White Paper

Computer Vision/Deep Learning
OpenVINO™ Integration with TensorFlow

Developer Guide: OpenVINO™ Integration with TensorFlow

Deploying and Optimizing TensorFlow Applications on Intel® Hardware

Two lines of code can convert and optimize your TensorFlow models to run on Intel® architecture

OpenVINO™ integration with TensorFlow makes it simple to convert and optimize TensorFlow models to run on Intel® hardware.

Here's all you need to do to get started.

Step 1: Download and install OpenVINO integration with TensorFlow

The open source integration is available on GitHub.

Python wheel packages for installation are available via PyPi.

The OpenVINO integration with TensorFlow package comes with the latest prebuilt OpenVINO™ libraries. You do not have to install OpenVINO separately. Consult the interactive installation table for a menu of installation options. The table will help you configure your installation process.

View the latest system requirements for the OpenVINO integration with TensorFlow ›

See the hardware supported by the most recent release ›

Installation instructions

You can view the installation documentation for your operating system online:

- **Windows ›**
- **Linux ›**
- **MacOS ›**

To leverage Intel® Vision Accelerator Design with Movidius™ (Intel® VAD-M) for inference, install OpenVINO integration with TensorFlow alongside the Intel® Distribution of OpenVINO™ toolkit. You can check requirements for your needs using this online tool.

For more details on installation, please refer to INSTALL.md, and for build-from-source options, please refer to BUILD.md.

Step 2: Add these two lines to your Python code

Once you've installed OpenVINO integration with TensorFlow, you can use TensorFlow to run inference using a trained model. All you need to do is add these two lines of code to your Python code:

```
import openvino_tensorflow
openvino_tensorflow.set_backend('<backend_name>')
```

OpenVINO™ integration with TensorFlow: Supported models

OpenVINO™ integration with TensorFlow supports many widely used models, including EfficientNet, which is used for image classification; MobileNet, which is used for object detection; and BERT, which is used for natural language processing.

Click here to see the full list of supported models.

Supported back ends include 'CPU', 'GPU', 'GPU_FP16', 'MYRIAD', and 'VAD-M'.

For the best results, we suggest enabling oneAPI Deep Neural Network Library (oneDNN) by setting the environment variable `TF _ ENABLE _ ONEDNN _ OPTS=1.`

We also recommend enabling the following variable:

```
os.environ["OPENVINO _ TF _ CONVERT _ VARIABLES _ TO _ CONSTANTS"] = "1"
OPENVINO _ TF _ CONVERT _ VARIABLES _ TO _ CONSTANTS
```

This variable is disabled by default, freezing variables from TensorFlow's ReadVariableOp as constants during the graph translation phase. We highly recommend that you enable it to ensure optimal inference latencies on eagerly executed models. Disable it when model weights are modified after loading the model for inference.

By default, the Intel® CPU is used to run inference. However, you can change the default option to either an Intel® integrated GPU or Intel® VPU for AI inferencing. Invoke the following function to change the hardware on which inferencing is done:

```
openvino _ tensorflow.set _ backend('<backend _ name>')
```

To determine what processing units are available on your system for inference, use this function:

```
openvino _ tensorflow.list _ backends()
```

For more API calls and environment variables, see USAGE.md.

(Note: If a CUDA-capable device is present in the system, then set the environment variable `CUDA _ VISIBLE _ DEVICES` to -1.)

Step 3: Verify your installation

To see if OpenVINO integration with TensorFlow is properly installed, run:

```
python3 -c "import tensorflow as tf; print('TensorFlow version: ',tf. _ _ version _ _ );\
            import openvino _ tensorflow; print(openvino _ tensorflow. _ _ version _ _ )"
```

For version 2.0.0, this should produce the following output:

```
TensorFlow version: 2.8.0
OpenVINO integration with TensorFlow version: b'2.0.0'
OpenVINO version used for this build: b'2022.1.0'
TensorFlow version used for this build: v2.8.0
CXX11 _ ABI flag used for this build: 0
```

Click here to view the latest instructions for verifying your installation ›

The result

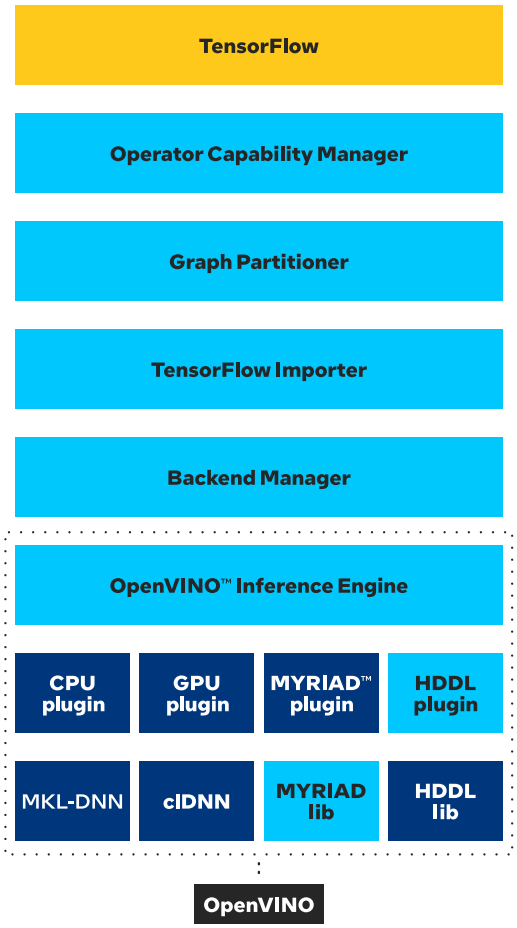
You can now deploy optimized TensorFlow models on Intel hardware.

Next steps

See examples	Try it for yourself	Download the integration	Read on
To see what you can do with OpenVINO™ integration with TensorFlow, explore the demos located in the examples directory on GitHub.	Sample tutorials are available on Intel® DevCloud. You can run them on Intel DevCloud nodes and compare the results of OpenVINO integration with TensorFlow to native TensorFlow and OpenVINO.	GitHub PyPi	Learn more about our integration and review performance testing results.

TensorFlow integration: Under the hood

OpenVINO integration with TensorFlow provides accelerated performance by smartly partitioning TensorFlow graphs into multiple subgraphs that are then dispatched to either the TensorFlow runtime or the OpenVINO runtime for optimal accelerated inferencing. The results are assembled to provide the final inference output.



Process

A TensorFlow graph is passed to the Operator Capability Manager (OCM) module, which traverses the graph and marks all supported operators. The Graph Partitioner module uses this information to efficiently divide the graph into supported and unsupported subgraphs. Supported subgraphs are translated to OpenVINO intermediate representation in-line and executed on OpenVINO back ends. Unsupported subgraphs fall back to native TensorFlow.

OpenVINO integration with TensorFlow modules

- **Operator Capability Manager:** This module implements checks on TensorFlow operators to determine which abstraction layers go to OpenVINO integration back ends and which layers should fall back on stock TensorFlow runtime.
- **Graph Partitioner:** This module examines the nodes that OCM marked for clustering and assigns them to clusters. Some clusters are dropped after further analysis. Each cluster of operators is then encapsulated into a custom operator that is executed in the OpenVINO runtime.
- **TensorFlow Importer:** This module translates TensorFlow operators to OpenVINO integration operators and creates nGraph functions wrapped into a convolutional neural network to run on the toolkit back end.
- **Backend Manager:** This module creates a back end to run the CNN. There are two types of back ends: basic back end and VAD-M back end. The basic back end supports CPU, iGPU, and MYRIAD VPU. The VAD-M back end is used for Intel Vision Accelerator Design with eight VPUs (referred to as VAD-M or HDDL).

Testing results

Extreme Vision notebooks/IDE: Unlocking 10x performance gains with OpenVINO integration with TensorFlow

Extreme Vision, a computer vision technology and cloud services provider, has tested the OpenVINO TensorFlow integration on their IDE platform. Their results show a 10x performance gain.

View detailed testing results ›

Terra: Accelerating machine learning tool performance by 21 percent with OpenVINO integration with TensorFlow

Terra, a scalable cloud platform provider for biomedical research, tested the performance of their deep learning convolutional neural network tool using the OpenVINO TensorFlow integration. The results showed a 21 percent increase in operation speed—reducing runtime from 8.67 minutes to 7.6 minutes with only a few lines of code.

See the details ›

More Intel® edge software development tools

Intel Distribution of OpenVINO toolkit

The Intel Distribution of OpenVINO toolkit converts models to run on any Intel® architecture and optimizes them to meet your performance expectations. OpenVINO integration with TensorFlow is included in our distribution of the toolkit.

Find out more and download the toolkit ›

The development advantages of the Intel Distribution of OpenVINO toolkit

Our free resource helps you deploy inference models with maximum flexibility and minimal compromises.

High performance, deep learning

Convert and optimize models for high deep learning inference performance on Intel hardware.

Streamlined development

Facilitate a smoother development process using the included inference tools for low-precision optimization and media processing, computer vision libraries, and preoptimized kernels.

Write once, deploy anywhere

Deploy across any mix of host processors and accelerators, including CPUs, GPUs, and VPUs. The same model and inference engine can run on edge devices, servers, and in the cloud.

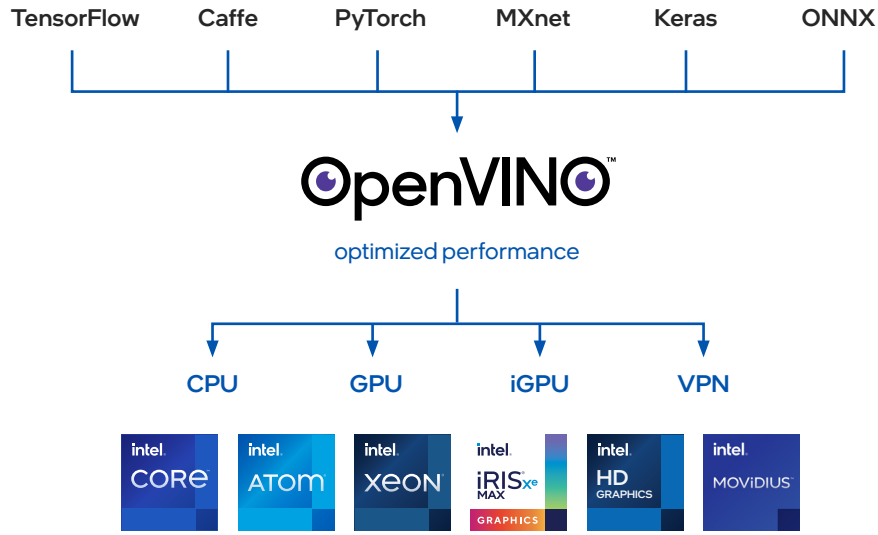


Figure 1: The Intel® Distribution of OpenVINO™ toolkit enables high-performance inference with write-once, deploy-anywhere efficiency. While this white paper focuses on TensorFlow applications, the OpenVINO™ toolkit can convert and optimize models from a variety of frameworks to work across Intel® architectures.

Intel® DevCloud for the Edge

AI developers can learn, prototype, test, and run workloads for free on a cluster of the latest Intel® hardware and software using Intel Dev Cloud for the Edge. It's an OpenVINO sandbox that allows you to prototype and test deep learning AI on actual Intel hardware. Preinstalled Intel-optimized frameworks, tools, and libraries give you what you need to fast-track your learning and project prototyping.

What's included:

- Intel CPUs
- Accelerators
- Intel® FPGAs
- Intel® GPUs
- Toolkits, tools, and libraries

Try the TensorFlow integration for yourself on Intel DevCloud for the Edge ›

Intel® Edge Software Hub

Download prevalidated software to learn, develop, and test your deep learning solutions for the edge. The Intel Edge Software Hub enables you to experiment, test, and create—all with less prework. Many of the ready-to-use tools on the hub are built using the OpenVINO toolkit.

Our preoptimized applications make it simple to optimize edge solutions, including computer vision and deep learning applications, for Intel architecture.

Set your objectives and ramp up quickly. Choose your architecture type, and then select from Intel, third-party, or open source software. Use containers, run multiple workloads on a single converged edge system, and manage data flow between sensors and applications. Whether building from scratch or customizing a ready-made reference implementation, we offer helpful resources to accelerate your development.

See what's available on the Intel Edge Software Hub ›

Intel® oneAPI toolkits

These free resources contain optimized compilers, libraries, frameworks, and analysis tools purpose-built for developers who perform similar tasks. They include implementations of the oneAPI specification along with complementary tools to develop and deploy applications and solutions across Intel® CPU and XPU architectures. The Intel Distribution of OpenVINO toolkit is powered by oneAPI.

We offer a range of toolkits to help you meet your development goals, including:

- Intel® oneAPI Base Toolkit: Develop performant, data-centric applications across Intel CPUs, GPUs, and FPGAs with this foundational toolset.
- Intel® oneAPI IoT Toolkit: Fast-track development of applications and solutions that run at the network's edge.
- Intel® oneAPI AI Analytics Toolkit: Accelerate end-to-end data science and machine learning pipelines using Python tools and frameworks.

See all available toolkits ›

Native OpenVINO toolkit

OpenVINO integration with TensorFlow offers a simple path to optimization. But for maximum performance, efficiency, tooling customization, and hardware control, we recommend the full Intel Distribution of OpenVINO toolkit. Those already using the Intel Distribution of OpenVINO toolkit should continue to leverage optimizations natively available on OpenVINO.

Learn more about the native OpenVINO toolkit ›

Get started with OpenVINO integration with TensorFlow today

Optimized cross-architecture performance for TensorFlow AI applications is easier to achieve than you think. See for yourself how simple it can be:

Take our free introductory course ›

Download OpenVINO integration with TensorFlow via GitHub ›

Download OpenVINO integration with TensorFlow via PyPi ›

Additional resources

GitHub home page ›

Documentation ›

Introduction to OpenVINO integration with TensorFlow ›

Integration benefits ›

Solution brief ›

Prebuilt images on Docker Hub and Azure Marketplace ›

TensorFlow serving support ›

FAQ ›

Gitee ›

How to use on Azure ›

How to use on AWS ›

Object detection and classification resources

Sample application ›

Example folder ›

Intel DevCloud ›

Google Colab ›