Our goal is to compare training times and BLEU scores after using mixed precision, distributed training, and gradient accumulation during fine tuning of a bilingual sentence translation. We use the HuggingFace Seq2SeqTrainer to easily apply optimizations on the SMaLL-100 and T5-small transformer models as well as compare speedups and accuracy within each model after optimizations.

Directory:

* Requirements.txt
* Translation.py (small100model)
* T5\_translation.py (t5 model)

Example command: python3 t5\_translation.py

T5\_results:

A black screen with white text

Description automatically generated with low confidence

A black screen with white text

Description automatically generated with low confidence

A black screen with white text

Description automatically generated with low confidence

A black screen with white text

Description automatically generated with low confidence

We noticed that increasing batch size often led to CUDA out of memory issues. Additionally the combination of multiple optimizations, as expected led to faster training than one on its own. Batch size increase resulted in the greatest loss of accuracy and moderate speedup. It seems the best combination to maintain relatively high accuracy while benefiting from faster training time would be to run the model with both distributed data parallel and mixed precision.