Coding Challenge: In-Silico Perturbation Optimisation

Objective

One of our multiple use-cases with Foundation Models is In-Silico Perturbations. In-Silico Perturbations consist of small variations of the expression data (usually over or under-expressed/deleted genes), that result in changes of the latent space. One common example is shown in the Geneformer paper (Fig.2-d) (https://pubmed.ncbi.nlm.nih.gov/37258680/). The amount of perturbations can increase quite significantly and the bigger the model, the longer the inference will take, directly impacting cost and applicability.

Background:

- Helical Repo: https://github.com/helicalAI/helical
- Geneformer Paper (https://pubmed.ncbi.nlm.nih.gov/37258680/)
- Other Papers are listed as part of the Helical Package

This challenge aims at improving the one model in the helical package to run more efficiently over a large amount of perturbations (i.e. inference).

Task Details:

1. **Baseline & Profiling**
   - Pick one model from the Helical package.
   - Run a baseline inference on a set of perturbations (e.g., 100–1k).
   - Record runtime, GPU/CPU utilization, and memory use.
   - Save a small sample of model outputs (latent vectors) for comparison.
2. **Optimise ISP**
   - Leverage 2-3 Different Methods of your choice (e.g., batching, mixed precision, quantization, ONNX/TensorRT export, distributed inference or anything else, you can be creative!).
   - Keep the code modular and well-documented.
   - Ensure results remain comparable to baseline outputs.
3. **Benchmark Each Optimization**
   - Report different metrics to show that your optimisation worked, while still being correct
   - Summarize results in a short table or plot.
4. **Submission:**
   - Share the solution via GitHub (Python script or Jupyter Notebook).

    - Use Colab for free GPU access if needed (you can have a look at our example notebooks on Google Colab) - Note: Remove the mamba-ssm dependency (See our Notebook).

Note: Try to not constrain the design by the resources you have at hand (e.g. the code should be scalable over available resources which could be tens of GPU Nodes, most existing libraries allow this through different flags). We care about the idea and implementation so no worries if you haven't tested it on 2 GPUs (or 1 for that matter) as long as it could hypothetically be run like that.

Good luck 🤗! Don't be afraid to be creative or to reach out if something is unclear. We would love you to present your findings at our next call for 20-25mins. Please share the code beforehand.

Your Helical Team