Introduction to Loss Functions in Deep Learning

Lê Minh Khôi

February 27, 2026

Contents

1 Introduction

In deep learning, a model learns by minimizing something called a **loss function**. The loss function tells the model how wrong its prediction is.

If the loss is large, the prediction is bad. If the loss is small, the prediction is good.

In this report, we study:

- Mean Squared Error (MSE) for regression
- Cross Entropy (CE) for classification
- The meaning of loss functions
- Probability distributions

2 Mean Squared Error (MSE) – For Regression

Regression means predicting a continuous value, such as:

- House price
- Temperature
- Stock value

The formula of MSE is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where:

- y_i = true value
- $\hat{y}_i$ = predicted value
- n = number of samples

2.1 Meaning of MSE

MSE calculates the average squared difference between prediction and true value.

Why square the difference?

- To make all errors positive
- To penalize large errors more

If prediction is very wrong, the loss becomes very large.

2.2 Example in Python (PyTorch)

```python
import torch
import torch.nn as nn

y_true = torch.tensor([10.0,5.0,9.0])
y_pred = torch.tensor([8.0,7.0,6.0])

criterion = nn.MSELoss()
loss = criterion(y_pred, y_true)

print(loss.item())
```

3 Cross Entropy (CE) – For Classification

Classification means predicting a category, such as:

- Cat or Dog
- Spam or Not Spam
- Digit 0-9

Cross Entropy loss formula:

$$CE = -\sum_i y_i \log(\hat{y}_i)$$

Where:

- y_i = true probability (usually 0 or 1)
- $\hat{y}_i$ = predicted probability
- log: logarithm base e

3.1 Why Cross Entropy?

In classification, the model outputs probabilities.

Example:

True label = Cat

Model prediction:

$$[0.9, 0.1]$$

This means:

- 90% Cat
- 10% Dog

If the model predicts high probability for the correct class, the loss is small.

If it predicts low probability for the correct class, the loss is large.

3.2 Example in Python (PyTorch)

```python
import torch
import torch.nn as nn

criterion = nn.CrossEntropyLoss()

# Example: 2 classes
# Raw scores (logits)
outputs = torch.tensor([[2.0, 0.5]])

# True class index (0 = first class)
labels = torch.tensor([0])

loss = criterion(outputs, labels)
print(loss.item())
```

4 Meaning of Loss Functions

Loss functions guide the learning process.

Training steps:

1. Model makes prediction
2. Loss function calculates error

3. Backpropagation computes gradients
4. We update weights

The goal is:

$$\text{Minimize Loss}$$

Different tasks require different loss functions:

- Regression $\rightarrow$ MSE
- Classification $\rightarrow$ Cross Entropy

5 Probability Distribution

In classification, outputs are probabilities.

A probability distribution must satisfy:

- All values between 0 and 1
- Sum of probabilities equals 1

Example:

$$[0.7, 0.2, 0.1]$$

Sum:

$$0.7 + 0.2 + 0.1 = 1$$

Neural networks often use the **Softmax function** to convert outputs into probabilities:

$$Softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

This ensures the outputs form a valid probability distribution.

6 Conclusion

In this report, we learned:

- MSE is used for regression
- Cross Entropy is used for classification
- Loss functions measure prediction error
- Probability distributions are important in classification

Understanding loss functions is important because they control how a neural network learns.