**SE4050 – DL**

**Lab 03**

The increase in validation error when the number of epochs is increased often indicates that the model is starting to overfit the training data. Overfitting happens when the model becomes too familiar with the training data and loses its ability to generalize to new, unseen data. This situation arises as the model continues to improve its performance on the training dataset to the point that it begins to capture noise or irregularities present only in that data.

* **Reduce Learning Rate:** The learning rate controls the size of the adjustments made to the model's weights during each training iteration. A smaller learning rate can slow down the learning process, preventing rapid convergence that might lead to overfitting. This measured approach allows the model to avoid getting stuck in small irregularities in the training data.
* **Apply Regularization Techniques:** Regularization techniques, such as L1 or L2 regularization, add a penalty term to the model's loss function. This encourages the model to favor simpler weight configurations and discourages it from memorizing the training data. By promoting smoother weight values, the model becomes less prone to overfitting.
* **Use Validation Set:** Utilizing a validation set involves setting aside a portion of the data exclusively for evaluating the model's performance during training. Unlike the training data, the model is not trained on this set. Instead, it serves as a gauge of how well the model generalizes to new, unseen data. By monitoring the validation error, you can identify the point at which the model's performance plateaus or starts to worsen. This guides you in determining when to halt training, preventing overfitting.
* **Early Stopping:** This technique is closely related to using a validation set. With early stopping, you monitor the validation error during training, and if it starts to increase consistently over several epochs, you stop training. This prevents the model from overfitting beyond the point where it's still performing well on unseen data.
* **Simplify Model Complexity:** If the model architecture is overly complex for the problem at hand, it can more easily memorize the training data. Consider simplifying the model by reducing the number of layers or nodes. A simpler model is less prone to overfitting.

The mini-batch Stochastic Gradient Descent (SGD) algorithm offers faster convergence compared to the traditional batch gradient descent. This is attributed to its more frequent updates of the model's weights during training. Unlike batch gradient descent, which updates weights only after processing the entire training dataset, mini-batch SGD updates weights after handling a smaller subset (mini-batch) of the training data.

**Batch Gradient Descent**

In batch gradient descent, the model accumulates the gradients of all training examples before performing a weight update. While this comprehensive approach gives accurate updates, it can be slow, particularly when dealing with substantial training data.

**Mini-Batch SGD**

Conversely, mini-batch SGD divides the training data into smaller batches. After processing each mini-batch, the model immediately updates its weights. This frequent updating injects dynamism into the learning process, allowing the model to learn more quickly. This speed-up results from the model's ability to react swiftly to new information in each mini-batch, steering it towards better weight configurations faster.