# A. Artifact Appendix

## A.1 Abstract

This Artifact Appendix provides the instructions, scripts, and configurations necessary to run the experiments of our paper on federated large language model (LLM) pre-training using the *Photon* system. We focus on the script, `scripts/fed_125m_example.sh`, that orchestrates the entire process: downloading dependencies, launching the federated server, spinning up clients, and training a 125M-parameter model end to end. However, we recommend following carefully the `README.md` file and the provided example scripts for a more detailed understanding of the setup and execution. By running the `scripts/fed_125m_example.sh` script, users can witness how Photon handles Hydra-based configuration resolution, aggregator (server) bootstrapping, and client participation.

## A.2 Artifact check-list (meta-information)

- **Algorithm:** LocalSGD-based federated optimization with integrated distributed data-parallel (DDP) or fully sharded data parallel (FSDP) when applicable.
- **Program:** Python scripts employing PyTorch, integrated with Flower (for federated coordination) and Ray for model updates communication.
- **Compilation:** No explicit compilation. A Python-based environment setup is mandatory.
- **Transformations:** Data tokenization, normalization, optional data preprocessing (compression), and partitioning in client shards.
- **Binary:** No direct binaries; entire artifact is Python-based.
- **Data set:** A small subset of C4 is included for demonstration. For larger training, full C4 or The Pile can be substituted (scripts not included here).
- **Run-time environment:** Linux system (Ubuntu 22.04 recommended), Python 3.11, CUDA(12.4)-enabled PyTorch 2.1.5, plus Hydra for configuration resolution.
- **Hardware:** At least one NVIDIA GPU (NVIDIA A40, RTX2080Ti, V100, A100, H100, etc.), stable network links (1–10Gbps) if multiple machines are used.
- **Run-time state:** Users can run everything on a single machine with multiple GPUs, or distribute across multiple nodes.
- **Execution:** A single script `scripts/fed_125m_example.sh` that performs the entire flow (setup, server launch, client launches, local training).
- **Metrics:** Primary metric is validation perplexity, with secondary metrics including GPU utilization, throughput, and communication overhead. Wandb logging is supported but requires custom configuration for which guidelines are provided in the code docstrings.
- **Output:** Model checkpoints, logs of training progress, final perplexity.
- **Experiments:** Demonstration of the federated pre-training and centralized training of a 125M-parameter decoder-only LLM, which can be scaled up if desired.
- **Disk space required:** Approximately $5/15$GB for the small subset of C4 plus checkpoints. (Larger experiments may require $300/1000$GB).
- **Time needed to prepare workflow:** Approximately 1 hour for environment setup, $30/60$ minutes to download and preprocess the small dataset.
- **Time needed to complete experiments:** A few hours for the 125M demonstration. Larger-scale runs can take days.
- **Publicly available:** Yes, code repository is licensed (Apache-2.0 license) and will be made public.
- **Code licenses (if publicly available):** Apache License 2.0.
- **Data licenses (if publicly available):** C4 is under the ODC-BY license.
- **Workflow framework used:** Flower + Ray + PyTorch + Hydra, plus a single orchestrating shell script.
- **Archived (provide DOI):** We will archive on Zenodo or similar databases.

## A.3 Description

### A.3.1 How delivered

The artifact is provided in a zipped repository containing:

- `README.md`: A quick overview and key instructions.
- `scripts/system_setup.sh`: Installs base dependencies, sets up the environment.
- `scripts/convert_c4_dataset.sh`: Acquires a small version of C4 for demonstration. Prepare the dataset for training.
- `scripts/fed_125m_example.sh`:
  The single script that launches everything for a 125M-parameter model. It internally invokes Hydra-based configs for server and clients, then orchestrates the run.
- `scripts/cen_125m_example.sh`:
  The single script that launches centralized training of a 125M-parameter model. It internally invokes Hydra-based configs. It is prepared to operate on a single machine setup launching a parallelized training on the available GPUs.
- `configs/`: YAML files specifying hyperparameters (learning rate, batch size, etc.), aggregator properties, and Hydra overrides.

### A.3.2 Hardware dependencies

- **GPU:**
  - For the 125M example, a single GPU with $\geq$12GB memory is sufficient, even though a larger memory ($\geq$40GB) is recommended.
  - For multi-node, each node should have a CUDA-capable GPU and at least 1–10Gbps network connectivity.

### A.3.3 Software dependencies

- **OS:** Linux (Ubuntu 22.04+).
- **Python:** 3.11 or higher.
- **CUDA/CuDNN:** Version 12.4 is recommended, being compatible with PyTorch 2.1.5 and your specific GPU driver. These can be installed automatically via `scripts/system_setup.sh`
- **Package managers:** Poetry is supported for dependency management.
- **Libraries:** PyTorch 2.1.5, Flower (custom version), Ray, Hydra, and standard Python utilities (NumPy, Pandas, etc.). Installed automatically via the `scripts/system_setup.sh` and `scripts/install_env.sh` scripts.

### A.3.4 Data sets

- A small subset of C4 is included for demonstration.
- It is fetched, unpacked locally, and tokenized by `scripts/convert_c4_dataset.sh`.

Users can later replace this with the full C4 or other corpora by adjusting parts of the code and configuration files.

## A.4 Installation and Usage

Refer to the `README.md` file for a more detailed guide. Below is a quick start guide to run the federated pre-training of a 125M-parameter model.

**System prep and environment:**

1. **Download the zip file and run the setup script:** A system-wide pip package manager is required to install the gdown library that downloads the zipped repository at this link.

```
sudo apt update
sudo apt install python3-pip unzip
pip3 install gdown
FILE_ID=1R-LOpedSJx2_i7Jm0lH8PjkXeGxTUU9e
gdown "https://drive.google.com/uc?id=$FILE_ID"
unzip photon.zip
cd photon/scripts
. system_setup.sh
```

   This can install build tools, CUDA drivers (Ubuntu-based).

2. **Install dependencies:**

```
cd photon/scripts
. install_env.sh
```

**Download, prepare/convert dataset with the provided script.**

```
cd photon
bash scripts/convert_c4_dataset.sh
```

**Run the single script for federate pre-training of the 125M model:**

```
cd photon
bash scripts/fed_125m_example.sh
```

This command executes the following steps internally:

- **Hydra configs interpretation:** Hydra interprets the configs and dumps them to a file that is read by the other processes. The file `photon/hydra_resolver.py` is used.
- **Launch Flower Superlink:** The command used is `poetry run flower-superlink`.
- **Launch Flower ServerApp:** The command used is `poetry run flower-server-app photon.server_app:app`.
- **Launch Flower ClientApps:** The command used is `poetry run flower-client-app photon.client_app:app`
- **Federated rounds:** The aggregator orchestrates local training (LocalSGD) across clients, synchronizes updates after each round.
- **Checkpoints and logs:** Intermediate global checkpoints and logs are saved in `checkpoints/` and `runs/` respectively.
- **Completion:** The script logs periodically several metrics, e.g., perplexity and throughput.

## A.5 Evaluation and expected result

**Targets of interest:**

- **Validation perplexity:** For the 125M demo, you should observe perplexity dropping towards the low 40s or upper 30s after sufficient rounds, depending on configuration.
- **Runtime logs:** Both aggregator and client logs are found under `runs/`, indicating the number of tokens processed, average GPU utilization, and steps per round.
- **Checkpoints:** Partial and final checkpoints are saved in the `checkpoints/` folder.

## A.6 Experiment customization

- **Config override:** Edit `scripts/fed_125m_example.sh` or pass Hydra overrides to change client count or hyperparameters.
- **Hardware scaling:** By default, the script spawns multiple clients on a single node. For multi-node, adapt the aggregator IP and client addresses in `scripts/fed_125m_example.sh`.
- **Batch sizes / epochs:** Controlled by Hydra configs in the `configs/` folder.
- **Dataset:** Replace the small C4 path with your own local data for more extended training.

## A.7 Notes

- **Partial or intermittent clients:** If a client crashes or is not reachable, the aggregator continues with remaining clients in subsequent rounds.
- **Performance considerations:** For minimal overhead, ensure a stable GPU environment. Larger-scale runs (1.3B+) require more disk space, memory, and multi-GPU setups.

## A.8 Methodology

We adhere to artifact evaluation guidelines:

- Single-blind AE with emphasis on reproducibility and clarity.
- Clear *build* (ffrom the scripts `scripts/system_setup.sh` and `scripts/install_env.sh`), *run* ( using the script `scripts/fed_125m_example.sh`), and *analysis* (logs, final checkpoint) phases.
- **ACM Artifact Badging** [1] best practices: code will be made public, well-documented, and tested on a standard environment.

## References

[1] ACM Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-badging.