
PyTorch
CONFERENCE
EUROPE 2026
Lights, Camera, Inference!
Video Generation as a Service with vLLM-Omni

@dougbtv

Doug Smith

- MLOps Engineer (currently) working on vLLM
- Blog: https://dougbtv.com/

@oglok

Ricardo Noriega

- Principal Software Engineer at Red Hat's Office of the CTO
- Blog: http://oglok.es

AGENDA

- What is vLLM-Omni
- Omni in action
- vLLM-Omni is moving at light speed.
- Gaps
- Run it yourself! And get involved.

What is vLLM-Omni?

vLLM-Omni

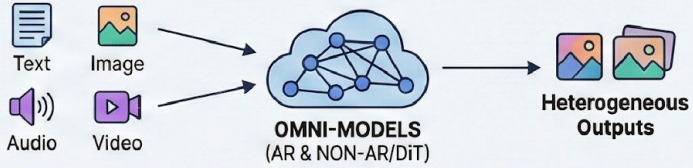
Easy, Fast, and Cheap Omni-Modality Model Serving

THE SHIFT TO OMNI-MODALITY

TRADITIONAL SERVING (Text-only AR)

EVOLVING INFRASTRUCTURE

OMNI-MODALITY LANDSCAPE



vLLM-OMNI ARCHITECTURE: RE-IMAGINED HETEROGENEOUS PIPELINE

KEY FEATURES & BENEFITS

SIMPLICITY

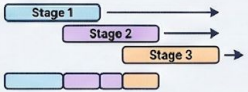
Seamless Integration
(OpenAI API, Hugging Face Models).
Easy to Use if you know vLLM.

FLEXIBILITY

OmniStage Abstraction
Supports various Omni-Models
(e.g., Qwen-Omni, Qwen-Image).

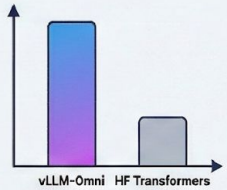
PERFORMANCE

PIPELINED STAGE EXECUTION

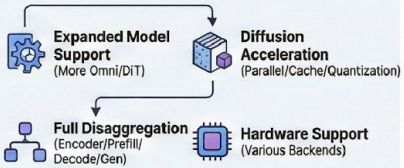


Overlapping Computation for High Throughput
Efficient Memory Management.

PROVEN EFFICIENCY & FUTURE ROADMAP



BENCHMARKED EFFICIENCY



ROADMAP & DEVELOPMENT

GET STARTED & JOIN THE COMMUNITY

Installation & Examples: github.com/vllm-project/vllm-omni
Documentation: vllm-omni.readthedocs.io

Slack: #sig-omni at slack.vllm.ai
Weekly Meeting: Tuesdays 19:30 PDT

Offline Inference

A Python interface for offline batched inference for Qwen3-Omni/Qwen-Image

```
from vllm_omni import Omni

# Example prompts.
inputs = {"prompt": prompt,
                "multi_modal_data": {"video":
video_frames, "audio": audio_data,},}
# Create an omni with HF model name.
omni =
Omni(model="Qwen/Qwen3-Omni-30B-A3B
-Instruct")
# Generate texts and audio from the
multi-modality inputs.
outputs = omni.generate(inputs)
```

```
from vllm_omni import Omni

# Example prompts.
inputs = "A cup of coffee on the table"


# Create an omni with HF model name.
omni =
Omni(model="Qwen/Qwen-Image-2512")
# Generate image from multi-modality inputs.
outputs = omni.generate(inputs)
```

Online Serving

A FastAPI-based server for online serving for Qwen3-Omni

Server

```
$ vllm serve Qwen/Qwen3-Omni-30B-A3B-Instruct --omni --port 8091
```

Client

```
$ curl -sS -X POST http://localhost:8091/v1/chat/completions\
    -H "Content-Type: application/json" \
    -d '{
        "model": "Qwen/Qwen3-Omni-30B-A3B-Instruct",
         "messages": "Why is this video funny? "
         "sampling_params_list": $sampling_params_list,
        }'
```

Online Serving

A FastAPI-based server for online serving for Qwen-Image

Server

```
$ vllm serve Qwen/Qwen-Image-2512 --omni --port 8091
```

Client

```
$ curl -X POST http://localhost:8091/v1/images/generations \
  -H "Content-Type: application/json" \
  -d '{
    "prompt": "a dragon laying over the spine of the Green Mountains of Vermont",
    "size": "1024x1024",
    "seed": 42
  }' | jq -r '.data[0].b64_json' | base64 -d > dragon.png
```

Broad Model Support

Qwen @Alibaba_Qwen · Jan 2
Big thanks to @vllm_project for day-zero support of Qwen-Image-2512!

vLLM @vllm_project · Dec 31, 2025
Congrats to @Alibaba_Qwen on the release of Qwen-Image-2512!
We are thrilled to announce Day-0 support in vLLM-Omni. You can now serve this SOTA open-source image model with our optimized pipelined architecture immediately.

Qwen-Omni
Qwen-Image

BAGEL

LongCat

Z-Image

Tencent Hunyuan
Image/3D

SD3

Wan

MiMo

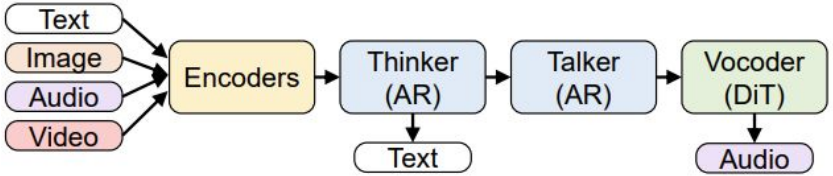
Flux

StepFun

GLM

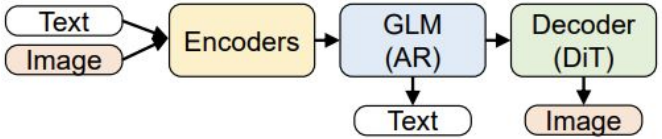
Ovis-Image

Multi-modality Models



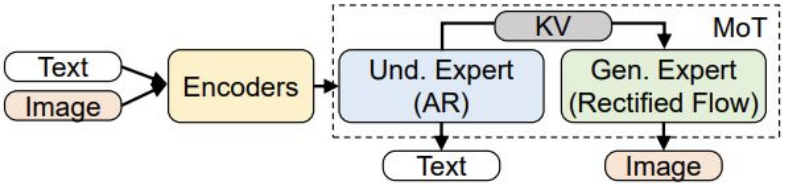
(a) Thinker-Talker (Qwen2.5-Omni)

Backbone: **(multi)** AR + DiT
Models: Qwen-Omni/Ming-Omni
Tasks: any-to-any



(b) AR + DiT (GLM-Image)

Backbone: AR + DiT
Models: Qwen-Image/GLM-Image
Tasks: t2i, t2v, i2i...



(c) AR + Specialized Generator (BAGEL)

Backbone: AR + Spec. Gen.
Models: BAGEL, Hunyuan Image 3.0
Tasks: t2i, i2i, i2t...

How does it interoperate with vLLM?

Right now, everything gets monkey patched in [vllm_omni/patch.py]

```python
for module_name, module in sys.modules.items():
    # only do patch on module of vllm, pass others
    if "vllm" not in module_name:
        continue
    if hasattr(module, "EngineCoreOutput") and module.EngineCoreOutput == _OriginalEngineCoreOutput:
        module.EngineCoreOutput = OmniEngineCoreOutput
    if hasattr(module, "EngineCoreOutputs") and module.EngineCoreOutputs == _OriginalEngineCoreOutputs:
        module.EngineCoreOutputs = OmniEngineCoreOutputs
    if hasattr(module, "TokensPrompt") and module.TokensPrompt == _OriginalTokensPrompt:
        module.TokensPrompt = OmniTokensPrompt
    if hasattr(module, "MRotaryEmbedding") and module.MRotaryEmbedding == _OriginalMRotaryEmbedding:
        module.MRotaryEmbedding = OmniMRotaryEmbedding
    if hasattr(module, "Request") and module.Request == _OriginalRequest:
        module.Request = OmniRequest
    if hasattr(module, "EngineCoreRequest") and module.EngineCoreRequest == _OriginalEngineCoreRequest:
        module.EngineCoreRequest = OmniEngineCoreRequest
```

Omni in action.

What we love…

vLLM-Omni

- Builds on an awesome stack
- vLLM is where we're building a lot of our integrations as it is

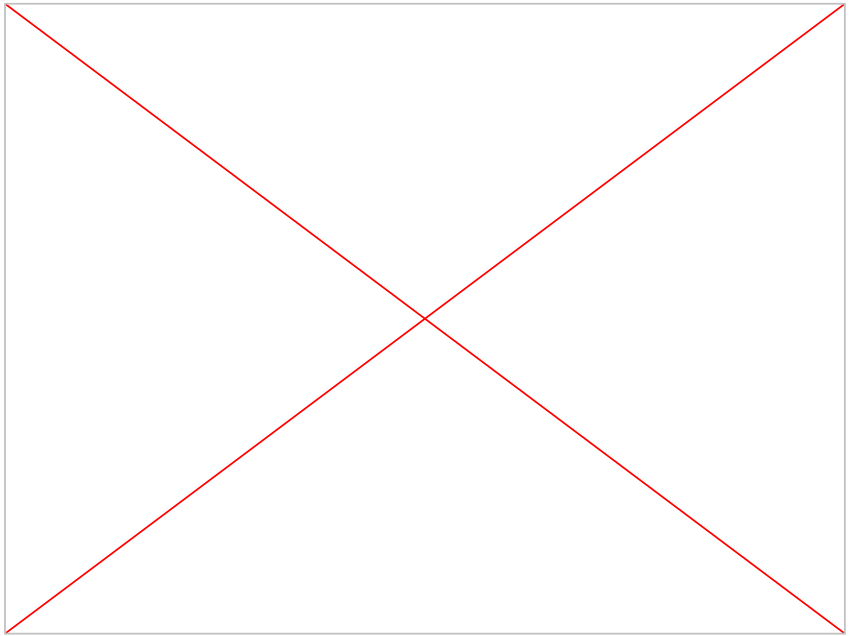
LTX-2

- Incredible feature set for an open weight video gen model

We don't love… the custom license.

Pretty typical limitations…

- ~20-second ceiling
- Temporal Drift
- “The Slideshow Effect”
- Spaghetti Limbs
- Weird music.



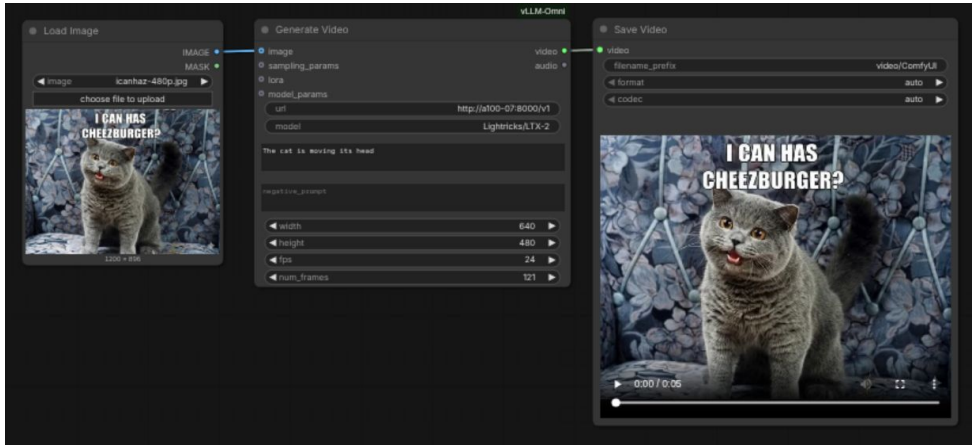
vLLM-Omni is moving at light speed.

vLLM-Omni LTX-2 integration when we wrote the talk abstract…

And where it is today…

ALL THE FEATURES!

- LTX-2 support
- Text-to-video (T2V)
- Image-to-video (I2V)
- /v1/videos endpoint
- ComfyUI nodes
- Asynchronous video serving



Gaps.

Location, location, location.

How do get requests to the right places?

- Different model types expose different API surfaces
  - An omni-modal model like Qwen3-Omni supports chat completions, while a diffusion model like Wan2.2 only supports video generation endpoints
- No standardized capability discovery mechanism
  - Clients can't easily query "what endpoints does this model support?"
- Routing complexity
  - You might need to route requests to the right model based on the endpoint AND the model's capabilities
- Version skew
  - Different model versions might support different subsets of endpoints

Architecture under pressure.

- Lots of duplication in diffusion pipeline.
  - Copied diffusers code, it'll be harder to find later instead of directly using the dependency.
- Architectural splits / weird structures.
  - We want to do "more cool stuff" with omni, but, there's kind of a split between omni-modality models and "just diffusion" (for example)
- Not because of carelessness, mostly because of light speed.
- All the *tech debt*.

Schrödinger's platform

(Basically all) Open source is a moving target

- APIs, models, and pipelines are constantly evolving

Shipping requires freezing a moment in time

- Choosing a version and calling it "stable"

This enables *production product*.

Run it yourself!

The gisthub gist!

Available as a github gist.

It’s a great time to get involved!!!

Seriously – you can make a big impact.

- Agenda doc
- Meeting time: 7pm-8pm GMT+1 every other Friday *(tell your friends you’ll be late to dinner)*
    - Let’s try to fix this! We want global community
- Meeting link
- Meeting Calendar invite
- Github: https://github.com/vllm-project/vllm-omni
- Website: https://docs.vllm.ai/projects/vllm-omni/en/latest/

THANK YOU!

Merci beaucoup!

PyTorch CONFERENCE EUROPE 2026