

TPAC 2021

WebNN ML JS Frameworks Performance

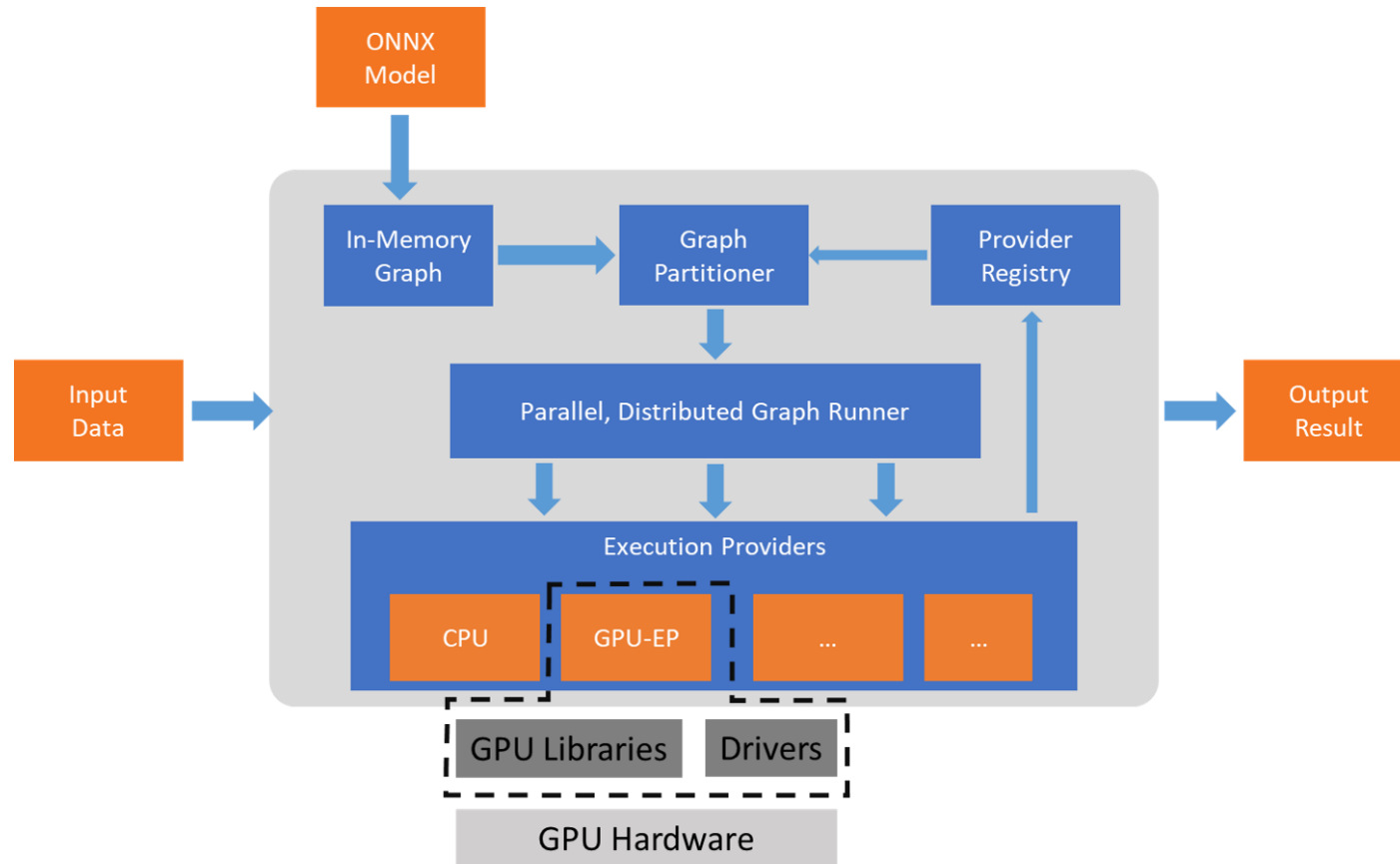
Ningxin Hu (Intel)

Oct 2021

Agenda

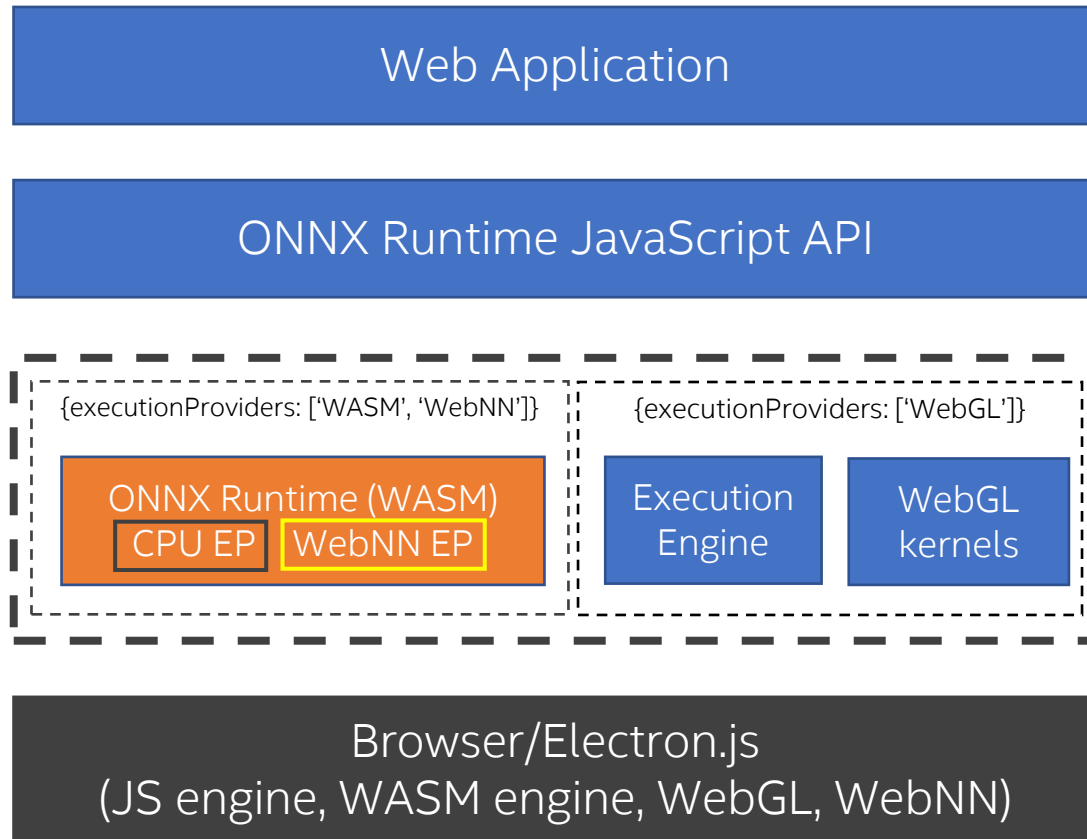
- ONNX Runtime Web
- TensorFlow Lite Web
- OpenCV.js
- Summary

ONNX Runtime Execution Providers

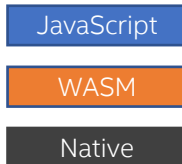


ONNX Runtime works with different hardware acceleration libraries through its extensible [Execution Providers](#) (EP) framework to allocate specific nodes or sub-graphs for execution by the EP library in supported hardware.

WebNN Execution Provider Prototype

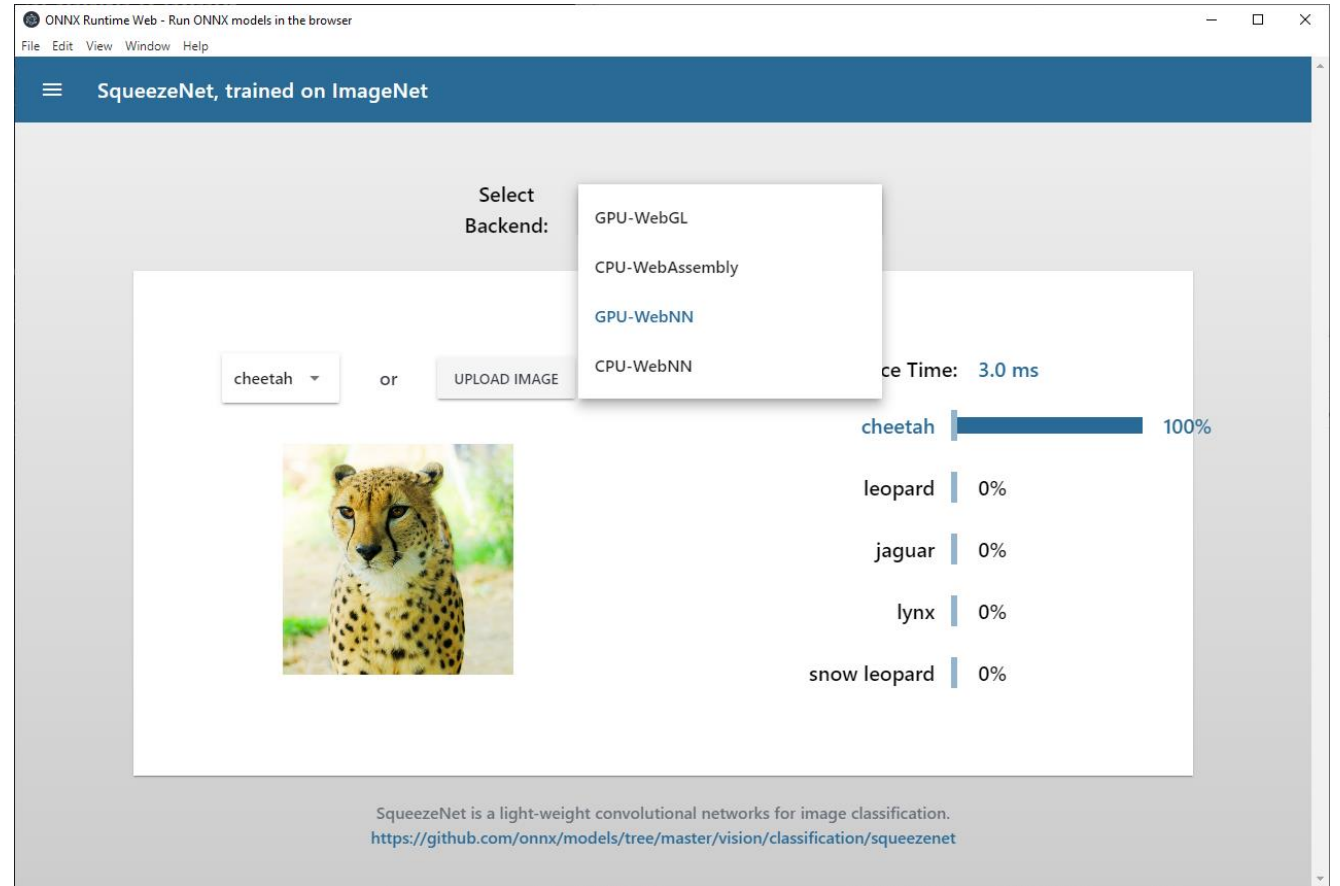


- Based on [ONNX Runtime Web 1.9.0](#)
- Implemented in C++ and uses [webnn.h](#)
- Support 14 ONNX ops
 - Add, Conv, Gemm/Matmul, Poolings, BatchNorm, Relu, LeakyRelu, Clip, Concat and Reshape
- Run CPU EP (Wasm) op if not supported by WebNN EP
 - E.g., Softmax, ImageScalar, Shape, Gather, Unsqueeze
- Compile to Wasm with a [customized Emscripten](#) with WebNN support
 - [library_webnn.js](#) maps C++ to JavaScript call
- Source code available at [github repo](#)



ONNX Runtime Web Demo with WebNN EP

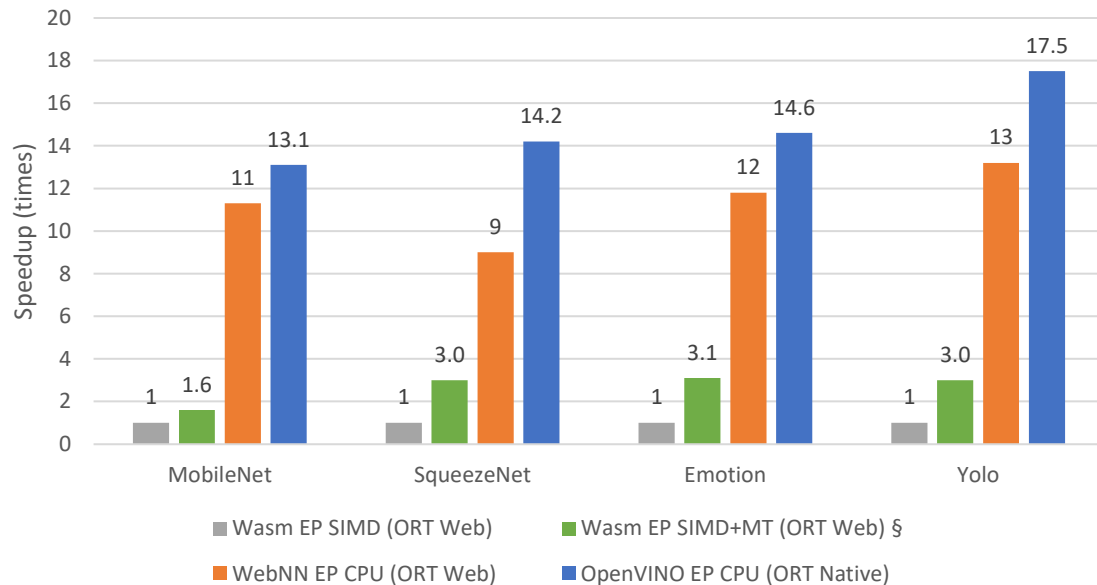
- Based on [ONNX Runtime Web Demo](#).
- Support “GPU-WebNN” and “CPU-WebNN” for all demos
- Access native ML API through [WebNN-native](#) node.js addon when served as an [Electron.js app](#) for Windows
 - DirectML for GPU-WebNN
 - OpenVINO for CPU-WebNN
- Source code available at [github repo](#)



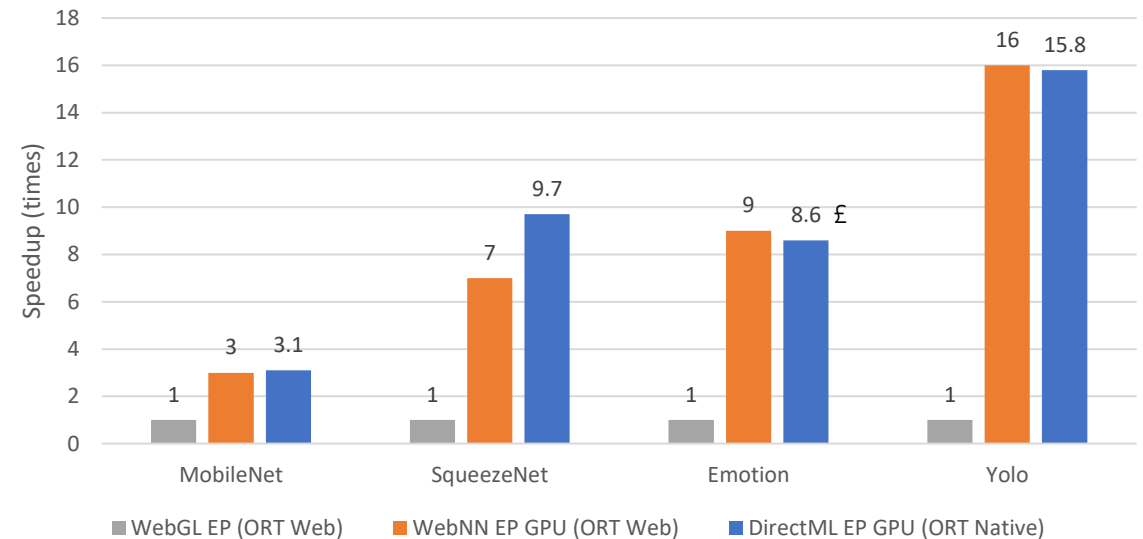
The packaged Electron.js app runs ORT Web demo with WebNN EP

ONNX Runtime Web Demo with WebNN EP Performance*

Inference Latency Speedup on CPU
(higher is better)



Inference Latency Speedup on GPU
(higher is better)

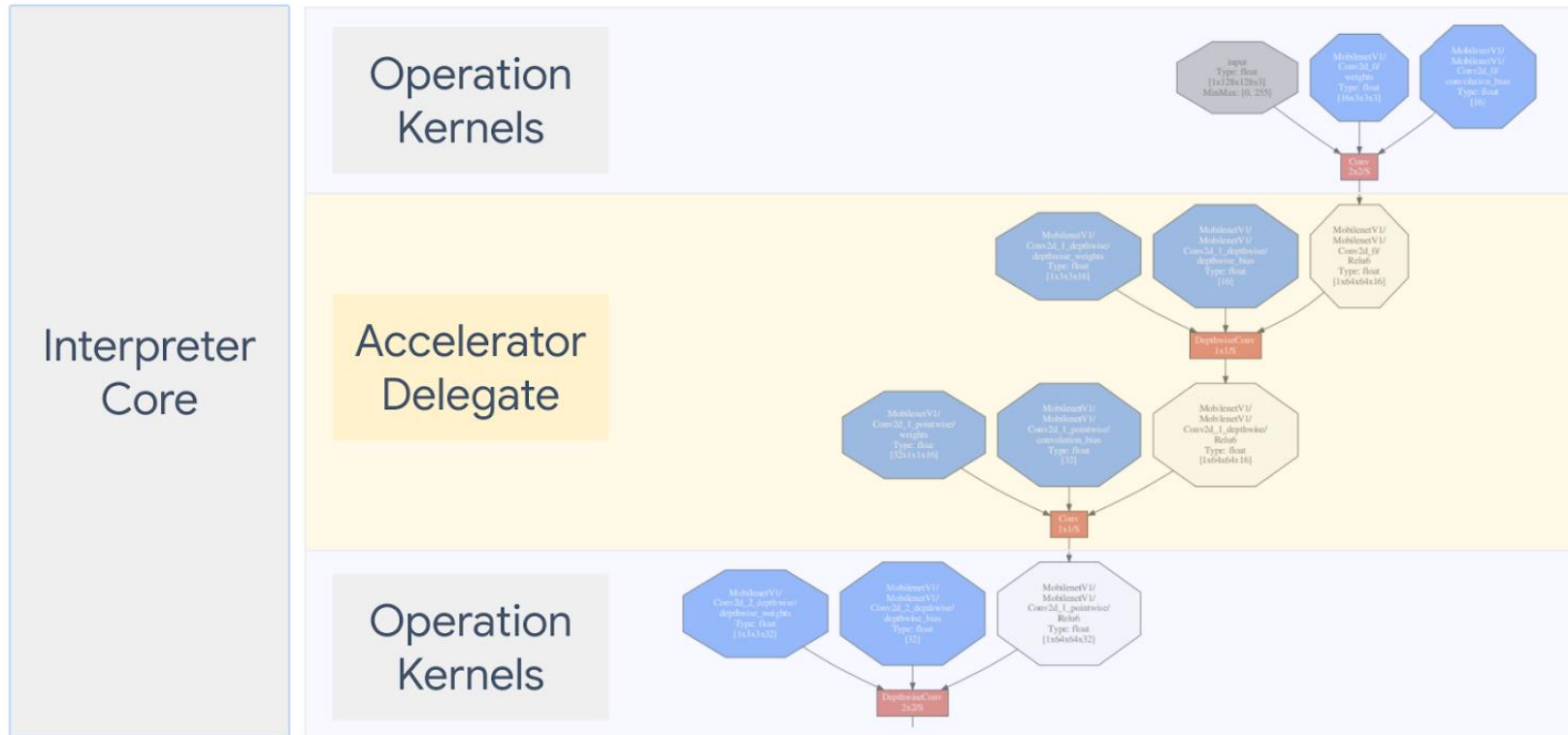


* Tested on a Windows laptop. All models are FP32 precision. See backup for workloads and configurations. Results may vary.

§ Enabling Wasm MT (Multi-Threading/SharedArrayBuffer) requires "[cross-origin isolation](#)".

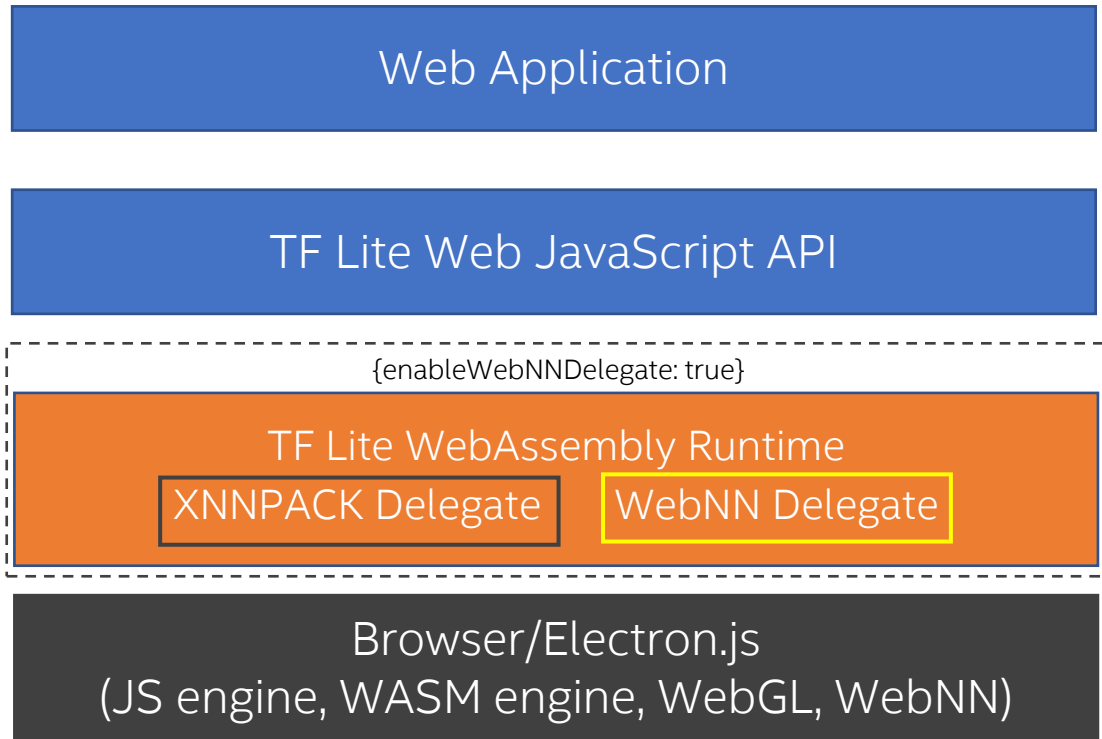
£ Tested with ONNX Runtime benchmark tool (onnxruntime_perf_test)

What is a TensorFlow Lite Delegate?



A TensorFlow Lite [Delegate](#) allows you to run your models (part or whole) on another executor. This mechanism can leverage a variety of on-device accelerators such as the GPU or Edge TPU (Tensor Processing Unit) for inference.

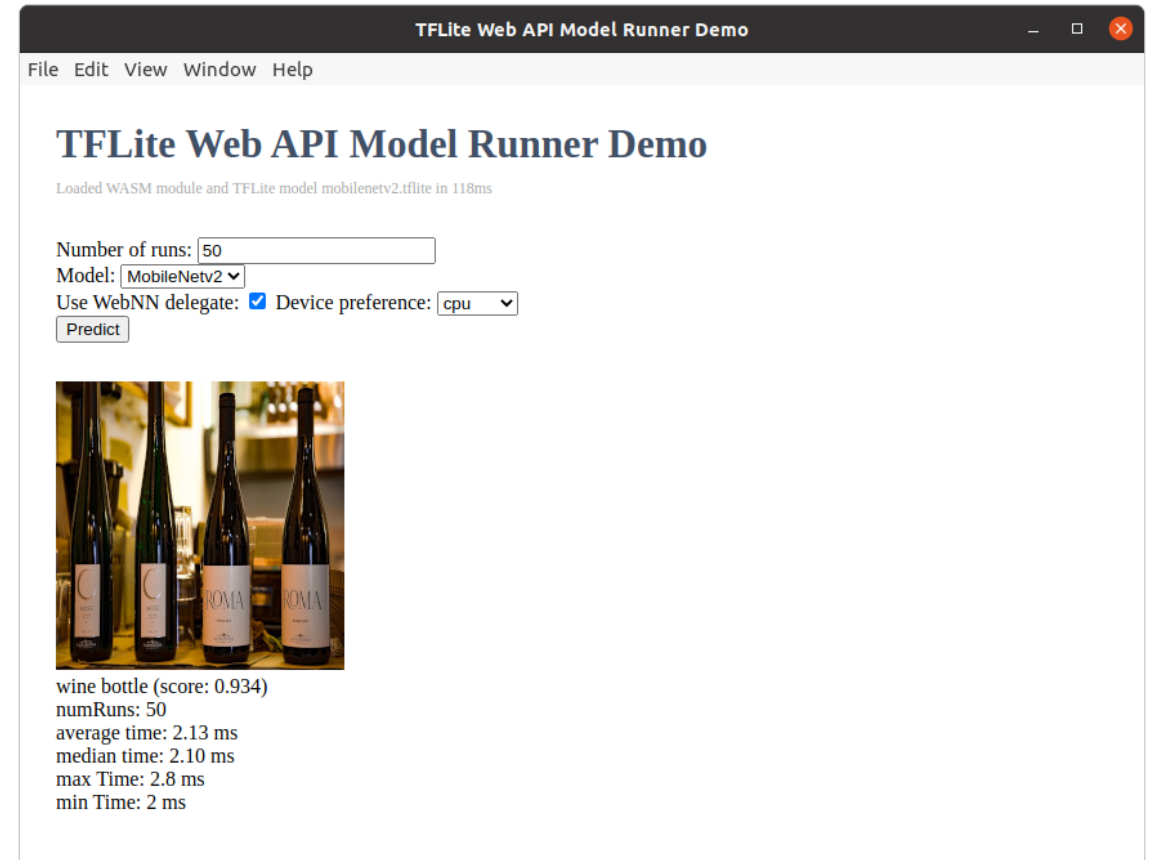
WebNN Delegate Prototype



- Based on [TF Lite support for Web](#)
- Implemented in C++ and uses `webnn.h` (see [RFC](#) for design details)
- Support 10 ops:
 - Add, Mul, Conv2d, DepthwiseConv2d, Poolings, ResizeBilinear, Softmax, Reshape, Concat
- Run Wasm ops if not supported by WebNN delegate
 - e.g., ConvTranspose2d
- Compile to Wasm with a [customized Emscripten](#) with WebNN support
- Source code is available at [github repo](#)

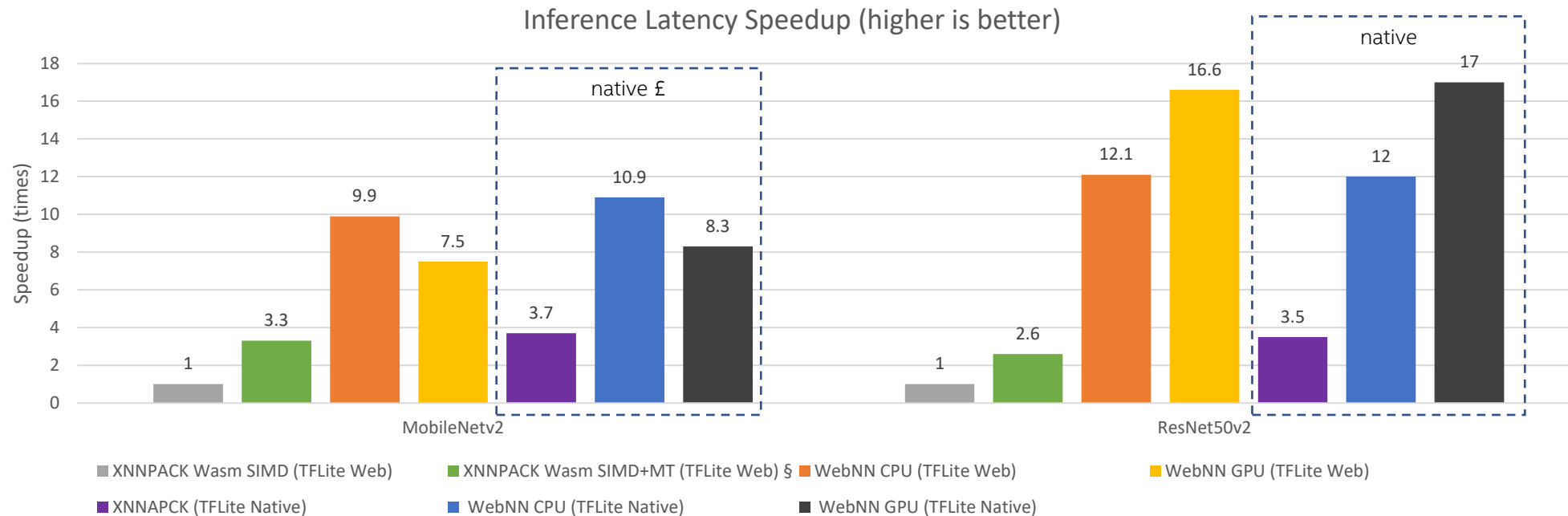
TFLite Web API Demo with WebNN Delegate

- Based on [TFLite Web API Model Runner Demo](#)
- Supports “Use WebNN delegate” for CPU and GPU.
- Access native ML API through [WebNN-native node.js addon](#) when served as an [Electron.js app](#) for Linux:
 - OpenVINO/CPU for WebNN/CPU
 - OpenVINO/GPU for WebNN/GPU
- Source code is available at [github repo](#)



The packaged Electron.js app runs TFLite Web API demo with WebNN delegate

TFLite Web API Demo with WebNN Delegate Performance *

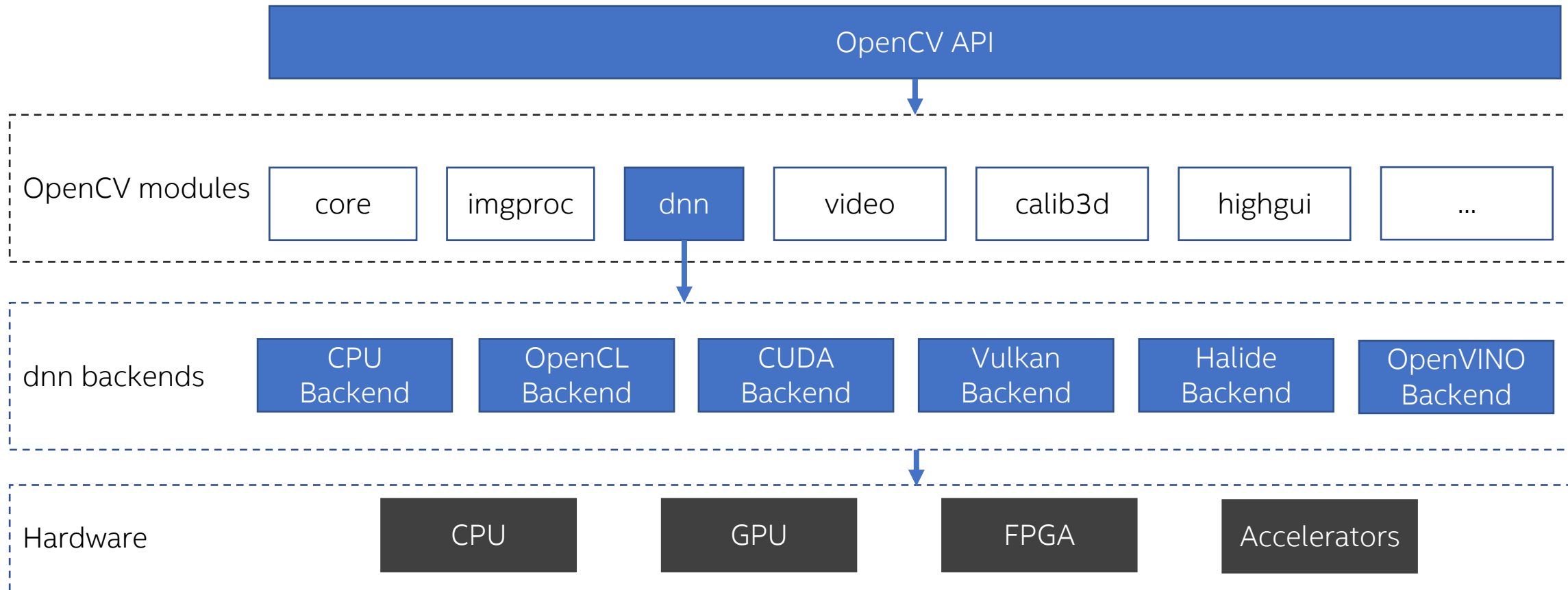


* Tested on a Linux laptop. All models are FP32 precision. See backup for workloads and configurations. Results may vary.

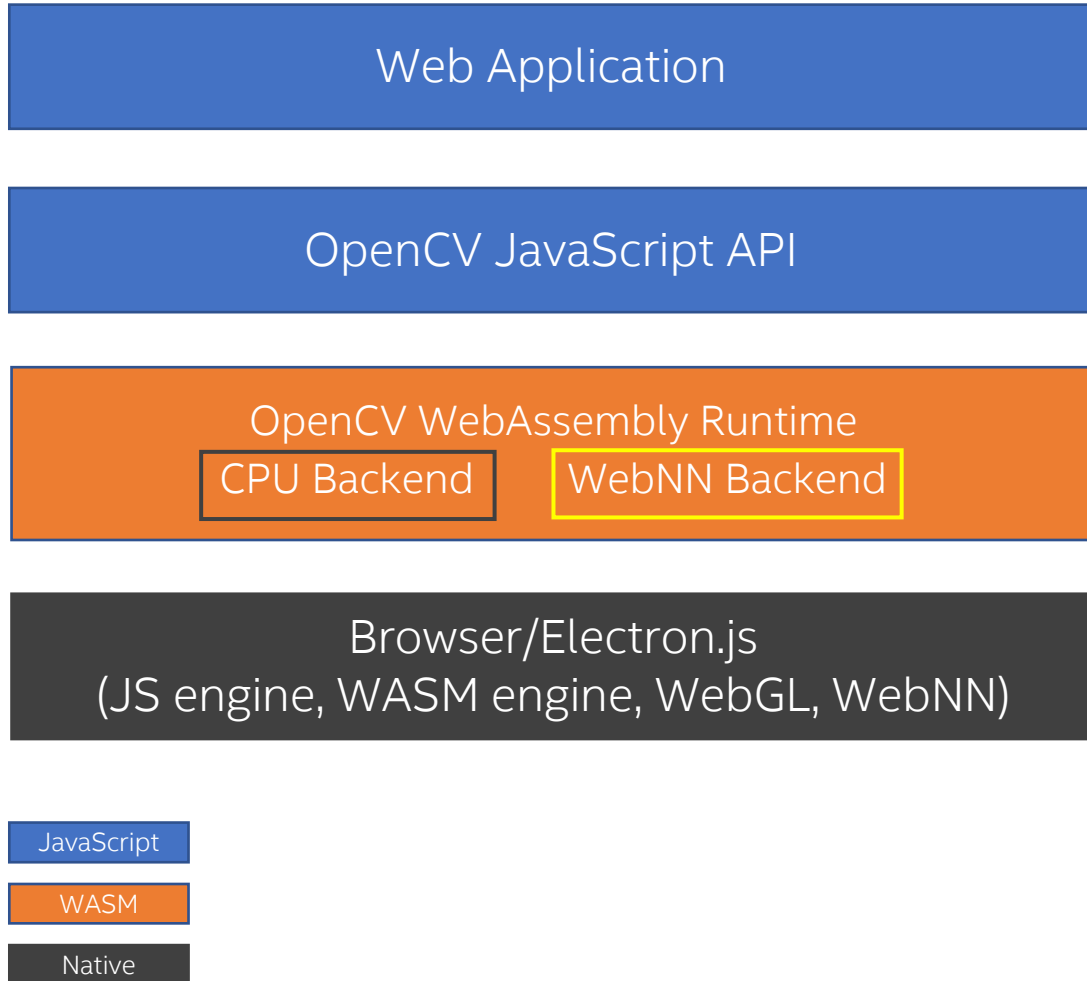
§ Enabling Wasm MT (Multi-Threading/SharedArrayBuffer) requires "[cross-origin isolation](#)".

£ Tested with TFLite benchmark tool (benchmark_model)

OpenCV DNN Module and Backends



OpenCV.js and WebNN backend



- Google Summer of Code [project](#) of OpenCV
- Implemented in C++ and uses webnn.h
- Support 11 layers (ops):
 - BatchNorm, Convolution, FullyConnected, ReLU, ReLU6, Pooling, Reshape, Softmax, Permute, Concat
- Run CPU backend (Wasm) ops if not supported by WebNN backend
 - E.g., LRN
- Compile to Wasm with a [customized Emscripten](#) with WebNN support
- Source code is available at [github pull request](#)

OpenCV.js DNN Examples with WebNN Backend

- Based on [OpenCV.js Image Classification Example](#)
- Use WebNN backend by `net.setPreferableBackend("webnn")`.
- Access native ML API through [WebNN-native node.js addon](#) when served as an Electron.js app for Linux:
 - OpenVINO/CPU for WebNN/CPU
 - OpenVINO/GPU for WebNN/GPU
- Source code is available at [github repo](#)

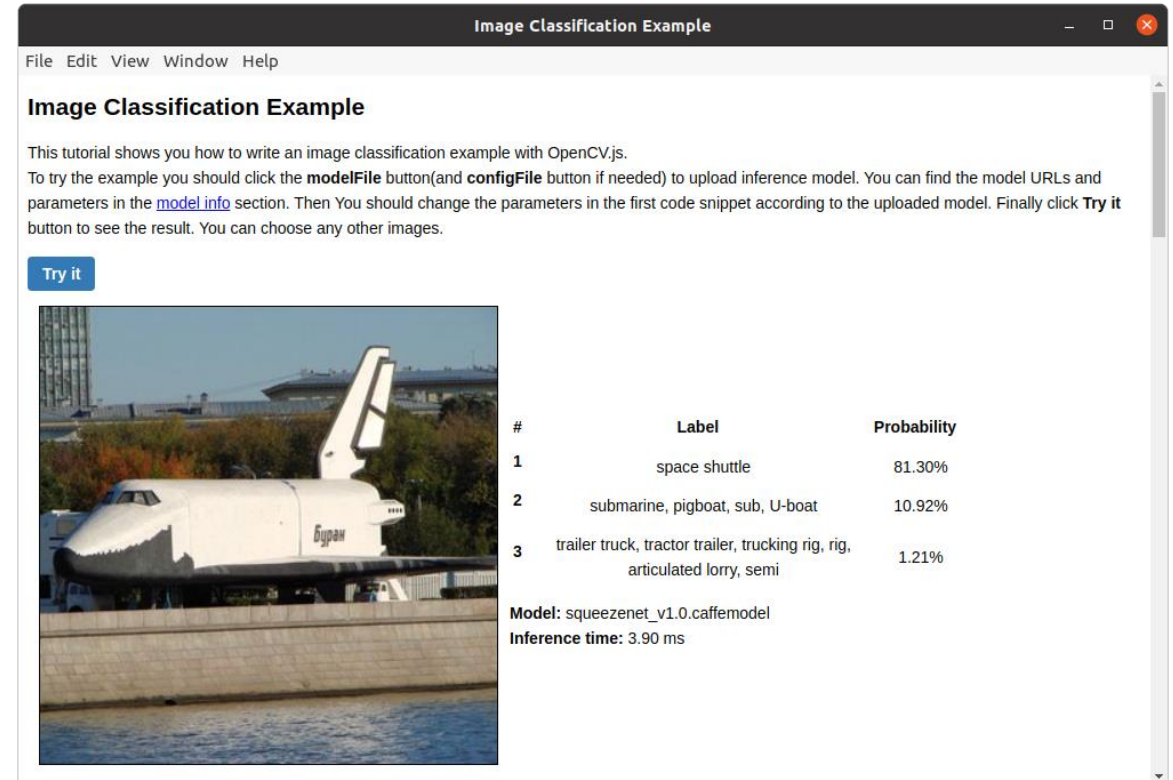



Image Classification Example

This tutorial shows you how to write an image classification example with OpenCV.js.
To try the example you should click the **modelFile** button (and **configFile** button if needed) to upload inference model. You can find the model URLs and parameters in the [model info](#) section. Then you should change the parameters in the first code snippet according to the uploaded model. Finally click **Try it** button to see the result. You can choose any other images.

Try it

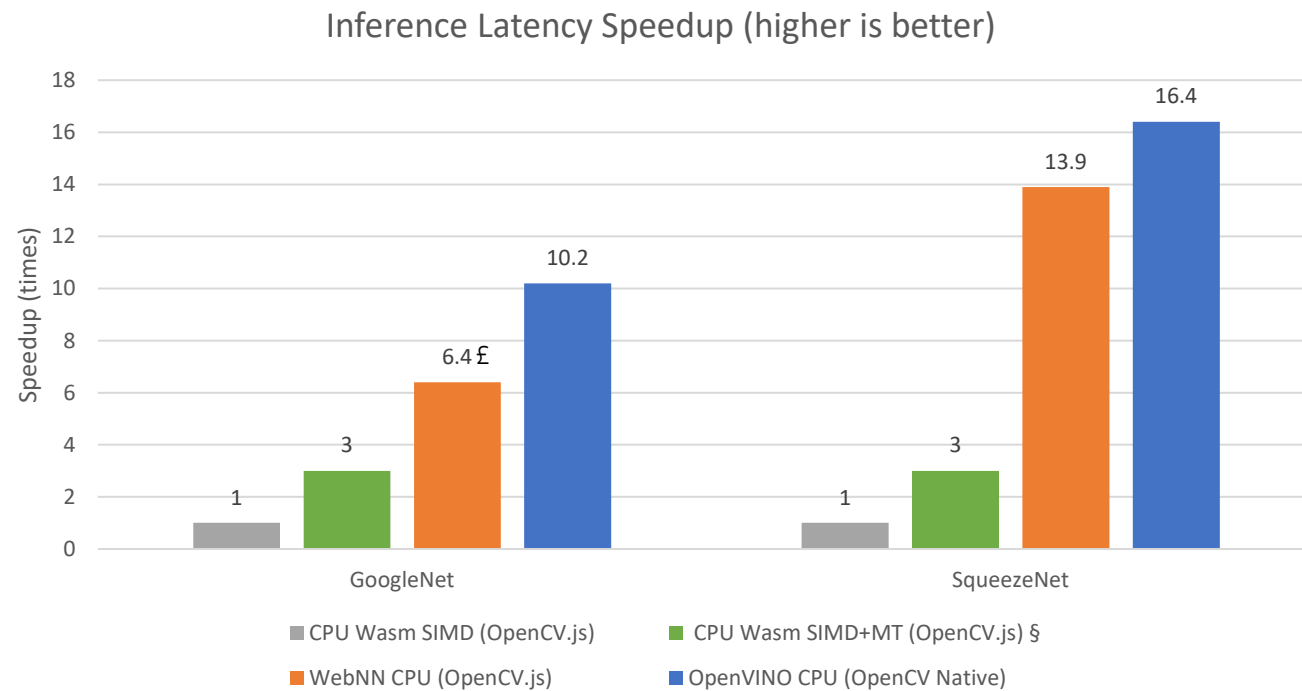


#	Label	Probability
1	space shuttle	81.30%
2	submarine, pigboat, sub, U-boat	10.92%
3	trailer truck, tractor trailer, trucking rig, rig, articulated lorry, semi	1.21%

Model: squeezeNet_v1.0.caffemodel
Inference time: 3.90 ms

The packaged Electron.js app runs OpenCV.js DNN example with WebNN backend

OpenCV.js DNN Module with WebNN Backend Performance *



* Tested on a Linux laptop. All models are FP32 precision. See backup for workloads and configurations. Results may vary.

§ Enabling Wasm MT (Multi-Threading/SharedArrayBuffer) requires "[cross-origin isolation](#)".

£ LRN is currently not supported by WebNN backend and fallbacks to Wasm backend

WebNN Design Highlights for ML Framework

- Graph API
 - Work with framework's graph partitioner
 - Always able to fallback to default backend, allow progressive improvement
- Separate build and compute
 - map to framework's model compilation and inference stages
- Sync API
 - Work with framework's C++ code base
 - Blocking main thread issue: move to worker
- Produce results in standard layout into pre-allocated output buffer
 - work with tensor buffers of frameworks without memory copying and conversion
- Fused ops (e.g., conv2d with bias and activation):
 - Work with framework's graph optimizer that fuses ops
- Rich options (e.g., tensor layout)
 - maximize the compatibility of frameworks

Summary

- WebNN would help ML JS framework get close-to-native performance by accessing native ML API
 - Proven by Electron.js/node.js implementation
 - Help project the performance of browser implementation
- Proposals:
 - Implement WebNN API in Web browsers
 - Implement WebNN backend in ML JS frameworks
 - Support WebNN in Emscripten

Backup - Workloads and Configurations

- **ONNX Runtime workload:** Device ASUS ZenBook Flip S laptop with Windows 10 Home/version 21H1/os build 19043.1237, Hardware CPU: 11th Generation Intel® Core™ i7 Processors/ TGL i7-1165G7, # of Cores: 4, # of Threads: 8, Max Turbo Frequency: 4.70 GHz, Cache: 12 MB Intel® Smart Cache, Bus Speed: 4 GT/s, Configurable TDP-up Frequency: 2.80 GHz, GPU: Intel® Iris® Xe Graphics, Graphics Max Dynamic Frequency: 1.30 GHz, Execution Units: 96, Memory: 16G LPDDR4X/512G SSD, Advanced Technologies: Intel® Deep Learning Boost (Intel® DL Boost), Intel® Speed Shift Technology, Intel® Hyper-Threading Technology, Intel® Turbo Boost Technology: 2.0, Instruction Set: 64-bit, Instruction Set Extensions: Intel® SSE4.1, Intel® SSE4.2, Intel® AVX2, Intel® AVX-512, etc. Software native test tool [ONNXRuntime Performance Test](#), web test tool [ONNX Runtime Web Demo](#), OpenVINO™ toolkit version: 2021.4, Node.js version: 14.16.1, Electron: version 15.1.2, Chrome version: Stable 94.0.4606.81, Python version: 3.8, etc. tested by Intel on 10/18/2021.
- **TFLite workload:** Device MSI Prestige 14 Evo laptop with Ubuntu 20.04.1 LTS, Hardware CPU: 11th Generation Intel® Core™ i7 Processors/ TGL i7-1185G7, # of Cores: 4, # of Threads: 8, Max Turbo Frequency: 4.80 GHz, Cache: 12 MB Intel® Smart Cache, Bus Speed: 4 GT/s, Configurable TDP-up Frequency: 3.00 GHz, GPU: Intel® Iris® Xe Graphics, Graphics Max Dynamic Frequency: 1.35 GHz, Execution Units: 96, Memory: 16G LPDDR4X/512G SSD, Advanced Technologies: Intel® Deep Learning Boost (Intel® DL Boost), Intel® Speed Shift Technology, Intel® Hyper-Threading Technology, Intel® Turbo Boost Technology: 2.0, Instruction Set: 64-bit, Instruction Set Extensions: Intel® SSE4.1, Intel® SSE4.2, Intel® AVX2, Intel® AVX-512, etc. Software native test tool [TensorFlow Model Benchmark Tool](#), web test tool [TFLite Web API Model Runner Demo](#), OpenVINO™ toolkit version: 2021.4, Node.js version: 14.17.0, Electron: version 15.1.2, Chrome version: Stable 94.0.4606.81, Python version: 3.8, etc., tested by Intel on 10/18/2021.
- **OpenCV workload:** Device MSI Prestige 14 Evo laptop with Ubuntu 20.04.1 LTS, Hardware CPU: 11th Generation Intel® Core™ i7 Processors/ TGL i7-1185G7, # of Cores: 4, # of Threads: 8, Max Turbo Frequency: 4.80 GHz, Cache: 12 MB Intel® Smart Cache, Bus Speed: 4 GT/s, Configurable TDP-up Frequency: 3.00 GHz, GPU: Intel® Iris® Xe Graphics, Graphics Max Dynamic Frequency: 1.35 GHz, Execution Units: 96, Memory: 16G LPDDR4X/512G SSD, Advanced Technologies: Intel® Deep Learning Boost (Intel® DL Boost), Intel® Speed Shift Technology, Intel® Hyper-Threading Technology, Intel® Turbo Boost Technology: 2.0, Instruction Set: 64-bit, Instruction Set Extensions: Intel® SSE4.1, Intel® SSE4.2, Intel® AVX2, Intel® AVX-512, etc. Software workload source code [\[GSoC\] OpenCV.js: Accelerate OpenCV.js DNN via WebNN](#), OpenVINO™ toolkit version: 2021.4, Node.js version: 14.17.0, Electron: version 15.1.2, Chrome version: Stable 94.0.4606.81, Python version: 3.8, etc., tested by Intel on 10/18/2021.

Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Your costs and results may vary.

Intel technologies may require enabled hardware, software or service activation.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.