WebNN API - 将硬件加速的深度学习带入开放式 Web 平台

元凯宁

Intel 软件技术经理
kaining.yuan@intel.com

Geekbang. 极客邦科技 InfoQ

极客邦科技
会议推荐2019

5月

QCon 北京
全球软件开发大会
大会：5月6–8日
培训：5月9–10日

QCon 广州
全球软件开发大会
培训：5月25–26日
大会：5月27–28日

6月

GTLC GLOBAL TECH LEADERSHIP CONFERENCE 上海
技术领导力峰会
时间：6月14–15日

GMTC 北京
全球大前端技术大会
大会：6月20–21日
培训：6月22–23日

7月

ArchSummit 深圳
全球架构师峰会
大会：7月12–13日
培训：7月14–15日

10月

QCon 上海
全球软件开发大会
大会：10月17–19日
培训：10月20–21日

11月

GMTC 深圳
全球大前端技术大会
大会：11月8–9日
培训：11月10–11日

AiCon 北京
全球人工智能与机器学习大会
大会：11月21–22日
培训：11月23–24日

12月

ArchSummit 北京
全球架构师峰会
大会：12月6–7日
培训：12月8–9日

Geekbang 极客邦科技 | TGO 鲲鹏會

TGO鲲鹏会

汇聚全球科技领导者的高端社群

全球12大城市

850+高端科技领导者

使命 Mission

为社会输送更多优秀的科技领导者

愿景 Vision

构建全球领先的有技术背景优秀人才的学习成长平台

扫描二维码，了解更多内容

自我介绍

Intel 亚太研发有限公司 Web Platform Engineering Team 软件技术经理。参与 Intel 开源的 Crosswalk、RealSense 等项目，目前带领的团队负责 PWA 以及 Web Machine Learning 等方向。

目录

- 前端 JavaScript* ML 的演进与问题
- WebNN API 的提案与实现
- WebNN API 的代码及示例
- WebNN API 的性能对比
- W3C 社区组的进展及合作

* 其他名称和品牌可能是第三方的财产。

前端 JavaScript* ML 的演进

* 其他名称和品牌可能是第三方的财产。

前端 JavaScript* ML 的优势

延迟

访问本地资源的浏览器内推理。

成本

客户端计算意味着不需要服务器端算力支持。

可用性

初始资源缓存并离线后，不再依赖网络。

共享

在浏览器中运行 ML，而无需任何额外的安装，并易于共享。

隐私

敏感数据本地访问。

跨平台

可以轻松开发在几乎所有平台上运行的 AI 应用。

* 其他名称和品牌可能是第三方的财产。

前端 JavaScript* ML 框架的挑战与努力

性能

设备中的机器学习推理需要高性能的数值计算能力。

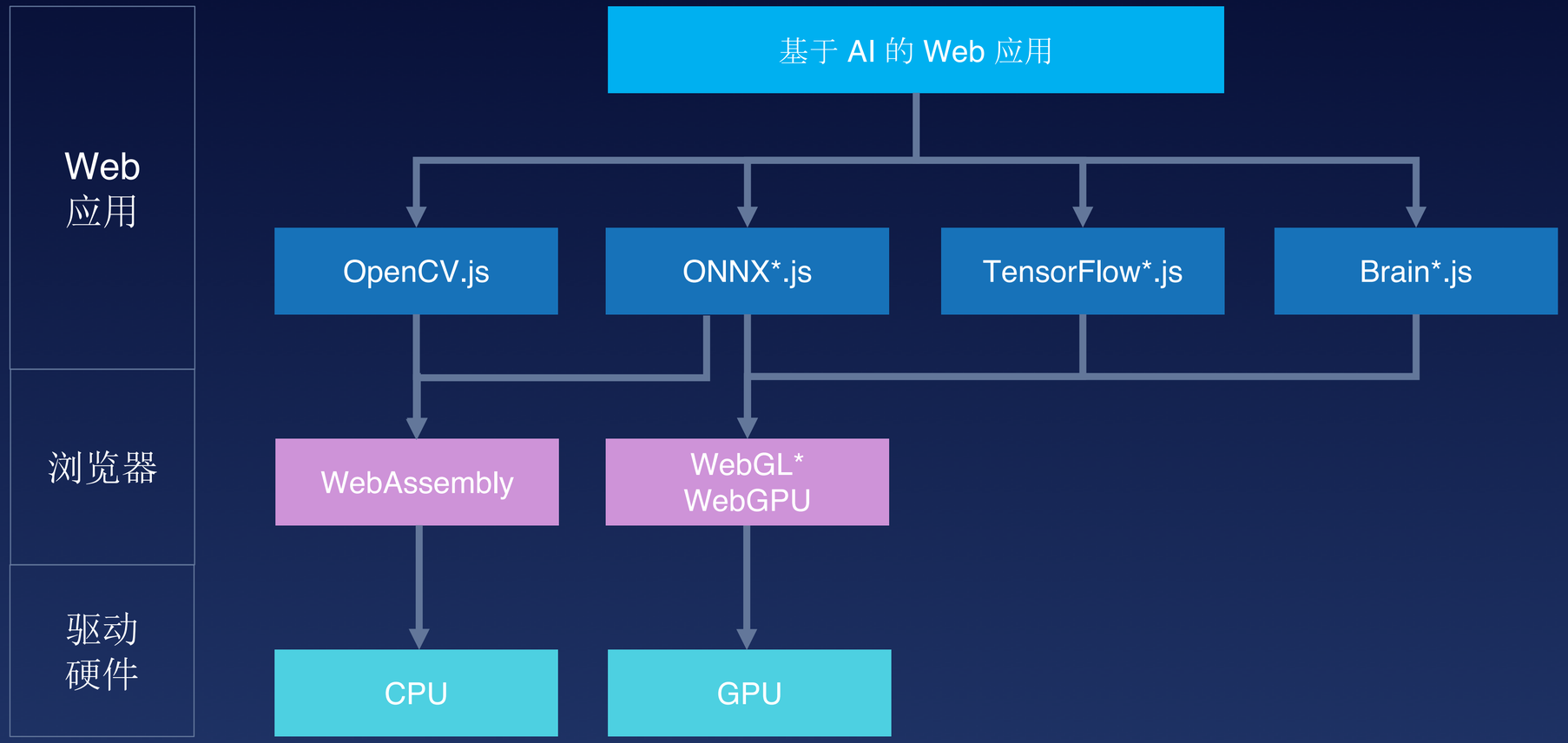
WEBASSEMBLY

将 C++ 编译为高效的字节码，以便在浏览器中进行解释和执行。

WebGL™

重新调整图形 API 用于并行计算。

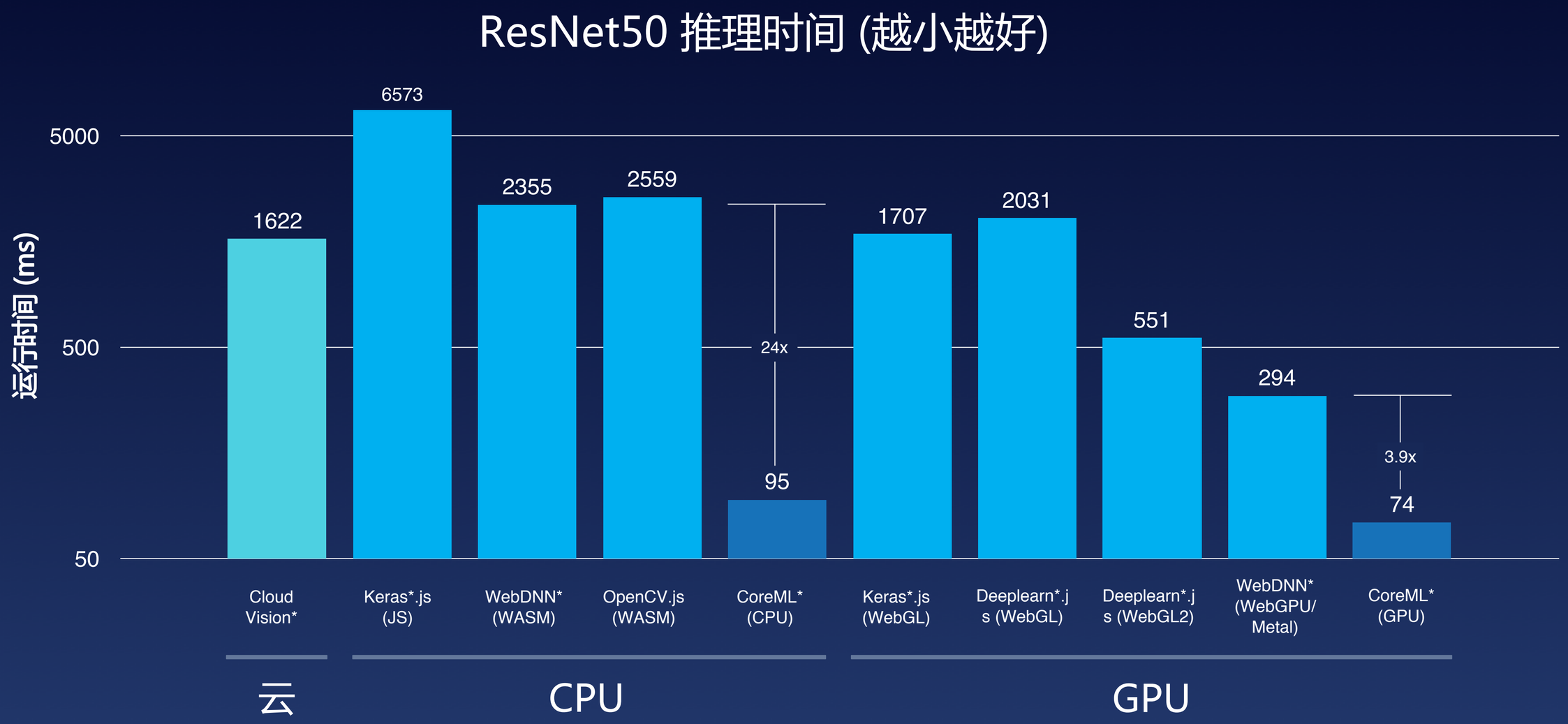
前端 JavaScript* ML 框架



新兴的基于
AI 的 Web 应用
和 JavaScript* 框架

* 其他名称和品牌可能是第三方的财产。

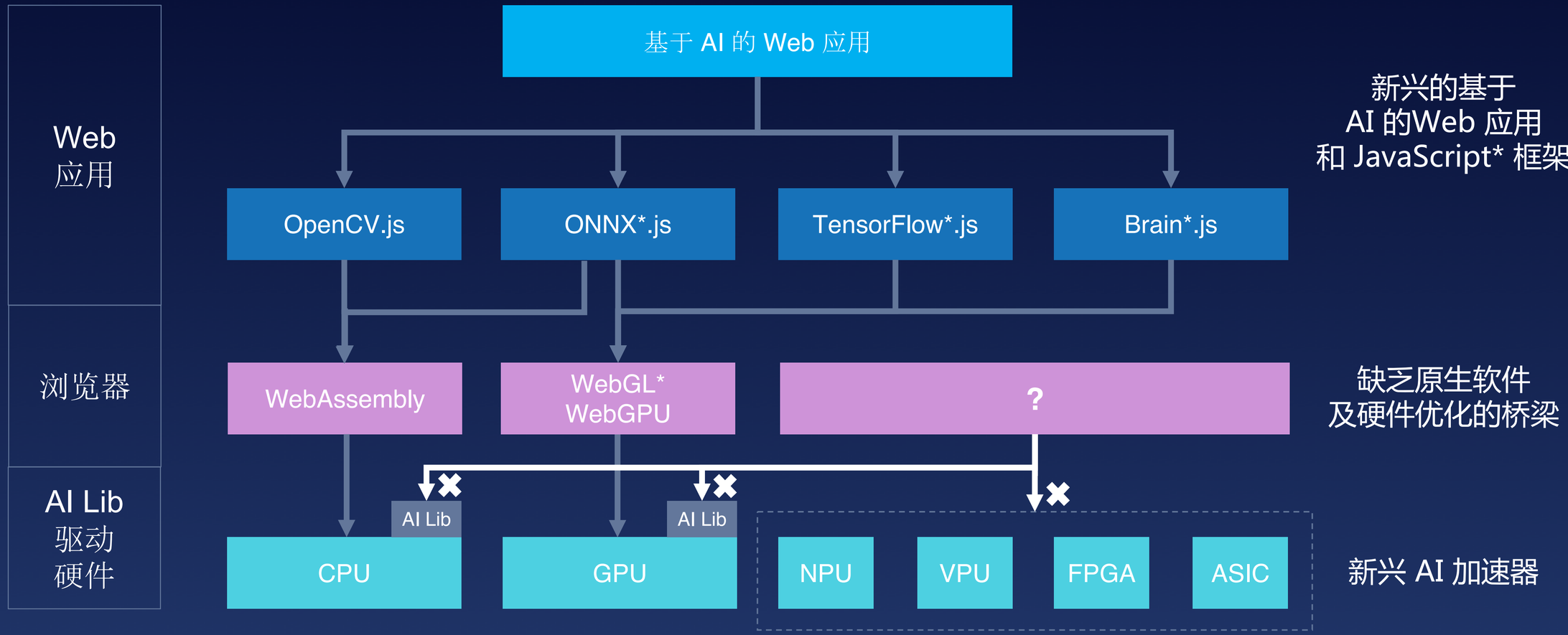
性能: JavaScript* ML 框架的主要问题



* 其他名称和品牌可能是第三方的财产。

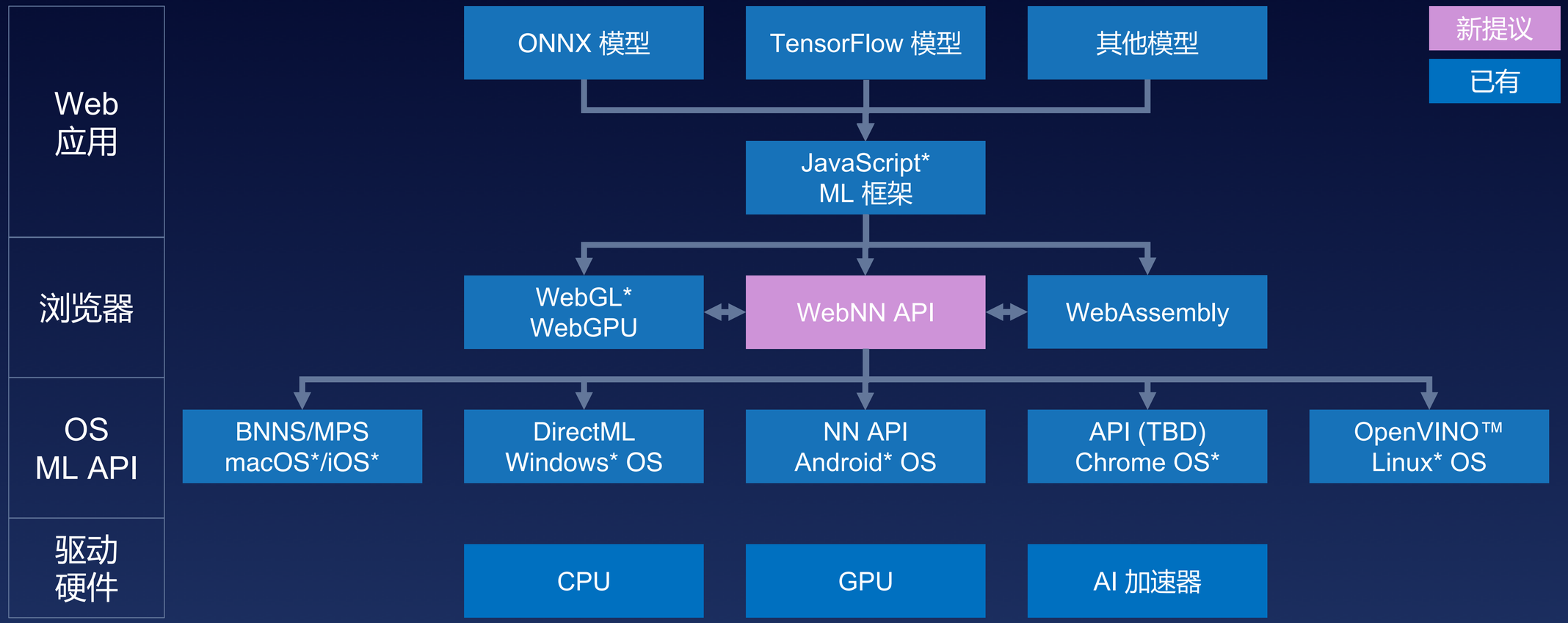
MacBook Pro (13-inch), SkyLake Core i5 2.9GHz, HD 550, macOS 10.14.4
ResNet50, trained by ImageNet, inference, batch size 1, warm up 1, iteration 10

前端 JavaScript* ML 框架与原生优化的隔绝



* 其他名称和品牌可能是第三方的财产。

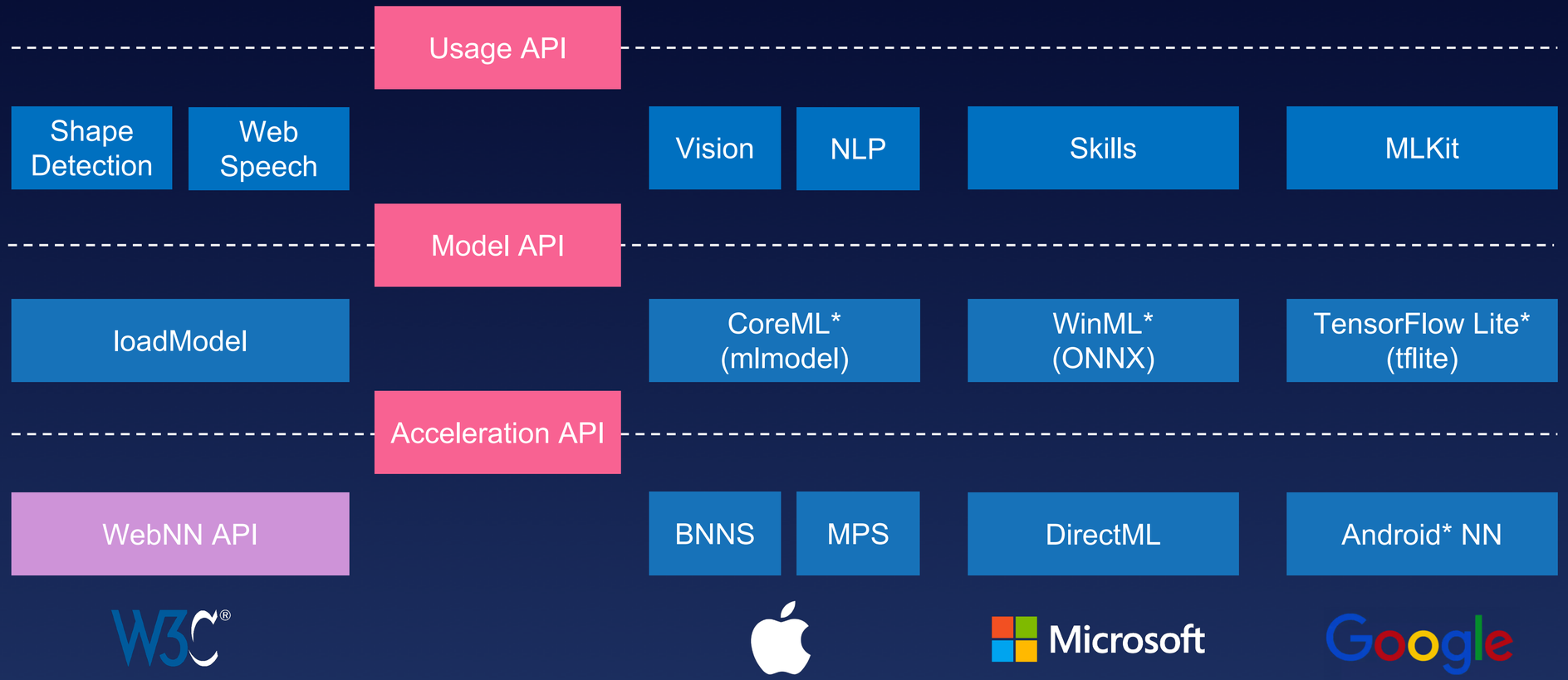
WebNN API: 为硬件加速而生



- WebNN API: 用于深度神经网络推理的基于标准的 Web API
- 与文本、多媒体、传感器和 XR 等其他 Web API 集成
- 充分利用硬件功能，将 Web 深度学习运算交由系统 API，实现神经网络推理的硬件加速

* 其他名称和品牌可能是第三方的财产。

前端 Web API 分层架构



- Usage API: 内置模型，易于集成 ⇒ W3C 图形检测 API
- Model API: 模型预先训练，格式存在碎片化问题
- Acceleration API: NN 底层 API，接近硬件优化，灵活适配 JS 框架

* 其他名称和品牌可能是第三方的财产。

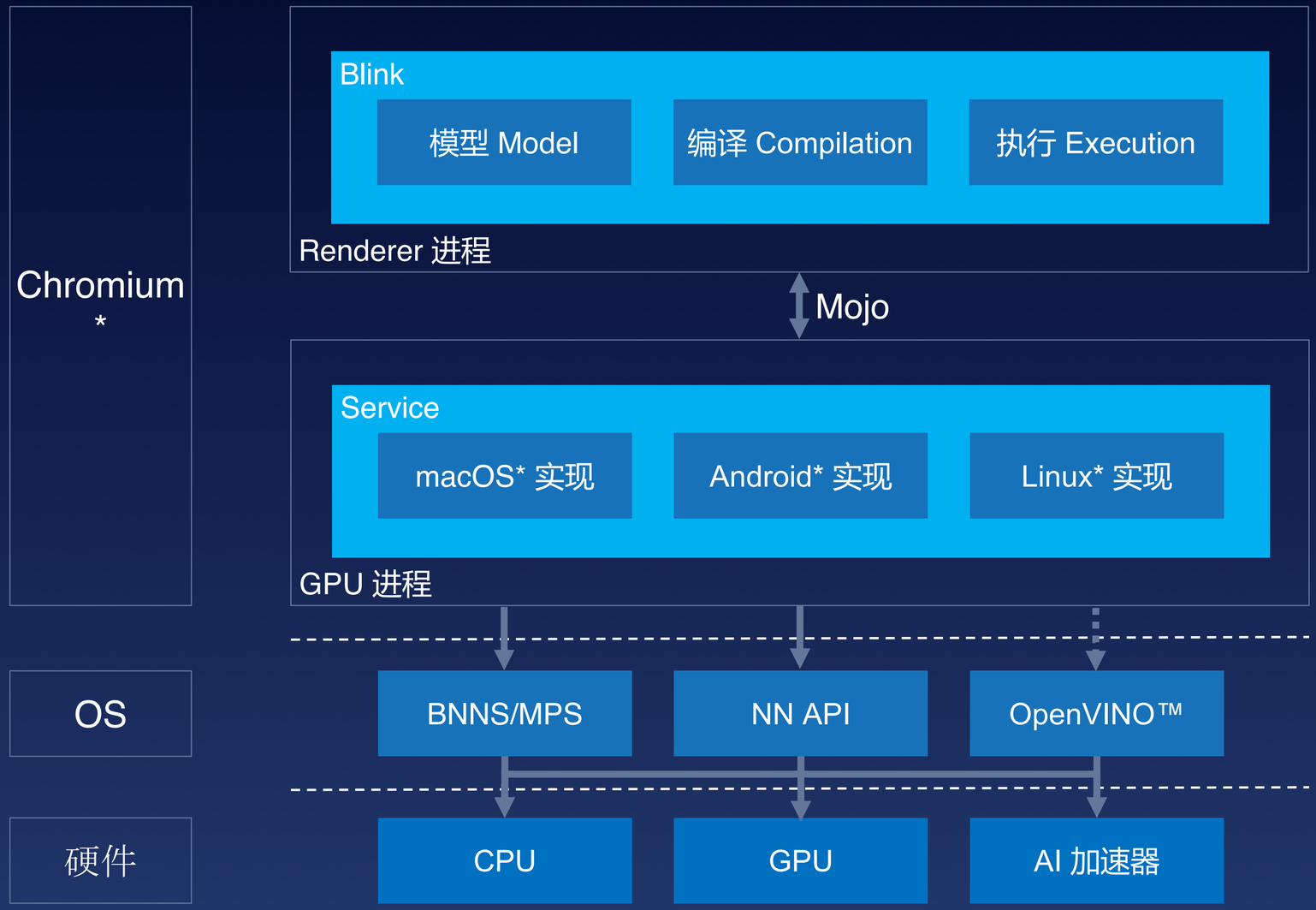
WebNN API 提案


```
interface Model {
    void addOperand(OperandOptions options);
    void setOperandValue(
             unsigned long index,
             ArrayBufferView data);
    void addOperation(
            long type,
            sequence<unsigned long> inputs,
            sequence<unsigned long> outputs);
    void identifyInputsAndOutputs(
            sequence<unsigned long> inputs,
            sequence<unsigned long> outputs);
    Promise<long> finish();
    Promise<Compilation> createCompilation();
};
```

```
interface Compilation {
     void setPreference(long preference);
    Promise<long> finish();
    Promise<Execution> createExecution();
};
```

```
interface Execution {
    void setInput(
            unsigned long index,
            ArrayBufferView data);
    void setOutput(
            unsigned long index,
            ArrayBufferView data);
    Promise<long> startCompute();
};
```

WebNN API: Chromium* 中的实现



JavaScript:
Promise<Compilation>
createCompilation();

C++:
void
ModelImpl::CreateCompilation(CreateCompilationCallback callback)

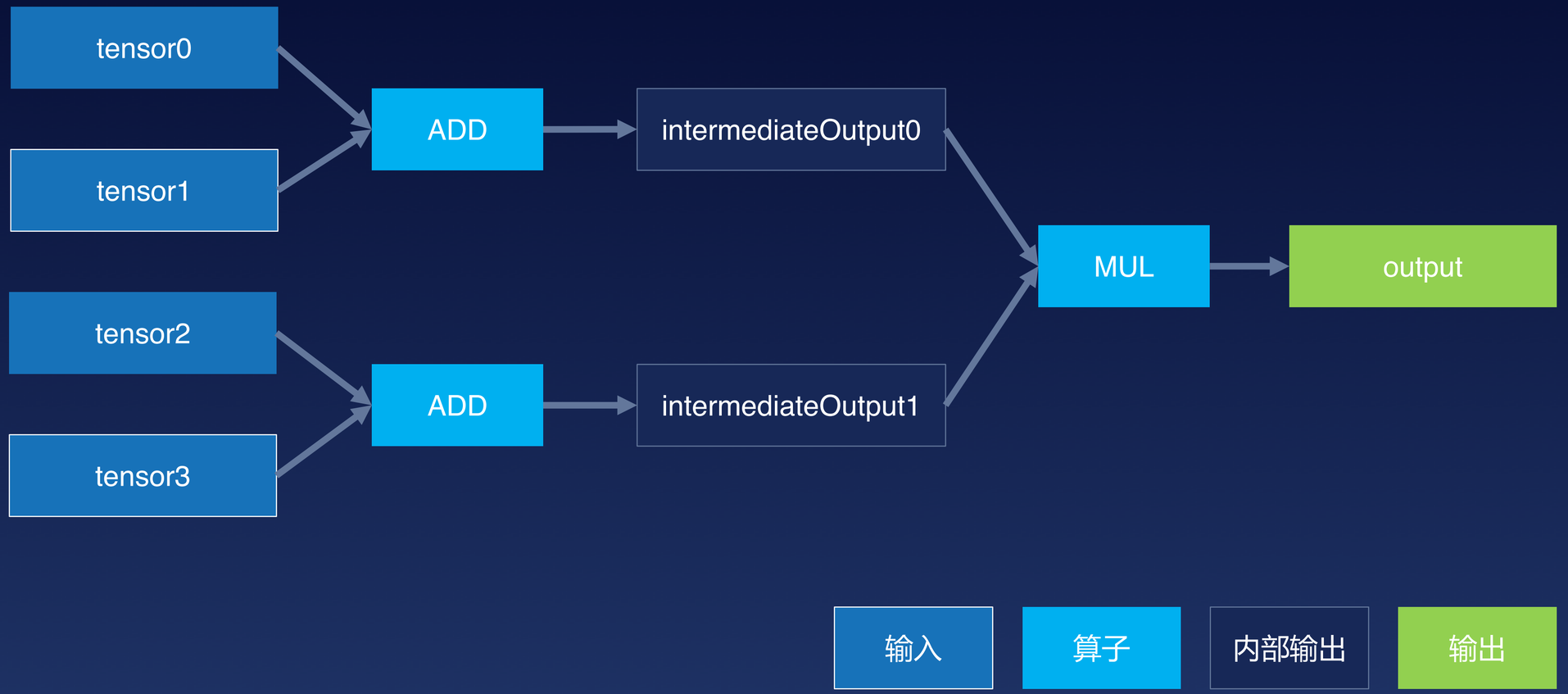
JavaScript:
CONV_2D

C/C++:
NN API: ANEURALNETWORKS_CONV_2D
MPS: MPSCNNConvolution
BNNS: BNNSFilterCreateConvolutionLayer

https://github.com/otcshare/chromium-src

* 其他名称和品牌可能是第三方的财产。

WebNN API 示例: 简单图



WebNN API 示例: 创建模型

```
// Get the neural network context.
const nn = navigator.ml.getNeuralNetworkContext();

// Create a Model object.
const model = await nn.createModel();
```

WebNN API 示例: 添加操作数

```
// Use 4-D tensor.
const TENSOR_DIMS = [2, 2, 2, 2];
const TENSOR_SIZE = 16;
const float32TensorType = { type: nn.TENSOR_FLOAT32,
                            dimensions: TENSOR_DIMS };

// Track operand index.
let operandIndex = 0;

// tensor0 is a constant tensor, set its value from an ArrayBuffer object.
// The ArrayBuffer object may contain the training data loaded before hand.
const tensor0 = operandIndex++;
model.addOperand(float32TensorType);
model.setOperandValue(tensor0, new Float32Array(arrayBuffer, 0, TENSOR_SIZE));

// tensor1 is one of the input tensors. Its value will be set before
// execution.
const tensor1 = operandIndex++;
model.addOperand(float32TensorType);
```

WebNN API 示例: 添加算子

```
// Add the first ADD operation.
model.addOperation(nn.ADD, [tensor0, tensor1,
                            fusedActivationFuncNone], [intermediateOutput0]);

// Add the second ADD operation.
model.addOperation(nn.ADD, [tensor2, tensor3,
                            fusedActivationFuncNone], [intermediateOutput1]);

// Add the MUL operation.
// Note that intermediateOutput0 and intermediateOutput1 are specified
// as inputs to the operation.
model.addOperation(nn.MUL, [intermediateOutput0, intermediateOutput1,
                            fusedActivationFuncNone], [output]);

// Identify the input and output tensors to the model.
model.identifyInputsAndOutputs([tensor1, tensor3], [output]);

// Finish building the model.
await model.finish();
```

WebNN API 示例: 编译模型

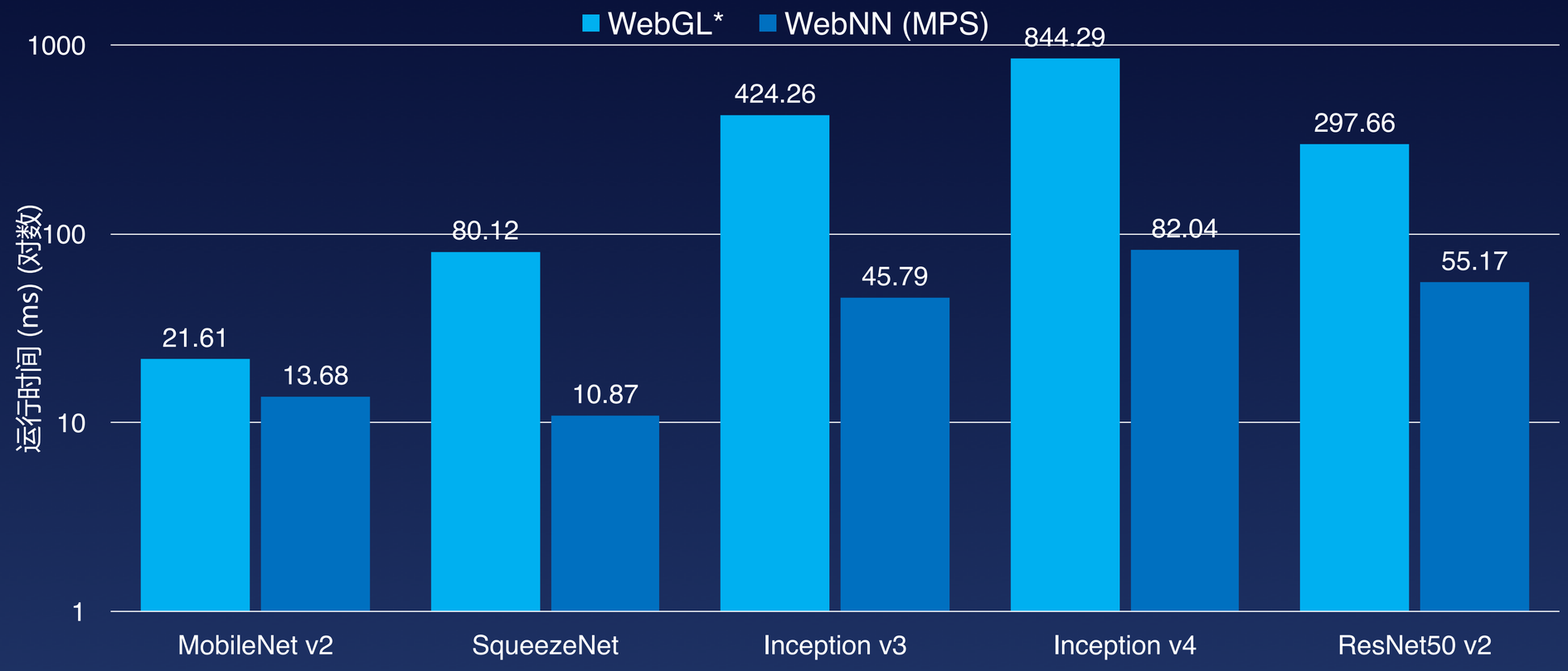
```
// Create a Compilation object for the constructed model.
let compilation = await model.createCompilation();

// Set the preference for the compilation.
// Other options include SUSTAINED_SPEED and LOW_POWER
compilation.setPreference(nn.PREFER_FAST_SINGLE_ANSWER);

// Finish the compilation.
await compilation.finish();
```

WebNN API 示例: 模型执行

```
// Create an Execution object for the compiled model.
let execution = await compilation.createExecution();

// Setup the input tensors.
// They may contain data provided by user.
let inputTensor1 = new Float32Array(TENSOR_SIZE);
inputTensor1.fill(inputValue1);
let inputTensor2 = new Float32Array(TENSOR_SIZE);
inputTensor2.fill(inputValue2);

// Associate input tensors to model inputs.
execution.setInput(0, inputTensor1);
execution.setInput(1, inputTensor2);

// Associate output tensor to model output.
let outputTensor = new Float32Array(TENSOR_SIZE);
execution.setOutput(0, outputTensor);

// Start the asynchronous computation.
await execution.startCompute();

// The computed result is now in outputTensor.
```

性能: iGPU / macOS

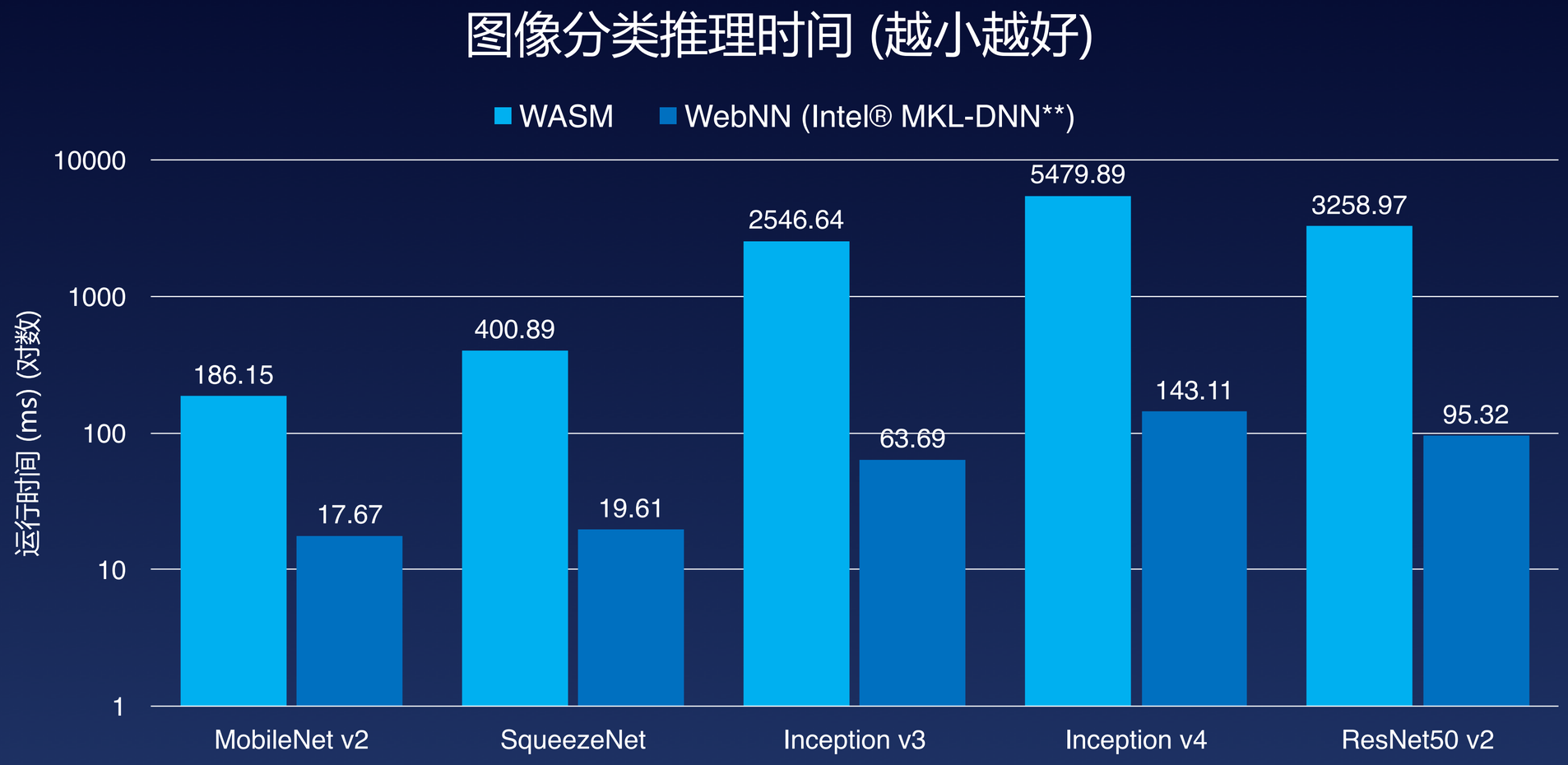
图像分类推理时间 (越小越好)



	MobileNet v2	SqueezeNet	Inception v3	Inception v4	ResNet50 v2
WebNN 加速	1.6X	7.4X	9.3X	10.3X	5.4X

MacBook Pro (13-inch, 2018), Intel i5-8259U, Iris Plus Graphics 655, 16GB, macOS 10.14.4

性能: CPU / Linux



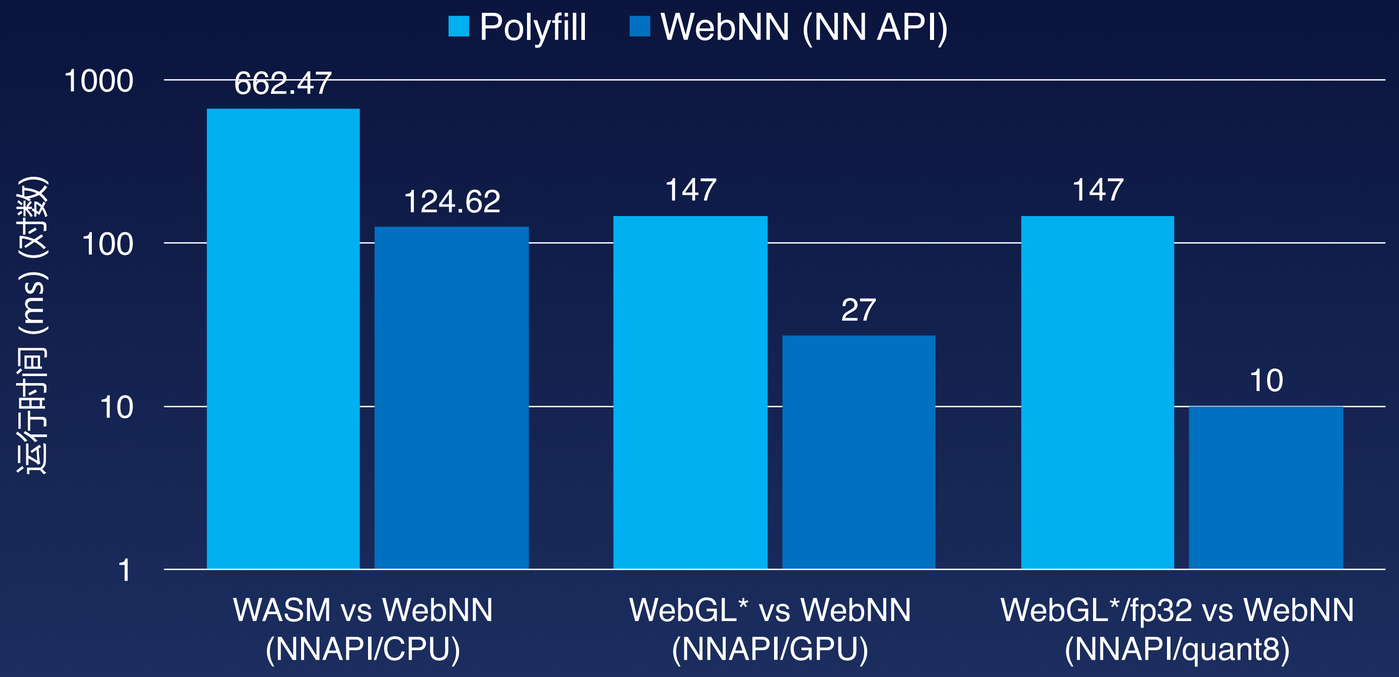
WebNN 加速	10.5X	20.4X	40.0X	38.3X	34.2X

* 其他名称和品牌可能是第三方的财产。
** Intel® Math Kernel Library for Deep Neural Networks

Dell XPS 13, Intel i7-8550U, Intel UHD Graphics 620 (Kabylake GT2), 16GB, Linux Ubuntu 16.04

性能: 移动设备 / Android

图像分类推理时间 (越小越好)



WebNN 加速	5.3X	5.4X	14.7X

* 其他名称和品牌可能是第三方的财产。

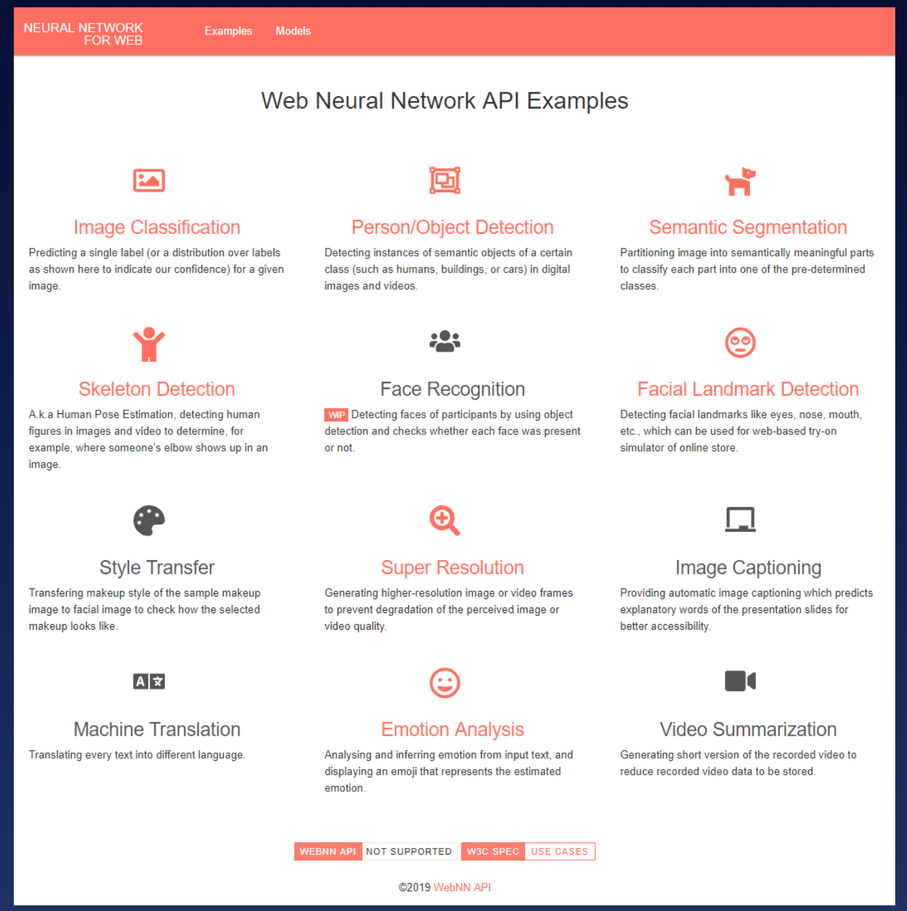
Google Pixel 3, Qualcomm Snapdragon 845, Adreno 630, 4GB, Android 9

WebNN API: W3C 规范及示例

Web
Neural Network API

This version:
webmachinelearning.github.io/webnn/

Editor:
Ningxin Hu (Intel Corporation)



https://intel.github.io/webml-polyfill/examples/

WebNN API: W3C 规范及 API

Web
Neural Network API

This version:
webmachinelearning.github.io/webnn/

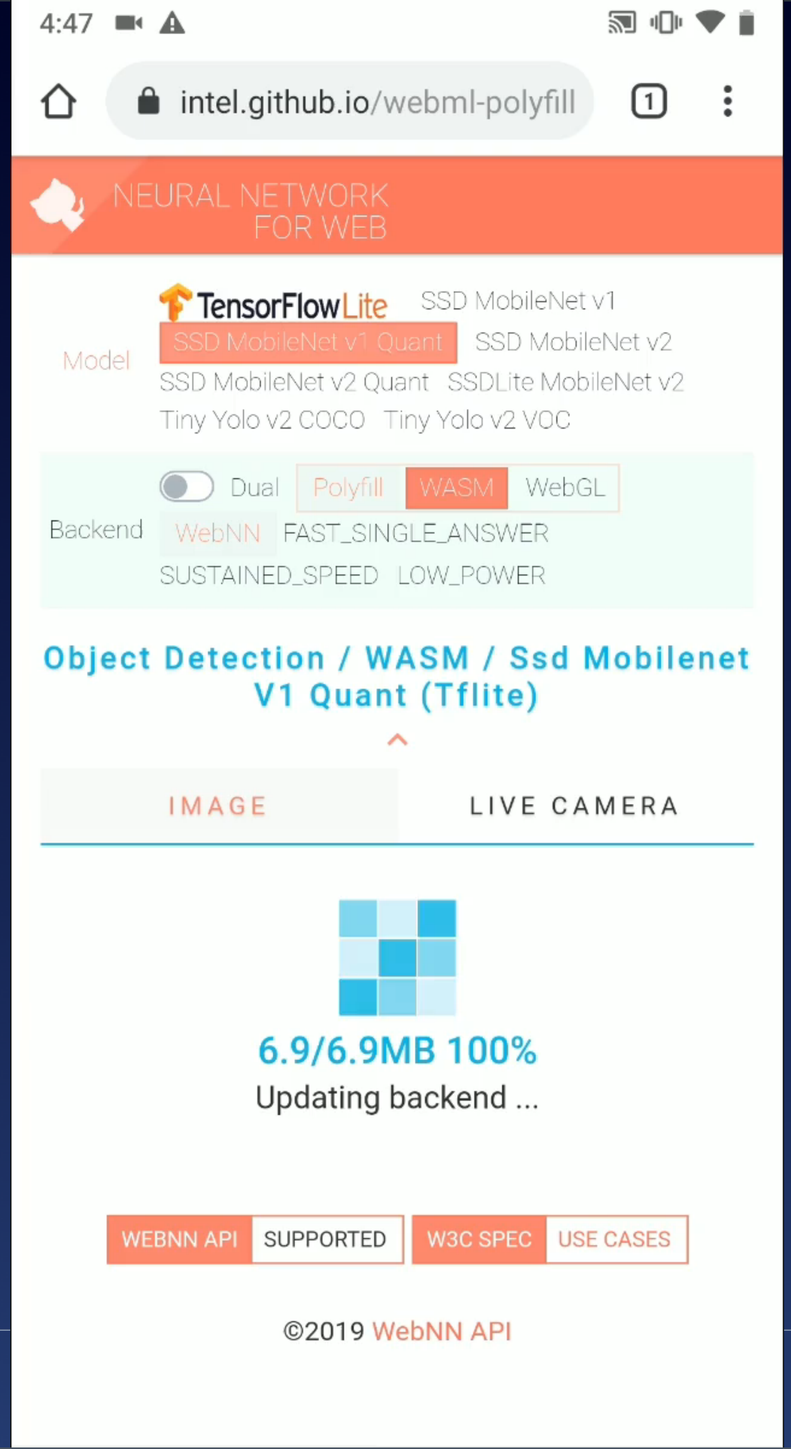
Editor:
Ningxin Hu (Intel Corporation)


```
// Operation types.
const long ADD = 0;
const long AVERAGE_POOL_2D = 1;
const long CONCATENATION = 2;
const long CONV_2D = 3;
const long DEPTHWISE_CONV_2D = 4;
const long DEPTH_TO_SPACE = 5;
const long DEQUANTIZE = 6;
const long EMBEDDING_LOOKUP = 7;
const long FLOOR = 8;
const long FULLY_CONNECTED = 9;
const long HASHTABLE_LOOKUP = 10;
const long L2_NORMALIZATION = 11;
const long L2_POOL_2D = 12;
const long LOCAL_RESPONSE_NORMALIZATION = 13;
const long LOGISTIC = 14;
const long LSH_PROJECTION = 15;
const long LSTM = 16;
const long MAX_POOL_2D = 17;
const long MUL = 18;
const long RELU = 19;
const long RELU1 = 20;
const long RELU6 = 21;
const long RESHAPE = 22;
const long RESIZE_BILINEAR = 23;
const long RNN = 24;
const long SOFTMAX = 25;
const long SPACE_TO_DEPTH = 26;
const long SVDF = 27;
const long TANH = 28;
const long BATCH_TO_SPACE_ND = 29;
const long TRANSPOSE = 37;
```

- 通过性能、一致性和实现质量的差异，获得更好的用户体验。
- 基于标准的 Web API 的目标都是规范化硬件差异，为应用开发人员提供统一的接口。
- WebNN API 被提议作为用于深度神经网络的硬件加速 API。

示例: 目标检测

目标检测性能		WASM	WebNN
运行时间(推理)	图片	3696 ms	13.60 ms
	视频	3776 ms	14.60 ms
帧率	视频	0-1 FPS	30 FPS



* 其他名称和品牌可能是第三方的财产。

WebNN API: 更多的集成与合作

* 其他名称和品牌可能是第三方的财产。

W3C 社区组

W3C® Machine Learning for the Web

https://webmachinelearning.github.io
https://github.com/webmachinelearning/webnn/issues

标准孵化

Web Neural Network API

有用的资源

- Community Group
- Participants
- Meetings
- Charter
- GitHub*
- Mailing List

项目

https://github.com/intel/webml-polyfill

2018-10-03: W3C Web ML 社区组成立, CG 主席: Anssi (Intel)
2018-10-11: WebML CG 章程: 用于神经网络推理硬件加速的专用 API
2019-04-25: Google、微软等代表同意将 Intel WebNN POC API 作为基础规范
W3C 社区组邀请浏览器引擎开发人员，硬件供应商，Web 应用开发人员以及对机器学习感兴趣的更广泛的 Web 社区参与

Google Microsoft moz://a Baidu 百度 intel SAMSUNG HUAWEI

免责声明

本文档不以禁止翻供或其它的任何方式，明示或暗示授予任何知识产权下的许可证。

英特尔不承担任何明示或暗示保证，包括与特定意图的适用性、商销性或违反专利、版权或其它知识产权等有关的责任或保证，以及因性能，交易过程或交易使用而产生的任何保证。

本文档包含有关正在开发的产品，服务和/或流程的信息。此处提供的所有信息如有更改，恕不另行通知。请联系您的英特尔代表，以获取最新的预测，时间表，规格和路线图。

所描述的产品和服务可能包含已知为勘误表的缺陷或错误，可能会导致与已发布的规范不符。可根据要求提供当前特征勘误表。没有任何产品或组件可以绝对安全。

可致电 1-800-548-4725 或访问 www.intel.com/design/literature.htm 获取本文件中带有订货号并参考的文件副本。

Intel，Intel 徽标和 OpenVINO 是 Intel Corporation 或其子公司在美国和/或其他国家/地区的商标。

* 其他名称和品牌可能是第三方的财产。

Geekbang 极客邦科技 | InfoQ 极客传媒

InfoQ官网 全新改版上线

促进软件开发领域知识与创新的传播

关注InfoQ网站
第一时间浏览原创IT新闻资讯

免费下载迷你书
阅读一线开发者的技术干货

THANKS

GMTC
全球大前端技术大会