Loss Functions

- The loss function is the function that computes the distance between the current output of the algorithm and the expected output. It's a method to evaluate how your algorithm models the data. It can be categorized into two groups. One for classification (discrete values, 0,1,2…) and the other for regression (continuous values).

TYPES OF LOSS FUNCTION:

1. Regression Loss Functions

```
Mean Absolute Error
Mean Squared Error
Root Mean Square error (RMSE)
```

2. Binary Classification Loss Functions

```
Binary Cross-Entropy
```

3. Multi-Class Classification Loss Functions

```
Multi-Class Cross-Entropy Loss
Sparse Multiclass Cross-Entropy Loss
```

Regression Losses

We know all of this regression loss function but here i discuss a brief

Mean Absolute Error

- Regression metric which measures the average magnitude of errors in a group of predictions, without considering their directions. In other words, it's a mean of absolute differences among predictions and expected results where all individual deviations have even importance.

$$\mathrm{MAE} = \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\text{test set}} \underbrace{|y_i}_{\text{predicted vaue}} - \underbrace{\hat{y}_i|}_{\text{actual value}}$$

where:

i — index of sample,

ŷ — predicted value,

y — expected value,

m — number of samples in dataset.

Sometimes it is possible to see the form of formula with swapped predicted value and expected value, but it works the same.

Mean Squared Error

- One of the most commonly used and firstly explained regression metrics. Average squared difference between the predictions and expected results. In other words, an alteration of MAE where instead of taking the absolute value of differences, they are squared.
- In MAE, the partial error values were equal to the distances between points in the coordinate system. Regarding MSE, each partial error is equivalent to the area of the square created out of the geometrical distance between the measured points. All region areas are summed up and averaged.

$$\underbrace{\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}}_{\text{test set}} \underbrace{(y_i}_{\text{predicted vaue}} - \underbrace{\hat{y}_i}_{\text{actual value}})^2$$

Where

i — index of sample,

ŷ — predicted value,

y — expected value,

m — number of samples in dataset.

Root Mean Square error (RMSE)

- Root Mean Square error is the extension of MSE — measured as the average of square root of sum of squared differences between predictions and actual observations.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

Classification Losses

Binary Classification Loss Functions

Binary Cross Entropy

- Also called Sigmoid Cross-Entropy loss. It is a Sigmoid activation plus a Cross-Entropy loss. Unlike Softmax loss it is independent for each vector component (class), meaning that the loss computed for every CNN output vector component is not affected by other component values.
- Binary cross entropy measures how far away from the true value (which is either 0 or 1) the prediction is for each of the classes and then averages these class-wise errors to obtain the final loss.
- We can define cross entropy as the difference between two probability distributions p and q, where p is our true output and q is our estimate of this true output.
- it Only use for binary classification problem

$$H(x) = \sum_{i=1}^{N} \overset{\text{true label}}{p}(x) \log \underset{\text{estimate}}{q}(x)$$

Multi-Class Classification Loss Functions

Categorical cross-entropy

- Used binary and multiclass problem, the label needs to be encoded as categorical, one-hot encoding representation (for 3 classes: [0, 1, 0], [1,0,0]…)
- It is a loss function that is used for single label categorization. This is when only one category is applicable for each data point. In other words, an example can belong to one class only.

$$L(y, \hat{y}) = - \sum_{j=0}^{M} \sum_{i=0}^{N} (y_{ij} * log(\hat{y}_{ij}))$$

- Use categorical crossentropy in classification problems where only one result can be correct.
- Example: In the MNIST problem where you have images of the numbers 0,1, 2, 3, 4, 5, 6, 7, 8, and 9. Categorical crossentropy gives the probability that an image of a number is, for example, a 4 or a 9.

- Categorical cross-entropy will compare the distribution of the predictions (the activations in the output layer, one for each class) with the true distribution, where the probability of the true class is set to 1 and 0 for the other classes. To put it in a different way, the true class is represented as a one-hot encoded vector, and the closer the model's outputs are to that vector, the lower the loss.

Sparse Categorical cross-entropy

- Used binary and multiclass problem (the label is an integer — 0 or 1 or … n, depends on the number of labels)

If your targets are **one-hot encoded**, use `categorical_crossentropy`.

- Examples of one-hot encodings:
  - `[1,0,0]`
  - `[0,1,0]`
  - `[0,0,1]`

But if your targets are **integers**, use `sparse_categorical_crossentropy`.

- Examples of integer encodings (*for the sake of completion*):
  - `1`
  - `2`
  - `3`

All Losses : [https://keras.io/api/losses/ (https://keras.io/api/losses/)](https://keras.io/api/losses/)

Summary

There are three kinds of classification tasks:

```
1. Binary classification: two exclusive classes
2. Multi-class classification: more than two exclusive classes
3. Multi-label classification: just non-exclusive classes
```

Here, we can say

```
1. In the case of (1), you need to use binary cross entropy.
2. In the case of (2), you need to use categorical cross entropy.
3.In the case of (3), you need to use binary cross entropy.
```