IOTG Russia

OpenVINO™

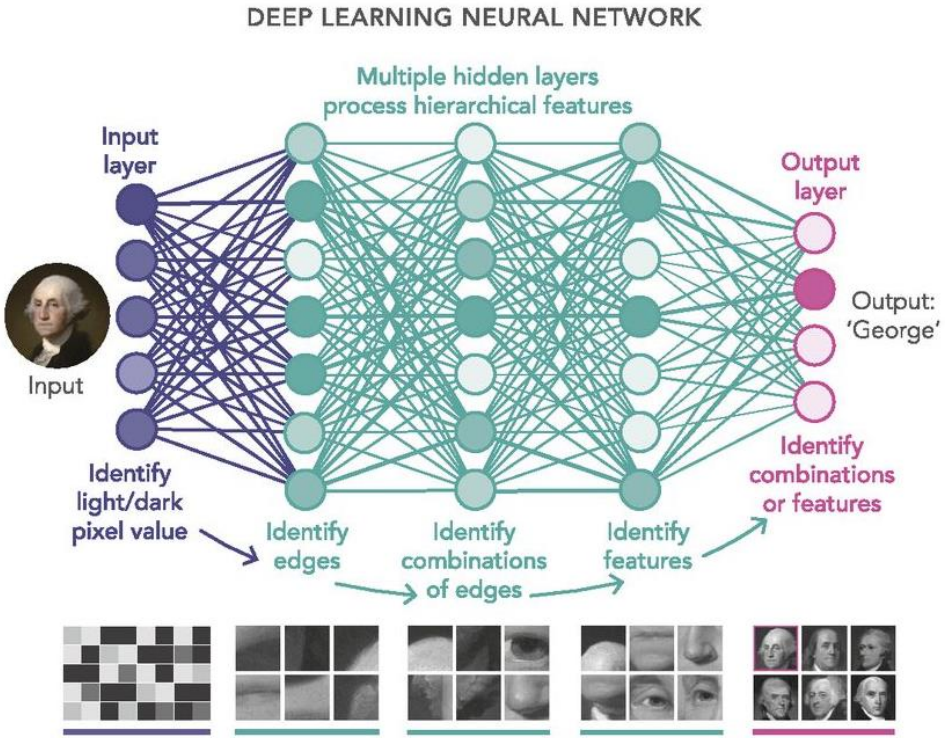
Visual Inference & Neural Network Optimization

Денис Орлов

О чем сегодня пойдет речь?

- Краткое введение в нейронные сети
- Основы OpenVINO (Model Optimizer, Inference Engine)
- Поддерживаемые устройства
- Оптимизация с помощью OpenVINO
- Дополнительные компоненты OpenVINO
- Способы распространения OpenVINO
- Дополнительные материалы

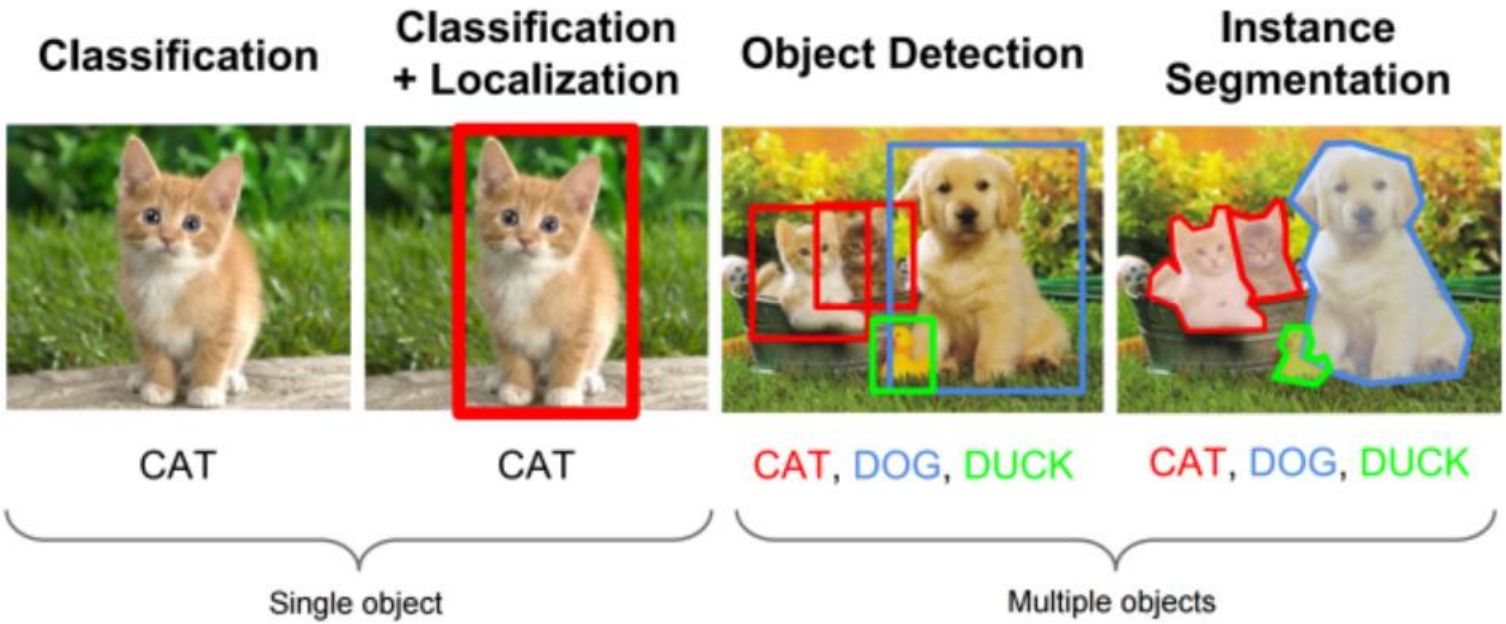
Краткое введение в нейронные сети



M. Mitchell Waldrop PNAS 2019;116:4:1074-1077

Методы deep learning для компьютерного зрения

- Классификация объектов



https://medium.com/analytics-vidhya/yolov3-real-time-object-detection-54e69037b6d0

Методы deep learning для компьютерного зрения

- Семантическая сегментация

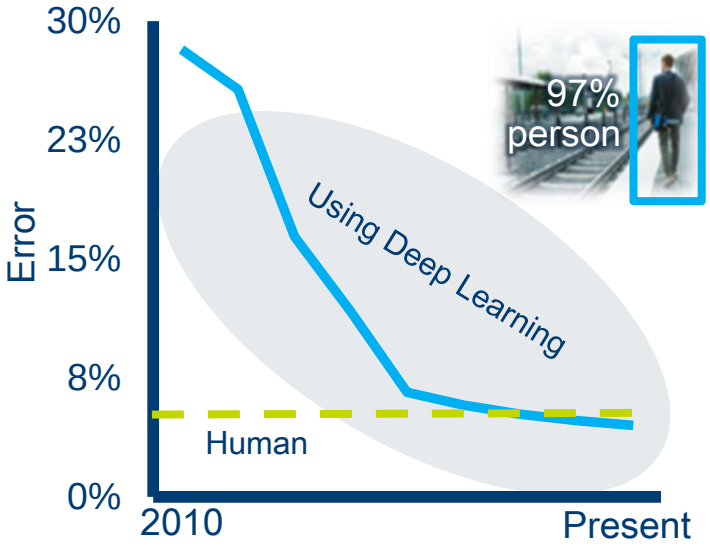
https://mc.ai/introduction-to-semantic-image-segmentation/

Новые применения методов deep learning

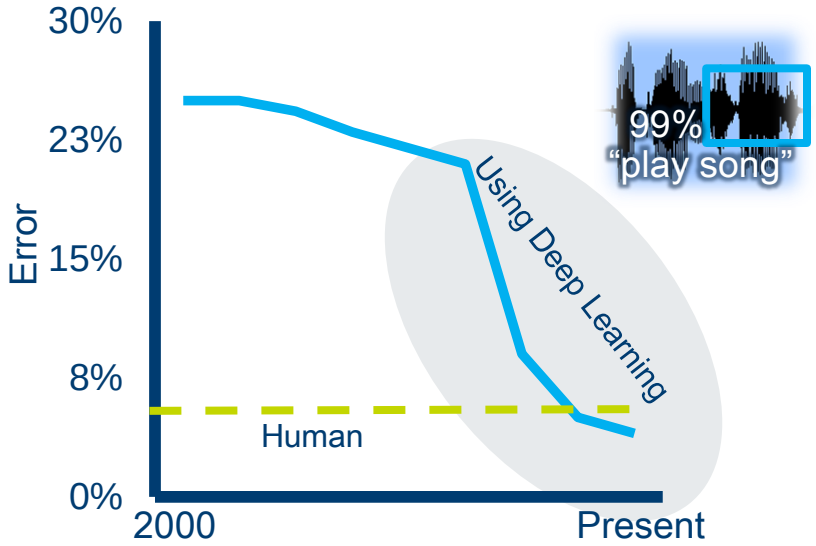
- Машинный перевод
- Распознавание голоса
- Устранение шумов и отражений в звуке
- Классификация звука
- Классификация текста
- Анализ тональности текста (sentiment analysis)
- Идентификация говорящего
- Генерация голоса
- Рекомендательные системы
- ...

Прогресс в области глубокого обучения

Image Recognition

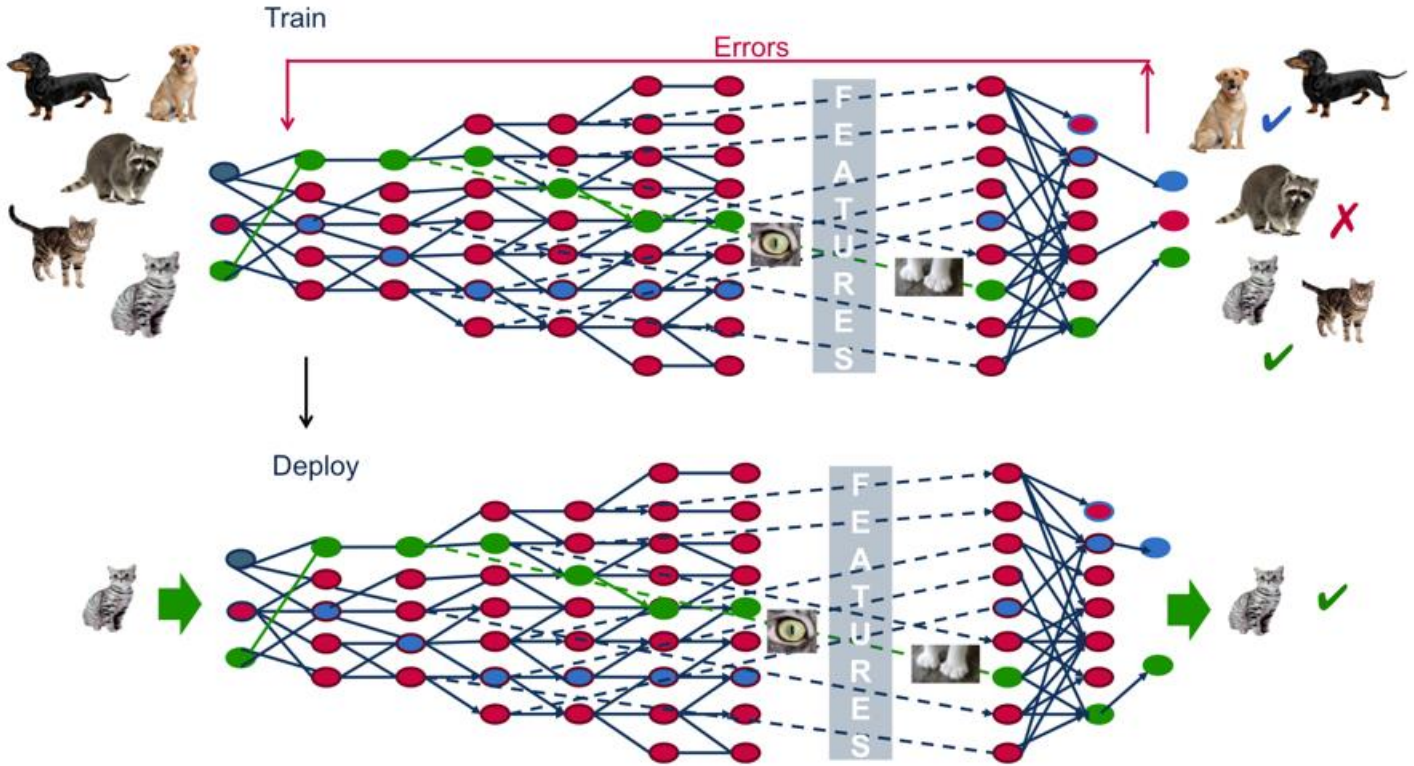


Speech Recognition



Source: ILSVRC ImageNet winning entry classification error rate each year 2010-2016 (Left), https://www.microsoft.com/en-us/research/blog/microsoft-researchers-achieve-new-conversational-speech-recognition-milestone/ (Right)
Source: https://www.mckinsey.com/featured-insights/artificial-intelligence/notes-from-the-ai-frontier-applications-and-value-of-deep-learning

Тренировка vs Запуск («Инференс»)

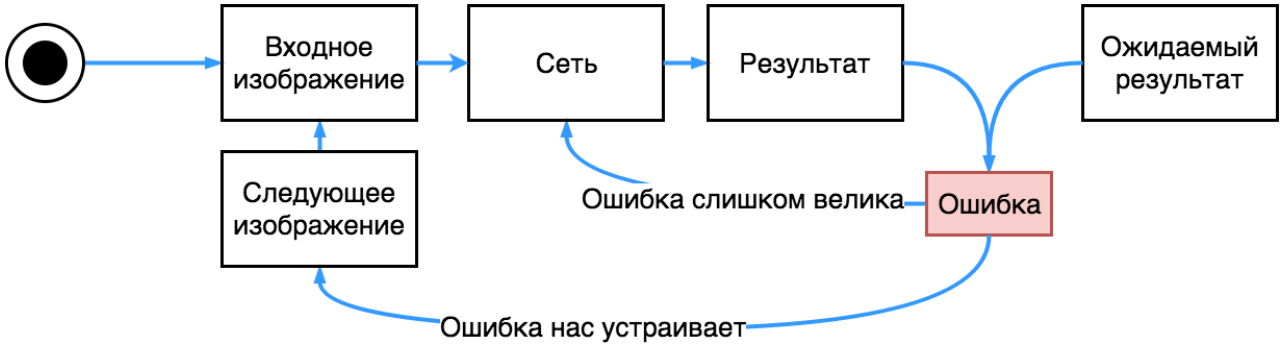


https://www.slideshare.net/caroljmcdonald/demystifying-ai-machine-learning-and-deep-learning

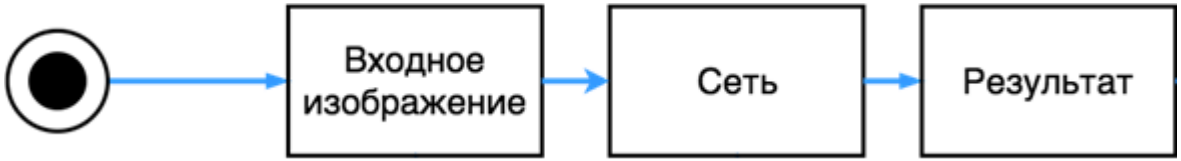
Тренировка vs Запуск («Инференс»)

Тренировка требует:

- больших объёмов данных
- времени (дни, недели)
- значительных вычислительных ресурсов



Инференс – запуск натренированной сети как готовой программы



Популярные фреймворки и инструменты

ONNX

PyTorch

mxnet

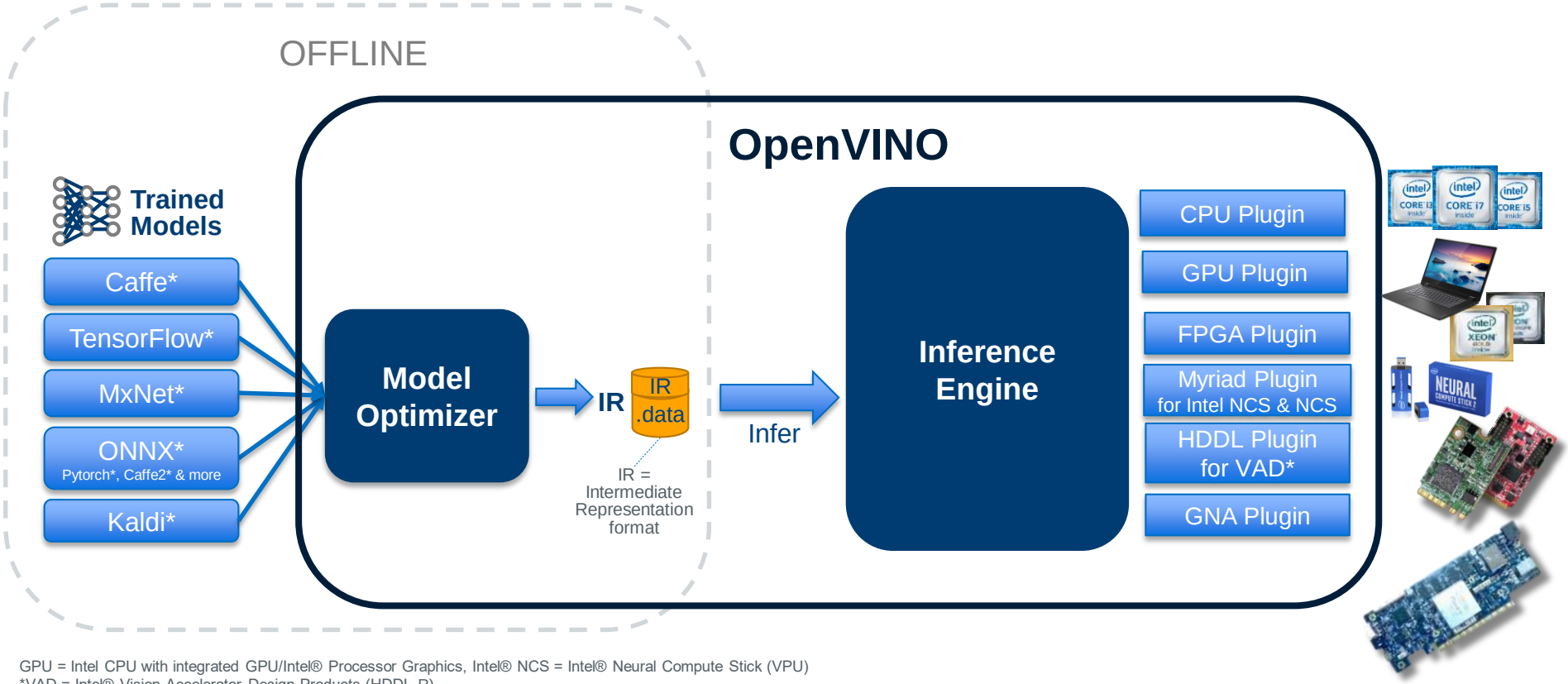
Caffe

Keras

TensorFlow

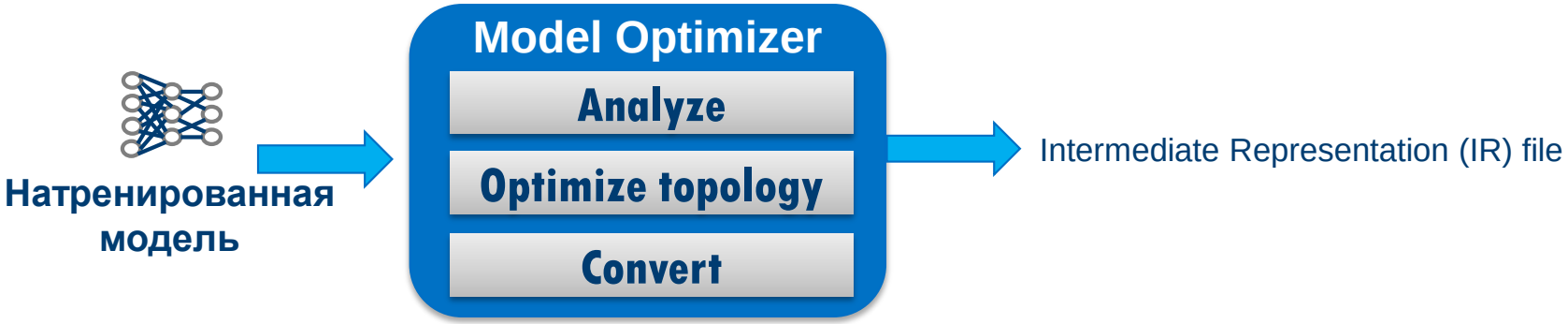
KALDI

Основы OpenVINO



GPU = Intel CPU with integrated GPU/Intel® Processor Graphics, Intel® NCS = Intel® Neural Compute Stick (VPU)
*VAD = Intel® Vision Accelerator Design Products (HDDL-R)

Model Optimizer

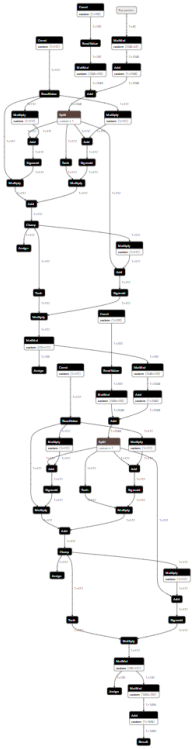


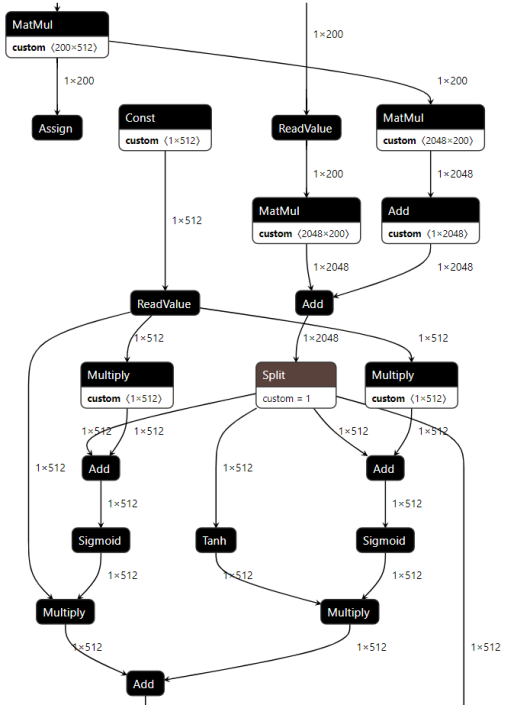
Поддержимаемые фреймворки: Caffe, TensorFlow, MXNet, Kaldi; формат ONNX (Pytorch, Caffe2 и другие, использующие ONNX).

Intermediate Representation (IR) состоит из:

- Xml file (описание топологии)
- Bin file (веса)
- *Альтернативный способ описания модели – с помощью API*

Пример нейронной сети





Как выглядит модель в формате IR?

```
<?xml version="1.0" ?>
<net name="nsnet2-20ms-baseline" version="10">
        <layers>
                <layer id="0" name="input" type="Parameter" version="opset1">
                        <data element_type="f32" shape="1,100,161"/>
                        <output>
                                <port id="0" precision="FP32">
                                        <dim>1</dim>
                                        <dim>100</dim>
                                        <dim>161</dim>
                                </port>
                        </output>
                </layer>
                <layer id="1" name="MatMul_0/1_port_transpose1025_const" type="Const" version="opset1">
                        <data element_type="f32" offset="0" shape="400,161" size="257600"/>
                        <output>
                                <port id="1" precision="FP32">
                                        <dim>400</dim>
                                        <dim>161</dim>
                                </port>
                        </output>
                </layer>
                <layer id="2" name="MatMul_0" type="MatMul" version="opset1">
                        <data transpose_a="False" transpose_b="True"/>
                        <input>
                                <port id="0">
                                        <dim>1</dim>
                                        <dim>100</dim>
                                        <dim>161</dim>
[ … ]
```

Как выглядит модель в формате IR?

```
<edges>
        <edge from-layer="0" from-port="0" to-layer="2" to-port="0"/>
        <edge from-layer="1" from-port="1" to-layer="2" to-port="1"/>
        <edge from-layer="2" from-port="2" to-layer="4" to-port="0"/>
        <edge from-layer="3" from-port="1" to-layer="4" to-port="1"/>
        <edge from-layer="4" from-port="2" to-layer="5" to-port="0"/>
        <edge from-layer="5" from-port="1" to-layer="7" to-port="0"/>
        <edge from-layer="6" from-port="1" to-layer="7" to-port="1"/>
        <edge from-layer="7" from-port="2" to-layer="9" to-port="0"/>
        <edge from-layer="8" from-port="1" to-layer="9" to-port="1"/>
        <edge from-layer="9" from-port="3" to-layer="11" to-port="0"/>
        <edge from-layer="10" from-port="1" to-layer="11" to-port="1"/>
        <edge from-layer="11" from-port="2" to-layer="12" to-port="0"/>
        <edge from-layer="9" from-port="2" to-layer="14" to-port="0"/>
        <edge from-layer="13" from-port="1" to-layer="14" to-port="1"/>
        <edge from-layer="14" from-port="2" to-layer="16" to-port="0"/>
        <edge from-layer="15" from-port="1" to-layer="16" to-port="1"/>
        <edge from-layer="16" from-port="2" to-layer="18" to-port="0"/>
        <edge from-layer="17" from-port="1" to-layer="18" to-port="1"/>
        <edge from-layer="18" from-port="3" to-layer="20" to-port="0"/>
        <edge from-layer="19" from-port="1" to-layer="20" to-port="1"/>
        <edge from-layer="20" from-port="2" to-layer="21" to-port="0"/>
        <edge from-layer="18" from-port="2" to-layer="23" to-port="0"/>
        <edge from-layer="22" from-port="1" to-layer="23" to-port="1"/>
        <edge from-layer="23" from-port="2" to-layer="25" to-port="0"/>
        <edge from-layer="24" from-port="1" to-layer="25" to-port="1"/>
        <edge from-layer="25" from-port="2" to-layer="27" to-port="0"/>
        <edge from-layer="26" from-port="1" to-layer="27" to-port="1"/>
</edges>
```

OpenVINO Inference Engine

– библиотека на C++ (Python / C), позволяющая приложению:

- прочитать модель из файла (IR) или создать с помощью API
- загрузить модель в модуль, работающий с конкретным устройством
- отправить данные для обработки (картинка, текст, звук, ...)
- получить результаты обработки (вероятности, координаты, ...)

Главная идея: единый API для разных устройств, выпускаемых Intel

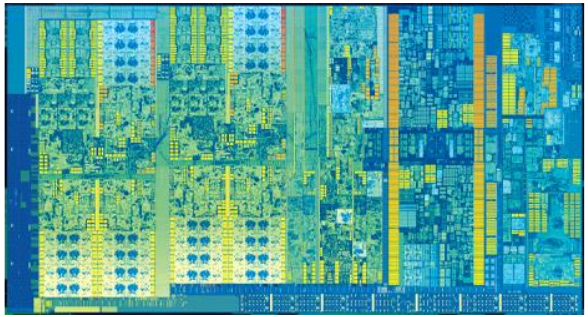
(оставляя возможность «тонкой настройки» для конкретных устройств)

Пример кода, использующего Inference Engine

```
// Базовый объект Inference Engine
Core ie;
// Чтение сети из файла IR (intermediate representation)
CNNNetwork network = ie.ReadNetwork(input_model);
// Определение имен входов и выходов
std::string input_name = network.getInputsInfo().begin()->first;
std::string output_name = network.getOutputsInfo().begin()->first;
// Загрузка модели в плагин
ExecutableNetwork executable_network = ie.LoadNetwork(network, device_name);
// Создание infer request'а
InferRequest infer_request = executable_network.CreateInferRequest();
// Задание входных данных
infer_request.SetBlob(input_name, imgBlob);
// Инференс
infer_request.Infer();
// Чтение выходных данных
Blob::Ptr output = infer_request.GetBlob(output_name);
```

Поддерживаемые устройства

Процессоры (CPU)



Графические карты (GPU)

Field-programmable gate array (FPGA)

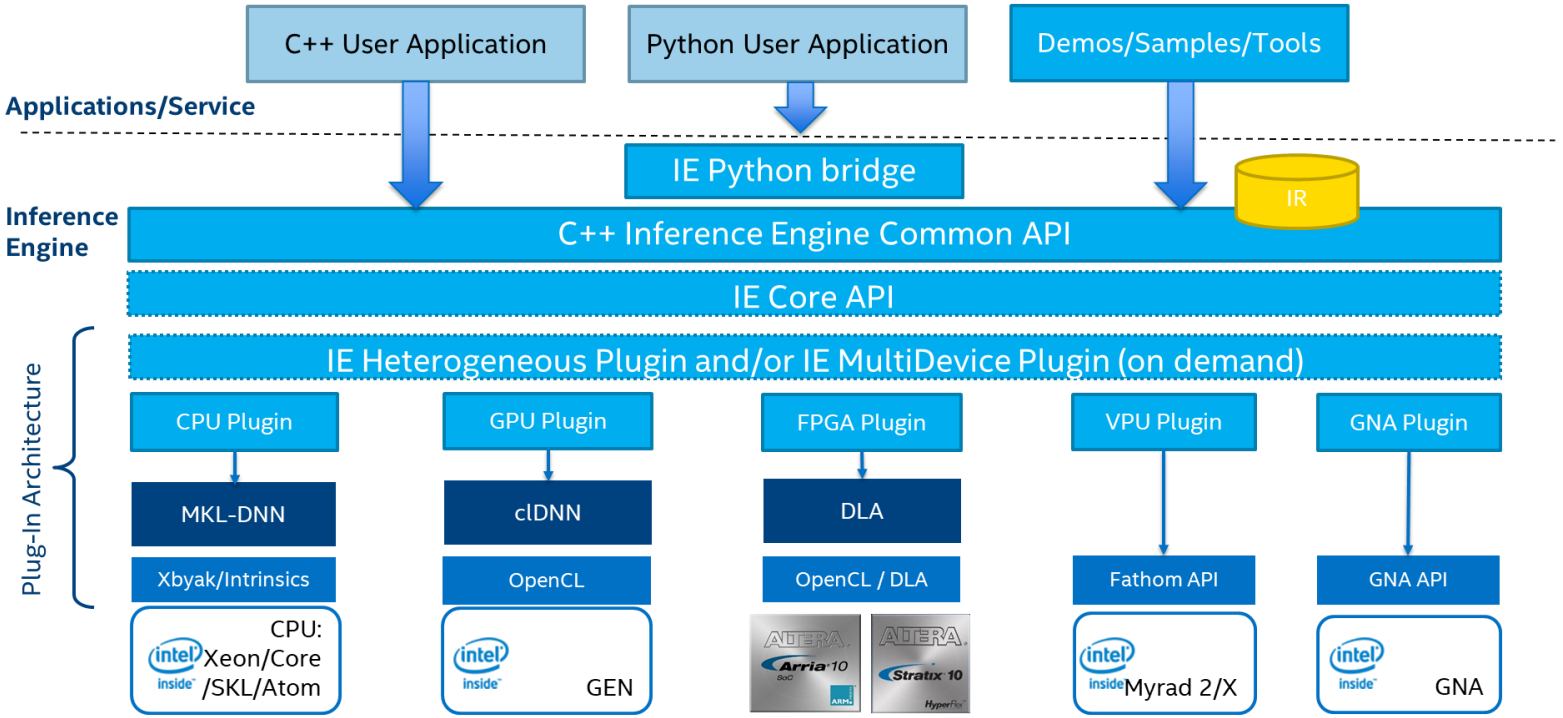
Процессоры машинного зрения (VPU)

Поддерживаемые устройства

Gaussian & Neural Accelerator (GNA)

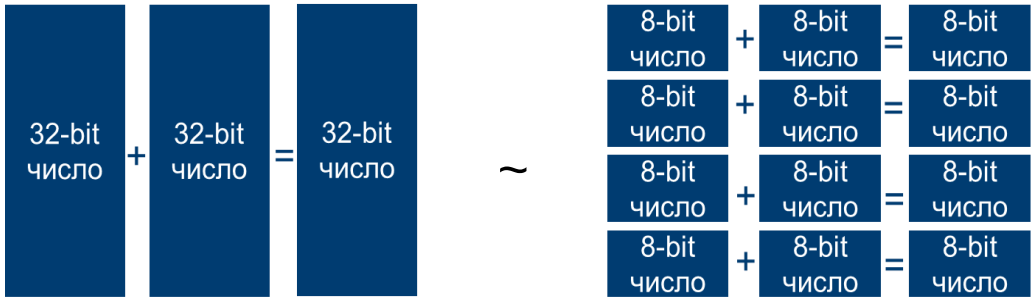
- маломощный сопроцессор для обработки звука

Программный стек при использовании Inference Engine



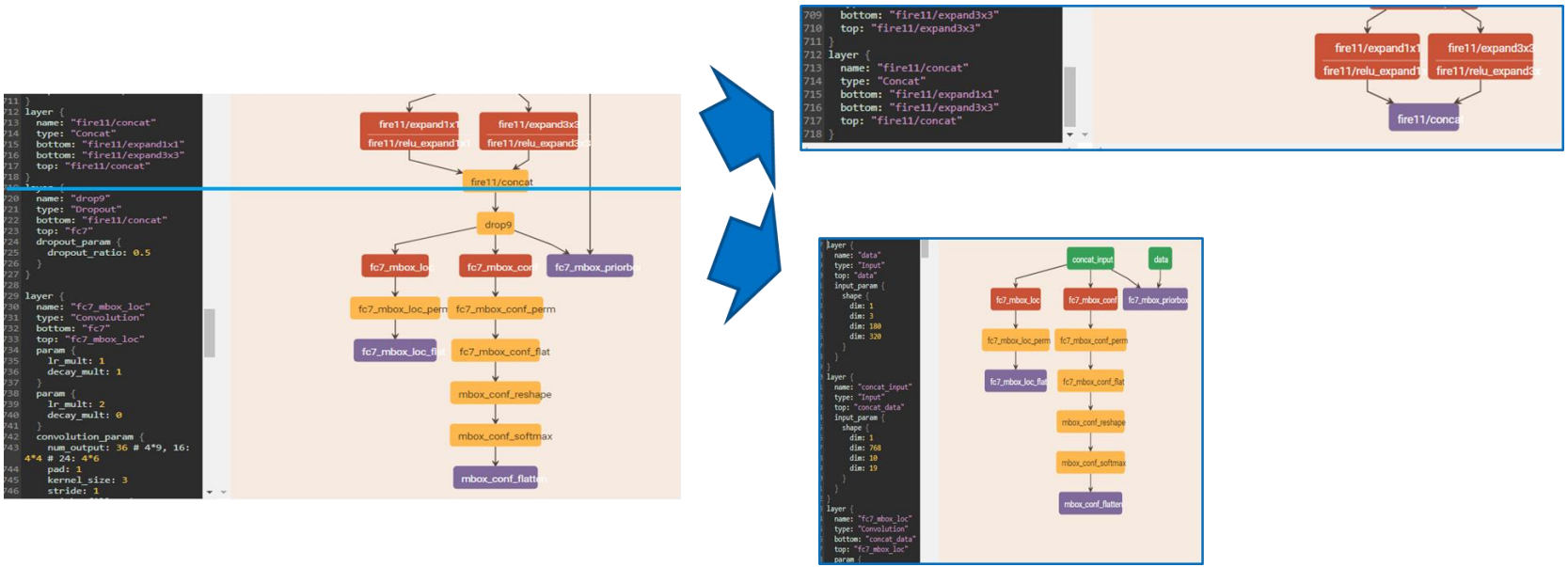
Оптимизация с помощью Inference Engine

- Оптимальное использование аппаратных особенностей
- Объединение нескольких операций в одну (fusing)
- Пакетная обработка данных (несколько картинок обрабатываются одновременно)
- «Стримы» (несколько экземпляров сети запускаются одновременно)
- Использование вычислений с меньшей разрядностью



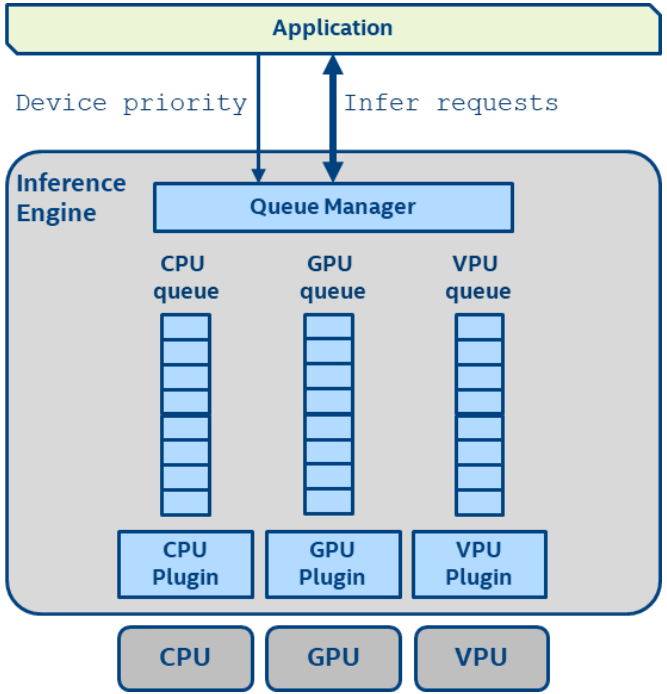
Гетерогенный режим

Не поддерживаемые слои отправляются на другое устройство (fallback)



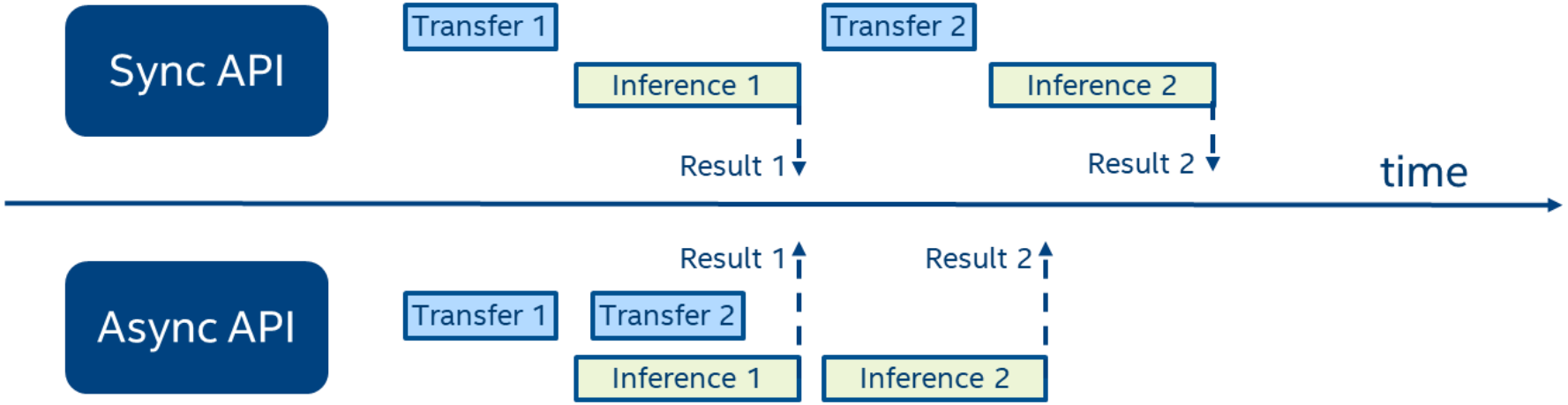
«Multi-device» режим

Задачи могут автоматически распределяться между несколькими устройствами



Синхронный и асинхронный режим

- Синхронный режим: выполнение блокируется до исполнения
- Асинхронный режим: выполнение продолжается; окончание отслеживается с помощью механизма callback



Дополнительные средства OpenVINO

[NEW] Post-training Optimization

- Reduce model size into low precision data types, such as INT8
- Reduces model size while also improving latency

Deployment Manager

- Generate an optimal, minimized runtime package for deployment
- Deploy with smaller footprint compared to development package

Model Analyzer

- Provides theoretical data on models: computational complexity (flops), number of neurons, memory consumption

Accuracy Checker

- Check for accuracy of the model (original and after conversion) to IR file using a known data set

Benchmark App

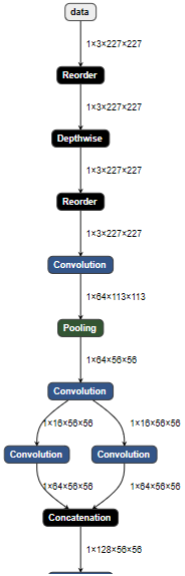
- Measure performance (throughput, latency) of a model
- Get performance metrics per layer and overall basis

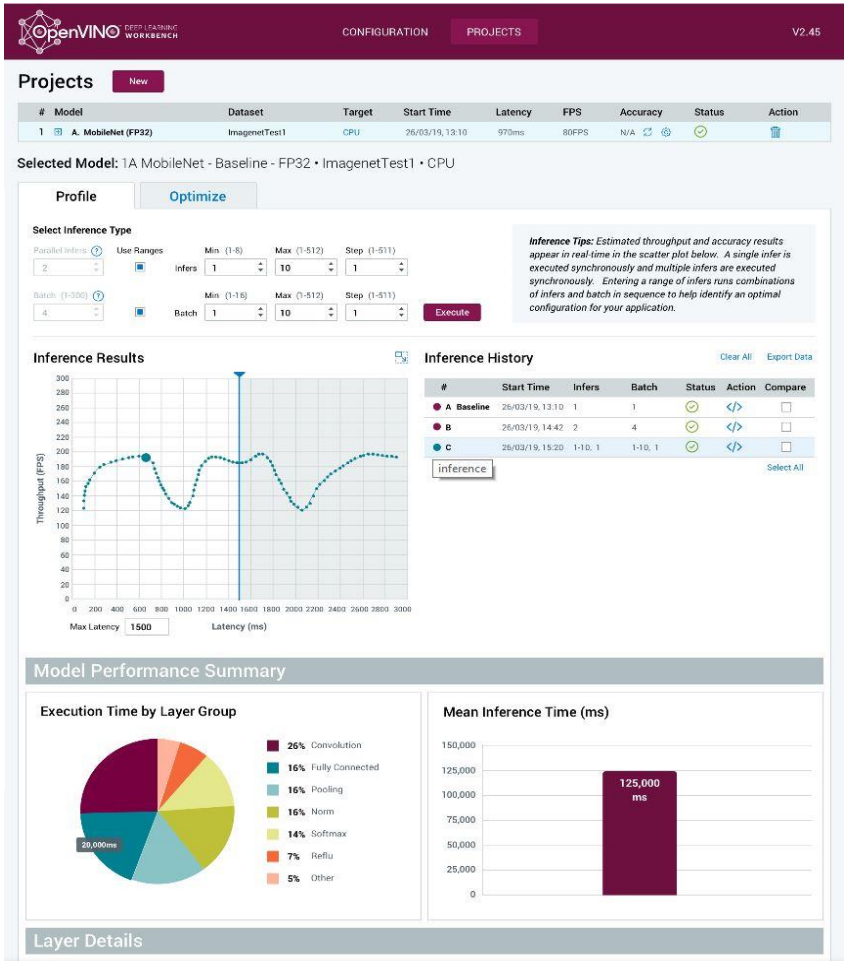
Model Downloader

- Provides an easy way of accessing a number of public models as well as a set of pre-trained Intel models

Deep Learning Workbench

- Конвертация сетей в IR
- Визуализация и профилировка сетей
- Подбор оптимальных параметров запуска
- Измерение точности сетей
- Работа с Open Model Zoo





OpenVINO samples

OpenVINO поставляется вместе с примерами, демонстрирующими использование OpenVINO для различных задач:

- классификация
- обнаружение объектов
- автоматическое распознавание голоса
- оценка performance для конкретных моделей
- ...

OpenVINO Open Model Zoo

Computer Vision

Object detection

Object recognition

Reidentification

Semantic segmentation

Instance segmentation

Human pose estimation

Image processing

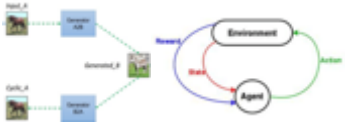
Audio, Speech, Language

Text detection

Text recognition

Recommender

Action recognition



Other

(Data Generation, Reinforcement Learning)

Compression models

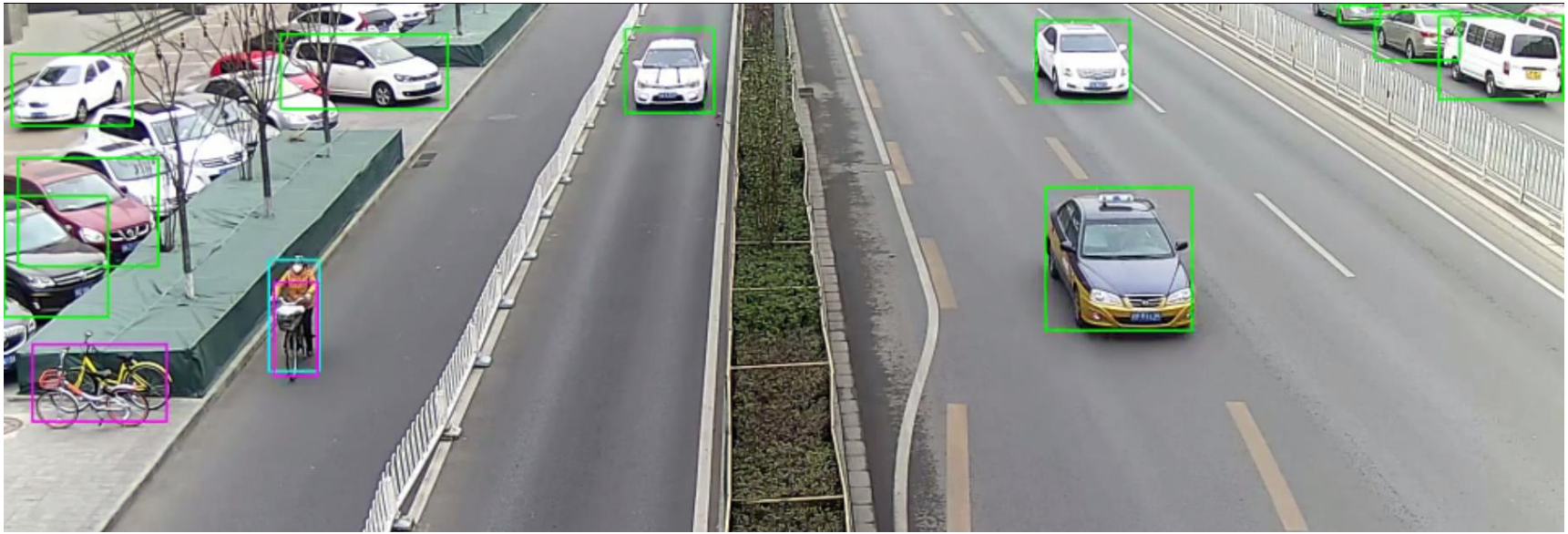
Image retrieval

And more..

https://github.com/opencv/open_model_zoo

Модели от Intel – Open Model Zoo (1)

Open Model Zoo – набор готовых бесплатных нейронных сетей, натренированных компанией Intel



Модель: person-vehicle-bike-detection-crossroad-1016

Модели от Intel – Open Model Zoo (2)

Type: car
Color: black

Модель: vehicle-attributes-recognition-barrier-0039

Модели от Intel – Open Model Zoo (3)

Модель: person-reidentification-retail-0288

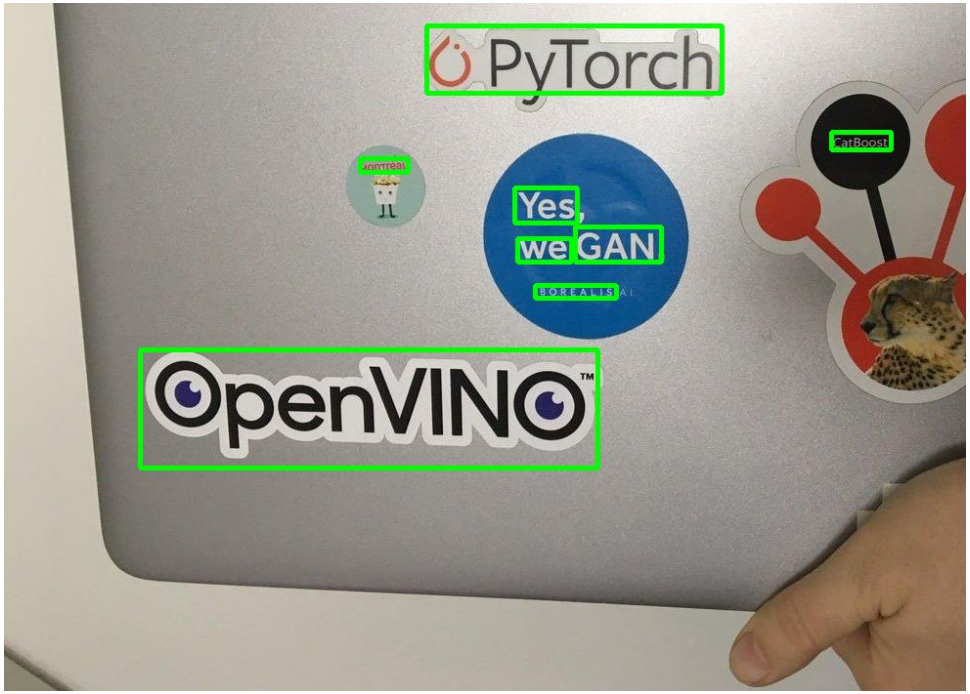
Модели от Intel – Open Model Zoo (4)

Модель: semantic-segmentation-adas-0001

Модели от Intel – Open Model Zoo (5)

Модель: instance-segmentation-security-0010

Модели от Intel – Open Model Zoo (6)



Модель: text-detection-0004

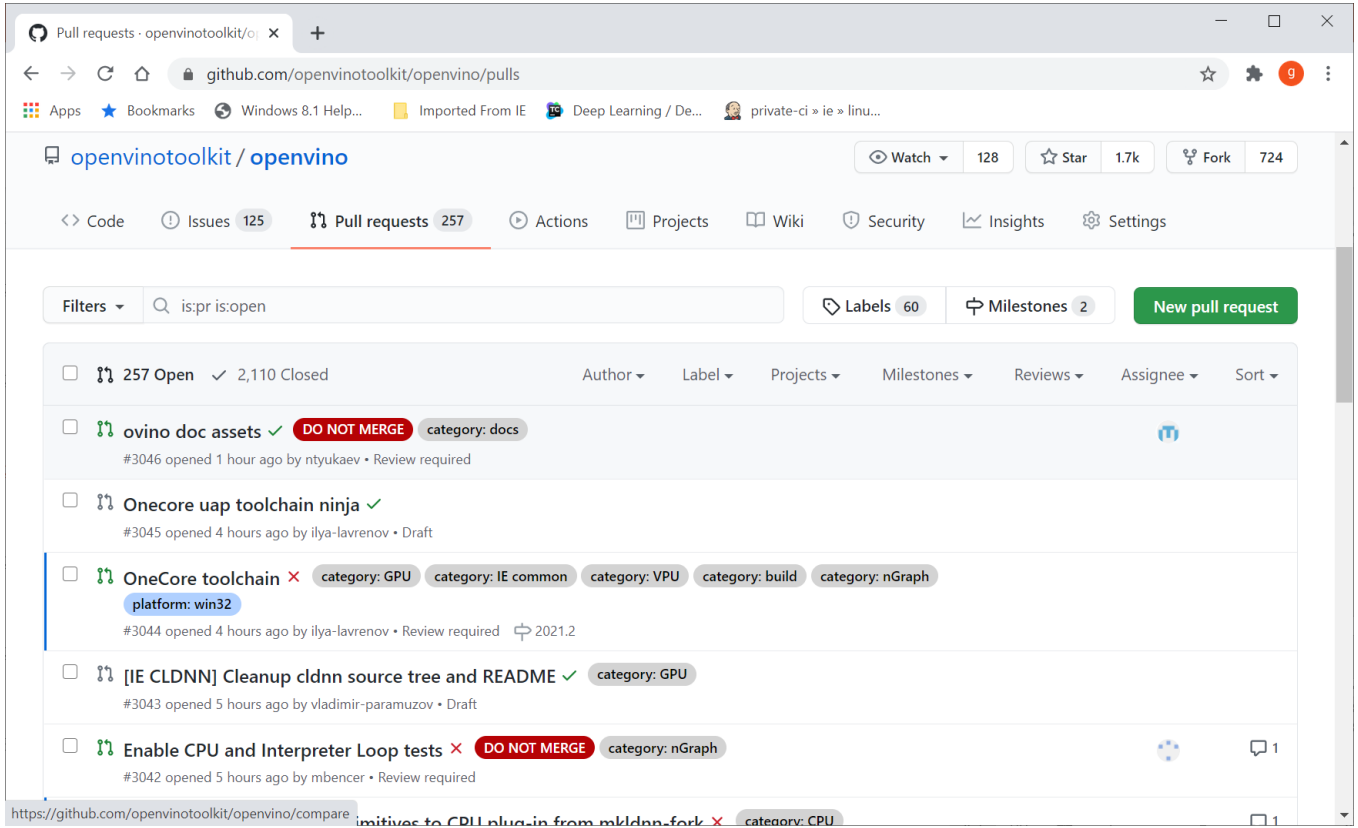
Модели от Intel – Open Model Zoo (7)

Модель: driver-action-recognition-adas-0002-decoder

Способы распространения OpenVINO

- Бинарный дистрибутив
- APT
- YUM
- Anaconda
- PyPI
- Docker Hub
- **Open source**

OpenVINO на GitHub



«Экосистема» OpenVINO

- Открытая архитектура, открытый исходный код
- Дополнительные средства:
  - NNCF (neural network compression framework)
  - Training extensions

  } Поддержка тренировки

  - OpenVINO Model Server
  - DevCloud for the Edge

  } Удаленное выполнение

  - DL Streamer (поддержка OpenVINO в gstreamer)

```
gst-launch-1.0 filesrc location=cut.mp4 ! decodebin ! videoconvert ! gvadetect
model=face-detection-adas-0001.xml ! gvaclassify model=emotions-recognition-retail-
0003.xml model-proc=emotions-recognition-retail-0003.json ! gvawatermark ! xvimagesink
sync=false
```

Дополнительные материалы

Тренинги

- Курсы по Deep Learning на Coursera

Книги

- Николенко С.И., Кадурин А. А. Глубокое обучение. Погружение в мир нейронных сетей
- Н.Будума, Н.Локашо. Основы глубокого обучения

Ресурсы в интернете

- Документация по OpenVINO
- Papers with Code

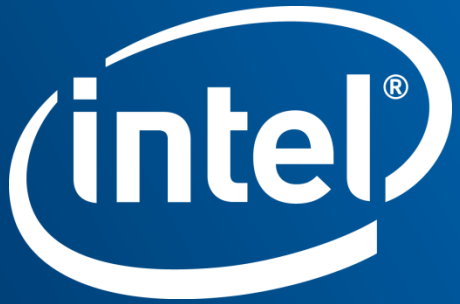
We are hiring!

У нас много сложной и интересной работы!

JR0205104 – Deep Learning Software Engineering Intern (OpenVINO, GNA)

JR0152999 – Deep Learning Software Engineer (OpenVINO, GNA)

Контакт: denis.orlov@intel.com


intel®