09

Forward & Backward Propagation

Back & Forth, Back & Forth, Back & Forth…

9.1
Mathematical Functions in DL.

Importance of understanding the math

- Deep Learning is still just a mathematical model
- Understanding how these math concepts are crucial to understand important algorithms
- This enabled engineers to optimize, adapt, and customize different algorithms of neural networks to different problems

Mathematical Functions Essential for ML Engineers

- **Derivatives & Partial Derivatives**: Fundamental for understanding how each parameter affects the overall loss.
- **Chain Rule**: Essential for computing derivatives of composite functions in backpropagation.
- **Sigmoid & ReLU Functions**: Examples of activation functions that introduce non-linearity to the model.
- **Mean Squared Error & Cross-Entropy**: Examples of loss functions for regression and classification tasks, respectively.

Derivatives & Partial Derivative

- Derivatives measure the rate at which a function changes, and partial derivatives do so for multivariable functions
- They are foundational for gradient descent and optimizing loss functions, showing how each parameter (weight/bias) affects the overall loss
- **Application**: Used to update the weights and biases during backpropagation
- Learn more: [Full derivative vs partial derivative]

Chain Rule

- The chain rule is a fundamental principle in calculus used to compute the derivative of composite functions
- Essential for backpropagation, allowing the computation of gradients through layers in a neural network
- **Application**: Applied when calculating the gradient of the loss function with respect to the weights and biases
- Learn more: Chain Rule for Finding Derivatives

Sigmoid & ReLU Functions

- Sigmoid compresses the output between 0 and 1, while ReLU (Rectified Linear Unit) outputs the input directly if positive; otherwise, it will output zero
- Introduce non-linearity to the model, allowing the neural network to learn from the error and make adjustments
- **Application**: Utilized as activation functions in the hidden layers of neural networks
- Learn more: Activation Functions in Neural Networks

Mean Squared Error & Cross-Entropy

- Mean Squared Error measures the average of the squares of the errors for regression, while Cross-Entropy measures the dissimilarity between the predicted probability distribution and the true distribution for classification
- Essential for evaluating the performance of a model and for optimizing the model during training
- **Application**: Used as loss functions to compute the error between predicted and true labels
- Learn more: Why do we need Cross Entropy Loss?

9.2
Gradients.

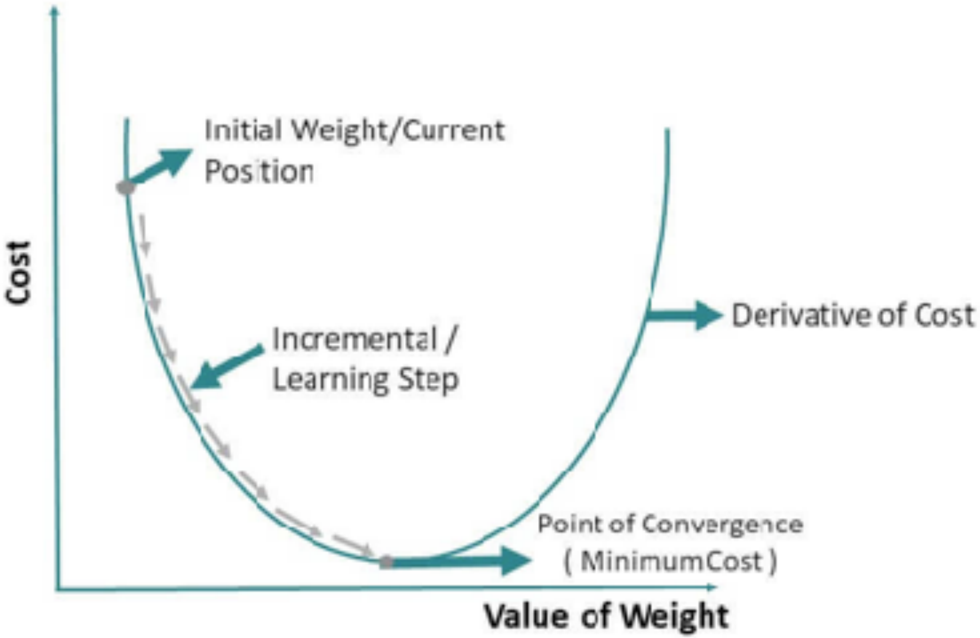
Understanding Gradient in DL

- Gradient is a vector containing all the partial derivative information of a function
- It is essential for optimizing a neural network

The Pivotal Role of Gradients in Deep Learning

- **Driving Optimization**: Gradients drive the optimization of neural networks by indicating the direction and rate of the fastest increase of a function.
- **Guiding Learning**: They guide the learning process by adjusting the model parameters (weights and biases) to minimize the loss function.
- **Facilitating Generalization**: By optimizing the model on the training data, gradients facilitate the generalization of the model to unseen data.
- **Enhancing Model Performance**: Proper utilization of gradients is essential for enhancing the predictive performance of deep learning models.

Gradients Leading to Optimization



What Engineers Need to Know about Gradients

- **Understanding the Basics:** Engineers should have a firm grasp of basic calculus, particularly derivatives and partial derivatives, as they form the components of gradients
- **Computing Gradients Efficiently:** Knowledge of efficient computation techniques like backpropagation is vital for handling complex neural network architectures
- **Managing Vanishing & Exploding Gradients:** Familiarity with issues such as vanishing and exploding gradients and strategies to mitigate them are critical

What Engineers Need to Know about Gradients

- **Applying Adaptive Learning Rates:** Engineers should know how to implement adaptive learning rate algorithms like AdaGrad, RMSProp, and Adam for efficient optimization.
- **Experimenting and Tuning:** Practical experience in experimenting with different optimization strategies and tuning hyperparameters is essential for model optimization.

9.2
Forward Propagation.

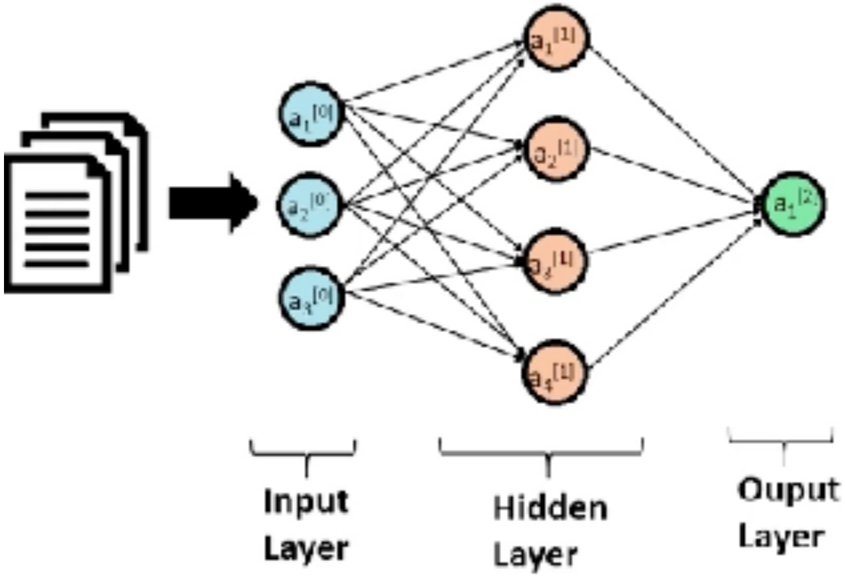
What is Forward Propagation?

- Forward Propagation involves passing the input data through each layer of the neural network to compute the output
- It is the initial step in the learning process
- It determines the initial predictions of the network

Mechanics of Forward Propagation

- **Weighted Sum**: This is calculated at each neuron based on weights and connections to the previous set of neurons
- **Activation Function**: Applied to the weighted sum to introduce non-linearity
- Non-linearity creates a boundary between each of the layers so that they cannot be simplified

Forward Propagation



$$a_1^{[1]} = \text{activation\_function}(W_{11}^{[1]} * a_1^{[0]} + W_{12}^{[1]} * a_2^{[0]} + W_{13}^{[1]} * a_3^{[1]} + B1)$$

$$a_2^{[1]} = \text{activation\_function}(W_{21}^{[1]} * a_1^{[1]} + W_{22}^{[1]} * a_2^{[1]} + W_{23}^{[1]} * a_3^{[1]} + B1)$$

9.3
Loss Function.

Introduction to Loss Functions

- Loss Functions measure the disparity between the actual and predicted outputs
- They guide the optimization of neural networks
- Loss functions differ for every different task of
  - MSE for regression
  - Cross-Entropy for classification

Importance of Loss Functions

- **Tuning Weights & Biases**: Directly influences the adjustment of parameters
- **Model Performance**: A lower loss indicates better performance

Types of Loss Functions

- Mean Squared Error (MSE) Loss
  - Primarily used for regression problems and model fitting, where the goal is to minimize the average of the squares of the errors between predicted and true values.
- Cross-Entropy Loss
  - Employed in binary and multiclass classification tasks, where the objective is to minimize the dissimilarity between the predicted probability distribution and the true distribution.

Types of Loss Functions

- Categorical Cross-Entropy Loss
  - Used in multiclass classification problems, particularly in neural networks, as a generalization of cross-entropy loss for scenarios with more than two classes
- Log-cosh Loss
  - Serves as an alternative to MSE for regression problems, providing a smoother error gradient by measuring the logarithm of the hyperbolic cosine of the prediction error

9.4
Backpropagation.

Introduction to Loss Functions

- Backpropagation refers to the backward pass of a neural network
- During backpropagation the weights and biases are adjusted to minimize loss
- It computes gradients of loss functions and updates the parameters

Introduction to Loss Functions

- Backpropagation refers to the backward pass of a neural network
- During backpropagation the weights and biases are adjusted to minimize loss
- It computes gradients of loss functions and updates the parameters

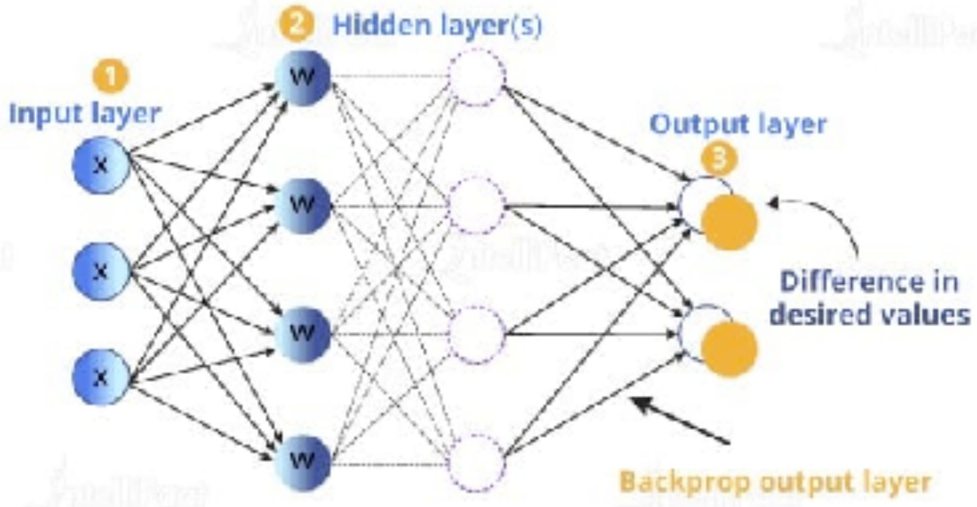
Essential Knowledge on Backpropagation for Engineers

- **Learning Rate:** Understanding the impact of the learning rate on convergence is crucial, as it influences the step size in the weight updates.
- **Weight Updates**: Familiarity with how weights are adjusted based on computed gradients is essential for tuning and optimizing neural networks.

Essential Knowledge on Backpropagation for Engineers

- **Vanishing & Exploding Gradients:** Awareness of issues such as vanishing and exploding gradients, along with mitigation strategies like normalized initialization and gradient clipping.
- **Computational Efficiency:** Knowledge of efficient computation techniques and libraries/frameworks that implement backpropagation is beneficial.

Back Propagation



9.5
Building a **custom model**.

Building Layers in PyTorch

- In **PyTorch**, neural networks are constructed using layers.
- **Layers** are building blocks that process data as it flows through the network.
- Each layer type serves a specific purpose in **feature extraction** and **transformation**.

Types of Layers in PyTorch

- **Linear layers**: Transform input by linearly combining input features.
- **Convolutional layers**: Capture spatial hierarchies in images.
- **Recurrent layers**: Process time series and sequential data.
- **Activation layers**: Introduce non-linearities into the network (e.g., ReLU, Sigmoid).

Training loop essentials

- **Forward pass:** Compute the predicted outputs.
- **Compute loss:** Measure the error.
- **Backward pass:** Compute gradients.
- **Update weights:** Apply optimization algorithm.

Concepts to keep in mind

- **Epochs:** An epoch represents one complete pass through the entire training dataset.
- **Batch size:** Number of samples processed before updating the model.
- **Iterations:** Number of batches needed to complete one epoch.
- **Learning rate:** Step size at each iteration while moving toward a minimum of the loss function.

END.