

## vLLM의 OpenAI-Compatible Server 툴아보기

---

Hyogeun Oh

---

## Speaker Introduction

# Hygeun Oh

(**오효근**, ZeroHertz)

- 기계공학 학/석사 후 Machine Learning Engineer (전문연구요원)로 재직 중입니다.
- Python과 Kubernetes를 주로 다루며, MLOps에 깊은 관심이 있습니다.
- Neovim을 애용하고 생산적인 개발 환경을 추구합니다.
- 더 나은 ML 파이프라인과 자동화를 고민합니다.

GitHub [1]



[1] <https://github.com/ZeroHertz>

---

# Sold Out, Thank You!

---

0. Before We Begin...

## vLLM의 OpenAI-Compatible Server 툽아보기 [KR Only]



진행자: Hyogeun Oh

난이도: 중급(Intermediate)

진행 시간: 10:00 AM - 12:00 PM (2시간)

### [설명]

최근 대규모 언어 모델(LLM)을 활용한 서비스가 급증함에 따라, 고성능·저지연·효율적인 LLM 서빙 기술에 대한 수요도 함께 증가하고 있습니다.

본 튜토리얼에서는 Meta의 LLaMA, Mistral, OpenChat 등 다양한 오픈소스 LLM을 OpenAI API 호환 형식으로 빠르게 서빙할 수 있는 vLLM 프레임워크를 중심으로 실습과 개념을 다룹니다.

상품이 매진되었어요!

## Tutorial Code

0. Before We Begin...

□ 노트북 가져오신 분들은 아래 QR의 README.md를 참고하여 환경 설정해주세요!

□ 실행 환경이 없다면, GitHub에서 코드를 볼 수 있습니다.

Tutorial Code [2]



# macOS MPS (Metal Performance Shaders)

0. Before We Begin...

Does vllm support the Mac/Metal/MPS? #1441

Closed

Phil-U-U opened on Oct 21, 2023

I ran into the error when pip install vllm in Mac:  
RuntimeError: Cannot find CUDA\_HOME. CUDA must be available to build the package.

82

WoosukKwon on Oct 22, 2023

Hi @Phil-U-U, vLLM does not support MPS backend at the moment as its main target scenario is to be deployed as a high-throughput server running on powerful accelerators like NVIDIA A100/H100.

5 33 2

WoosukKwon closed this as completed on Oct 22, 2023

ibehnam on Dec 13, 2023

Hi @Phil-U-U, vLLM does not support MPS backend at the moment as its main target scenario is to be deployed as a high-throughput server running on powerful accelerators like NVIDIA A100/H100.

That's understandable. Although, with Mac Studio, many people and companies are starting to use the Mac as LLM servers. I get very good t/s on Mac Studio M2 Ultra using llama.cpp, but something like vLLM on the Mac would be a game changer. I hope the dev team considers this.

122

---

## Table of Contents

---

0. Before We Begin...

1. Introduction

2. OpenAI-Compatible Server

3. Architecture

4. Hands-on

5. Wrap-up

PART 1

---

# Introduction

---

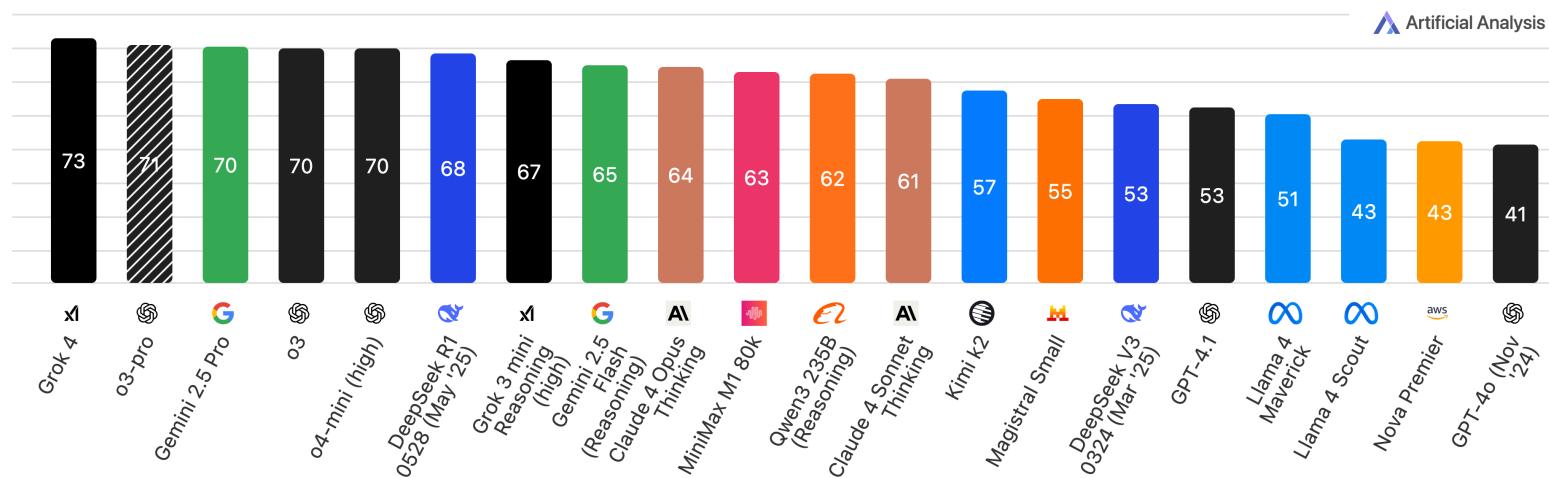
# Why Self-Host LLMs When Powerful Commercial APIs Exist?

- 비용 부담: traffic 사용량 증가 시 높은 과금
- Privacy/보안 이슈: 민감 data 외부 전송 불가
- Customizing의 어려움: prompt, parameter, 응답 format등 제한적 제어
- API 가용성: 속도, 안정성, 국가별 접근 제한 가능성

## Artificial Analysis Intelligence Index [4]

Artificial Analysis Intelligence Index incorporates 7 evaluations: MMLU-Pro, GPQA Diamond, Humanity's Last Exam, LiveCodeBench, SciCode, AIME, MATH-500

 Estimate (independent evaluation forthcoming)



# Why Self-Host LLMs When Powerful Commercial APIs Exist?

❑ transformers의 AutoModelForCausalLM을 통해 추론하면 되지 않나?

```
def main():
    logger.info(f"MODEL_NAME={MODEL_NAME}")

    processor = AutoProcessor.from_pretrained(MODEL_NAME)
    model = AutoModelForCausalLM.from_pretrained(MODEL_NAME)
    logger.info("Model & processor Loaded!")

    messages = [{"role": "user", "content": "Hello, PyCon Korea 2025!"}]
    prompt = processor.apply_chat_template(
        messages, tokenize=False, add_generation_prompt=True
    )
    logger.info("prompt:")
    print(prompt)
    inputs = processor(prompt, return_tensors="pt")

    with torch.no_grad():
        generated_ids = model.generate(
            *inputs,
            max_new_tokens=1024,
            do_sample=True,
            top_p=0.95,
            temperature=0.8,
            pad_token_id=processor.eos_token_id,
        )

    output_text = processor.batch_decode(generated_ids, skip_special_tokens=True)[0]
    logger.info("output_text:")
    print(output_text)
```



```
2025-07-24 23:51:30.031 | INFO  | __main__:main:31 - MODEL_NAME='Qwen/Qwen3-0.6B'
2025-07-24 23:51:33.554 | INFO  | __main__:main:35 - Model & processor Loaded!
2025-07-24 23:51:33.583 | INFO  | __main__:main:41 - prompt:
<|im_start|>user
Hello, PyCon Korea 2025!<|im_end|>
<|im_start|>assistant
2025-07-24 23:51:44.305 | INFO  | __main__:main:56 - output_text:
user
Hello, PyCon Korea 2025!
assistant
<think>
Okay, the user sent me a message saying "Hello, PyCon Korea 2025!" So I need to respond to that. Let me think. PyCon is a global event, so I should acknowledge it. Maybe mention that it's happening in Korea. Let me check the date again. Oh, PyCon Korea 2025 is scheduled for June 10th, 2025. That's important to include.

I should make sure the response is friendly and enthusiastic. Maybe start with a greeting, then mention the event details. Also, since they asked for help, I should offer assistance. Let me put that all together in a natural way.
</think>

Hello, PyCon Korea 2025! 🎉
We're excited to have you join us for the event! It's scheduled for June 10th, 2025, and we're looking forward to celebrating tech innovation and showcasing the world's best trends! Let me know if you need help or have any questions! 🚀
```

# Why Self-Host LLMs When Powerful Commercial APIs Exist?

❑ vLLM의 LLM을 통해 추론하면 되지 않나?

The screenshot shows a terminal window titled "nvim" displaying Python code for offline inference using vLLM. The code defines a main function that initializes an LLM, sends a message, and prints the response. It then runs the main function if the script is executed directly.

```
nvim nvim nvim ~/PyCon_KR_2025_Tutorial_vLLM/src/02_offline_inference_with_vllm.py
52 def main():
53     logger.info(f"MODEL_NAME={MODEL_NAME}")
54
55     llm = LLM(model=MODEL_NAME, max_model_len=2048)
56     messages = [
57         {"role": "user", "content": "Hello, PyCon Korea 2025!"},
58     ]
59     sampling_params = SamplingParams(top_p=0.95, temperature=0.8, max_tokens=2048)
60     outputs = llm.chat(messages, sampling_params=sampling_params)
61     for output in outputs:
62         logger.info("output_text:")
63         print(output.outputs[0].text)
64
65
66 if __name__ == "__main__":
67     main()

1: zeroherz@zeroherz-mm:~/PyCon_KR_2025_Tutorial_vLLM
INFO 08-11 20:12:22 [executor_base.py:113] # cpu blocks: 2340, # CPU blocks: 0
INFO 08-11 20:12:22 [executor_base.py:118] Maximum concurrency for 2048 tokens per request: 18.28x
INFO 08-11 20:12:22 [llm_engine.py:428] init engine (profile, create kv cache, warmup model) took 0.55 seconds
INFO 08-11 20:12:25 [chat_utils.py:444] Detected the chat template content format to be 'string'. You can set `--chat-template-content-format` to overri
de this.
Adding requests: 100% | 1/1 [00:00<00:00, 5652.70it/s]
Processed prompts: 0% | 0/1 [00:00<?, ?it/s, est. speed input: 0.00 toks/s, output: 0.00 toks/s]
WARNING 08-11 20:12:26 [cpu.py:250] Pin memory is not supported on CPU.
Processed prompts: 100% | 1/1 [00:03<00:00, 3.91s/it, est. speed input: 4.86 toks/s, output: 41.93 toks/s]
2025-08-11 20:12:29.797 | INFO      | __main__:main:62 - output_text:
<think>
Okay, the user is asking me to respond to "Hello, PyCon Korea 2025!" as if they are a guest. I need to make sure I address them in a friendly and welcom
ing manner. Let me think... They might be joining the event, so a positive and enthusiastic reply would be appropriate. I should mention the event's pur
pose, the audience, and maybe invite them to participate. Also, I should keep the tone friendly and enthusiastic. Let me put that together in a natural
way.
</think>

Hello, PyCon Korea 2025! It's great to meet you in person. Joining this event is fantastic—your presence will make it even more exciting! What can I do
for you? Let's make it a memorable experience! 🌎✨
NORMAL ➤ ↵ main ➤ $ term:../../bin/zsh
```

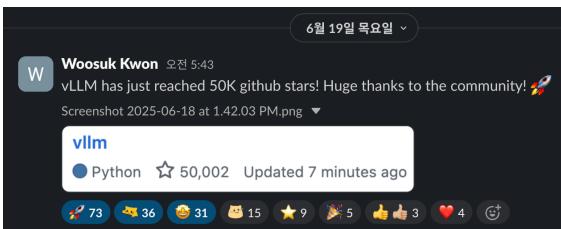
The terminal output shows the execution of the script, including logs from the vLLM library and the generated response text. The response is a friendly welcome message.

# Why vLLM?

1. Introduction

## ❑ vLLM history

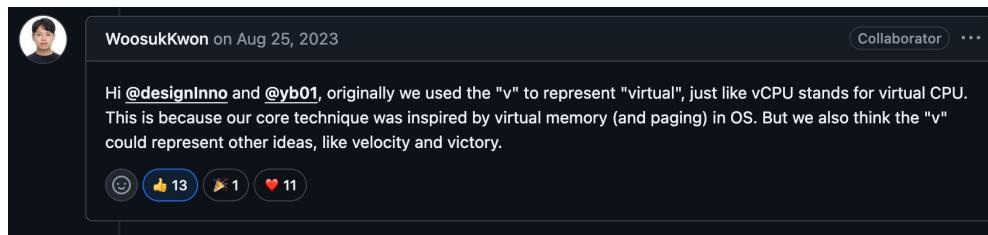
- 2023년 2월 9일, GitHub에서 CacheFlow<sup>[5]</sup>라는 이름으로 시작
- 2023년 6월 17일, vLLM으로 이름 변경<sup>[6]</sup>
- 2023 9월 12일, UC Berkeley Sky Computing Lab에서 “Efficient Memory Management for Large Language Model Serving with PagedAttention” 논문 발표<sup>[7]</sup>
- 2025년 6월 19일, GitHub star 50k 달성



- 2025년 7월 24일, v0.10.0 release

## ❑ License: Apache-2.0<sup>[8]</sup>

## ❑ v of vLLM<sup>[9]</sup>



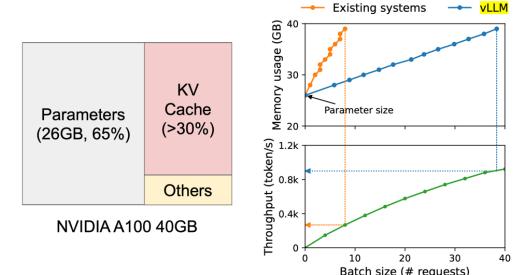
## Efficient Memory Management for Large Language Model Serving with *PagedAttention*

Woosuk Kwon<sup>1,\*</sup> Zhuohan Li<sup>1,\*</sup> Siyuan Zhuang<sup>1</sup> Ying Sheng<sup>1,2</sup> Lianmin Zheng<sup>1</sup> Cody Hao Yu<sup>3</sup>  
Joseph E. Gonzalez<sup>1</sup> Hao Zhang<sup>4</sup> Ion Stoica<sup>1</sup>

<sup>1</sup>UC Berkeley <sup>2</sup>Stanford University <sup>3</sup>Independent Researcher <sup>4</sup>UC San Diego

### Abstract

High throughput serving of large language models (LLMs) requires batching sufficiently many requests at a time. However, existing systems struggle because the key-value cache (KV cache) memory for each request is huge and grows and shrinks dynamically. When managed inefficiently, this memory can be significantly wasted by fragmentation and redundant duplication, limiting the batch size. To address this problem, we propose PagedAttention, an attention algorithm inspired by the classical virtual memory and paging techniques in operating systems. On top of it, we build vLLM, an LLM serving system that achieves (1) near-zero waste in KV cache memory and (2) flexible sharing of KV cache within and across requests to further reduce memory usage. Our evaluations show that vLLM improves the throughput of popular LLMs by 2-4x with the same level of latency compared to the state-of-the-art systems, such as FasterTransformer and Orca. The improvement is more pronounced with longer sequences, larger models, and more complex decoding algorithms. vLLM’s source code is publicly available at <https://github.com/vllm-project/vllm>.



**Figure 1.** Left: Memory layout when serving an LLM with 13B parameters on NVIDIA A100. The parameters (gray) persist in GPU memory throughout serving. The memory for the KV cache (red) is (de)allocated per serving request. A small amount of memory (yellow) is used ephemerally for activation. Right: vLLM smooths out the rapid growth curve of KV cache memory seen in existing systems [31, 60], leading to a notable boost in serving throughput.

[5] <https://github.com/vllm-project/vllm/commit/e7d9d9c08e79b386f6d0477e87b7a572390317d>

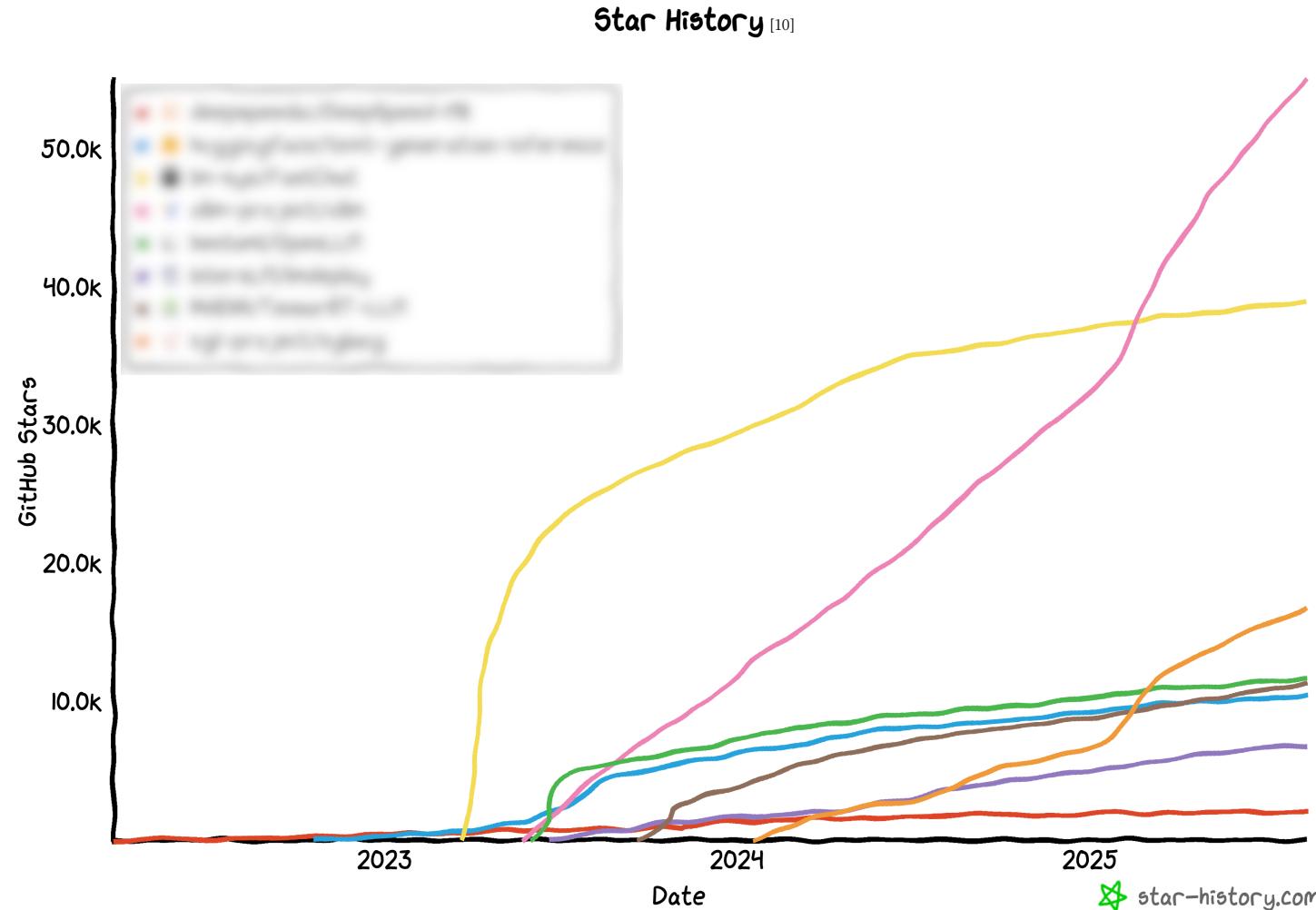
[6] <https://github.com/vllm-project/vllm/commit/0b98ba15c744fd1fb0ea4f2135e85ca23d572ae1>

[7] <https://arxiv.org/abs/2309.06180>

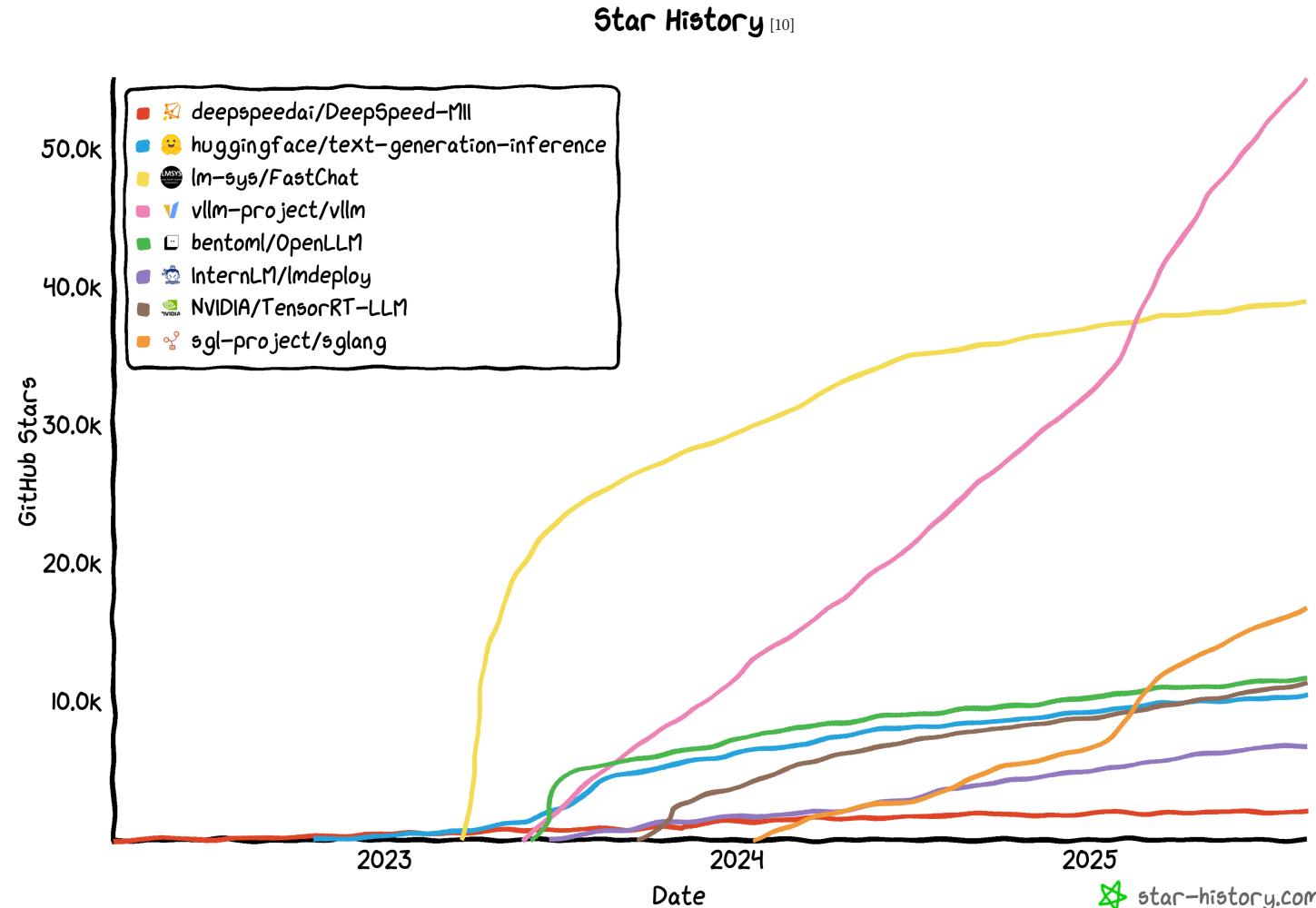
[8] <https://github.com/vllm-project/vllm/blob/main/LICENSE>

[9] <https://github.com/vllm-project/vllm/issues/835>

# Why vLLM?



# Why vLLM?



# Why vLLM?

## ▣ 초고속 LLM 서빙 성능

- **State-of-the-art throughput:** 대규모 요청 처리
- **Continuous batching:** 실시간 요청에 대한 효율적 처리

## ▣ 메모리 효율 및 최적화된 연산

- **PagedAttention**<sup>[7]</sup>: Attention key/value 메모리의 효율적 관리
- **Optimized CUDA kernels:** FlashAttention<sup>[11]</sup>, FlashInfer<sup>[12]</sup> 등 최신 커널 통합
- **Chunked prefill**<sup>[13]</sup>, **Speculative decoding**<sup>[14]</sup>

## ▣ 유연성 및 확장성

- **HuggingFace 통합:** 손쉽게 다양한 모델 서빙
- **다양한 디코딩 알고리즘:** 병렬 샘플링, 빔서치 등 고성능 추론 지원
- **분산 추론 지원:** Tensor, pipeline (Ray 기반), data, expert parallelism 지원

## ▣ 실용적 API 및 운영 편의성

- **OpenAI-Compatible API:** 기존 AI 서비스 (e.g., LangChain, Gemini CLI, ...)와 손쉽게 연동
- **Streaming output:** 스트리밍 방식 결과 제공
- **Prefix caching, Multi-LoRA**

[7] <https://arxiv.org/abs/2309.06180>

[11] <https://github.com/vllm-project/flash-attention>

[12] <https://github.com/flashinfer-ai/flashinfer>

[13] <https://docs.vllm.ai/en/v0.9.2/configuration/optimization.html#chunked-prefill>

[14] [https://docs.vllm.ai/en/v0.9.2/features/spec\\_decode.html](https://docs.vllm.ai/en/v0.9.2/features/spec_decode.html)

# How to serving LLM with vLLM?

## □ Installation

### ▪ Local (CPU) 사용 시

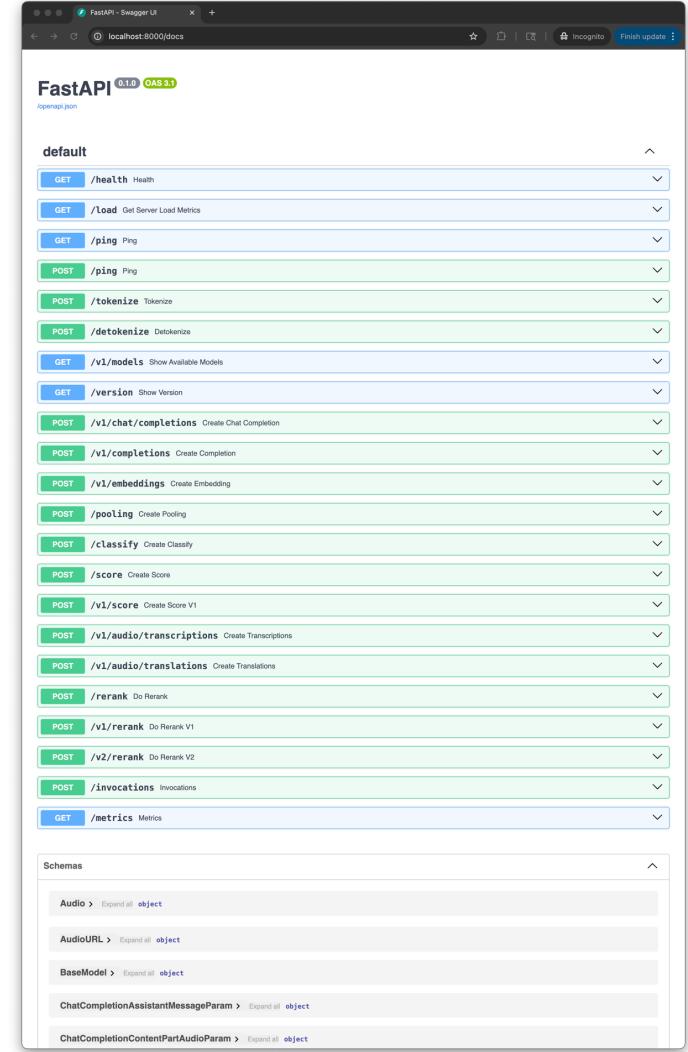
```
$ uv pip install vllm==0.9.2
Using Python 3.12.11 environment at: /opt/venv/main
Resolved 120 packages in 47ms
Installed 1 package in 10ms
+ vllm==0.9.2
```

### ▪ GPU 서버 사용 시 [15]

```
1 docker run --runtime nvidia --gpus all \
2   --name vllm \
3   -v ~/.cache/huggingface:/root/.cache/huggingface \
4   -p 8000:8000 \
5   --ipc=host \
6   vllm/vllm-openai:v0.9.2 \
7   --model Qwen/Qwen3-0.6B \
8   --max-model-len 8192
```

## □ vllm serve

```
$ vllm serve Qwen/Qwen3-0.6B --max-model-len 8192
INFO 07-27 01:38:11 [__init__.py:244] Automatically detected platform cpu.
INFO 07-27 01:38:12 [api_server.py:1395] VLLM API server version 0.9.2
INFO 07-27 01:38:12 [cl1_args.py:325] non-default args: {'model': 'Qwen/Qwen3-0.6B', 'max_model_len': 8192}
INFO 07-27 01:38:15 [config.py:841] This model supports multiple tasks: ('reward', 'classify', 'generate', 'embed'). Defaulting to 'generate'.
WARNING 07-27 01:38:15 [config.py:328] Your device (cpu) doesn't support torch.bfloat16. Falling back to torch.float16 for compatibility.
WARNING 07-27 01:38:15 [config.py:272] Using max model len 8192.
INFO 07-27 01:38:15 [cpu_utils.py:1746] cpu is experimental on VLLM_USE_V1_1. Falling back to V0 Engine.
WARNING 07-27 01:38:15 [gpu.py:131] Environment variable VLLM_CPU_KVCACHE_SPACE (GiB) for CPU backend is not set, using 4 by default.
INFO 07-27 01:38:16 [api_server.py:268] Started engine process with PID 83075
INFO 07-27 01:38:17 [__init__.py:244] Automatically detected platform cpu.
INFO 07-27 01:38:18 [llm_engine.py:230] Initializing a VLLM engine (v0.9.2) with config: model='Qwen/Qwen3-0.6B', speculative_config=None, tokenizer='Qwen/Qwen3-0.6B', skip_tokenizer_init=False, tokenizer_mode=auto, revision=None, override_neuron_configs(), tokenizer_neuron_configs(), trust_remote_code=False, dtype=torch.float16, max_seq_len=8192, download_dir=None, load_format=LoadFormat.AUTO, tensor_parallel_size=1, disable_custom_all_reduce=True, quantization=None, enforce_eager=False, kv_cache_dtype=auto, device_config=cpu, decoding_config=DecodingConfig(backends='auto', disable_fallback=False, disable_any_whitespace=False, disable_additional_properties=False, reasoning_backend=''), observability_config=ObservabilityConfig(show_hidden_metrics_for_version=None, otlp_traces_endpoint=None, collect_detailed_traces=None), seeds=0, served_model=name='Qwen/Qwen3-0.6B', num_scheduler_steps=1, multi_step_stream_outputs=True, enable_prefix_caching=None, chunked=True, enable_auto_functionalized_v2=False, use_async_output=False, pooler_config=None, compilation_config={"level":0, "debug_dump_path": "", "backend": "", "custom_ops": [], "use_inductor": true, "compile_sizes": []}, "inductor_compile_config": {"enable_inductor": true, "inductor_passes": []}, "use_cudagraph": true, "cudagraph_num_of_warmups": 0, "cudagraph_capture_sizes": [], "cudagraph_copy_inputs": false, "full_cuda_graph": false, "max_capture_size": 256, "local_cache_dir": null}, use_cached_outputs=True, INFO 07-27 01:38:23 [launcher.py:37] Route: /v1/models, Methods: GET
INFO 07-27 01:38:23 [launcher.py:37] Route: /version, Methods: GET
INFO 07-27 01:38:23 [launcher.py:37] Route: /v1/chat/completions, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /v1/completions, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /v1/embeddings, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /pooling, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /score, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /v1/audio/transcriptions, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /v1/audio/translations, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /rerank, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /v1/rerank, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /v2/rerank, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /invocations, Methods: POST
INFO 07-27 01:38:23 [launcher.py:37] Route: /metrics, Methods: GET
INFO: Started server process [8309]
INFO: Waiting for application startup...
INFO: Application startup complete.
```



PART 2

---

# OpenAI-Compaitable Server

---

# OpenAI API Spec [16]

## □ /v1/models [17]

```
zerohertz@zerohertz-mm:~/PyCon_KR_2025_Tutorial_vLLM$ curl http://localhost:8000/v1/models | jq
% Total % Received % Xferd Average Speed Time Time Current
          Dload Upload Total Spent Left Speed
100 480 100 480 0 0 487k 0 --:-- --:-- --:-- 468k
{
  "object": "list",
  "data": [
    {
      "id": "Qwen/Qwen3-0.6B",
      "object": "model",
      "created": "1754911042",
      "owned_by": "vLLM",
      "root": "Qwen/Qwen3-0.6B",
      "parent": null,
      "max_model_len": 8192,
      "permission": [
        {
          "id": "modelperm-46d848c0ebbb4d5ba5581510b4e92652",
          "object": "model_permission",
          "created": "1754911042",
          "allow_create_engine": false,
          "allow_sampling": true,
          "allow_logprobs": true,
          "allow_search_indices": false,
          "allow_view": true,
          "allow_fine_tuning": false,
          "organization": "*",
          "group": null,
          "is_blocking": false
        }
      ]
    }
  ]
}
```

## □ /v1/chat/completions [18]

```
zerohertz@zerohertz-mm:~/PyCon_KR_2025_Tutorial_vLLM$ curl -X POST http://localhost:8000/v1/chat/completions \
-H "Content-Type: application/json" \
-d '{
  "model": "Qwen/Qwen3-0.6B",
  "messages": [
    {
      "role": "user",
      "content": "Hello, PyCon Korea 2025!"
    }
  ]
}' | jq
% Total % Received % Xferd Average Speed Time Time Current
          Dload Upload Total Spent Left Speed
100 1655 100 1509 180 146 100 0:00:16 0:00:15 0:00:01 312
{
  "id": "chatmpl-bd919d6b7f4474994eb728369ca85b5",
  "object": "chat.completion",
  "created": 1754911158,
  "model": "Qwen/Qwen3-0.6B",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "ethan>Okay, the user is asking for a greeting at PyCon Korea 2025. Let me start by understanding the context. PyCon is a major tech conference, so the response should be formal and enthusiastic.\nFirst, I should address the event with its name, PyCon Korea 2025. Then, mention that it's a big event for tech professionals. It's important to highlight its significance and the excitement of attending. Maybe include some key points like the speakers, the schedule, and the impact of the event. Also, keep the tone positive and inviting. Need to make sure the response is concise but covers all necessary aspects without being too lengthy.\n</think>\nHello, PyCon Korea 2025! > This is a fantastic opportunity to connect with industry leaders, innovators, and passionate developers. As the largest tech event in Asia, it will bring together the brightest minds and create unforgettable experiences for all attendees. Let's look forward to a vibrant and inspiring conference!",
        "logprobs": null,
        "finish_reason": "stop",
        "stop_reason": null
      }
    }
  ],
  "stream": false
}
data: {"id": "chatmpl-cac268da0404eb78867d289617fcfa4", "object": "chat.completion.chunk", "created": 1754911235, "model": "Qwen/Qwen3-0.6B", "choices": [{"index": 0, "delta": {"content": ""}, "logprobs": null, "finish_reason": null}]}
data: {"id": "chatmpl-cac268da0404eb78867d289617fcfa4", "object": "chat.completion.chunk", "created": 1754911235, "model": "Qwen/Qwen3-0.6B", "choices": [{"index": 0, "delta": {"content": "\n"}, "logprobs": null, "finish_reason": null}]}
data: {"id": "chatmpl-cac268da0404eb78867d289617fcfa4", "object": "chat.completion.chunk", "created": 1754911235, "model": "Qwen/Qwen3-0.6B", "choices": [{"index": 0, "delta": {"content": "\nHello, PyCon Korea 2025!\n"}, "logprobs": null, "finish_reason": null}]}
data: {"id": "chatmpl-cac268da0404eb78867d289617fcfa4", "object": "chat.completion.chunk", "created": 1754911235, "model": "Qwen/Qwen3-0.6B", "choices": [{"index": 0, "delta": {"content": "\n\nThis is a fantastic opportunity to connect with industry leaders, innovators, and passionate developers. As the largest tech event in Asia, it will bring together the brightest minds and create unforgettable experiences for all attendees. Let's look forward to a vibrant and inspiring conference!\n"}, "logprobs": null, "finish_reason": null}]}
data: {"id": "chatmpl-cac268da0404eb78867d289617fcfa4", "object": "chat.completion.chunk", "created": 1754911235, "model": "Qwen/Qwen3-0.6B", "choices": [{"index": 0, "delta": {"content": "\n\n</think>\nHello, PyCon Korea 2025! > This is a fantastic opportunity to connect with industry leaders, innovators, and passionate developers. As the largest tech event in Asia, it will bring together the brightest minds and create unforgettable experiences for all attendees. Let's look forward to a vibrant and inspiring conference!\n"}, "logprobs": null, "finish_reason": null}]}
data: {"id": "chatmpl-cac268da0404eb78867d289617fcfa4", "object": "chat.completion.chunk", "created": 1754911235, "model": "Qwen/Qwen3-0.6B", "choices": [{"index": 0, "delta": {"content": "\n\nHello, PyCon Korea 2025! > This is a fantastic opportunity to connect with industry leaders, innovators, and passionate developers. As the largest tech event in Asia, it will bring together the brightest minds and create unforgettable experiences for all attendees. Let's look forward to a vibrant and inspiring conference!\n"}, "logprobs": null, "finish_reason": null}]}
data: {"id": "chatmpl-cac268da0404eb78867d289617fcfa4", "object": "chat.completion.chunk", "created": 1754911235, "model": "Qwen/Qwen3-0.6B", "choices": [{"index": 0, "delta": {"content": "\n\nThis is a fantastic opportunity to connect with industry leaders, innovators, and passionate developers. As the largest tech event in Asia, it will bring together the brightest minds and create unforgettable experiences for all attendees. Let's look forward to a vibrant and inspiring conference!\n"}, "logprobs": null, "finish_reason": null}]}' | jq
% Total % Received % Xferd Average Speed Time Time Current
          Dload Upload Total Spent Left Speed
100 1655 100 1509 180 146 100 0:00:16 0:00:15 0:00:01 312
{
  "id": "chatmpl-bd919d6b7f4474994eb728369ca85b5",
  "object": "chat.completion",
  "created": 1754911158,
  "model": "Qwen/Qwen3-0.6B",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "ethan>Okay, the user is asking for a greeting at PyCon Korea 2025. Let me start by understanding the context. PyCon is a major tech conference, so the response should be formal and enthusiastic.\nFirst, I should address the event with its name, PyCon Korea 2025. Then, mention that it's a big event for tech professionals. It's important to highlight its significance and the excitement of attending. Maybe include some key points like the speakers, the schedule, and the impact of the event. Also, keep the tone positive and inviting. Need to make sure the response is concise but covers all necessary aspects without being too lengthy.\n</think>\nHello, PyCon Korea 2025! > This is a fantastic opportunity to connect with industry leaders, innovators, and passionate developers. As the largest tech event in Asia, it will bring together the brightest minds and create unforgettable experiences for all attendees. Let's look forward to a vibrant and inspiring conference!",
        "logprobs": null,
        "finish_reason": "stop",
        "stop_reason": null
      }
    }
  ],
  "stream": false
}
```

## Tool calling

## 2. OpenAI-Compatible Server

□ OpenAI 규격에 맞춰 호출

v1.97.1 openai-python / src / openai / types / shared\_params / function\_definition.py

stainless-app[bot] chore(api): event shapes more accurate

Code Blame 45 lines (32 loc) · 1.47 KB ⚙

```
1 # File generated from our OpenAPI spec by Stainless. See CONTRIBUTING.md for details.
2
3 from __future__ import annotations
4
5 from typing import Optional
6 from typing_extensions import Required, TypedDict
7
8 from .function_parameters import FunctionParameters
9
10 __all__ = ["FunctionDefinition"]
11
12 ...
13 class FunctionDefinition(TypedDict, total=False):
14     name: Required[str]
15     """The name of the function to be called.
16
17     Must be a-z, A-Z, 0-9, or contain underscores and dashes, with a maximum length
18     of 64.
19     """
20
21     description: str
22     """
23     A description of what the function does, used by the model to choose when and
24     how to call the function.
25     """
26
27     parameters: FunctionParameters
28     """The parameters the functions accepts, described as a JSON Schema object.
29
30     See the [guide](https://platform.openai.com/docs/guides/function-calling) for
31     examples, and the
32     [JSON Schema reference](https://json-schema.org/understanding-json-schema/) for
33     documentation about the format.
34
35     Omitting `parameters` defines a function with an empty parameter list.
36     """
37
38     strict: Optional[bool]
39     """Whether to enable strict schema adherence when generating the function call.
40
41     If set to true, the model will follow the exact schema defined in the
42     `parameters` field. Only a subset of JSON Schema is supported when `strict` is
43     `true`. Learn more about Structured Outputs in the
44     [function calling guide](https://platform.openai.com/docs/guides/function-calling).
45     """
46
```

```
zerohertz@zerohertz-mm:~/PyCon_KR_2025_Tutorial_vLLM
```

```
$ curl -X POST http://localhost:8000/v1/chat/completions \
-H "Content-Type: application/json" \
-d '{
  "model": "Qwen/Qwen3-0.6B",
  "messages": [
    {
      "role": "user",
      "content": "What is the weather like in Seoul?"
    }
  ],
  "tools": [
    {
      "type": "function",
      "function": {
        "name": "get_weather",
        "description": "Get the current weather in a given location",
        "parameters": {
          "type": "object",
          "properties": {
            "location": {
              "type": "string",
              "description": "The city and state, e.g. San Francisco, CA"
            },
            "unit": {
              "type": "string",
              "enum": ["celsius", "fahrenheit"]
            }
          },
          "required": ["location"]
        }
      }
    ]
  ],
  "tool_choice": "auto"
}' | jq
```

Total	% Received	% Xferd	Average Speed	Time Dload	Time Upload	Time Total	Time Spent	Time Left	Current Speed
100	981	100	168	100	813	19739	95523	--:--:--	--:--:-- 119k

```
{
  "object": "error",
  "message": "\"/auto\" tool choice requires --enable-auto-tool-choice and --tool-call-parser to be set",
  "type": "BadRequestError",
  "param": '',
  "code": 400
}
```

# Tool calling

- “--tool-call-parser”를 통해 모델에 따른 적절한 parser 설정 필요 [20, 21]

```
vllm serve Qwen/Qwen3-0.6B --max-model-len 8192 --enable-auto-tool-choice --tool-call-parser hermes
$ vllm serve Qwen/Qwen3-0.6B --max-model-len 8192 --enable-auto-tool-choice --tool-call-parser hermes
INFO 07-27 21:13:11 [__init__.py:244] Automatically detected platform: hermes
INFO 07-27 21:13:13 [api_server.py:1395] VLLM API server version 0.9.2
INFO 07-27 21:13:13 [cli_args.py:325] non-default args: {'enable_auto_tool_choice': True, 'tool_call_parser': 'hermes', 'model': 'Qwen/Qwen3-0.6B', 'max_model_len': 8192}
INFO 07-27 21:13:16 [config.py:841] This model supports multiple tasks: {'classify', 'generate', 'embed', 'reward'}. Defaulting to 'generate'.
WARNING 07-27 21:13:16 [config.py:3320] Your device 'cpu' doesn't support torch.bfloat16. Falling back to torch.float16 for compatibility.
WARNING 07-27 21:13:16 [config.py:3371] Casting torch.bfloat16 to torch.float16.
INFO 07-27 21:13:16 [config.py:1472] Using max model len 8192
INFO 07-27 21:13:16 [arg_utils.py:1746] cpu is experimental on VLLM_V1=1. Falling back to V0 Engine.
WARNING 07-27 21:13:16 [cpu.py:131] Environment variable VLLM_CPU_KVCACHE_SPACE (GiB) for CPU backend is not set, using 4 by default.
INFO 07-27 21:13:16 [api_server.py:258] Started engine process with PID 93561
INFO 07-27 21:13:18 [__init__.py:244] Automatically detected platform: cpu.
INFO 07-27 21:13:19 [llm_engine.py:230] Initializing a V0 LLM engine (v0.9.2) with config: model='Qwen/Qwen3-0.6B', speculative_config=None, tokenizer='Qwen/Qwen3-0.6B', skip_tokenizer_init=False, tokenizer_mode=None, override_neuron_config={}, tokenizer_revision=None, trust_remote_code=False, dtype=torch.float16, max_seq_len=8192, download_dir=None, load_format=LoadFormat.AUTO, tensor_parallel_size=1, pipeline_parallel_size=1, disable_custom_all_reduce=True, quantization=None, enforce_eager=False, kv_cache_dtype='auto', device_config=cpu, decoding_config=DecodingConfig(backend='auto', disable_fallback=False, disable_any_whitespace=False, disable_additional_properties=False, reasoning_backend=''), observability_config=ObservabilityConfig(gishov_hidden_metrics_for_version=None, otlp_traces_endpoint=None, collect_detailed_traces=None), seed=0, served_model_name='Qwen/Qwen3-0.6B', num_scheduler_steps=1, multi_step_stream_outputs=True, enable_prefix_caching=None, chunked_prefill_enabled=False, use_async_output_proc=False, pooler_config=None, compilation_config=[{"level": "debug_dump_path": "", "backend": "", "custom_ops": [], "splitting_ops": []}, {"use_inductor": true, "compile_sizes": []}], "inductor_compile_config": {"enable_auto_functionalized_v2": false}, {"use_cudagraph": true, "cudagraph_num_of_warmups": 0, "cudagraph_capture_sizes": []}, {"cudagraph_copy_inputs": false, "full_cuda_graph": false, "max_capture_size": 256, "local_cache_dir": null}, {"use_cached_outputs": true}, {"WARNING 07-27 21:13:20 [cpu_worker.py:447] Auto thread-binding is not supported due to the lack of package num and psutil, fallback to no thread-binding. To get better performance, please try to manually bind threads.
INFO 07-27 21:13:20 [cpu.py:69] Using Torch SDPA backend.
INFO 07-27 21:13:20 [importing.py:63] Triton not installed or not compatible; certain GPU-related functions will not be available.
INFO 07-27 21:13:20 [parallel_state.py:1076] rank 0 in world size 1 is assigned as DP rank 0, PP rank 0, TP rank 0, EP rank 0
INFO 07-27 21:13:20 [weight_utils.py:292] Using model weights format ['*.safetensors']
INFO 07-27 21:13:21 [weight_utils.py:345] No model.safetensors.index.json found in remote.
Loading safetensors checkpoint shards:  0% Completed | 0/1 [00:00:00, ?it/s]
Loading safetensors checkpoint shards: 100% Completed | 1/1 [00:01:00:00, 1.13s/it]
Loading safetensors checkpoint shards: 100% Completed | 1/1 [00:01:00:00, 1.13s/it]

INFO 07-27 21:13:22 [default_loader.py:272] Loading weights took 1.13 seconds
INFO 07-27 21:13:22 [executor_base.py:113] # cpu blocks: 2340, # CPU blocks: 0
INFO 07-27 21:13:22 [executor_base.py:118] Maximum concurrency for 8192 tokens per request: 4.57x
INFO 07-27 21:13:23 [llm_engine.py:428] init engine (profile, create kv cache, warmup model) took 0.58 seconds
INFO 07-27 21:13:23 [serving_chat.py:85] "auto" tool choice has been enabled, please note that while the parallel_tool_calls client option is preset for
```

```
zerohertz@zerohertz-mm:~/PyCon_KR_2025_Tutorial_vLLM
$ curl -X POST http://localhost:8000/v1/chat/completions \
-H "Content-Type: application/json" \
-d '{
  "model": "Qwen/Qwen3-0.6B",
  "messages": [
    {
      "role": "user",
      "content": "What is the weather like in Seoul?"
    }
  ],
  "tools": [
    {
      "type": "function",
      "function": {
        "name": "get_weather",
        "description": "Get the current weather in a given location",
        "parameters": {
          "type": "object",
          "properties": {
            "location": {
              "type": "string",
              "description": "The city and state, e.g. San Francisco, CA"
            },
            "unit": {
              "type": "string",
              "enum": ["celsius", "fahrenheit"]
            }
          },
          "required": ["location"]
        }
      }
    ]
  ],
  "tool_choice": "auto"
}' | jq
% Total    % Received % Xferd  Average Speed   Time     Time     Time  Current
100 1946  100 1133  100  813  135   97  0:00:08  0:00:08  --:--:-- 273
{
  "id": "chatmpl-9f3123c435ba4bfcb5b2373df31af0e5",
  "object": "chat.completion",
  "created": 1754911578,
  "model": "Qwen/Qwen3-0.6B",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "

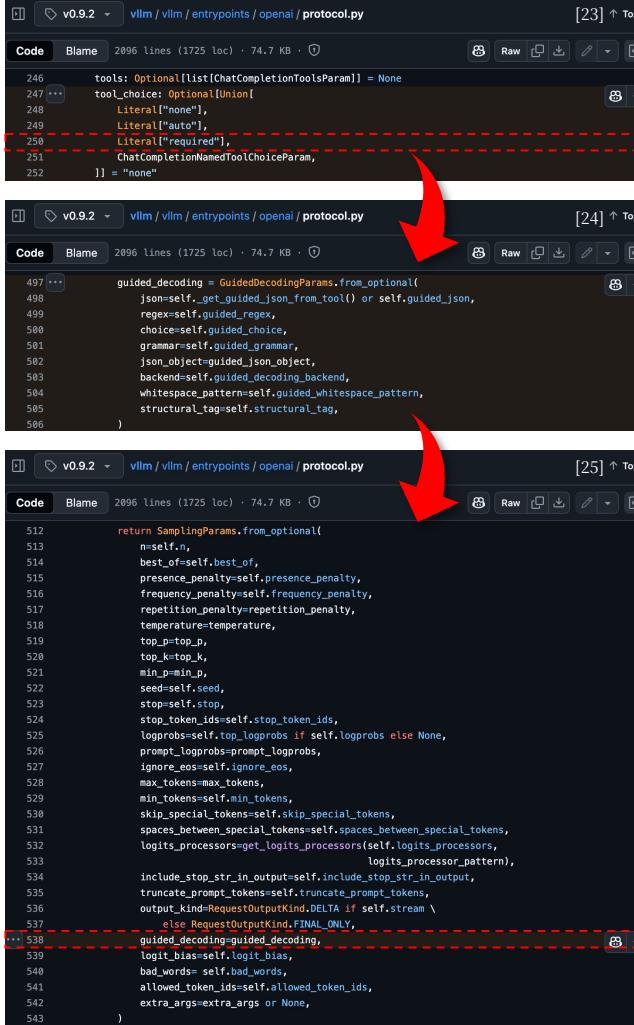
```

[20] [https://docs.vllm.ai/en/v0.9.2/features/tool\\_calling.html](https://docs.vllm.ai/en/v0.9.2/features/tool_calling.html)

[21] <https://qwen.readthedocs.io/en/latest/deployment/vllm.html#parsing-tool-calls>

# Tool calling

## 요청에 required를 입력한다면? [22]



```

v0.9.2 - vllm/vllm/entrypoints/openai/protocol.py [23] ↑ Top
Code Blame 2096 lines (1725 loc) · 74.7 KB · Raw Download Edit
246     tools: Optional[list[ChatCompletionToolsParam]] = None
247     ...
248     tool_choice: Optional[Union[
249         Literal["none"],
250         Literal["auto"],
251         Literal["required"],
252     ]] = "none"

```

Red box highlights the 'required' parameter in the ChatCompletionToolsParam class definition.

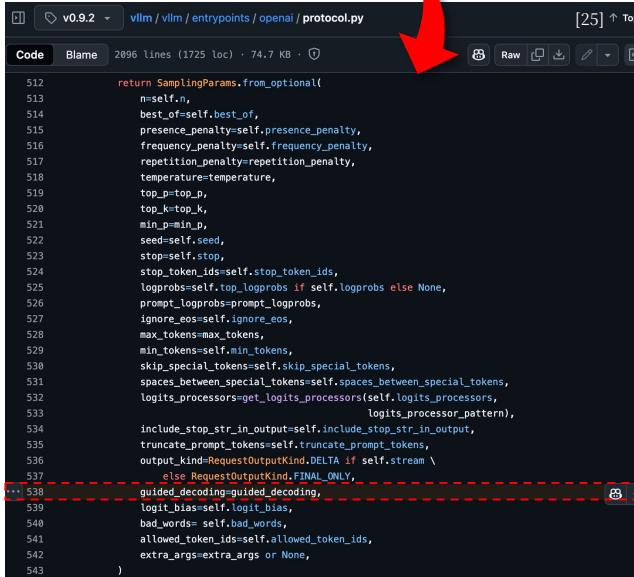


```

v0.9.2 - vllm/vllm/entrypoints/openai/protocol.py [24] ↑ Top
Code Blame 2096 lines (1725 loc) · 74.7 KB · Raw Download Edit
497     ...
498     guided_decoding = GuidedDecodingParams.from_optional(
499         json=self._get_guided_json_from_tool() or self.guided_json,
500         regex=self.guided_regex,
501         choice=self.guided_choice,
502         grammar=self.guided_grammar,
503         json_object=guided_json_object,
504         backend=self.guided_decoding_backend,
505         whitespace_pattern=self.guided_whitespace_pattern,
506         structural_tag=self.structural_tag,
507     )

```

Red box highlights the 'guided\_decoding' parameter in the ChatCompletionParams class definition.



```

v0.9.2 - vllm/vllm/entrypoints/openai/protocol.py [25] ↑ Top
Code Blame 2096 lines (1725 loc) · 74.7 KB · Raw Download Edit
512     return SamplingParams.from_optional(
513         ...
514         best_of=self.best_of,
515         presence_penalty=self.presence_penalty,
516         frequency_penalty=self.frequency_penalty,
517         repetition_penalty=repetition_penalty,
518         temperature=temperature,
519         top_p=top_p,
520         top_k=top_k,
521         min_p=min_p,
522         seed=seed,
523         stop=stop,
524         stop_token_ids=self.stop_token_ids,
525         logprobs=self.top_logprobs if self.logprobs else None,
526         prompt_logprobs=prompt_logprobs,
527         ignore_eos=ignore_eos,
528         max_tokens=max_tokens,
529         min_tokens=min_tokens,
530         skip_special_tokens=skip_special_tokens,
531         spaces_between_special_tokens=spaces_between_special_tokens,
532         logits_processors=get_logits_processors(self.logits_processors,
533             ...
534             logits_processor_pattern),
535         include_stop_str_in_output=include_stop_str_in_output,
536         truncate_prompt_tokens=truncate_prompt_tokens,
537         output_kind=RequestOutputKind.DELTA if self.stream \
538             ...
539             else RequestOutputKind.FINAL_ONLY,
540             ...
541             ...
542             ...
543         )

```

Red box highlights the 'presence\_penalty' parameter in the SamplingParams class definition.

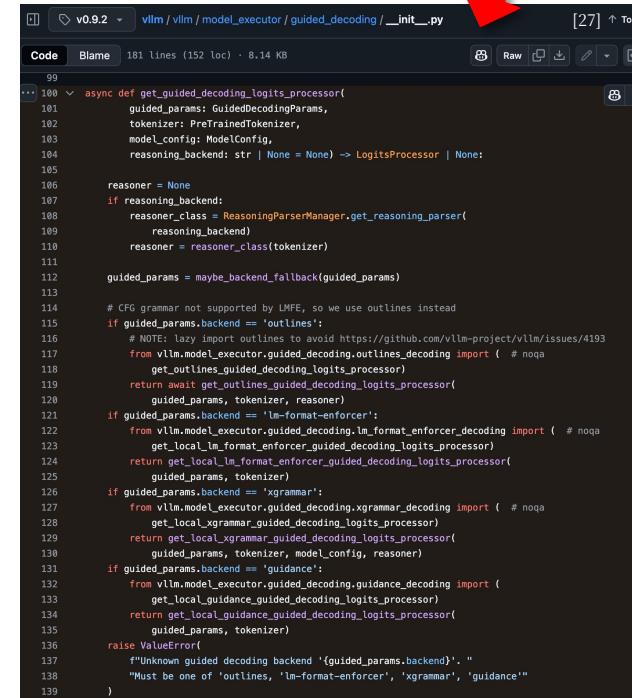


```

v0.9.2 - vllm/vllm/engine/async_llm_engine.py [26] ↑ Top
Code Blame 1200 lines (1012 loc) · 47.9 KB · Raw Download Edit
473     ...
474     if isinstance(params, SamplingParams) and \
475         params.guided_decoding is not None:
476         # Guided decoding has an async implementation for building logits
477         # processors in a separate threadpool.
478         # We want to invoke that here instead of using the blocking
479         # implementation in the LLMEngine
480         params = await build_guided_decoding_logits_processor_async(
481             sampling_params=params,
482             tokenizer=await self.get_tokenizer_async(clora_request),
483             default_guided_backend=self.decoding_config.backend,
484             reasoning_backend=self.decoding_config.reasoning_backend,
485             model_config=self.model_config)

```

Red box highlights the 'params' parameter in the await build\_guided\_decoding\_logits\_processor\_async function call.



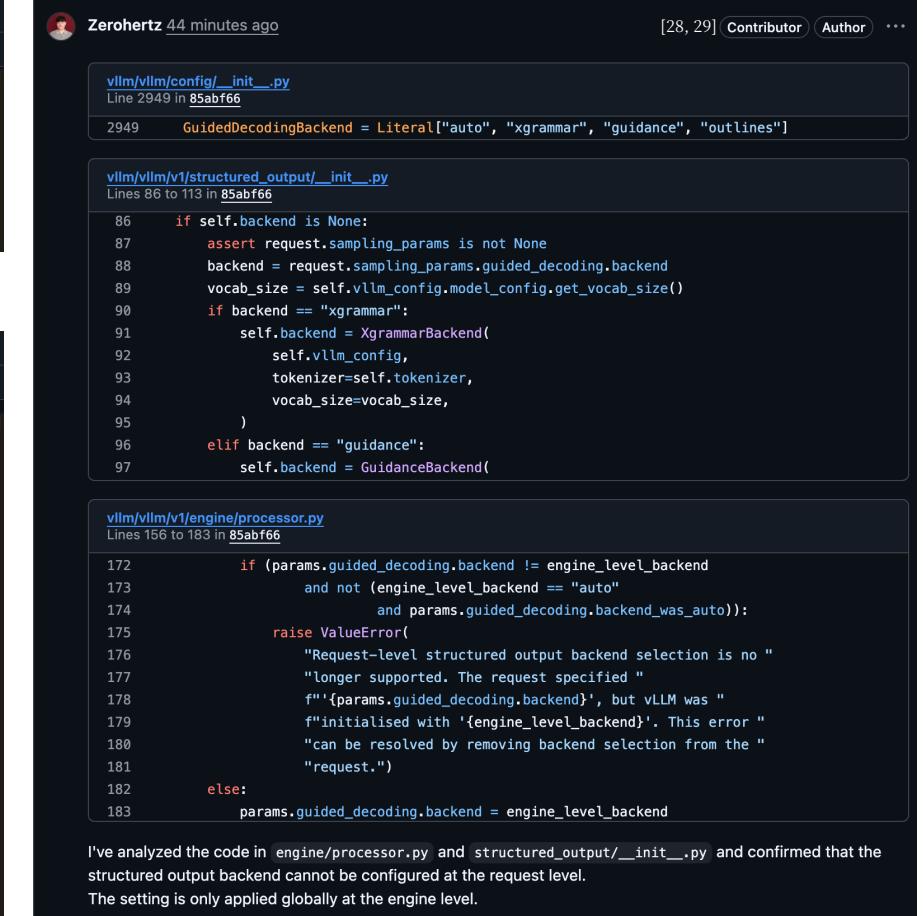
```

v0.9.2 - vllm/vllm/model_executor/guided_decoding/_init_.py [27] ↑ Top
Code Blame 181 lines (152 loc) · 8.14 KB · Raw Download Edit
99
100     ...
101     @async def get_guided_decoding_logits_processor(
102         guided_params: GuidedDecodingParams,
103         tokenizer: PreTrainedTokenizer,
104         model_config: ModelConfig,
105         reasoning_backend: str | None = None) -> LogitsProcessor | None:
106
107         reasoner = None
108         if reasoning_backend:
109             reasoner_class = ReasoningParserManager.get_reasoning_parser(
110                 reasoning_backend)
111             reasoner = reasoner_class(tokenizer)
112
113         guided_params = maybe_backend_fallback(guided_params)
114
115         # CFG grammar not supported by LMFE, so we use outlines instead
116         if guided_params.backend == 'outlines':
117             # NOTE: lazy import outlines to avoid https://github.com/vllm-project/vllm/issues/4193
118             from vllm.model_executor.guided_decoding.outlines_decoding import ( # noqa
119                 get_outlines_guided_decoding_logits_processor)
120             return await get_outlines_guided_decoding_logits_processor(
121                 guided_params, tokenizer, reasoner)
122         if guided_params.backend == 'lm-format-enforcer':
123             from vllm.model_executor.guided_decoding.lm_format_enforcer_decoding import ( # noqa
124                 get_local_lm_format_enforcer_guided_decoding_logits_processor)
125             return get_local_lm_format_enforcer_guided_decoding_logits_processor(
126                 guided_params, tokenizer)
127         if guided_params.backend == 'xgrammar':
128             from vllm.model_executor.guided_decoding.xgrammar_decoding import ( # noqa
129                 get_local_xgrammar_guided_decoding_logits_processor)
130             return get_local_xgrammar_guided_decoding_logits_processor(
131                 guided_params, tokenizer, model_config, reasoner)
132
133         if guided_params.backend == 'guidance':
134             from vllm.model_executor.guided_decoding.guidance_decoding import ( # noqa
135                 get_local_guidance_guided_decoding_logits_processor)
136             return get_local_guidance_guided_decoding_logits_processor(
137                 guided_params, tokenizer)
138
139         raise ValueError(
140             f"Unknown guided decoding backend '{guided_params.backend}'."
141             "Must be one of 'outlines', 'lm-format-enforcer', 'xgrammar', 'guidance'")

```

Red box highlights the 'backend' parameter in the 'if backend == 'xgrammar'' block.

## 요청에 따른 backend 지정은 deprecated



Zerohertz 44 minutes ago

vllm/vlm/config/\_init\_.py Line 2949 in 85abf66

```

2949     GuidedDecodingBackend = Literal["auto", "xgrammar", "guidance", "outlines"]

```

Red box highlights the 'GuidedDecodingBackend' parameter.

vllm/vlm/v1/structured\_output/\_init\_.py Lines 86 to 113 in 85abf66

```

86     if self.backend is None:
87         assert request.sampling_params is not None
88         backend = request.sampling_params.guided_decoding.backend
89         vocab_size = self.vlm_config.model_config.get_vocab_size()
90         if backend == "xgrammar":
91             self.backend = XgrammarBackend(
92                 self.vlm_config,
93                 tokenizer=self.tokenizer,
94                 vocab_size=vocab_size,
95             )
96         elif backend == "guidance":
97             self.backend = GuidanceBackend(

```

vllm/vlm/v1/engine/processor.py Lines 156 to 183 in 85abf66

```

172     if (params.guided_decoding.backend != engine_level_backend
173         and not (engine_level_backend == "auto"
174                  and params.guided_decoding.backend_was_auto)):
175         raise ValueError(
176             "Request-level structured output backend selection is no "
177             "longer supported. The request specified "
178             f"'{params.guided_decoding.backend}', but VLLM was "
179             f"initialised with '{engine_level_backend}'. This error "
180             "can be resolved by removing backend selection from the "
181             "request.")
182     else:
183         params.guided_decoding.backend = engine_level_backend

```

I've analyzed the code in engine/processor.py and structured\_output/\_init\_.py and confirmed that the structured output backend cannot be configured at the request level.  
The setting is only applied globally at the engine level.

- [22] [https://docs.vllm.ai/en/v0.9.2/features/structured\\_outputs.html](https://docs.vllm.ai/en/v0.9.2/features/structured_outputs.html)
- [23] <https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/entrypoints/openai/protocol.py#L247-L252>
- [24] <https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/entrypoints/openai/protocol.py#L497-L506>
- [25] <https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/entrypoints/openai/protocol.py#L538>
- [26] [https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/engine/async\\_llm\\_engine.py#L473-L484](https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/engine/async_llm_engine.py#L473-L484)
- [27] [https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/model\\_executor/guided\\_decoding/\\_init\\_.py#L100-L139](https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/model_executor/guided_decoding/_init_.py#L100-L139)
- [28] <https://github.com/vllm-project/vllm/pull/22740>
- [29] <https://github.com/vllm-project/vllm/pull/22772>

# Reasoning

- “chat\_template\_kwargs”的 “enable\_thinking”을 통해 reasoning 유무 설정 가능

```
zerohertz@zerohertz-mm:~/PyCon_KR_2025_Tutorial_vLLM$ curl -X POST http://localhost:8000/v1/chat/completions \
-H "Content-Type: application/json" \
-d '{
  "model": "Qwen/Qwen3-0.6B",
  "messages": [
    {
      "role": "user",
      "content": "Hello, PyCon Korea 2025!"
    }
  ],
  "chat_template_kwargs": {"enable_thinking": false}
}' | jq
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Left Speed
100  739  100  526 100  213  113  45  0:00:04  0:00:04  ---:--- 158
{
  "id": "chatmpl-8eb6c33541a84eb4b96562ef7c858290",
  "object": "chat.completion",
  "created": 1754912052,
  "model": "Qwen/Qwen3-0.6B",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "reasoning_content": "",
        "content": "Hello, PyCon Korea 2025! 🎉 What's the best way to celebrate this event? Let me know!",
        "tool_calls": []
      },
      "logprobs": 0,
      "finish_reason": "stop",
      "stop_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 23,
    "total_tokens": 52,
    "completion_tokens": 29,
    "prompt_tokens_details": {},
    "prompt_logprobs": 0,
    "kv_transfer_params": {}
  }
}
```

```
zerohertz@zerohertz-mm:~/PyCon_KR_2025_Tutorial_vLLM$ curl -X POST http://localhost:8000/v1/chat/completions \
-H "Content-Type: application/json" \
-d '{
  "model": "Qwen/Qwen3-0.6B",
  "messages": [
    {
      "role": "user",
      "content": "Hello, PyCon Korea 2025!"
    }
  ],
  "chat_template_kwargs": {"enable_thinking": true}
}' | jq
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Left Speed
100 1400  100 1188 100  212  121  21  0:00:10  0:00:09  0:00:01 263
{
  "id": "chatmpl-2eb0a23e56c14eb2b8db3eaf0330c21f",
  "object": "chat.completion",
  "created": 1754912334,
  "model": "Qwen/Qwen3-0.6B",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "reasoning_content": {
          "content": "<|im_start|>nOkay, the user just said \"Hello, PyCon Korea 2025!\" So I need to respond to that. Let me start by acknowledging their message. Maybe say something like \"Hello, PyCon Korea 2025!\" to confirm the date. Then, I should invite them to attend the event. I can mention the date, location, and maybe the main themes. Also, I should keep the tone friendly and enthusiastic. Let me make sure to include all the key details they might need. Alright, let's put that together.<|im_end|>nHello, PyCon Korea 2025! 🎉 I'm excited to welcome you to the prestigious event where innovation and creativity collide. Let me know if you're looking forward to the themes, workshops, or networking opportunities. Have a great time at PyCon Korea 2025!",
          "tool_calls": []
        },
        "logprobs": 0,
        "finish_reason": "stop",
        "stop_reason": "stop"
      }
    },
    "usage": {
      "prompt_tokens": 19,
      "total_tokens": 200,
      "completion_tokens": 181,
      "prompt_tokens_details": {},
      "prompt_logprobs": 0,
      "kv_transfer_params": {}
    }
}
```

```
INFO 08-11 20:34:12 [logger.py:43] Received request chatmpl-8eb6c33541a84eb4b96562ef7c858290: prompt: '<|im_start|>user\nHello, PyCon Korea 2025!<|im_end|>\n<|im_start|>assistant\n<think>\n</think>\n\n', params: SamplingParams(n=1, presence_penalty=0.6, temperature=0.6, top_k=20, top_p=0.95, seed=None, stop=[], stop_token_ids=[], bad_words=[], include_stop_str_in_output=False, ignore_eos=False, max_tokens=8169, min_tokens=0, logprobs=None, prompt_logprobs=None, guided_decoding=None, extra_args=None), prompt_token_ids: None, prompt_embeds shape: None, lora_request: None, prompt_adapter_request: None.
INFO 08-11 20:34:12 [engine.py:317] Added request chatmpl-8eb6c33541a84eb4b96562ef7c858290.
WARNING 08-11 20:34:12 [cpu.py:250] Pin memory is not supported on CPU.
INFO: 127.0.0.1:60893 - "POST /v1/chat/completions HTTP/1.1" 200 OK
INFO 08-11 20:34:23 [metrics.py:417] Avg prompt throughput: 1.9 tokens/s, Avg generation throughput: 2.4 tokens/s, Running: 0 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 0.0%, CPU KV cache usage: 0.0%.
INFO 08-11 20:34:33 [metrics.py:417] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 0.0 tokens/s, Running: 0 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 0.0%, CPU KV cache usage: 0.0%.
INFO 08-11 20:38:54 [logger.py:43] Received request chatmpl-2eb0a23e56c14eb2b8db3eaf0330c21f: prompt: '<|im_start|>user\nHello, PyCon Korea 2025!<|im_end|>\n<|im_start|>assistant\n', params: SamplingParams(n=1, presence_penalty=0.6, temperature=0.6, top_k=20, top_p=0.95, seed=None, stop=[], stop_token_ids=[], bad_words=[], include_stop_str_in_output=False, ignore_eos=False, max_tokens=8173, min_tokens=0, logprobs=None, prompt_logprobs=None, skip_special_tokens=None, extra_args=None), prompt_embeds shape: None, lora_request: None, prompt_adapter_request: None.
```

# Reasoning

- “--reasoning-parser”를 통해 모델에 따른 적절한 parser 설정 필요 [30, 31]
- 기존에는 “--enable-reasoning” 옵션이 존재했으나 deprecated [32]

```
vllm serve Owen/Owen3-0.6B --max-model-len 8192 --reasoning-parser qwen3
$ vllm serve Owen/Owen3-0.6B --max-model-len 8192 --reasoning-parser qwen3
INFO 07-27 21:17:55 [api_server.py:1395] VLLM API server version 0.9.2
INFO 07-27 21:17:55 [cli_args.py:325] non-default args: {'model': 'Owen/Owen3-0.6B', 'max_model_len': 8192, 'reasoning_parser': 'qwen3'}
INFO 07-27 21:17:56 [config.py:841] This model supports multiple tasks: {'reward', 'classify', 'generate', 'embed'}. Defaulting to 'generate'.
WARNING 07-27 21:17:58 [config.py:3320] Your device 'cpu' doesn't support torch.bfloat16. Falling back to torch.float16 for compatibility.
WARNING 07-27 21:17:58 [config.py:3371] Casting torch.bfloat16 to torch.float16.
INFO 07-27 21:17:58 [config.py:1472] Using max model len 8192
INFO 07-27 21:17:58 [arg_utils.py:1746] cpu is experimental on VLLM USE_V1=1. Falling back to V0 Engine.
WARNING 07-27 21:17:58 [cpu.py:131] Environment variable VLLM_CPU_KVCACHE_SPACE (GiB) for CPU backend is not set, using 4 by default.
INFO 07-27 21:17:58 [api_server.py:268] Started engine process with PID 9484
INFO 07-27 21:18:00 [__init__.py:244] Automatically detected platform cpu
INFO 07-27 21:18:01 [llm_engine.py:230] Initializing a V0 LLM engine (v0.9.2) with config: model='Owen/Owen3-0.6B', speculative_config=None, tokenizer='Owen/Owen3-0.6B', skip_tokenizer_init=False, tokenizer_mode=auto, revision=None, override_neuron_config={}, tokenizer_revision=None, trust_remote_code=False, dtype=torch.float16, max_seq_len=8192, download_dir=None, load_format=LoadFormat.AUTO, tensor_parallel_size=1, pipeline_parallel_size=1, disable_custom_all_reduce=True, quantization=None, enforce_eager=False, kv_cache_dtype=auto, device_config=cpu, decoding_config=DecodingConfig(backend='auto', do_isable_fallback=False, disable_any_whitespace=False, disable_additional_properties=False, reasoning_backend='qwen3'), observability_config=ObservabilityConfig(show_hidden_metrics_for_version=None, otlp_traces_endpoint=None, collect_detailed_traces=None), seed=0, served_model_name='Owen/Owen3-0.6B', num_sharder_steps=1, multi_step_stream_outputs=True, enable_prefix_caching=None, chunked_prefill_enabled=False, use_async_output_proc=False, pooler_config=None, compilation_config={'level': 0}, debug_dump_path='', cache_dir='', custom_ops=[], splitting_ops=[], use_inductor=True, compile_size='[]', inductor_compile_config={'enable_auto_functionalized_v2': False}, inductor_passes={}, use_cudagraph=True, cudagraph_num_of_warmups=0, cudagraph_h_capture_sizes=[], cudagraph_copy_inputs=False, full_cuda_graph=False, max_capture_size=256, local_cache_dir=null, use_cached_outputs=True, WARNING 07-27 21:18:02 [cpu_worker.py:447] Auto thread-binding is not supported due to the lack of package numa and putils, fallback to no thread-binding. To get better performance, please try to manually bind threads.
INFO 07-27 21:18:02 [cpu.py:69] Using Torch SDDPA backend.
INFO 07-27 21:18:02 [importing.py:63] Triton not installed or not compatible; certain GPU-related functions will not be available.
INFO 07-27 21:18:02 [parallel_state.py:1076] rank 0 in world size 1 is assigned as DP rank 0, PP rank 0, TP rank 0, EP rank 0
INFO 07-27 21:18:03 [weight_utils.py:292] Using model weights format ['*.safetensors']
INFO 07-27 21:18:04 [weight_utils.py:345] No model.safetensors.index.json found in remote.
```

```
zerohertz@zerohertz-mm:~/PyCon_KR_2025_Tutorial_vLLM$ curl -X POST http://localhost:8000/v1/chat/completions \
-H "Content-Type: application/json" \
-d '{
  "model": "Owen/Owen3-0.6B",
  "messages": [
    {
      "role": "user",
      "content": "Hello, PyCon Korea 2025!"
    },
    "chat_template_kwarg": {"enable_thinking": true}
  ]' | jq
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload   Total   Spent  Left  Speed
100 1704  100 1492 100  212  100    14  0:00:15  0:00:14  0:00:01 324
{"id": "chatcmpl-c78c1ce30a844a71b70ff0445693075d",
"object": "chat.completion",
"created": 1754912819,
"model": "Owen/Owen3-0.6B",
"choices": [
{
  "index": 0,
  "message": {
    "role": "assistant",
    "content": "\nOkay, the user is asking for a greeting at PyCon Korea 2025. Let me start by understanding the context. PyCon is a major tech conference, so the response should be formal and enthusiastic.\nFirst, I should address the event with its name, PyCon Korea 2025. Then, mention that it's a big event for tech professionals. It's important to highlight the significance and the excitement of attending. Maybe include some key points like the theme, the number of participants, and the impact of the event. Also, express enthusiasm for the attendees and the opportunities they'll have. Keep the tone positive and inviting. Need to make sure the response is concise but covers all necessary aspects without being too lengthy.\n",
    "tool_calls": []
  },
  "logprobs": ,
  "finish_reason": "stop",
  "stop_reason": "
},
"usage": {
  "prompt_tokens": 19,
  "total_tokens": 240,
  "completion_tokens": 221,
  "prompt_tokens_details": "
},
"prompt_logprobs": ,
"kv_transfer_params": "
}
]
[30] https://docs.vllm.ai/en/v0.9.2/features/reasoning\_outputs.html
[31] https://qwen.readthedocs.io/en/latest/deployment/vllm.html#parsing-thinking-content
[32] https://github.com/vllm-project/vllm/blob/v0.9.2/vllm/engine/arg\_utils.py#L626-L634
```

# Chat Template

## ▣ 기본적으로 tokenizer\_config.json의 “chat\_template” 값 사용

```

1  {%- if tools %}
2    {{ '<|im_start|>system\n' }}
3    {%- if messages[0].role == 'system' %}
4      {{ messages[0].content + '\n\n' }}
5    {%- endif %}
6    {%- for tool in tools %}
7      {{ '|tool|' | toJSON }}
8    {%- endfor %}
9    {%- for message in messages[1:-1] %}
10      {{ '|im_start|>' + message.content + '|im_end|>\n' }}
11    {%- endif %}
12    {%- if messages[-1].role == 'system' %}
13      {{ '|im_start|>system\n' + messages[-1].content + '|im_end|>\n' }}
14    {%- endif %}
15    {%- if ns.multi_step_tool and message.role == 'user' and message.content.startswith('<tool_response>') and message.content.endswith('</tool_response>') %}
16      {{ '|im_start|>system\n' + messages[-1].content + '|im_end|>\n' }}
17    {%- endif %}
18    {%- if message in messages[1:-1] %}
19      {{ '|set index = (messages.length - 1) |op.index0 |' }}
20      {{ '|if ns.multi_step_tool == false |' }}
21        {{ '|set ns.last_query_index = index |' }}
22      {{ '|endif |' }}
23    {%- endif %}
24  {%- endif %}
25  {%- for message in messages %}
26    {%- if message.role == 'user' or message.role == 'assistant' %}
27      {{ '|set content = message.content |' }}
28    {%- else %}
29      {{ '|set content = '' |' }}
30    {%- endif %}
31    {%- if message.role == 'user' or (message.role == "system" and not loop.first) %}
32      {{ '|<|im_start|>' + message.role + '\n' + content + '<|im_end|> + '\n' }}
33    {%- elif message.role == "assistant" %}
34      {{ '|set reasoning_content = '' |' }}
35      {{ '|if message.reasoning_content is string |' }}
36        {{ '|set reasoning_content = message.reasoning_content |' }}
37      {%- else %}
38        {{ '|if <thinking> in content |' }}
39        {{ '|set reasoning_content = content.split(</thinking>)[0].rstrip(\n).split(<thinking>)[-1].lstrip(\n) |' }}
40        {{ '|set content = content.split(</thinking>)[-1].lstrip(\n) |' }}
41      {%- endif %}
42    {%- endif %}
43    {%- if loop.index0 == ns.last_query_index %}
44      {%- if loop.last or (not loop.last and reasoning_content) %}
45        {{ '|<|im_start|>' + message.role + '\n|thinking\n' + reasoning_content.strip('\n') + '\n</thinking>\n\n' + content.lstrip('\n') |' }}
46      {%- else %}
47        {{ '|<|im_start|>' + message.role + '\n' + content |' }}
48      {%- endif %}
49    {%- endif %}
50    {%- if <|im_start|> + message.role + '\n' + content %}
51    {%- endif %}
52  {%- endif %}
53  {%- if message.tool_calls %}
54    {{ '|for tool_call in message.tool_calls |' }}
55    {%- if (loop.first and content) or (not loop.first) %}
56      {{ '|&nbsp; |' }}
57    {%- endif %}
58    {%- if tool_call.function %}
59      {{ '|&nbsp; set tool_call = tool_call.function |' }}
60    {%- endif %}
61    {{ '|<|im_start|>|name: "' + tool_call.function + '"\n' |' }}
62    {{ '|<|im_start|>|arguments: "' + tool_call.arguments + '"\n' |' }}
63    {%- if tool_call.arguments is string %}
64      {{ '|&nbsp; set tool_call.arguments = ' + tool_call.arguments |' }}
65    {%- elif tool_call.arguments | toJSON %}
66      {{ '|&nbsp; set tool_call.arguments = ' + tool_call.arguments | toJSON |' }}
67    {%- endif %}
68    {{ '|<|im_end|>\n' |' }}
69  {%- endif %}
70  {%- if <|im_end|>\n' %}
71  {%- elif message.role == "tool" %}
72  {%- if loop.first or (messages[loop.index0 - 1].role != "tool") %}
73    {{ '|<|im_start|>user\n' |' }}
74  {%- endif %}
75  {%- endif %}
76  {%- if <|im_start|>|name: "tool_response"\n' %}
77  {%- content %}
78  {%- if <|im_start|>|name: "tool_response"\n' %}
79  {%- if loop.last or (messages[loop.index0 + 1].role != "tool") %}
80    {{ '|<|im_end|>\n' |' }}
81  {%- endif %}
82  {%- endif %}
83  {%- add_generation_prompt %}
84  {%- if add_generation_prompt %}
85    {{ '|<|im_start|>assistant\n' |' }}
86  {%- if enable_thinking is defined and enable_thinking is false %}
87    {{ '|<think>\n\n</think>\n\n' |' }}
88  {%- endif %}
89  {%- endif %}

```

[33]

# Chat Template

2. OpenAI-Compatible Server

## □ “--chat-template”으로 새로운 chat template 적용 가능 [34]

```
messages = [
    {"role": "system", "content": "You are a helpful assistant."},
    {"role": "user", "content": "Hello, InstructKR!"},
    {"role": "assistant", "content": "Hello! How can I help you today?"},
    {"role": "user", "content": "Can you explain what InstructKR is?"},
    {
        "role": "assistant",
        "content": "InstructKR is a Korean instruction-following dataset and research initiative focused on improving language models' ability to follow instructions in Korean.",
    },
    {"role": "user", "content": "What's the weather like in Seoul today?"},
    {
        "role": "assistant",
        "content": "I'll help you check the weather in Seoul.",
        "tool_calls": [
            {
                "id": "call_1",
                "type": "function",
                "function": {
                    "name": "get_weather",
                    "arguments": {"location": "Seoul, South Korea"}
                }
            }
        ],
        "content": "The weather in Seoul today is sunny with a temperature of 22°C (72°F). There's a light breeze and clear skies."
    },
    {
        "role": "assistant",
        "content": "Based on the weather information, Seoul is having a pleasant day today! It's sunny with a comfortable temperature of 22°C (72°F), light breeze, and clear skies."
    },
    {"role": "user", "content": "Thanks! Can you help me with something else?"}
]
prompt = processor.apply_chat_template(
    messages, tokenize=False, add_generation_prompt=True
)
```

```
<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
Hello, InstructKR!<|im_end|>
<|im_start|>assistant
Hello! How can I help you today?<|im_end|>
<|im_start|>user
Can you explain what InstructKR is?<|im_end|>
<|im_start|>assistant
InstructKR is a Korean instruction-following dataset and research initiative focused on improving language models' ability to follow instructions in Korean.<|im_end|>
<|im_start|>user
What's the weather like in Seoul today?<|im_end|>
<|im_start|>assistant
I'll help you check the weather in Seoul.
<tool_call>
{'name': 'get_weather', 'arguments': {'location': 'Seoul, South Korea'}}
</tool_call><|im_end|>
<|im_start|>user
<tool_response>
The weather in Seoul today is sunny with a temperature of 22°C (72°F). There's a light breeze and clear skies.
</tool_response><|im_end|>
<|im_start|>assistant
Based on the weather information, Seoul is having a pleasant day today! It's sunny with a comfortable temperature of 22°C (72°F), light breeze, and clear skies. Perfect!
<|im_start|>user
Thanks! Can you help me with something else?<|im_end|>
<|im_start|>assistant
```

Name	Last commit message	Last commit date
...		
offline_inference	[VLM] Add video support for Intern-S1 (#2167)	40 minutes ago
online_serving	[CI/Build][Doc] Clean up more docs that point to old ...	8 hours ago
others	[CI/Build][Doc] Move existing benchmark scripts in C...	yesterday
pyproject.toml	Convert examples to ruff-format (#18400)	2 months ago
template_alpaca.jinja	Support chat template and echo for chat API (#1756)	2 years ago
template_baichuan.jinja	Fix Baichuan chat template (#3340)	last year
template_chatglm.jinja	Add chat templates for ChatGLM (#3418)	last year
template_chatglm2.jinja	Add chat templates for ChatGLM (#3418)	last year
template_chatml.jinja	Support chat template and echo for chat API (#1756)	2 years ago
template_dse_qwen2_vijinjia	[Model] Adding Support for Qwen2VL as an Embedd...	8 months ago
template_falcon.jinja	Add chat templates for Falcon (#3420)	last year
template_falcon_180b.jinja	Add chat templates for Falcon (#3420)	last year
template_inkbot.jinja	Support chat template and echo for chat API (#1756)	2 years ago
template_telmlm.jinja	[Model] Support TelmeLM Model (#15023)	4 months ago
template_vlm2vec.jinja	[Frontend] Use a proper chat template for VLM2Vec ...	9 months ago
tool_chat_template_deepsseek1.jinja	Fix DeepSeek-R1-0528 chat template (#20717)	2 weeks ago
tool_chat_template_deepsseek3.jinja	[Feature] Support DeepSeekV3 Function Call (#17784)	2 months ago
tool_chat_template_granite.jinja	Change granite chat template to keep json list format...	8 months ago
tool_chat_template_granite_20b_fc.jinja	[Model] tool calling support for ibm-granite/granite-2...	9 months ago
tool_chat_template_hermes.jinja	[Bugfix] Fix Hermes tool call chat template bug (#82...	10 months ago
tool_chat_template_hunyuan_s13b.jinja	[Model] Add ToolParser and MoE Config for Hunyuan...	last week
tool_chat_template_internlm2_tool.jinja	[Frontend][Feature] support tool calling for internlm/...	9 months ago
tool_chat_template_llama3.1_json.jinja	[Bugfix][Frontend] Update Llama Chat Templates to ...	8 months ago
tool_chat_template_llama3.2_json.jinja	[Misc] Update llama 3.2 template to support system ...	7 months ago
tool_chat_template_llama3.2_pythonic.ji...	[Frontend] Fix typo in tool chat templates for llama3...	3 months ago
tool_chat_template_llama4_4_json.jinja	Add chat template for Llama 4 models (#16428)	3 months ago
tool_chat_template_llama4_pythonic.jinja	[Frontend][Bug Fix] Update llama4 pythonic jinja tem...	2 months ago
tool_chat_template_minimax_m1.jinja	[Feature] Support MiniMax-M1 function calls features...	3 weeks ago
tool_chat_template_mistral.jinja	[Feature] OpenAI-Compatible Tools API + Streaming ...	10 months ago
tool_chat_template_mistral3.jinja	[Bugfix] Fix tool call template validation for Mistral m...	2 months ago
tool_chat_template_mistral_parallel.jinja	[Bugfix] example template should not add parallel_to...	10 months ago
tool_chat_template_phi4_minijna	[Frontend] Add Phi-4 mini function calling support ...	4 months ago
tool_chat_template_toolace.jinja	[Frontend] Fix typo in tool chat templates for llama3...	3 months ago
tool_chat_template_xlam_llama.jinja	Add XLAM tool parser support (#17148)	last month
tool_chat_template_xlam_qwen.jinja	Add XLAM tool parser support (#17148)	last month

[34] [https://docs.vllm.ai/en/v0.9.2/serving/openai\\_compatible\\_server.html#chat-template\\_1](https://docs.vllm.ai/en/v0.9.2/serving/openai_compatible_server.html#chat-template_1)

[35] <https://github.com/vllm-project/vllm/tree/main/examples>

PART 3

---

# Architecture

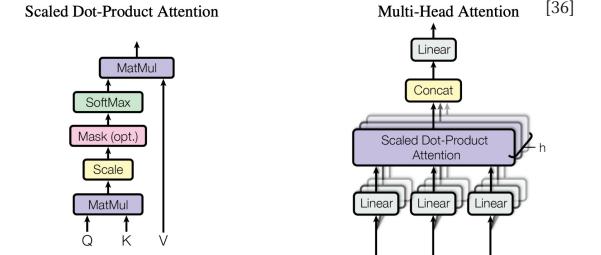
---

# KV Cache, PagedAttention<sup>[7]</sup>

3. Architecture

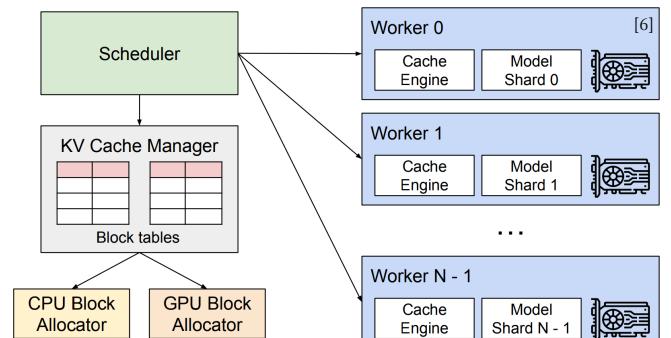
## □ Attention<sup>[36]</sup>

- $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$   
→  $d_k$ : Dimension of  $Q, K$



## □ KV Cache<sup>[37]</sup>

- Autoregressive 생성 방식 특성 상 각 token을 예측할 때 과거 전체 sequence를 입력으로 재처리  
→ time complexity:  $O(n^2)$
- 과거 Key/Value 값 (KV cache)을 사용하여 매 단계의 반복 연산 생략  
→ time complexity:  $O(n)$  수준



## □ PagedAttention<sup>[7, 38, 39]</sup>

- 운영체제 (OS)의 virtual memory에서 영감을 받아 제안
- 기존의 KV cache는 memory 연속성 요구로 memory fragmentation 문제 존재  
→ 특히 많은 요청을 병렬로 처리하는 상황에서 memory 할당/해제의 비효율 심각
- Block 단위의 memory 할당 및 page table을 이용하여 논리적 연속성 (logical continuity) 유지로 물리적 memory 분산
- 각 요청을 고정된 크기의 page에 mapping하여 실제 memory는 non-continuous하게 구성하여 효율적 접근

```
%timeit -n 1
# Generate the text
generation_output = model.generate(
    input_ids=input_ids,
    max_new_tokens=100,
    use_cache=True
)
6.66 s ± 2.22 s per loop (mean ± std. dev. of 7 runs, 1 loop each)

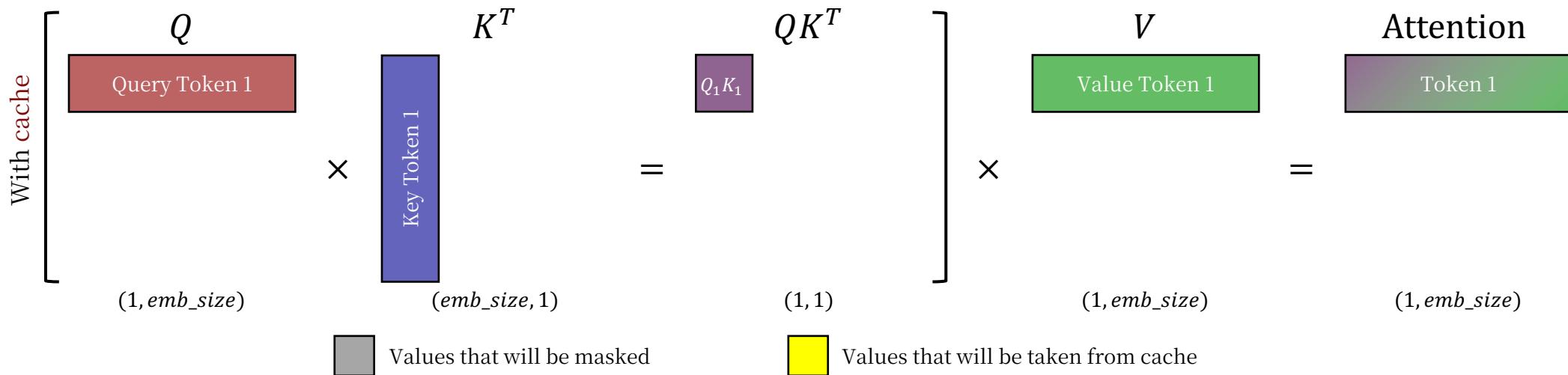
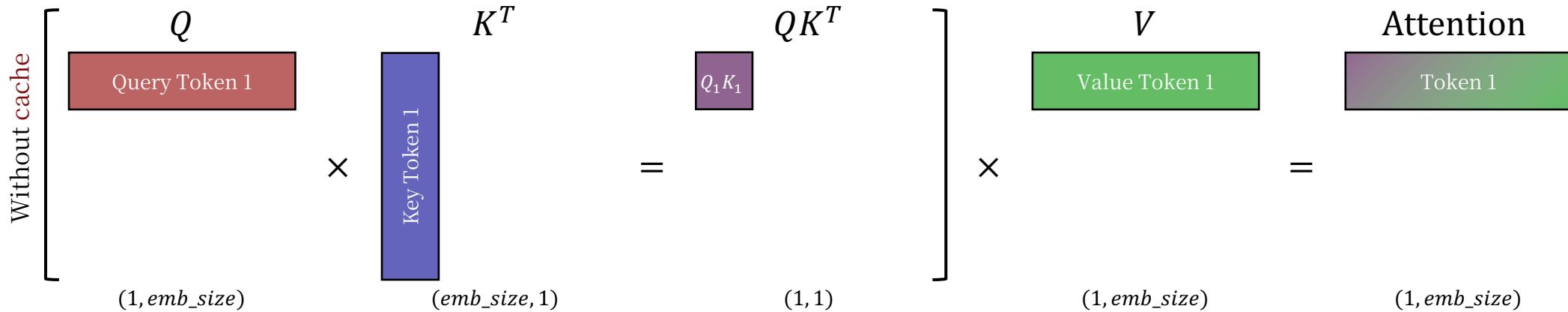
%timeit -n 1
# Generate the text
generation_output = model.generate(
    input_ids=input_ids,
    max_new_tokens=100,
    use_cache=False
)
21.9 s ± 94.6 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

[7] <https://arxiv.org/abs/2309.06180>[36] <https://arxiv.org/abs/1706.03762>[37] <https://huggingface.co/blog/not-lain/kv-caching>[38] [https://docs.vllm.ai/en/v0.9.2/design/kernel/paged\\_attention.html](https://docs.vllm.ai/en/v0.9.2/design/kernel/paged_attention.html)[39] <https://blog.vllm.ai/2023/06/20/vllm.html>

# KV Cache, PagedAttention<sup>[7]</sup>

3. Architecture

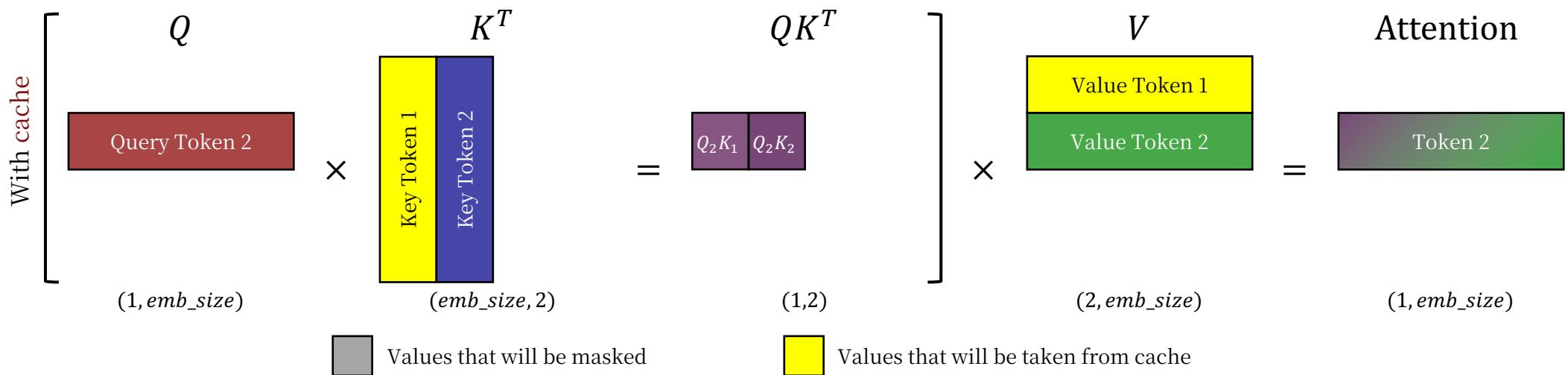
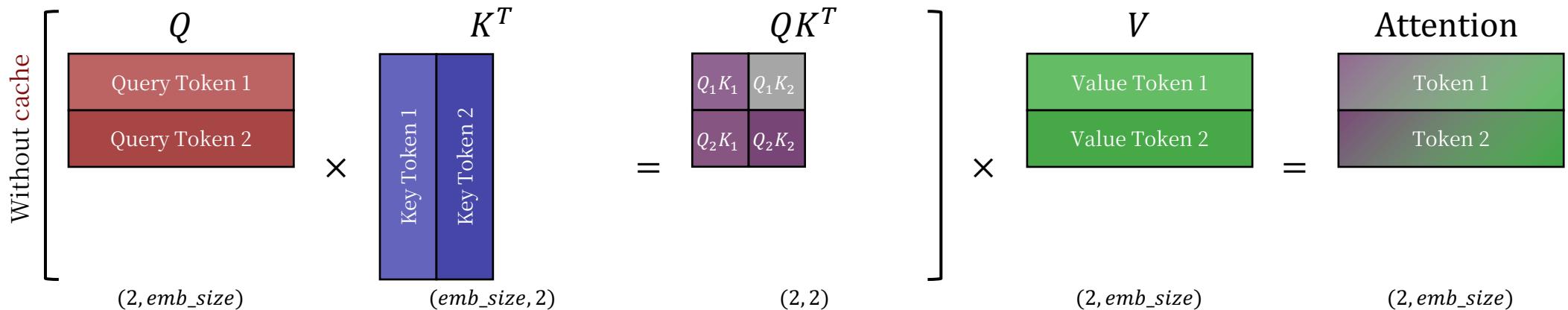
□ Step 1.



# KV Cache, PagedAttention<sup>[7]</sup>

3. Architecture

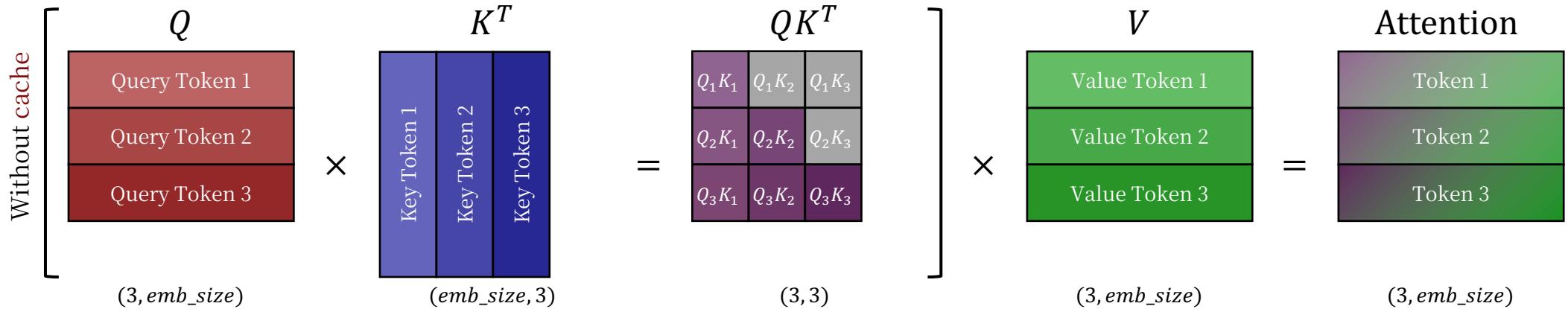
□ Step 2.



# KV Cache, PagedAttention<sup>[7]</sup>

3. Architecture

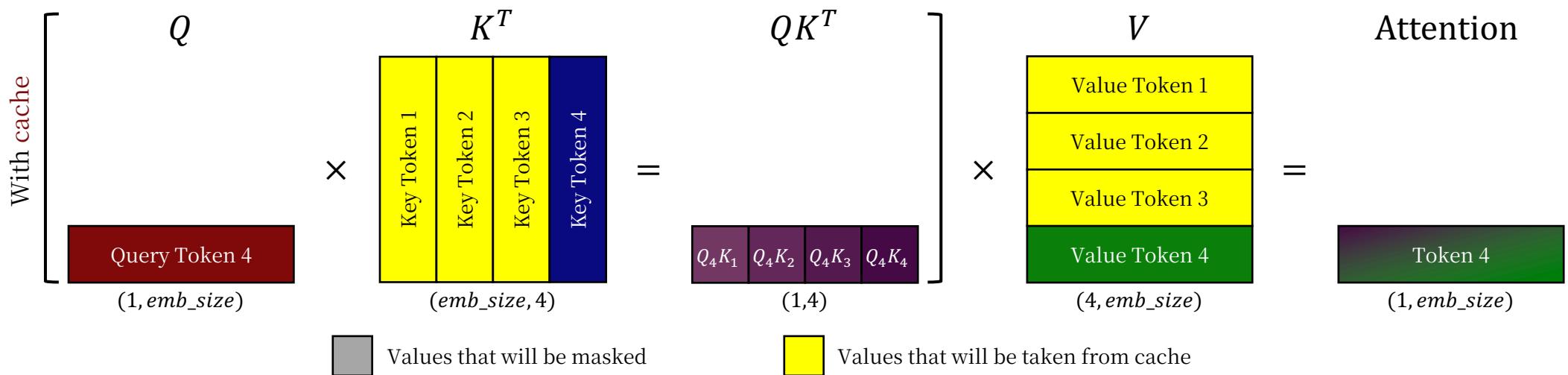
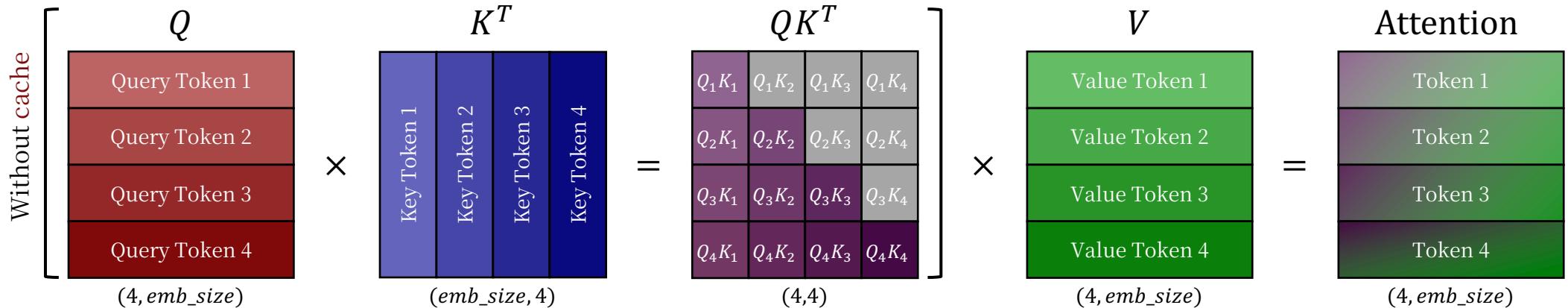
❑ Step 3.



# KV Cache, PagedAttention<sup>[7]</sup>

3. Architecture

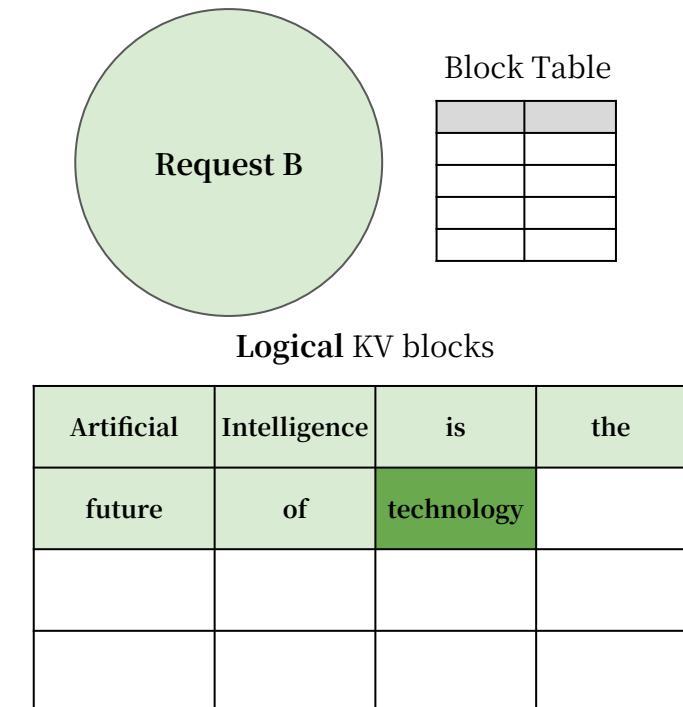
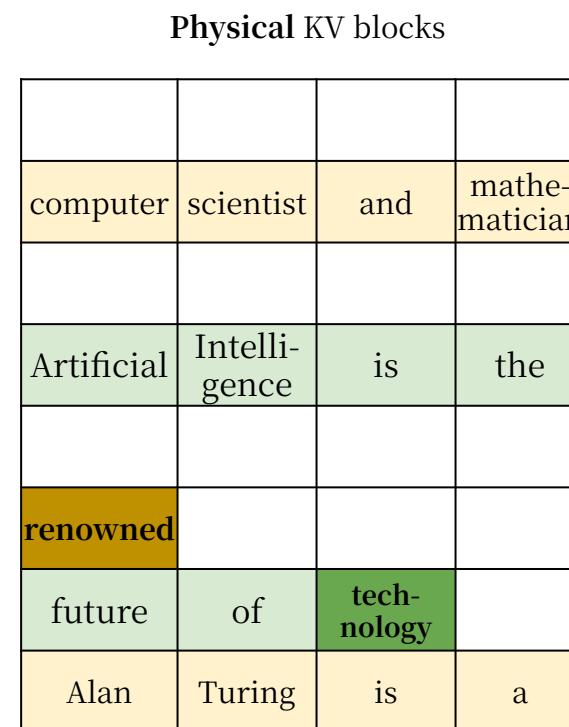
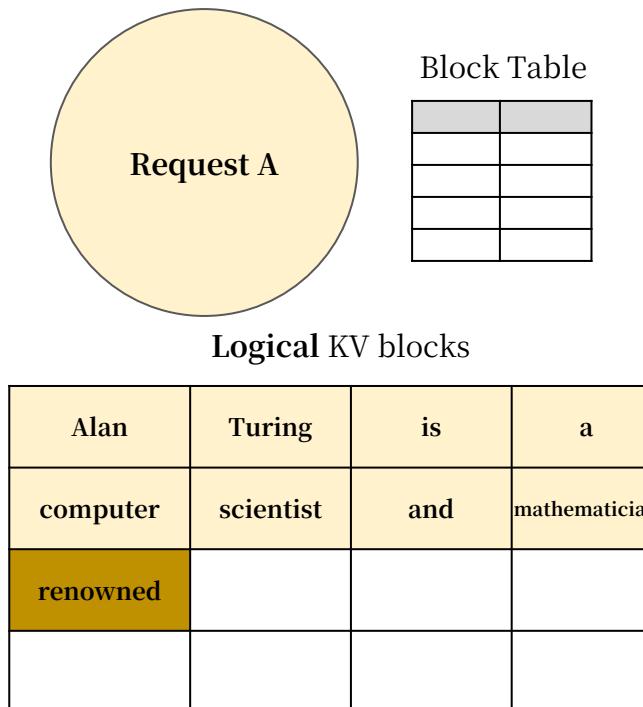
## Step 4.



# KV Cache, PagedAttention<sup>[7]</sup>

3. Architecture

- Logical KV Blocks: 하나의 request 관점의 연속적 memory view
- Physical Blocks: GPU memory 상의 실제 저장 위치 (비연속적, memory fragmentation 방지)
- Block Table: Logical block과 physical block을 연결하는 mapping 정보



# Continuous Batching<sup>[41]</sup>

3. Architecture

## ❑ Batching

- 여러 개의 요청을 하나로 묶어 한 번의 kernel 실행 (큰 행렬곱)으로 동시에 처리하는 기술
- 큰 GEMM (General Matrix Multiplication)으로 연산 집약도 (Arithmetic Intensity) 상승

## ❑ Static Batching

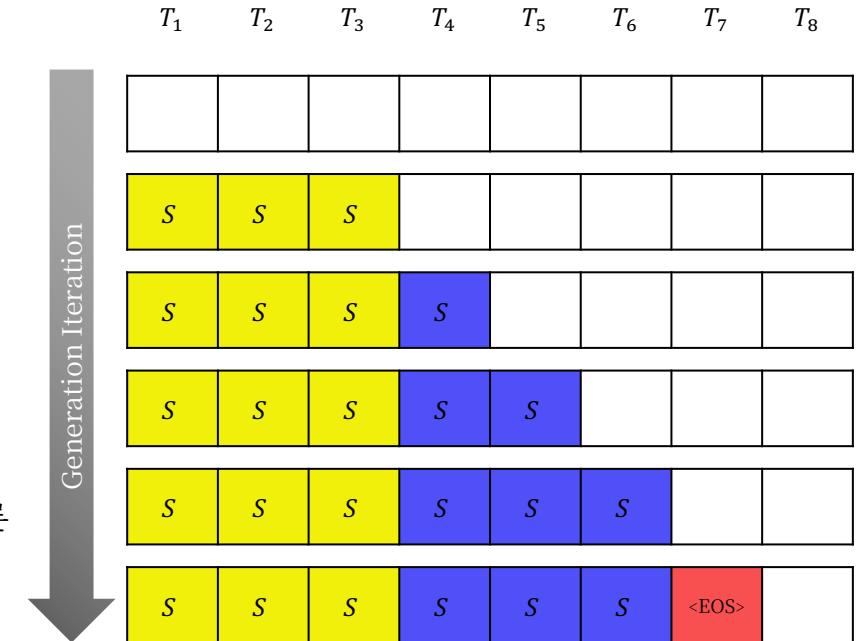
- Fixed batch size의 요청이 모일 때까지 대기 후 추론
- 요청량이 적은 경우 무한정 대기 → GPU 낭비

## ❑ Dynamic Batching<sup>[42]</sup>

- 짧은 수집 window에 대한 요청의 batch 구성 후, 시간 제한 도달 시 batch size 미달이어도 추론
- Batch 내 하나라도 step이 완료되지 않으면 step 종료까지 batch 고정 → GPU 낭비

## ❑ Continuous Batching

- Batch를 요청 단위가 아닌 하나의 token 생성 단계 (iteration) 단위로 운영
- 새 요청은 다음 단계에 즉시 합류, 완료된 sequence는 즉시 제거



$T_n$	Token position in the sequence
$S_n$	Prefill phase (Processing input prompt tokens)
$S_n$	Generation phase (Iterative token generation)
$<\text{EOS}>$	End-of-sequence token

# Continuous Batching<sup>[41]</sup>

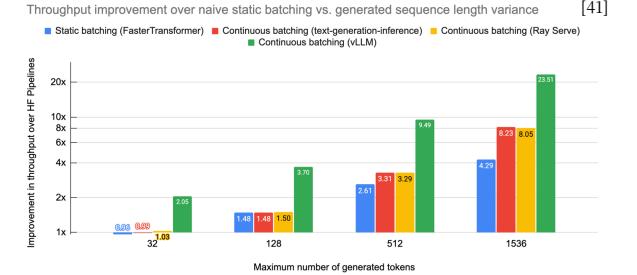
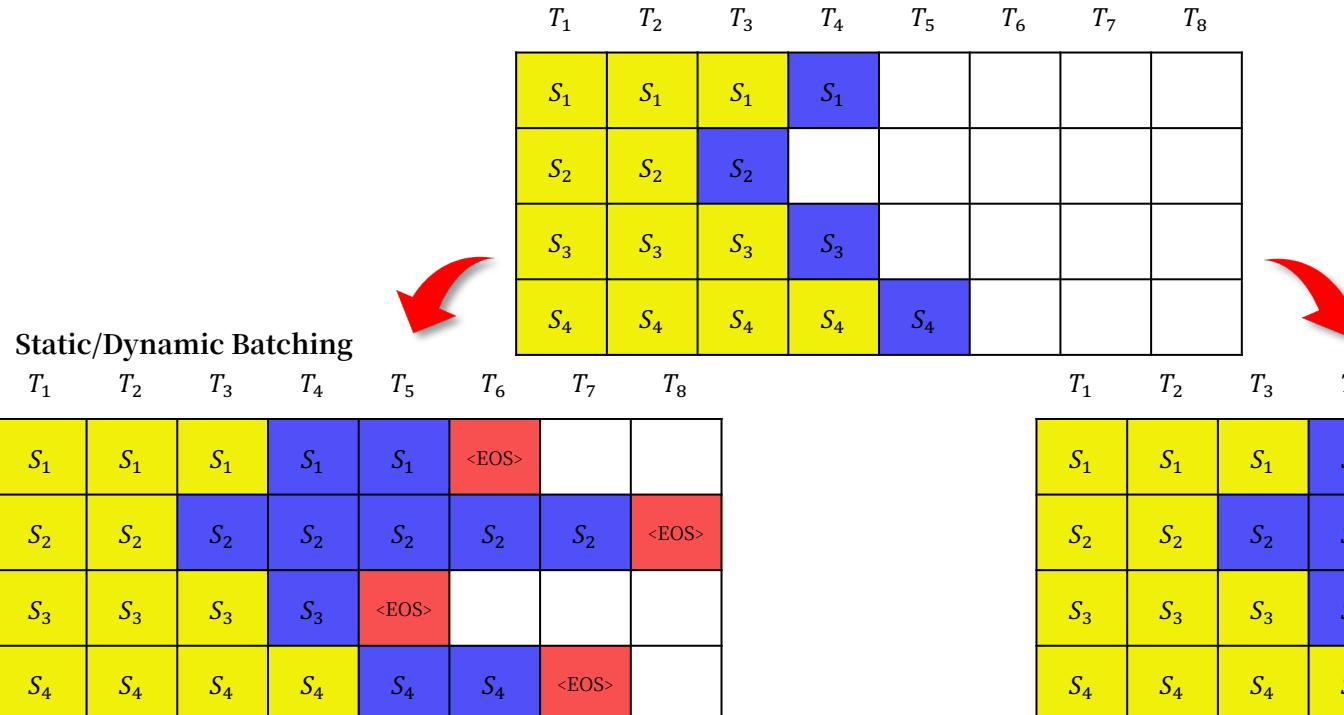
3. Architecture

## □ Limitations of Continuous Batching

- Memory Fragmentation: 각 sequence의 KV cache가 연속적 memory 공간 요구
- Memory Pre-allocation Overhead: 각 요청에 대해 최대 sequence 길이를 기준으로 memory 사전 할당

## □ vLLM + PagedAttention

- Block-wise Storage: KV cache를 물리적으로 비연속적인 block 단위로 저장
- Dynamic Allocation: 필요한 만큼의 page만 동적 할당하여 불필요한 memory overhead 방지
- Naive static batching에 비해 2배 이상 성능 차이



$T_n$	Token position in the sequence
$S_n$	Prefill phase (Processing input prompt tokens in parallel)
$S_n$	Generation phase (Iterative token generation)
<EOS>	End-of-sequence token

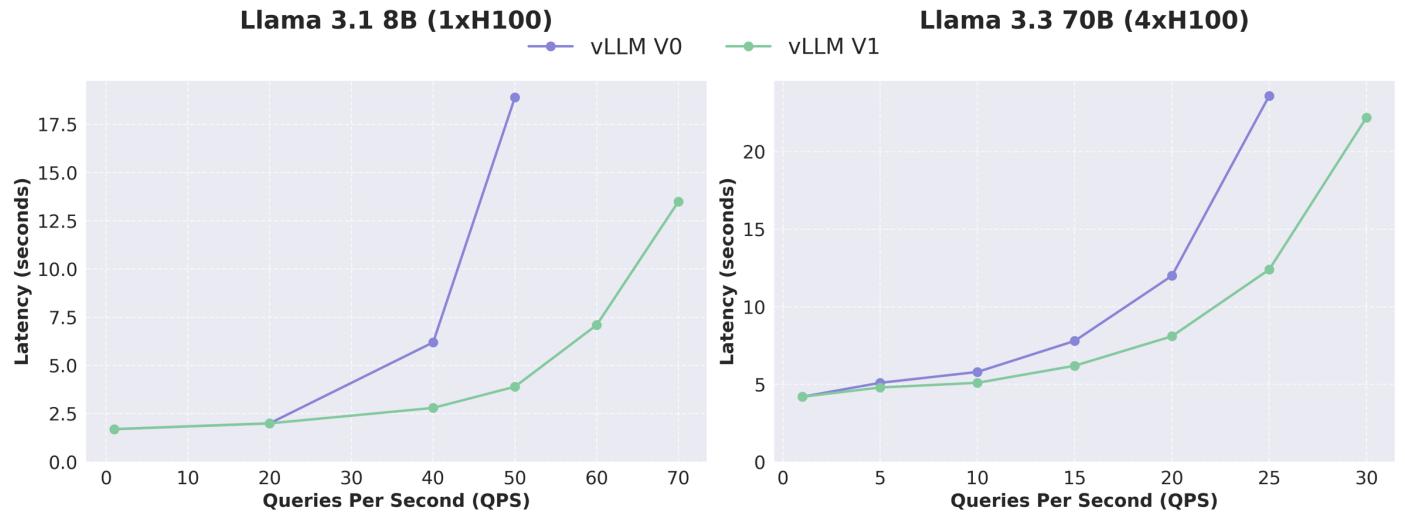
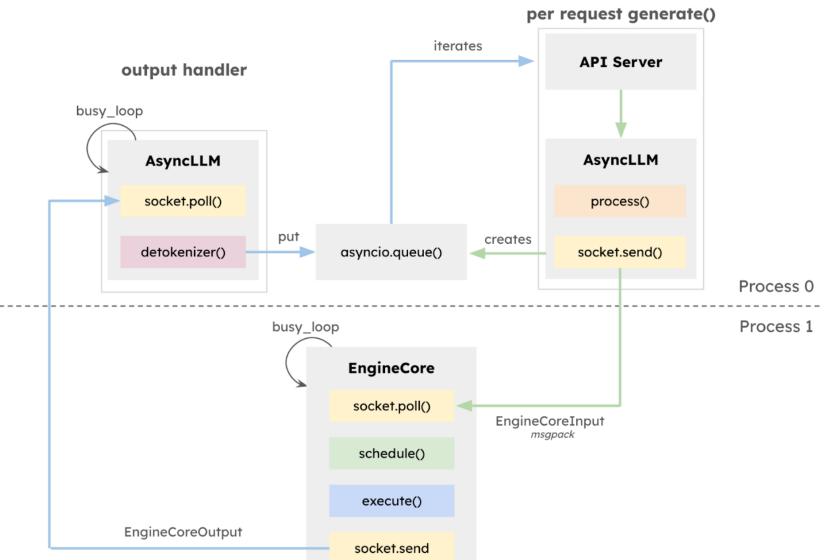
# V0 Engine<sup>[43]</sup> vs. V1 Engine<sup>[44, 45]</sup>

3. Architecture

- ❑ Optimized Execution Loop & API Server: EngineCore와 AsyncLLM 분리로 API server, token화 등 CPU 작업과 GPU model 실행을 완전 비동기 병렬화
- ❑ Simple & Flexible Scheduler: “prefill”, “decode” 구분 제거, “[request\_id: num\_tokens]” 기반 동적 token 할당으로 chunked-prefill, 사전 caching, 추측 decoding 지원
- ❑ Zero-Overhead Prefix Caching: Hash+LRU cache 구조 최적화로 cache hit rate 0%여도 1% 미만 성능 저하
- ❑ Clean Architecture for Tensor-Parallel Inference: Worker 상태 caching 후 diff만 전송, scheduler, worker 분리로 IPC overhead 대폭 감소
- ❑ Efficient Input Preparation: Persistent batch 기법으로 입력 tensor 재생성 없이 diff만 적용, Numpy 활용으로 CPU overhead 최소화
- ❑ torch.compile & Piecewise CUDA Graphs: Model 최적화 자동화 및 유연한 CUDA graph 통합으로 kernel customizing 최소화
- ❑ Enhanced Support for Multimodal LLMs: 비차단 image 처리, image hash 기반 KV cache, encoder cache를 활용한 chunked-prefill 구현
- ❑ FlashAttention 3: 동적 batch 환경에서 최적화된 고성능 attention kernel 제공
- ❑ 환경 변수 `VLLM_USE_V1=1`를 통해 V1 Engine 활성화

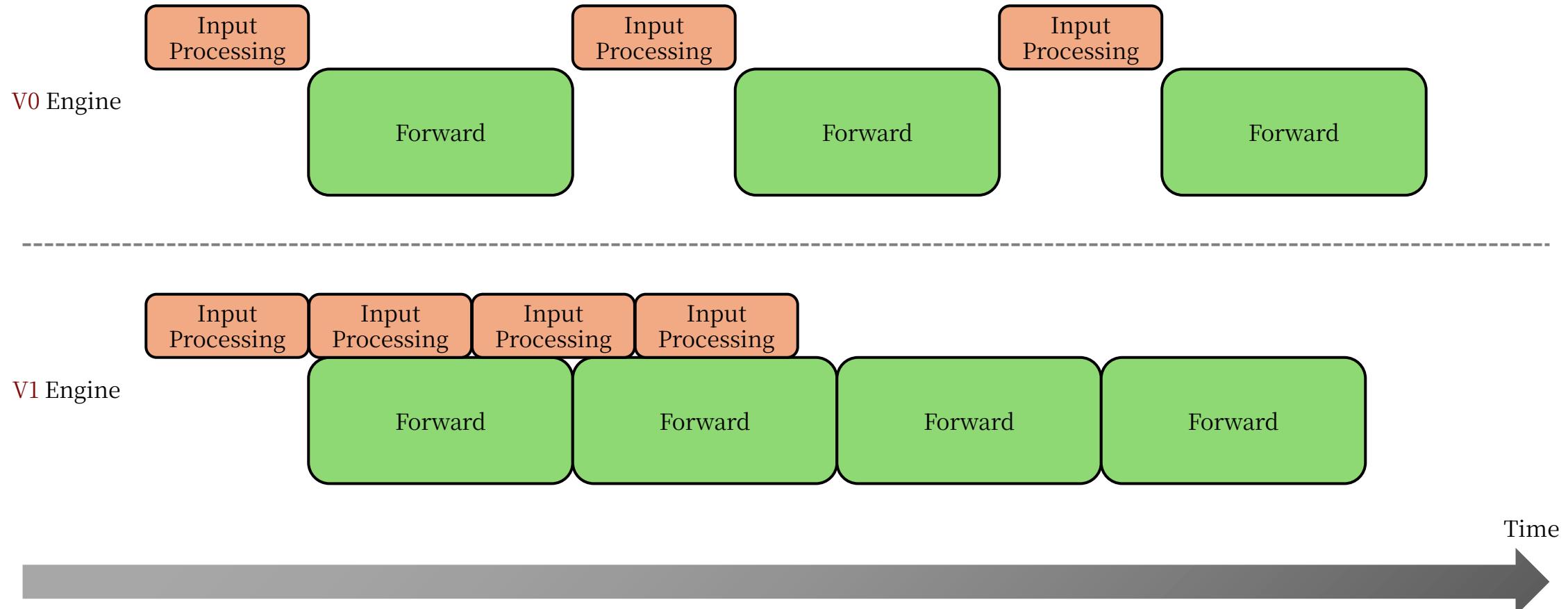
**Versioning Scheme Explained** [46]

- v0.9: Last supported version of the V0 engine, with frozen features and continued testing.
- v0.10: Version marking the start of V0 code removal.
- v0.11: First version without any V0 components.

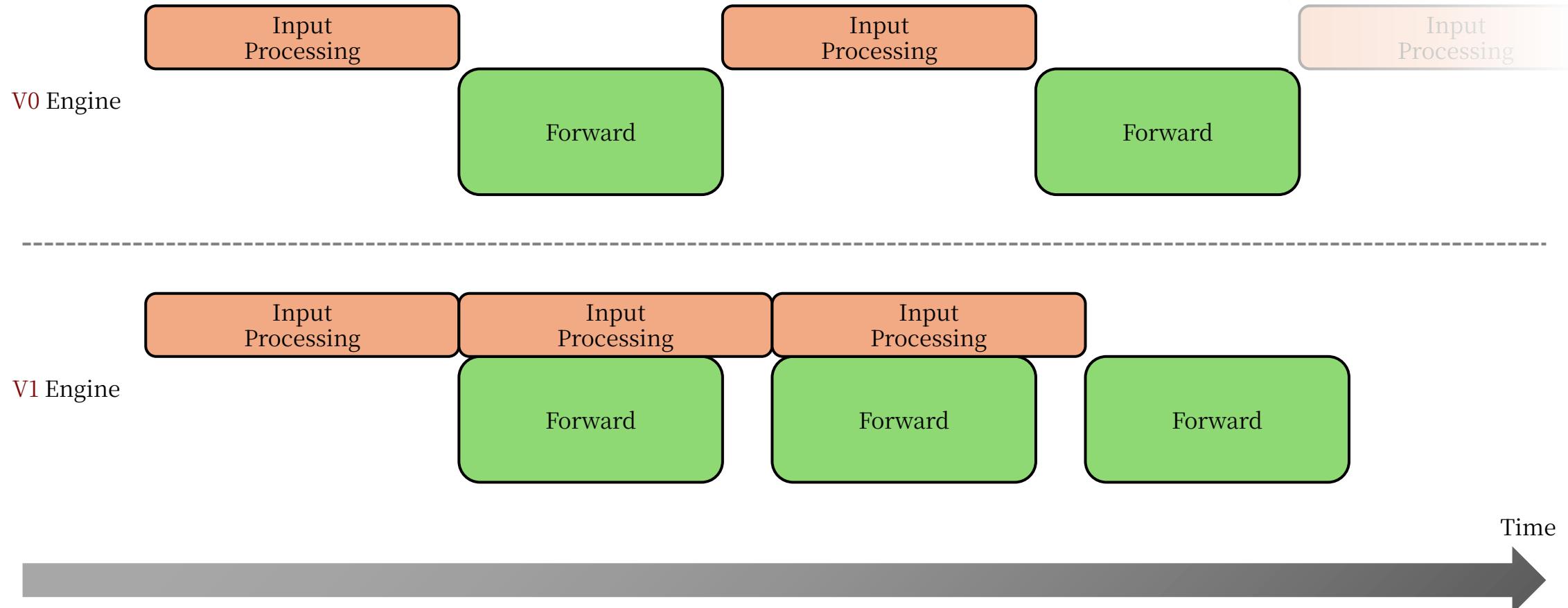
[43] [https://docs.vllm.ai/en/v0.9.2/design/arch\\_overview.html](https://docs.vllm.ai/en/v0.9.2/design/arch_overview.html)[44] [https://docs.vllm.ai/en/v0.9.2/usage/v1\\_guide.html](https://docs.vllm.ai/en/v0.9.2/usage/v1_guide.html)[45] <https://blog.vllm.ai/2025/01/27/v1-alpha-release.html>[46] <https://github.com/vllm-project/vllm/issues/18571>

# V0 Engine<sup>[43]</sup> vs. V1 Engine<sup>[44, 45]</sup>

3. Architecture

[43] [https://docs.vllm.ai/en/v0.9.2/design/arch\\_overview.html](https://docs.vllm.ai/en/v0.9.2/design/arch_overview.html)[44] [https://docs.vllm.ai/en/v0.9.2/usage/v1\\_guide.html](https://docs.vllm.ai/en/v0.9.2/usage/v1_guide.html)[45] <https://blog.vllm.ai/2025/01/27/v1-alpha-release.html>

# V0 Engine<sup>[43]</sup> vs. V1 Engine<sup>[44, 45]</sup>



[43] [https://docs.vllm.ai/en/v0.9.2/design/arch\\_overview.html](https://docs.vllm.ai/en/v0.9.2/design/arch_overview.html)

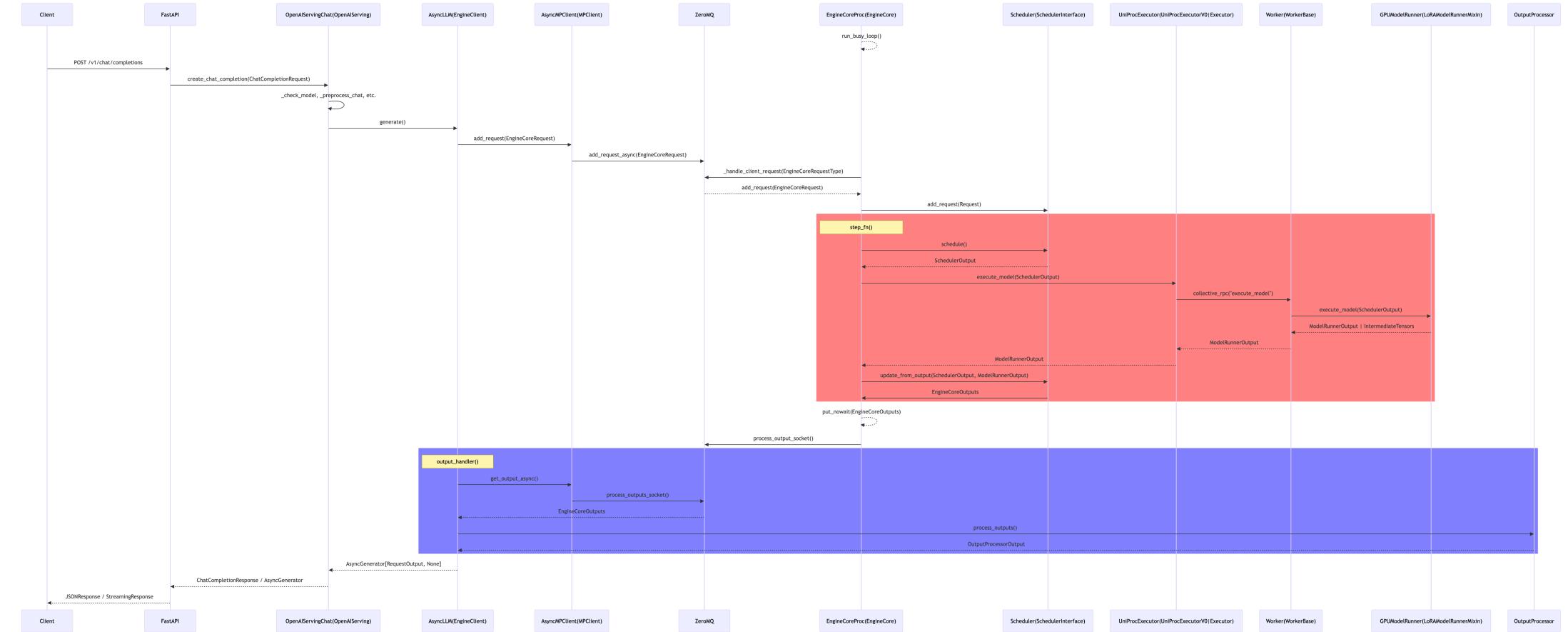
[44] [https://docs.vllm.ai/en/v0.9.2/usage/v1\\_guide.html](https://docs.vllm.ai/en/v0.9.2/usage/v1_guide.html)

[45] <https://blog.vllm.ai/2025/01/27/v1-alpha-release.html>

# Chat Completions

3. Architecture

- ❑ vLLM의 /v1/chat/completions 처리 과정 [47]



PART 4

---

# Production Deployment

---

# LoRA Adapters

## ❑ LoRA (Low-Rank Adaptation) [48]

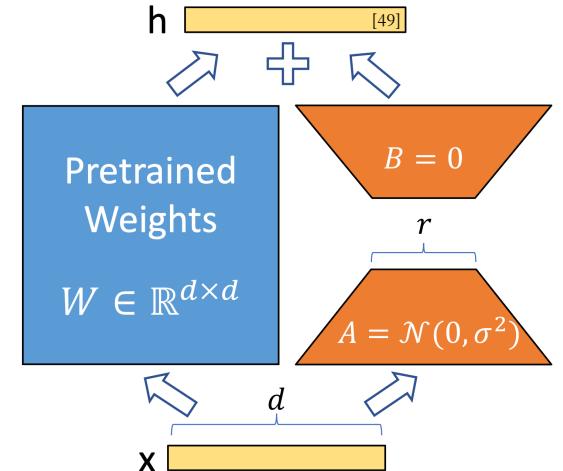
- 기존 model의 weight 행렬에 대해 전체를 학습하지 않고 low-rank matrix ( $A, B$ )만 학습
- $\Delta W = BA$ 를 통해 model이 비용 효율적으로 새로운 data에 적응

## ❑ Static serving LoRA adapters

```
vllm serve Qwen/Qwen3-0.6B --max-model-len 8192 \
    --reasoning-parser qwen3 \
    --enable-lora \
    --lora-modules phh/Qwen3-0.6B-TLDR-Lora=phh/Qwen3-0.6B-TLDR-Lora
```

## ❑ Dynamic serving LoRA adapters

```
VLLM_ALLOW_RUNTIME_LORA_UPDATING=True vllm serve Qwen/Qwen3-0.6B --max-model-len 8192 \
    --reasoning-parser qwen3 \
    --enable-lora
```



```
Fetching 16 files: 100%
WARNING 07-30 22:43:35 [cpu.py:250] Pin memory is not supported on CPU.
INFO 07-30 22:43:35 [serving_models.py:186] Loaded new LoRA adapter: name 'phh/Qwen3-0.6B-TLDR-Lora', path 'phh/Qwen3-0.6B-TLDR-Lora'
INFO: 127.0.0.1:54544 - "POST /v1/load_lora_adapter HTTP/1.1" 200 OK
$ curl -X POST http://localhost:8000/v1/load_lora_adapter \
    -H "Content-Type: application/json" \
    -d '{
        "lora_name": "phh/Qwen3-0.6B-TLDR-Lora",
        "lora_path": "phh/Qwen3-0.6B-TLDR-Lora"
    }'
Success: LoRA adapter 'phh/Qwen3-0.6B-TLDR-Lora' added successfully.
```

```
INFO 07-30 22:44:33 [serving_models.py:203] Removed LoRA adapter: name 'phh/Qwen3-0.6B-TLDR-Lora'
INFO: 127.0.0.1:54550 - "POST /v1/unload_lora_adapter HTTP/1.1" 200 OK
$ curl -X POST http://localhost:8000/v1/unload_lora_adapter \
    -H "Content-Type: application/json" \
    -d '{
        "lora_name": "phh/Qwen3-0.6B-TLDR-Lora"
    }'
Success: LoRA adapter 'phh/Qwen3-0.6B-TLDR-Lora' removed successfully.
```

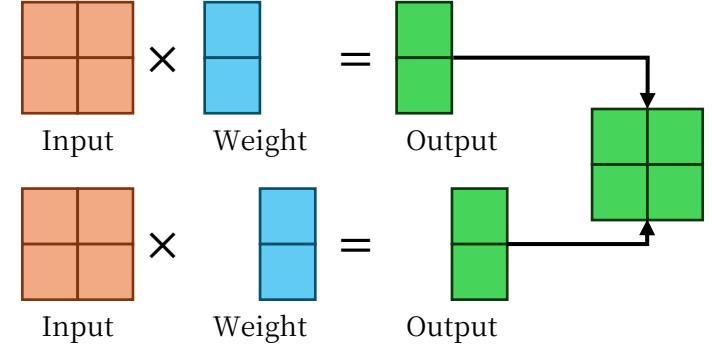
# LoRA Adapters

```
{ "id": "chatcmpl-5f9761a5127b434a950bd63bc7a3f537", "object": "chat.completion", "created": 1753882000, "model": "ephn/Owen3-0.6B-TLDR-Lora", "choices": [ { "index": 0, "message": { "role": "assistant", "reasoning_content": "content": "저작권자: [이름] \n작성자: [이름] \n\n---\n**강현성 (Logan Kang) / Dable & Coinbase (BASE) @Logan** \n**Ultrathinking AI Agent Framework**인 AgentKit을 소개합니다.** \n**블록체인 환경에서 AI Agent를 만들기 위한 빠르고 효율적인 방법은 AgentKit입니다. AgentKit은 다양한 블록체인 기술과 기술을 결합하여, AI Agent를 구축할 수 있도록 합니다. AgentKit은 단순한 기능을 제공하지만, 기술의 혁신과 협업을 통해 빠르게 개발할 수 있도록 설계되었습니다. \n\n---\n**고석현 / Sionic AI @정자초등학교** \n**LLM API 습득** / 하프셋 @cyo** \n**단일 LLM 요청으로 이루어진 기존 AI 플랫폼이 시스템의 한계** \n**astrsk에서 예이전 톡 워크플로로 만든 과정** \n**n이러한 기능은 단순한 LLM을 요청하는 것보다, AI를 활용한 기능을 추가하는 것이 더 효과적입니다. astrsk의 예이전 톡 워크플로는 기존 시스템의 한계를 넘어, 다양한 라이브러리를 활용해 복잡한 문제를 해결하고 있습니다. \n\n---\n**현자윤 / 웨튼테크놀로지스 @jtwunghyun** \n**Chat 인터페이스와 플랫폼은 공존할 수 있을까?** \n**각 서비스를 풀고 있는 전략** \n**AI 서비스를 운영하는 입장에서 틀었던 생각 및 고민** \n**n이러한 질문은 기술의 혁신을 위해 필요합니다. AI 서비스를 직접 운영하는 것이 아니라, 플랫폼을 통해 다양한 기능을 제공하는 것이 더 효과적입니다. \n\n---\n**김우영 / 모니모니 @woong** \n**Agentic Coding, Max Mode: 지지지 않는 일꾼, 극한 활용 가능** \n**Agentic Coding으로 Production까지** \n**VLLM과 비교 및 특장점** \n**내부 구조를 간단히 살펴보겠습니다.** \n**n이러한 설명은 기술의 혁신과 효율성을 강조합니다. \n\n---\n**오효근 / 제노 @zeroherz** \n**온프로토콜 LLM으로 Production까지** \n**VLLM과 비교 및 특장점** \n**내부 구조를 간단히 살펴보겠습니다.** \n**n이러한 질문은 기술의 혁신과 효율성을 강조합니다. \n\n---\n**홍승우 (Martin Hong) / Baryon Labs @martin** \n**백그라운드 예이전 톡 개발자와 로봇의 전문가** \n**Domain Border를 지정하는 것 같아요** \n**n이러한 설명은 기술의 혁신을 위한 프레임워크를 설명합니다. \n\n---\n**Robin Hwang / 카카오** \n**AI Agent 시대, 코딩을 넘어 AI 오케스트레이션으로 진화** \n**실제 사례들을 통해 복잡한 문제를 해결하고, 창의적인 서비스를 가능하게 하는지** \n**n이러한 질문은 AI의 역할을 이해하고, 기술의 혁신을 위한 실례를 제공합니다. \n\n---\n**Maywell / Instruct, KR & 벙커키즈 @jeonghwan Park** \n**TBD***" } ] }
```

# Parallelism Strategies<sup>[50, 51]</sup>

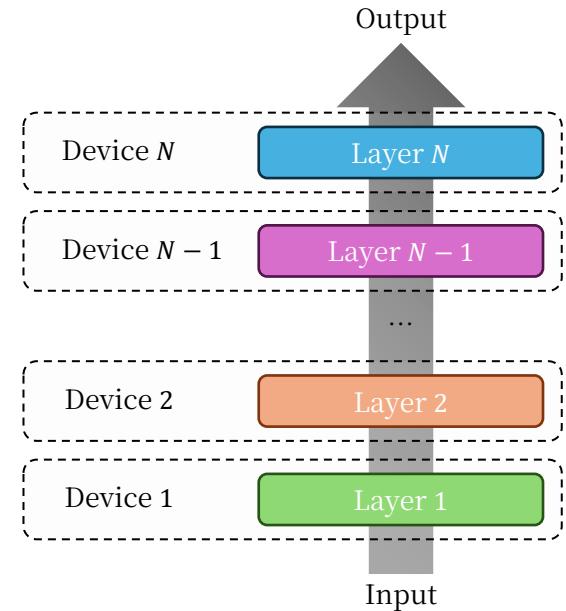
## Tensor Parallelism (TP)

- 각 model layer 내 model parameter를 여러 GPU에 분산하여 처리
  - Model이 너무 커서 여러 GPU를 single node로 추론할 때
  - 더 높은 처리량을 위해 더 많은 KV cache 공간을 마련하기 위해 GPU 당 memory pressure를 줄여야 할 때



## Pipeline Parallelism (PP)

- Model layer를 여러 GPU에 분산하여 model의 여러 부분 순차적 처리
  - Model이 너무 커서 여러 node에 걸쳐 분산해야 할 때
  - Layer 분산이 tensor sharding보다 더 효율적인 매우 깊고 좁은 model일 때



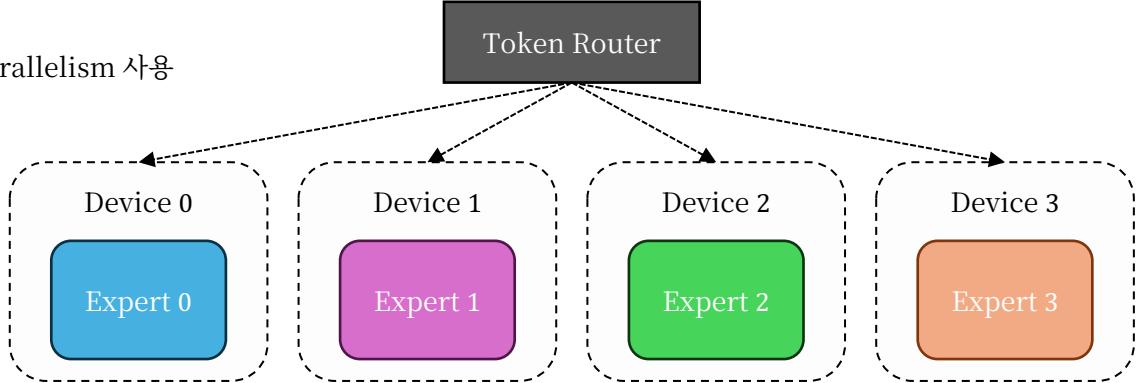
[50] <https://docs.vllm.ai/en/v0.9.2/configuration/optimization.html#parallelism-strategies>

[51] <https://www.youtube.com/watch?v=4i76hmmnIEo>

# Parallelism Strategies<sup>[50, 51]</sup>

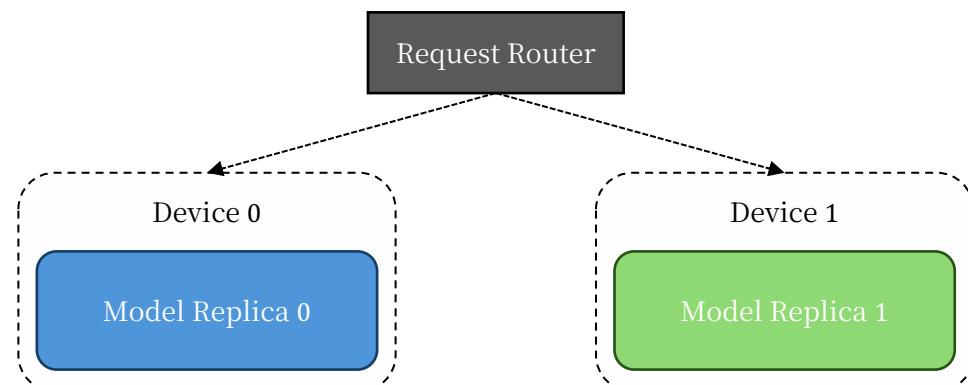
## □ Expert Parallelism (EP)

- Mixture of Experts (MoE) model을 위한 특수한 형태의 병렬 처리
  - “--enable-expert-parallel” 사용 시 MoE layer에서 tensor parallelism 대신 expert parallelism 사용
  - MoE model을 사용할 때
  - GPU 간 expert 연산 부하를 분산할 때



## □ Data Parallelism (DP)

- 여러 GPU에 걸쳐 전체 model을 복제하고 여러 요청 batch를 병렬 처리
  - 전체 model을 복제하기에 충분한 GPU를 보유한 때
  - Model 크기보다 처리량을 확장해야 할 때
  - 요청 batch 간 격리가 유리한 다중 사용자 환경일 때



# Multi-node Distributed Inference

## ■ Ray를 통한 cluster 구성 과정 [52]

- VLLM\_HOST\_IP [53]: vLLM 내부 통신에 사용할 node의 IP 주소
- GLOO\_SOCKET\_IFNAME [54]: PyTorch Gloo backend가 사용할 network interface 이름
- NCCL\_IB\_DISABLE [55]: NCCL의 InfiniBand (IB) network 사용 여부

```
# NOTE: master
worker run \
    -d \
    --entrypoint /bin/bash \
    --image vllm:v1.0 \
    --shm-size 10.24g \
    -v /dev/shm:/dev/shm \
    -v "/root/.cache/huggingface" \
    -e VLLM_HOST_IP=$NODE_IP \
    -e GLOO_SOCKET_IFNAME=$GLOO_SOCKET_IFNAME \
    -e NCCL_IB_DISABLE=1 \
    vlm/vlm-openai:v0.9.2 \
    -c "uv_ip install ray/default" system && ray start --head --port=6379 --disable-usage-stats --dashboard-host=0.0.0.0 uv -f /dev/null

# ray start --head --port=6379 --disable-usage-stats --dashboard-host=0.0.0.0
Usage stats collection is disabled.

Local node IP:

Ray runtime started.

Next steps
To add another node to this Ray cluster, run
ray start --address='*:6379'

To connect to this Ray cluster:
import ray
ray.init()

To submit a Ray job using the Ray Jobs CLI:
RAY_ADDRESS='http://*:8265' ray job submit --working-dir . -- python my_script.py

See https://docs.ray.io/en/latest/cluster/running-applications/job-submission/index.html
for more information on submitting Ray jobs to the Ray cluster.

To terminate the Ray runtime, run
ray stop

To view the status of the cluster, use
ray status

To monitor and debug Ray, view the dashboard at
:8265

If connection to the dashboard fails, check your firewall settings and network configuration.

# ray start --block --address=:6379
Local node IP: 192.168.75.174
[2025-07-29 05:37:09,112 W 646 646] global_state_accessor.cc:435: Retrying to get node with node ID 6395285e5a2a36e5773e77fd095490d63eda1f33869da1d09291b522

Ray runtime started.

To terminate the Ray runtime, run
ray stop
--block
This command will now block forever until terminated by a signal.
Running subprocesses are monitored and a message will be printed if any of them terminate unexpectedly. Subprocesses exit with SIGTERM will be treated as graceful, thus NOT reported.
```

Head Node

NODE_ID	NODE_IP	IS_HEAD_NODE	STATE
05ala18c99ab485e6af30b864bd1227c09815f71f40eac26ddbc3f9a	192.168.75.174	True	ALIVE
6395285e5a2a36e5773e77fd095490d63eda1f33869da1d09291b522	192.168.75.174	False	ALIVE

Worker Node

[52] [https://docs.vllm.ai/en/v0.9.2/examples/online\\_serving/run\\_cluster.html](https://docs.vllm.ai/en/v0.9.2/examples/online_serving/run_cluster.html)[53] <https://docs.vllm.ai/en/v0.9.2/usage/security.html>[54] <https://docs.pytorch.org/docs/stable/distributed.html#common-environment-variables>[55] [https://docs.nvidia.com/deeplearning/ncc/user-guide/docs/env.html#nccl\\_ib-disable](https://docs.nvidia.com/deeplearning/ncc/user-guide/docs/env.html#nccl_ib-disable)

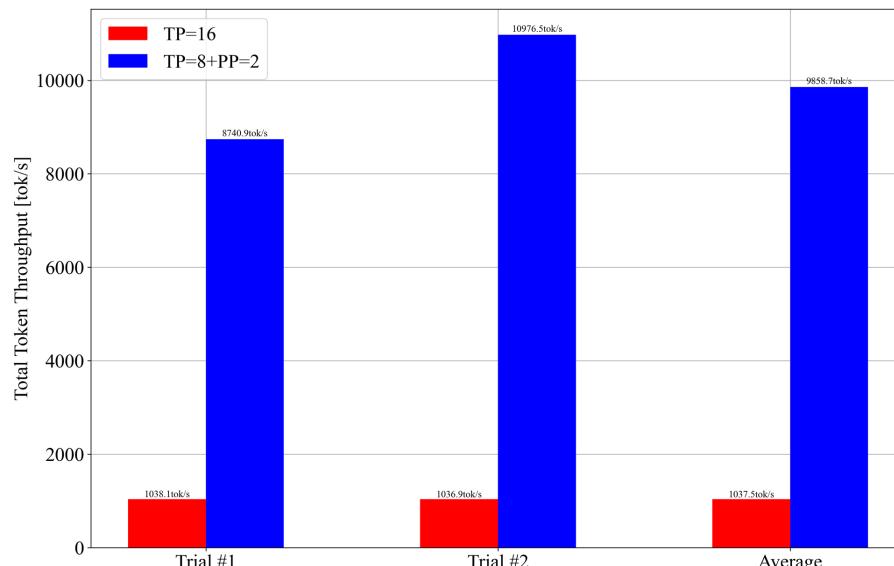
## Multi-node Distributed Inference

## □ Multi-Node Multi-GPU [56, 57]

(tensor parallel plus pipeline parallel inference)

- If your model is too large to fit in a single node, you can use tensor parallel together with pipeline parallelism.
  - The tensor parallel size is the number of GPUs you want to use in each node, and the pipeline parallel size is the number of nodes you want to use.
  - E.g., if you have 16 GPUs in 2 nodes (8 GPUs per node), you can set **the tensor parallel size to 8** and **the pipeline parallel size to 2**.

### TP 16 vs. TP 8 + PP 2



Host / Worker Process name	State	State Message	ID	IP / PID	Actions	CPU ⓘ	Memory ⓘ	GPU ⓘ	GRAM
>	<span>ALIVE</span>	-	a3b57...	(Head)	<a href="#">Log</a>	<div style="width: 17.7%;">17.7%</div>	<div style="width: 30.53%;">30.53GB/1007.51GB(3.0%)</div>	<div style="display: flex; justify-content: space-around;"><span>[0]: 100.0%</span><span>[1]: 100.0%</span><span>[2]: 100.0%</span><span>[3]: 100.0%</span><span>[4]: 100.0%</span><span>[5]: 100.0%</span><span>[6]: 100.0%</span><span>[7]: 100.0%</span></div>	<span>80813MB/81559MB</span> <span>81021MB/81559MB</span> <span>81019MB/81559MB</span> <span>81019MB/81559MB</span> <span>81019MB/81559MB</span> <span>81019MB/81559MB</span> <span>80987MB/81559MB</span> <span>81019MB/81559MB</span> <span>80967MB/81559MB</span>
>	<span>ALIVE</span>	-	4a5627...		<a href="#">Log</a>	<div style="width: 10.7%;">10.7%</div>	<div style="width: 21.21%;">21.21GB/1007.51GB(2.1%)</div>	<div style="display: flex; justify-content: space-around;"><span>[0]: 100.0%</span><span>[1]: 100.0%</span><span>[2]: 100.0%</span><span>[3]: 100.0%</span><span>[4]: 100.0%</span><span>[5]: 100.0%</span><span>[6]: 100.0%</span><span>[7]: 100.0%</span></div>	<span>80963MB/81559MB</span> <span>80525MB/81559MB</span> <span>80523MB/81559MB</span> <span>80523MB/81559MB</span> <span>80523MB/81559MB</span> <span>80491MB/81559MB</span> <span>80523MB/81559MB</span> <span>80513MB/81559MB</span>

# Multi-node Distributed Inference

## ❑ RDMA (Remote Direct Memory Access)<sup>[58]</sup>

- Network에 연결된 두 node의 memory 간 CPU, cache, 운영 체제의 개입 없이 data를 직접 전송하는 기술
- E.g., InfiniBand, RoCE, iWARP

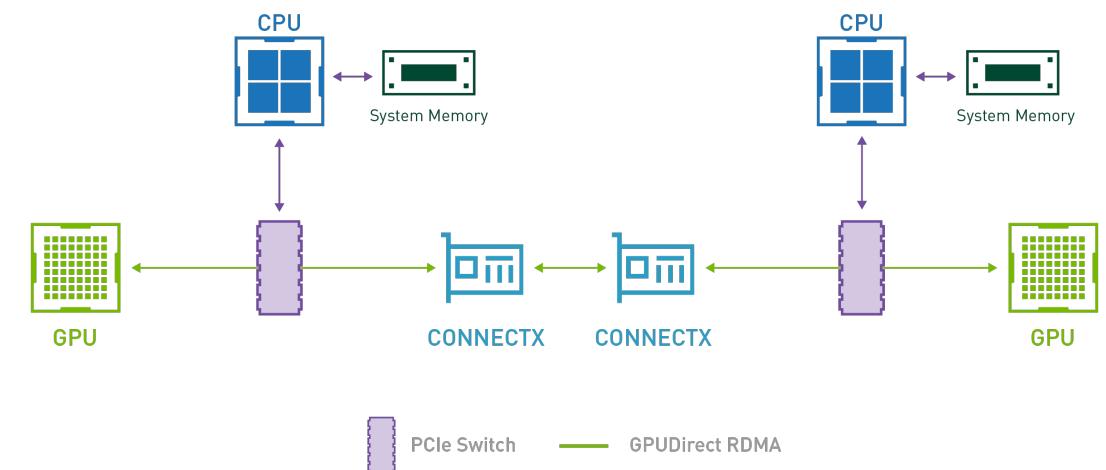
## ❑ InfiniBand<sup>[59]</sup>

- RDMA를 native로 지원하여 CPU 개입 없이 node 간 memory 직접 전송 가능
- 전용 switch와 network adapter (Host Channel Adapter, HCA) 사용 (Ethernet network와는 다른 독자적 architecture)

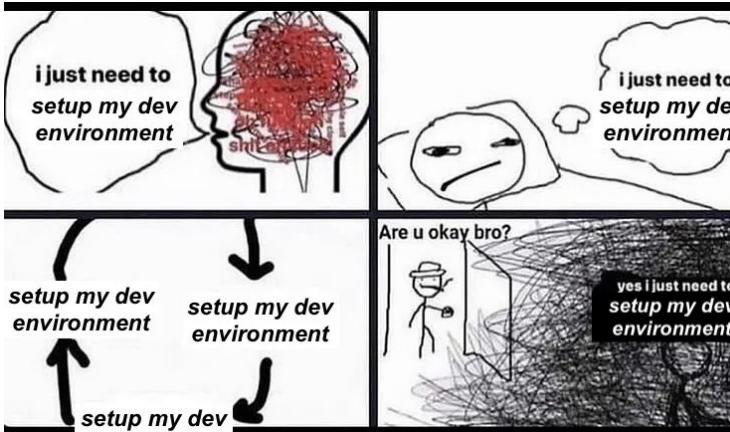
## ❑ RoCEv2 (RDMA over Converged Ethernet v2)<sup>[58]</sup>

- 표준 ethernet network를 통해 RDMA를 구현하는 protocol

항목	InfiniBand	RoCE
Network type	전용 InfiniBand fabric	Ethernet 기반
구축 비용	별도 장비 필요, 비용 높음	상대적으로 저렴, 기존 infra 활용 가능
성능	최고 성능, ultra-low latency	Lossless Ethernet 환경에서 InfiniBand에 근접
호환성	HPC/AI 특화, 범용성 낮음	범용성 높음, 기존 infra와 통합 용이
관리 복잡도	전용 환경, 관리 일관성	Ethernet tuning 필요, 설정 복잡할 수 있음



# Multi-node Distributed Inference



# Production Deployment with GPU Cluster (Kubernetes)

## ❑ Kubernetes 배포 [61]

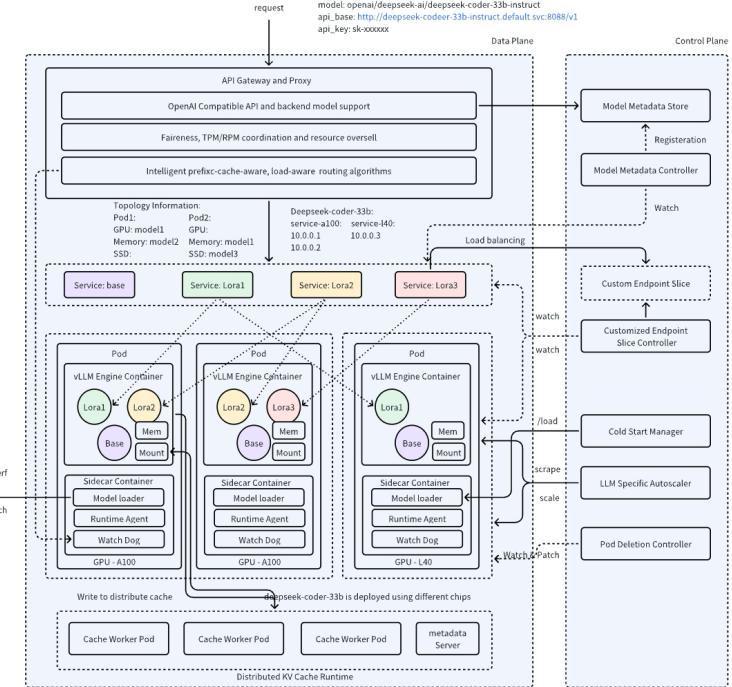
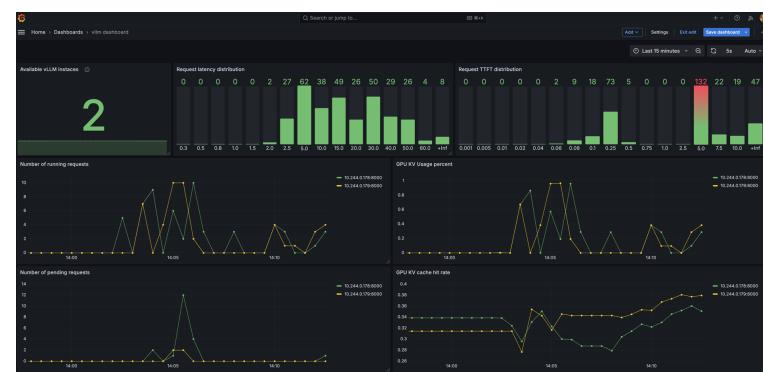
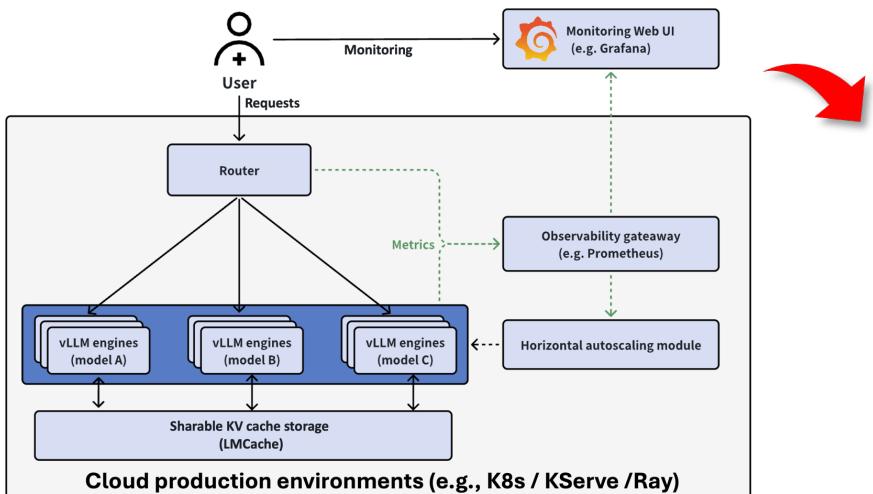
- Model store, Network, IPC, 환경 변수 등의 자유도가 많은 만큼 관리가 어려움

## ❑ AIBrix [62, 63, 64, 65]

- vLLM을 위한 확장 가능하고 비용 효율적인 cloud native control plane
- 고밀도 LoRA 관리, LLM gateway 및 routing, autoscaler, 분산 추론 및 분산 KV cache, ...

## ❑ Production-stack [66, 67, 68, 69]

- vLLM을 통한 LLM 배포 시 production에 최적화된 codebase
- LMCache 기반 KV cache 공유 및 저장, prefix-aware routing, Helm chart를 통한 배포, ...



[61] <https://docs.vllm.ai/en/v0.9.2/deployment/k8s.html>

[62] <https://github.com/vllm-project/aibrix>

[63] <https://arxiv.org/abs/2504/03648>

[64] <https://blog.vllm.ai/2025/02/21/aibrix-release.html>

[65] <https://aibrix.readthedocs.io/latest/>

[66] <https://github.com/vllm-project/production-stack>

[68] <https://blog.vllm.ai/2025/01/21/stack-release.html>

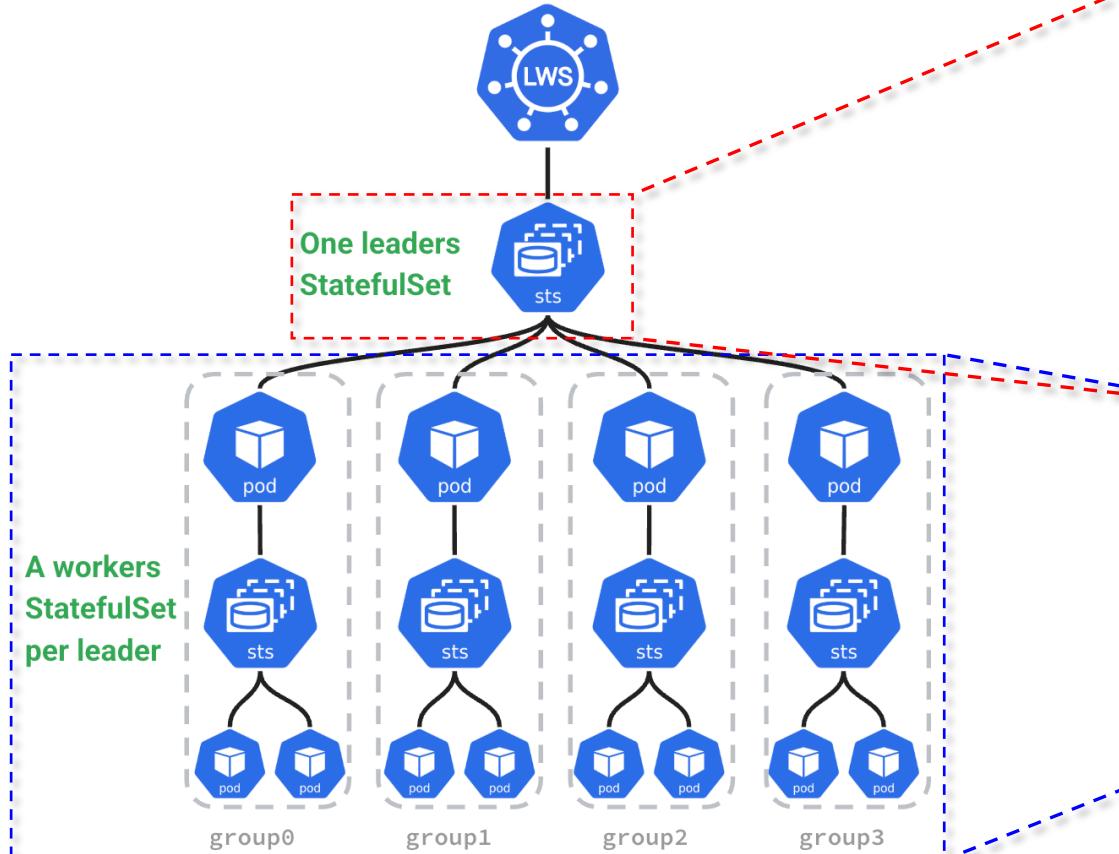
[69] <https://blog.vllm.ai/production-stack/>

# Production Deployment with GPU Cluster (Kubernetes)

## 4. Production Deployment

## □ LWS (LeaderWorkerSet) [70, 71]

- Kubernetes 상에서 Leader-Worker architecture를 손쉽게 구현하고 관리할 수 있도록 도와주는 CRD 및 controller



```
1  apiVersion: leaderworkerset.sigs.k8s.io/v1
2  kind: LeaderWorkerSet
3  metadata:
4    name: vilm
5    spec:
6      replicas: 2
7      leaderWorkerTemplate:
8        size: 2
9        restartPolicy: RecreateOnPodRestart
10       leaderTemplate:
11         metadata:
12           labels:
13             role: leader
14         spec:
15           containers:
16             - name: vilm-leader
17               image: docker.io/vilm/vilm-openai:latest
18               env:
19                 - name: HUGGING_FACE_HUB_TOKEN
20                   value: <your-hf-token>
21               command:
22                 - sh
23                 - -c
24                 - "bash /vilm-workspace/examples/online_serving/multi-node-serving.sh leader --ray_cluster_size=$(LWS_GROUP_SIZE); python3 -m vilm.entrypoints.openai.api_server --port 8080 --model meta-llama/Llama-3.1-405B-Instruct --tensor-parallel-size 8 --pipeline_parallel_size 2"
25             resources:
26               limits:
27                 nvidia.com/gpu: "8"
28                 memory: 1124Gi
29               ephemeral-storage: 800Gi
30             requests:
31               ephemeral-storage: 800Gi
32                 cpu: 125
33             ports:
34               - containerPort: 8080
35             readinessProbe:
36               tcpSocket:
37                 port: 8080
38               initialDelaySeconds: 15
39               periodSeconds: 10
40             volumeMounts:
41               - mountPath: /dev/shm
42                 name: dshm
43             volumes:
44               - name: dshm
45                 emptyDir:
46                   medium: Memory
47                   sizeLimit: 15Gi
48       workerTemplate:
49         spec:
50           containers:
51             - name: vilm-worker
52               image: docker.io/vilm/vilm-openai:latest
53               command:
54                 - sh
55                 - -c
56                 - "bash /vilm-workspace/examples/online_serving/multi-node-serving.sh worker --ray_address=${LWS_LEADER_ADDRESS}"
57             resources:
58               limits:
59                 nvidia.com/gpu: "8"
60                 memory: 1124Gi
61               ephemeral-storage: 800Gi
62             requests:
63               ephemeral-storage: 800Gi
64                 cpu: 125
65             env:
66               - name: HUGGING_FACE_HUB_TOKEN
67                   value: <your-hf-token>
68             volumeMounts:
69               - mountPath: /dev/shm
70                 name: dshm
71             volumes:
72               - name: dshm
73                 emptyDir:
74                   medium: Memory
75                   sizeLimit: 15Gi
76
77   apiVersion: v1
78   kind: Service
79   metadata:
80     name: vilm-leader
81   spec:
82     ports:
83       - name: http
84         port: 8080
85         protocol: TCP
86         targetPort: 8080
87     selector:
88       leaderworkerset.sigs.k8s.io/name: vilm
89       roles: leader
90     type: ClusterIP
```

# Observability (Prometheus + Grafana)

## "/metrics"를 통해 vLLM server의 Prometheus format metric 수집 가능 [72, 73]

- vllm:request\_success\_total: 요청 완료 수 (EOS 도달 또는 max 길이 도달)
- vllm:request\_queue\_time\_seconds: queue 대기 시간
- vllm:request\_prefill\_time\_seconds: prefill 단계 소요 시간
- vllm:request\_decode\_time\_seconds: decoding 단계 소요 시간
- vllm:request\_max\_num\_generation\_tokens: 생성된 token 최대값
- ...

## Grafana와 연결하여 dashboard 구성 가능

```

zerohertz@zerohertz-mm:~$ curl http://localhost:8000/metrics
# HELP vlm:lora_requests_info Running stats on lora requests.
# TYPE vlm:lora_requests_info gauge
vlmlora_requests_info{max_lora="0",running_lora_adapters=""} 1.753628624009126e+09
# HELP vlmcache_config_info Information of the LLMEngine CacheConfig
# TYPE vlmcache_config_info gauge
vlmcache_config_info{block_size="16",cache_dtypes="auto",calculate_kv_scales="False",cpu_kvcache_space_bytes="4294967296",cpu_offload_gb="0.0",enable_prefix_caching="None",gpu_memory_utilization="0.9",is_attention_free="False",num_cpu_blocks="0",num_gpu_blocks="2348",num_gpu_blocks_overrides="None",prefix_caching_hash_algo="builtin",sliding_window="None",swap_space="4.0",swap_space_bytes="4294967296",version="1.0"}
# HELP vlm:num_premptions_total Cumulative number of preemption from the engine.
# TYPE vlm:num_premptions_total counter
vlmlm_num_premptions_total{model_name="Owen/Owen3-0.6B"} 0.0
# HELP vlm:tokens_per_step_hist Total tokens per step histogram of prefill tokens processed.
# TYPE vlm:tokens_per_step_hist histogram
vlmlm_tokens_per_step_hist{model_name="Owen/Owen3-0.6B"} 14.0
# HELP vlm:generation_tokens_total Number of generation tokens processed.
# TYPE vlm:generation_tokens_total counter
vlmlm_generation_tokens_total{model_name="Owen/Owen3-0.6B"} 107.0
# HELP vlm:request_success_total Count of successfully processed requests.
# TYPE vlm:request_success_total counter
vlmlm_request_success_total{model_name="Owen/Owen3-0.6B"} 1.0
# HELP vlm:iteration_tokens_total Histogram of number of tokens per engine_step.
# TYPE vlm:iteration_tokens_total histogram
vlmlm_iteration_tokens_total_sum{model_name="Owen/Owen3-0.6B"} 121.0
vlmlm_iteration_tokens_total_bucket{le="1.0",model_name="Owen/Owen3-0.6B"} 1099.0
vlmlm_iteration_tokens_total_bucket{le="8.0",model_name="Owen/Owen3-0.6B"} 1099.0
vlmlm_iteration_tokens_total_bucket{le="16.0",model_name="Owen/Owen3-0.6B"} 1100.0
vlmlm_iteration_tokens_total_bucket{le="32.0",model_name="Owen/Owen3-0.6B"} 1100.0
vlmlm_iteration_tokens_total_bucket{le="64.0",model_name="Owen/Owen3-0.6B"} 1100.0
vlmlm_iteration_tokens_total_bucket{le="128.0",model_name="Owen/Owen3-0.6B"} 1100.0
vlmlm_iteration_tokens_total_bucket{le="256.0",model_name="Owen/Owen3-0.6B"} 1100.0
vlmlm_iteration_tokens_total_bucket{le="512.0",model_name="Owen/Owen3-0.6B"} 1100.0
vlmlm_iteration_tokens_total_bucket{le="1024.0",model_name="Owen/Owen3-0.6B"} 1100.0
vlmlm_iteration_tokens_total_bucket{le="2048.0",model_name="Owen/Owen3-0.6B"} 1100.0

```



[72] <https://docs.vllm.ai/en/v0.9.2/usage/metrics.html>

[73] [https://docs.vllm.ai/en/v0.9.2/examples/online\\_serving/prometheus\\_grafana.html](https://docs.vllm.ai/en/v0.9.2/examples/online_serving/prometheus_grafana.html)

# Benchmark

## □ “vllm bench” CLI 명령을 통해 간단한 benchmark 가능 [74]

```
$ vllm bench serve \
  --base-url http://192.168.75.173:8080 \
  --model Owen/Owen3-235B-A22B \
  --request-rate 2000 \
  --num-prompts 10000 \
  --random-input-len 600 \
  --random-input-len 125 \
  --ignore-eos \
  --seed 42 \
  --percentile-metrics "ttft,tpot,itl,e2el" \
  --save-result
INFO 07-29 23:35:50 [__init__.py:244] Automatically detected platform cpu.
Namespaces[parser='bench', bench_type='serve', dispatch_type='function', BenchmarkServingSubCommand.cmd at 0x1458ad620>, seed=42, num_prompts=10000, dataset_name='random', dataset_path=None, custom_output_len=256, custom_skip_chat_template=False, sonnet_input_len=550, sonnet_output_len=150, sonnet_prefix_len=200, sharegpt_output_len=None, random_input_len=600, random_output_len=125, random_range_ratio=0.0, random_prefix_len=0, hf_subset=None, hf_split=None, hf_output_len=None, endpoint_type='openai', label=None, backend='vllm', base_url='http://192.168.75.173:8080', host='127.0.0.1', port=8080, endpoint='/v1/completions', max_concurrency=None, model='Owen/Owen3-235B-A22B', tokenizer=None, use_beam_search=False, logprobs=None, request_rate=2000.0, burstiness=1.0, trust_remote_code=False, disable_tqdm=False, profile=False, save_result=True, save_detailed=False, append_result=False, metadata=None, result_dir=None, result_file_name=None, ignore_eos=True, percentile_metrics='ttft,tpot,itl,e2el', metric_percentiles='99', goodput=None, top_p=None, top_k=None, min_p=None, temperature=None, tokenizer_mode='auto', served_model_name=None, lora_modules=None, ramp_up_strategy=None, ramp_up_start_rps=None, ramp_up_end_rps=None)
INFO 07-29 23:35:52 [datasets.py:355] Sampling input_len from [600, 600] and output_len from [125, 125]
Starting initial single prompt test run...
Initial test run completed. Starting main benchmark run...
Traffic request rate: 2000.0
Burstiness factor: 1.0 (Poisson process)
Maximum request concurrency: None
  0% | 0/10000 [00:00<?, ?/s]
huggingface/tokenizers: The current process just got forked, after parallelism has already been used. Disabling parallelism to avoid deadlocks...
To disable this warning, you can either:
  - Avoid using 'tokenizers' before the fork if possible
  - Explicitly set the environment variable TOKENIZERS_PARALLELISM=(true | false)
100% | 10000/10000 [03:01<00:00, 55.10it/s]
===== Serving Benchmark Result ======
Successful requests: 2548
Benchmark duration (s): 181.49
Total input tokens: 1525710
Total generated tokens: 318500
Request throughput (req/s): 14.04
Output token throughput (tok/s): 1754.87
Total Token throughput (tok/s): 10161.24
-----Time to First Token-----
Mean TTFT (ms): 85567.69
Median TTFT (ms): 90915.17
P99 TTFT (ms): 168053.72
-----Time per Output Token (excl. 1st token)-----
Mean TPOT (ms): 452.68
Median TPOT (ms): 522.39
P99 TPOT (ms): 590.92
-----Inter-token Latency-----
Mean ITL (ms): 452.69
Median ITL (ms): 161.38
P99 ITL (ms): 1666.37
-----End-to-end Latency-----
Mean E2EL (ms): 141699.53
Median E2EL (ms): 155198.55
P99 E2EL (ms): 180117.52
=====
```

## □ vLLM repository 내 benchmarks 내 코드 사용 [75, 76]

vLLM benchmarks [Benchmark] Add support for multiple batch size benchmark through CLI... codebase · 3 weeks ago · History

- batch\_size\_benchmarks** [Benchmark] Remove duplicate res1,div (#20023) last month
- usage\_benchmarks** [Misc] Add SPoX+CopyRightTest (#19990) last month
- fixed\_kernels** [Misc] Add SPoX+CopyRightTest (#19990) last month
- kernels** [Benchmark] Add support for multiple batch size benchmark through CLI... 5 weeks ago
- overheads** [Misc] Add SPoX+CopyRightTest (#19990) last month
- structured\_schemas** benchmarks: simplify test (sonneca) (#19967) 4 months ago
- README.md** [Misc] Use codeblock block for benchmark examples (#20017) last month
- wireframe** [Misc] [WIP] benchmarks: Add profile to astore script (#19971) last month
- decided\_request\_func.py** [Benchmark] Fix request loss if "prompt" is returned (#19933) last month
- benchmark\_database.py** [Benchmark] Fix multiple bugs in bench and add args to spec... decod... last month
- benchmark\_latency.py** [Misc] Modulates Cu Argument Parsing in Benchmark Scripts (#19933) last month
- benchmark\_long\_document\_orthography.py** [Misc] Modulates Cu Argument Parsing in Benchmark Scripts (#19933) last month
- benchmark\_prefix\_caching.py** [Misc] Modulates Cu Argument Parsing in Benchmark Scripts (#19933) last month
- benchmark\_prioritization.py** [Misc] Modulates Cu Argument Parsing in Benchmark Scripts (#19933) last month
- benchmark\_pennying.py** [Misc] Remove redundant char (#20087) last month
- benchmark\_serving\_structured\_output.py** [Misc] Modulates Cu Argument Parsing in Benchmark Scripts (#19933) last month
- benchmark\_throughput.py** [Benchmark] Fix vote of type 'None' (leftovers) is not iterable (#19932) last month
- benchmark\_utils.py** Handle non-serializable objects when dumping benchmark results (#19714) last month
- wireframe.html** [Doc] More examples and further usage example (#19666) 2 months ago
- unstructured\_output\_benchmark.h** [Benchmark] Refactor unstructured\_output\_benchmark.h (#17722) 2 months ago
- server.c** [Misc] fix(benchmarks): Add Profiling Benchmark to Serving benchmark (#... last year

## □ vLLM project의 Guidellm 사용 [77, 78]

Benchmarks Metadata:																		
Run Id:	f4d92c1-9a1d-4cd3-b237-83fcc971bd3																	
Duration:	17.3 seconds																	
Strategies:	'synchronous', 'throughput', 'constant', 'constant', 'constant', 'constant', 'constant', 'constant', 'constant', 'constant', 'constant', 'constant'																	
Args:	max_number_workers: 8, max_duration: 10.0, warmup_number_workers: 1, cooldown_number_workers: 1, cooldown_duration: 1.0, worker_type: "generative_requests_worker", backend_type: "openai", backend_target: "http://192.168.4.13:8080", backend_model: "neuralmagic/Owen3-235B-A22B", request_loader_type: "generative_request_loader", data: {"prompt_tokens": 128, "output_tokens": 64}, data_args: "none", processor: "neuralmagic/Owen3-7.8-quarantine"																	
Request Loader:	type: "generative_request_loader", data: {"prompt_tokens": 128, "output_tokens": 64}, data_args: "none", processor: "neuralmagic/Owen3-7.8-quarantine"																	
Extras:	None																	
Benchmarks Info:																		
Metadata																		
Benchmark	Start Time	End Time	Duration (s)	Requests Made	Comp	Inc	Err	Prompt Comp	Tok/Req	Output Comp	Tok/Req	Prompt Tok Total	Tok Total	Output Tok Total				
synchronous	14:42:15	14:42:15	0.00	6	1	0	128.0	128.0	0.0	64.0	43.0	0	768	128	0	384	43	0
throughput	14:42:16	14:42:26	10.00	126	510	0	128.1	128.1	0.0	63.9	36.9	0	16144	65398	0	8064	7979	0
constant	14:42:27	14:42:27	0.00	10	10	0	128.0	128.0	0.0	64.0	29.0	0	4354	5268	0	2175	174	0
constant3.74	14:42:44	14:42:54	10.00	24	61	0	128.1	128.1	0.0	64.0	29.0	0	4354	5268	0	2175	174	0
constant5.27	14:42:55	14:43:05	10.00	48	9	0	128.0	128.0	0.0	63.9	32.9	0.0	6146	1152	0	3068	296	0
constant6.77	14:43:06	14:43:16	10.00	60	12	0	128.0	128.0	0.0	64.0	32.9	0.0	6146	1152	0	3068	296	0
constant8.24	14:43:17	14:43:27	10.00	76	15	0	128.0	128.0	0.0	64.0	32.9	0.0	7938	177	0	3397	317	0
constant9.88	14:43:29	14:43:39	10.00	89	19	0	128.1	128.1	0.0	63.9	31.1	0.0	11266	2433	0	5627	598	0
constant11.41	14:43:40	14:43:50	10.00	101	24	0	128.1	128.2	0.0	64.0	32.2	0.0	12934	3077	0	6461	740	0
constant12.95	14:43:51	14:44:01	10.00	113	26	0	128.1	128.0	0.0	64.0	31.7	0.0	24475	3505	0	7238	655	0

Benchmarks Stats:															
Metadata	Benchmark	Request Rate	Per Second	Concurrency	Out Tok/sec	Total Tok/sec	Req Latency (ms)	TFTT (ms)	ITL (ms)	TOTP (ms)	TPOT (ms)				
					mean	median	mean	median	mean	median	mean				
	synchronous	0.67	1.00	4.27	213.1	1.49	1.40	1.59	46.0	49.9	22.9	29.1	22.6	22.7	
	throughput	12.95	113.50	826.5	4133.1	8.77	6.65	9.66	1196.8	1112.2	2057.8	129.1	124.6	118.2	122.7
	constant	2.06	3.22	131.9	657.9										

[74] <https://docs.vllm.ai/en/v0.9.2/cli/index.html#bench>

[75] <https://docs.vllm.ai/en/v0.9.2/contributing/profiling.html>

[76] <https://github.com/vllm-project/vllm/tree/v0.9.2/benchmarks>

[77] <https://arxiv.org/pdf/2502.06494.pdf>

# Production Issues

- ## ☐ Gloo connectFullMesh failed with… [79]

- Multi node로 process 연결 시 GLOO 연결 실패

→ “GLOO\_SOCKET\_IFNAME” 환경 변수를 사용할 intend

Debugging vLLM NCCL (PyNcclCommunicator) all\_reduce Issue in Multi-Node Environment #11353

**Closed** Zerohertz announced in Q&A

**Zerohertz** on Dec 20, 2024

edited - ...

영어로 된 본문 주석 - 한국어로 번역

Hi, I'm in the process of building a multi-node LLM serving environment via vLLM.  
However, some settings were not set correctly, causing an error, and I'm verifying the environment through [official document](#).  
I have confirmed that steps 1 and 2 ([Pytorch NCCL](#) and [Pytorch GL00](#)) was successful.

But there was a problem in `vLLM NCCL` .  
So I checked as follows.

```
dist.init_process_group(backend="nccl")
local_rank = dist.get_rank() % torch.cuda.device_count()
torch.cuda.set_device(local_rank)
world_size = dist.get_world_size()
gloo_group = dist.new_group(ranks=list(range(world_size)), backend="gloo")
pynccl = PyccCommunicator(group=gloo_group, device=local_rank)

s = torch.cuda.Stream()
data = torch.FloatTensor(
    [
        3,
        128
    ], device="cuda"
).to("cuda")
with torch.cuda.stream(s):
    mean_all_reduce = torch.mean(data, [0], local_rank, data before all_reduce: (data))
    pynccl.all_reduce(data, streams=s)
    logger.debug(f"Rank {local_rank}, data after all_reduce: (data)")
    value = data.mean().item()
```

Also all scripts run in Docker and I ran the Docker container as below:

- Qwen 계열 model의 무한 token 생성 [80, 81]

- FlashInfer kernel 내 cascade inference 사용으로 인해 발생  
→ “--disable-cascade-attn”를 사용하여 해결

[Bug]: Degradation of Qwen/Qwen3-30B-A3B performance depending on batch size #17652

PART 5

---

# Wrap-up

---

# Roadmap Q3 2025 [82]

5. Wrap-up

## ❑ V1 Engine

- V0 Engine 완전 제거
- Core scheduler 최적화 및 확장
- Async scheduling, multi-modal 처리 등 기능 구현

## ❑ Large Scale Serving

- Mixture-of-Experts (MoE) model의 안정적 scale-out serving
- 분산 서빙 표준화 및 autoscaling

## ❑ Models

- 다양한 framework (training, authoring)의 tokenizer, configuration, processor 지원
- Sparse attention mechanism
- 1B 이하의 소형 모델 성능 향상

## ❑ Use Cases

- RLHF
  - 동기화 및 resharding을 위한 가중치 로딩 최적화
  - Multi-turn scheduling
- Evaluation
  - Batching order에 영향받지 않는 full determinism 지원 (with/without prefix cache)
- Batch Inference
  - Prefix caching과 함께 scale-out을 위한 data parallel router
  - CPU KV cache offloading

# Conclusion

## ❑ OpenAI-Compatible Server

- LangChain, Gemini CLI 등 기존 생태계와의 호환성 유지
- Tool Calling, Reasoning 등 확장된 기능 활용 가능

## ❑ Architecture

- KV cache와 PagedAttention 기반의 효율적 메모리 관리
- V1 Engine 도입으로 단순한 고성능 구조 구현

## ❑ Production Deployment

- TP/PP/DP/EP 등 다양한 병렬 처리 전략 적용
- Kubernetes 기반 multi-node 분산 추론
- LoRA adapter를 활용한 사용자 맞춤형 경량화 serving
- Prometheus + Grafana를 활용한 observability 확보
- vllm bench 및 benchmark script를 통한 성능 평가

# vLLM Meetup in Korea<sup>[83, 84]</sup>

5. Wrap-up



Junghwan Park · 1촌

AI/Data Engineer @ SK Telecom | Lead Maintainer @ PyTorch Korea User Group | PyT...  
17시간 ·

The first vLLM Meetup in Korea will be held on the evening of Aug 19th!

Hosted by RedHat and [Rebellions](#), with [PyTorch Korea User Group](#) and [SqueezeBits](#).

For more details, check out the link below: <https://lnkd.in/gRTGeKuH>

번역 표시

8/19(화) 저녁, 한국에서 열리는 첫번째 vLLM meetup에 함께 해주세요!  
[discuss.pytorch.kr](https://discuss.pytorch.kr)

8월 8일 오후 6:00

[확정 안내] First vLLM Korea Meetup 참석 관련 안내드립니다

안녕하세요, First vLLM Korea Meetup with Red Hat and Rebellions 에 관심 가져 주시고 소중한 관심 보내주셔서 진심으로 감사드립니다.

이번 밋업은 저희가 처음 예상했던 규모보다 무려 **3배 이상 많은 분들께서 신청해주셔서**, vLLM에 대한 국내 유저분들의 뜨거운 관심을 실감할 수 있었습니다. 그만큼 모든 분을 모실 수 없었던 점이 매우 아쉬웠지만, ...

# EoD

---

GitHub



LinkedIn



Coffee Chat

