Hypothesis

Larger networks have more weights, meaning each gradient update changes more parameters simultaneously. I expect that large networks are more sensitive to high learning rates, requiring a smaller learning rate to train stably compared to smaller networks.

The experiment

Testing the influence of learning rate on network size. The following configurations are choosen.

Small = 64-64 ± 55k parameters
Medium = 256-256 ± 270k parameters
Large = 1024-1024 ± 1800 k parameters

Batch size = 64 -> 937 batches per epoch

Number of epochs = 50

Learning rates 1 = [0.01, 0.001, 0.0001]
Learning rates 2 = [1.0, 0.1, 0.01]

Optimizer = SGD to ensure the learning rate remains a controlled independent variable. Adam and others change the learning rate internally.

Scheduler = None, the scheduler also changes the learning rate, therefore it is disabled.

Loss function = CrossEntropyLoss

Results

Testing with the initial learning rate range (0.01 to 0.00001) revealed that 20 epochs was insufficient for the models to converge, so this was increased to 50 epochs. At 50 epochs, most configurations converged, with the exception of the very lowest learning rate (0.00001), which was still progressing but too slow to be practically useful. Importantly, none of the configurations in this range showed unstable training, meaning this range was too conservative to test the hypothesis.

A second set of learning rates (1.0, 0.5, 0.1, 0.01) was therefore used to push the models into a regime where instability could occur and the effect of network size on learning rate sensitivity could be observed.

At a learning rate of 1.0, all three networks failed to learn meaningfully. The large network collapsed entirely, outputting NaN for every epoch. The small network stagnated at a loss of 2.3 across all epochs, which corresponds to random chance for a 10-class classification problem ($\log(10) \approx 2.3$), meaning it learned nothing. The medium network showed some movement, reaching a loss of 1.7, but this is still far from useful performance. A learning rate of 1.0 is too aggressive for all network sizes.

At a learning rate of 0.1, all three network sizes show meaningful learning. Training loss improves steadily across all models. However, test loss decreases initially but begins rising after around epoch 20 and does not recover, a classic sign of overfitting. The effect is most pronounced in the largest network, which ends with the highest test loss of the three at around 0.43, compared to 0.40 for the small network. This is expected behavior: larger

networks have higher capacity and are therefore more prone to memorizing the training data when no regularization is applied. The unsmoothed loss curves also reveal more erratic behavior in the larger network, with more pronounced spikes throughout training. This could be an early indication of learning rate sensitivity.

At lr=0.01 the picture is considerably cleaner. Test loss decreases smoothly for all three networks without the upward turn seen at lr=0.1, and while some minor spiking is still visible in the unsmoothed curves, it is far less pronounced than at the higher learning rate. All three network sizes perform very similarly, with final test losses clustered tightly between 0.33 and 0.36, and the larger network is marginally the best of the three.

Conclusion

The results do not support the hypothesis. All three network sizes failed at lr=1.0 and showed overfitting at lr=0.1, regardless of the number of parameters. At lr=0.01, all three converged to roughly similar performance. This suggests that the learning rate thresholds were similar across network sizes, meaning larger networks did not require a meaningfully lower learning rate than smaller ones.

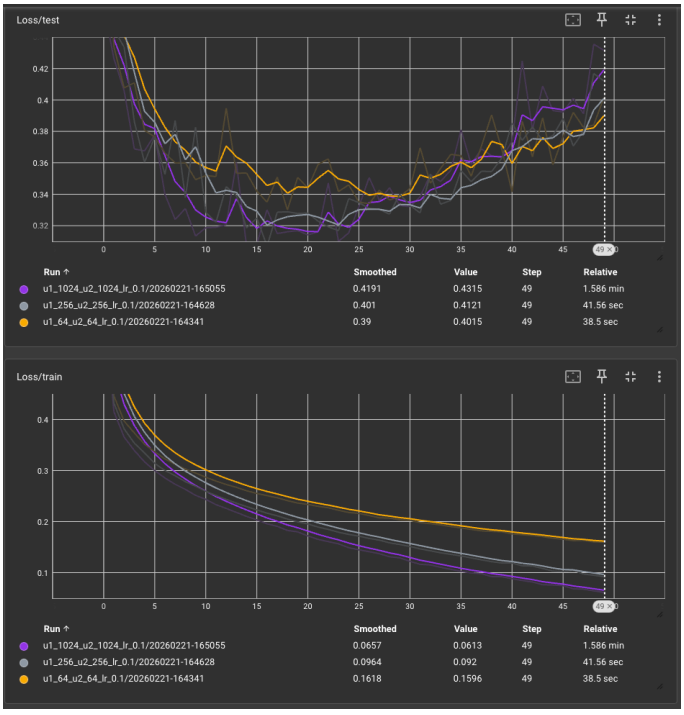

Fig 1. Train and test loss @ lr-0.1

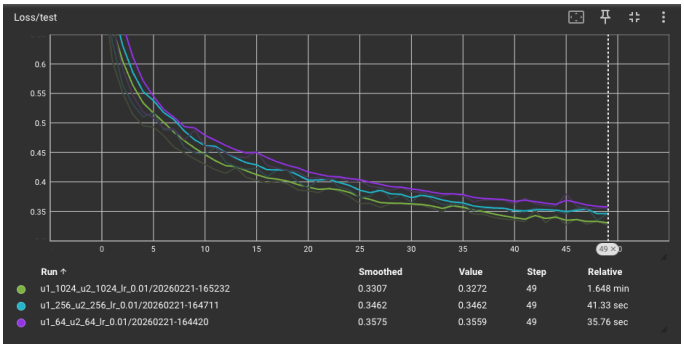

Fig 1. Test loss @ lr-0.01