* Adamw most versatile and highest accuracy
  + Cosine annealing scheduler standard for this type of architecture
* Reducing token size to a max of 256 (with padding if needed) speeds up training quadratically
  + Byte pair encoding
  + Model accepts 1024, but due to the nature/domain bias of sentiment analysis, we can exploit that sentiment is apparent early on. By shortening token size, we can speed up training (which means we can use more data too), increase generalization (compared to high token size) since tokens are more generic, and increasing final accuracy as a result.
  + GPT-2 scales quadratically with sequence length. Shorter length = dramatically faster.
* Decreasing training data from 40k (50k) to 12k (15k) had little effect on final accuracy.
  + Helped make training times manageable whilst maintaining statistical distribution and diversity
* As token size increases, batch size needs to decrease for CUDA memory
  + Slight increase in accuracy (less than 2%), extremely worse training time (20 mins vs 80 mins), extremely unstable loss curve.
* Bigger model (more parameters) + More tokens = less batch size to accommodate infrastructure
  + Small batch size = high variance = noisy loss curve
* Lora fine tuning
  + Needs more epochs (2) to hit similar accuracy to full model fine tuning as it likely needs more time to converge as only a small subset of the model is available for optimizations.
    - Increasing epoch from 1 to 2 resulted in 4% increase (88% to 92%)!
  + More transformers = higher accuracy, negligible increase in training time
    - More transformer blocks results in sharper early peak in loss
    - Most accuracy gains happens in last 4 transformers blocks, going up to last 24 only makes accuracy go from 87 to 92%, while going from 1 to 4 goes from 68% to 87%
      * Must mean that later layers have the strongest influence on finetuning ability for specializing model to domain
  + Higher rank = slightly higher accuracy, negligible increase in training time
    - The slight difference in accuracy, although improved, might actually not be “worth it” for the amount of extra compute it requires.
* Higher batch size = better accuracy, but not past 32
  + Cant get past 32 in 99% of cases anyways due to VRAM
* Warmup not helpful, higher accuracy without it
* More cases of overtraining on loss curves with large GPT2 model
  + Likely because overparameterized, especially for my dataset
* LoRA on small GPT2 model suffers in accuracy in some cases, maybe because taking subset of smallest model bottlenecks trainable parameters intensely causing underfitting.