Tuning Memo

After changing the batch size each time the program ran, I received results from each epoch. I ran 20 epochs of each batch size: 8, 16, and 32.

Something important to take note of is as the batch size increased, the accuracy of the program decreased. For example, batch size 8 had the highest accuracy at 52.3% (more testing would increase this number). We want our accuracy to be as close to 100% as possible to ensure the classifications and annotations of our tool are useful. A speedy yet inaccurate tool is not optimal for our goals of identifying certain objects on the battlefield.

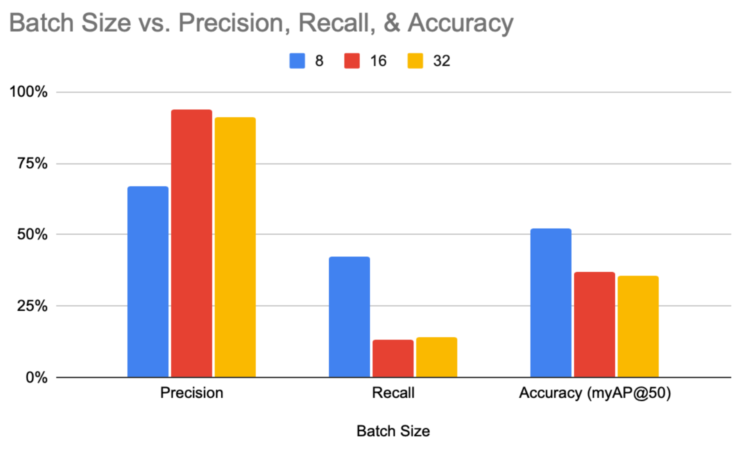
In addition, larger batches take more time to complete than smaller batches. Training time is essential to our program because we want to be able to provide the most accurate results as quickly as possible.

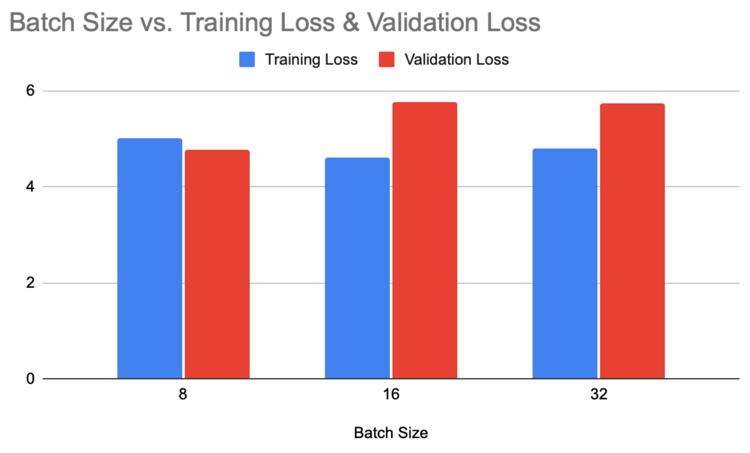
Batch size epoch times:	Total (20 epochs)	Avg (sec/epoch)
Batch size 8:	1199.42s	59.97s
Batch size 16:	1228.45s	61.42s
Batch size 32:	1668.41s	83.42s

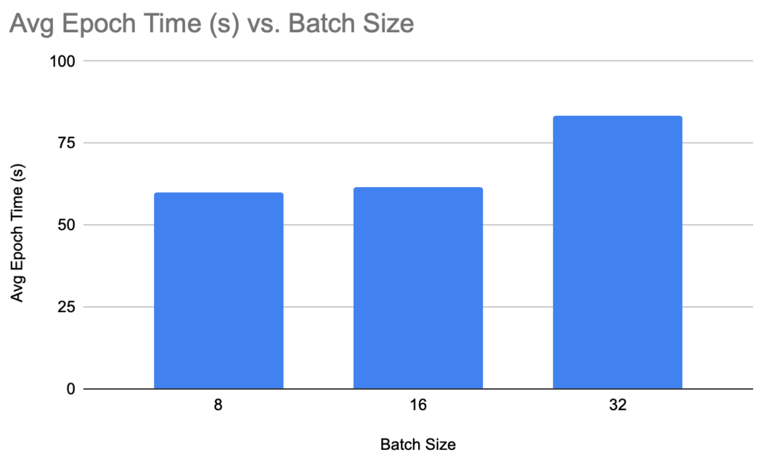
Precision and recall are important metrics in ML for evaluating the performance of classification models and datasets. Also, loss is an important metric to follow because it guides the optimization process. The batch size with the most precision was size 16, best recall was size 8, and least loss was size 8.

Conclusion:
We want to optimize accuracy and speed while keeping in mind these other metrics. As the batch size increases, some metrics improve and others worsen. So, we need to find the perfect balance between accuracy and speed for our classification tool. In the end, **I believe batch size 8 is the best choice** because it has the highest accuracy, speed, recall, least loss, and best generalization.

Charts of Data/Findings:


Batch Size vs. Precision, Recall, & Accuracy
Precision
Recall
Accuracy (myAP@50)
Batch Size


Batch Size vs. Training Loss & Validation Loss
Training Loss
Validation Loss
Batch Size


Avg Epoch Time (s) vs. Batch Size
Avg Epoch Time (s)
Batch Size