Regularisation Strategies in Multilayer Perceptrons (MLPs)

Parth Agrawal

December 2025

introduction-the-variance-bias-tradeoff

1 Introduction: The Variance-Bias Tradeoff

Machine Learning models are designed to predict and understand data that is unseen. However, usually during this process, we encounter two difficulties -

- **Overfitting:** This is when the model grasps the training data exactly instead of finding generalized patterns. This causes it to perform really poorly on unseen data. This usually happens in large models.
- **Underfitting:** This is when the model is too small and simple to capture the patterns and complexities of the training data, causing it to perform poorly on the test set.

A key point to note is that if a model overfits, it will achieve a high accuracy on the training data but a low accuracy on the test data. However, an underfit model will perform poorly on both.

Regularisation is a set of methods for reducing overfitting in machine learning models. Typically, regularisation trades a marginal decrease in training accuracy for an increase in generalizability.

Regularisation differs from optimization. Essentially, the former increases model generalizability while the latter increases model training accuracy. Both are important concepts in machine learning and data science.

parameter-norm-penalties-l1-l2

2 Parameter Norm Penalties (L1 & L2)

These techniques limit the capacity or learnability of the model by adding a parameter norm penalty $\Omega(\theta)$ to the objective function $J(\theta)$. This is usually used in linear regression-like problems.

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha\Omega(\theta)$$

Where $\alpha \in [0, \infty)$ is a hyperparameter.

l2-regularisation-ridge

2.1 L2 Regularisation (Ridge)

Ridge Regression is a method of estimating the coefficients of multiple regression models in scenarios where linearly independent variables are highly correlated. It drives the weights closer to the origin but rarely keeps them zero.

The penalty term is the squared Euclidean norm of the weights:

$$\Omega(\theta) = \frac{1}{2}\|w\|_2^2 = \frac{1}{2}\sum_i w_i^2$$

The gradient update rule becomes:

$$w \leftarrow w - \epsilon(\nabla_w J(w) + \alpha w) = (1 - \epsilon\alpha)w - \epsilon\nabla_w J(w)$$

(Where ϵ is the learning rate). The term $(1 - \epsilon\alpha)$ shrinks the weight vector at every step.B. L1 Regularization (Lasso)

l1-regularisation-lasso

2.2 L1 Regularisation (Lasso)

Lasso (Least Absolute and Selection Operator) regression performs an L1 regularisation, which adds a penalty equal to the absolute value of the magnitude of the coefficients. It pushes the weights towards exactly 0, including sparsity. This acts as feature selection, effectively removing noisy or irrelevant inputs.

The penalty term is the sum of the absolute values of the weights:

$$Omega(\theta) = \|w\|_1 = \sum_i |w_i|$$

The gradient includes the sign function ($sgn(w)$), applying a constant force subtracting from the weight regardless of its magnitude.

weight-decay-vs.-l2-regularisation

3 Weight Decay vs. L2 Regularisation

Weight Decay is often confused with L2 Regularisation. In reality, there is a subtle difference in different cases -

- **In standard SGD:** They are equivalent or same. Adding $\frac{1}{2}\|w\|^2$ to the loss function results in a gradient of αw. In the update step, this subtracts $\epsilon\alpha w$ from the weights, which is identical to explicitly decaying the weights. - **In Adaptive optimizers like Adam:** L2 regularization adds gradient of penalty to loss gradient. Adam scales this by the moving average of past gradients. However, weight decay is directly applied to the weights in the previous step, bypassing the adaptive scaling.

This causes L2 Regularisation to be less effective when using Adam compared to weight decay.

noise-injection-dropout

4 Noise Injection: Dropout

Dropout is a computationally inexpensive method for reducing overfitting by preventing the co-adaptation of training data.

During training, the model randomly drops out a fraction of data (sets it to zero). This prevents the model from grasping exact specifications.

It forces the network to grasp onto general patterns rather than specific points to reduce loss.

During training, the output y of a neuron is masked by a Bernoulli random variable r:

$$r \sim Bernoulli(p)$$

$$\tilde{y} = r * y$$

During testing, we do not drop units. To preserve the expected total input to the next layer, we scale the weights by the keep probability $(1 - p)$.

normalization-layers

5 Normalization layers

Normalization stabilizes the learning process and accelerates the rate of convergence. It regularises by introducing noise.

batch-normalization-batchnorm

5.1 Batch Normalization (BatchNORM)

In this method, we normalize the inputs to have zero mean and unit variance across the batch dimension.

For a mini-batch $\mathcal{B} = \{x_{1...m}\}$, we calculate:

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum x_i, \quad \sigma^2_{\mathcal{B}} = \frac{1}{m}\sum (x_i - \mu_{\mathcal{B}})^2$$

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

Scale and Shift: The network learns parameters γ and β to restore representational power: $y_i = \gamma \hat{x}_i + \beta$.

The mean and variance are calculated across the mini-batch, which contains some noise. This noise helps to prevent overfitting.

layer-normalization

5.2 Layer Normalization

It normalizes the features across the feature dimension of a single sample rather than across the batch. It is used in RNNs and Transformers, where batch dependency is not good.

$$\mu = \frac{1}{H}\sum_{j=1}^{H} x_j, \quad \sigma^2 = \frac{1}{H}\sum_{j=1}^{H} (x_j - \mu)^2$$

(Where H is the number of hidden units in the layer).

optimization-based—early-stopping

6 Optimization-Based - Early Stopping

Early stopping is the most widely used form of regularisation, as it requires no changes to the model architecture.

While training the model over multiple epochs, the accuracy on both the training set and the test set increases. However, after a certain point, the model overfits, i.e., the training accuracy continues to rise, but the accuracy on the test set begins to decrease. Early stopping prevents that from happening by stopping the training process when it overfits.

Let t be the number of training steps (epochs). We monitor validation error $E_{val}(t)$.

$$t_{stop} = targmin\ E_{val}(t)$$

Formally, this limits the volume of the parameter space reachable from the initialization point θ_0, effectively acting similarly to L2 regularization (constraining $\|\theta - \theta_0\|^2$).

sources

7 Sources

- IBM. (n.d.). *What is regularization?* Retrieved from https://www.ibm.com/think/topics/regularization
- Brownlee, J. (2020). *Understanding L1 and L2 regularization*. Towards Data Science. Retrieved from https://towardsdatascience.com/understanding-l1-and-l2-regularization-93918a5ac8d0
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Regularization for deep learning* (Chapter 7). In *Deep Learning*. Retrieved from https://www.deeplearningbook.org/contents/reg