Cross Entropy Loss

MD, TANVIR ROUF SHAWON
LECTURER
AUST

Loss Function

- When working on a Machine Learning or a Deep Learning Problem, loss/cost functions are used to optimize the model during training.

- The objective is almost always to minimize the loss function.

- The lower the loss the better the model.

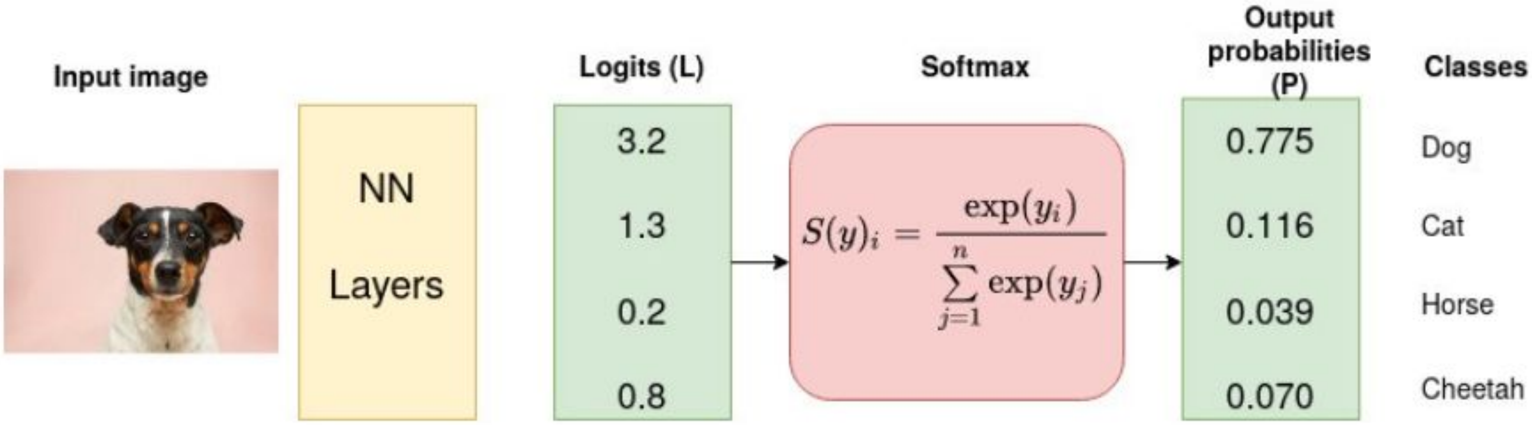
- Cross-Entropy loss is a most important cost function.

- It is used to optimize classification models.

Loss Function

- Cross-Entropy
- Hinge
- Huber
- Kullback-Leibler
- RMSE
- MAE (L1)
- MSE (L2)

Cross Entropy Loss



Input image

NN

Layers

Logits (L)

3.2

1.3

0.2

0.8

Softmax

$$S(y)_i = \frac{\exp(y_i)}{\sum_{j=1}^{n} \exp(y_j)}$$

Output probabilities (P)

0.775

0.116

0.039

0.070

Classes

Dog

Cat

Horse

Cheetah

Why use softmax as opposed to standard normalization?

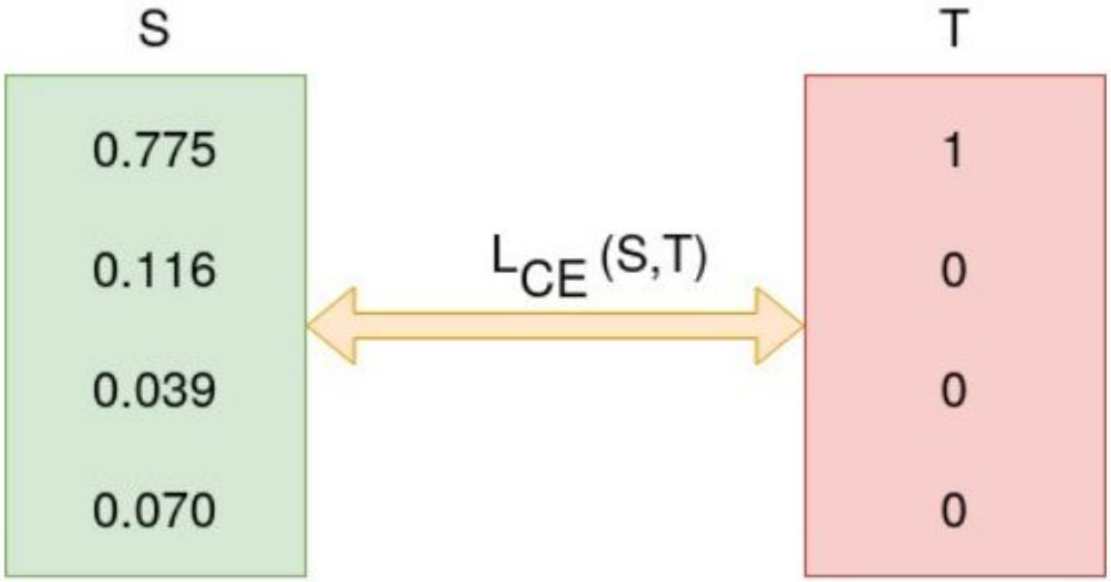
```
>>> softmax([1,2])                      # blurry image of a ferret
[0.26894142, 0.73105858])                    # it is a cat perhaps !?
>>> softmax([10,20])                    # crisp image of a cat
[0.0000453978687, 0.999954602])         # it is definitely a CAT !


>>> std_norm([1,2])                     # blurry image of a ferret
[0.3333333333333333, 0.6666666666666666]        #    it is a cat perhaps !?
>>> std_norm([10,20])                   # crisp image of a cat
[0.3333333333333333, 0.6666666666666666]        #    it is a cat perhaps !?
```

Logits

the vector of raw (non-normalized) predictions that a classification model generates, which is ordinarily then passed to a normalization function. If the model is solving a multi-class classification problem, logits typically become an input to the softmax function. The softmax function then generates a vector of (normalized) probabilities with one value for each possible class.

Cross Entropy Loss



S	T
0.775	1
0.116	0
0.039	0
0.070	0

$L_{CE}(S,T)$

Entropy

The concept of cross-entropy traces back into the field of Information Theory where Claude Shannon introduced the concept of entropy in 1948.

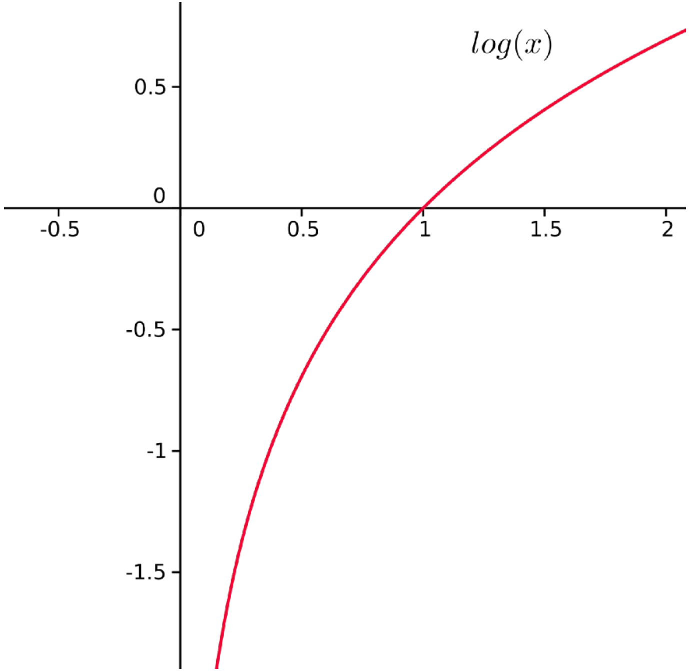
Entropy of a random variable X is the level of uncertainty inherent in the variables possible outcome.

$$H(X) = \begin{cases} -\int_x p(x)\log p(x), & \text{if } X \text{ is continous} \\ -\sum_x p(x)\log p(x), & \text{if } X \text{ is discrete} \end{cases}$$

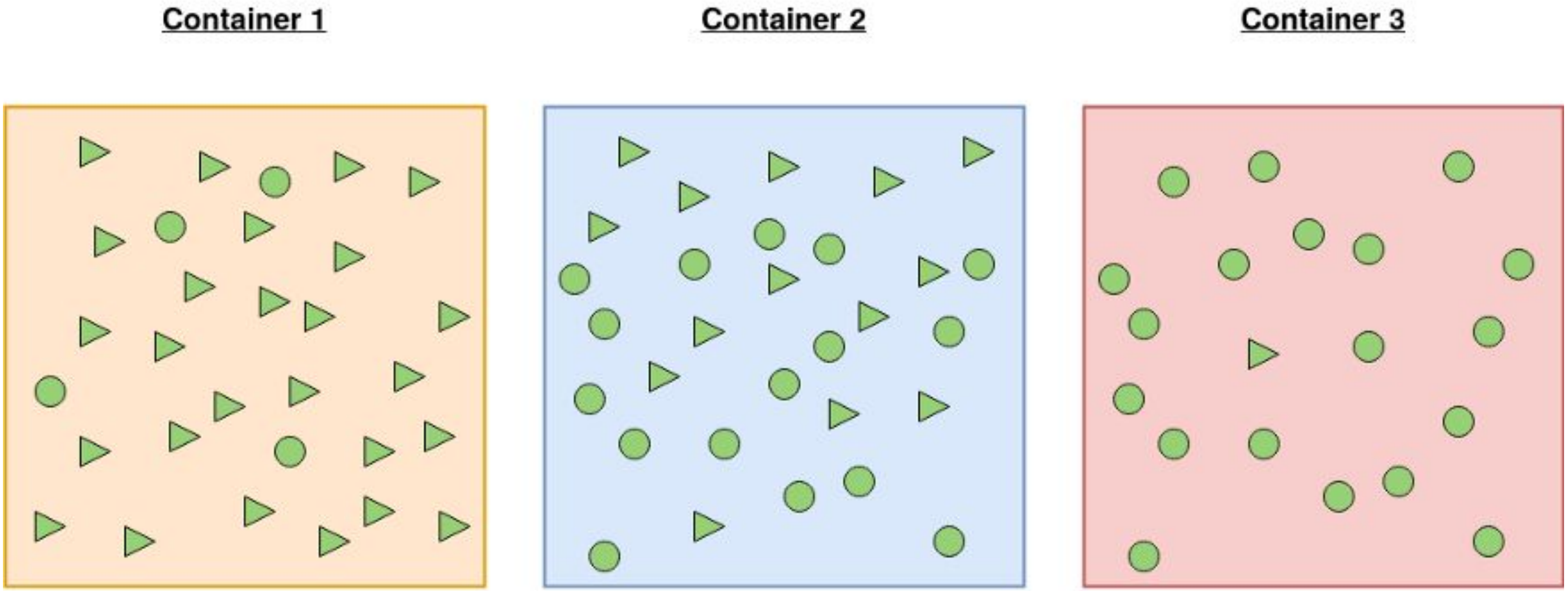
logarithm is to make it growing linearly with system size and "behaving like information". When probability is very low the calculation can be tends to zero.

Reason of Negative Sign

log(p(x))<0 for all p(x) in (0,1) . p(x) is a probability distribution and therefore the values must range between 0 and 1.



Entropy (Example)



Container 1

Container 2

Container 3

#▷ =26, #● = 4

#▷ =14, #● = 16

#▷ = 1 #● = 29

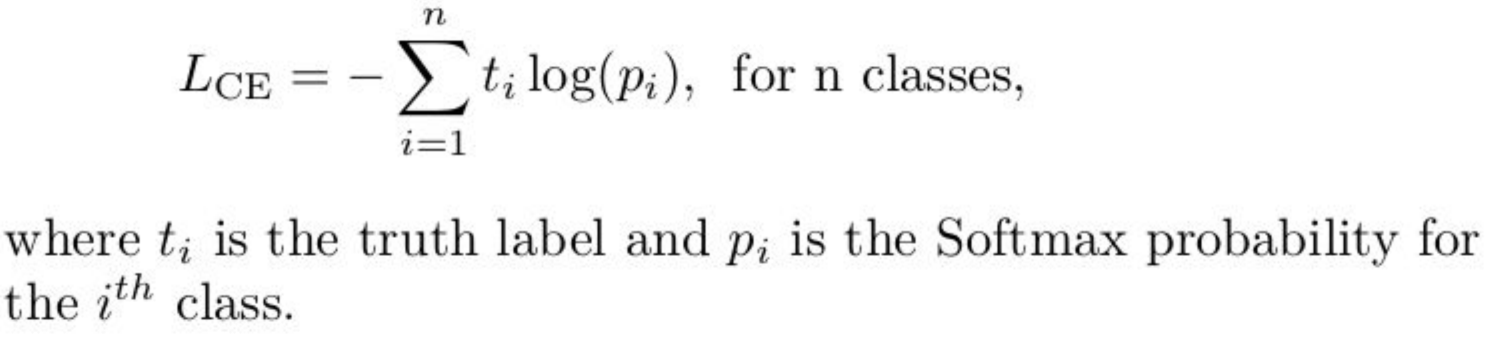
Cross Entropy Loss

- Cross-Entropy Loss Function is also called logarithmic loss, log loss or logistic loss.

- Each predicted class probability is compared to the actual class desired output 0 or 1

- A score/loss is calculated that penalizes the probability based on how far it is from the actual expected value.

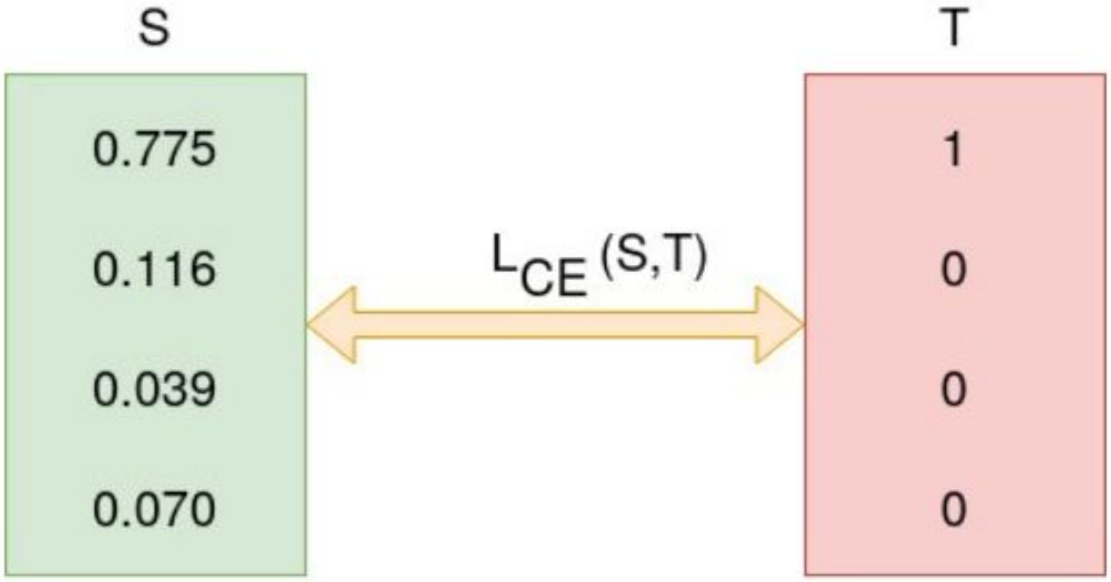
Cross Entropy Loss



$$L_{\mathrm{CE}} = -\sum_{i=1}^{n} t_i \log(p_i), \quad \text{for n classes,}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

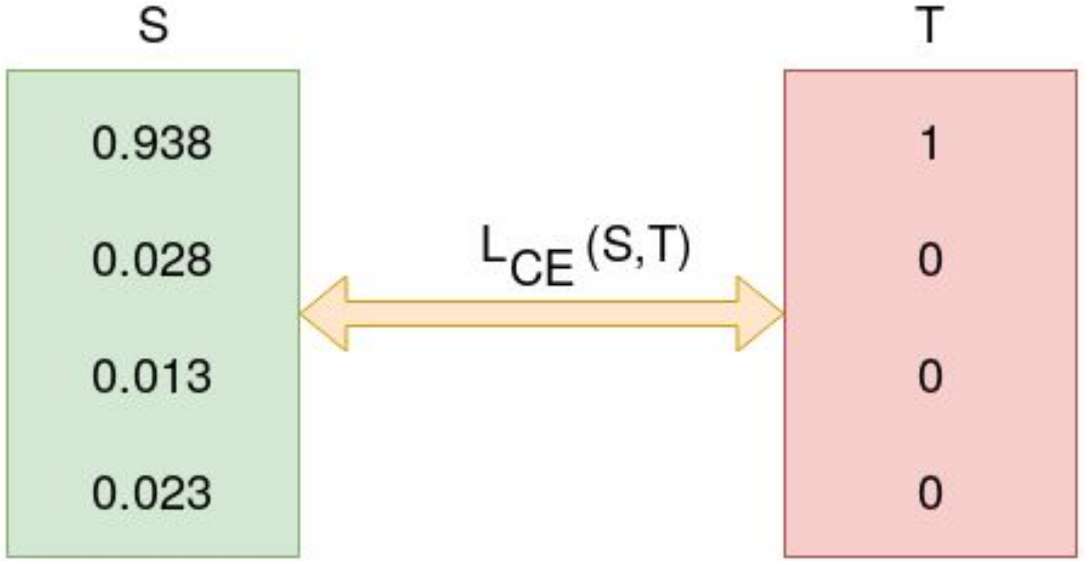
Cross Entropy Loss (Example)



Cross Entropy Loss (Example)

$$\begin{aligned}
L_{CE} &= -\sum_{i=1} T_i \log(S_i) \\
&= -\left[1\log_2(0.775) + 0\log_2(0.126) + 0\log_2(0.039) + 0\log_2(0.070)\right] \\
&= -\log_2(0.775) \\
&= 0.3677
\end{aligned}$$

Cross Entropy Loss (Example)

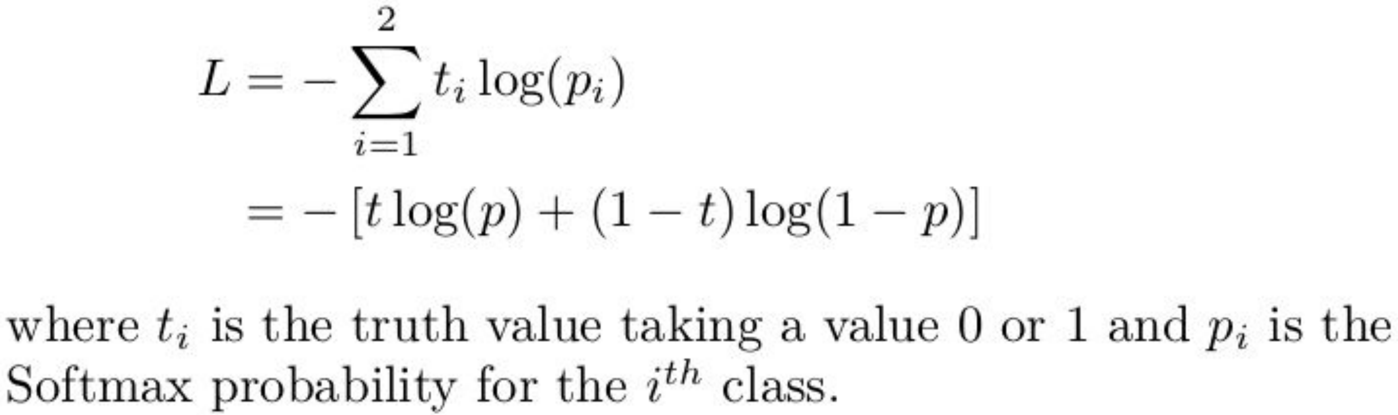


S	T
0.938	1
0.028	0
0.013	0
0.023	0

L_{CE} (S,T)

$$\begin{aligned} L_{CE} &= -1\log_2(0.936) + 0 + 0 + 0 \\ &= 0.095 \end{aligned}$$

Binary Cross Entropy Loss



$$\begin{aligned} L &= -\sum_{i=1}^{2} t_i \log(p_i) \\ &= -\left[t \log(p) + (1-t)\log(1-p)\right] \end{aligned}$$

where t_i is the truth value taking a value 0 or 1 and p_i is the Softmax probability for the i^{th} class.

When using log base 2, the unit of entropy is bits, where as with natural log, the unit is nats. One isn't better than the other. It's kind of like the difference between using km/hour and m/s. It is possible that log base 2 is faster to compute than the logarithm. However, in practice, computing cross-entropy is pretty much never the most costly part of the algorithm, so it's not something to be overly concerned with.

Categorical Cross-Entropy and Sparse Categorical Cross-Entropy

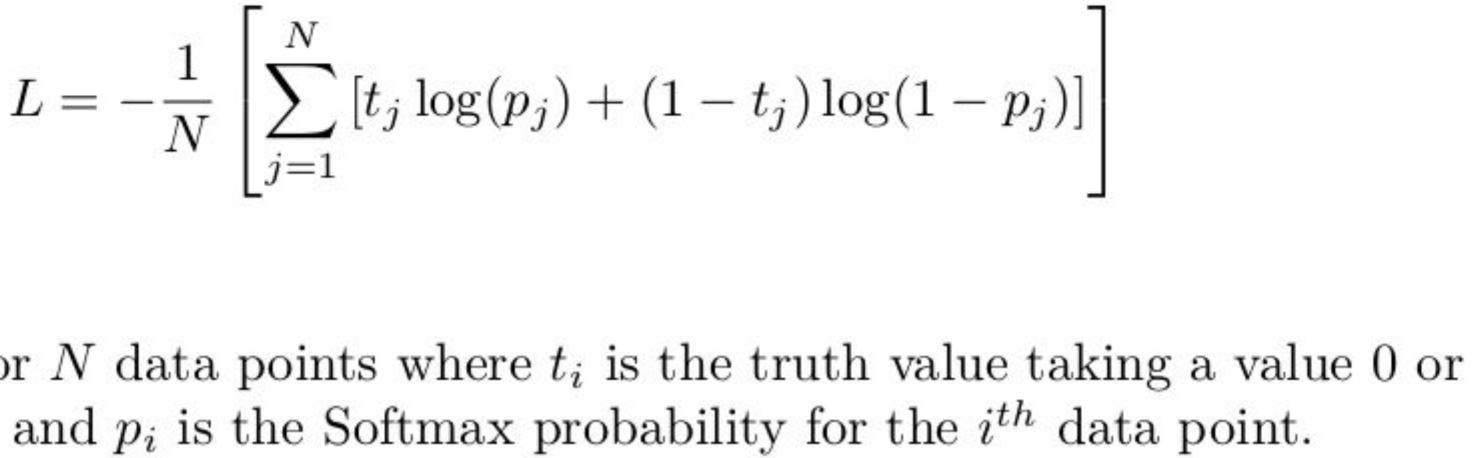
Both categorical cross entropy and sparse categorical cross-entropy have the same loss function. The only difference between the two is on how truth labels are defined.

Categorical cross-entropy is used when true labels are one-hot encoded, for example, we have the following true values for 3-class classification problem [1,0,0], [0,1,0] and [0,0,1].

In sparse categorical cross-entropy , truth labels are integer encoded, for example, [1], [2] and [3] for 3-class problem.

Loss Vs Cost

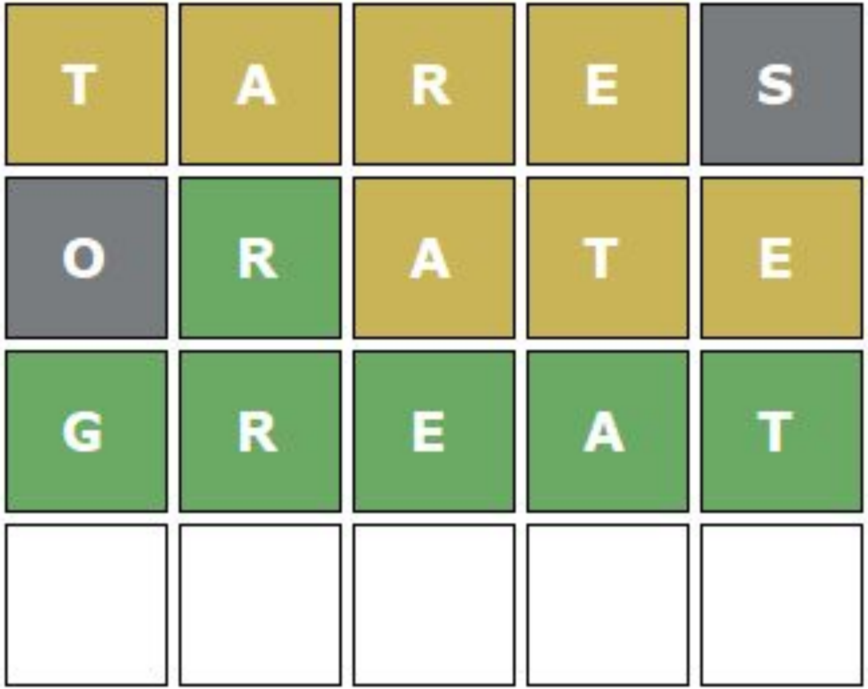
The loss function computes the error for a single training example, while the cost function is the average of the loss functions of the entire training set.



$$L = -\frac{1}{N}\left[\sum_{j=1}^{N}[t_j \log(p_j) + (1 - t_j)\log(1 - p_j)]\right]$$

for N data points where t_i is the truth value taking a value 0 or 1 and p_i is the Softmax probability for the i^{th} data point.

Let’s Solve Wordle!



T	A	R	E	S
O	R	A	T	E
G	R	E	A	T

What is Information?

Information is measured in bits. When measuring information, we are measuring the number of bits, or the number of “yes or no” questions we would have had to ask to reach the same conclusion.

Wordle

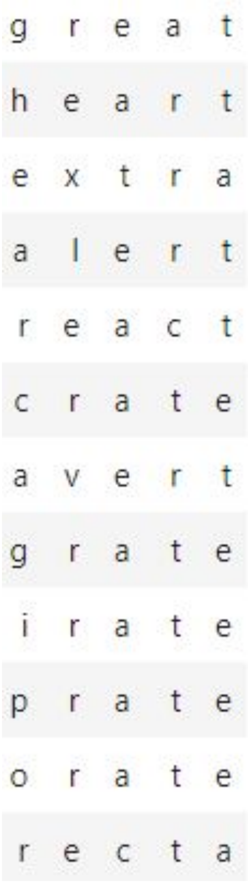
- The goal of Wordle is to guess the 5 letter word.
- You are given 6 guesses, every time you guess you receive information about the hidden word depending on what you guessed.
- After answering each letter will either be green, amber or gray.
- Green means the letter is in the correct spot.
- Amber means the letter is in the word but not on that spot.
- Gray means the letter is not in the hidden word.

Wordle

- 5757 five-letter words in the English language.
- At the beginning of the game, there are ($\log_2(5757)$) or 12.49 bits of hidden information /uncertainty.
- As you play the game, you acquire information about the hidden word.
- The goal is to create an algorithm that in each guess chooses the word that adds the most information (that reduces the uncertainty the most).

Wordle

- Let the initial guess is “tares”
- There are only 12 words in the English language that fit these criteria
- we have reduced the uncertainty from 12.5 bits (5757 possible words) to 3.6 bits (12 words)
- Our guess made us gain 8.9 bits of information
- After the second guess “orate”, the only possible word is great



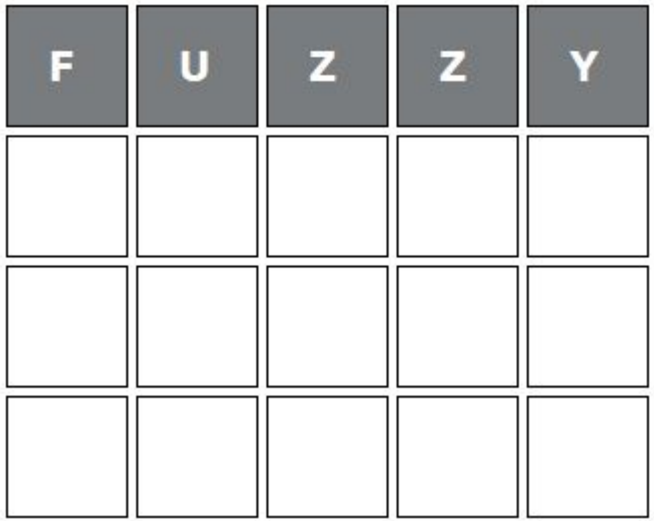
great
heart
extra
alert
react
crate
avert
grate
irate
prate
orate
recta

Wordle

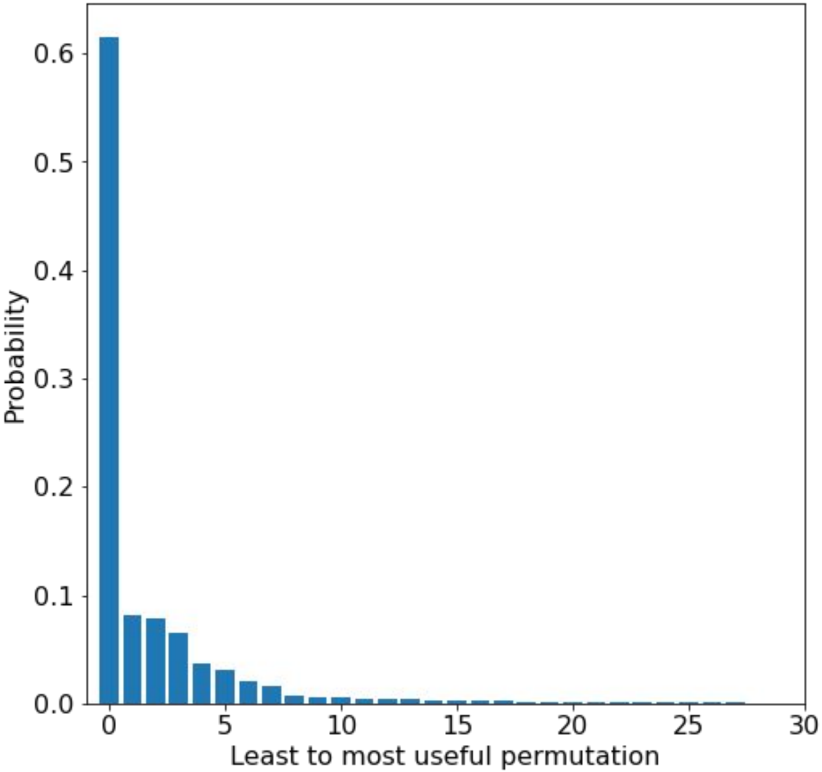
The goal is to choose the word that reduces our possible words list as much as possible. This is the word that maximizes the amount of information gained by guessing it. We cannot know the information gained without knowing the answer, so instead, we guess the word with the largest expected value of information gained!

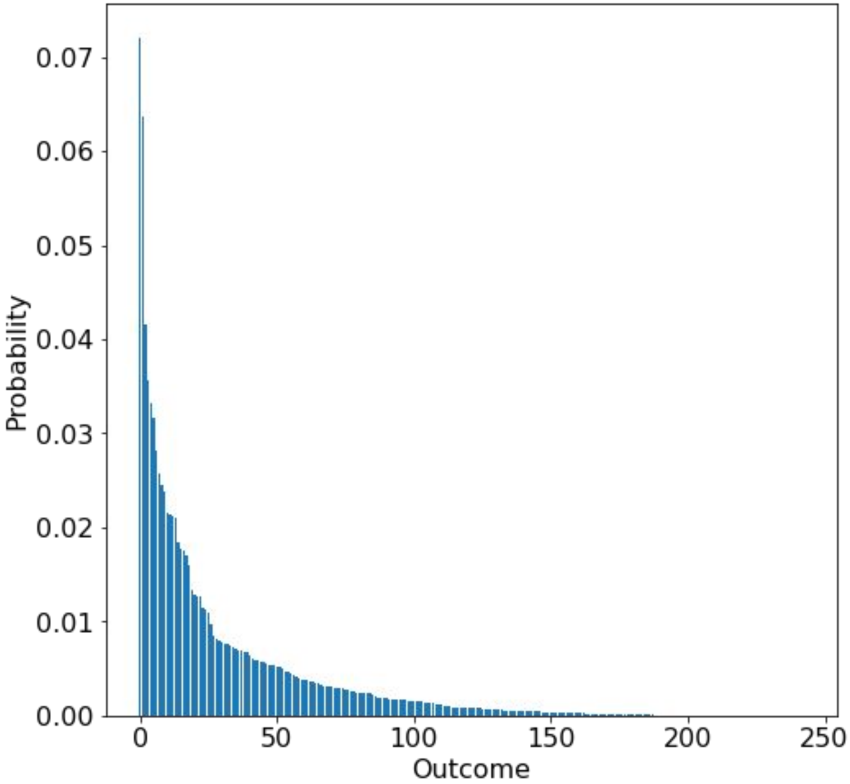
Wordle

- Take the word “fuzzy” as initial word
- If we look at the words that do not contain the letters “F”, “U”, “Z” and “Y”, there are a total of 3543/5757 (62% of all 5 letter words).
- This means that if we guess the word “fuzzy”, 62% of the time we will get a Wordle that looks like this.
- In this case, all grays is the outcome with the least amount of information gained



Wordle (Fuzzy vs Tares)





Wordle

- Now take a look at “tares” as an initial guess.
- The probability of getting all grays is only around 7%.
- Therefore, 93% of the time we will know something about at least one of the letters.

With these probabilities, we can compute the information gained by each outcome. Then, we can take the expected value of the information gained by each word using the formulas showed previously.

Wordle

The word “fuzzy” has an expected information gain of 2.28 bits, whereas the word “tares” has an expected information gain of 6.21 bits. This means that on average the word “fuzzy” will reduce our possible words list from 5757 to 1185, whereas “tares” will reduce it to 78.

Algorithm

- Compute the entropy for all possible words
- Choose the word that has the highest entropy
- Submit guess to Wordle and fetch the output
- Update list of remaining possible words
- Repeat steps 1–4 until the guess is equal to the answer

References

- https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e
- https://towardsdatascience.com/information-theory-applied-to-wordle-b63b34a6538e
- https://www.youtube.com/watch?v=v68zYyaEmEA