Johns Hopkins Engineering

Applied Machine Learning for Mechanical Engineers

Machine Learning Fundamentals, Part 1, D

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Learning Algorithms

- By the end of this lecture you will be able to:
  - Describe gradient descent
  - Describe backpropagation
  - Describe universal approximation theorem

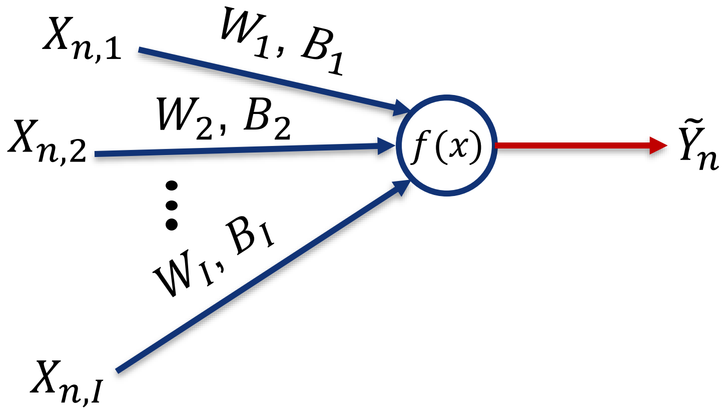
Learning Algorithms

- Loss functions
  - Mean squared error
  - Mean absolute error
  - Cross entropy

Learning Algorithms

- Loss functions

  - Loss function in multi-layer neural network; extremely complicated!



Learning Algorithms

- Loss functions
  - Loss function in perceptron

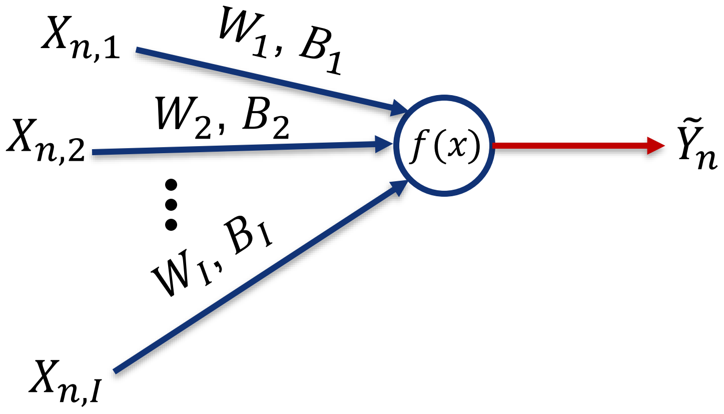
$$l(p) = \frac{1}{2N}\sum_{n=1}^{N}\left\|Y_n - \tilde{Y}_n\right\|^2$$

Estimated

Actual/real

where

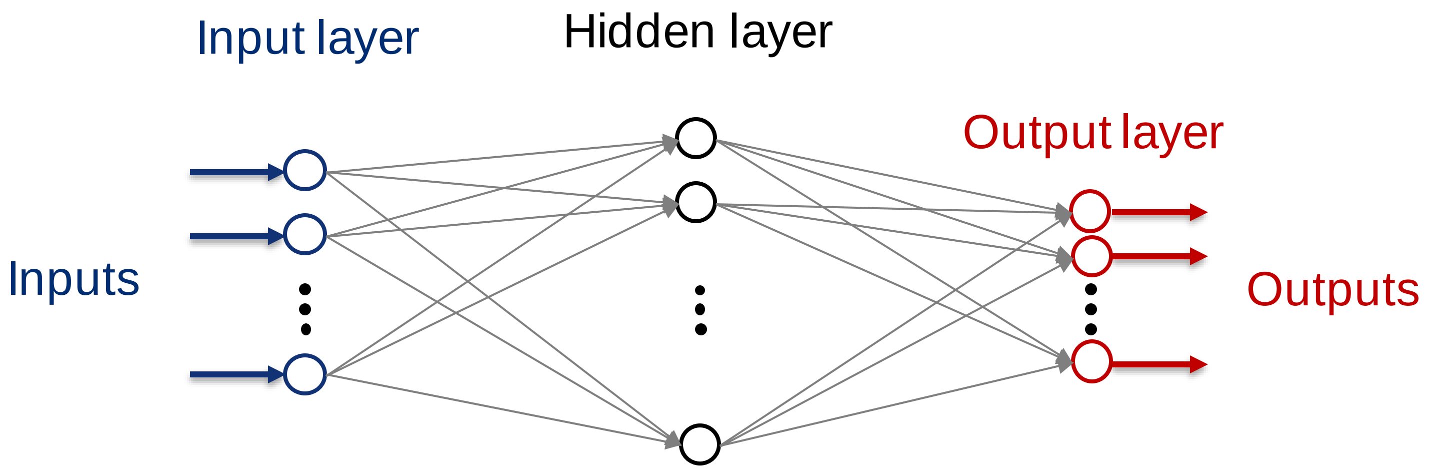
$$\tilde{Y}_n = f(x) = f\left(\sum_{i=1}^{I} X_{n,i}W_i + B_i\right)$$



p includes I weights and I biases to be optimized

Learning Algorithms

- Loss functions
    - 10 inputs, 100 hidden neurons, and 5 outputs; 3000 weights and biases!



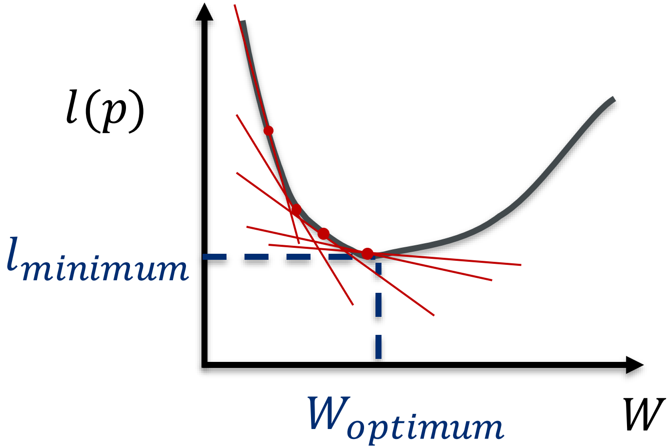
Learning Algorithms

- Optimizers or learning algorithms
  - Gradient descent
  - Stochastic gradient descent
  - Conjugate gradient descent
  - Adaptive moment estimation (ADAM)

Learning Algorithms

- Gradient descent and backpropagation

  - Gradient descent is simply gradient (slope in 2d) calculation over loss function in a proper step, called learning rate (similar to penalty constant in mathematical optimization).



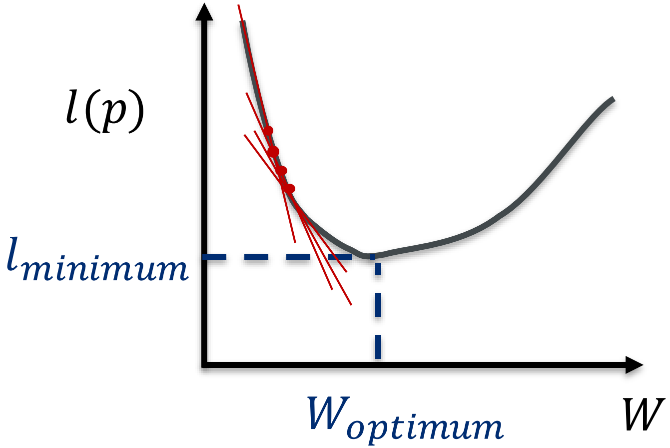
Small learning rate
- ✓ Time consuming

Large learning rate
- ✓ Higher error, less accurate

Adoptive learning rate
- ✓ Start with large magnitude and make it smaller as slope decreases

Learning Algorithms

- Gradient descent and backpropagation
  - Gradient descent is simply gradient (slope in 2d) calculation over loss function in a proper step, called learning rate (similar to penalty constant in mathematical optimization).



Small learning rate
- ✓ Time consuming

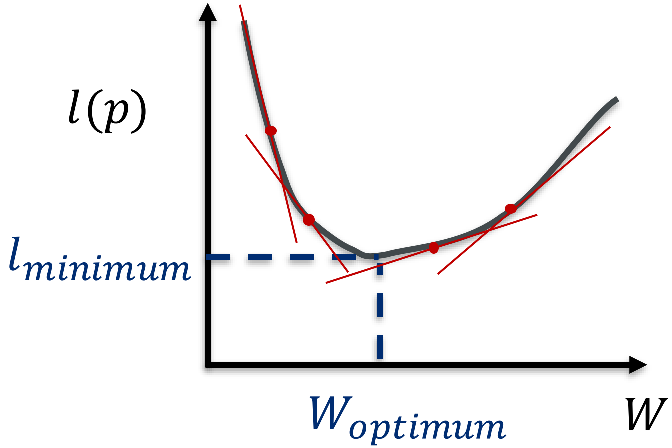
Large learning rate
- ✓ Higher error, less accurate

Adoptive learning rate
- ✓ Start with large magnitude and make it smaller as slope decreases

Learning Algorithms

- Gradient descent and backpropagation

  - Gradient descent is simply gradient (slope in 2d) calculation over loss function in a proper step, called learning rate (similar to penalty constant in mathematical optimization).



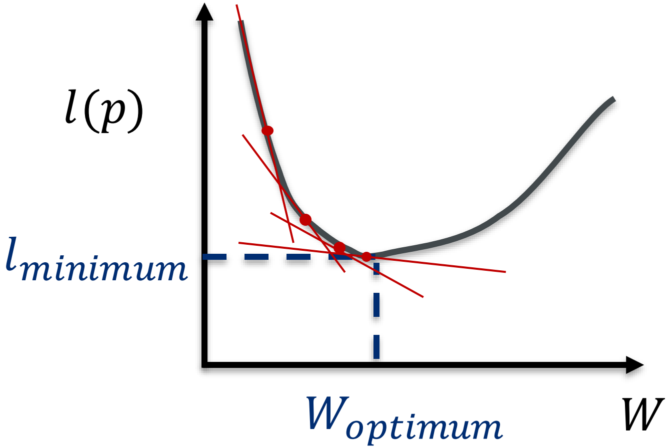
Small learning rate
- ✓ Time consuming

Large learning rate
- ✓ Higher error, less accurate

Adoptive learning rate
- ✓ Start with large magnitude and make it smaller as slope decreases

Learning Algorithms

- Gradient descent and backpropagation
  - Gradient descent is simply gradient (slope in 2d) calculation over loss function in a proper step, called learning rate (similar to penalty constant in mathematical optimization).



Small learning rate
- ✓ Time consuming

Large learning rate
- ✓ Higher error, less accurate

Adoptive learning rate
- ✓ Start with large magnitude and make it smaller as slope decreases

Learning Algorithms

- Gradient descent and backpropagation

  Backpropagate the error and update the weight and biases

Feedforward

Backpropagate

Learning rate
$(0 \leq \gamma \leq 1)$

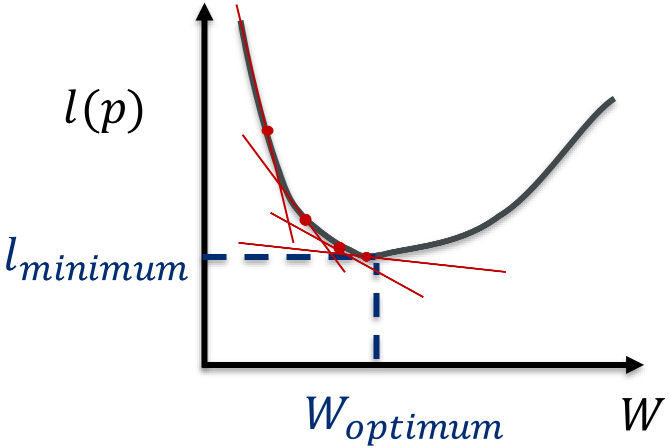
$$p_{k,updated} = p_{k,old} - \gamma \frac{\partial l}{\partial p_k}$$

A network variable (e.g., the weight corresponding to the weight connection of neuron 5 in 3rd layer and neuron 23 in the 4th layer)

Learning Algorithms

- Gradient descent and backpropagation

  Backpropagate the error and update the weight and biases



In each iteration:

- ✓ Compute $\frac{\partial l}{\partial p_k}$ using all training repository at once in each iteration
- ✓ Time-consuming and memory intensive for large networks and/or large repositories

Learning Algorithms

- Stochastic gradient descent and backpropagation
  - Divide training repository to batches of training datapoints, each with a memory-friendly number of datapoints
    - Batch size
    - Batch number
  - Feed each batch to the network one at a time and update the variables
  - Once all batches pass the network once, it is counted as one iteration
  - Train the network for a number of iterations

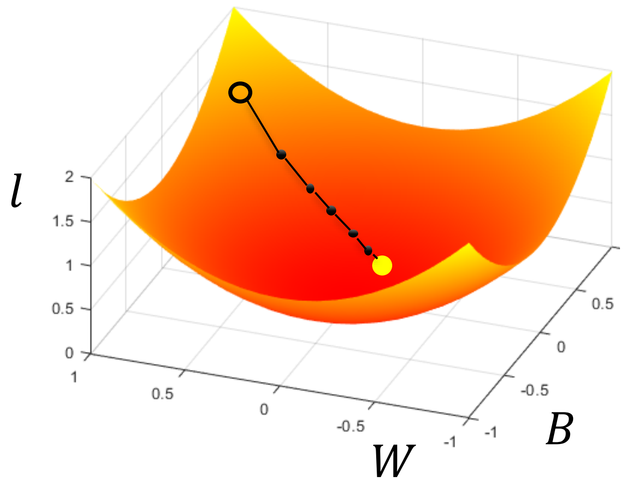
Learning Algorithms

- Stochastic gradient descent and backpropagation
  - Example: 32 batches each with 32 training datapoints for $N = 1000$ (notations are based on slide 3 of lecture B of current module):

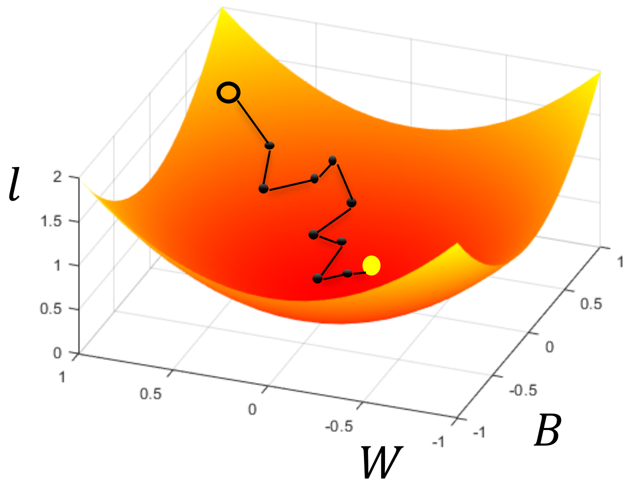
Batch 01 (32 datapoints)	Batch 02 (32 datapoints)		Batch 31 (32 datapoints)	Batch 32 (8 datapoints)
$\boldsymbol{X}_1, \boldsymbol{Y}_1$	$\boldsymbol{X}_{33}, \boldsymbol{Y}_{33}$		$\boldsymbol{X}_{961}, \boldsymbol{Y}_{961}$	$\boldsymbol{X}_{993}, \boldsymbol{Y}_{993}$
$\boldsymbol{X}_2, \boldsymbol{Y}_2$	$\boldsymbol{X}_{34}, \boldsymbol{Y}_{34}$		$\boldsymbol{X}_{962}, \boldsymbol{Y}_{962}$	$\boldsymbol{X}_{994}, \boldsymbol{Y}_{994}$
.	.		.	.
.	.	. . .	.	.
.	.		.	.
$\boldsymbol{X}_{32}, \boldsymbol{Y}_{32}$	$\boldsymbol{X}_{64}, \boldsymbol{Y}_{64}$		$\boldsymbol{X}_{992}, \boldsymbol{Y}_{992}$	$\boldsymbol{X}_{1000}, \boldsymbol{Y}_{1000}$

- Stochastic gradient descent versus gradient descent

Gradient Descent

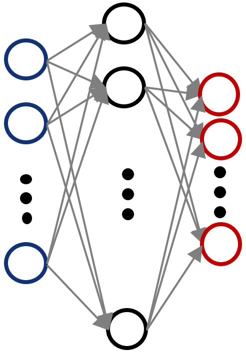


Stochastic Gradient Descent



Learning Algorithms

- Universal Approximation Theorem
  - With mild assumptions on the transfer function, a single hidden layer feed-forward neural network with a finite number of neurons can approximate any convex continues function.



Cybenko, G. (1989) "Approximations by superpositions of sigmoidal functions", Mathematics of Control, Signals, and Systems, 2(4), 303–314. doi:10.1007/BF02551274
Hanin, B. (2018). Approximating Continuous Functions by ReLU Nets of Minimal Width. arXiv preprint arXiv:1710.11278.

Learning Algorithms

- In this lecture, you learned about:
  - Gradient descent
  - Stochastic gradient descent
  - Backpropagation
  - Universal approximation theorem
- In the next module, we will practice training and testing artificial neural networks over available programming packages

JOHNS HOPKINS
WHITING SCHOOL *of* ENGINEERING