CODE BASICS

DEEP LEARNING TAKEAWAYS

codebasics.io

Chapter:

Model Optimization: Regularization Techniques

What is Regularization

1. Regularization is a set of techniques used to prevent overfitting – which is when a model performs well on training data but poorly on new, unseen data
2. There are several techniques used for regularization such as dropout regularization, L1/L2 regularization, batch normalization, early stopping etc.

Dropout Regularization

1. Dropout regularization drops certain neurons in each hidden layer during the training process. This generalizes the model and stops the network from learning specific details of training samples
2. Dropout ratio can be 50%, 20%, 30% etc. based on the situation. In PyTorch one can add a dropout layer after activation layer using **nn.Dropout(p=0.5)** where p indicates the percentage of neurons being dropped out. 0.5 means drop 50% of the neurons.

L1 and L2 Regularization

1. Both L1 and L2 help prevent overfitting by adding a penalty to the cost function for large weights.
2. L1 regularization is also known as Lasso Regression
3. L2 regularization is also known as Ridge Regression

Batch Normalization

1. Batch normalization normalizes layer inputs to have zero mean and unit variance, enhancing model performance.
2. Allows for higher learning rates, reducing the training time for deep networks.
3. Adds robustness to model initialization, making it less sensitive to initial weights.
4. Reduces overfitting, particularly when used with dropout, by adding slight regularization effects.
5. Reduces overfitting, particularly when used with dropout, by adding slight regularization effects.

Early Stopping

1. Early Stopping monitors model performance on validation data to stop training when there is no further improvement for some fixed number of iterations.
2. This parameter of fixed number of iteration where there is no improvement is called **patience**
3. It prevents overfitting by halting training before the model starts to memorize noise.
4. Saves time and resources by avoiding unnecessary training epochs.