**Interpreting Training/Validation Accuracy and Loss**

[[Frederik vl](https://medium.com/@frederik.vl?source=post_page-----cf16f0d5329f--------------------------------)](https://medium.com/@frederik.vl?source=post_page-----cf16f0d5329f--------------------------------)

[Frederik vl](https://medium.com/@frederik.vl?source=post_page-----cf16f0d5329f--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F3b95e44a5f7c&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40frederik.vl%2Finterpreting-training-validation-accuracy-and-loss-cf16f0d5329f&user=Frederik+vl&userId=3b95e44a5f7c&source=post_page-3b95e44a5f7c----cf16f0d5329f---------------------post_header-----------)

4 min read

·

Nov 8

16

Interpreting training and validation accuracy and loss is crucial in evaluating the performance of a machine learning model and identifying potential issues like underfitting and overfitting.

**Accuracy**

Accuracy is calculated: correct\_predictions / all\_predictions. The following code is part of a typical Pytorch training loop. It is good practice to not only report accuracy, because the the value alone comes with some drawbacks.

'''  
 Inside the pytorch's training loop:  
 '''  
   
 #outputs: [batch\_size, nr\_classes] -> e.g.: [64,10]  
 \_, predicted = torch.max(outputs, 1)  
 #predicted: [batch\_size,1]  
 correct += (predicted == labels).sum().item() #.item() converts to an integer  
   
 total += labels.size(0)  
   
 train\_accuracy = 100 \* correct / total

**Problems with using accuracy only:**

In case of mistakes, we don’t now what kind of mistake it was. For instance, we wouldn’t know if its a false-positive or false-negative.

**In case of a very skewed data distribution,**

For instance when 1% of images in a Dataset are faces and 99% are non-faces. Then we could get a accuracy of 99% without ever classifying a face correctly by simply always predict the current sample is a non-face.

It is good practice to illustrate the classification accuracy in a confusion matrix. Each row represents the true class (ground truth), and each column represents the predicted class. The elements in the matrix provide information about the classifier’s performance for each class. In Figure 1, we see a confusion matrix with 10 classes. Row 0, shows the accuracy of class 0, and how often samples from class 0 were wrongly predicted with other classes.

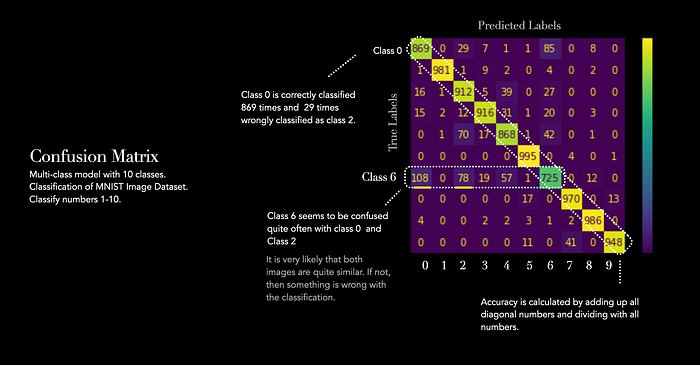


Figure 1: Confusion Matrix

**Loss**

In general, one distinguishes between training and validation loss. The training graph (depicted in green) indicates if the model is learning, while the validation graph (depicted in red) indicates if the model is able to generalise well on new data. One would like to see that error loss decreases steadily over time, while accuracy should increase. This suggests that the model is still learning from the training data and has not yet overfitted. When the loss plateaus, it suggests that the model has reached its maximum capacity to learn from the training data.

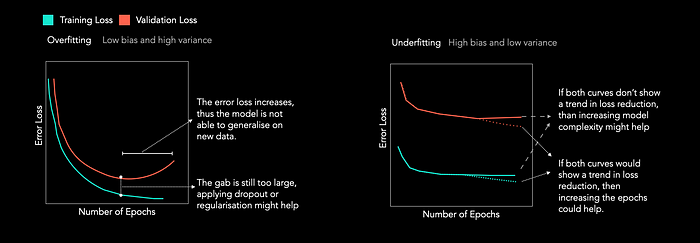


Figure 1: Training and Validation Loss (Figure by Author)

The two main techniques that are used to mitigate overfitting: Dropout and Regularisation. The following article explains how regularisation (L1 and L2) works:

**[Understanding Bias and Variance in Machine Learning](https://medium.com/advanced-deep-learning/understanding-bias-and-variance-in-machine-learning-5231dd117e12?source=post_page-----cf16f0d5329f--------------------------------" \t "_blank)**

[The terms bias and variance describe how well the model fits the actual unknown data distribution. In general one never…](https://medium.com/advanced-deep-learning/understanding-bias-and-variance-in-machine-learning-5231dd117e12?source=post_page-----cf16f0d5329f--------------------------------" \t "_blank)

[medium.com](https://medium.com/advanced-deep-learning/understanding-bias-and-variance-in-machine-learning-5231dd117e12?source=post_page-----cf16f0d5329f--------------------------------" \t "_blank)

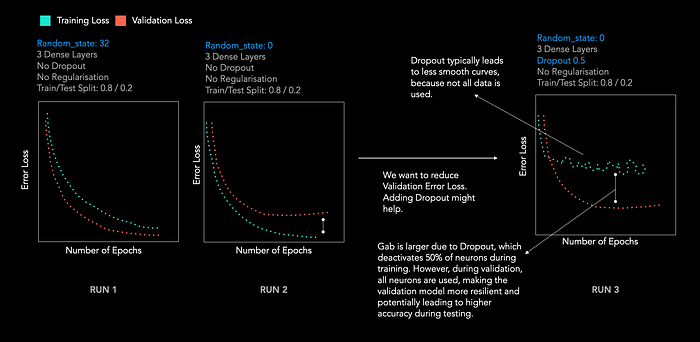


Figure 2: Changing Training/Test + adding Dropout

Let’s have a small experiment. In Figure 2, we see how the loss changes for the same model, with the same dataset for 3 different runs. Each run contains a slight change.

Run 1 demonstrates a lower validation loss than training loss, which often suggests that the training data is more challenging to model than the validation data. It’s possible that this model may not be suitable for effectively training the training data, so one can change the training/ test composition by reassigning the train/test split.

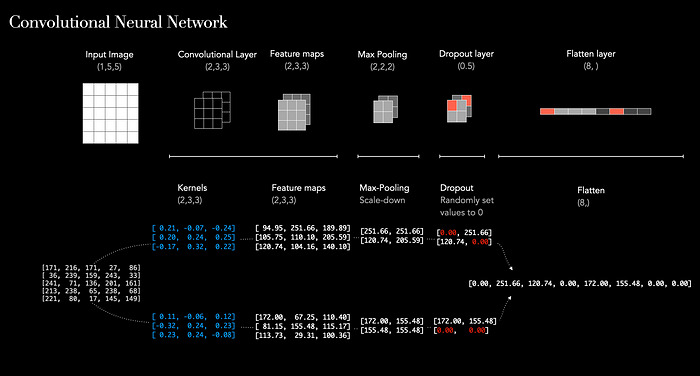
For Run 2, we only changed the **random\_state**of the training data. The random state regulates the data shuffling process before performing the split. Use an integer value to ensure consistent and reproducible results when the function is executed multiple times.

The dataset remains unchanged, but because we changed the **random\_state**, we are randomly selecting a different part as training data. This has led to a minor distinction between the training and validation sets.

In order to further reduce validation loss, we can add dropout. Dropout helps to generalise the model better by putting some values with a given probability p to zero. The probability p is a hyperparameter. We can see in Run 3 that the training loss is much more fluctuating after adding dropout.

**A quick reminder how dropout works:**

We take the output matrix of the previous operation, in this case, the MaxPooling Layer. We then set every value to zero, with a probability of 50% (that value can be changed).



[Machine Learning](https://medium.com/tag/machine-learning?source=post_page-----cf16f0d5329f---------------machine_learning-----------------)

[Loss](https://medium.com/tag/loss?source=post_page-----cf16f0d5329f---------------loss-----------------)

[Overfitting](https://medium.com/tag/overfitting?source=post_page-----cf16f0d5329f---------------overfitting-----------------)

[Accuracy](https://medium.com/tag/accuracy?source=post_page-----cf16f0d5329f---------------accuracy-----------------)

[Performance](https://medium.com/tag/performance?source=post_page-----cf16f0d5329f---------------performance-----------------)