

Links to presentation(s) and code(s) on GitHub

### 📄 Presentation 3 - STAT 390

[Code Link](#)

- What did you do?

1. I learned how to run the batch files on Quest. I initially faced issues with my Python version and attempted to edit the batch file to fix, but the TAs resolved these issues. Therefore, I could submit jobs to run the train and test files of the model for both 5 and 20 epochs.
2. To tune learning rate, I initially tried 5 learning rates with 5 epochs, but performance was not adequate. Then, I ran the same 5 learning rates with 20 epochs.
3. ChatGPT showed that using a Scheduler LR would work better for a MIL model. Therefore, I added code to the trainer to create a One Cycle Learning Rate Scheduler. I could not find any papers that prove this method is better than a fixed LR for MIL, but I believe that the code is simple and worth trying.

- How does it help the project?

Learning how to run the code on Quest allowed me to fully run for 20 epochs in ~6 hours, which is much quicker than Google Colab. Tuning learning rate increased performance, and provides a baseline to continue tuning at higher learning rates. I will continue trying to find the optimal learning rate, whether it is fixed or variable based on the Scheduler LR code.

- Issues faced (if any)

The main issue was getting the files to run on Quest, but that has been resolved. Although the models now take ~6 hours to run, Quest is often busy during the day and the file takes a couple more hours waiting in the queue.

- Attempts to resolve issues (if any)

The mentor meeting and help from the TAs resolved my code running issues. I initially used ChatGPT to try many different batch file variations to resolve my issue.

- Issues resolved (if any)

The code now properly runs on Quest.

- Next steps

1. Test additional learning rate values at 20 epochs.
2. Run the code for the Scheduler LR. Based on these results, I will determine if I should continue tuning a fixed LR or follow this method.
3. Implement Noah's idea of adding additional on-linear layers before the final linear layer.

- References (Mention if you built up on someone else's work)

ChatGPT

Used Jack's MIL Trainer [linked here](#)