OpenVINO™
DEVCON中国
系列工作坊 2023

OpenVINO™ 2023.1 关键亮点解析

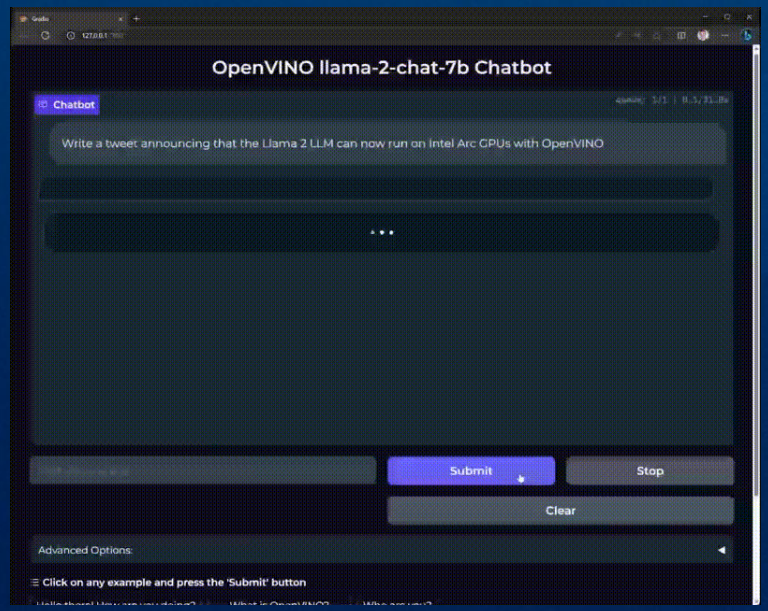
武卓

英特尔 OpenVINO 布道师

杨亦诚

英特尔 AI 软件工程师

2023.1



GenAI

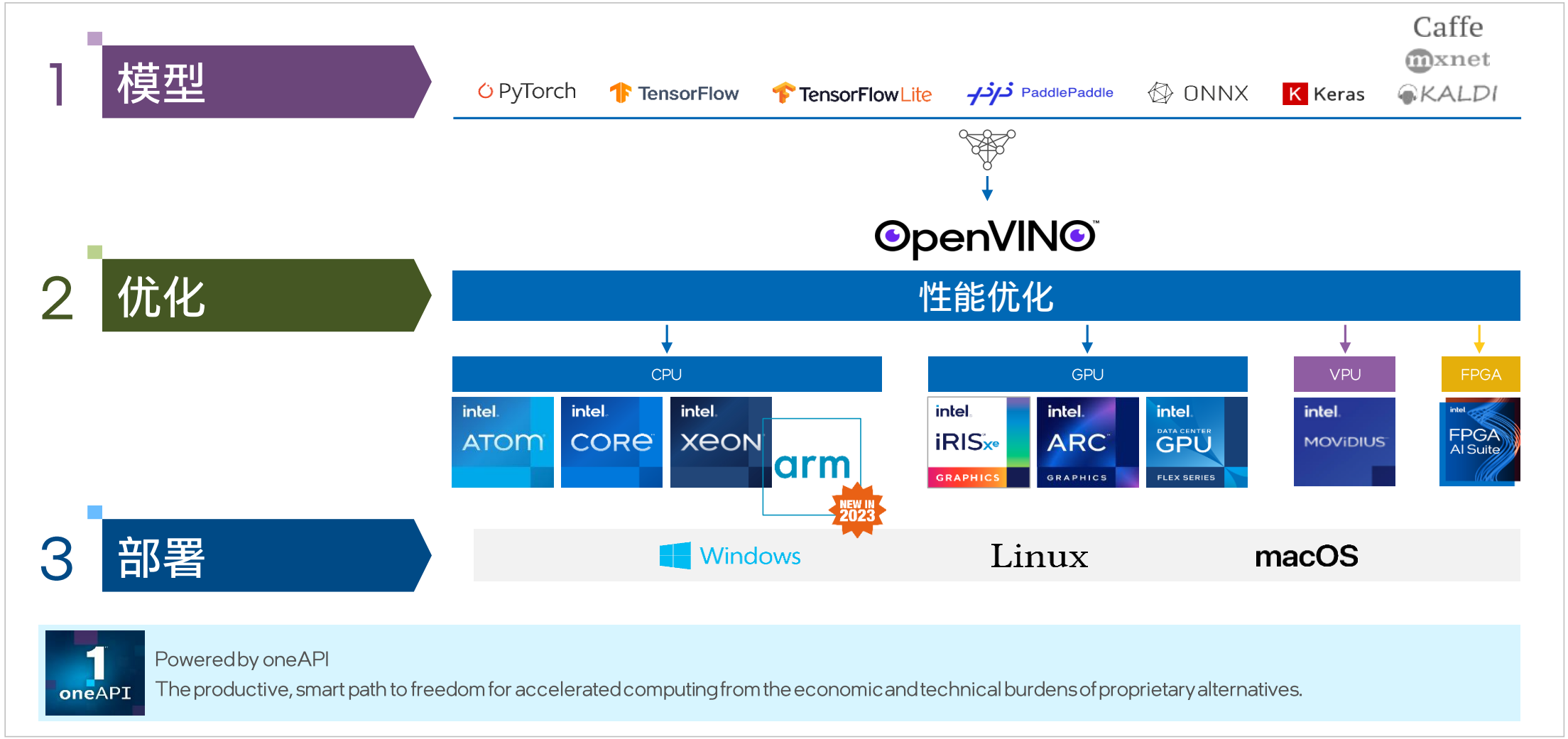
LLM

OpenVINO™ 2023.1

降低生成式 AI 接入门槛

- **更简单的工作流程**
- 更好的 PyTorch 兼容性
- 针对大模型的优化
- 最新的notebook示例

OpenVINO™ 工具套件



oneAPI

Powered by oneAPI

The productive, smart path to freedom for accelerated computing from the economic and technical burdens of proprietary alternatives.

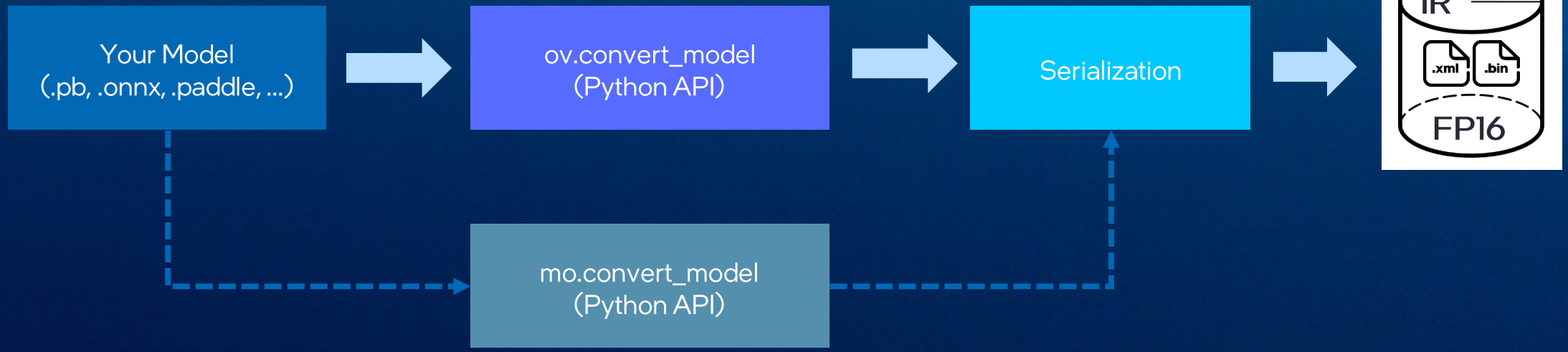
mo -h

pip install openvino-dev

ovc -h

pip install openvino

使用 Python接口： ov.convert_model




```
ov_model = ov.convert_model("model.onnx", compress_to_fp16=True)

ov.save_model(ov_model, "converted_model.xml")
```

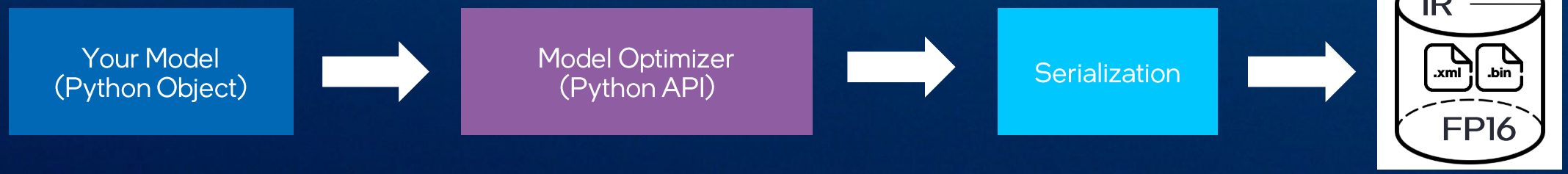
OpenVINO™ 2023.1

更容易进行AI部署和加速

- 更简单的工作流程
- **更好的 PyTorch 兼容性**
- 针对大模型的优化
- 最新的 notebook 示例

New PyTorch frontend

- torch.nn.Module derived classes
- torch.jit.ScriptModule
- torch.jit.ScriptFunction



New PyTorch frontend

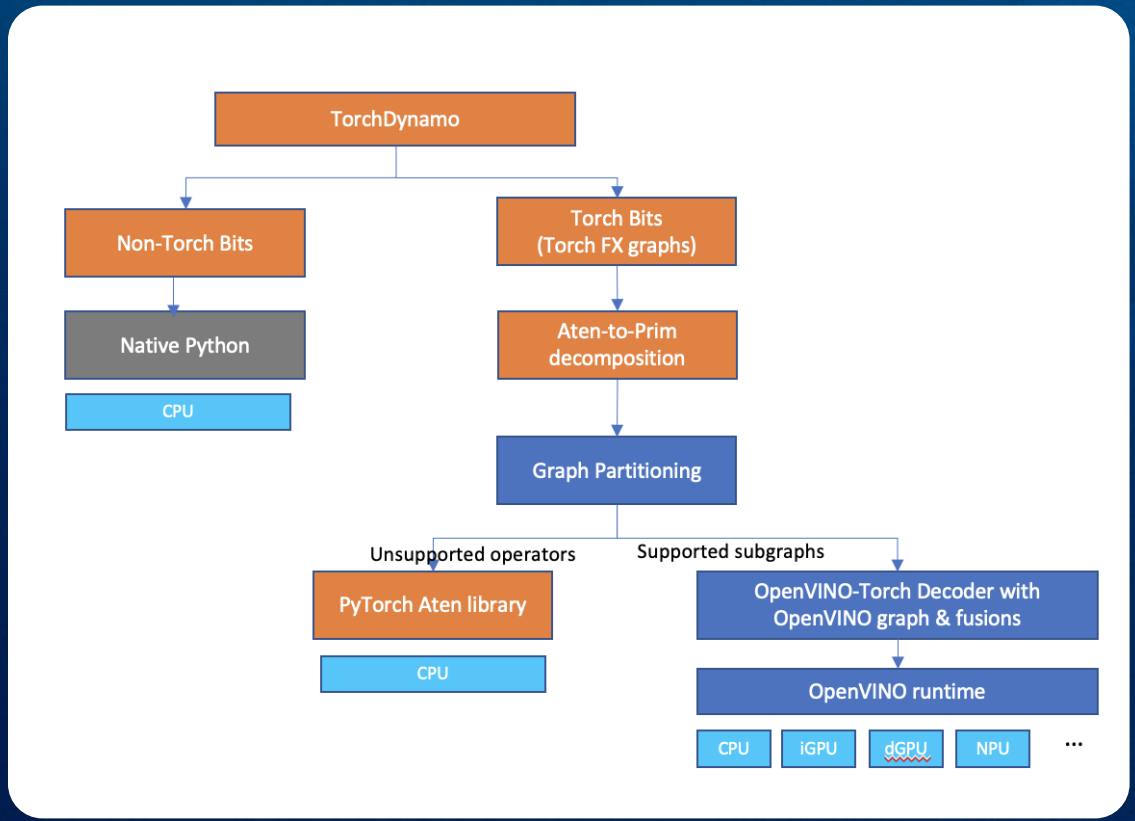
```
import openvino as ov
import torch
from torchvision.models import resnet50

model = resnet50(pretrained=True)

# prepare input_data
input_data = torch.rand(1, 3, 224, 224)

ov_model = ov.convert_model(model, example_input=input_data)

###### Option 1: Save to OpenVINO IR:

# save model to OpenVINO IR for later use
ov.save_model(ov_model, 'model.xml')

###### Option 2: Compile and infer with OpenVINO:

# compile model
compiled_model = ov.compile_model(ov_model)

# run the inference
result = compiled_model(input_data)
```

torch.compile

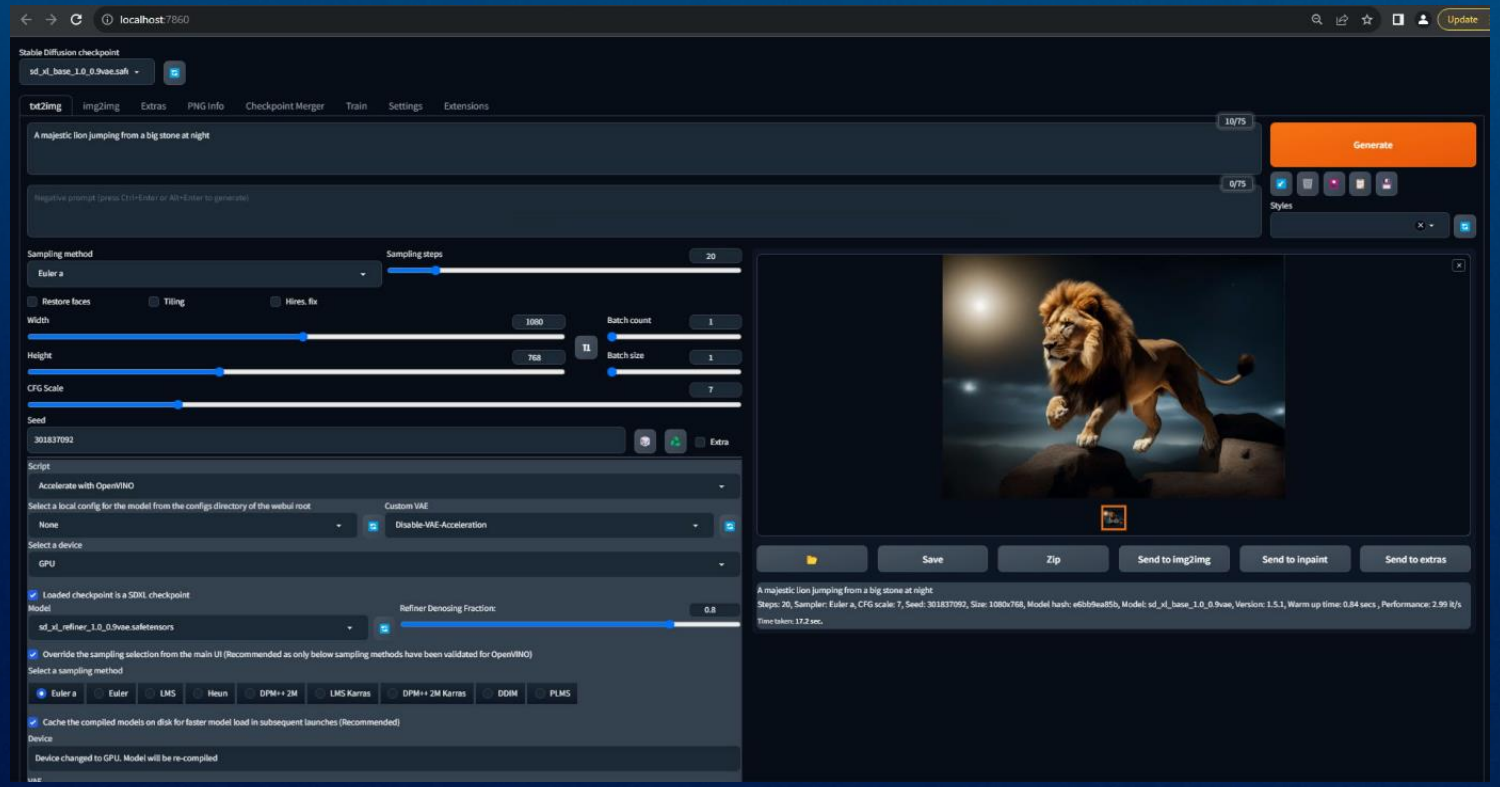
在 PyTorch 中调用 OpenVINO

```python
import torch
import torchvision.models as models
import openvino.torch
model = models.resnet50(pretrained=True)
input = torch.rand((1,3,224,224))
model = torch.compile(model, backend='openvino')
pred = model(input)
```

两行代码



Automatic1111 Stable Diffusion Web UI



https://github.com/openvinotoolkit/stable-diffusion-webui

Demo

OpenVINO™ 2023.1

更容易进行AI部署和加速

- 更简单的工作流程
- 更好的 PyTorch 兼容性
- **针对大模型的优化**
- 最新的 notebook 示例

Optimum-Intel

支持 LLM 任务无缝切换

```
- from transformers import AutoModelForCausalLM
+ from optimum.intel.openvino import OVModelForCausalLM

- model = AutoModelForCausalLM.from_pretrained(model_id)
+ ov_model = OVModelForCausalLM.from_pretrained(model_id)

generate_ids = ov_model.generate(input_ids)
```

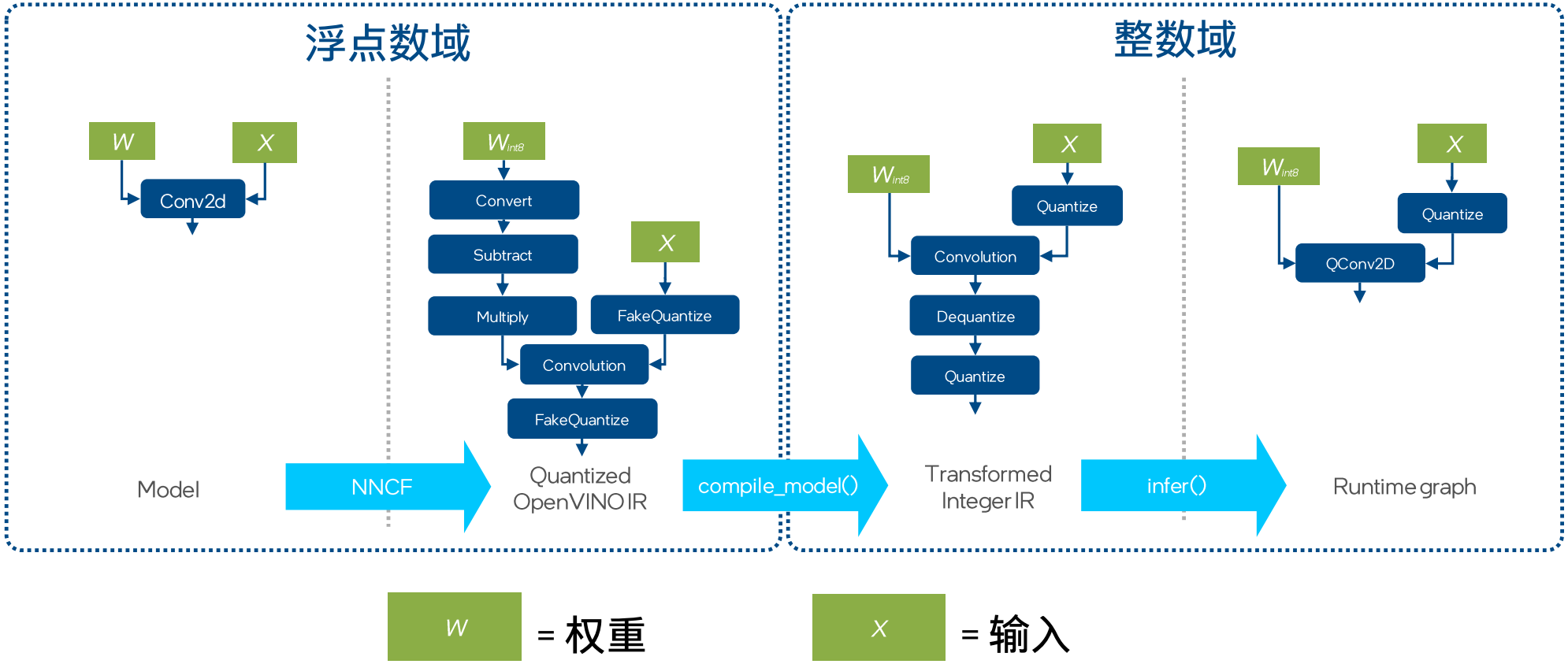
- bart,
- blenderbot,
- blenderbot-small,
- bloom,
- codegen,
- gpt2,
- gpt_neo,
- gpt_neox,
- llama,
- marian,
- opt,
- pegasus,
- ...

通过 LangChain 部署 LLM

集成 Optimum-intel 推理后端

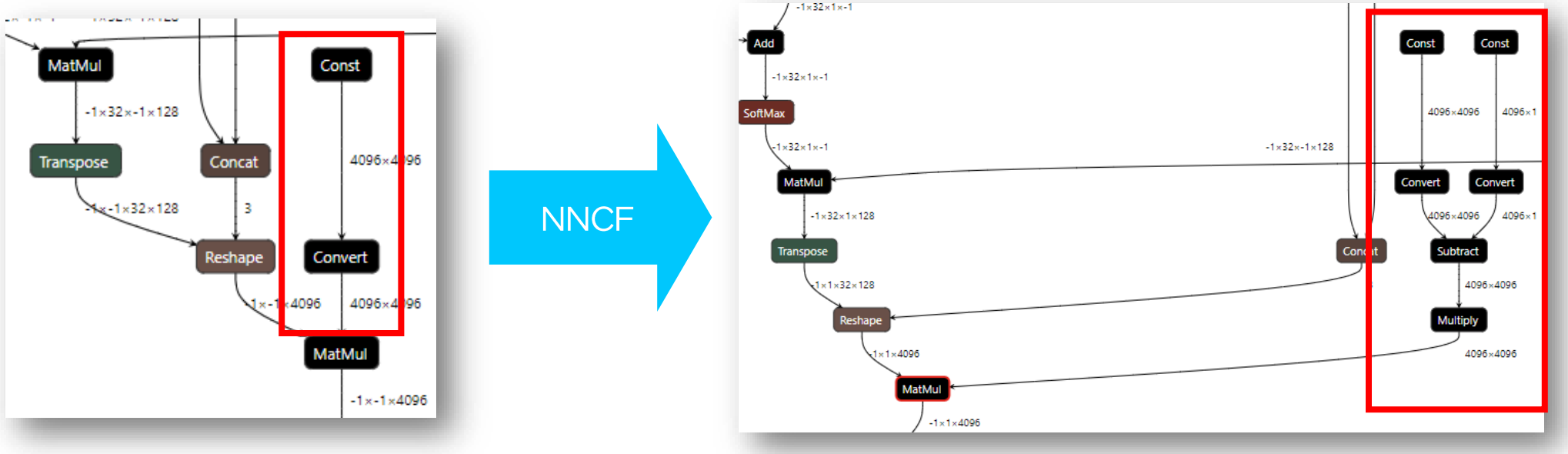
```
from langchain.llms import HuggingFacePipeline
from transformers import pipeline
- from transformers import AutoModelForCausalLM
+ from optimum.intel.openvino import OVModelForCausalLM

- model = AutoModelForCausalLM.from_pretrained(model_id)
+ ov_model = OVModelForCausalLM.from_pretrained(model_id)

pipe = pipeline("text-generation", model=ov_model, tokenizer=tokenizer, max_new_tokens=128,
pad_token_id=tokenizer.eos_token_id)

hf = HuggingFacePipeline(pipeline=pipe)

llm_chain = LLMChain(prompt=prompt, llm=hf)

output = llm_chain.run(question)
```

模型量化的阶段（OpenVINO + NNCF）



权重压缩 Weights Compression

https://github.com/openvinotoolkit/nncf/blob/develop/docs/compression_algorithms/CompressWeights.md




```
from nncf import compress_weights
compressed_model = compress_weights(model)
```

权重压缩 Weights Compression

https://github.com/openvinotoolkit/nncf/blob/develop/docs/compression_algorithms/CompressWeights.md

Model	Size (GB) Reduction
llama-2-7b-chat	25 → 6
open-llama-3b	13 → 3
dolly-v2-12b	44 → 11
gpt-neox-20b	77 → 19
llama-7b	25 → 6
gpt-j-6b	23 → 6

CPU Xeon Gold 6338

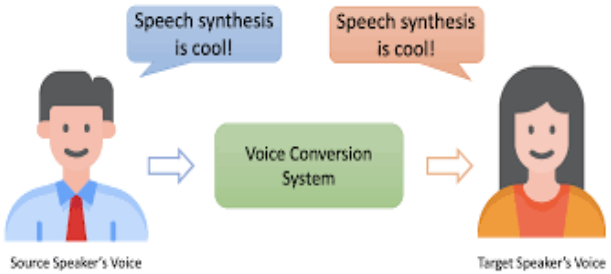
OpenVINO™ 2023.1

更容易进行AI部署和加速

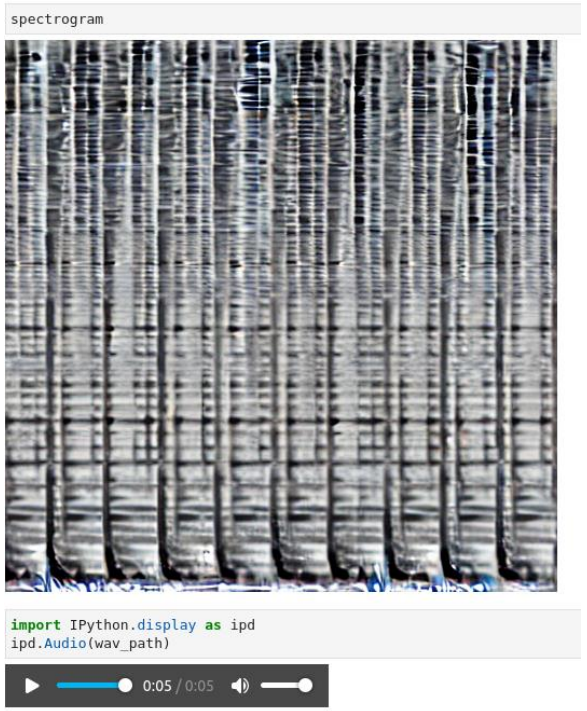
- 更简单的工作流程
- 更好的 PyTorch 兼容性
- 针对大模型的优化
- 最新的 notebook 示例

音频生成

FreeVC: text-free voice conversion



Riffusion: Text-to-audio spectrogram

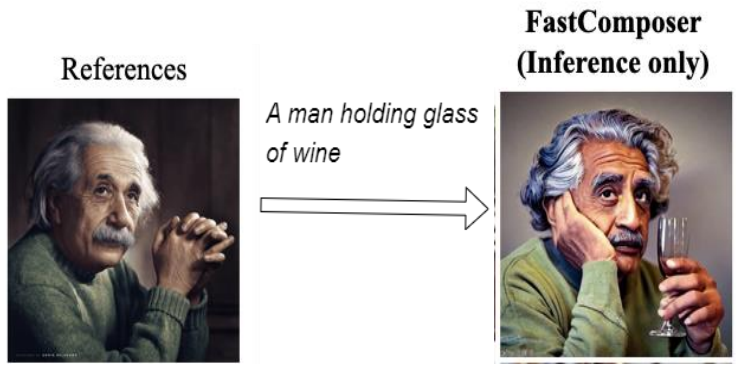


MusicGen: Text-to-Music

图片生成

Stable Diffusion XL

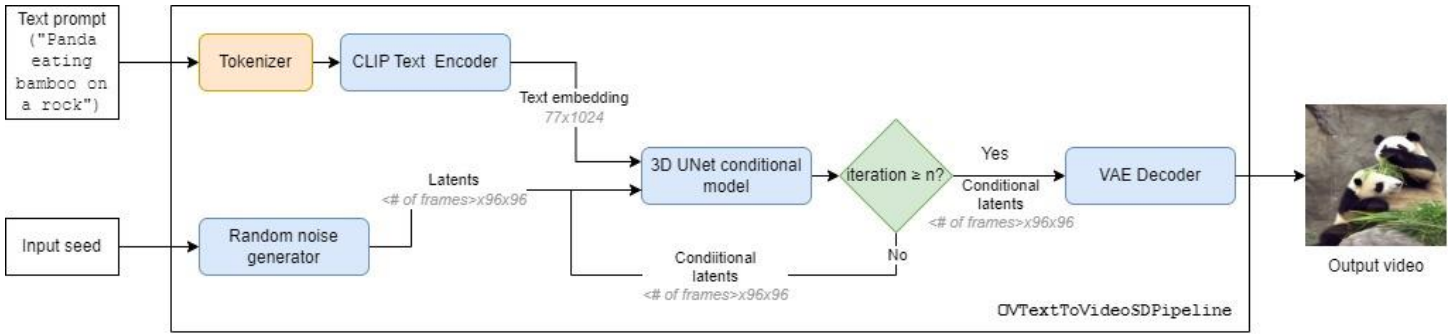
FastComposer: generation personalized images without model fine-tuning



TinySD

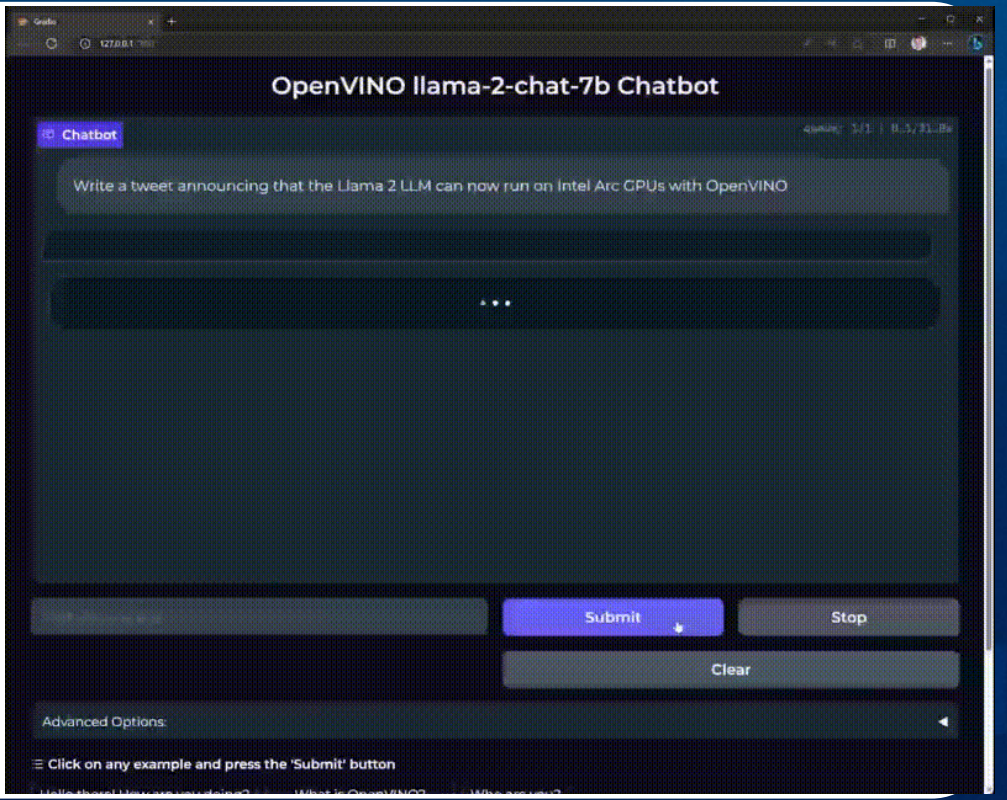
视频生成

ZeroScope: Text-to-Video generation



大语言模型（LLM）聊天机器人

- 支持多个 LLM，包括 LLAMA2
- 可以在 CPU 以及英特尔 GPU 上运行推理
- 展示了是用 INT8 权重压缩的优势



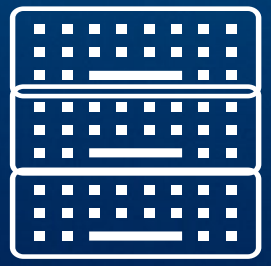
Demo

Hybrid AI 介绍

边缘 AI 无处不在

*Source: Grand View Research, April 2023

发挥边缘的独特优势



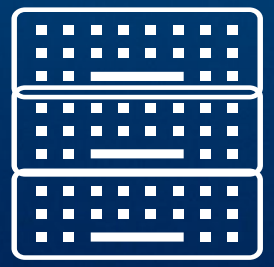
边缘

- 实时数据处理
- 更广阔的触达
- 数据主权
- 成本效益

云端

- 海量数据，无限按需计算
- 集中化

Hybrid AI = 边缘 ⟷ 云端



边缘

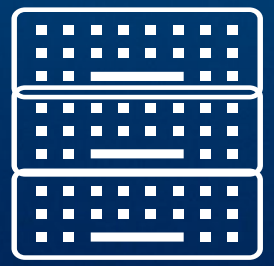
- 实时数据处理
- 更广阔的触达
- 数据主权
- 成本效益

Hybrid AI指使用边缘或云中的可用/目标系统资源和加速器自动处理AI工作负载。

云端

- 海量数据，无限按需计算
- 集中化

Hybrid AI 面临的挑战



边缘

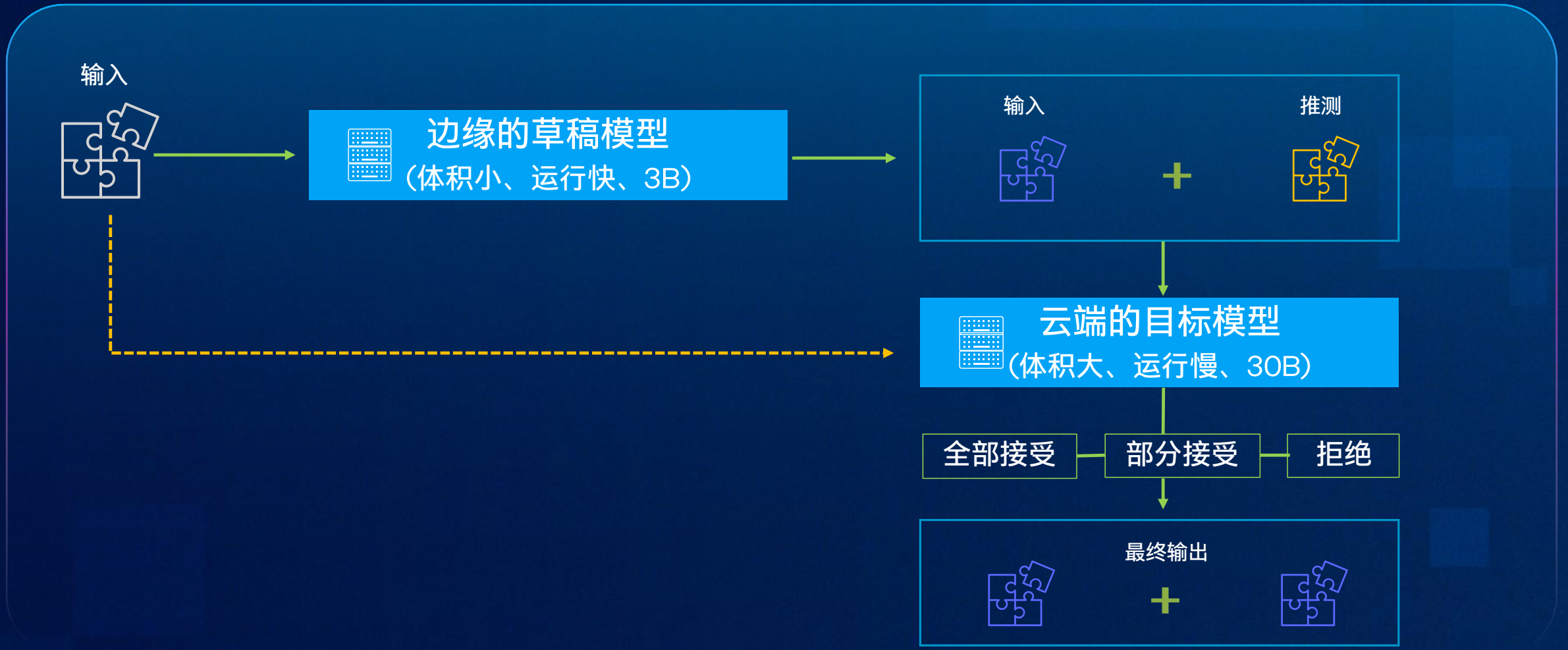
- 实时数据处理
- 更广阔的触达
- 数据主权
- 成本效益

时延
带宽和网络阻塞
数据隐私和安全
可扩展性
成本和能源效率

云端

- 海量数据，无限按需计算
- 集中化

Hybrid AI 举例：推测性采样



使用 OpenVINO™ 在 Hybrid AI 边缘端

使用 Intel 产品打造优化的端到端 Hybrid AI 解决方案

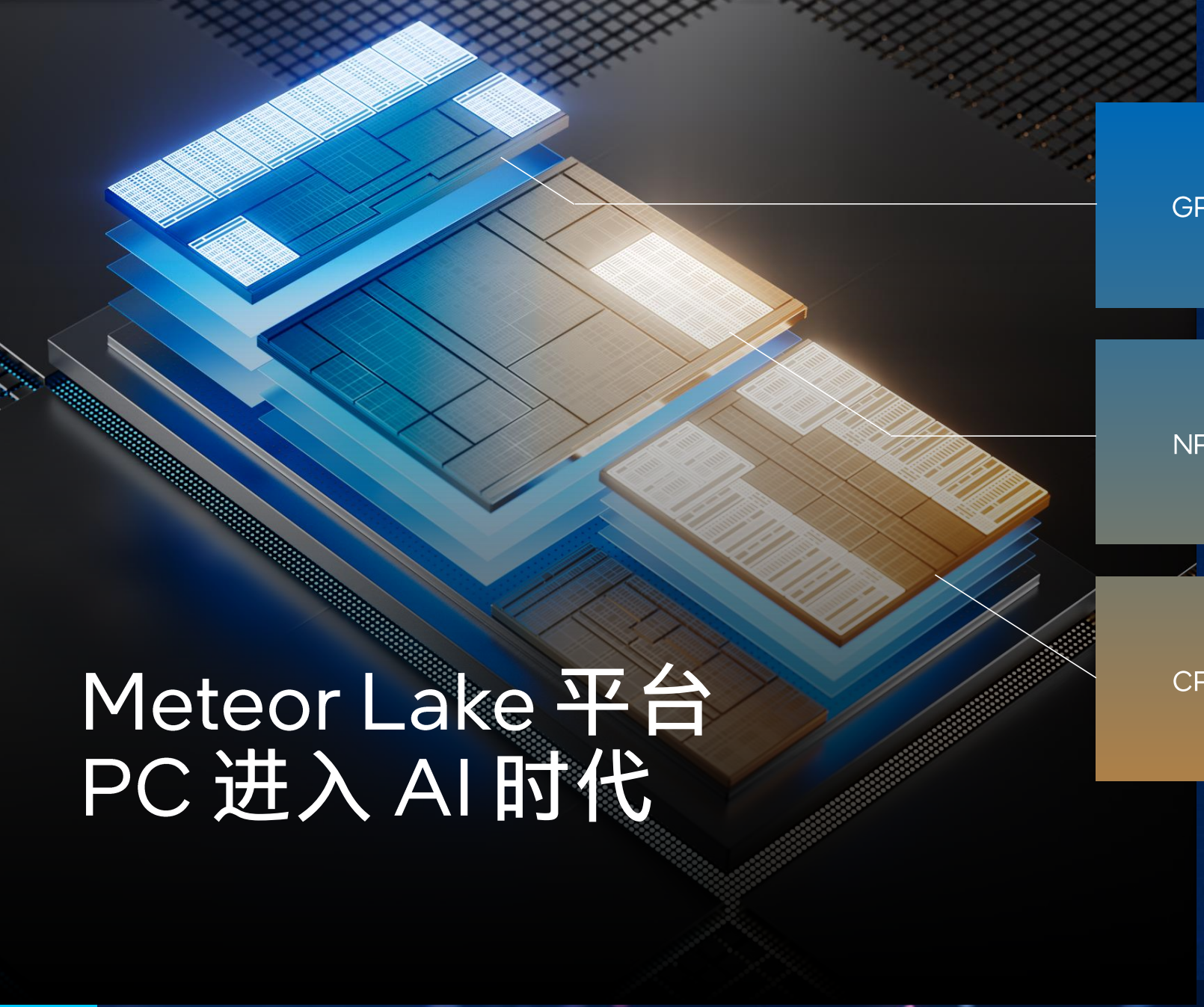
全系列硬件选项
Intel 边缘AI芯片

开放的，开发者友好的软件

OpenVINO™

1 oneAPI Powered by oneAPI

伙伴生态系统



Meteor Lake 平台
PC 进入 AI 时代

GPU

并行计算和高吞吐

通过媒体编解码器、三维图形加速器和渲染器实现 AI 推理加速

NPU

低功耗高能效人工智能加速引擎

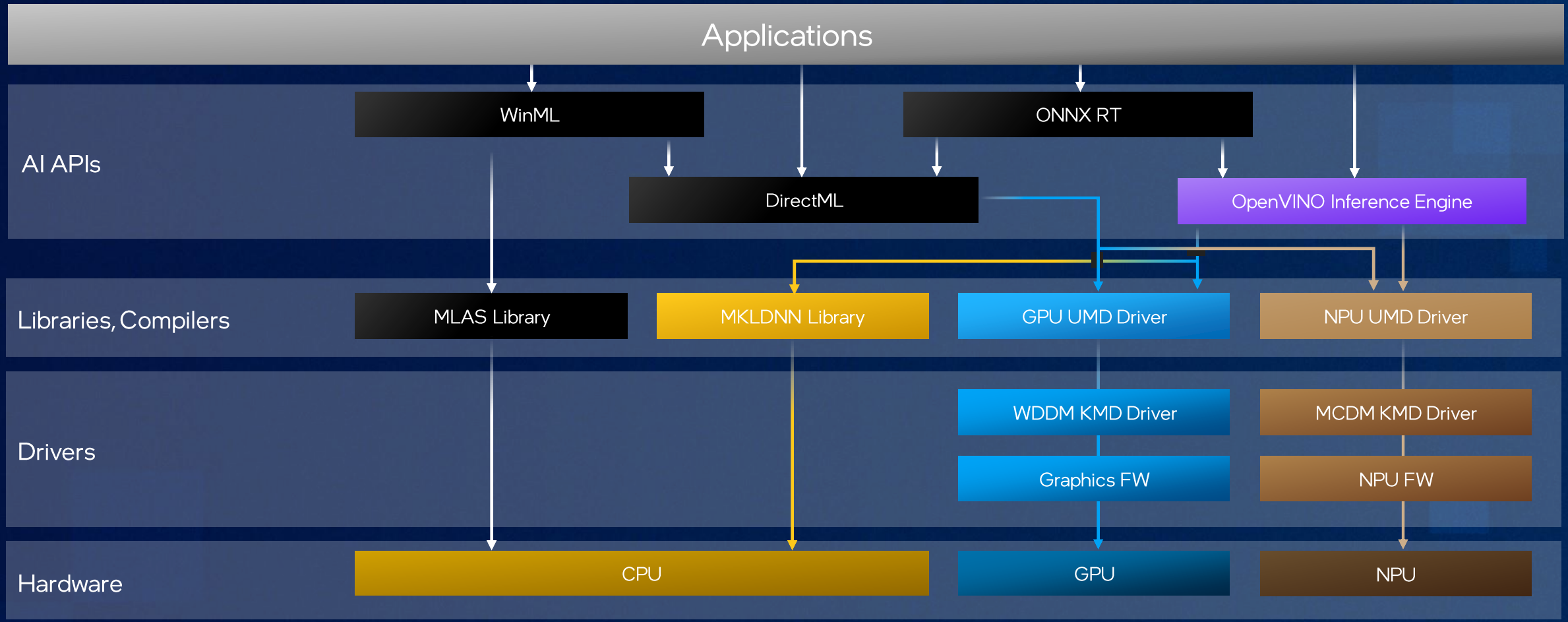
为持续AI应用场景加速，并实现AI算力从CPU和GPU脱机运行，实现低功耗

CPU

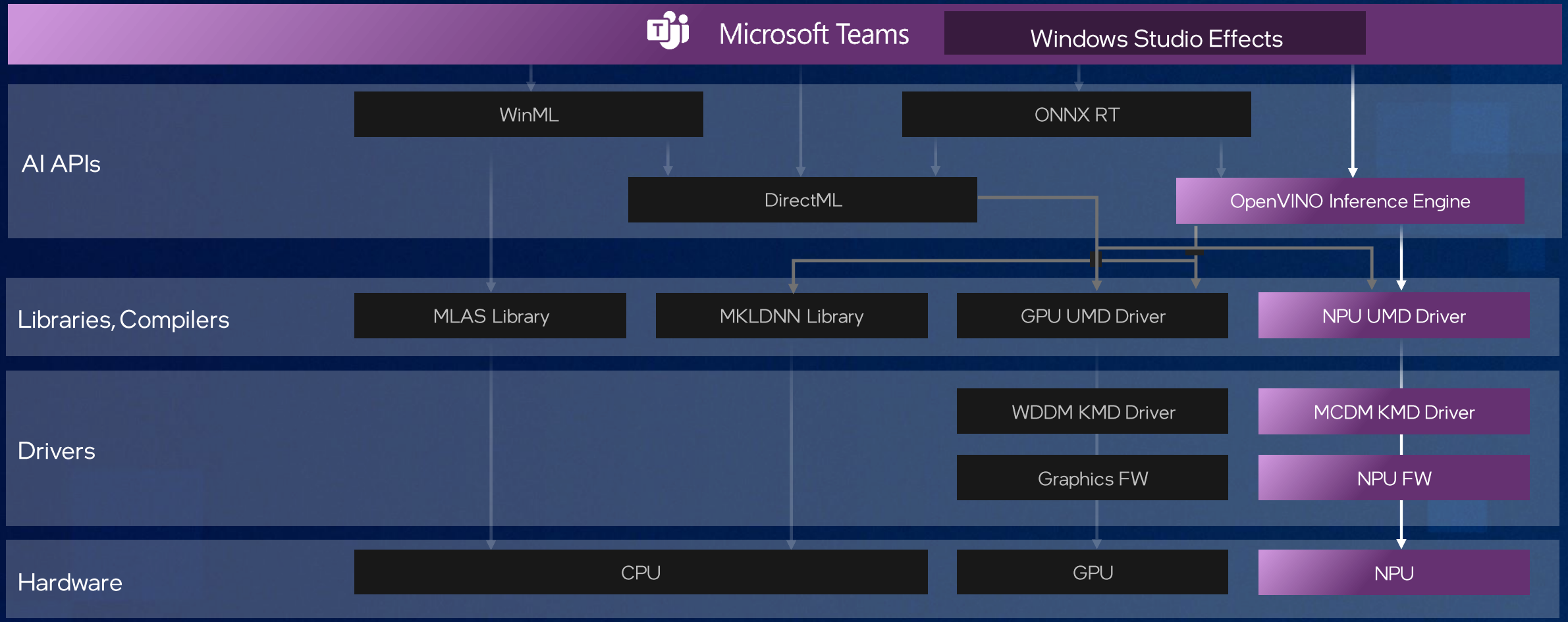
快速反应和通用性

为轻负载AI推理加速，具有低延时的特性

PC AI 软件架构


Applications
AI APIs
WinML
ONNX RT
DirectML
OpenVINO Inference Engine
Libraries, Compilers
MLAS Library
MKLDNN Library
GPU UMD Driver
NPU UMD Driver
Drivers
WDDM KMD Driver
MCDM KMD Driver
Graphics FW
NPU FW
Hardware
CPU
GPU
NPU

PC AI 软件架构实例


Microsoft Teams
Windows Studio Effects
AI APIs
WinML
ONNX RT
DirectML
OpenVINO Inference Engine
Libraries, Compilers
MLAS Library
MKLDNN Library
GPU UMD Driver
NPU UMD Driver
Drivers
WDDM KMD Driver
MCDM KMD Driver
Graphics FW
NPU FW
Hardware
CPU
GPU
NPU