Profiling vLLM

We support tracing vLLM workers using the `torch.profiler` module. You can enable tracing by setting the `VLLM_TORCH_PROFILER_DIR` environment variable to the directory where you want to save the traces: `VLLM_TORCH_PROFILER_DIR=/mnt/traces/`

The OpenAI server also needs to be started with the `VLLM_TORCH_PROFILER_DIR` environment variable set.

When using `benchmarks/benchmark_serving.py`, you can enable profiling by passing the `--profile` flag.

:::{warning}
Only enable profiling in a development environment.
:::

Traces can be visualized using <https://ui.perfetto.dev/>.

:::{tip}
Only send a few requests through vLLM when profiling, as the traces can get quite large. Also, no need to untar the traces, they can be viewed directly.
:::

:::{tip}
To stop the profiler - it flushes out all the profile trace files to the directory. This takes time, for example for about 100 requests worth of data for a llama 70b, it takes about 10 minutes to flush out on a H100.

Set the env variable VLLM_RPC_TIMEOUT to a big number before you start the server. Say something like 30 minutes.

`export VLLM_RPC_TIMEOUT=1800000`

:::

Example commands and usage

Offline Inference

Refer to <gh-file:examples/offline_inference/simple_profiling.py> for an example.

OpenAI Server

```bash
VLLM_TORCH_PROFILER_DIR=./vllm_profile python -m vllm.entrypoints.openai.api_server --model meta-llama/Meta-Llama-3-70B
```

benchmark_serving.py:

```bash
python benchmarks/benchmark_serving.py --backend vllm --model meta-llama/Meta-Llama-3-70B --dataset-name sharegpt --dataset-path sharegpt.json --profile --num-prompts 2
```