Advanced Deep Learning Assignment 3 pwn274

1 The Implementation

The pre-training:
Initially I made three different pre-training targets based on transformations seen in the lectures. These are: Rotations, noise detection and detecting whether a digit is actually two different MNIST-pictures on top of each other. After having implemented these without any hurdles except for the fact that they made no improvements on subsequent training, I also wrote a pre-training mode where the model switched between these targets. This did the least amount of damage to the models training, allowing it to converge in about the same way as the models that where not previously trained.

I tried on both the standard MNIST dataset, on four different languages from the MIX-MNIST data set, on a combination of the data sets and finally on the MNSIT fashion data set.

The learning graphs can be seen in figure 1.

None of the implemented pre-training targets made any actual improvements regardless of model size, training time, pre-training time, model architecture, batch-size, learning rate, data type, amount of data, training time, pre-training target, number of targets, choice of output layer, number of frozen weights and so on. If you cannot tell, this was quite a grueling, week-long hyper-parameter search. The only thing that did not change the model for the worse was having very short (sub-epoch) training on multiple, changing targets.

Pros and cons:
Rotating the input seems to be too simple for the model to learn any features of the data.

Introducing noise: I would have imagined could be useful as it will learn some of the features. Here I surmise that MNIST is a far too simple data-set for the pre-training to have any effect.

Combining two pictures: I would also have guessed could be useful. Here, once again, simply training any model is still

d) all of the above: Here the the model will learn some features of the data without actually learning the features of the pre-training targets. This has proved to be preferable to the others, but still not better than simply training the model for long enough.

2 Conclusion

To summarize: No matter the amount of work I have put into pre-training on the MNIST data set, it has not improved the actual subsequent training in any way. The only thing pre-training amounted to, was confusing the model into converging slower or tricking it into a sub-optimal minimum thereby making it not converge at all.

Intuitively this makes some sense, as both the objective (10-way classification) and the models architecture are quite simple compared to the huge NLP-models that pre-training is mainly used on.

I really hope I was not supposed to conclude that pre-training was some magical unambiguous quality-improver.

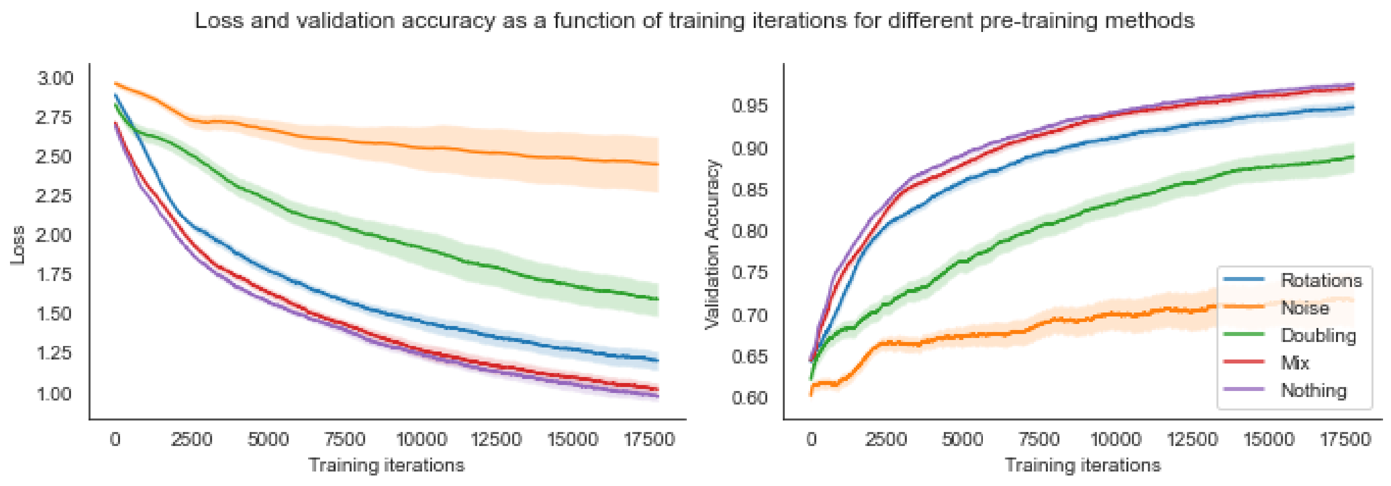

Figure 1: Pre-training consisted of 2 epochs. Downstream test-accuracies: 0.97, 0.97, 0.95, 0.86, 0.69