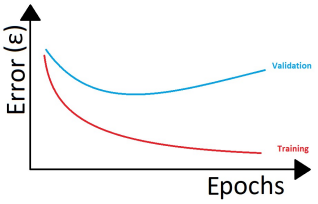
Introduction to Machine Learning Assignment 2

Matthijs Prinsen (s4003365)
Marinus van den Ende (s5460484)
Group 54

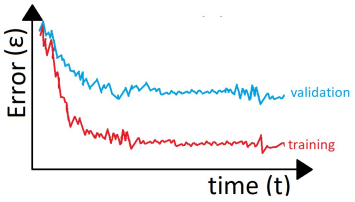
December 12, 2024

1 Error Analysis

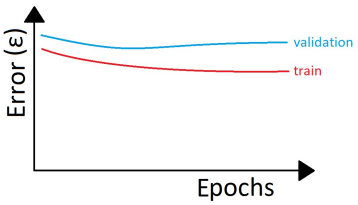
Overfitting(1a)(1b): We see an increase in the error over epochs in the validation data. The training dataset fits nicely, which we see due to its decrease in error.
Underfitting(1c)(1d): We see that neither the validation nor the training dataset fit particularly well. The error remains high and decreases very slowly. There does not seem to be an improvement in the reduction of errors.
Neither(1e): Both curves decrease over time and reach a stable threshold beyond which the model will not improve substantially.



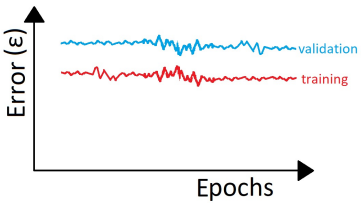
(a) Overfitting



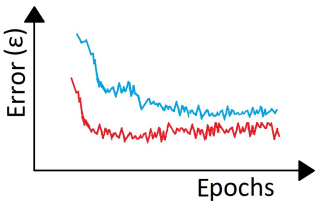
(b) Overfitting



(c) Underfitting



(d) Underfitting



(e) Neither

Figure 1: Training and Validation Errors Aggregated by Category

2 Dataset Splitting

Dataset splitting is crucial in training and testing a machine learning model. We need to split our dataset into three parts: train, evaluate and test. Our train dataset will be used to train our model. During the training of the model, we will use the evaluation(validation) dataset to fine-tune the hyper-parameters. This ensures that the model generalizes well and does not overfit the training dataset. After the model has been trained(on the training and evaluation data), we will use the untouched, test dataset to determine the actual performance of our model. The test dataset should be mutually exclusive from the training and validation data sets.

If, for some reason, you do not have a validation dataset, you cannot determine how well it generalizes during training and leads to a less generalizable model. This happens because there is no hyper-parameter tuning during training to more accurately capture the features of the training data.

Furthermore, if for some other reason, you do not have a test dataset and you test your model on the original training dataset your model will seem to be predicting values perfectly—bad practice. You cannot test something you made with the things you made it with, you have to test the model with unseen and new data, you want to make something useful that does not overgeneralize. Overgeneralizing is when the model fails to capture nuances in the data features.

3 Auxiliary task

Auxiliary tasks are tasks that are added next to the primary task to increase performance. The auxiliary task is a task related to the primary task. An example of such a relation would be to split off a partial task from the primary task and to make that the auxiliary task.

The additional task we would like to propose is an **edge detection** task. This would be auxiliary for the primary task of **object recognition**.

Object recognition is a known hard task for machine learning, to split up this task to include the auxiliary task should divide up the necessary calculations to make a simpler more effective model.

As explained in Lecture 8 - Neural Network, we could implement this task into a Convolutional Neural Network (CNN) by having one layer do **edge detection**. (Ahmed et al., 2019) [1]

References

Ahmed, S. M., Liang, P., & Chew, C. M. (2019). Epn: Edge-aware pointnet for object recognition from multi-view 2.5d point clouds. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3445–3450. https://doi.org/10.1109/iros40897.2019.8967705

[1] The paper describes a special form of edge detection, in which a point cloud is used. Here, it is particularly interesting to look at the number of points (as more points increase calculation time). "[This architecture] directly takes 3D points as input and creates clusters to aggregate features at different scales to generate global point cloud signature"