Accelerate Generative AI

OpenVINO™

GenAI Cheat Sheet

2024 | Updates are here.
Post your questions here.
Read the documentation here.

Deploy Generative AI with ease: the OpenVINO™ GenAI library provides developers with necessary tools to optimize and deploy Generative AI models. Based on OpenVINO™ tools and runtime, it provides best performance and characteristics for inference on the supported platforms.

Which use cases are supported by OpenVINO™ GenAI?

OpenVINO™ GenAI provides intuitive and simple C++/Python APIs to run:

- Text generation models (text summarization, rewriting, chatbots, etc.)
- Image generation models (using Diffuser-based architectures)
- Whisper-based speech transcription and translation
- Image processing with Visual Language Models (LLaVa and others)

Use OpenVINO™ GenAI to run models from

🤗 Hugging Face | ModelScope

Install OpenVINO™ GenAI

Linux install | Windows install | macOS install

PyPI example for Linux, macOS & Windows:

```
#set up python venv
python -m pip install openvino-genai
```

A full list of installation options is here.

Download converted and optimized models from 🤗 Hub

Use the huggingface_hub package to download:

```
pip install huggingface_hub
```

Download a model to a local folder:

```
huggingface-cli download "OpenVINO/phi-2-fp16-ov" --local-dir model_path
```

Browse models from OpenVINO Toolkit in our collections:

- Large Language Models
- Text2Speech Models

OpenVINO™ GenAI: Supported Models

A list of validated models is here.

Note that even if your model is not listed, it may still work.

Run LLMs for Text Generation

GenAI Cheat Sheet

2024 | Updates are here.
Post your questions here.
Read the documentation here.

1a. Convert and optimize LLMs from Hugging Face 🤗

Use the optimum-intel package to convert and optimize models

```
pip install optimum-intel[openvino]
```

Download and convert a model to the OpenVINO IR format while keeping full model precision

```
optimum-cli export openvino --model meta-llama/Llama-2-7b-chat-hf --weight-format
fp16 ov_llama_2
```

[Recommended] Download, convert a model, and compress weights to the int4 precision

```
optimum-cli export openvino --model meta-llama/Llama-2-7b-chat-hf --weight-format
int4 ov_llama_2
```

A full list of conversion options is here.

Pre-converted LLMs are here

1b. Convert and optimize LLMs from Model Scope

Use modelscope and optimum-intel packages to convert and optimize models

```
pip install modelscope optimum-intel[openvino]
```

Download the required model to a local folder

```
modelscope download --model 'Qwen/Qwen2-7b' --local_dir model_path
```

[Recommended] convert the model and compress weights to the int4 precision (or int8 or fp16)

```
optimum-cli export openvino -m model_path --task text-generation-with-past
--weight-format int4  ov_qwen_25
```

2. Run model using OpenVINO™ GenAI

In Python:

```
import openvino_genai as ov_genai
pipe = ov_genai.LLMPipeline(model_path, ”GPU”)
print(pipe.generate("What is OpenVINO?", max_length=200))
```

In C++:

```
#include "openvino/genai/llm_pipeline.hpp"
#include <iostream>

int main(int argc, char* argv[]) {
    ov::genai::LLMPipeline pipe(model_path, “GPU");
    std::cout << pipe.generate(”What is Large Language Model?",
                                ov::genai::max_new_tokens(200));
}
```

Use CPU or GPU as devices without any other code change

When running LLMs it is also possible to:

- Use different generation parameters (sampling types, etc.)
- Optimize for chat scenarios by using the chat mode
- Load LoRA adapters and dynamically switch between them without recompilation
- Use draft models to accelerate generation via Speculative Decoding

Check out our Python and C++ samples.

Generate Images using Diffusers

GenAI Cheat Sheet

2024 | Updates are here.
Post your questions here.
Read the documentation here.

1a. Convert and optimize Models from Hugging Face 🤗

Use the optimum-intel package to convert and optimize models

```
pip install optimum-intel[openvino]
```

Download and convert a model to the OpenVINO IR format while keeping full model precision

```
optimum-cli export openvino --model stabilityai/stable-diffusion-xl-base-1.0
--weight-format fp16 ov_SDXL
```

[Recommended] Download, convert the model, and compress weights to the int4 precision

```
optimum-cli export openvino --model stabilityai/stable-diffusion-xl-base-1.0
--weight-format int4 ov_SDXL
```

A full list of conversion options is here.

1b. Convert and optimize Models from Model Scope

Use modelscope and optimum-intel packages to convert and optimize models

```
pip install modelscope optimum-intel[openvino]
```

Download the required model to a local folder

```
modelscope download --model AI-ModelScope/stable-diffusion-xl-base-1.0 '
--local_dir model_path
```

[Recommended] convert the model and compress weights to the int4 precision (can use int8 & fp16)

```
optimum-cli export openvino -m model_path --task text-generation-with-past
--weight-format int4  ov_SDXL
```

2. Run model using OpenVINO™ GenAI

In Python:

```
import openvino_genai as ov_genai
pipe = openvino_genai.Text2ImagePipeline(model_dir, “GPU”)
image_tensor = pipe.generate(prompt)
```

In C++:

```
#include "openvino/genai/image_generation/text2image_pipeline.hpp"

int main(int argc, char* argv[]) {
    ov::genai::Text2ImagePipeline pipe(models_path, “GPU”);
    ov::Tensor image = pipe.generate(prompt);
}
```

Use CPU or GPU as devices without any other code change

When generating images, it is also possible to:

- Alter parameters (width, height, iterations) and compile models for static size
- Load LoRA adapters (in safetensor format) and dynamically switch between them
- Generate multiple images per one request

Check out our Python and C++ samples.

Processing speech by Whisper

GenAI Cheat Sheet

2024 | Updates are here.
Post your questions here.
Read the documentation here.

1a. Convert and optimize models from Hugging Face

Use the optimum-intel package to convert and optimize models

```
pip install optimum-intel[openvino]
```

Download and convert the model to the OpenVINO format while keeping full model precision

```
optimum-cli export openvino --trust-remote-code --model openai/whisper-base
ov_whisper
```

Optimization sample is available here.

Pre-converted Models are here

1b. Convert and optimize models from Model Scope

Use modelscope and optimum-intel packages to convert and optimize models

```
pip install modelscope optimum-intel[openvino]
```

Download the required model to a local folder

```
modelscope download --model 'AI-ModelScope/whisper-large-v3-turbo’
--local_dir model_path
```

[Recommended] convert the model

```
optimum-cli export openvino -m model_path
--task automatic-speech-recognition-with-past ov_whisper
```

2. Run model using OpenVINO™ GenAI

In Python:

```
import openvino_genai as ov_genai
pipe = openvino_genai.WhisperPipeline(model_dir, ”CPU”)
#Pipeline expects normalized audio with Sample Rate of 16kHz
raw_speech = read_wav(…)
result = pipe.generate(raw_speech)
```

In C++:

```
#include "openvino/genai/whisper_pipeline.hpp"

int main(int argc, char* argv[]) {
    ov::genai::WhisperPipeline pipe(model_path, “CPU”);
    //Pipeline expects normalized audio with Sample Rate of 16kHz
    ov::genai::RawSpeechInput raw_speech = read_wav(…);
    auto result = pipe.generate(raw_speech, config);
}
```

Use CPU or GPU as devices without any other code change

When running Whisper models, it is also possible to:

- Translate transcription to English
- Predict timestamps
- Process Long-Form (>30 seconds) audio

Check out our Python and C++ samples.

Analyzing images with VLMs

2024 | Updates are here.
Post your questions here.
Read the documentation here.

GenAI Cheat Sheet

1a. Convert and optimize VLMs from Hugging Face 🤗

Use optimum-intel package to convert and optimize models

```
pip install optimum-intel[openvino]
```

Download and convert a model to the OpenVINO IR format while keeping full model precision

```
optimum-cli export openvino --model openbmb/MiniCPM-V-2_6
--trust-remote-code -weight-format fp16 ov_MiniCPM-V-2_6
```

[Recommended] Download, convert the model, and compress weights to the int4 precision

```
optimum-cli export openvino --model openbmb/MiniCPM-V-2_6
--trust-remote-code -weight-format int4 ov_MiniCPM-V-2_6
```

A full list of conversion options is here.

1b. Convert and optimize VLMs from Model Scope

Use modelscope and optimum-intel packages to convert and optimize models

```
pip install modelscope optimum-intel[openvino]
```

Download the required model to a local folder

```
modelscope download --model OpenBMB/MiniCPM-V-2_6 ' --local_dir model_path
```

[Recommended] convert the model and compress weights to the int4 precision (or int8 or fp16)

```
optimum-cli export openvino -m model_path --task image-text-to-text
--weight-format int4  ov_MiniCPM-V-2_6 --trust-remote-code
```

2. Run model using OpenVINO™ GenAI

In Python:

```
import openvino_genai as ov_genai
pipe = ov_genai.VLMPipeline(model_dir, “GPU”)
#read images to OV Tensors
rgbImages = read_images(…)
print(pipe.generate(prompt, images=rgbImages, max_new_tokens=100))
```

In C++:

```
#include “openvino/genai/visual_language/pipeline.hpp”
#include <iostream>

int main(int argc, char* argv[]) {
    ov::genai::VLMPipeline pipe(model_dir, ”GPU”);
    //Read images into vector of OV Tensors
    std::vector<ov::Tensor> rgbImages = read_images(…);
    std::cout << pipe.generate(prompt, ov::genai::images(rgbImages),
                                       ov::genai::max_new_tokens(100));
}
```

Use CPU or GPU as devices without any other code change

When running VLMs, it is also possible to:

- Use different generation parameters (sampling types, etc.)
- Optimize for chat scenarios by using chat mode
- Pass multiple images to a model

Check out our Python and C++ samples.