Evaluating different training methods on QA datasets

Rushabh Musthyala rm6416@nyu.edu
Charvi Gupta
cg4177@nyu.edu

Executive Summary

- Problem Statement - For different QA datasets, what is the best training strategy for a large language model with limited resources.

- Solution Approach - fine tune a t5 model for a question-answering task on BioQA and GSM8k datasets and evaluate the performance and training metrics across different training strategies

- Value - we aim to figure out what is the best method for completing such tasks keeping in mind various factors

Problem Motivation

- As the field of DL grows, deciding *how* to train a model has become a difficult question to answer as hardware requirements continue to grow at a rapid rate
- In particular, sequence-to-sequence language models are known for having a huge number of parameters and for taking a long time to train
- There are also a plethora of GPUs and training techniques available
- Deciding which one to use can be a difficult task

Background Work

- T5 - Text to Text Transfer Transformer[1]
  - Teacher Forcing technique - always need input sequence and corresponding output sequence
  - achieves SOTA results on many NLP benchmarks
  - flexible enough to be fine-tuned to a variety of important downstream tasks
- Datasets -
  - BioQA - Biomedical QA dataset containing - a question (Q), human-annotated answers (A), and the relevant contexts (C)


```
{'qas': [{'id': '52bf208003868f1b06000019_002',
   'question': 'What is the inheritance pattern of Li–Fraumeni syndrome?',
   'answers': [{'text': 'autosomal dominant', 'answer_start': 213}]}],
 'context': 'Balanced t(11;15)(q23;q15) in a TP53+/+ breast cancer patient from a Li-Fraumeni syndrome family. Li-Fraumeni Syndrome (LFS) is characterized by early-onset carcinogenesis involving multiple
tumor types and shows autosomal dominant inheritance. Approximately 70% of LFS cases are due to germline mutations in the TP53 gene on chromosome 17p13.1. Mutations have also been found in the CHEK2 gene
on chromosome 22q11, and others have been mapped to chromosome 11q23. While characterizing an LFS family with a documented defect in TP53, we found one family member who developed bilateral breast cancer
at age 37 yet was homozygous for wild-type TP53. Her mother also developed early-onset primary bilateral breast cancer, and a sister had unilateral breast cancer and a soft tissue sarcoma. Cytogenetic
analysis using fluorescence in situ hybridization of a primary skin fibroblast cell line revealed that the patient had a novel balanced reciprocal translocation between the long arms of chromosomes 11 and
15: t(11;15)(q23;q15). This translocation was not present in a primary skin fibroblast cell line from a brother with neuroblastoma, who was heterozygous for the TP53 mutation. There was no evidence of
acute lymphoblastic leukemia in either the patient or her mother, although a nephew did develop leukemia and died in childhood. These data may implicate the region at breakpoint 11q23 and/or 15q15 as
playing a significant role in predisposition to breast cancer development.'}
```


  - GSM8k - Dataset of 8.5K high quality linguistically diverse grade school math word problems.

Problem: Beth bakes 4, 2 dozen batches of cookies in a week. If these cookies are shared amongst 16 people equally, how many cookies does each person consume?
Solution: Beth bakes 4 2 dozen batches of cookies for a total of 4*2 = <<4*2=8>>8 dozen cookies
There are 12 cookies in a dozen and she makes 8 dozen cookies for a total of 12*8 = <<12*8=96>>96 cookies
She splits the 96 cookies equally amongst 16 people so they each eat 96/16 = <<96/16=6>>6 cookies
Final Answer: 6

1 C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *arXiv.org*, 28-Jul-2020. [Online]. Available: https://arxiv.org/abs/1910.10683.

Technical Challenges

- Larger models take more time to train -
  - Simplified by increase batch size

- Larger model (3 Billion parameters) could not fit into GPU memory
  - Faced OOM issues - did not work
  - Used variation of t5 model with 770M params

- GSM8k - Expectation: Break down mathematical question into multiple reasoning steps in the answer
  - Simplified by predicting the final answer only

Approach

Basic Training Framework -

- Define LightningModule - organize pytorch.nn.Module code into sections - initializations (loading a pretrained model), Train Loop (training_step), Validation Loop (validation_step), Test Loop (test_step), Optimizers and LR Schedulers (configure_optimizers)
    - Pretrained model **t5-base : 222M params**
    - Pretrained model **t5-large: 770M params**
    - Pretrained models **t5-3b: 3B params** (failed to load in memory)
- Define datasets - BioQA data loader and GSM8k data loader
    - BioQA - Training size (~13000 instances, reduced to 2500 after preprocessing)
    - GSM8k - Training size (~7000 instances)
- Define Trainer - allows mixing any LightningModule with any dataset
- Train model
- Record training data and validation performance

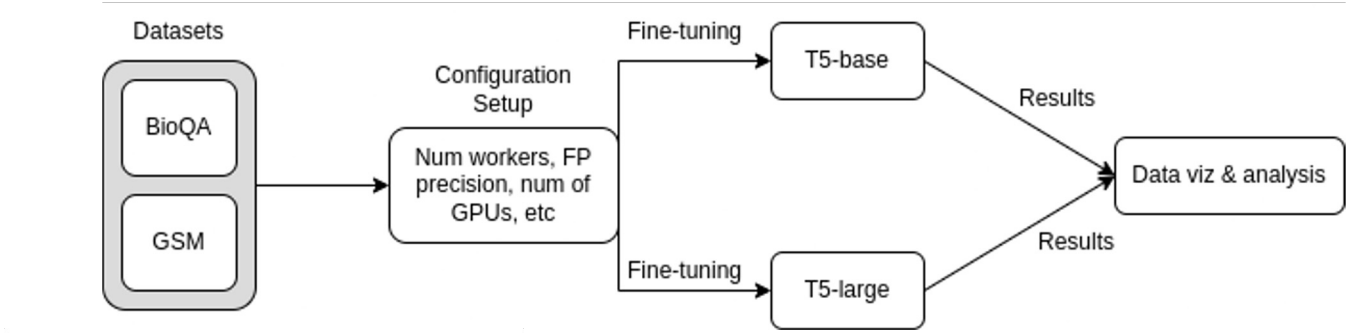
Approach

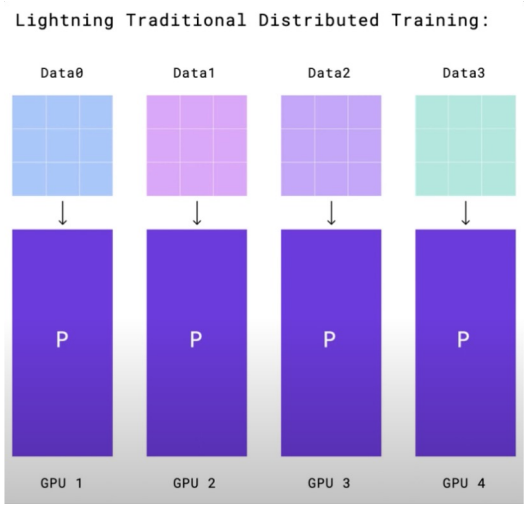
Experimental Training Framework -

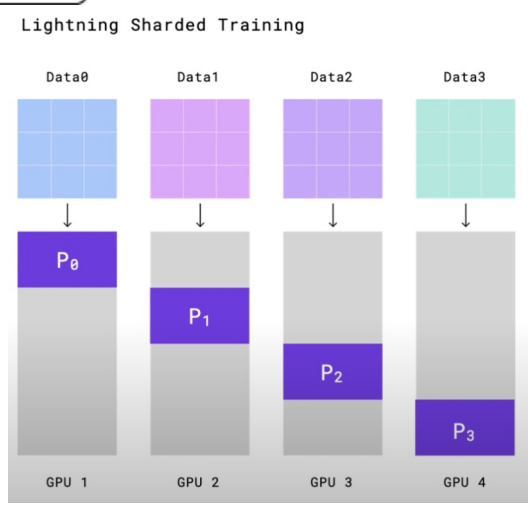
- We early stop at epoch 20 to trade computation resources per experiment for more experiments (12 experiments in total, 6 for each dataset)
- Data Parallelism - Trainer Strategy - ddp
  - Each GPU gets visibility into a subset of the overall dataset.
  - Each process performs a full forward and backward pass in parallel.
  - The gradients are synced and averaged across all gpus.
- Model Parallelism - Trainer Strategy - ddp_sharded
  - optimizer states, gradients and parameters are sharded across GPUs.

Model number	Number of workers	Number of GPUs	FP precision	Strategy
1 (t5-base)	1	1	32	-
2 (t5-base)	2	1	32	-
3 (t5-base)	2	1	16	-
4 (t5-base)	2	2	32	DP
5 (t5-base)	2	2	32	Sharded
6 (t5-large)	2	2	32	Sharded

Solution Diagram / Architecture







Implementation Details

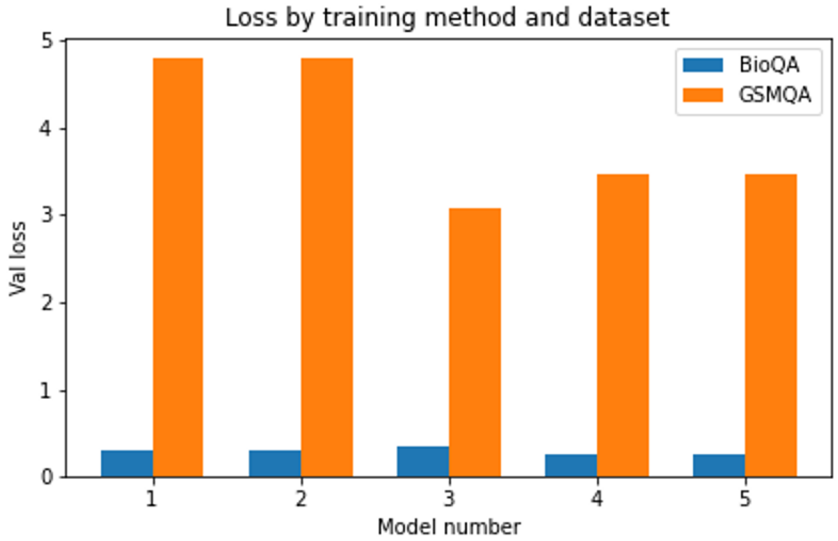
- Models - Huggingface library[1]
- Pytorch Lightning[2] - Lightweight pytorch wrapper that abstracts boilerplate Deep Learning code
  - Readability, Robustness, Hardware agnostic
- Training Strategy - distributed data parallelism and sharded training
- Training Hardware - 1-2 Nvidia RTX8000 GPUs on NYU HPC Greene Cluster
- Visualization and Metrics Analysis - Matplotlib, Numpy and Pandas

1 https://huggingface.co/docs/transformers/model_doc/t5
2 https://www.pytorchlightning.ai/

Experimental Design Flow

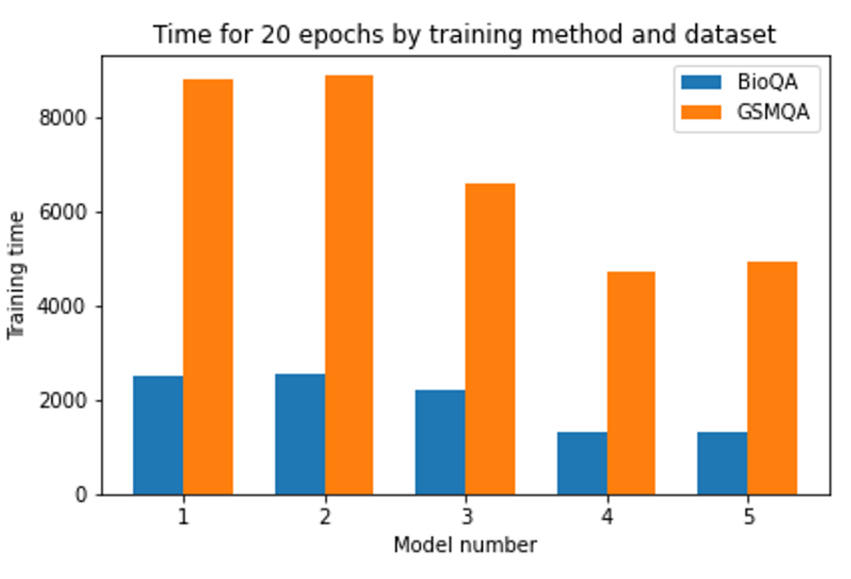
- Minimum loss (to see how well the model converged)
- Total training time
- GPU utilization
- GPU temperature (since high temperatures can damage hardware and reduce their lifespan)
- Visualizing scaling efficiency
- Plotting potential Pareto Curves

Validation Loss



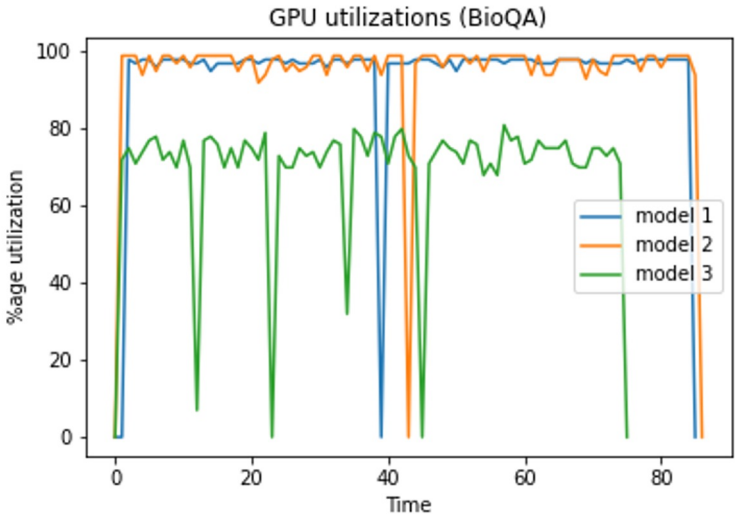
- For the BioQA dataset, we get relatively consistent values of loss for all configurations, with an increase for model 3 which is to be expected given we're working with lower precision
- Surprisingly, we see the opposite of this for the GSM dataset

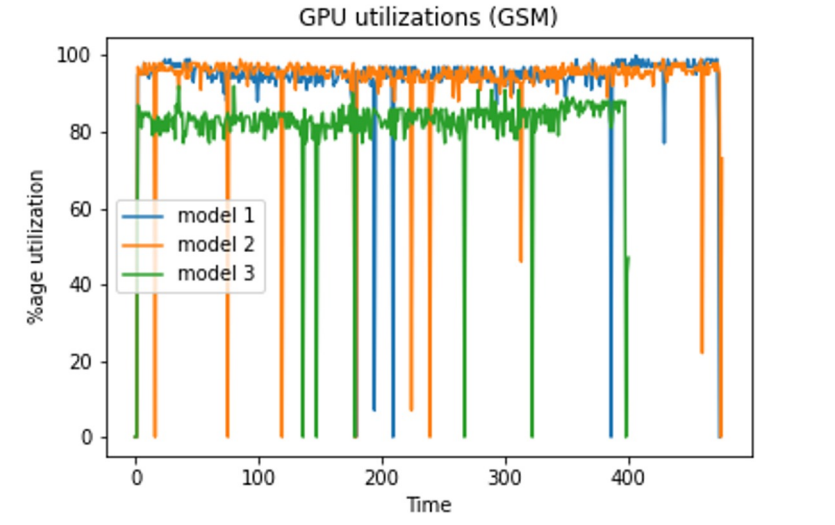
Training Time



- We observe similar trends across both datasets
- Training with lower precision was significantly faster
- Taking advantage of 2 GPUs (4, 5) also helped speed up performance

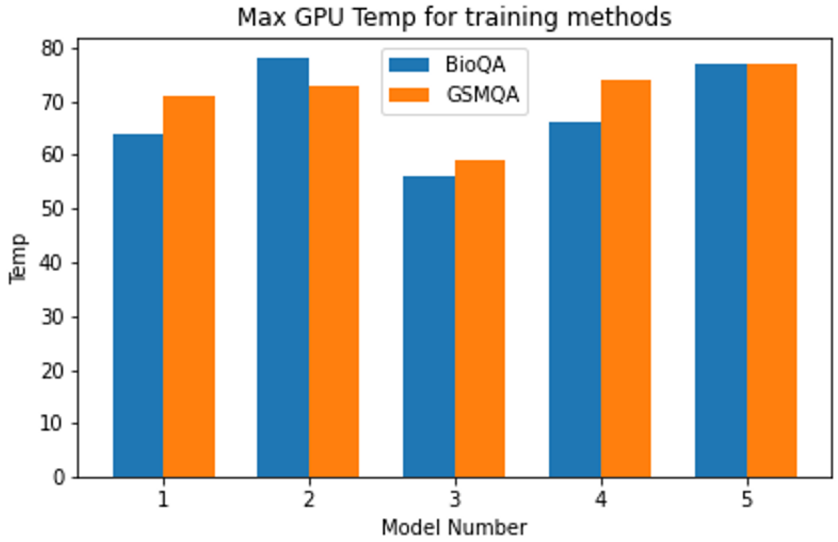
GPU Utilization





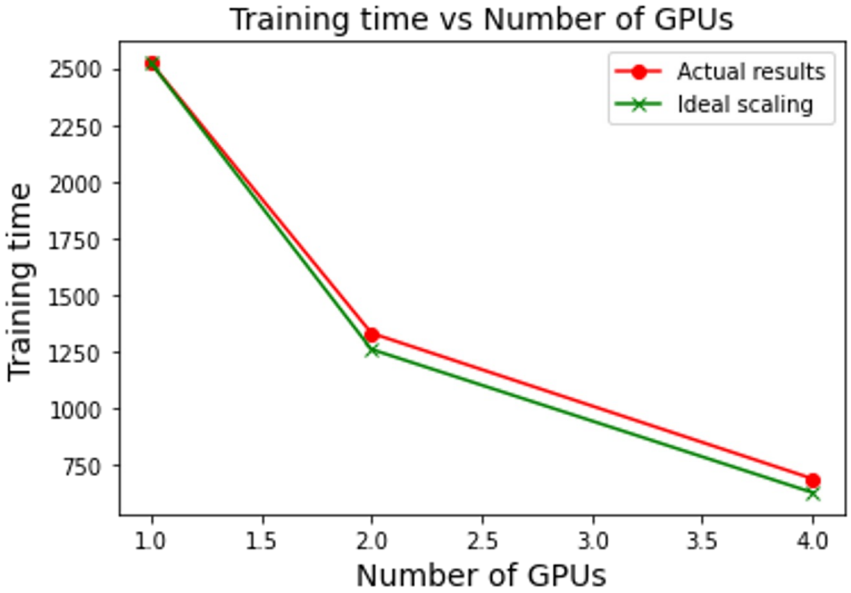
- Performing an identical task on 2 GPUs reduced the average utilization on each GPU -> we can increase the model size and effective batch size in search of better performance
- Reducing FP precision also reduced the GPU utilization

Max GPU temperature



- We observe similar trends across both datasets
- Training with lower precision led to lower temps

Scaling Performance

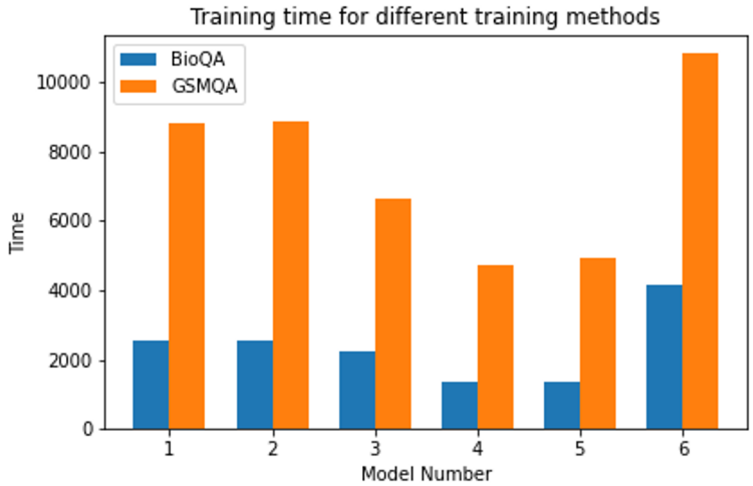


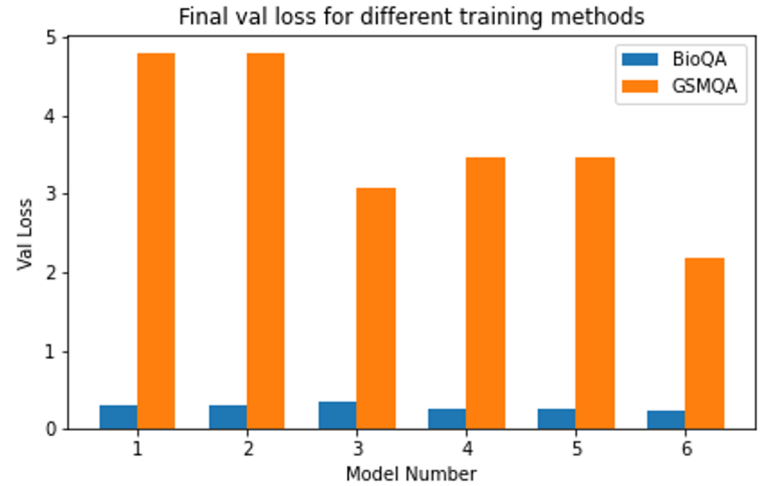
- On the BioQA dataset we observed almost ideal scaling
- The difference between actual and ideal will increase as model size and batch size increases

Training an even larger model

- Throughout our experiments, we were still able to fine-tune a t5-base model (220M parameters) on a single GPU, albeit slowly
- The ability of distributed training is better exhibited by training models that are too big to run on a single GPU
- To this end, we fine-tuned the t5-large model (770M params) on the BioQA dataset using model parallelism
    - This failed when attempted on a single GPU

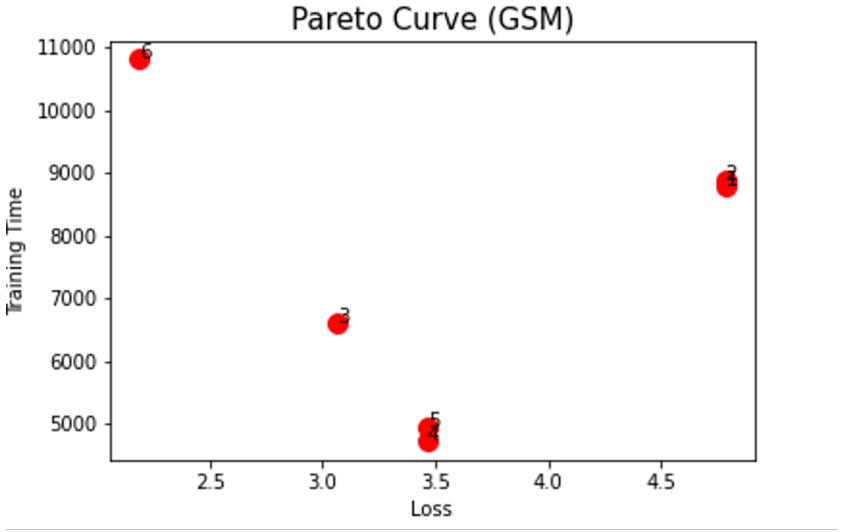
T5-large results

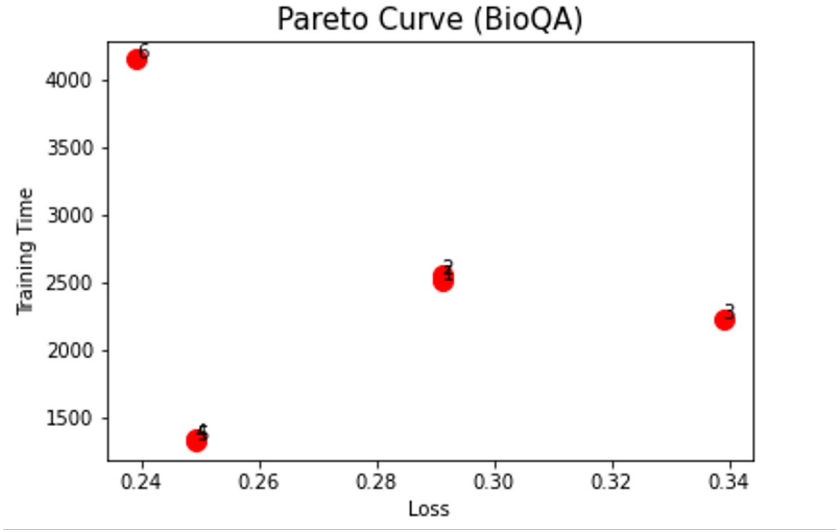




- By training a larger model - we were able to converge to a lower loss in the same number of epochs
- However, this comes at the expense of increased training time

Pareto Curves





- For BioQA, quite clear that it is optimal to use some form of distributed training and reducing the FP precision is very detrimental
- In GSM, reducing FP precision can also help improve training time and val loss

Conclusion

Optimization	Description	Results (Speed)	Results (Loss)
Increasing the number of processes for data loading	Changing num_workers to 2 from 1	N/A	Dependent on type and structure of dataset, no real change observed here
Lower precision training	Changing all the values from FP32 to FP16	1.3x faster for GSM 1.15x faster for BioQA	1.15x worse loss for BioQA Lower loss for GSM
Data Parallelism	Splitting the mini-batch across different GPUs	1.85x faster for GSM and BioQA	Lower losses for both of them too
Sharded	Splitting the model across different GPUs	1.8x faster for GSM 1.9x faster for BioQA	Lower losses for both of them too

Github Repo Link

https://github.com/gcharvi31/t5-tuning