LeNet CNN Classification

Maxim Khovansky

December 2024

1 Loss function

Since the problem described in the question is a classification problem, the standard and usually most appropriate loss function to use is **cross-entropy**. The general formula for mean cross-entropy, expressed in terms of true class distribution p and predicted class distribution q, is as follows:

$$L(p, q) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{x \in X} p_i(x) \log q_i(x), \tag{1}$$

where X is the set of all classes and N is the size of the data.

Note that, for classifiers, the ground truth is discrete, meaning that $p_i(x)$ is 1 for the true class (y_i) for data point i and 0 otherwise. Therefore, the formula can be simplified to the following, as all the terms in the inner sum except the term corresponding to the true class vanish:

$$L(y, q) = -\frac{1}{N} \sum_{i=1}^{N} \log q_i(y_i) \tag{2}$$

Moreover, the py.torch implementation of cross-entropy applies softmax to the raw logits s, or "class scores", to compute $q_i(x)$. In terms of class scores, the formula thus becomes:

$$L(y, s) = -\frac{1}{N} \sum_{i=1}^{N} \log(softmax_{y_i}(s_i)) \tag{3}$$

Due to the cancelling out of the log with the exponential inside the softmax function, the gradient of the loss function in (3) with respect to the class scores becomes simply:

$$\frac{\partial L}{\partial s} = q - p, \tag{4}$$

where q is the predicted class distribution and p is the true class distribution. Note that this gradient is identical to the gradient MSE (Mean Squared Error).

Cross-entropy computes the average amount of information about the true classes that is missing from the predicted class distribution. Naturally, the more information a model has about the data it tries to predict, the better its predictions will be. Cross-entropy is suitable for classification tasks because information is defined in terms of probabilities, which is precisely what classifiers output.

Given the above, it is intuitive that the gradient of cross-entropy loss with respect to the class scores is simply the difference between the predicted and true classes, as cross-entropy loss fundamentally measures the information difference between the predicted and true classes.

An alternative way to think about cross-entropy is that, in case of discrete ground-truth, it computes the sum of the likelihoods of each of the true classes given our model. Therefore, it can be though of as a form of MLE (Maximum Likelihood Estimator).

2 Performance analysis

The final training accuracy of the model was **0.9883** to 4 s.f., while the final validation accuracy of the model was **0.9050** to 4 s.f.

The accuracy and loss of the model throughout the epochs are plotted in Figures 1 and 2, respectively; both plots include training performance as well as validation performance.

From the two plots and the final accuracies, it is evident that the model is almost certainly overfitting: the final training accuracy is significantly higher than the final validation accuracy, and the loss per epoch curve is typical of an overfitting model - the validation loss went down for the first 5 or so epochs, but then started increasing, all the while the training loss continuously decreased throughout the entire training process.

In order to address the overfitting issue, I planned to add dropout layers to the model and potentially use other regularisation techniques, such as L1 or L2 regularisation on the weights; however, I wasn't sure if this was allowed, as Dr. Iran advised strongly against changing the architecture of the model (link: https://qmplus.qmul.ac.uk/mod/forum/discuss.php?d=559932), so I kept the model largely unchanged.

As for performance change speed, it steadily decreased throughout training; validation accuracy appears to have stabilised around epoch 5, while training accuracy and training loss kept improving, but at an ever-decreasing rate. The only exception to this trend was validation loss, which had a pretty stable rate of change ever since epoch 5; however, as mentioned above, this is almost certainly due to overfitting. The decreasing rate of change in performance is an indicator that the convergence of the model was efficient.

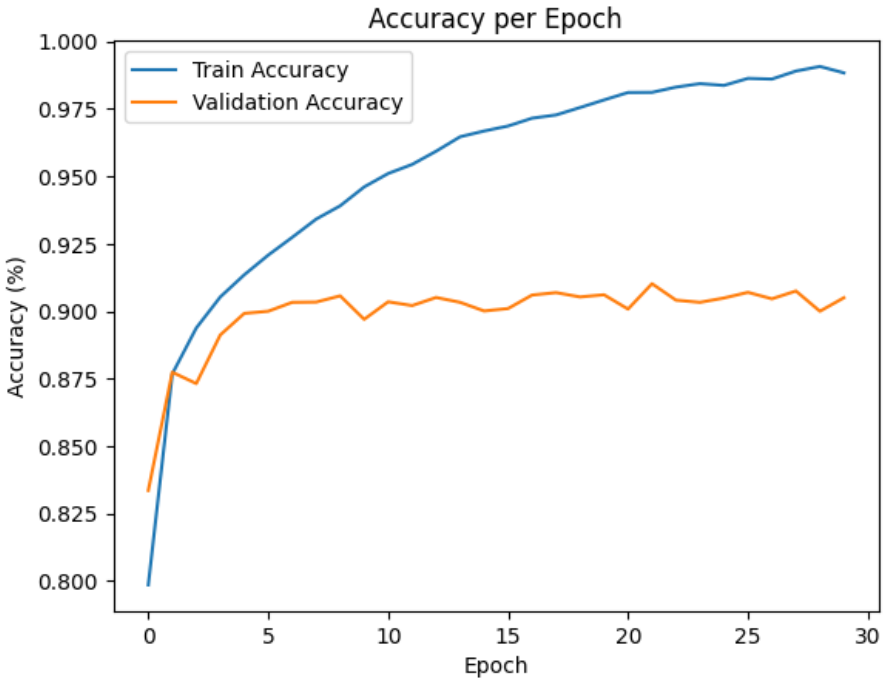

Figure 1: Accuracy throughout the epochs

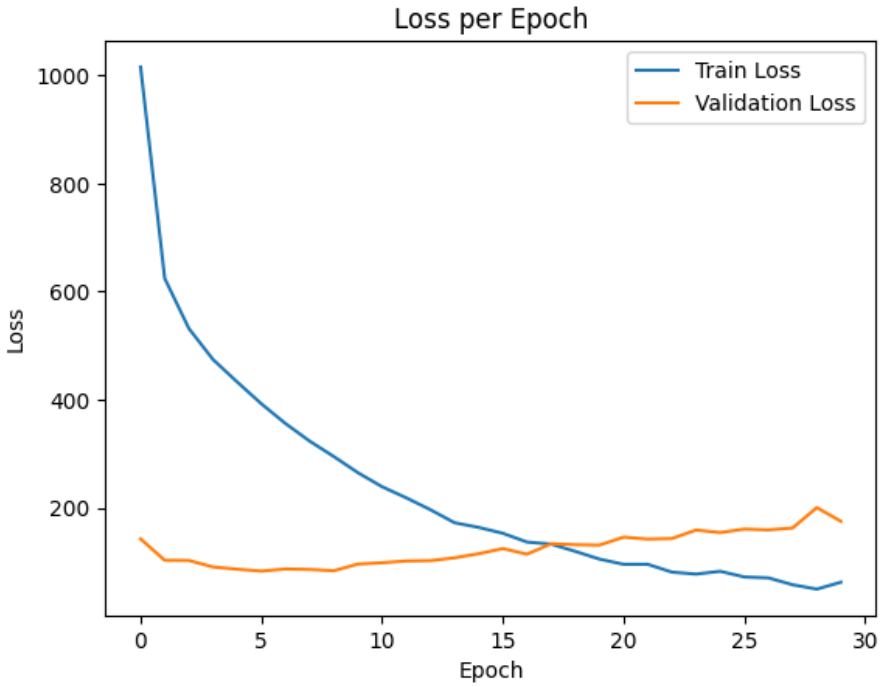

Figure 2: Loss throughout the epochs