

КАЗАХСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ
ИМЕНИ АЛЬ-ФАРАБИ

ЛЕКЦИЯ № 4

Нейронные микроконтроллеры и аппаратные ускорители искусственного интеллекта Intel Neural Compute Stick и Google Coral. Особенности интеграции с Raspberry Pi

Бельгибаев Бауржан Абдрахимович
Д.т.н., доцент · Кафедра «Искусственный интеллект и big data»
«Технологии в интеллектуальных системах управления в IoT» · 4 курс, бакалавриат

1 Цель и задачи лекции

Цель — сформировать системное представление об аппаратном ускорении нейросетевого вывода на периферийных устройствах IoT и научить обосновывать выбор вычислительной платформы для задач интеллектуального управления техническими объектами.

Задачи: раскрыть архитектуру Edge AI; разграничить микроконтроллер, одноплатный компьютер и нейронный ускоритель; изучить аппаратные и программные особенности Intel Neural Compute Stick и Google Coral; рассмотреть подготовку моделей и процедуру интеграции с Raspberry Pi; определить показатели быстродействия, энергопотребления, точности и надёжности.

2 План лекции

1. Edge AI в интеллектуальных системах управления и IoT
2. Аппаратные классы устройств: MCU, SBC, NPU, VPU и TPU
3. Принципы вычисления и ускорения нейронных сетей
4. Intel Neural Compute Stick: назначение, архитектура и программная экосистема
5. Google Coral Edge TPU: архитектура и ограничения моделей
6. Подключение ускорителей к Raspberry Pi и программный конвейер
7. Математические показатели производительности и выбор платформы
8. Практический кейс и исследовательская методика

1 Edge AI как компонент интеллектуальной системы управления

Интеллектуальная система управления (ИСУ) использует сенсорную информацию, алгоритмы оценивания состояния и принятия решений, исполнительные устройства и обратную связь. В классическом контуре алгоритм регулятора часто вычисляется непосредственно на промышленном контроллере. В IoT-системе данные могут передаваться на удалённый сервер, однако облачный путь увеличивает задержку, требует устойчивого канала связи и создаёт дополнительные риски для конфиденциальности и доступности.

Edge AI — выполнение алгоритмов искусственного интеллекта вблизи источника данных: на микроконтроллере, одноплатном компьютере или специализированном ускорителе. Для системы видеонаблюдения Edge AI позволяет локально выявлять объекты и отправлять в сеть события вместо непрерывного потока видеоданных. Для робота — получать признаки препятствий без обязательной передачи каждого кадра в облако.

1 Edge AI как компонент интеллектуальной системы управления (продолжение)

Типовая архитектура: датчик/камера → получение данных → предобработка → нейросетевой вывод → логика принятия решений → исполнительный механизм; параллельно модуль IoT публикует телеметрию, журнал событий и состояние устройства. Для критически важных функций безопасности нейросетевая подсистема не должна заменять независимые аппаратные блокировки и аварийный останов.

1.1. Уровни системы

1 Edge AI как компонент интеллектуальной системы управления (продолжение)

Уровень	Основная функция	Пример
Физический	Измерение и воздействие	Камера, датчик расстояния, двигатель
Периферийные вычисления	Обработка данных и инференс	Raspberry Pi + USB-ускоритель
Связь	Обмен событиями и командами	MQTT, HTTPS, локальная сеть
Диспетчеризация	Визуализация и архивирование	SCADA, панель мониторинга
Управление безопасностью	Ограничения и аварийные действия	Контроллер безопасности, E-stop

2 Микроконтроллер, мини-компьютер и нейронный ускоритель

Микроконтроллер (MCU) объединяет вычислительное ядро, память и периферийные интерфейсы для встроенного управления. Одноплатный компьютер (SBC) представляет собой полноценную вычислительную платформу с операционной системой, файловой системой и сетевыми службами. Нейронный ускоритель — специализированное устройство для эффективного выполнения операций, типичных для нейросетевых моделей; обычно он работает совместно с хост-процессором.

Следовательно, Intel Neural Compute Stick и Google Coral USB Accelerator корректнее относить к аппаратным ускорителям нейросетевого вывода, а не к самостоятельным микроконтроллерам. Формулировка «нейронный микроконтроллер» может использоваться как обобщённый учебный термин, но при техническом проектировании необходимо различать эти классы.

2 Микроконтроллер, мини-компьютер и нейронный ускоритель (продолжение)

Критерий	MCU	Raspberry Pi (SBC)	USB AI-ускоритель
Типовые задачи	Считывание датчиков, ПИД, PWM	Linux, видео, сети, логика	Ускоренный инференс
ОС	Bare-metal / RTOS	Linux	Зависит от хоста
Детерминизм	Высокий при корректном проектировании	Ограничен обычной Linux	Не гарантирует жёсткое реальное время
Автономность	Может работать самостоятельно	Самостоятельный компьютер	Нуждается в хосте

3 Математические основы нейросетевых вычислений

Базовый нейрон вычисляет линейную комбинацию входов и применяет нелинейную функцию активации: $z = \Sigma_i w_i x_i + b$; $y = \phi(z)$. Для полносвязного слоя матричная запись имеет вид $y = \phi(Wx + b)$. Основная вычислительная нагрузка нейронных сетей возникает при многократных операциях умножения с накоплением (MAC).

Для двумерной свёртки выходной элемент карты признаков может быть записан как $Y[h,w,o] = \Sigma_k \Sigma_l \Sigma_i K[k,l,i,o] X[h+k,w+l,i] + b[o]$. Специализированные блоки NPU/TPU/VPU эффективно исполняют такие операции за счёт параллелизма, локальной памяти и оптимизированных форматов чисел.

3.1. Квантование и компромиссы точности

При квантовании вещественные веса и активации отображаются в целочисленное представление. Упрощённая формула: $x_real \approx s(x_int - z)$, где s — масштаб, z — нулевая точка. Переход от FP32 к INT8 сокращает размер хранения одного значения примерно в четыре раза, но реальная экономия памяти и ускорение зависят от архитектуры, поддерживаемых операций и затрат на преобразования.

Математические основы нейросетевых вычислений (продолжение)

Квантование после обучения (post-training quantization) использует калибровочные данные для выбора диапазонов; quantization-aware training моделирует ошибки квантования во время обучения. После преобразования модель необходимо повторно оценить на независимом тестовом наборе.

3.2. Разделение обучения и вывода

Обучение — оптимизация параметров модели на размеченных данных с помощью вычисления градиентов. Инференс — применение обученной модели к новым входным данным. Рассматриваемые USB-ускорители прежде всего ориентированы на инференс; обучение обычно выполняется на рабочей станции или сервере, после чего модель экспортируется в совместимый формат.

4 Intel Neural Compute Stick: особенности платформы

Линейка Intel Neural Compute Stick предназначена для ускорения нейросетевого вывода на периферийных устройствах. Особенно известна версия Neural Compute Stick 2 (NCS2), построенная на Intel Movidius Myriad X VPU и подключаемая по USB. Хост-компьютер отвечает за получение кадров, подготовку данных, запуск приложения и передачу результатов в систему управления.

В исторически применявшихся учебных проектах для NCS2 использовался Intel OpenVINO, включая преобразование модели в промежуточное представление и запуск через соответствующий программный интерфейс. Важно: поддержка Myriad/NCS2 зависит от версии OpenVINO; в современных выпусках она может отсутствовать. Поэтому перед развёртыванием требуется проверить совместимость конкретной версии ПО, ОС и архитектуры хоста. Устаревшие примеры команд нельзя без проверки переносить на актуальные версии Raspberry Pi OS.

4.1. Последовательность развёртывания

- Выбрать модель с операциями, поддерживаемыми выбранной версией инструментария.
- Проверить версию ОС Raspberry Pi, архитектуру процессора и поддержку USB-ускорителя.

4 Intel Neural Compute Stick: особенности платформы (продолжение)

- Экспортировать и оптимизировать модель в совместимый формат.
- Установить подтверждённую совместимую среду выполнения и зависимости.
- Подключить ускоритель через надёжный USB-порт; проверить обнаружение устройства.
- Измерить задержку инференса и полное время обработки кадра.
- Добавить обработку ошибок, контроль доступности устройства и безопасное резервное поведение.

Поскольку экосистема NCS2 является наследуемой, использование в новом промышленном проекте требует отдельной оценки жизненного цикла компонентов, безопасности обновлений и доступности замены оборудования.

6 Конвейер Edge AI с USB-ускорителем



Рис. 6. Нейросеть выполняется на ускорителе, мини-компьютер координирует поток данных.

5 Google Coral Edge TPU

Google Coral USB Accelerator содержит Edge TPU — специализированный процессор для выполнения поддерживаемых операций нейросетевого вывода. Распространённый сценарий — использование моделей TensorFlow Lite, скомпилированных для Edge TPU. Для выполнения на ускорителе модель должна соответствовать требованиям к типам данных и поддерживаемым операциям; часто применяется полное целочисленное квантование INT8.

При наличии неподдерживаемых операций возможна передача части вычислений центральному процессору. Такая гибридная схема увеличивает задержку из-за переключений и передачи данных. Поэтому недостаточно проверить, что модель «запускается»: нужно оценить долю операций на ускорителе и фактическое время инференса.

5.1. Типовой рабочий процесс

- Обучить или выбрать модель обнаружения объектов или классификации изображений.
- Преобразовать модель в TensorFlow Lite и выполнить необходимое квантование.
- Проверить компиляцию для Edge TPU и отчёт о поддержанных операциях.

5 Google Coral Edge TPU (продолжение)

- Установить совместимую среду исполнения на Raspberry Pi.
- Подключить USB Accelerator, выполнить тестовый инференс и измерить время.
- Связать результаты распознавания с логикой IoT, журналированием и исполнительными механизмами.

6 Интеграция Raspberry Pi и нейронного ускорителя

Для видеозадачи система состоит из камеры, Raspberry Pi, USB-ускорителя, сетевого интерфейса и контроллера исполнительных устройств. Камера передаёт кадры, Raspberry Pi выполняет масштабирование и нормализацию, ускоритель обрабатывает модель, после чего CPU выполняет постобработку (например, подавление немаксимумов для детекции) и формирует управляющее решение.

Типовой программный конвейер: capture() → preprocess() → infer() → postprocess() → decision() → publish(). Для минимизации задержки следует использовать ограниченную очередь кадров, избегать накопления устаревших изображений, контролировать пропуски кадров и отделять периодический контур управления от нерегулярной обработки видео.

6.1. Удалённое управление и защита

Команды исполнительным механизмам должны проверяться по допустимым диапазонам и текущему состоянию объекта. Взаимодействие с удалённым сервером рекомендуется защищать аутентификацией и шифрованием, а устройства — изолировать сетевыми правилами. При потере связи, зависании процесса или отсутствии новых кадров система должна переходить в заранее определённое безопасное состояние.

8 Intel NCS2 и Google Coral

Intel Neural Compute Stick 2	Google Coral USB Accelerator
• Процессор Movidius Myriad X (VPU) • Набор инструментов OpenVINO • Интерфейс USB	• Сопроцессор Edge TPU • Модели TensorFlow Lite, квантование INT8 • Интерфейс USB

Рис. 7. Различия в аппаратуре и программных средствах определяют подготовку модели.

7 Показатели эффективности и математические модели

Полная задержка обработки кадра: $T_{total} = T_{capture} + T_{pre} + T_{transfer} + T_{infer} + T_{post} + T_{decision} + T_{act}$. Здесь $T_{transfer}$ учитывает передачу данных между хостом и ускорителем. В конвейере с перекрытием стадий пропускная способность определяется наиболее медленным этапом, а не только суммой времен.

Частота обработки кадров: $FPS = N_{frames} / T_{observation}$. Среднее время инференса: $T_{infer} = (1/N)\ \Sigma_i\ T_{infer,i}$. Для задач управления важно оценивать также 95-й и 99-й процентили задержки, а не только среднее.

Энергетическая эффективность: $\eta_E = N_{inferences} / E$, где $E = \int P(t)dt$. В случае постоянной мощности приближённо $E \approx P_{avg} \cdot T$. Сравнение ускорителей корректно только при одинаковой модели, разрешении входа, размере пакета и условиях измерения.

Для классификации используют $Accuracy = (TP + TN)/(TP + TN + FP + FN)$, $Precision = TP/(TP + FP)$, $Recall = TP/(TP + FN)$, $F1 = 2PR/(P + R)$. Для детекции применяются IoU и mAP с указанием порога и протокола расчёта. При несбалансированных классах одной accuracy недостаточно.

Показатели эффективности и математические модели (продолжение)

7.1. Пример расчёта временного бюджета

Пусть экспериментально получены условные значения: захват кадра 12 мс, предобработка 8 мс, передача 4 мс, инференс 22 мс, постобработка 7 мс, принятие решения 2 мс, формирование управляющего воздействия 5 мс. Тогда T_total = 60 мс. Теоретический верхний предел частоты последовательной обработки равен примерно 16,7 кадра/с. Это иллюстративный расчёт, а не характеристика конкретного ускорителя.

Если объект движется со скоростью v = 0,8 м/с, то за задержку 0,060 с он проходит Δs = vT = 0,048 м. При проектировании роботизированного объекта это смещение необходимо учитывать в запасе безопасного расстояния, наряду с тормозным путём и погрешностью распознавания.

8 Сравнение Intel NCS2 и Google Coral для учебного проекта

Признак	Intel NCS2	Google Coral USB Accelerator
Вычислительная архитектура	Movidius Myriad X VPU	Edge TPU
Типовой инструментарий	Наследуемые версии OpenVINO	TensorFlow Lite / Edge TPU Compiler
Формат и совместимость	Зависят от версии OpenVINO и модели	Ограничения по операциям и INT8
Подключение	USB	USB
Основной риск	Поддержка старого ПО и драйверов	Совместимость модели и среды исполнения
Критерий выбора	Подтверждённый стек и измерения	Подтверждённый стек и измерения

8 Сравнение Intel NCS2 и Google Coral для учебного проекта (продолжение)

Не следует выбирать устройство по одному заявленному значению TOPS: реальная производительность зависит от типа операций, точности вычислений, размера модели, памяти, версии компилятора и доли выполнения на CPU. Для научного сравнения необходим единый экспериментальный протокол.

9 Практический пример: интеллектуальное видеонаблюдение за производственной зоной

Задача: камера фиксирует вход человека в заданную производственную зону. Raspberry Pi выполняет захват видеопотока, ускоритель запускает модель обнаружения человека, программная логика оценивает пересечение виртуальной границы и отправляет сообщение в систему мониторинга. При необходимости контроллер оборудования реагирует в соответствии с установленными правилами безопасности.

Исходные условия исследования: одинаковые кадры разрешения 640×480, одна обученная модель в двух совместимых представлениях, фиксированное освещение, несколько сценариев движения и отсутствие посторонней нагрузки. Для каждого варианта проводят не менее 30 повторных измерений задержки и дополнительно измеряют энергопотребление.

Вычислительная логика может использовать условие: alarm = 1, если confidence ≥ τ и центр обнаруженного прямоугольника находится внутри опасной зоны в течение K последовательных кадров. Значения τ и K выбирают по валидационным данным, чтобы уменьшить ложные срабатывания, не допуская недопустимой задержки обнаружения.

9 Практический пример: интеллектуальное видеонаблюдение за производственной зоной (продолжение)

9.1. Псевдокод прототипа

$$camera = open_camera()$$

$$model = load_compatible_model()$$

while system_active:

$$frame = camera.read()$$

$$input_tensor = preprocess(frame)$$

$$detections = model.infer(input_tensor)$$

$$valid = filter_by_confidence_and_zone(detections)$$

9 Практический пример: интеллектуальное видеонаблюдение за производственной зоной (продолжение)

$$event = temporal_filter(valid)$$

publish_telemetry(event, timestamp())

if event: request_safe_action()

Приведённый алгоритм является псевдокодом и требует реализации под выбранную библиотеку. Для промышленной установки действие request_safe_action() должно быть согласовано с независимой системой безопасности, а не напрямую заменять её.

10 Методика научного сравнения аппаратных решений

Гипотеза исследования: специализированный ускоритель при одинаковой нейросетевой задаче снижает задержку и/или удельное энергопотребление относительно выполнения на CPU. Независимая переменная — вычислительная платформа; зависимые переменные — T_total, T_infer, FPS, энергия на вывод, точность и доля ошибок.

Эксперимент включает: подготовку репрезентативного датасета; фиксацию модели и параметров предобработки; проверку корректности вывода; прогрев устройства; серию измерений; регистрацию температуры и мощности; расчёт средних, стандартных отклонений и доверительных интервалов. При сравнении разных квантованных версий дополнительно оценивают расхождение результатов распознавания.

Для N измерений стандартное отклонение: $s = \sqrt{[\Sigma_i(x_i - \bar{x})^2/(N - 1)]}$. При близости распределения к нормальному 95%-й доверительный интервал среднего оценивают как $\bar{x} \pm t_{0.975,N-1} \cdot s/\sqrt{N}$. При сильной асимметрии полезнее дополнительно приводить медиану и перцентили.

11 Ограничения и направления развития

Главные ограничения Edge AI: зависимость от поддерживаемых операций, память и питание, перегрев, деградация качества при изменении освещения, обновления ПО, защита моделей и данных, отсутствие гарантированного жёсткого реального времени в обычной Linux. Для мобильных роботов следует предусматривать резервирование датчиков, контроль времени последнего кадра и безопасное отключение привода.

Перспективные направления исследований: TinyML на микроконтроллерах, распределённый вывод между несколькими устройствами, адаптивный выбор частоты инференса, совместная оптимизация модели и вычислительной архитектуры, устойчивость к доменному сдвигу и мониторинг качества модели после внедрения.

14 Заключение

Нейронные ускорители расширяют возможности периферийных IoT-систем, позволяя выполнять задачи компьютерного зрения и интеллектуального анализа непосредственно рядом с объектом управления. Intel Neural Compute Stick и Google Coral иллюстрируют разные аппаратно-программные экосистемы. Эффективность решения определяется не только производительностью ускорителя, но и совместимостью модели, полным временным бюджетом, качеством распознавания, энергопотреблением, безопасностью и поддерживаемостью системы.

15 Контрольные вопросы

1. Что понимается под Edge AI и чем он отличается от облачного инференса?
2. Каковы функциональные различия между MCU, SBC и USB-нейроускорителем?
3. Почему NCS2 и Coral USB Accelerator не являются самостоятельными микроконтроллерами?
4. Какие операции нейросети чаще всего ускоряются аппаратно?
5. Каково назначение квантования INT8 и как оно влияет на точность?
6. В чём состоит различие между обучением модели и инференсом?
7. Какие компоненты входят в аппаратную архитектуру Intel NCS2?
8. Почему при работе с NCS2 важно учитывать версию OpenVINO?
9. Какие ограничения возникают при компиляции моделей для Edge TPU?
10. Из каких стадий складывается полная задержка обработки кадра?
11. Почему показатель TOPS недостаточен для выбора ускорителя?

15 Контрольные вопросы (продолжение)

12. Какие метрики используют для оценки качества классификации и детекции?
13. Как обеспечить безопасную реакцию системы при потере связи или зависании нейромодели?
14. Как организовать воспроизводимое экспериментальное сравнение двух платформ?
15. Какие исследовательские задачи актуальны для Edge AI в промышленных IoT-системах?

16 Рекомендуемая литература и ресурсы

- **Ли П. Архитектура интернета вещей. М.:** ДМК Пресс, 2018.
- Матюшин А. О. Программирование микроконтроллеров: стратегия и тактика. М.: ДМК, 2016.
- **Arduino Documentation:** https://docs.arduino.cc/learn/
- **Raspberry Pi Documentation:** https://www.raspberrypi.com/documentation/
- Intel OpenVINO Documentation (проверять поддержку конкретного устройства в выбранной версии): https://docs.openvino.ai/
- **Coral Documentation:** https://coral.ai/docs/
- **TensorFlow Lite Documentation:** https://www.tensorflow.org/lite

Примечание: изложение соответствует тематике лекции № 4 исходного силлабуса. Разъяснения о классификации аппаратуры, ограничениях совместимости и исследовательской методике добавлены для академической полноты.