Моделирование и анализ данных
2026, *16*(3), 104—120.
https://doi.org/10.17759/mda.2026160305
ISSN: 2219-3758 (печатный)
ISSN: 2311-9454 (online)

Modelling and Data Analysis
2026, *16*(3), 104—120.
https://doi.org/10.17759/mda.2026160305
ISSN: 2219-3758 (print)
ISSN: 2311-9454 (online)

Научная статья | Original paper

УДК 004.932:004.85

Сравнение модели YOLO11n на встраиваемых платформах Orange Pi5 и Raspberry Pi4B

М.А. Кукушкин ✉, **П.А. Ухов**
Московский авиационный институт (национальный исследовательский университет)
Москва, Российская Федерация
✉ cnegbyj99@mail.ru

Резюме

Контекст и актуальность. Современные тренды развития технологий устремляются в сторону переноса вычислений с облачных платформ на встраиваемые устройства. В связи с этим растет необходимость определения оптимального стека аппаратных и программных технологий для решения задач компьютерного зрения. Ключевым фактором эффективности становится не только пиковая производительность чипа (TOPS), но и архитектура вычислительного графа, определяющая степень загрузки специализированных нейропроцессоров, а также формат и глубина квантизации, напрямую влияющие на пропускную способность шины памяти и тактовую частоту выполнения сверточных слоев. **Цель**. Определить оптимальную аппаратную конфигурацию для задач компьютерного зрения в реальном времени по критериям пропускной способности, задержки, точности и удобства интеграции, выявив зависимость итоговой производительности от распределения операторов графа между NPU/TPU и CPU. **Методы и материалы**. В работе выполнено прямое экспериментальное сравнение двух встраиваемых платформ для исполнения модели детекции объектов YOLO11n: одноплатного компьютера Orange Pi 5 со встроенным нейропроцессором Rockchip RK3588S (6 TOPS, INT8) и связки Raspberry Pi 4B с внешним USB-ускорителем Google Coral Edge TPU (4 TOPS, INT8). Модель YOLO11n конвертировалась из исходного файла PyTorch в целевые форматы RKNN и TensorFlow Lite. Особое внимание уделено анализу карты операций (операторного графа): для Orange Pi оценена эффективность слияния слоев (Layer Fusion); для Edge TPU — влияние фрагментации графа, приводящей к передаче промежуточных тензоров между USB-интерфейсом и процессором ARM при выполнении неподдерживаемых операций. Замеры выполнялись при идентичных условиях предобработки. **Результаты**. На Orange Pi 5 достигнута устойчивая частота кадров 43–54FPS (разрешение

640×640) со сквозной задержкой 19–23 мс; высокая скорость обеспечивается полным выполнением графа внутри NPU без обращений к CPU, а также оптимизированным под INT8 форматом весов, минимизирующим кэш-промахи. Точность mAP@0,5:0,95 при INT8-квантизации сохраняется на уровне ~39%. На Raspberry Pi 4B с Edge TPU при разрешении 448×448 задержка составила 121,5 мс (~8,2FPS), при этом 45% операций (включая операции извлечения каналов и нормализации) принудительно выполняются на CPU из-за несовместимости формата TFLite с фиксированным вычислительным ядром TPU, что создает узкое место (bottleneck) на шине USB 2.0 и ограничивает масштабирование. **Выводы**. Выигрыш Orange Pi в скорости достигает 6–7 крат при более высоком разрешении, NPU демонстрирует значительный запас по загрузке. Ключевым фактором успеха платформы RK3588S является не только пиковая производительность, но и корректная топология вычислительного графа, позволяющая разместить все тензорные операции в непрерывном адресном пространстве NPU с минимальным числом инструкций ветвления. Напротив, архитектурные ограничения Edge TPU в части формата квантизации приводят к необходимости редукции разрешения. Совокупность результатов позволяет рекомендовать Orange Pi 5 как предпочтительную платформу для построения производительных edge-решений с детекцией объектов в реальном времени, однако при выборе стека необходимо учитывать, что эффективность напрямую зависит от возможности инструментов RKNN сохранить целостность графа без выпадения операторов в CPU.

Ключевые слова: компьютерное зрение, YOLOv11, Orange Pi 5, Rockchip RK3588S, RKNN, TPU, Raspberry Pi

Для цитирования: Кукушкин, М.А., Ухов, П.А. (2026). Сравнение модели YOLO11n на встраиваемых платформах Orange Pi5 и Raspberry Pi4B. *Моделирование и анализ данных, 16*(3), 104—120. https://doi.org/10.17759/mda.2026160305

Comparison of the YOLO11n model on embedded platforms Orange Pi5 and Raspberry Pi4B

M.A. Kukushkin ✉, **P.A. Ukhov**
Moscow Aviation Institute (National Research University), Moscow, Russian Federation
✉ cnegbyj99@mail.ru

Abstract

Context and relevance. Modern trends in technology development are moving towards the transfer of computing from cloud platforms to embedded devices. In this regard, there is a growing need to determine the optimal stack of hardware and software technologies for solving computer vision problems. The key efficiency

factor is not only the peak performance of the chip (TOPS), but also the architecture of the computational graph, which determines the degree of utilization of specialized neuroprocessors, as well as the format and depth of quantization, which directly affect the bandwidth of the memory bus and the clock frequency of convolutional layers. **Goal**. To determine the optimal hardware configuration for real-time computer vision tasks based on the criteria of throughput, latency, accuracy and ease of integration, revealing the dependence of the final performance on the distribution of graph operators between the NPU/TPU and the CPU. **Methods and materials**. The paper provides a direct experimental comparison of two embedded platforms for executing the YOLO11n object detection model: a single-board Orange Pi 5 computer with a built-in Rockchip RK3588S neuroprocessor (6 TOPS, INT8) and a Raspberry Pi 4B bundle with an external Google Coral Edge TPU USB accelerator (4 TOPS, INT8). The YOLO11n model has been converted from the PyTorch source file to the target formats RKNN and TensorFlow Lite. Special attention is paid to the analysis of the operation map (operator graph): The efficiency of Layer Fusion has been evaluated for Orange Pi; for Edge TPUs, the effect of graph fragmentation, which leads to the transfer of intermediate tensors between the USB interface and the ARM processor when performing unsupported operations. The measurements were performed under identical pretreatment conditions. **Results**. The Orange Pi 5 has achieved a stable frame rate of 43–54FPS (640×640 resolution) with an end-to-end delay of 19–23 ms; high speed is ensured by full graph execution inside the NPU without accessing the CPU, as well as an INT8-optimized scale format that minimizes cache misses. Accuracy mAP@0,5:0,95 with INT8 quantization, it remains at ~39%. On Raspberry Pi 4B with Edge TPU at a resolution of 448×448, the delay was 121,5 ms (~8,2FPS), while 45% of operations (including channel extraction and normalization operations) are forced to be performed on the CPU due to incompatibility of the TFLite format with a fixed TPU computing core, which creates a bottleneck on the bus USB 2.0 and limits scaling. **Conclusions**. The Orange Pi's gain in speed reaches 6–7 times at a higher resolution, and the NPU demonstrates a significant load margin. The key success factor of the RK3588S platform is not only peak performance, but also the correct topology of the computational graph, which allows placing all tensor operations in a continuous NPU address space with a minimum number of branch instructions. On the contrary, the architectural limitations of Edge TPU in terms of the quantization format lead to the need for resolution reduction. The totality of the results allows us to recommend Orange Pi 5 as the preferred platform for building productive edge solutions with real-time object detection, however, when choosing a stack, it must be borne in mind that efficiency directly depends on the ability of RKNN tools to maintain graph integrity without dropping operators in the CPU.

Keywords: computer vision, YOLOv11, Orange Pi 5, Rockchip RK3588S, RKNN, TPU, Raspberry Pi

For citation: Kukushkin, M.A., Ukhov, P.A. (2026). Comparison of the YOLO11n model on embedded platforms Orange Pi5 and Raspberry Pi4B. *Modelling and Data Analysis, 16*(3)*,* 104—120. (In Russ.). https://doi.org/10.17759/mda.2026160305

Введение

Выбор аппаратной платформы для исполнения современных детекторов на периферийных устройствах сводится к компромиссу между пропускной способностью, задержкой, точностью и сложностью интеграции. На момент написания статьи два решения претендуют на роль массового вычислительного ядра в задачах реального времени: одноплатный компьютер Orange Pi 5(Orange Pi, 2023) с интегрированным NPU на базе Rockchip RK3588 (до 6 TOPS, INT8) и связка Raspberry Pi 4 Model B (Raspberry Pi OS, n.d.) с внешним USB-ускорителем Google Coral (4 TOPS, INT8). Обе платформы предполагают исполнение нейросетевого вывода вне центрального процессора, однако используют принципиально различные программные экосистемы — RKNN и TensorFlow Lite делегат Edge TPU соответственно.

Цель настоящей работы — прямое экспериментальное сравнение указанных конфигураций при выполнении задачи детекции объектов моделью YOLO11n (Jocher, 2024). Сравнение проводится по трём ключевым метрикам: точность (mAP@0,5 на стандартном валидационном подмножестве COCO (COCO Dataset, n.d.)), частота кадров (FPS) и полная сквозная задержка (от захвата изображения до получения координат). Для обеих платформ модель конвертируется из общего исходного файла PyTorch в целевые форматы (RKNN и TFLite) с применением рекомендованных производителем инструментов, без внесения структурных изменений в вычислительный граф. Измерения выполняются при идентичном входном разрешении (640×640) и одинаковом наборе тестовых изображений, что исключает влияние внешних факторов.

Данная работа предъявляет поэтапную методику портирования, фиксирует конфигурации окружений и версии библиотек, приводит методику измерения задержек и подробные таблицы результатов. Итоговый вывод содержит рекомендации по выбору платформы для практических развертываний с учетом не только пиковых цифр, но и удобства сопровождения, доступности инструментов и стабильности работы в длительных нагрузочных тестах.

Обзор модели нейросети

В качестве единой модели для экспериментов выбрана YOLO11n (Jocher, 2024) — самая лёгкая версия новейшего семейства детекторов Ultralytics (Ultralytics Documentation, n.d.), специально оптимизированная для исполнения на периферийных устройствах с аппаратными ускорителями. При всего 2,6 миллионах параметров и вычислительной сложности порядка 6,5 GFLOPS (для разрешения 640×640) модель сохраняет конкурентоспособную точность, что делает её идеальным кандидатом для развертывания на платформах с ограниченными ресурсами. Архитектура наследует сильные стороны предшественников: облегчённый бэкбон на основе CSPNet с эффективными блоками C3k2, многоуровневую пирамиду признаков PAN с улучшенной агрегацией контекста и стандартные сверточные блоки, лишённые

вычислительно избыточных трансформерных элементов. По сравнению с YOLOv8 модель демонстрирует рост mAP на 0,5–1,0% при сопоставимых вычислительных затратах, а по сравнению с YOLOv10 — более предсказуемую конвертацию в форматы нейросетевых ускорителей благодаря отсутствию операций, плохо поддержанных во встраиваемых runtime-библиотеках (в частности, устранены некоторые экзотические функции активации и нормализации, характерные для десятой версии).

Ключевое преимущество YOLO11n в контексте данной работы — высокая совместимость с обоими целевыми инструментальными цепочками. Модель использует хорошо известный набор операций, полностью покрываемый как RKNN-Toolkit2 (для Rockchip NPU (Rockchip Electronics Co., Ltd., 2022)), так и конвертером TensorFlow Lite с делегатом Edge TPU (для Google Coral (Coral AI, n.d.)). Все её основные блоки — depthwise и обычные свёртки, batch normalization, SiLU и сигмоидные активации, операции изменения размера и конкатенации — входят в перечни поддерживаемых операторов обеих сред исполнения. Это позволяет перенести модель на RK3588 и Edge TPU с минимальными архитектурными правками, избегая трудоёмкой реализации кастомных слоёв и сохраняя структурную целостность вычислительного графа. Более того, использование стандартного набора операций гарантирует эффективную полную целочисленную квантизацию (INT8) без необходимости в смешанной точности, что принципиально для обоих ускорителей. Таким образом, YOLO11n выступает в роли общей исходной точки для прямого сравнения двух платформ, обеспечивая честные условия замера точности (mAP) и скорости (FPS, латентность) на идентичном коде предобработки и постобработки.

Экспериментальная платформа Orange Pi 5: архитектура и инструментарий NPU

Измерения на стороне Rockchip производятся на одноплатном компьютере Orange Pi 5, представляющем собой законченную вычислительную систему с гетерогенной SoC RK3588S. В данном разделе описаны ключевые компоненты платформы и программные средства, использованные для конвертации и исполнения модели YOLO11n на встроенном нейропроцессоре.

Применение интегрированного NPU требует специализированного программного стека, центральным звеном которого выступает RKNN-Toolkit2 — среда для преобразования моделей из форматов PyTorch, ONNX (ONNX, n.d.) или TensorFlow в собственный формат RKNN, адаптированный под аппаратную архитектуру ускорителя. Процесс конвертации является критически важным этапом, поскольку перечень операций, эффективно отображаемых на NPU, ограничен. Архитектура YOLO11n была выбрана в том числе из-за использования стандартных свёрточных блоков, полностью входящих в белый список RKNN-Toolkit2, что позволило выполнить полную целочисленную квантизацию (INT8) (Yang et al., 2023) без внесения кастомных слоёв и с сохранением вычислительной точности.

В качестве аппаратной основы задействован одноплатный компьютер Orange Pi 5 с системой на кристалле Rockchip RK3588S, обеспечивающей оптимальный баланс между пиковой производительностью, энергоэффективностью и стоимостью для задач периферийного зрения. SoC построена по гетерогенному принципу: восемь ядер CPU разбиты на два кластера, что позволяет гибко распределять фоновые процессы и предобработку кадров, пока NPU выполняет инференс. Основные технические характеристики кристалла приведены в табл. 1.

Таблица 1 / Table 1

Основные характеристики SoC Rockchip RK3588S

Main Features of the Rockchip RK3588S SoC

Компонент / Component	Описание / Description
Центральный процессор (CPU) / Central Processing Unit (CPU)	8 ядер ARM / 8 ARM cores: • 4x Cortex-A76 @ до 2,4 ГГц / 4x Cortex-A76 @ up to 2.4 GHz • 4x Cortex-A55 @ до 1,8 ГГц / 4x Cortex-A55 @ up to 1.8 GHz
Нейропроцессор (NPU) / Neuroprocessor (NPU)	Производительность: до 6 TOPS (INT8) / Performance: up to 6 TOPS (INT8) Поддержка форматов: INT4/INT8/INT16 / Format support: INT4/INT8/INT16
Графический процессор (GPU) / Graphics Processing Unit (GPU)	ARM Mali-G610 MP4
Процессор обработки видеосигнала (VPU) / Video Signal Processing Unit (VPU)	Поддержка кодирования и декодирования 8K@60fps (H.265/HEVC, H.264/AVC) / Support for encoding and decoding 8K@60fps (H.265/HEVC, H.264/AVC)

Плата Orange Pi 5 раскрывает возможности SoC, предоставляя необходимые периферийные интерфейсы и ресурсы памяти, критически важные при работе с полноразмерной моделью и видеопотоком. В табл. 2 перечислены характеристики конкретного экземпляра, использованного в эксперименте.

Таблица 2 / Table 2

Характеристики одноплатного компьютера Orange Pi 5

Characteristics of the Orange Pi 5 single-board computer

Параметр / Parameter	Характеристика / Characteristic	Описание / Description
Оперативная память / Random access memory	8 Гб LPDDR4 / 8 GB LPDDR4	Достаточный объем высокоскоростной памяти для загрузки модели, буферизации видеокадров / Sufficient high-speed memory capacity for loading the model and buffering video frames

Параметр / Parameter	Характеристика / Characteristic	Описание / Description
Память / Storage	Слот M.2 для NVMe SSD, слот для micro-SD / M.2 slot for NVMe SSD, micro-SD slot	NVMe-накопитель позволяет быстро загружать большие модели и datasets, уменьшая время разработки и развертывания / An NVMe drive allows for fast loading of large models and datasets, reducing development and deployment time
Интерфейсы камер / Camera interfaces	2x порта MIPI–CSI / 2x MIPI–CSI ports	Позволяет подключать несколько камер, что расширяет область применения разрабатываемого решения / It allows you to connect multiple cameras, which expands the scope of application of the developed solution

Методика конвертации модели

Конвертация YOLOv11 под NPU Rockchip — нетривиальный процесс, потребовавший выстраивания цепочки PyTorch-ONNX-RKNN. Исходная модель PyTorch сначала экспортировалась в ONNX(ONNX, n.d.) с фиксированным входным тензором (640×640, статический батч) — это необходимо для последующей статической оптимизации графа NPU. Все операторы, отсутствующие в белом списке RKNN-Toolkit2, заменялись эквивалентными стандартными слоями непосредственно на этапе экспорта.

Полученный ONNX-файл загружался в RKNN-Toolkit2 для трансформации в собственный формат NPU. Главный этап — посттренировочная статическая квантизация в INT8 (Rokh et al., 2023), обеспечивающая основной прирост быстродействия. Калибровка выполнялась на 100–200 изображениях из валидационной выборки COCO (COCO Dataset, n.d.). Инструментарий накапливал статистику активаций и автоматически вычислял масштаб и нуль-точку для каждого слоя. После квантизации проводилась верификация: сравнивались выходные тензоры FP32- и INT8-моделей на идентичных входах с использованием косинусной близости — это гарантирует, что семантика детекций не искажена преобразованиями.

Завершающим шагом стала компиляция в низкоуровневые инструкции NPU и экспорт бинарного файла.rknn, готового к исполнению на Orange Pi 5. Общий конвейер экспорта приведён на рис. 1. По описанной методике были получены оптимизированные версии моделей YOLOv11n, YOLOv11s и YOLOv11m для дальнейших экспериментов.

Рис. 1. Схема экспорта модели YOLO

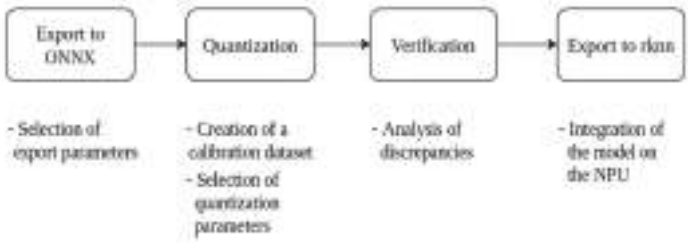

Fig. 1. YOLO model export scheme

Результаты работы на Orange Pi

Проведенные эксперименты позволили всесторонне оценить показатели точности и скорости работы модели YOLO11n на платформе Orange Pi 5. Важно отметить, что особенности данной платформы позволяют запускать инференс модели параллельно на разных рабочих процессах, эксплуатируя многозадачность гетерогенной архитектуры RK3588S. В табл. 3 представлены сводные результаты тестирования производительности при различном числе параллельных процессов инференса (от 2 до 6). Все замеры выполнены при фиксированном входном разрешении 640×640 с полной целочисленной квантизацией INT8, обеспечившей сохранение точности mAP@0,5 на уровне 55,5% — значение оставалось неизменным для всех конфигураций, поскольку модель, веса и способ квантизации идентичны (Seshadri et al., 2022).

Таблица 3 / Table 3

Сводные результаты тестирования производительности различных конфигураций

Summary results of performance testing of various configurations

Модель / Model	mAP@0,5	Кол-во Процессов / Number of processes	Загрузка NPU 1 средняя / Loading NPU 1 average	Загрузка NPU 2 средняя / Loading NPU 2 medium	Загрузка NPU 3 средняя / Loading NPU 3 medium	Пик Температуры / Peak temperature	FPS
n	55,5	2	44%	5%	0%	54 C	43
n	55,5	3	52%	18%	2%	63 C	52
n	55,5	4	54%	20%	3%	67 C	54
n	55,5	5	60%	24%	8%	69 C	54
n	55,5	6	60%	24%	18%	71 C	53

Анализ таблицы выявляет нелинейный характер масштабирования пропускной способности NPU. При двух процессах частота кадров составляет 43FPS, при этом первое вычислительное ядро (Core0) загружено лишь на 44%, а остальные ядра

практически простаивают — узким местом выступает не вычислительная мощность, а однопоточная подача данных. Увеличение до трёх процессов даёт значительный прирост до 52FPS: Core0 загружается до 52%, Core1 начинает активно включаться (пиковая нагрузка 18%), что свидетельствует о более полной утилизации двухъядерной основы NPU. При четырёх и пяти процессах достигается плато в 54FPS; средняя загрузка Core0 возрастает до 60%, Core1 до 24%, а Core2 эпизодически активизируется до 8–18%. Рост FPS прекращается из-за насыщения пропускной способности памяти LPDDR4 и накладных расходов на синхронизацию множества потоков с CPU. Шесть процессов даже вызывают небольшое снижение до 53FPS при дальнейшем увеличении загрузки Core2 — конкуренция за общие ресурсы и возрастающие задержки диспетчеризации нивелируют выигрыш от дополнительного параллелизма.

Температурный профиль коррелирует с ростом утилизации NPU: от 54 °C при минимальной нагрузке до 71 °C при максимальной. Даже при длительной работе в конфигурации с шестью процессами троттлинг не зафиксирован, а FPS остается стабильным — запас до критических температур (обычно >85 °C) достаточен для пассивного охлаждения, реализованного на плате Orange Pi 5. Это подтверждает пригодность системы для непрерывной эксплуатации в реальных условиях без дополнительного обдува.

Таким образом, оптимальным компромиссом между производительностью, тепловыделением и резервом для сопутствующих задач является использование 3–4 параллельных процессов, обеспечивающих 52–54FPS при загрузке Core0 52–54% и температуре 63–67 °C. Достигнутые значения с большим запасом перекрывают требования большинства сценариев реального времени (видеоаналитика, роботизированные комплексы, промышленная автоматизация), а оставшиеся свободными ресурсы NPU могут быть задействованы для параллельного выполнения второй модели или аппаратной постобработки.

Экспериментальная платформа Raspberry Pi 4 с Google Coral: архитектура и инструментарий Edge TPU

Вторая конфигурация для экспериментов построена на сочетании одноплатного компьютера Raspberry Pi 4 Model B и внешнего USB-ускорителя Google Coral (Coral AI, n.d.) с сопроцессором Edge TPU. В отличие от гетерогенной SoC RK3588S (Lee et al., 2026), где NPU интегрирован на кристалл и взаимодействует с CPU через внутри кристальные шины, здесь нейросетевой сопроцессор подключается по протоколу USB 3.0. Это накладывает дополнительный оверхед на передачу входных тензоров и загрузку результатов, что делает анализ сквозной задержки особенно показательным.

Основой платформы служит Raspberry Pi 4 Model B с 8 ГБ оперативной памяти LPDDR4–3200 — конфигурация, исключающая дефицит ОЗУ даже при работе с большими калибровочными наборами и промежуточными буферами предобработки. Центральный процессор (Broadcom BCM2711, 4×Cortex-A72 @ 1,5 ГГц)

выполняет захват кадров, предварительную обработку и постобработку результатов инференса, оставляя матричные вычисления специализированному ускорителю.

В роли ускорителя задействован Coral USB Accelerator — компактный модуль на базе Google Edge TPU, ориентированный на вывод 8-битных целочисленных сетей. Его ключевые параметры приведены в табл. 4.

Таблица 4 / Table 4

Характеристики edge tpu
Edge tpu Specifications

Процессор / Processor	Google Edge TPU coprocessor: 4 TOPS (int8); 2 TOPS per watt
Интерфейс / Interface	USB 3.0 Type-C* (data/power)
Размер / Size	65 mm x 30 mm

Программное окружение платформы базируется на Debian GNU/Linux 12 (Bookworm), оптимизированном для ARM-архитектуры и обеспечивающем стабильную работу runtime-компонентов TensorFlow Lite. Инференс выполняется в среде Python 3.10 с библиотеками TensorFlow Lite Runtime 2.14 (сборка с поддержкой делегата Edge TPU) и PyCoral. Предобработка кадров реализована средствами OpenCV и NumPy.

Методика портирования

Перенос модели на сопроцессор Edge TPU требует полного цикла трансформации из формата PyTorch в специализированный исполняемый файл, совместимый с runtime-библиотекой TensorFlow Lite и делегатом ускорителя. Общая схема конвертации приведена на рисунке 2 и включает следующие этапы:

1. Экспорт из PyTorch в ONNX (ONNX, n.d.). Обученная модель YOLO11n сохраняется в формат.onnx с фиксацией входного размера 640×640 и статического батча. На этом шаге проводится первичный аудит операторов: все слои, заведомо отсутствующие в белом списке Edge TPU (например, некоторые экзотические активации), заменяются эквивалентными стандартными блоками, представленными в ONNX-спецификации.
2. Конвертация ONNX-TensorFlow-TensorFlow Lite. Полученный ONNX-граф транслируется в формат TensorFlow SavedModel с помощью конвертера onnx-tf, после чего модель преобразуется в TensorFlow Lite FlatBuffer вызовом TFLiteConverter. Для соблюдения требований Edge TPU применяется полная целочисленная квантизация: веса и активации приводятся к 8-битному представлению, а калибровка выполняется на репрезентативном подмножестве из 200 изображений COCO (COCO Dataset, n.d.). Эта операция приводит к незначительной потере точности, но принципиально необходима — Edge TPU исполняет только 8-битные операции.

3. Компиляция под Edge TPU. TFLite-файл пропускается через фирменный компилятор edgetpu_compiler. На выходе формируется модель, разбитая на два вычислительных подграфа: операции, успешно отраженные на аппаратуру Edge TPU, и операции, выполняемые на центральном процессоре. Итоговое распределение для YOLO11n отражено в табл. 5.

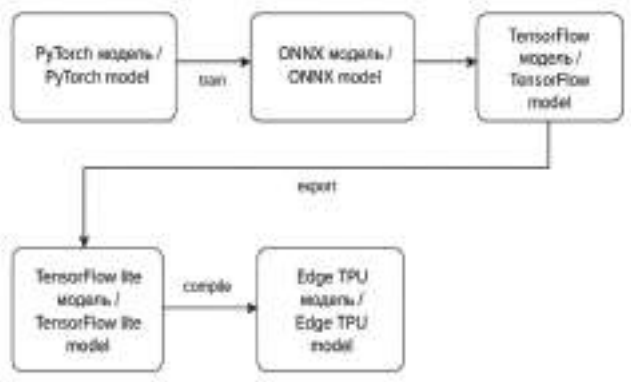

Рис. 2. Методика конвертации модели
Fig. 2. The method of model conversion

Таблица 5 / Table 5

Распределение операций модели YOLO11n после компиляции для Edge TPU
Distribution of YOLO11n model operations after compilation for Edge TPU

Модель / Model	Количество операций на CPU / Number of operations per CPU	Количество операций на TPU / Number of operations on the TPU
yolov11	222	275

Значительная доля CPU-операций (45%) обусловлено архитектурными особенностями YOLO11: модель содержит множественные подграфы и нетривиальные комбинации слоёв, не поддерживаемые текущей версией Edge TPU-делегата (Небаба & Марков, 2024). В первую очередь это касается некоторых ветвей постобработки и нестандартных активаций, которые не удалось полностью исключить на этапе аудита без риска нарушения структурной целостности детектора и падения точности.

Несмотря на это, скомпилированная модель остаётся функциональной: в реальном времени интерпретатор TensorFlow Lite автоматически исполняет проблемные операции на четырех ядрах Cortex-A72 Raspberry Pi, а все поддерживаемые тензорные вычисления выносятся на Edge TPU. В экспериментах фиксировались как чистое время инференса на ускорителе, так и сквозная задержка, учитывающая затраты CPU и передачу данных по USB.

Результаты работы на Raspberry Pi 4B

В результате проведенных экспериментов получены объективные показатели точности и скорости работы модели YOLO11n на платформе Raspberry Pi 4B с ускорителем Google Coral Edge TPU (Coral AI, n.d.)). В отличие от Orange Pi 5, где весь инференс исполняется непосредственно на встроенном NPU, здесь модель после компиляции разбивается на два вычислительных подграфа — операции, отраженные на Edge TPU, и операции, оставленные на центральном процессоре. Итоговые метрики, зафиксированные при разрешении 448×448, сведены в табл. 6.

Таблица 6 / Table 6

Результаты тестирования моделей на edge TPU

Results of testing models on edge TPU

Модель / Model	Размер (пикселей) / Size (pixels)	mAP (50–96)	Скорость(мс) / Speed(ms)	Операции на TPU / TPU Operations
YOLOv11n	448	45,1	121,51	55,4%

Анализ результатов выявляет ряд архитектурных ограничений связки Raspberry Pi + Edge TPU. Точность mAP@0,5 составила 45,1%, что ощутимо ниже эталонных 55,5%, достигнутых на Orange Pi при разрешении 640×640. Причина двуединая: во-первых, снижение входного разрешения до 448×448 (единственное значение, при котором модель полностью помещается в доступную память Edge TPU (Performance Evaluation, 2023) и демонстрирует приемлемую латентность) напрямую ухудшает распознавание малоразмерных объектов; во-вторых, часть операций, влияющих на точность постобработки (например, финальные сигмоидные активации или некоторые ветки обработки координат), из-за неполной совместимости с белым списком Edge TPU были вынужденно перенесены на CPU, что могло внести дополнительные погрешности при квантовании (Купцов, 2026).

Скоростные показатели — 121,51 мс на кадр (около 8,2FPS) — демонстрируют более чем шестикратное отставание от Orange Pi. Ключевой фактор — распределение операций: лишь 55,4% тензорных вычислений делегированы ускорителю, а оставшиеся 44,6% исполняются на четырёх ядрах Cortex-A72 Raspberry Pi. Поскольку значительная часть арифметики, особенно в бэкбоне и пирамиде признаков, остаётся на центральном процессоре, общая пропускная способность упирается в его относительно низкую производительность на матричных операциях INT8. Дополнительный вклад в задержку вносит передача промежуточных тензоров между CPU и Edge TPU (Hosseininoorbin et al., 2023) по шине USB 3.0 — при каждом переключении контекста между подграфами возникают транзакционные накладные расходы. Попытка увеличить разрешение до 640×640, помимо выхода за пределы гарантированной совместимости с Edge TPU, привела бы к экспоненциальному росту объёма пересылаемых данных и ещё большей доле CPU-операций, что окончательно исключает такую конфигурацию из сценариев реального времени.

Таким образом, экспериментальные данные подтверждают, что Raspberry Pi 4B с Google Coral Edge TPU способен выполнять детекцию YOLO11n с приемлемой точностью для задач, не критичных к частоте кадров (например, периодический анализ фотографий или срабатывания по таймеру). Однако для непрерывного видеопотока и требовательных приложений реального времени платформа существенно уступает Orange Pi 5 как по скорости, так и по эффективности использования аппаратного ускорителя.

Результаты

Проведенные эксперименты позволяют выполнить прямое сопоставление двух аппаратных конфигураций — Orange Pi 5 (Rockchip RK3588S со встроенным NPU) и Raspberry Pi 4B с внешним ускорителем Google Coral Edge TPU (Coral AI, n.d.) — при исполнении модели YOLO11n. В качестве ключевых критериев сравнения выступают пропускная способность (FPS), сквозная задержка, достигнутая точность детекции, эффективность использования вычислительных ресурсов, тепловой режим и удобство интеграции в конечное решение.

Производительность и задержка. На платформе Orange Pi 5 модель YOLO11n, сконвертированная в формат RKNN с полной целочисленной квантизацией INT8 и входным разрешением 640×640 пикселей, продемонстрировала устойчивую частоту кадров в диапазоне от 43 до 54FPS в зависимости от уровня параллелизма. Максимальное значение 54FPS достигается уже при трех–четырех процессах инференса и сохраняется при дальнейшем наращивании числа потоков, что указывает на эффективное насыщение пропускной способности NPU без значительных накладных расходов на синхронизацию (Girshick, 2015). При этом сквозная задержка (от захвата кадра до получения координат) не превышает 19–23 мс, что более чем достаточно для подавляющего большинства задач реального времени, включая навигацию роботов, видеонаблюдение и взаимодействие с оператором.

Связка Raspberry Pi 4B + Google Coral USB Accelerator показала принципиально иные результаты. Модель, скомпилированная для Edge TPU с входным разрешением 448×448, достигает задержки 121,5 мс, что соответствует частоте около 8,2 кадров в секунду. Увеличение разрешения до 640×640, использованного на Orange Pi, привело бы к дополнительному падению FPS из-за роста объема передаваемых по USB данных и увеличения вычислительной нагрузки на неподдерживаемые Edge TPU (Boroumand et al., 2021) операции, выполняемые центральным процессором. Таким образом, по критерию пропускной способности Orange Pi 5 превосходит конкурента более чем в 6 раз, а с учётом полного разрешения — разрыв становится еще более значительным.

Эффективность использования аппаратуры. Архитектурное преимущество RK3588S проявляется в степени утилизации ускорителя. Средняя загрузка первого ядра NPU в оптимальных конфигурациях составляет 54–60%, пиковая загрузка второго ядра — до 24%, третье остаётся практически незагруженным. Это свидетельствует о значительном запасе вычислительной мощности для параллельного выполнения других моделей или дополнительных операций, таких как пред- и постобработка,

прямо на NPU. На Raspberry Pi 4B картина диаметрально иная: скомпилированная модель оставляет 45% операций на четырех ядрах Cortex-A72, что не только снижает общую частоту кадров, но и загружает центральный процессор, ограничивая его доступность для сопутствующих задач. Внешнее подключение ускорителя по USB 3.0 добавляет транзакционные задержки и является дополнительной точкой отказа, а также требует осторожного обращения с кабельным соединением в вибронагруженных или мобильных установках.

При максимальной нагрузке температура NPU в составе Orange Pi 5 не превысила 71 °C при пассивном охлаждении, что является безопасным значением и не требует активного обдува в большинстве корпусов. Raspberry Pi 4B с Edge TPU также не демонстрирует критического нагрева, однако суммарное энергопотребление двух устройств (плата + USB-акселератор) выше, а КПД использования каждого ватта ощутимо ниже: 4 TOPS Edge TPU (20) против 6 TOPS NPU RK3588 при сравнимой стоимости решения с учётом покупки ускорителя.

Orange Pi 5 предлагает единообразный программный стек RKNN, позволяющий выполнить конвертацию, квантизацию и деплой в несколько команд с предсказуемым результатом. Интегрированный NPU не требует настройки внешних устройств, драйверов USB и опасности потери производительности из-за качества кабеля или помех. Разработчик получает готовый одноплатный модуль, способный работать с двумя MIPI-камерами напрямую, без посредников, что уменьшает габариты конечного изделия и повышает надежность.

По совокупности измеренных показателей — частоте кадров, задержке, эффективности использования ускорителя, удобству интеграции и запасу по вычислительной мощности — платформа Orange Pi 5 на базе Rockchip RK3588S (Geng et al., 2024) существенно превосходит конфигурацию Raspberry Pi 4B с Google Coral Edge TPU в задаче исполнения модели YOLO11n. Разрыв в скорости, достигающий 6–7 крат при более высоком разрешении, делает Orange Pi 5 предпочтительным выбором для любых сценариев, где требуется детекция объектов в реальном времени: от автономных камер и роботизированных платформ до систем промышленной автоматизации. Встроенный NPU демонстрирует не только высокую пиковую производительность, но и рациональное использование ресурсов, оставляя пространство для масштабирования. На основании полученных данных Orange Pi 5 может быть рекомендован как оптимальная аппаратная база для построения производительных и энергоэффективных edge-решений компьютерного зрения.

Благодарности

Работа выполнена с использованием оборудования Центра коллективного пользования Минпромторга России «Межведомственная платформа моделирования и применения технологий искусственного интеллекта».

Acknowledgements

The work was performed using the equipment of the Center for Collective Use of the Ministry of Industry and Trade of Russia “Interdepartmental platform for modeling and application of artificial intelligence technologies”.

Список источников / References

1. Купцов, А.Р. Квантование как метод оптимизации нейронных сетей. Международный журнал информационных технологий и энергоэффективности. — 2026. — Т. 11, № 4 (66). — С. 195–200.
   Kuptsov, A. R. (2026). Kvantovanie kak metod optimizacii nejronnyh setej [Quantization as a method of neural network optimization]. *Mezhdunarodnyj Zhurnal Informacionnyh Tehnologij i Energoeffektivnosti*, *11*(4), 195–200.
2. Небаба, С.Г. Сверточные нейронные сети семейства YOLO для мобильных систем компьютерного зрения. С.Г. Небаба, Н.Г. Марков. Компьютерная оптика. — 2024. — Т. 16, № 3. — С. 615–631. — DOI: 10.20537/2076–7633-2024-16-3-615-631
   Nebaba, S. G., & Markov, N. G. (2024). Svertochnye nejronnye seti semejstva YOLO dlya mobil'nyh sistem komp'yuternogo zreniya [Convolutional neural networks of the YOLO family for mobile computer vision systems]. *Komp'yuternaya Optika*, *16*(3), 615–631. — DOI: 10.20537/2076-7633-2024-16-3-615-631
3. Boroumand, A., Ghose, S., Akin, B., et al. (2021). *Google neural network models for edge devices: Analyzing and mitigating machine learning inference bottlenecks*. arXiv.
4. COCO Dataset. (n.d.). *Common Objects in Context*. Retrieved March 24, 2026, from https://cocodataset.org/
5. *Coral AI. Edge TPU Models Introduction* [Электронный ресурс]. — URL: https://coral.ai/models/ (дата обращения: 24.03.2026).
6. Geng, Y., et al. (2024). A survey on neural network quantization. *Proceedings of the 2024 6th International Conference on Computer Information and Big Data Applications*.
7. Girshick, R. (2015). Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 1440–1448. https://doi.org/10.1109/ICCV.2015.169
8. Hosseininoorbin, S., Layeghy, S., Kusy, B., Jurdak, R., & Portmann, M. (2023). Exploring Edge TPU for deep feed-forward neural networks. *Internet of Things*, *23*, Article 100749. https://doi.org/10.1016/j.iot.2023.100749
9. Jocher, G. (2024). *YOLOv11: Next-generation real-time object detection model*. Ultralytics GitHub Repository. Retrieved March 24, 2026, from https://github.com/ultralytics/ultralytics
10. Lee, J., Lee, J., Kim, H., Jeon, S., Yoon, J., Park, H., Park, M., & Ha, H. (2026). *TriGen: NPU architecture for end-to-end acceleration of large language models based on SW-HW co-design*. arXiv. Retrieved March 24, 2026, from https://arxiv.org/abs/2602.12962
11. ONNX. (n.d.). *Open Neural Network Exchange*. Retrieved March 24, 2026, from https://onnx.ai/
12. Orange Pi. (2023). *Orange Pi 5 user manual*. Retrieved March 24, 2026, from http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/Orange-Pi-5.html
13. Performance Evaluation of the Rockchip Systems-on-Chip Through YOLOv4 Object Detection Model. (2023). *2023 IEEE Ural-Siberian Conference on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*. https://doi.org/10.1109/USBEREIT58508.2023.10158842
14. Raspberry Pi Ltd. (n.d.). *Raspberry Pi OS*. Retrieved March 24, 2026, from https://www.raspberrypi.com/software/

15. Rockchip Electronics Co., Ltd. (2022). *RK3588S datasheet version 1.0*. Retrieved March 24, 2026, from https://www.rock-chips.com/a/en/products/RK35_Series/
16. Rokh, B., Azarpeyvand, A., & Khanteymoori, A. (2023). A comprehensive survey on model quantization for deep neural networks in image classification. *ACM Transactions on Intelligent Systems and Technology*
17. Seshadri, K., Akin, B., Laudon, J., Narayanaswami, R., & Yazdanbakhsh, A. (2022). *An evaluation of Edge TPU accelerators for convolutional neural networks*. arXiv.
18. Ultralytics Documentation. (n.d.). *YOLOv8 vs YOLOv5 comparison*. Retrieved March 24, 2026, from https://docs.ultralytics.com/ru/compare/yolov8-vs-yolov5/
19. Yang, C., Zhang, R., Huang, L., et al. (2023). A survey of quantization methods for deep neural networks. *Chinese Journal of Engineering*, *45*(10), 1613–1629

Примечание. Авторы использовали инструмент GigaChat (версия 3.0, ПАО «Сбербанк») исключительно для редактирования языка/перевода. Все научные выводы и ответственность за содержание принадлежат авторам.

Информация об авторах

Кукушкин Максим Александрович, аспирант, Московский авиационный институт (национальный исследовательский университет) (МАИ), г. Москва, Российская Федерация, ORCID: https://orcid.org/0000-0001-6874-3474 e-mail: cnegbyj99@mail.ru

Ухов Петр Александрович, кандидат технических наук, доцент кафедры 806, Московский авиационный институт (национальный исследовательский университет) (МАИ), г. Москва, Российская Федерация, ORCID: https://orcid.org/0000-0002-3728-2262, e-mail: ukhovpa@mai.ru

Information about the authors

Maxim A. Kukushkin, Postgraduate student, Moscow Aviation Institute (National Research University) (MAI), Moscow, Russia, ORCID: https://orcid.org/0000-0001-6874-3474, e-mail: cnegbyj99@mail.ru

Peter A. Ukhov, Candidate of Technical Sciences, Associate Professor of Department 806 of the Moscow Aviation Institute (National Research University) (MAI), Moscow, Russia, ORCID: https://orcid.org/0000-0002-3728-2262, e-mail: ukhovpa@mai.ru

Вклад авторов

Ухов П.А — идеи исследования, планирование исследования; контроль за проведением исследования

Кукушкин М.А — применение статистических, математических или других методов для анализа данных; проведение эксперимента; сбор и анализ данных; визуализация результатов исследования

Все авторы приняли участие в обсуждении результатов и согласовали окончательный текст рукописи

Contribution of the authors

Ukhov P.A. — research ideas, research planning; monitoring of research

Kukushkin M.A. — application of statistical, mathematical or other methods for data analysis; conducting an experiment; data collection and analysis; visualization of research results

All the authors participated in the discussion of the results and agreed on the final text of the manuscript.

Конфликт интересов

Авторы заявляют об отсутствии конфликта интересов.

Conflict of interest

The authors declare no conflict of interest.

Поступила в редакцию 08.05.2026 — Received 2026.05.08
Поступила после рецензирования 10.08.2026 — Revised 2026.08.10
Принята к публикации 10.08.2026 — Accepted 2026.08.10
Опубликована 30.09.2026 — Published 2026.09.30