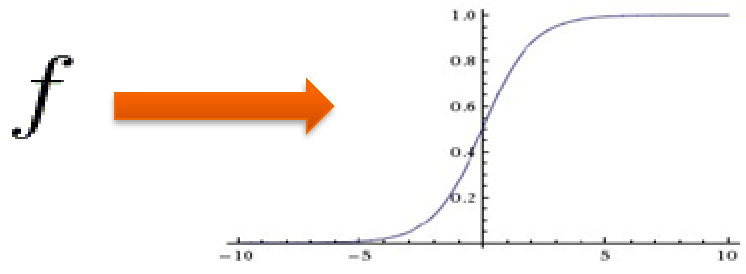
Neuron, Perceptron and MLP



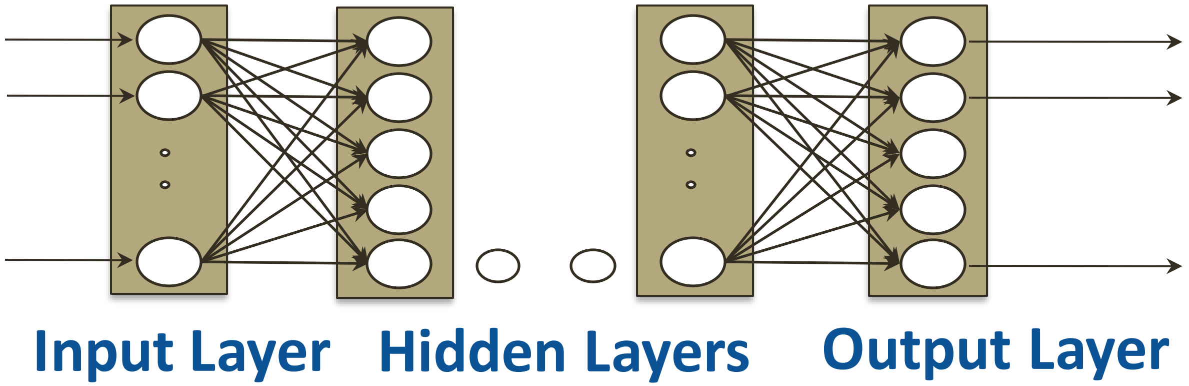
Hidden unit/Neuron



Perceptron

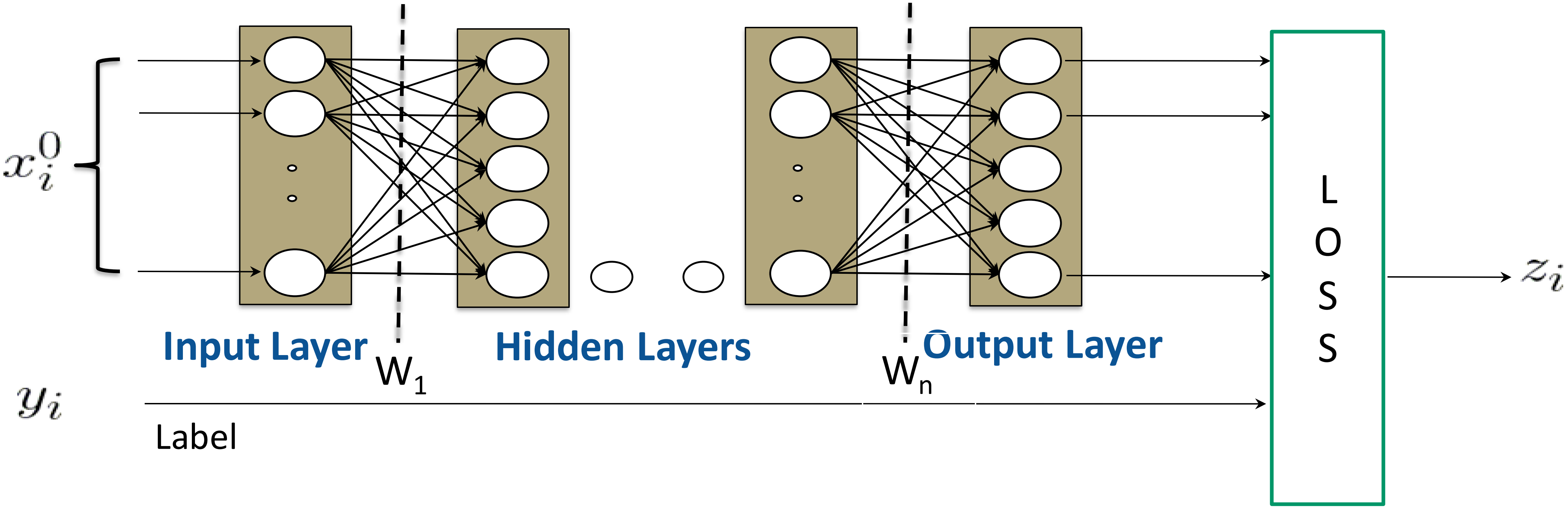


E.g. Sigmoid Activation Function



Multi Layer Perceptron

Loss or Objective

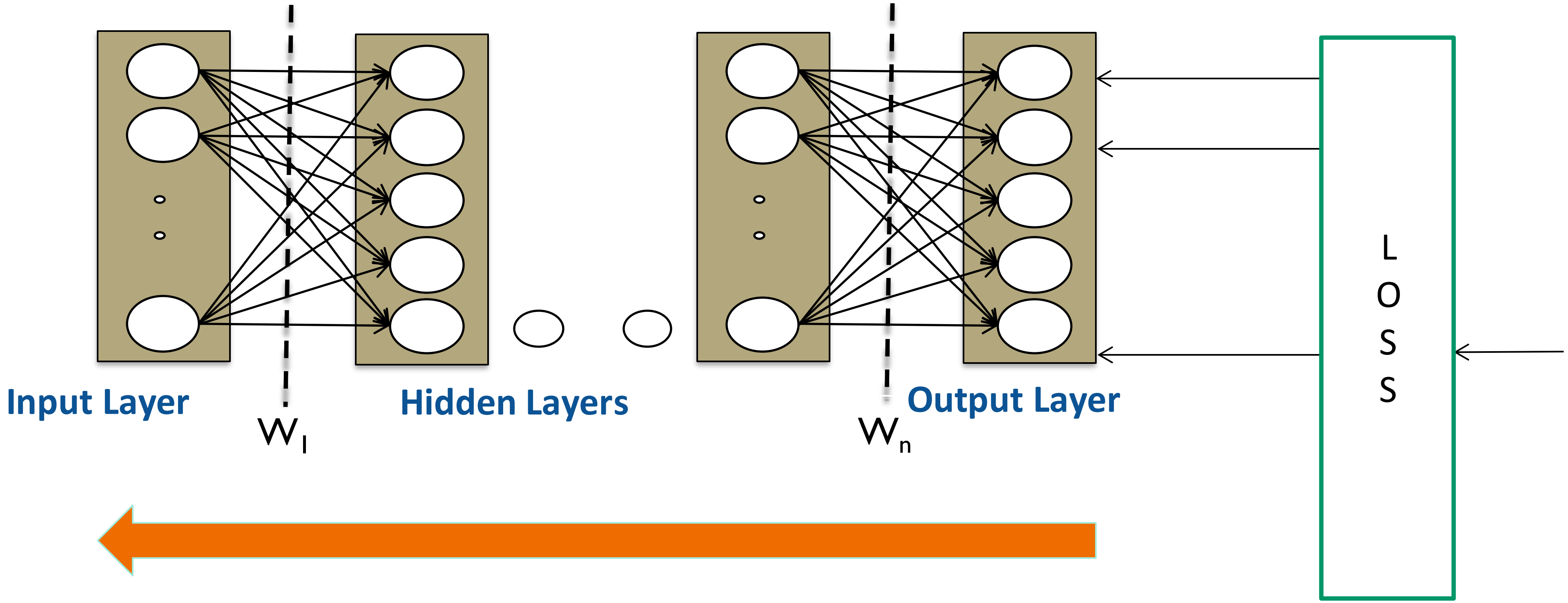


Objective: Find out the best parameters which will minimize the loss.

$$W^* = arg \min_W \sum_{i=1}^{N} L(x_i^n, y_i; W)$$ → Weight vector

$$z_i = \frac{1}{2} \| x_i^n - y_i \|_2^2$$ E.g. Squared Loss

Back Propagation

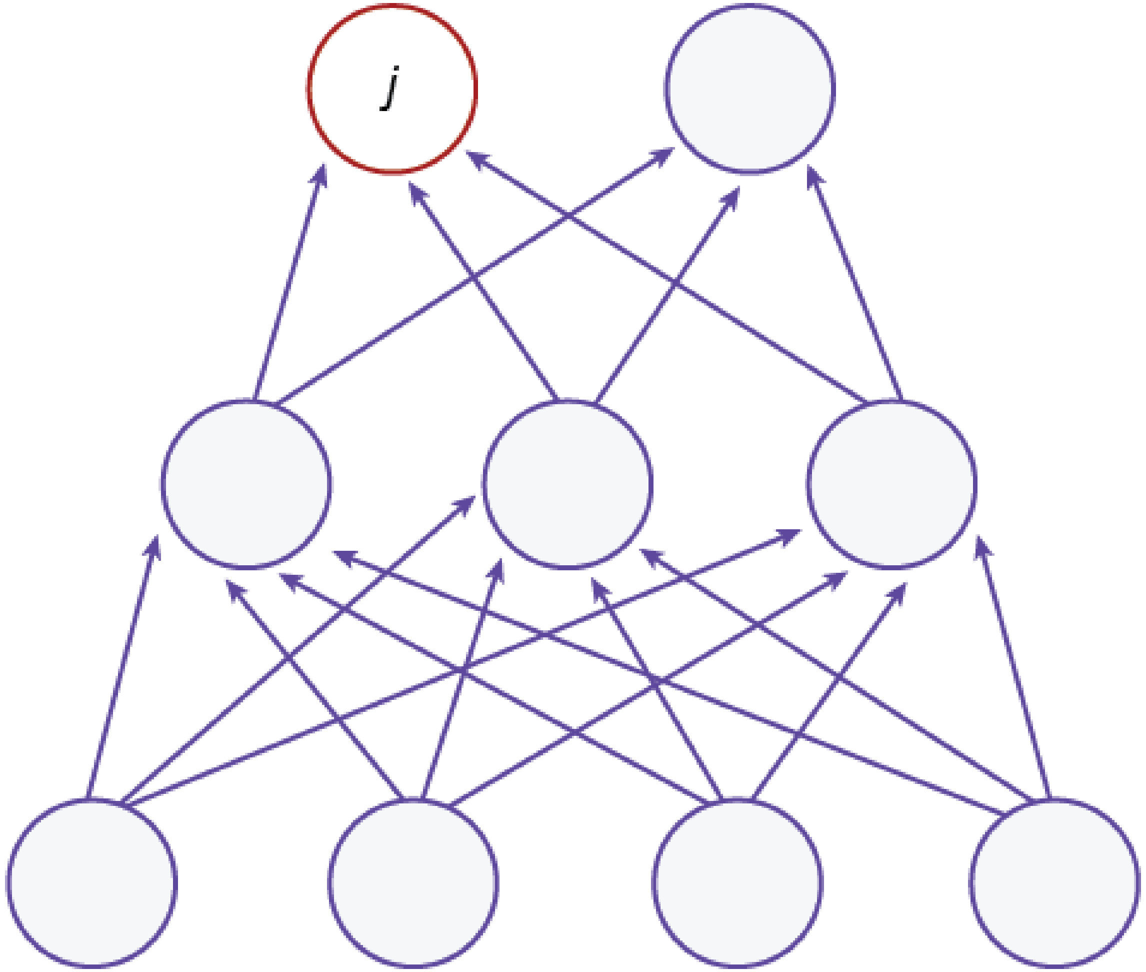


Solution: Iteratively update W along the direction where loss decreases.

Each layer's weights are updated based on the derivative of its output w.r.t. input and weights

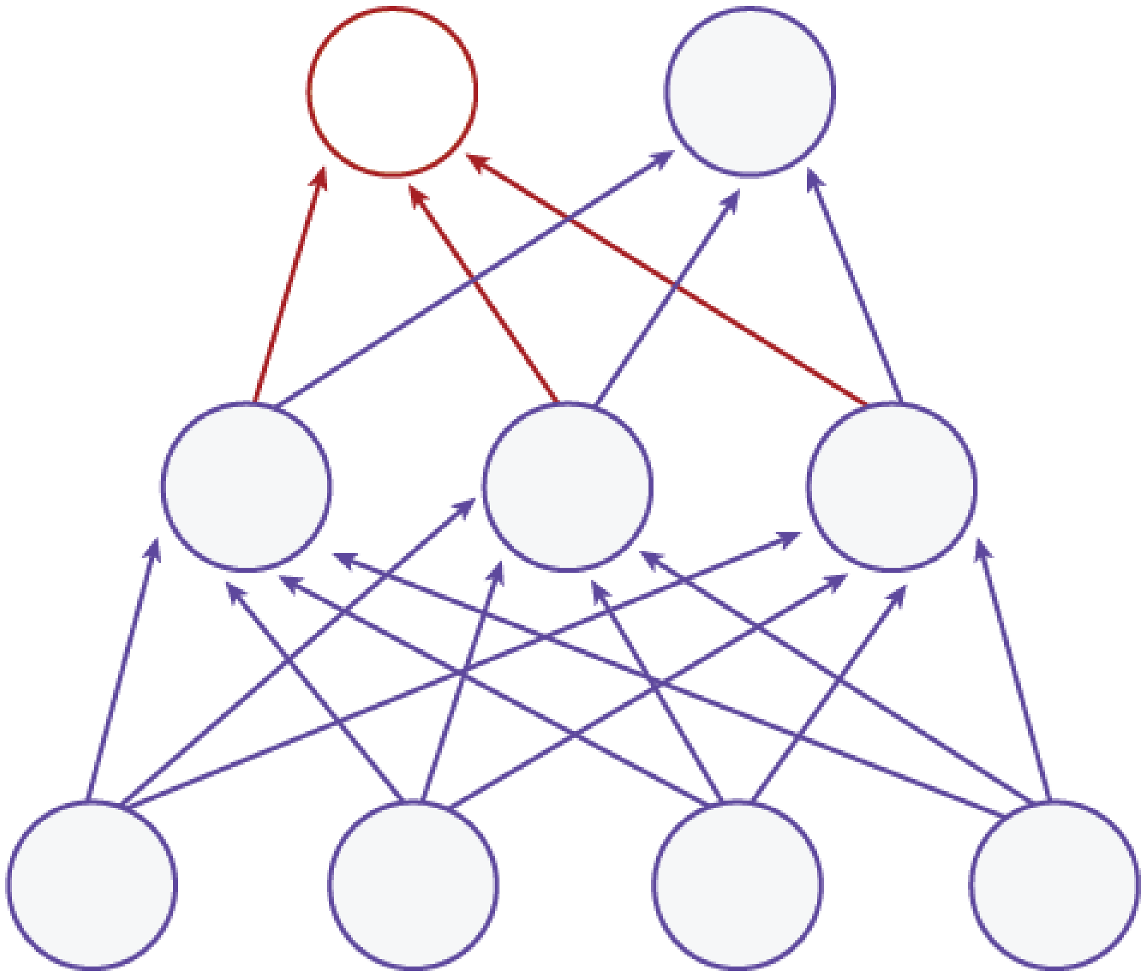
Error Back Propagation

- Calculate error



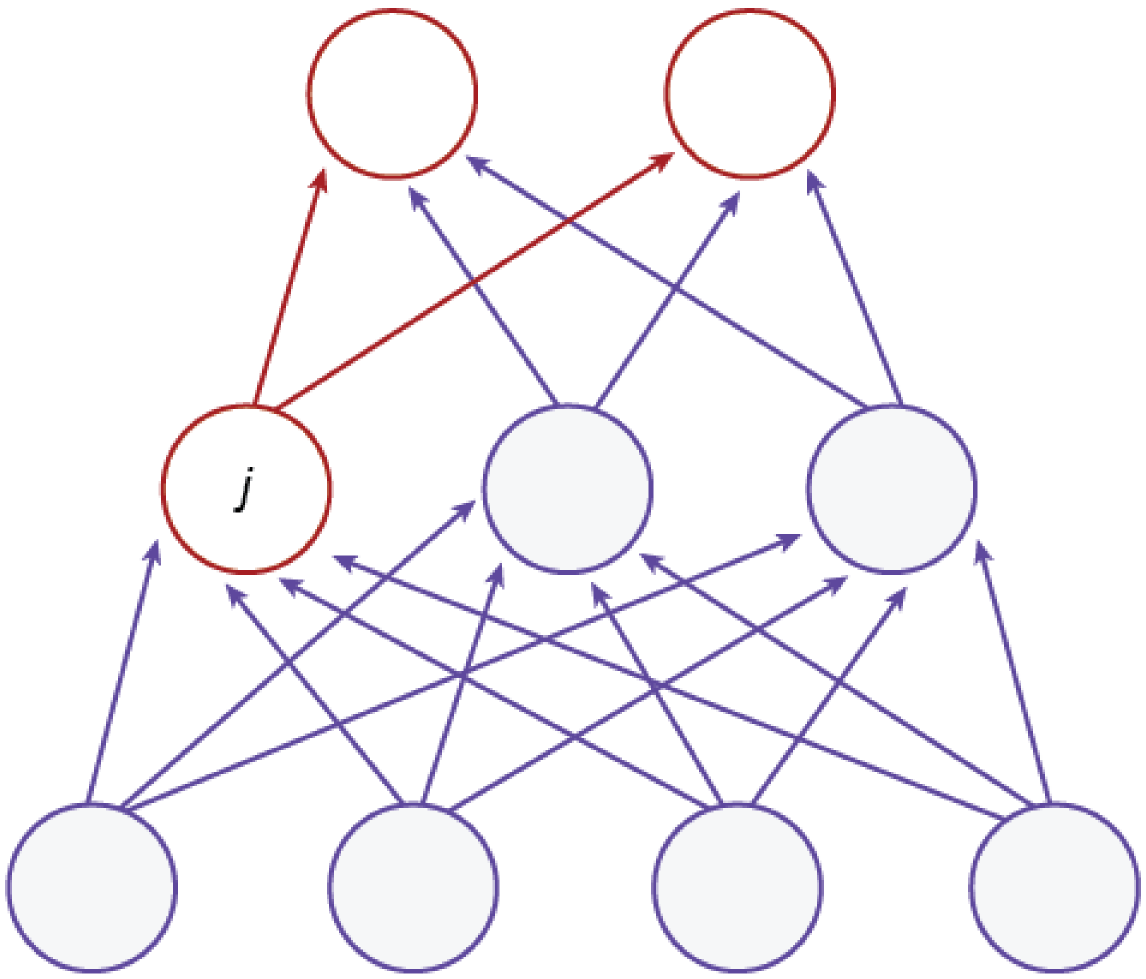
Error Back Propagation

- Determine updates for weights going to outputs.



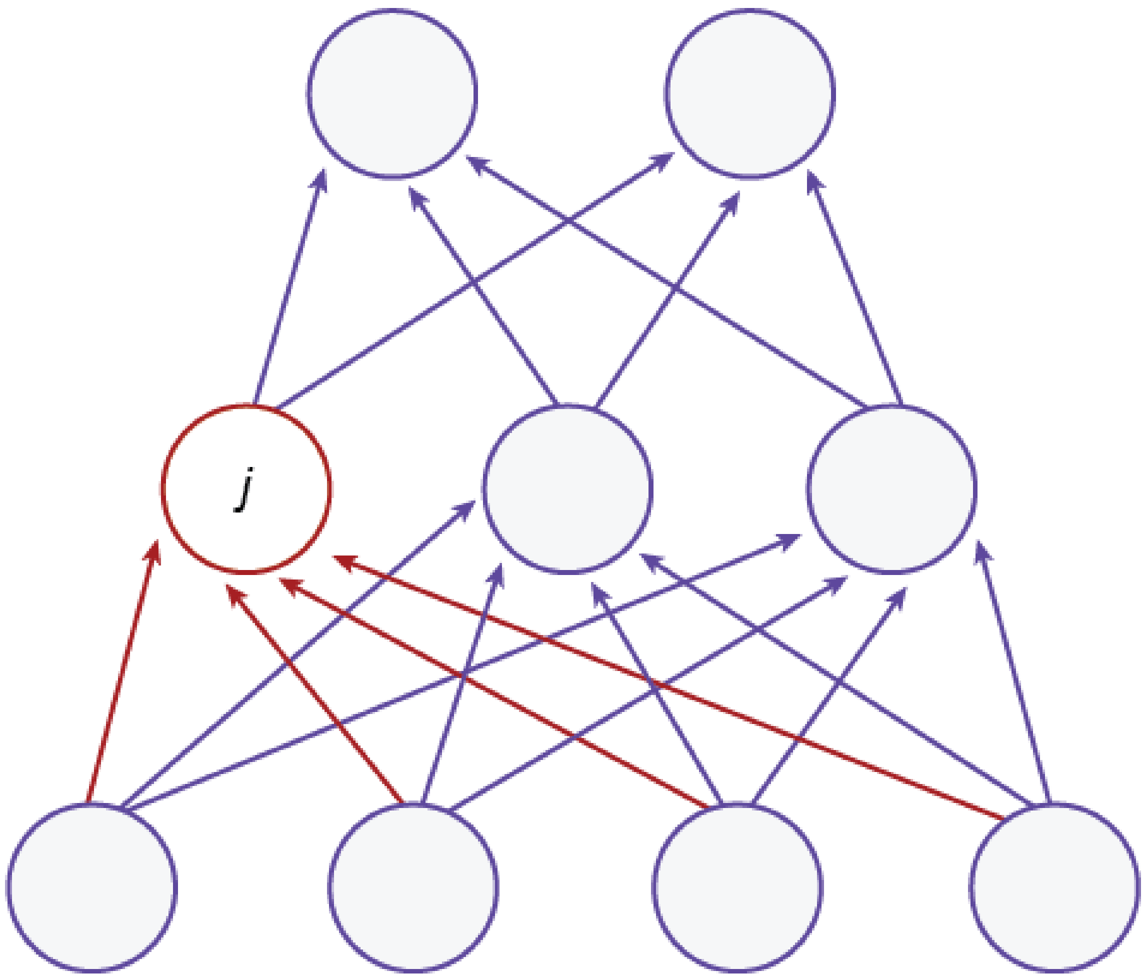
Error Back Propagation

- Calculate error for hidden units



Error Back Propagation

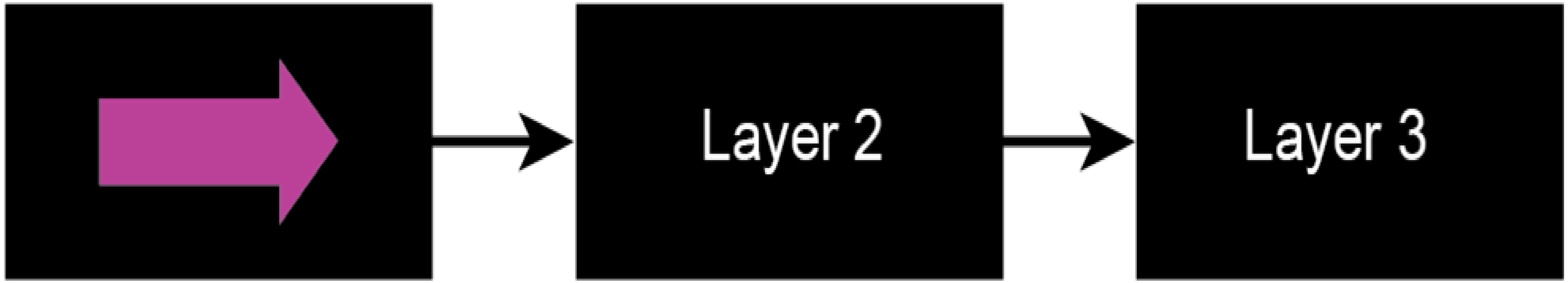
- Determine updates for weights to hidden units using hidden-unit errors.



BLANK SLIDE

Neural Network Training

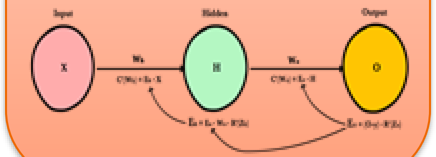
- Step 1: Compute loss on mini-batch [F-Pass]
- Step 2: Compute gradients w.r.t parameters[B-Pass]



$$\mathbf{W} \leftarrow \mathbf{W} - \eta \frac{\partial \mathbf{L}}{\partial \mathbf{W}}$$

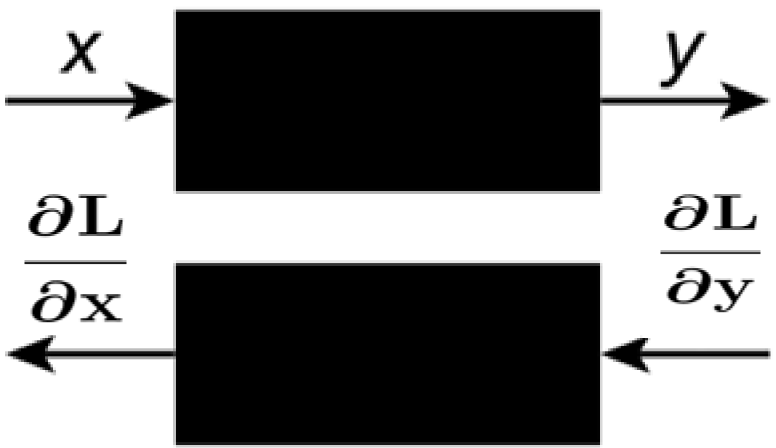
Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

- Back Propagation
  - ✓ Loss and Objective
  - Chain Rule



Chain Rule

Chain Rule



Given y(x) and $\frac{\partial L}{\partial y}$

What is $\frac{\partial L}{\partial x}$ *?*

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial x}$$

Chain Rule

Given $\frac{\partial y}{\partial x}$ **and** $\frac{\partial L}{\partial y}$ $\Rightarrow$ $$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial x}$$

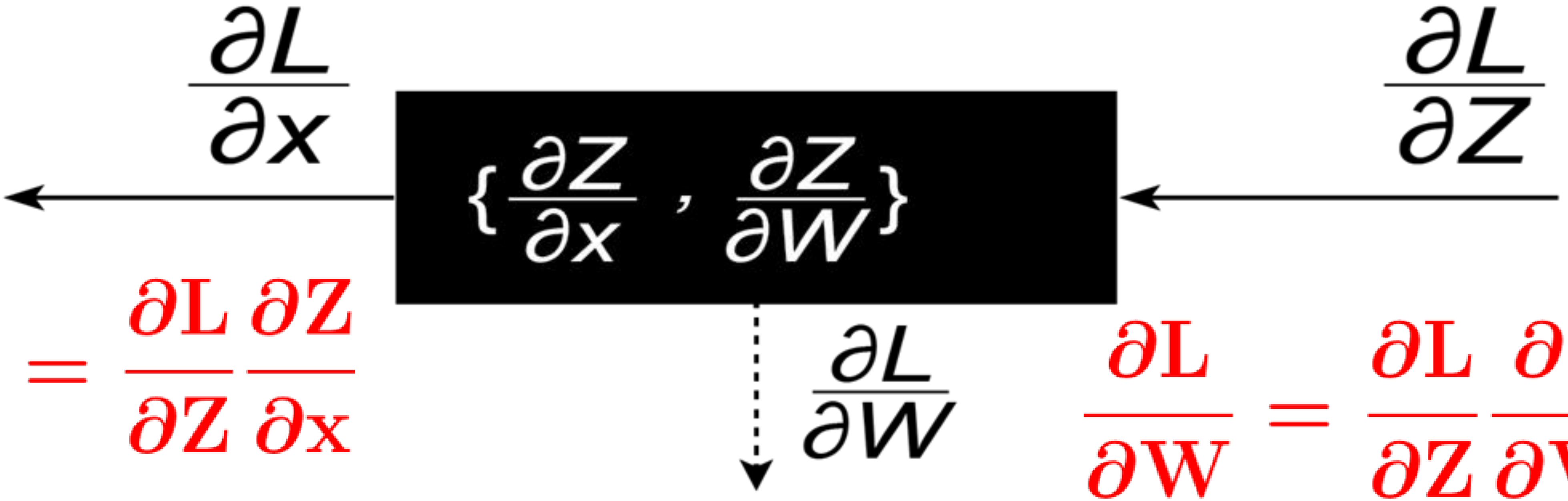
What is $\frac{\partial L}{\partial x}$ **?**

For each block/parameters, we only need to find $\frac{\partial y}{\partial x}$

Key Computation: Forward-Propagation

W is the parameter, say the weights within the box.

Key Computation: Backward-Propagation



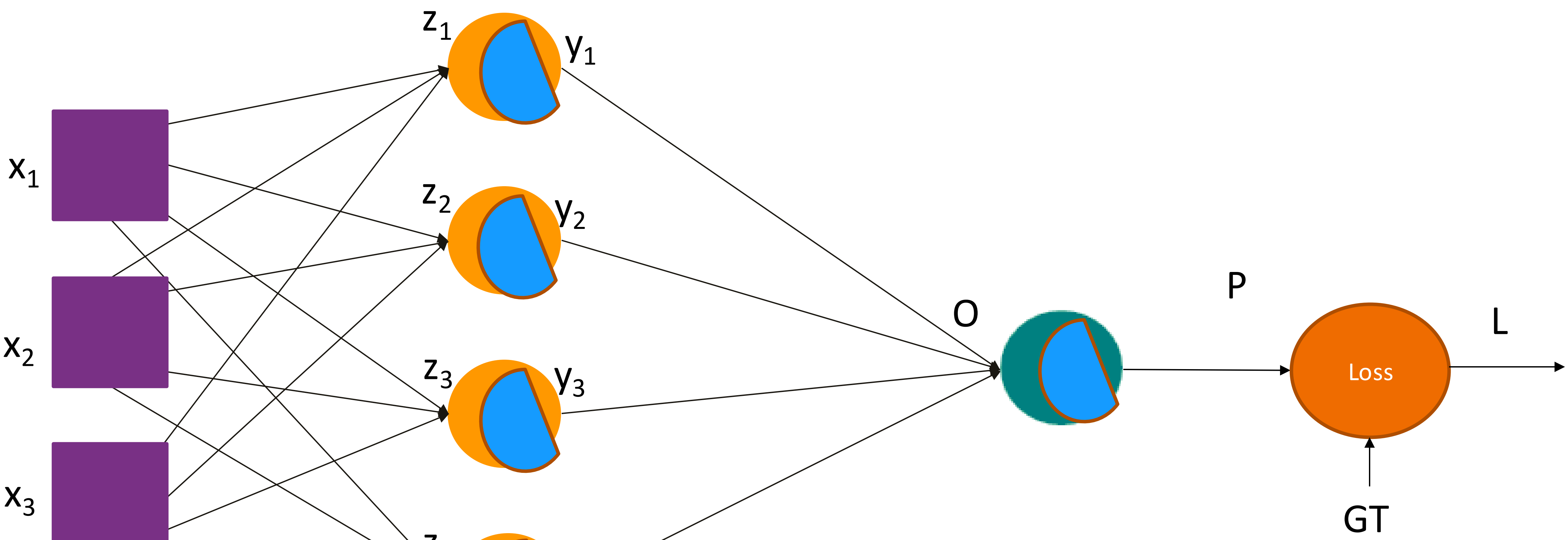
$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial Z}\frac{\partial Z}{\partial x}$$

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial Z}\frac{\partial Z}{\partial W}$$

$$W \leftarrow W - \eta\frac{\partial L}{\partial W}$$

BLANK SLIDE

Detailed View: Back Propagation



Ground Truth = GT Prediction = P

Loss = L = $(GT - P)^2$

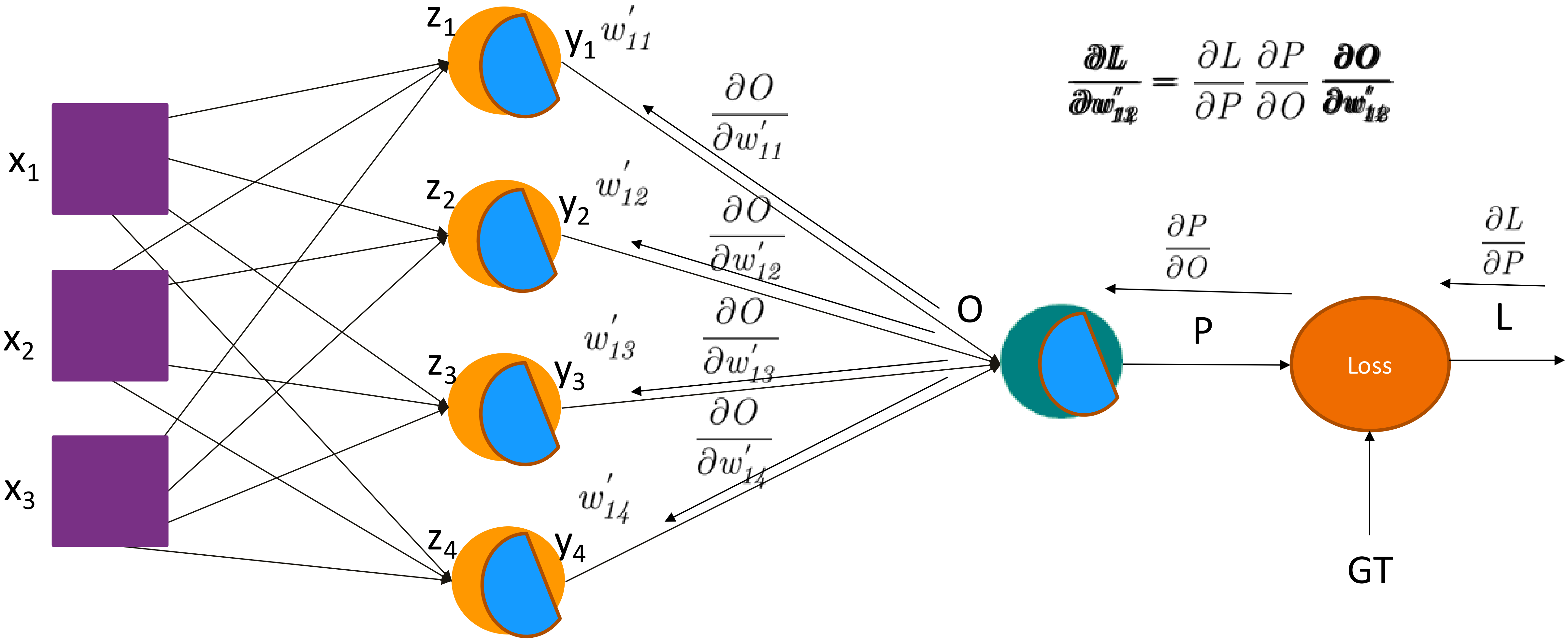
$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \phi(z_1) \\ \phi(z_2) \\ \phi(z_3) \\ \phi(z_4) \end{bmatrix}$$

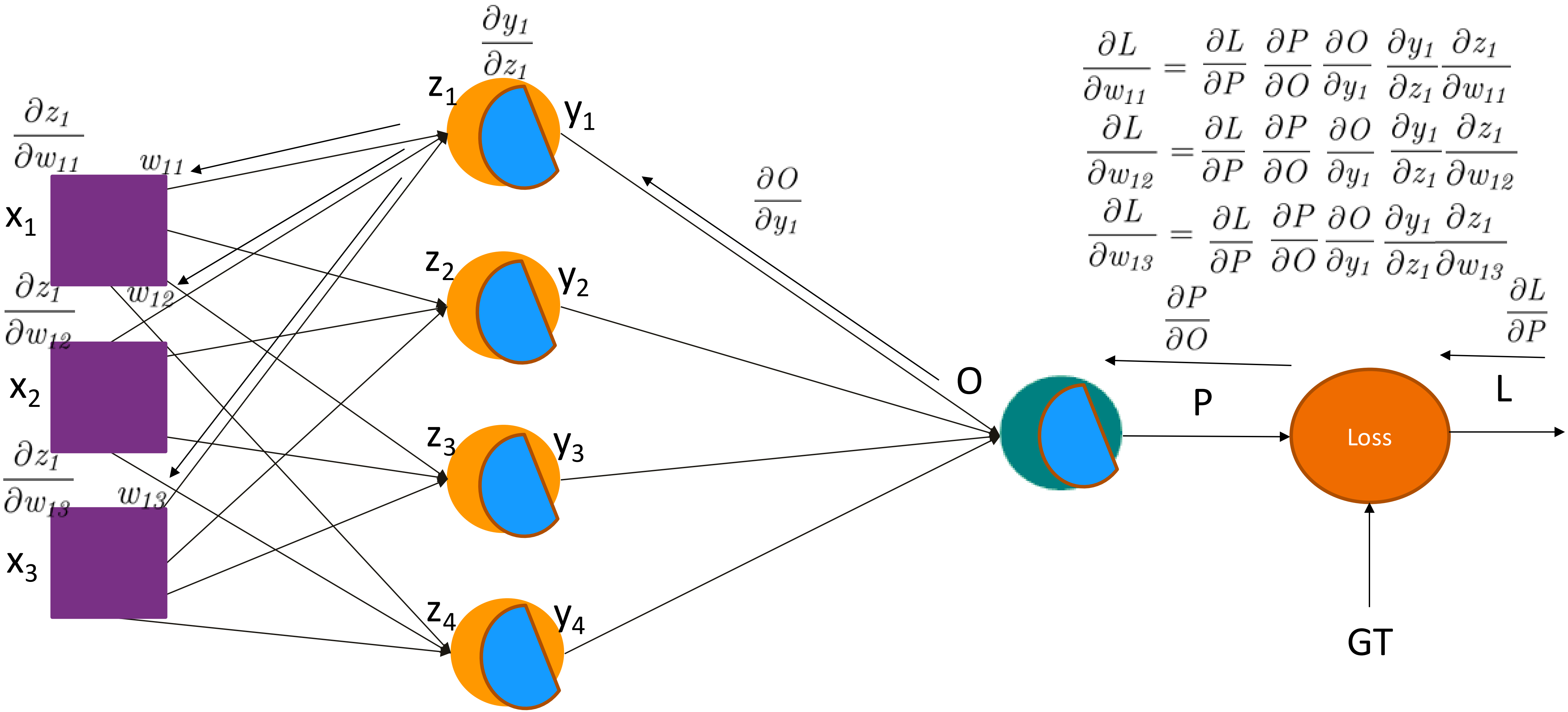
$$\mathrm{O} = \begin{bmatrix} w'_{11} & w'_{12} & w'_{13} & w'_{14} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$$\mathbf{P} = \phi(\mathbf{O})$$

Detailed View: Back Propagation



Detailed View: Back Propagation

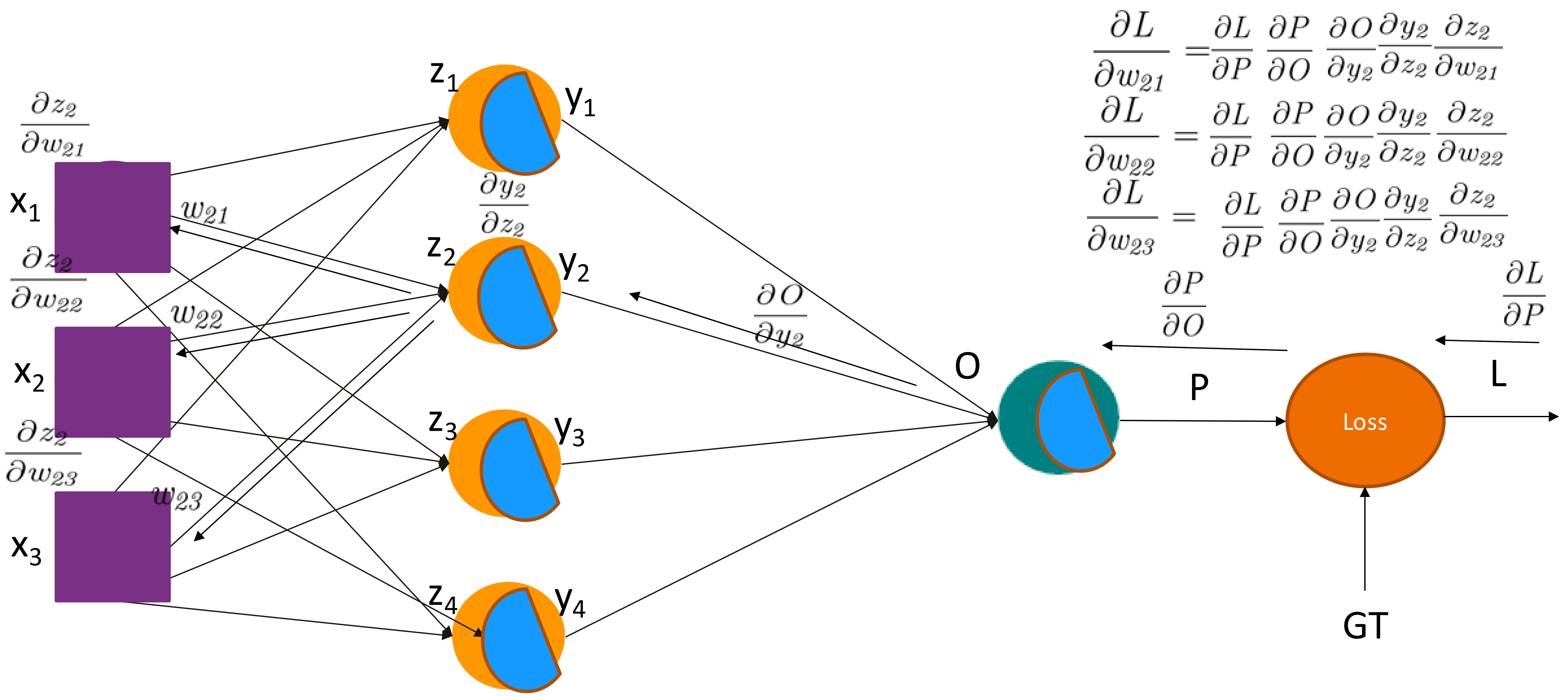


$$\frac{\partial L}{\partial w_{11}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_1}\frac{\partial y_1}{\partial z_1}\frac{\partial z_1}{\partial w_{11}}$$

$$\frac{\partial L}{\partial w_{12}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_1}\frac{\partial y_1}{\partial z_1}\frac{\partial z_1}{\partial w_{12}}$$

$$\frac{\partial L}{\partial w_{13}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_1}\frac{\partial y_1}{\partial z_1}\frac{\partial z_1}{\partial w_{13}}$$

Detailed View: Back Propagation

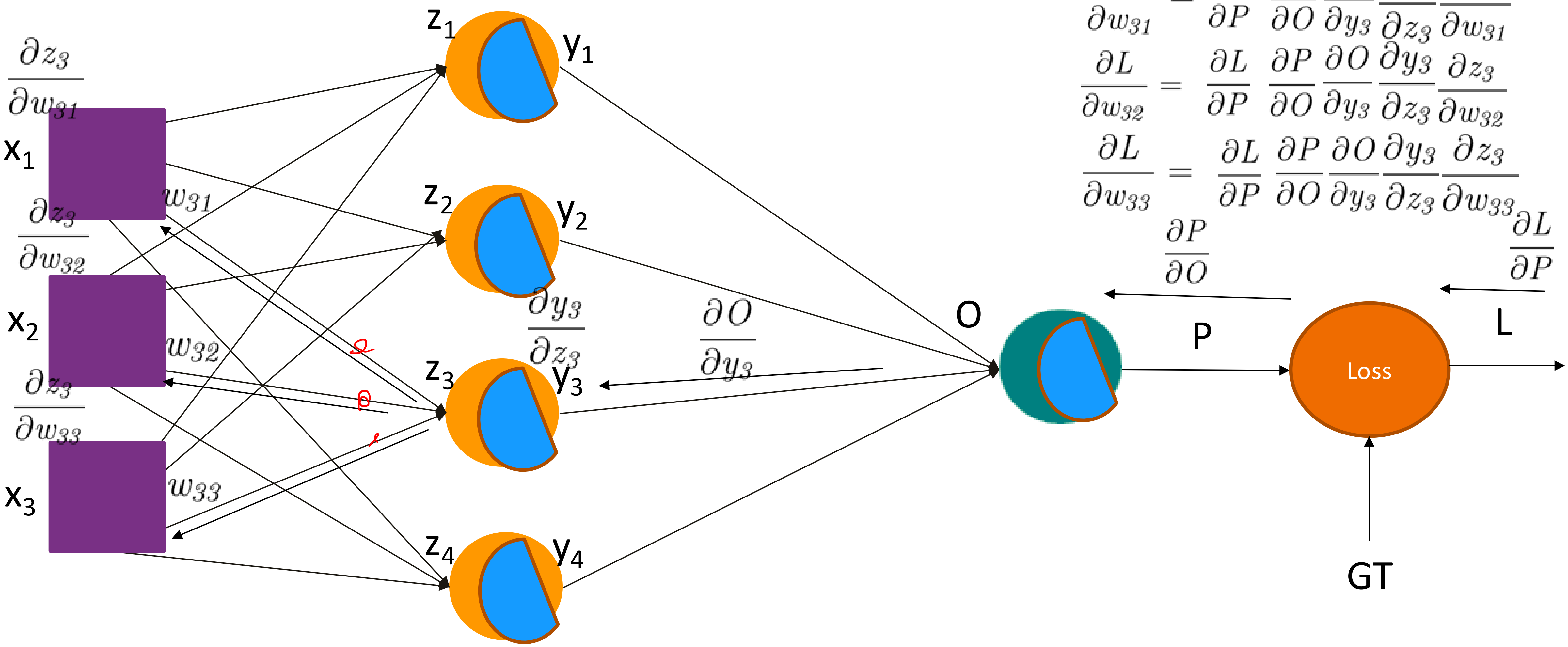


Detailed View: Back Propagation

$$\frac{\partial L}{\partial w_{31}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_3}\frac{\partial y_3}{\partial z_3}\frac{\partial z_3}{\partial w_{31}}$$

$$\frac{\partial L}{\partial w_{32}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_3}\frac{\partial y_3}{\partial z_3}\frac{\partial z_3}{\partial w_{32}}$$

$$\frac{\partial L}{\partial w_{33}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_3}\frac{\partial y_3}{\partial z_3}\frac{\partial z_3}{\partial w_{33}}$$

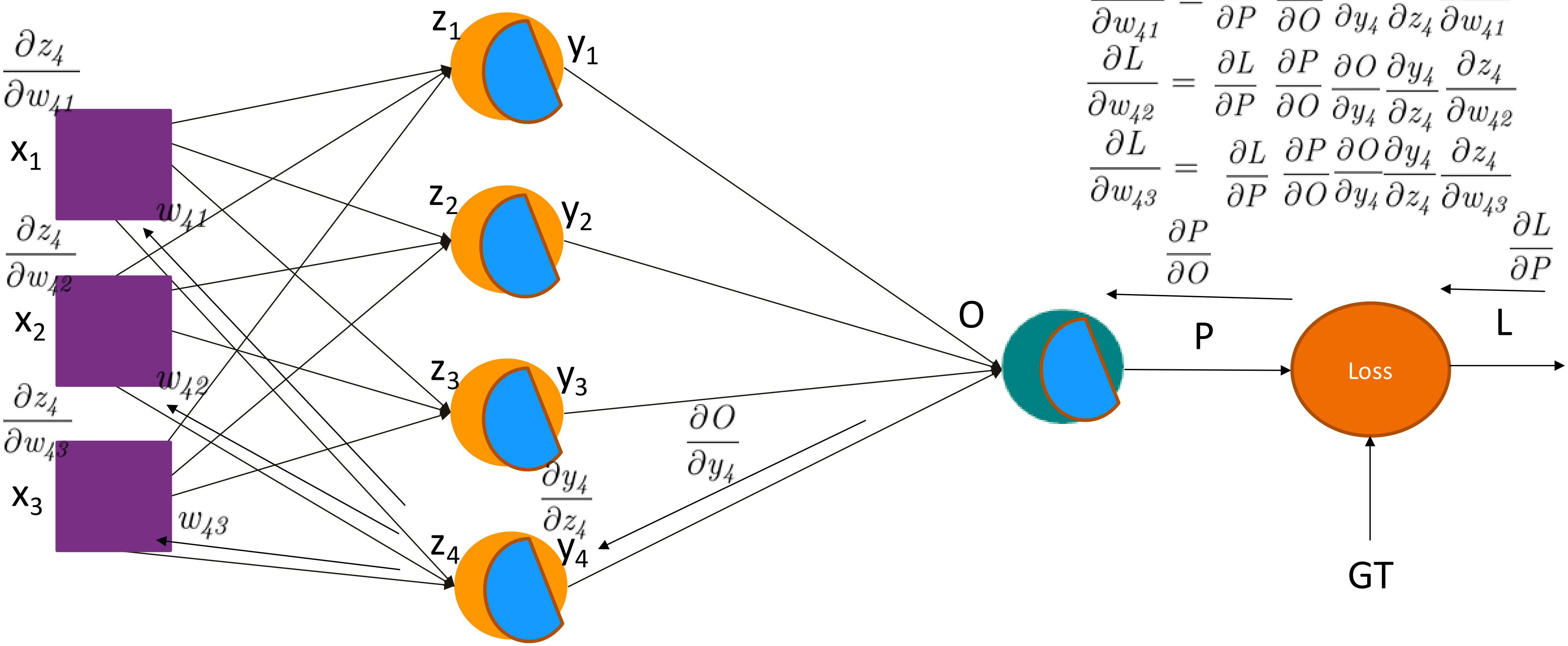


Detailed View: Back Propagation

$$\frac{\partial L}{\partial w_{41}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_4}\frac{\partial y_4}{\partial z_4}\frac{\partial z_4}{\partial w_{41}}$$

$$\frac{\partial L}{\partial w_{42}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_4}\frac{\partial y_4}{\partial z_4}\frac{\partial z_4}{\partial w_{42}}$$

$$\frac{\partial L}{\partial w_{43}} = \frac{\partial L}{\partial P}\frac{\partial P}{\partial O}\frac{\partial O}{\partial y_4}\frac{\partial y_4}{\partial z_4}\frac{\partial z_4}{\partial w_{43}}$$



BLANK SLIDE

Summary

- Step 0:
  - Initialize the Network (MLP), weights
- Step 1:
  - Do forward pass for a batch of randomly selected samples.
  - Predict outputs with the existing weights.
- Step 2:
  - Compute Loss for the set of samples.

Summary

- Step 3:
  - Update all the weights using gradient descent.

$$\mathbf{W} \leftarrow \mathbf{W} - \eta \frac{\partial \mathbf{L}}{\partial \mathbf{W}}$$

- Step 4:
  - Repeat all steps till the Loss is less than a threshold.

BLANK SLIDE

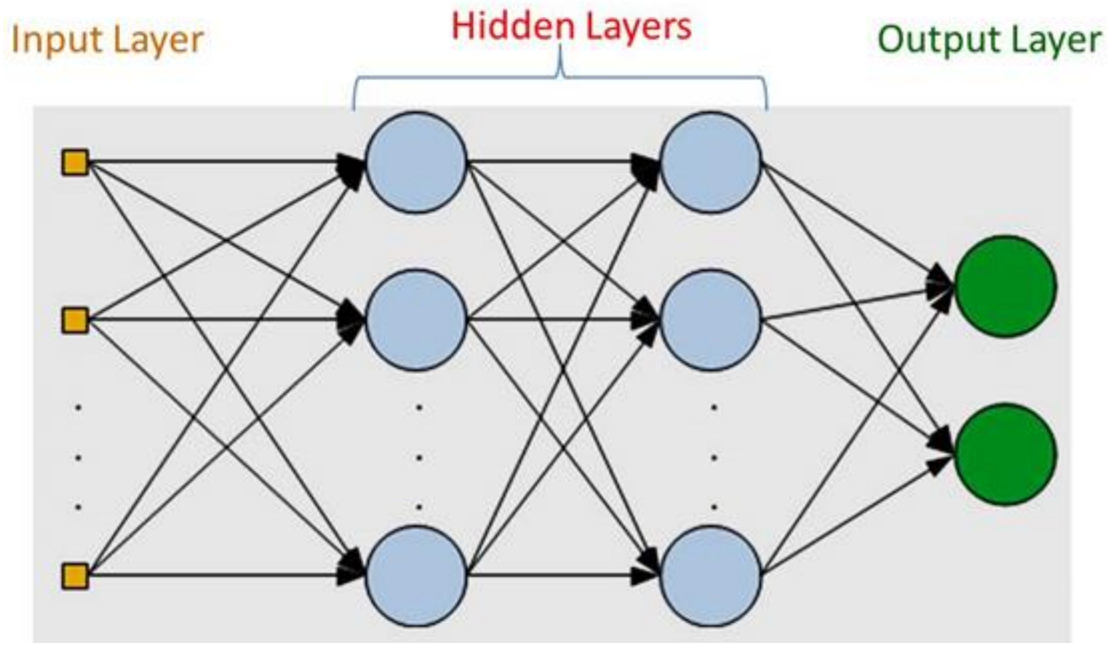
Back Propagation for MLP

- Two computational blocks/steps

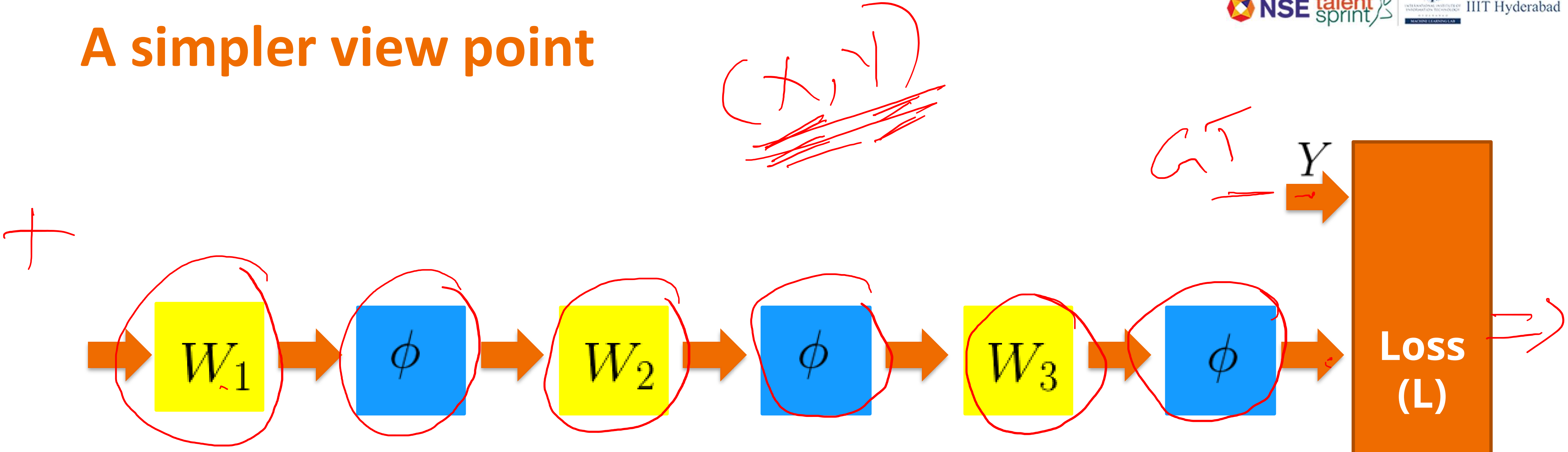
$$y = Wx$$

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

- In either case we can compute $\frac{\partial y}{\partial x}$ easily.



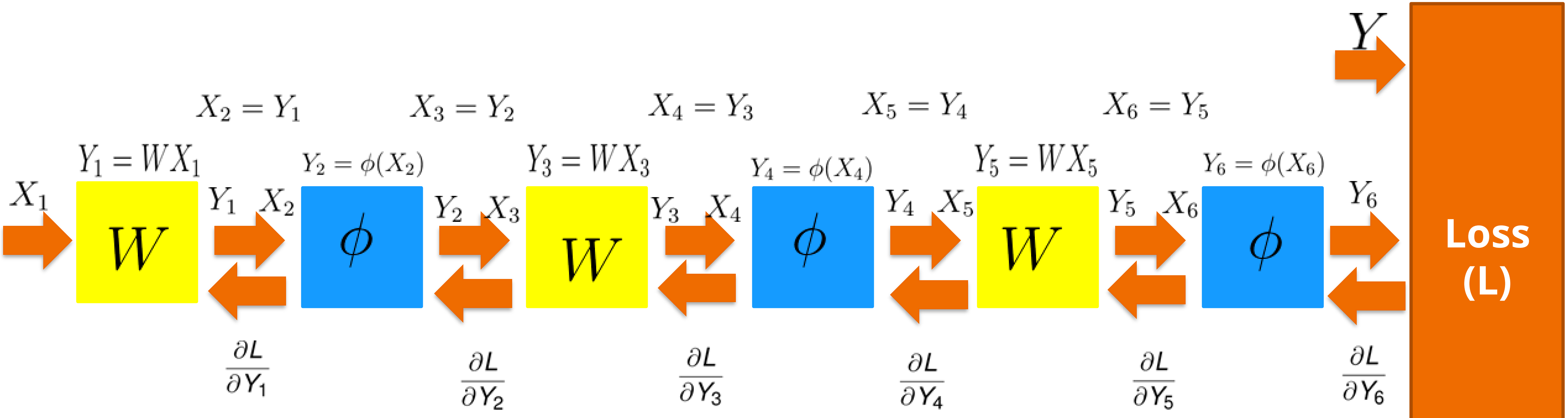
A simpler view point



Blocks with Learnable parameters Matrix Multiplication

Nonlinear functions (often non learnable)

Back Propagation (X,Y): Propagation



$$\frac{\partial L}{\partial Y_1} = \frac{\partial L}{\partial X_2} = \frac{\partial L}{\partial Y_2} \cdot \frac{\partial Y_2}{\partial X_1}$$

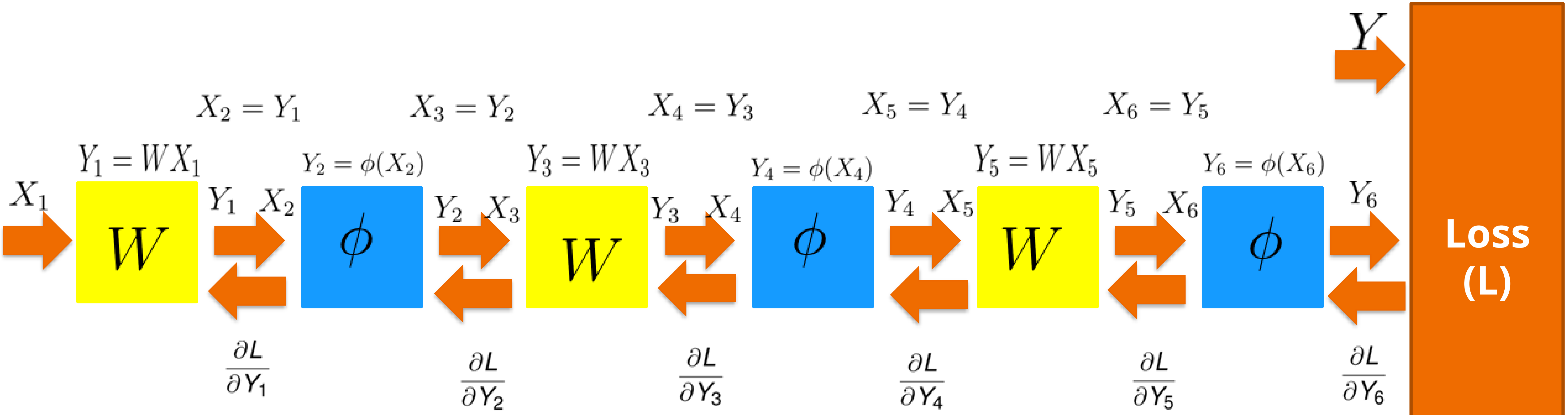
$$\frac{\partial L}{\partial Y_3} = \frac{\partial L}{\partial X_4} = \frac{\partial L}{\partial Y_4} \cdot \frac{\partial Y_4}{\partial X_4}$$

$$\frac{\partial L}{\partial Y_5} = \frac{\partial L}{\partial X_6} = \frac{\partial L}{\partial Y_6} \cdot \frac{\partial Y_6}{\partial X_6}$$

$$\frac{\partial L}{\partial Y_4} = \frac{\partial L}{\partial X_5} = \frac{\partial L}{\partial Y_5} \cdot \frac{\partial Y_5}{\partial X_5}$$

$$\frac{\partial L}{\partial Y_2} = \frac{\partial L}{\partial X_3} = \frac{\partial L}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial X_3}$$

Back Propagation (X,Y): Also Learning



$$\frac{\partial L}{\partial Y_1} = \frac{\partial L}{\partial X_2} = \frac{\partial L}{\partial Y_2} \cdot \frac{\partial Y_2}{\partial X_2}$$

$$\frac{\partial L}{\partial Y_3} = \frac{\partial L}{\partial X_4} = \frac{\partial L}{\partial Y_4} \cdot \frac{\partial Y_4}{\partial X_4}$$

$$\frac{\partial L}{\partial Y_5} = \frac{\partial L}{\partial X_6} = \frac{\partial L}{\partial Y_6} \cdot \frac{\partial Y_6}{\partial X_6}$$

$$\frac{\partial L}{\partial Y_2} = \frac{\partial L}{\partial X_3} = \frac{\partial L}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial X_3}$$

$$\frac{\partial L}{\partial Y_4} = \frac{\partial L}{\partial X_5} = \frac{\partial L}{\partial Y_5} \cdot \frac{\partial Y_5}{\partial X_5}$$

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial W}$$

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial Y_3} \cdot \frac{\partial Y_3}{\partial W}$$

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial Y_5} \cdot \frac{\partial Y_5}{\partial W}$$

$$W^{n+1} = W^n - \eta \frac{dL}{dW}$$

BLANK SLIDE

Back Propagation

$$(1) \rightarrow \frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial x} \quad (2) \rightarrow \frac{\partial L}{\partial W} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial W} \quad (3) \rightarrow W^{n+1} = W^{n}\ \eta \frac{\partial L}{\partial W}$$

- Let there be N stages. For a computational block ℓ,
  - Compute $\frac{\partial L}{\partial x}$ using equation 1
  - If the block as a learnable parameters **W** then,
    - Compute $\frac{\partial L}{\partial W}$ using equation 2
    - Update the parameters using equation 3
  - Set the $\frac{\partial L}{\partial x}$ of stage ℓ as $\frac{\partial L}{\partial y}$ of stage ℓ - 1, and repeat the steps 1-3, until we reach the first block.

Summary

- Step 0:
  - Initialize the Network (MLP), weights
- Step 1:
  - Do forward pass for a batch of randomly selected samples.
  - Predict outputs with the existing weights.
- Step 2:
  - Compute Loss for the set of samples.

Summary

- Step 3:
  - Update all the weights using gradient descent.

$$\mathbf{W} \leftarrow \mathbf{W} - \eta \frac{\partial \mathbf{L}}{\partial \mathbf{W}}$$

- Step 4:
  - Repeat all steps till the Loss is less than a threshold.

Questions?

Comments on BP

- A Non-Convex Optimization
  - Worry of getting trapped in Local Minima
- Why limit to simple gradient methods?
  - Why not second order methods?
- How to make the solution reliable?
  - Repeatable?
- Many refinements
  - Eg. Momentum
- More Later

Many Practical Challenges

- Vanishing Gradients in Deep Networks
- Slow Convergence
- Bad ``Local Minimas''
- How to initialize the network?
  - Random or "smart" initializations
- Better update rule
  - Faster and reliable convergence
- Termination Criteria
  - When do we stop?

BLANK SLIDE

Summary

Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

- ✓ Back Propagation
  - ✓ Loss and Objective
  - ✓ Chain Rule
  - ✓ Momentum
- ✓ Vanishing Gradients

Thanks!!

Questions?