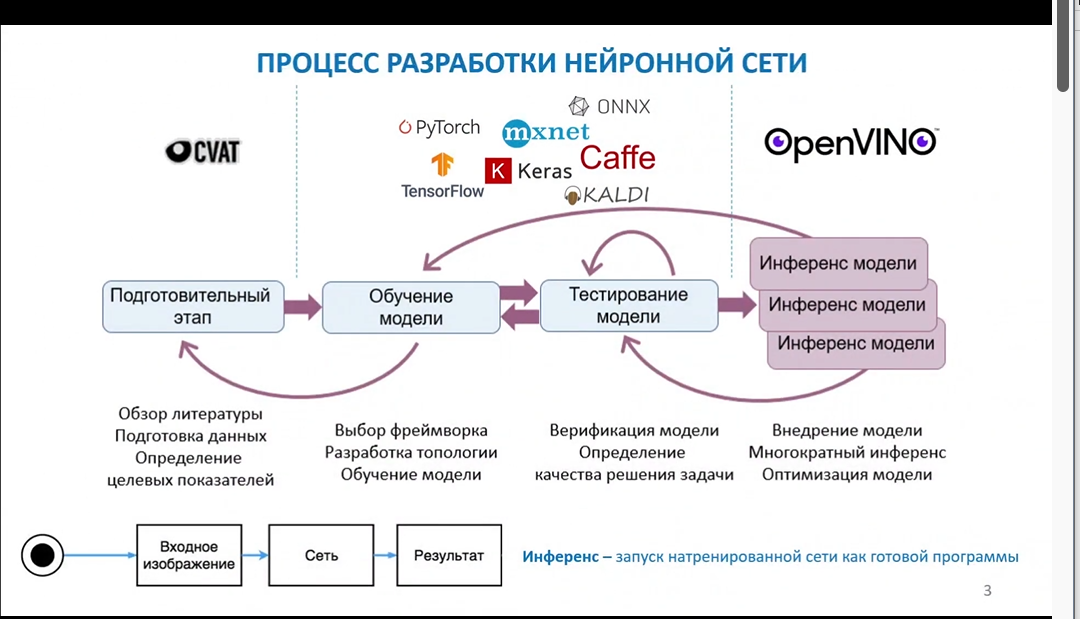
***Суть***

OpenVINO – если говорить простым языком и выделить самую главную мысль, то это набор инструментов, который призван максимально эффективно запустить нейронную сеть на вашем железе, то есть выполнить какие-то оптимизации, чтобы все работало максимально быстро на предоставляемом железе; набор полезных инструментов, которые объединяют обработку видео, компьютерное зрение, обработку естественного языка, машинное обучение и оптимизацию нейронных сетей в единый пакет.

***Процесс разработки нейронной сети***

******

Этапы:

Подготовительный – чтение литературы на предмет поиска близких для решения задачи архитектур; определение целевых показателей, которые необходимо достичь по точности; подготовка данных – самый важный этап, так как не всегда существует датасет, который подходит под конкретную задачу, иногда его приходится создавать самостоятельно, для создания датасета, относящегося к CV у Intel существует инструмент CVAT для разметки данных

Обучение и тестирование модели – выбираем фреймворк, разработка топологии сети, либо выбрать готовую обучить модель, определить качество решения задачи. Этим занимаются фреймворки.

Исполнение – инференс модели. Инференс – запуск готовой натренированной сети как готовой программы. OpenVINO не занимается ни обучением, ни тестированием, а направлен только на инференс.

***Фреймворки и OpenVINO***



Фреймворки можно использовать для инференса, но делать этого н стоит, так как:

Фреймворки обычно тяжеловесны – содержат большое количество библиотек, большое количество кода, которое направлено как раз на обучение и тестирование моделей

Не имеют никаких оптимизаций

Фреймворки часто не задействуют весь потенциал аппаратного обеспечения – например в TF, чтобы задействовать GPU надо устанавливать дополнительные драйвера, то есть в фреймворк они не встроены

OpenVINO в свою очередь:

Использует облегченную среду исполнения, которая отвечает только за инференс модели, то есть исполнение нейронной сети и не тащит за собой какие-то модули, отвечающие за обучение и тестирование модели

Софтверные оптимизации – например оптимизация топологии сети

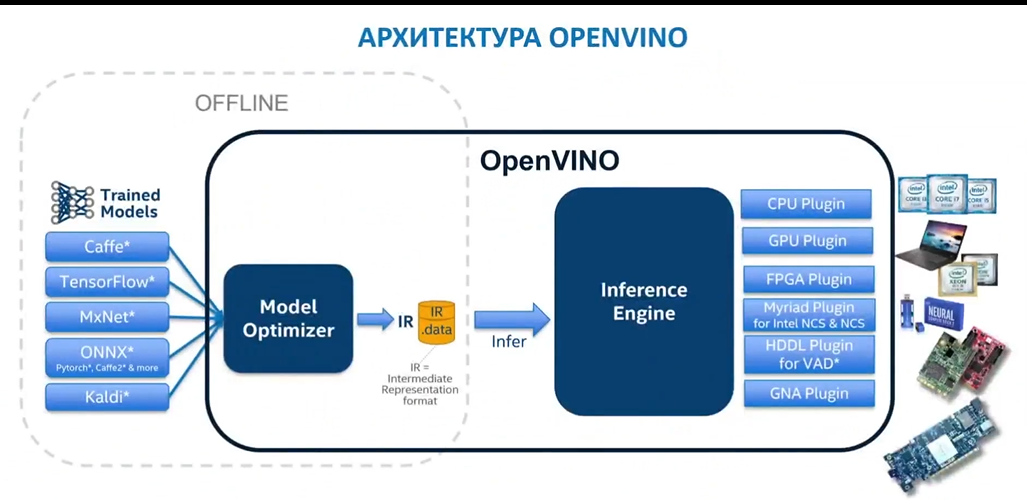
Оптимизации под железо Intel

Один и тот же API (Питон, C++) для всех целей

Зоопарк открытых моделей – можно найти модели обученные Intel и сторонними людьми, но оптимизированные для OpenVINO

Пример кода и демонстрации

***Архитектура OpenVINO***

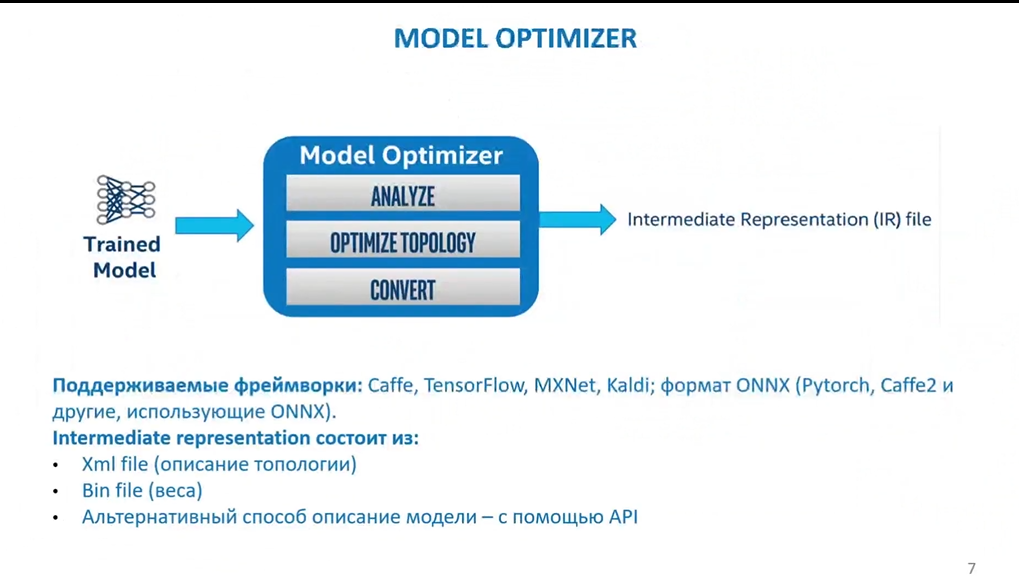


Архитектура глобально делится на две части:

Model Optimizer – набор скриптов, написанных на Python, которые позволяют перевести вашу сеть, обученную в каком-то фреймворке в IR-формат. IR – специальный формат, разработанный Intel (промежуточное представление), с которым дальше работает Inference Engine и все инструменты, которые есть в OpenVINO.

Inference engine (дословно механизм инференса) – отвечает за как раз таки исполнение сетей на каком-то железе (подгружаются плагины, которые отвечают за каждой железо и нейросеть исполняется)

***Model Optimizer***



MO получает на вход модель, а на выходе выдает файл в формате IR

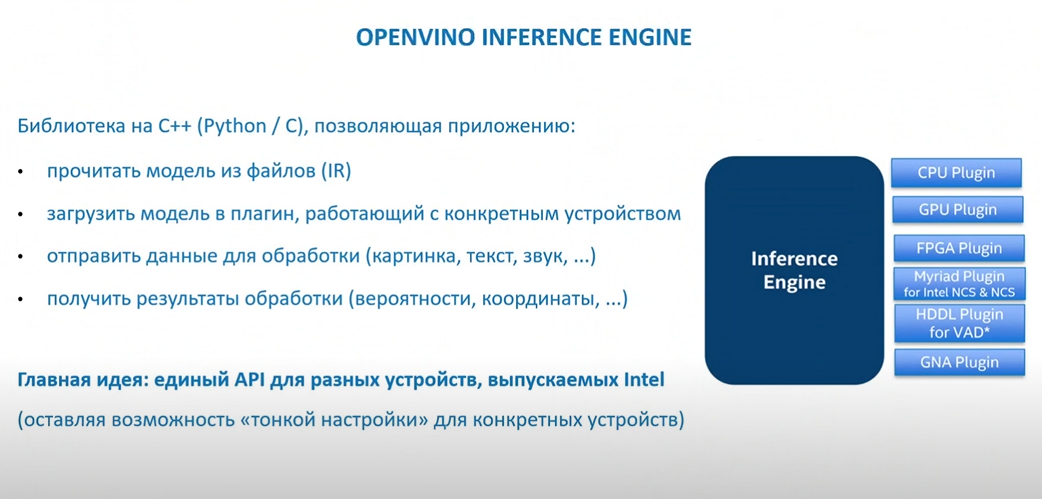
IR состоит из двух частей:

Xml, в котором находится описание топологии сети (какие слои, какого типа, как они соединены)

Bin файл, где в бинарном формате описаны веса модели

Однако есть еще один способ загрузки модели в IE – минуя MO можно это сделать с помощью API, каким-то программным способом, в общем именно поэтому сейчас в MO становится все меньше оптимизаций, потому что если модель будет подгружаться с помощью API, то теряются эти оптимизации, потому что MO не задействуется, поэтому все оптимизации сейчас переходят в IE и другие сторонние инструменты и сейчас MO, не смотря на свое название становится больше конвертером, чем оптимизатором.

***Inference Engine***

******

IE – библиотека, написанная на C++, позволяющая приложению:

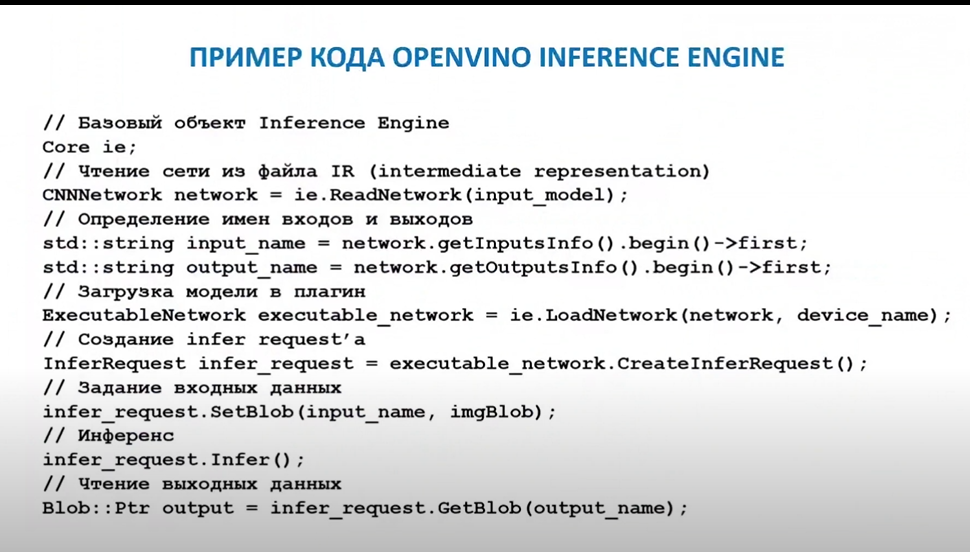
Прочитать модель из файлов (IR)

Загружает модель в плагин, работающий с конкретным устройством

Отправить данные для обработки

Получить результаты обработки

Главная идея – это единый API для разных устройств, выпускаемых Intel, позволяющий запускать нейронные сети, оставляя при этом возможность на каждом устройстве выполнять какую-то тонкую настройку, то есть передавать какие-то специфические параметры при запуске сети, если это необходимо.



***Поддерживаемые устройства***

******

OpenVINO поддерживает широкий набор устройств, разработанных Intel:

Процессоры (CPU) – так как с большой вероятностью поддержка слоев и операций присутствует на процессоре; поддерживаются все процессоры Intel, начиная с 4-го поколения (насчет процессоров AMD: так архитектура Intel и AMD X86 – схожая, то процессоры AMD по крайней мере года полтора назад работали, то есть они позволяли на себе запускать OpenVINO и работать с сетями, однако никто естественно не гарантирует то, что все будет максимально оптимизировано, так как все таки архитектура у AMD немного другая, поэтому возможность запустить OpenVINO скорее всего будет, но насколько эффективно неизвестно)

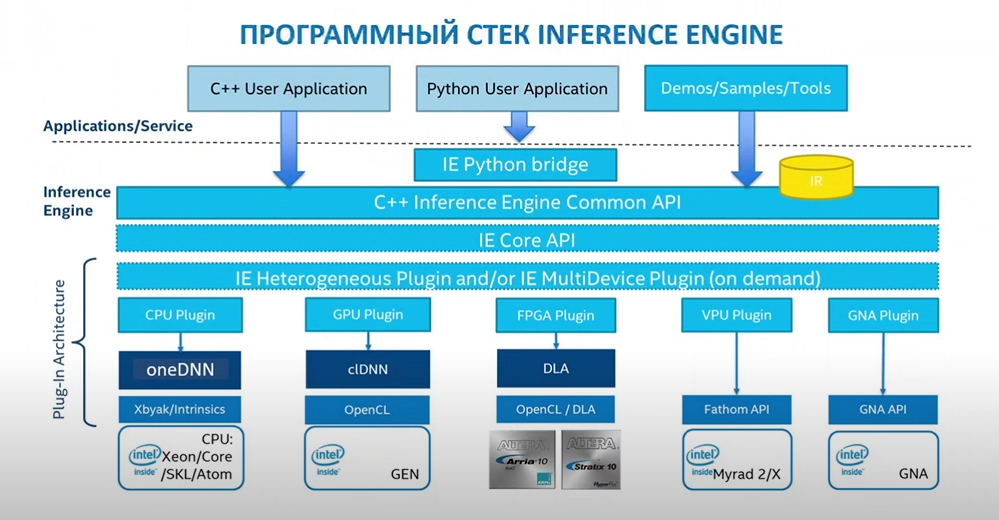
Видеокарты (GPU) – поддерживаются встроенные видеоядра в процессоре и прирост от использования графического ускорителя (даже встроенного) значительный

FPGA (программируемая логическая интегральная схема)

VPU (процессоры машинного зрения)

GNA (процессор для обработки звука)

***Программный стек IE***

******

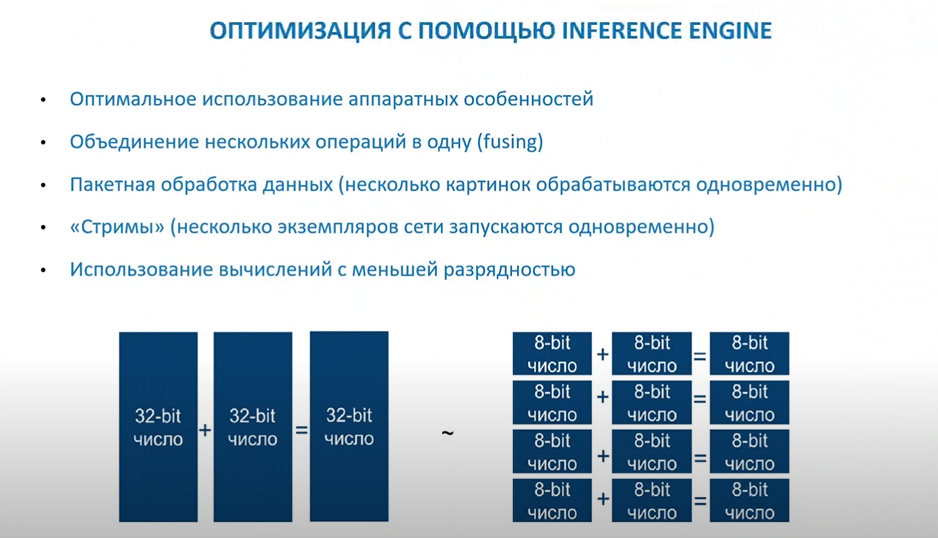
Основная идея в том, что приложение пользователя, написанное на Python/C++ через некий API, позволяет включить в работу один из плагинов устройства, на котором запускается сеть.

С одной стороны в IE есть интерфейс, которым пользуются пользователи для загрузки своего кода, с другой стороны есть интерфейс, который позволяет добавить поддержку любого устройства.

Каждый из плагинов, который относится к каждому конкретному устройству может иметь свои низкоуровневые библиотеки.

В общем IE имеет множество разноуровневых задач: есть какие-то общие задачи, общие оптимизации, которые не зависят от устройства (например оптимизация топологии сети), есть низкоуровневые оптимизации под конкретное устройство и есть совсем низкоуровневые оптимизации, уже перед включением железа в работу (например, примитивы из OpenCL или средства парализации, относящиеся к конкретному процессору). За счет всех оптимизаций IE как раз достигает высокой производительности.

***Оптимизация с помощью IE***

******

Оптимальное использование аппаратных особенностей – то, что происходит на самом низком уровне программного стека.

Объединение нескольких операций в одну (fusing)

Обработка батчей – обработка нескольких картинок одновременно

Запуск нескольких экземпляров сети одновременно

Использование вычислений с меньшей разрядностью – позволяет при небольшой потере точности выполнить гораздо больше операций за такт, например. Потери точности: при переходе из FP32 в FP16 теряются буквально десятые доли процентов, при переходе в INT8 теряется примерно полтора процента.

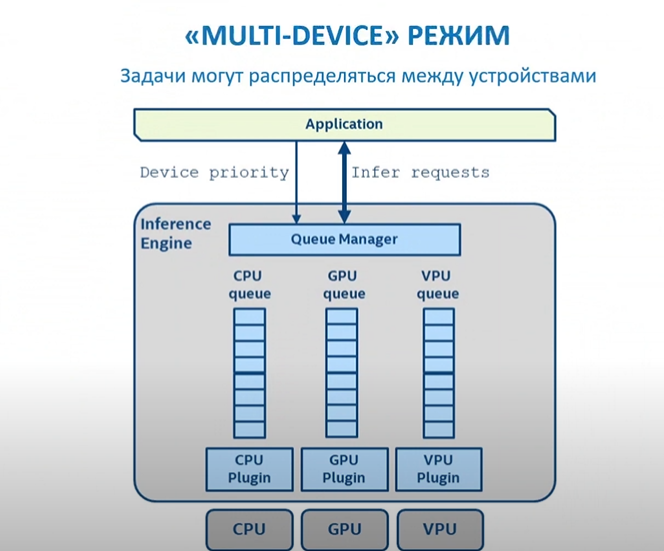
***Гетерогенный режим работы***

******

Данный режим позволяет отправлять не поддерживаемые слои на другое устройство (fallback).

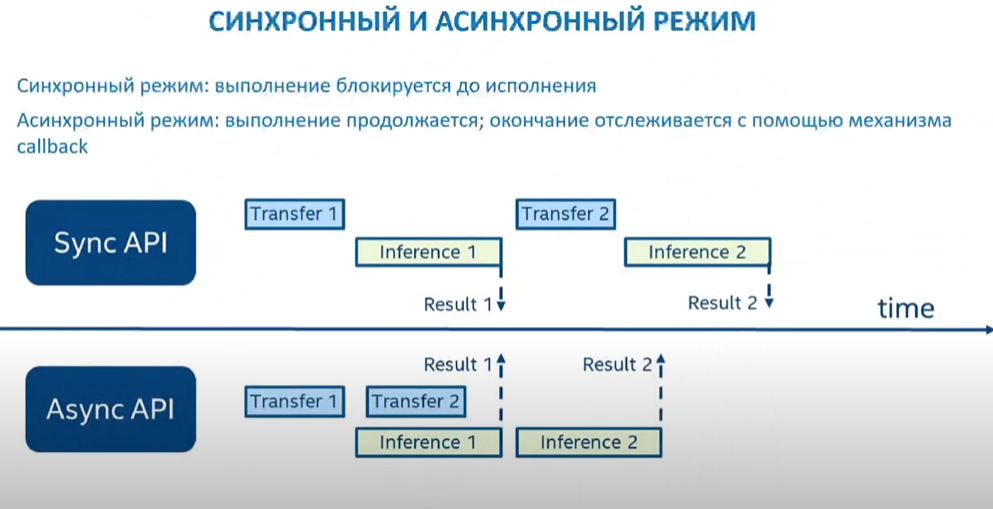
Проблема в том, что не все железо всегда поддерживает все операции. Например, GPU может не поддерживать какой-то слой или операцию, и в таком случае для достижения максимальной производительности эти операции будут отправляться на процессор.

***Multi-device режим***

******

Данный режим позволяет исполнять нейронную сеть сразу на нескольких устройствах. IE будет самостоятельно распределять свои операции.

***Синхронный и асинхронный режим***

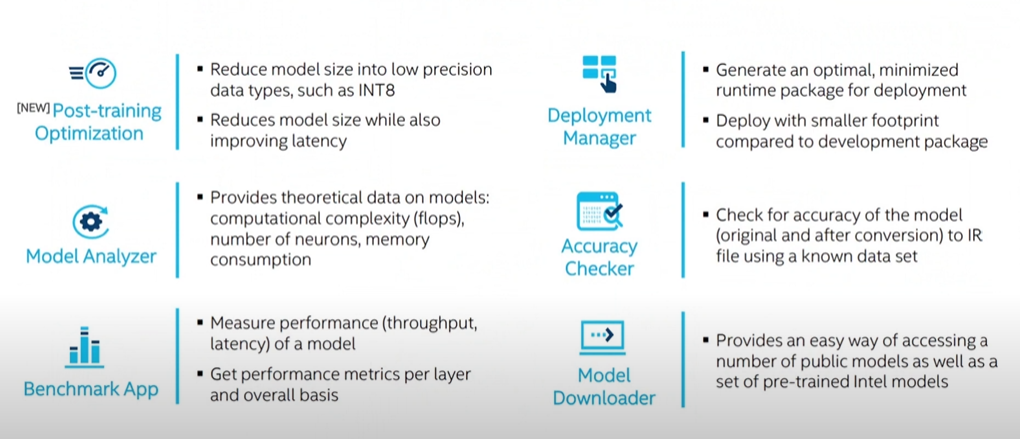


OpenVINO также поддерживает синхронный и асинхронный режим работы.

Асинхронный режим – накидываются запросы на обработку и не дожидаясь выполнения этого запроса вызывается следующий, таким образом они накапливаются. Окончание отслеживается с помощью механизма callback.

Синхронный режим – выполнение блокируется до исполнения предыдущего запроса.

***Дополнительные утилиты***

******

Post-training Optimization – позволяет оптимизировать нейронную сеть с использованием целочисленной арифметики, то есть например перейти из FP16 в INT8. Проблема в том, что теряется точность при переходе из чисел с плавающей запятой в целые числа. Возникает проблема нахождения scale-фактора, коэффициента преобразования из вещественных чисел в целые. Если взять слишком маленькое, то получится слишком маленький входной диапазон, а если слишком большое, то число может банально не влезть в INT и будет переполнение, что плохо скажется на работе сети. Данная утилита Post-training toolkit запускается на реальных данных и для каждого слоя сети выясняет диапазон входных данных и таким образом получается подобрать коэффициент преобразования scale.

Model Analyzer – позволяет узнать теоретический предел производительности, который можно достичь с вашим железом и сравнить с реальным результатом.

Benchmark App – позволяет с помощью специфичных методологий, разработанных Intel, измерить скорость работы сети.

Deployment Manager – позволяет минимизировать набор компонентов, который требуется для разработки, то есть позволяет устанавливать не весь пакет OpenVINO, а какие-то его определенные инструменты.

Accuracy Checker – позволяет измерить точность модели.

Model Downloader – позволяет загружать модели из открытого зоопарка OpenVINO

1. <https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html>
2. <https://docs.openvino.ai/latest/openvino_docs_install_guides_installing_openvino_windows.html>
3. <https://docs.openvino.ai/2021.4/openvino_docs_IE_DG_supported_plugins_MULTI.html?sw_type=switcher-python>
4. <https://gist.github.com/dkurt/bc567159e5173a6e86c5781badff9ac6>
5. <https://delta-course.org/docs/IntelCVSchool/IntelCVSchoolOpenVINO.pdf>
6. <https://www.youtube.com/watch?v=kY9nZbX1DWM&list=PLDKCjIU5YH6jMzcTV5_cxX9aPHsborbXQ>
7. <https://github.com/opencv/open_model_zoo>
8. <https://habr.com/ru/company/intel/blog/549634/>
9. <https://habr.com/ru/company/intel/blog/433772/>
10. <https://neptune.ai/blog/image-processing-python>