# Using compute and AI-focused hardware accelerators with WinML

WinML allows you to run any machine learning workload on CPUs and GPUs; optimized GPU support is provided by DirectML and DirectX12. In early Windows 10 20H2 Windows Insider Preview flights, WinML offers a developer preview of support for compute and AI-focused hardware accelerators, including Intel® Movidius™ Vision Processing Units (VPU). WinML uses the same DirectML and DirectX12 stack to support AI-focused accelerators; a hardware vendor writes a new Microsoft Compute Driver Model (MCDM) driver which allows DirectX to communicate with the accelerator, in much the same way that a Windows Display Driver Model (WDDM) driver exposes GPU hardware.

AI-focused hardware accelerators such as the MyriadX VPU are optimized for machine learning and compute workloads, and therefore can offer better performance or better power efficiency (performance / watt) than CPUs and GPUs. AI-focused accelerators are well suited for cases when your application is already fully using the CPU and/or GPU – for example a video game or compute-intensive task – and can benefit from the additional processing power. They also are advantageous if you want to offload a machine learning task to a more efficient processor to save power, for example running a long-running inferencing task in the background.

### Note

Support for compute and AI-focused hardware accelerators is in preview and may be substantially modified before it’s officially released. Microsoft makes no warranties, express or implied, with respect to the information provided here.

## How to access AI-focused accelerators in WinML

In early 20H2 WIP flights, API integration for AI-focused accelerators is in preview and is incomplete.

With CPUs and GPUs, you can have WinML manage the hardware accelerator for you with [LearningModelDeviceKind](https://docs.microsoft.com/en-us/uwp/api/windows.ai.machinelearning.learningmodeldevicekind); alternatively, you can manually select a DXGI adapter in native C++ with [CreateFromD3D12CommandQueue](native-apis/ILearningModelDeviceFactoryNative_CreateFromD3D12CommandQueue.md). For AI-focused accelerators, [LearningModelDeviceKind](https://docs.microsoft.com/en-us/uwp/api/windows.ai.machinelearning.learningmodeldevicekind) is not available. Instead, you should use the native C++ path with [CreateFromD3D12CommandQueue](native-apis/ILearningModelDeviceFactoryNative_CreateFromD3D12CommandQueue.md). In addition, DXGI is only aware of GPUs and does not provide access to AI-focused accelerators. Instead, you should use the preview DXCore API which is the replacement for DXGI for adapter enumeration.

Refer to the following code samples for more information: \* *SqueezeNetObjectDetection\_MCDM\_AdapterSelection*: Demonstrates how to access an MCDM adapter in native C++ and use it to accelerate the SqueezeNet object detection model. \* *DXCore\_WinRTComponent*: Demonstrates how to access an MCDM adapter in C# and other supported UWP languages using a Windows Runtime component.

## See also

* [API reference](https://docs.microsoft.com/uwp/api/windows.ai.machinelearning)
* [Code examples](https://github.com/Microsoft/Windows-Machine-Learning/tree/master)