IDC410
A course on Image Processing and Machine Learning
(Lecture 13)

Shashikant Dugad,

IISER Mohali

Reading Material

Suggested Books:

1. **Neural Networks and Deep Learning by Michael Nielsen**
2. **Fundamentals of Deep Learning by Nikhil Buduma**

Source for this presentation:

Neural Networks and Deep Learning by Michael Nielsen

https://www.scaler.com/topics/deep-learning/introduction-to-feed-forward-neural-network/

https://www.turing.com/kb/mathematical-formulation-of-feed-forward-neural-network

http://machine-learning-for-physicists.org. by Florian Marquardt

3Blue1Brown (Youtube Videos)

https://www.datacamp.com/tutorial/introduction-to-activation-functions-in-neural-networks

https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/

Network Training

Backpropagation: Alternate Approach

Source:

https://medium.com/towards-data-science/deriving-backpropagation-with-cross-entropy-loss-d24811edeaf9

https://medium.com/towards-data-science/backpropagation-the-natural-proof-946c5abf63b1

Entropy and Cross-Entropy

Entropy: 1

- The origin of the word *entropy* used in the context of ML, can be traced back to a paper written by Claude Shannon in 1948, where he laid the foundation for the communication infrastructure and the modern information age
- Suppose you want to communicate with your friend from Australia on a few popular cricketers say, a) Sachin b) Bradman, c) Kapil and d) Shane Warne
- Let us assume that vocabulary is just limited to these *4 players*. One can encode these *4* players into a binary pattern using *2 bits*. Each player can be represented by the two bit pattern as: *00, 01, 10 and 11*.
- If you are referring names of *4* cricketers to your friend *1000 times* then you need *2 bits* per player and therefore, total message length will be *2000 bits*.
- Cost of communication is driven by the length of the message
- If you do assignments of the bit pattern for each player *smartly* then, can you bring down the average message length to *< 2?* If so, you can save the communication cost!

Entropy: 2

- Let us say Sachin is your favourite player and you talk about him ***half*** of the time ***(50%)***, Bradman for ***25%*** and other 2 (Kapil and Warne) for ***12.5%*** of the time each. Try following bit assignment: a) Sachin → ***0***, b) Bradman → ***1***, c) Kapil → ***01*** and d) Warne → ***10***? Because Sachin is encoded into a single bit, communication cost is expected to fall since you talk about him most of the time. This would have worked except that when you send a signal like ***0000001***, it could mean ***6 times*** Sachin ***(bit: 0)*** followed by ***1 time*** Bradman ***(bit: 1)*** or it could also mean ***5 times*** Sachin followed by ***1 time*** Kapil ***(bit: 01***). Therefore, entire message cannot be interpreted correctly.
- However following bit assignment removes such ambiguity; a) Sachin → ***0***, b) Bradman → ***10***, c) Kapil → ***110*** and d) Warne → ***111***. Brilliant! Try any combination of 4 players and you will not find any ambiguous clash with multiple interpretation.
- Though number of bits are increased the for two of the cricketers, one *still saves the cost. With* talk of Sachin ***50%***, Bradman ***25%*** and other two cricketers for ***12.5%*** of the time, for a communication of 1000 names, the total length of message would be: 500 *(1 bit for each of the 500 Sachin)* + 500 *(2 bits each for 250 Bradman)* + 375 (*3 bits for 125 Kapil*) + 375 *(3 bits for 125 Warne*) **EQUALS to 1750 bits** and therefore, **average message length** has now come down from ***2 to 1.75 bits*** per name!

Entropy and Cross Entropy: 3

- ***Entropy:*** It is the minimum average message length when using the best code possible is called **entropy**. Note that the entropy depends on the probability distribution of the words that need to be transmitted. Because of your favouritism towards Sachin, it made sense to use 3-bit encoding for a few of the other words as long as Sachin is represented by 1 bit and this improved our entropy.
- Australian friend's favourite players are different than Indian: He talks 50% time about Warne, 25% Kapil and 12.5% remaining two players (Sachin and Bradman!). If he has to communicate 1000 names to his Indian friends with the same best possible code described above then total message length will be: 1500 bits (for 500 Warne) + 750 bits (250 Kapil) + 250 (125 Bradman) + 125 (125 Sachin) = 2625 bits and an ***average message length of 2.6***! Since, it is higher than the best possible average length, it ***CANNOT*** be referred as ***entropy***
- ***Cross-Entropy:*** New average message length ***(2.6)*** which is higher than the minimum average length ***(1.75)*** is called as the cross-entropy. We call it an ***entropy*** when one uses the ***optimized bit pattern*** derived for a ***certain probability distribution.*** Furthermore, we call it as a ***cross-entropy*** when the ***same bit pattern*** is used for ***another probability distribution*** resulting in ***higher average message length.*** Therefore, by definition, the value of ***Cross-Entropy*** is always higher than that of ***Entropy.***

Entropy and Cross Entropy: 4

- The ***cross-entropy*** basically measures the difference between two probability distributions
- For a given discrete probability distribution ***(p_i)***, the ***entropy*** can be calculated using:

$$E = \sum_{i=1}^{N} p_i \log_2 p_i$$

- In our example; ***N=4, p_1=0.5, p_2=0.25 and p_3=p_4=0.125***. Substituting these values in above equation we get, ***E=1.75***

- The optimised bit pattern (# of bits to define each person) chosen ($\boldsymbol{n_1^b = 1, n_2^b = 2, n_3^b = 3\ and\ n_4^b = 3}$) bit pattern that was chosen also gives the ***average message length (AML) to be 1.75!*** The ***AML*** can be calculated using:

$$\boldsymbol{AML = \sum_{i=1}^{N} n_i^b p_i}$$

Entropy and Cross Entropy: 5

- The ***cross-entropy*** has to deal two the probability distributions (PD) a) the PD of a person from India ($\boldsymbol{p_i^A}$) and b) the PD of another person from Australia ($\boldsymbol{p_i^B}$) who are communicating with each other. In such case the ***cross-entropy*** can be defined as,

$$\mathbf{C}E = -\sum_{i=1}^{N} \boldsymbol{p_i^A} \log_2 \boldsymbol{p_i^B}$$

- For example, if there are only 2 names ***(N=2)*** and your PD (p_i^A) is ***[0.2, 0.8]*** and your friend's PD (p_i^B) is ***[0.4, 0.6]***, then the ***CE = -(0.2*log$_2$ (0.4) + 0.8*log$_2$ (0.6))***.
- Another interpretation of ***entropy*** is that, it is a measure of ***uncertainty***. For example if Sachin had 100% probability then ***entropy=0*** implying ***NO uncertainty*** whereas, all 4 players having equal probability has a ***maximum uncertainty*** with the highest ***entropy=2.***

Backpropagation Algorithm for Multi-Binary Network

Choice of Loss Function for Classification Network

- Mean Square Error *(MSE)* loss function is quite suitable for regression output. The MSE has a nice convex U shape providing higher gradient value while away from the minima and lower gradient values when getting closer to the minima of loss function *(MSE)*
- However, the ***MSE*** is not an appropriate loss function for classification problem
- ***MSE*** for the classification problem does not have a convex shape. The ***MSE*** indeed has a lot more complicated shape making it difficult for the model to converge to the global minima.
- The model for classification network, needs to predict the discrete probability distribution (PD) for various classes with an expectation of predicting higher probabilities for the true input class. The ***MSE*** performs poorly in predicting such PD.
- If MSE is used as a loss function to measure the difference between the expected and predicted probability distributions, it gives a non-convex shape instead of the clean convex U shape!
- Task at hand is to identify loss function that will naturally minimise the difference between the expected and predicted probability distributions of output classes.

Choice of Loss Function for Classification Network

- The *cross-entropy* seems like a *natural* candidate for the loss function for classification network since its value represents the measure of difference between two PD.

- *Binary classification*: there is only one neuron in the last layer, the activation is a *sigmoid*, the output is *p* and can be represented as a probability distribution *[p, 1-p]*. The loss function is (binary) *cross-entropy*.

- *Categorical multi-class classification:* there are *n* output neurons, *ONLY* one the class is *TRUE* in a given event, the activation is a *softmax*, the output is a probability distribution of size *n*, the probabilities adding up to *1* for the loss is (categorical) cross-entropy

- *Multi-label classification:* there are *n* output neurons, *FEW* of the classes can be *TRUE* in a given event, the activation is *sigmoids* (one for each neuron), the output is a distribution of size *n* like [0.1, 0.3, 0.9, 0.1, 0.8] which does *NOT* add up to 1. This output is converted into n-binary probability distributions and the loss is the sum (or average) of the *n* binary cross-entropy losses.

Backpropagation: Multi-Class Categorical Classification

- The choice of loss function is imperative to the network's performance because eventually the parameters in the network are going to be set such that the loss is minimized.

- **Cross-Entropy loss function** is widely used for **multi-binary classification** problem. When ***ONLY*** one of the output must be ***TRUE*** then it is referred as **categorical cross-entropy loss function.**

- For **categorical cross-entropy function**, the column matrix of ***TRUE output*** (y_m) is **one-hot encoded** matrix which means ***ONLY*** one of the elements in the column matrix is ***ONE*** and all other elements are **ZERO**. ***m*** denotes the class number and number of elements in the matrix represents ***TOTAL*** number of classes.

- For ***one-hot encoded output***, activation function **Softmax** is strongly recommended for predicting the probability of each class at the output. Each element of the **PREDICTED** column matrix (a_m^H) has a value ranging between **0** to **1** with a boundary condition that sum of all the elements in this matrix to must be **UNITY** ($0 \ll a_m^H \ll 1 \;\; and \;\; \sum_m a_m^H = 1$)

Categorical Cross Entropy Loss Function is defined as: $J_{(x,y)} = -\sum_m y_m \cdot \ln(a_m^H)$

Backpropagation General Formalism in the Elemental Form

⇨ $z_m^L = \sum_i w_{im}^L * a_i^{L-1} + b_m^L$ *unless* $L = 0$ *then* $z_m^L = \sum_i w_{im}^L * x_i + b_m^L$ **Eqn. 1**

⇨ $a_m^L = h(z_m^L)$ **Eqn. 2**

⇦ $\delta_n^L = h'(z_n^L) \sum_m \delta_m^{L+1} \cdot w_{nm}^{L+1}$ *unless* $L = H$ *then* $\delta_n^L = (a_n^L - y_n) \cdot h'(z_n^L)$ **Eqn. 3**

⇦ $\frac{\partial J}{\partial w_{mn}^L} = \delta_n^L \cdot a_m^{L-1}$ *unless* $L = 0$ *then* $\frac{\partial J}{\partial w_{mn}^L} = \delta_n^L \cdot x_m$ **Eqn. 4**

⇦ $\frac{\partial J}{\partial b_n^L} = \delta_n^L$ **Eqn. 5**

Equations 1 and 2 are used for Forward Propagation

Equations 3, 4 and 5 are used for Backward Propagation

Backpropagation: Multi-Class Categorical Classification

We now describe the backpropagation algorithm procedure for **categorical cross-entropy loss function**

- All the steps for backpropagation summarised in previous slide remains same except the 3rd step in which δ_n^H will have evaluated for the **categorical cross-entropy function**
- Activation output of **n**th neuron in the last layer is determined using **Softmax** activation function defined as:

$$a_n^H = a_n = h(z_n) = \frac{e^{z_n}}{\sum_{m=1}^{C} e^{z_m}}$$

- Since we are dealing with ***ONLY*** output layer, the superscript **H** is dropped for convenience
- It is **VERY** important to note that the activation output of **n**th neuron **(a_n)** in the output layer also depends on the pre-activation of all ***other*** neurons **(z_m, m≠n)** in the output layer (arising from denominator in the above equation) which was ***NOT*** the case with the ***SIGMOID*** activation function

Backpropagation: Multi-Class Categorical Classification

- Due to this feature of activation function we can use the chain rule to find **δ_n.** as follows:

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_m \frac{\partial J}{\partial a_m} \cdot \frac{\partial a_m}{\partial z_n}$$

- It can further be expanded as follows:

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_{m \neq n} \left(\frac{\partial J}{\partial a_m} \cdot \frac{\partial a_m}{\partial z_n}\right) + \frac{\partial J}{\partial a_n} \cdot \frac{\partial a_n}{\partial z_n}$$

Eqn. 1

- Two of the term, **$\partial J/\partial a_n$** and **$\partial a_n/\partial z_n$** in above expression can be obtained as:

$$\frac{\partial J}{\partial a_n} = \frac{\partial(-\sum_m y_m \cdot \ln(a_m))}{\partial a_n} = \frac{\partial(\, y_n \cdot \ln(a_n)\,)}{\partial a_n} = -\frac{y_n}{a_n}$$

$$\frac{\partial a_n}{\partial z_n} = \frac{\partial(\, e^{z_n}/(\sum_m e^{z_m}\,)}{\partial z_n} = \frac{e^{z_n}}{\sum_m e^{z_m}} \left(\frac{e^{z_n}}{e^{z_n}} - \frac{e^{z_n}}{\sum_m e^{z_m}}\right) = a_n(1 - a_n)$$

Backpropagation: Multi-Class Categorical Classification

- Substituting these two terms in Eqn. 1 of previous slide:

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_{m\neq n}\left(\frac{\partial J}{\partial a_m}\cdot\frac{\partial a_m}{\partial z_n}\right) + -y_n(1-a_n) \qquad \textbf{Eqn. 2}$$

- For remaining terms $\partial J/\partial a_m$ and $\partial a_m/\partial z_n$ ($m\neq n$) in above equation:

$$\frac{\partial J}{\partial a_m} = \frac{\partial\left(-\sum_{m'} y_{m'}\cdot\ln(a_{m'})\right)}{\partial a_m} = -\frac{y_m}{a_m}$$

$$\frac{\partial a_m}{\partial z_n} = \frac{\partial\left(e^{z_m}/(\sum_{m'} e^{z_{m'}})\right)}{\partial z_n} = \frac{e^{z_m}}{\sum_{m'} e^{z_{m'}}}\left(\frac{\partial e^{z_m}/\partial z_n}{e^{z_m}} - \frac{\partial\sum_{m'} e^{z_{m'}}/\partial z_n}{\sum_{m'} e^{z_{m'}}}\right) = \frac{e^{z_m}}{\sum_{m'} e^{z_{m'}}}\left(0 - \frac{e^{z_n}}{\sum_{m'} e^{z_{m'}}}\right)$$

$$= -a_m a_n$$

- Combining these two terms: $\frac{\partial J}{\partial a_m}\cdot\frac{\partial a_m}{\partial z_n} = -\frac{y_m}{a_m}\cdot -a_m a_n = y_m a_n$

Backpropagation: Multi-Class Categorical Classification

- Substituting these two terms in **Eqn. 2** of previous slide:

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_{m \neq n} \left(\frac{\partial J}{\partial a_m} \cdot \frac{\partial a_m}{\partial z_n}\right) + -y_n(1 - a_n) = \sum_{m \neq n} y_m a_n + -y_n(1 - a_n)$$

- which can be further simplified as:

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_{m \neq n} y_m a_n - y_n(1 - a_n) = \sum_{m \neq n} y_m a_n + y_n a_n - y_n = \sum_{m} y_m a_n - y_n = a_n \sum_{m} y_m - y_n$$

- We know that $\sum \boldsymbol{y_m} = \mathbf{1}$ Therefore, finally $\boldsymbol{\delta_n}$ in elemental and matrix form can be written as:

$$\delta_n = \frac{\partial J}{\partial z_n} = a_n - y_n \qquad \delta^H = \frac{\partial J}{\partial z^H} = a^H - y$$

→ **Vector Form Representation**

Backpropagation: Multi-Class Categorical Classification

- **Backpropagation equations for Categorical Cross Entropy Loss Function with Softmax activation function in the output layer; the FIVE backpropagation equations can be written as:**

⇨ $z^L = (W^L)^t a^{L-1} + b^L$ *unless* $L = 0$ *then* $z^L = (W^L)^t x + b^L$

⇨ $a^L = h(z^L)$

- $\delta^L = h'(z^L) \odot W^{L+1}\delta^{L+1}$ *unless* $L = H$ *then* $\delta^L = (a^L - y)$
- $\frac{\partial J}{\partial b^L} = \delta^L$
- $\frac{\partial J}{\partial W^L} = a^{L-1} \cdot (\delta^L)^t$ *unless* $L = 0$ *then* $\frac{\partial J}{\partial W^L} = x \cdot (\delta^L)^t$