ECE595 / STAT598: Machine Learning I
Lecture 18.2: Back Propagation

Spring 2020

Stanley Chan

School of Electrical and Computer Engineering
Purdue University

Outline

Discriminative Approaches

- Lecture 16 Perceptron 1: Definition and Basic Concepts
- Lecture 17 Perceptron 2: Algorithm and Property
- Lecture 18 Multi-Layer Perceptron: Back Propagation

This lecture: Multi-Layer Perceptron: Back Propagation

- Multi-Layer Perceptron
  - Hidden Layer
  - Matrix Representation
- Back Propagation
  - Chain Rule
  - 4 Fundamental Equations
  - Algorithm
  - Interpretation

Back Propagation: A 20-Minute Tour

- You will be able to find **A LOT OF** blogs on the internet discussing how back propagation is being implemented.
- Some are **mystifying** back propagation
- Some literally just teach you the procedure of back propagation without telling you the intuition
- I find the following online book by Mike Nielsen fairly well-written
- http://neuralnetworksanddeeplearning.com/
- The following slides are written based on Nielsen's book
- We will not go into great details
- The purpose to get you exposed to the idea, and de-mystify back propagation
- As stated before, back propagation is chain rule + very careful book keeping

Back Propagation

- Here is the loss function you want to minimize:

$$J(\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L) = \sum_{i=1}^{N} \|\sigma(\boldsymbol{W}_L^T \ldots \sigma(\boldsymbol{W}_2^T \sigma(\boldsymbol{W}_1^T \boldsymbol{x}_i))) - \boldsymbol{y}_i\|^2$$

- You have a set of nonlinear activation functions, usually the sigmoid.
- To optimize, you need gradient descent. For example, for $\boldsymbol{W}_1$

$$\boldsymbol{W}_1^{t+1} = \boldsymbol{W}_1^t - \alpha \nabla J(\boldsymbol{W}_1^t)$$

- But you need to do this for all $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L$.
- And there are lots of sigmoid functions.
- Let us do the brute force.
- And this is back-propagation. (Really? Yes...)

Let us See an Example

- Let us look at two layers

$$J(\boldsymbol{W}_1, \boldsymbol{W}_2) = \|\underbrace{\sigma(\boldsymbol{W}_2^T \sigma(\boldsymbol{W}_1^T \boldsymbol{x}))}_{\boldsymbol{a}_2} - \boldsymbol{y}\|^2$$

- Let us go **backward**:

$$\frac{\partial J}{\partial \boldsymbol{W}_2} = \frac{\partial J}{\partial \boldsymbol{a}_2} \cdot \frac{\partial \boldsymbol{a}_2}{\partial \boldsymbol{W}_2}$$

- Now, what is $\boldsymbol{a}_2$?

$$\boldsymbol{a}_2 = \sigma(\underbrace{\boldsymbol{W}_2^T \sigma(\boldsymbol{W}_1^T \boldsymbol{x})}_{\boldsymbol{z}_2})$$

So let us compute:

$$\frac{\partial \boldsymbol{a}_2}{\partial \boldsymbol{W}_2} = \frac{\partial \boldsymbol{a}_2}{\partial \boldsymbol{z}_2} \cdot \frac{\partial \boldsymbol{z}_2}{\partial \boldsymbol{W}_2}.$$

Let us See an Example

$$J(\boldsymbol{W}_1, \boldsymbol{W}_2) = \|\sigma(\boldsymbol{W}_2^T \underbrace{\sigma(\boldsymbol{W}_1^T \boldsymbol{x})}_{\boldsymbol{a}_1}) - \boldsymbol{y}\|^2$$

- How about $\boldsymbol{W}_1$? Again, let us go **backward**:

$$\frac{\partial J}{\partial \boldsymbol{W}_1} = \frac{\partial J}{\partial \boldsymbol{a}_2} \cdot \frac{\partial \boldsymbol{a}_2}{\partial \boldsymbol{W}_1}$$

- But you can now repeat the calculation as follows (Let $\boldsymbol{z}_1 = \boldsymbol{W}_1^T \boldsymbol{x}$)

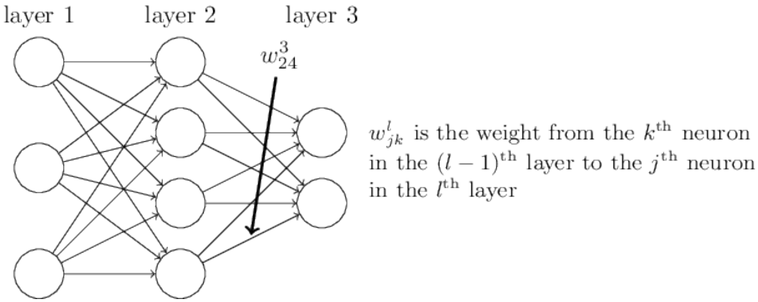
$$\begin{aligned} \frac{\partial \boldsymbol{a}_2}{\partial \boldsymbol{W}_1} &= \frac{\partial \boldsymbol{a}_2}{\partial \boldsymbol{a}_1} \frac{\partial \boldsymbol{a}_1}{\partial \boldsymbol{W}_1} \\ &= \frac{\partial \boldsymbol{a}_2}{\partial \boldsymbol{a}_1} \frac{\partial \boldsymbol{a}_1}{\partial \boldsymbol{z}_1} \frac{\partial \boldsymbol{z}_1}{\boldsymbol{W}_1} \end{aligned}$$

- So it is just a very long sequence of chain rule.

Notations for Back Propagation

- The following notations are based on Nielsen's online book.
- The purpose of doing these is to write down a concise algorithm.

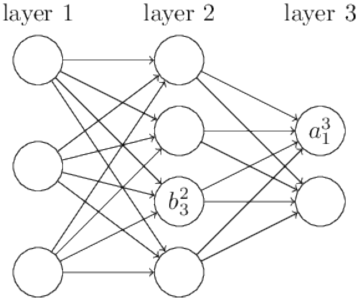
Weights:



- w^{3}_{24}: The 3rd layer
- w^{3}_{24}: From 4-th neuron to 2-nd neuron

Notations for Back Propagation

Activation and Bias:



- a_1^3: 3rd layer, 1st activation
- b_3^2: 2nd layer, 3rd bias
- Here is the relationship. Think of $\sigma(\boldsymbol{w}^T\boldsymbol{x} + w_0)$:

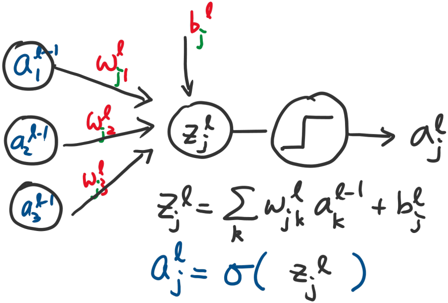
$$a_j^\ell = \sigma\left(\sum_k w_{jk}^\ell a_k^{\ell-1} + b_j^\ell\right).$$

Understanding Back Propagation

- This is the main equation

$$a_j^\ell = \sigma\underbrace{\left(\sum_k w_{jk}^\ell a_k^{\ell-1} + b_j^\ell\right)}_{z_j^\ell}, \quad \text{or} \quad a_j^\ell = \sigma(z_j^\ell).$$

- a_j^ℓ: activation, z_j^ℓ: intermediate.

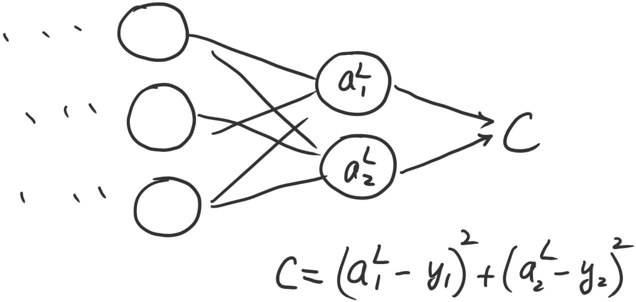


Loss

- The loss takes the form of

$$C = \sum_j (a_j^L - y_j)^2$$

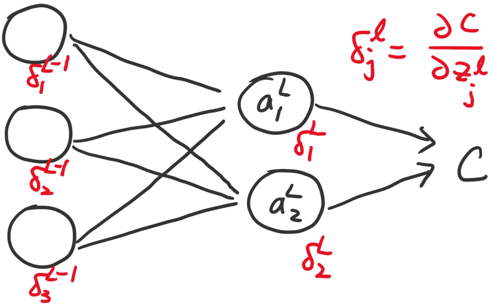
- Think of two-class cross-entropy where each $\boldsymbol{a}^L$ is a 2-by-1 vector



Error Term

- The error is defined as

$$\delta_j^\ell = \frac{\partial C}{\partial z_j^\ell}$$



- You can show that at the output,

$$\delta_j^L = \frac{\partial C}{\partial a_j^L}\frac{\partial a_j^L}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L}\sigma'(z_j^L).$$

4 Fundamental Equations for Back Propagation

BP Equation 1: For the error in the output layer:

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \sigma'(z_j^L). \tag{BP-1}$$

- First term: $\frac{\partial C}{\partial a_j^L}$ is rate of change w.r.t. a_j^L
- Second term: $\sigma'(z_j^L) =$ rate of change w.r.t. z_j^L.
- So it is just **chain rule**.
- Example: If $C = \frac{1}{2}\sum_j (y_j - a_j^L)^2$, then

$$\frac{\partial C}{\partial a_j^L} = (a_j^L - y_j)$$

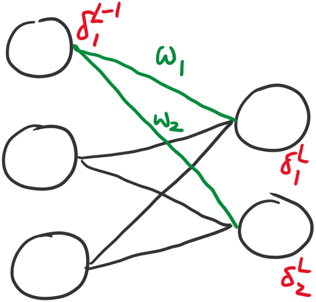
- Matrix-vector form: $\boldsymbol{\delta}^L = \nabla_a C \odot \sigma'(\boldsymbol{z}^L)$

4 Fundamental Equations for Back Propagation

BP Equation 2: An equation for the error δ^ℓ in terms of the error in the next layer, $\delta^{\ell+1}$

$$\boldsymbol{\delta}^\ell = ((\boldsymbol{w}^{\ell+1})^T \boldsymbol{\delta}^{\ell+1}) \odot \sigma'(\boldsymbol{z}^\ell). \qquad \text{(BP-2)}$$

- You start with $\boldsymbol{\delta}^{\ell+1}$. Take weighted average $\boldsymbol{w}^{\ell+1}$.



- (BP-1) and (BP-2) can help you determine error at any layer.

4 Fundamental Equations for Back Propagation

Equation 3: An equation for the rate of change of the cost with respect to any bias in the network.

$$\frac{\partial C}{\partial b_j^\ell} = \delta_j^\ell. \tag{BP-3}$$

- Good news: We have already known δ_j^ℓ from Equation 1na dn 2.
- So computing $\frac{\partial C}{\partial b_j^\ell}$ is easy.

Equation 4: An equation for the rate of change of the cost with respect to any weight in the network.

$$\frac{\partial C}{\partial w_{jk}^\ell} = a_k^{\ell-1}\delta_j^\ell \tag{BP-4}$$

- Again, everything on the right is known. So it is easy to compute.

Back Propagation Algorithm

- Below is a very concise summary of the BP algorithm

1. **Input x:** Set the corresponding activation a^1 for the input layer.

2. **Feedforward:** For each $l = 2, 3, \ldots, L$ compute $z^l = w^l a^{l-1} + b^l$ and $a^l = \sigma(z^l)$.

3. **Output error δ^L:** Compute the vector $\delta^L = \nabla_a C \odot \sigma'(z^L)$.

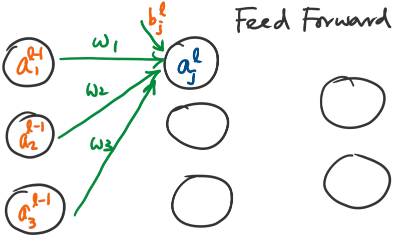
4. **Backpropagate the error:** For each $l = L-1, L-2, \ldots, 2$ compute $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$.

5. **Output:** The gradient of the cost function is given by $\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j$ and $\frac{\partial C}{\partial b^l_j} = \delta^l_j$.

Step 2: Feed Forward Step

- Let us take a closer look at Step 2
- The feed forward step computes the intermediate variables and the activations

$$
\begin{aligned}
\boldsymbol{z}^{\ell} &= (\boldsymbol{w}^{\ell})^T \boldsymbol{a}^{\ell-1} + b^{\ell} \\
\boldsymbol{a}^{\ell} &= \sigma(\boldsymbol{z}^{\ell}).
\end{aligned}
$$

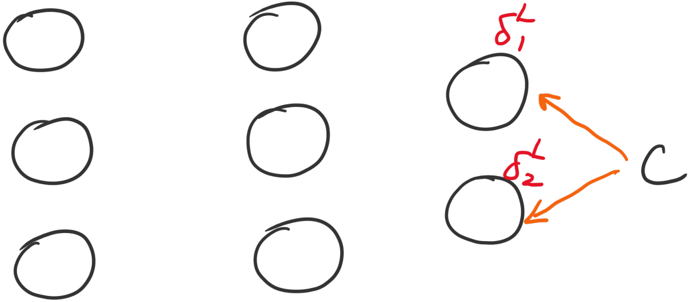


$$a_j^{\ell} = \sigma(z_j^{\ell}) = \sigma\left((w^{\ell})^T a^{\ell-1} + b_j^{\ell}\right)$$

Step 3: Output Error

- Let us take a closer look at Step 3
- The output error is given by (BP-1)

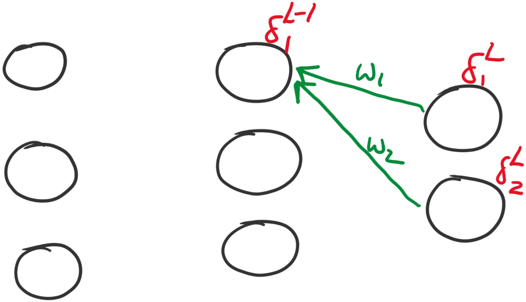
$$\boldsymbol{\delta}^L = \nabla_a C \odot \sigma'(\boldsymbol{z}^L)$$



Step 4: Output Error

- Let us take a closer look at Step 4
- The error back propagation is given by (BP-2)

$$\boldsymbol{\delta}^{\ell} = ((\boldsymbol{w}^{\ell+1})^{T}\boldsymbol{\delta}^{\ell+1}) \odot \sigma'(\boldsymbol{z}^{\ell}).$$



Summary of Back Propagation

- There is no dark magic behind back propagation
- It is literally just **chain rule**
- You need to do this chain rule very systematically and **carefully**
- Then you can derive the back propagation steps
- Nielsen wrote in his book that

> ... How backpropagation could have been discovered in the first place? In fact, if you follow the approach I just sketched you will discover a proof of backpropagation...You make those simplifications, get a shorter proof, and write that out....The result after a few iterations is the one we saw earlier, short but somewhat obscure...

- Most deep learning libraries have built-in back propagation steps.
- You don't have to implement it yourself, but you need to know what's behind it.

Reading List

- Michael Nielsen, Neural Networks and Deep Learning, http://neuralnetworksanddeeplearning.com/chap2.html
  - Very well written. Easy to follow.
- Duda, Hart, Stork, Pattern Classification, Chapter 5
  - Classical treatment. Comprehensive. Readable.
- Bishop, Pattern Recognition and Machine Learning, Chapter 5
  - Somewhat Bayesian. Good for those who like statistics
- Stanford CS 231N, http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf
  - Good numerical example.
- CMU https://www.cs.cmu.edu/~mgormley/courses/10601-s17/slides/lecture20-backprop.pdf
- Cornell https://www.cs.cornell.edu/courses/cs5740/2016sp/resources/backprop.pdf