Artifact Description for Paper 283 EFIM

1 Getting Started Guide

Our artifacts require a NVIDIA GPU with CUDA software on a linux operating system. It also requires to install conda software. Then we pack the instructions for preparing the environment in scripts/setup.sh. To set up the environment, you should just run

```
source scripts/setup.sh
```

It will automatically create a conda env named 'euro_par_artifact' and install required software. Since our work need access to large language models (LLMs), it also download four LLMs (about 50-60 GB) from huggingface into local directory 'models'. Then the following instructions should run within this conda env.

2 Step-by-Step Instructions

The detailed instructions can also be found in README.md.

– To launch vLLM server w/o prefix caching

  ```
  export MODEL=llama
  ./scripts/launch_server.sh
  ```

– To launch vLLM server w/ prefix caching

  ```
  export MODEL=llama
  ./scripts/launch_prefix_server.sh
  ```

Note that, 'MODEL' can be chose from 'llama', 'llama-enhance', 'deepseek' and 'deepseek-enhance'. 'llama' and 'deepseek' are oLLM, while 'llama-ehance' and 'deepseek-enhance' are eLLM.

Evaluate Infilling and Subtoken Generation Ability. For each test, we need to launch vLLM server first and run benchmark script. We also need to ensure the model passed to vLLM server match with that in the benchmark script. In this section, you can either launch vLLM server with or without prefix caching. Because it has little effect on the output quality.

– To evaluate HumanEval w/FIM

  ```
  python benchmark/async_benchmark_humaneval.py --model llama
  ```

- To evaluate HumanEval w/EFIM

```
python benchmark/async_benchmark_humaneval.py \
        --model llama --use-EFIM
```

- To evaluate CCEval w/FIM

```
python benchmark/async_benchmark_cceval.py --model llama
```

- To evaluate CCEval w/EFIM

```
python benchmark/async_benchmark_cceval.py \
        --model llama --use-EFIM
```

For HumanEval Infilling (single-line, multi-line and random-span), we print pass@1 metric on the terminal. For CCEval, we print EM and ES metric on the terminal.

Evaluate Inference Speed. For baseline scheme, you should launch vLLM server w/o prefix caching and evaluate model w/FIM. For FIM scheme, you should launch vLLM server w/ prefix caching and evaluate model w/FIM. For EFIM scheme, you should launch vLLM server w/ prefix caching and evaluate mmodel w/EFIM.

- To evaluate model w/FIM

```
python benchmark/async_benchmark_inference_speed.py \
        --model llama --num-round 5 --num-user 16
```

- To evaluate model w/EFIM

```
python benchmark/async_benchmark_inference_speed.py \
        --model llama-enhance --num-round 5 --num-user 16
```

After benchmarking, the script prints the average latency for each request and request throughput on the terminal. The key-value (KV) cache reuse rate can be seen on the vLLM server terminal.