Regularization

Idant Srivastava

December 2025

Contents

Chapter 1

Overfitting and Underfitting

1.1 Overfitting

Overfitting occurs when a model learns the training data too well, including its noise and outliers, resulting in poor performance on unseen data. The model exhibits low training error but high validation/test error, indicating it has memorized the patterns instead of learning from the training data. This typically happens when the model complexity exceeds what the data can support, or when training continues for too long without proper regularization.

$$\mathcal{L}_{train} \ll \mathcal{L}_{test}$$

The generalization gap is defined as: $\Delta = \mathcal{L}_{test} - \mathcal{L}_{train}$ If Δ is large, then the model is overfitted.

1.2 Underfitting

1.2.1 Concept

Underfitting occurs when a model is too simple to capture the underlying patterns in the data, resulting in poor performance on both training and test sets. The model lacks the capacity to learn the relationships between inputs and outputs, leading to high bias. This happens when the model architecture is insufficient, training time is inadequate, or regularization is too aggressive.

$$\mathcal{L}_{train} \gg \mathcal{L}_{optimal}$$

where $\mathcal{L}_{optimal}$ represents the achievable minimum loss. Both training and test errors remain high and are unacceptably large.

Chapter 2

L2 Regularization

2.1 Concept

L2 regularization adds a penalty term proportional to the squared magnitude of model weights to the loss function, encouraging smaller weight values throughout the network. This prevents any single weight from becoming too large, which reduces model complexity and helps prevent overfitting by preventing the model from relying too heavily on any particular feature. L2 regularization is also known as Ridge regression in linear models and weight decay in optimization contexts.

2.2 Mathematical Foundation

$$\mathcal{L}_{total} = \mathcal{L}_{data} + \lambda \sum_{l=1}^{L} \|W^{(l)}\|^2$$

- $\mathcal{L}_{data}$ is the original loss (e.g., cross-entropy or MSE)
- λ is the regularization hyperparameter controlling penalty strength
- $W^{(l)}$ represents the weight matrix at layer l

Chapter 3

L1 Regularization

3.1 Concept

L1 regularization adds a penalty proportional to the absolute value of model weights, forcing less important feature weights to become exactly zero. This results in simpler and more interpretable models. Unlike L2 regularization, which shrinks all weights proportionally, L1 creates sparse solutions where many weights become exactly zero.

3.2 Mathematical Foundation

The regularized loss function is:

$$\mathcal{L}_{total} = \mathcal{L}_{data} + \lambda \sum_{l=1}^{L} \|W^{(l)}\|$$

$\|W^{(l)}\|$ is the L1 norm (sum of absolute values)

L1 regularization subtracts a constant value (proportional to λ) from each weight regardless of magnitude, which drives small weights to exactly zero.

Chapter 4

Dropout

4.1 Concept

Dropout is a powerful regularization technique that prevents neural networks from overfitting by randomly deactivating (dropping out) a fraction of neurons during each training iteration, forcing the network to learn more robust, less co-dependent features, improving generalization to new data.

4.2 Working

How it works (Training Phase):

- For each neuron, sample a binary mask:
  - Keep the neuron with probability p
  - Drop it with probability $1 - p$
- If a neuron is dropped, it does not contribute to:
  - The forward pass
  - The backward pass (no gradients flow through it)

How it works (Testing Phase):

- No neurons are dropped.
- The full network is used.
- Because of the scaling during training, output magnitudes remain consistent.

Why it helps:

- Prevents the network from relying too much on specific neurons.
- Forces more distributed, robust feature learning.
- Acts like training many small sub-networks and averaging them.

Chapter 5

Batch Normalization

5.1 Concept

Batch Normalization normalizes the inputs of each layer across a mini-batch to have zero mean and unit variance. By standardizing activations, it reduces the sensitivity to weight initialization, allows higher learning rates, and acts as a regularizer by introducing noise through batch statistics. The technique learns scale (γ) and shift (β) parameters to restore representational power after normalization.

5.2 Mathematical Foundation

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma_{\mathcal{B}}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$$

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta$$

where γ and β are learnable parameters (initially $\gamma = 1$ and $\beta = 0$).

At test time, running averages are used:

$$\mu_{test} = E[\mu_{\mathcal{B}}]$$

$$\sigma_{test}^2 = E[\sigma_{\mathcal{B}}^2]$$

$$y_{test} = \gamma \frac{x - \mu_{test}}{\sqrt{\sigma_{test}^2 + \epsilon}} + \beta$$

Chapter 6

Layer Normalization

6.1 Concept

Layer Normalization normalizes activations across all features for each individual sample rather than across the batch dimension, making it suitable for RNNs and scenarios with small batch sizes. Unlike Batch Normalization which computes statistics across samples, Layer Normalization computes them across features within a single sample, providing consistent normalization regardless of batch size. This approach eliminates the discrepancy between training and test time behavior seen in Batch Normalization.

6.2 Mathematical Foundation

For a single sample $x \in R^d$ with d features:

$$\mu = \frac{1}{d}\sum_{i=1}^{d} x_i$$

$$\sigma^2 = \frac{1}{d}\sum_{i=1}^{d} (x_i - \mu)^2$$

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

$$y_i = \gamma_i \hat{x}_i + \beta_i$$

where γ_i and β_i are learnable per-feature parameters.

Chapter 7

Early Stopping

7.1 Concept

Early stopping is a regularization technique that monitors model performance on a validation set during training and terminates training when validation performance stops improving, preventing the model from overfitting. The method maintains a record of the best model weights based on validation loss and restores these weights when training is stopped. Early stopping effectively limits the model's effective capacity by controlling training duration rather than modifying the loss function or architecture.

7.2 Generalization Analysis

1. During training, you monitor validation loss after each epoch.
2. In the beginning, both training loss and validation loss decrease.
3. After a point, the validation loss starts increasing while training loss continues to decrease. This indicates overfitting.
4. Early stopping halts training at the epoch where validation loss is lowest

Early stopping can be viewed as implicit regularization. The effective number of training iterations T^* acts as a regularization parameter.

$$T^* = \arg\min_t \mathcal{L}_{val}(t)$$

The model then uses w_{t*} as the final parameters.