Секция «Программно-аппаратные комплексы систем искусственного интеллекта»

**ИСПОЛЬЗОВАНИЕ СРЕДСТВ APPLE CORE ML ДЛЯ ИНТЕГРАЦИИ МАШИННОГО ОБУЧЕНИЯ В МОБИЛЬНЫЕ ПРИЛОЖЕНИЯ**

***Мартынов Николай Владимирович,***

*cтудент гр. БВТ1901 МТУСИ, Москва, Россия,*

*kolyapusya@me.com*

***Полянцева Ксения Андреевна,***

*ассистент кафедры МКиИТ, МТУСИ, Москва, Россия,*  
*k.a.poliantseva@mtuci.ru*

*Ключевые слова: мобильные приложения, машинное обучение, распознавание образов, Apple Core ML, iOS.*

**На сегодняшний день машинное обучение является очень полезной технологией и находит применение в различных областях. Использование машинного обучения в мобильной разработке получило широкое распространение. Мобильные приложения с интеграцией технологий машинного обучения могут успешно распознавать и классифицировать изображения, голос человека, распознавание текста с изображений и другие различные задачи.**

**Архитектура систем тестирования**

На сегодняшний день машинное обучение является очень полезной технологией и находит применение в различных областях. Использование машинного обучения в мобильной разработке получило широкое распространение. По данным We Are Social [1] и Hootsuite в отчете Global Digital 2021: мобильные устройства используют 66,6 % мирового населения (5,22 миллиарда человек). Было выпущено множество средств, позволяющих разработчикам создавать мобильные приложения, которые используют машинное обучение, но не все были успешными. Мобильные приложения с интеграцией технологий машинного обучения могут успешно распознавать и классифицировать изображения, голос человека, распознавание текста с изображений и другие различные задачи. Однако главной проблемой для разработчиков остается перенос на мобильные платформы огромные модели с миллионами подключений без потери скорости и качества работы модели машинного обучения. Выпуск таких библиотек очень важен, так как они упрощают сложную задачу выполнения машинного обучения на мобильных устройствах и позволяют разработчикам мобильных приложений, не имеющим опыта работы с технологиями машинного обучения, внедрять в свои проекты функции с поддержкой машинного обучения.

Компания Apple в рамках ежегодной конференции разработчиков Worldwide Developers Conference (WWDC) 2017 выпустила Core ML, который получает обновления и в настоящее время. Core ML позволяет разработчикам приложений интегрировать модели машинного обучения в приложения для iOS и Mac OS. Это было одно из первых решений в этой области, и поначалу понравилось разработчикам так как имело множество преимуществ.

Core ML оптимизирован для производительности на устройстве, что позволяет использовать минимально необходимый объем памяти и эффективное энергопотребление модели. Работа модели машинного обучения строго на устройстве также обеспечивает безопасность пользовательских данных, а приложение работает даже при отсутствии сетевого подключения.

Существенным преимуществом Apple Core ML является то, что он чрезвычайно прост в использовании. Всего несколько строк кода позволяют интегрировать полную модель машинного обучения. С момента представления Apple Core ML, он получил широкое использование во множестве мобильных приложений. Однако возможности Core ML ограничены. Core ML может помочь только интегрировать предварительно обученные модели машинного обучения в приложение. Это позволяет делать только прогнозирование, обучение модели невозможно.

Apple Core ML является не единственным решением в мобильной разработке и имеет аналоги. На конференции Google I/O 2018 компания Firebase представила набор средств разработки для машинного обучения ML Kit. Данный набор средств разработки позволяет разработчикам использовать машинное обучение в мобильных приложениях двумя различными способами: разработчики приложений могут выполнять вывод модели в облаке через API или работать только на устройстве, как и в модели Apple Core ML.

Firebase ML Kit для устройств предлагает низкую точность по сравнению с облачной версией, но обеспечивает большую безопасность данных пользователей. Средства машинного обучения для мобильных приложений от Firebase предлагает шесть базовых API-интерфейсов, готовых к использованию для разработчиков приложений, с уже предоставленными моделями для маркировки изображений, распознавания текста, обнаружение ориентиров, обнаружение лиц, сканирование штрих-кода и интеллектуальный ответ. Базовые API-интерфейсы, которые предлагает Firebase ML Kit, охватывают общие варианты использования машинного обучения в мобильных приложениях, а возможность использования специально обученных моделей делает ML Kit полным решением для машинного обучения на мобильных платформах.

Если сравнивать Apple Core ML и Firebase ML Kit, у набора средств разработки ML Kit есть несколько преимуществ. Основное преимущество ML Kit заключается в том, что он поддерживает как iOS, так и Android, и на обеих платформах можно использовать одни и те же API. Firebase ML Kit имеет шесть базовых API, которые легко реализовать и не требуют специальных знаний в области машинного обучения. Если необходимость использования машинного обучения в приложении покрывается базовыми API-интерфейсами, предлагаемыми Firebase ML Kit, то это хороший вариант из предварительно обученных моделей. Еще одно преимущество Firebase ML Kit заключается в том, что он предлагает API-интерфейсы как на устройстве, так и в облаке. API-интерфейсы на устройстве в Firebase ML Kit разработаны для быстрой работы, которые имеют возможность предоставлять результаты даже при отсутствии подключения к Интернету. Облачные API используют платформу Google Cloud ML и обеспечивают повышенный уровень точности.

Недостатком Firebase ML Kit является то, что для работы данного набор средств разработки может потребоваться обновление тарифного плана до платного в зависимости от использования.

Если разработка мобильного приложения ориентирована на iOS, то использование Core ML будет более эффективно. Инструменты Apple позволяют как обучать, так и внедрять модели машинного обучения в мобильное приложение, используя меньше строк кода. Обучать модели с помощью Apple Create ML также проще, чем с помощью TensorFlow,[2] но продукт от Google предлагает более продвинутые алгоритмы обучения. Apple Core ML и Firebase ML Kit – являются самыми популярными библиотеками машинного обучения, доступными для мобильных платформ, но они не являются единственными решениями. Существуют множество библиотек машинного обучения с открытым исходным кодом для мобильных платформ, например такие как Shark – Machine Learning, AIToolbox и другие. Однако эти библиотеки не получили широкого распространения.

Инструменты Apple и Google позволяют реализовать технологию машинного обучения в мобильных приложениях, но у каждого из них есть ограничения. При разработки мобильного приложения «Camera Pro» для распознавания объектов на изображениях и в реальном времени были выбраны инструменты Apple Core ML. Поддержка только платформы iOS, простота интеграции моделей и наличие необходимых инструментов, распространяемых бесплатно, для реализации функциональности программы – являются основными факторами в пользу инструментов машинного обучения для мобильных приложений от Apple.

Можно выделить несколько ключевых преимуществ инструментов Apple Core ML для реализации машинного обучения на мобильных устройствах:

1. Обработка информации практически в реальном времени. Отсутствие необходимости выполнять вызовы сетевых функций API, отправлять данные и ожидать, пока модель предоставит результат для ответа. Этот фактор может быть критическим моментом для приложений, которые выполняют обработку видео в реальном времени с камеры на мобильном устройстве.
2. Доступность без интернета. Для работы приложения с моделью машинного обучения нет необходимости подключаться к сети.
3. Конфиденциальность данных. Данные, которые получает модель машинного обучения, никогда не покидают мобильное устройство. Это означает, что данные для обработки никуда отправлять не требуется, все вычисления производятся на устройстве.
4. Бесплатное распространение моделей. Мобильное приложение работает без подключения к сети и не требует вызовы API функций. Эти факторы позволяют использовать приложение с моделями машинного обучения от Apple, без каких-либо ограничениях по запросам и обращениям к модели, в отличии от Firebase ML Kit который был рассмотрен ранее.

Несмотря на указанные преимущества, модели машинного обучения Apple Core ML для мобильных приложений имеют так же следующие недостатки:

1. Размер приложения. Добавление модели машинного обучения в мобильное приложение значительно увеличивает размер этого приложения на устройстве.
2. Использование ресурсов системы. Прогнозирование на мобильном устройстве требует использование большого количества вычислительных ресурсов, которые могу увеличить расход заряда аккумулятора. У старых мобильных устройств могут возникнуть трудности с предоставлением прогнозов в режиме реального времени в зависимости от технических характеристик.
3. Обучение модели. Модель машинного обучения, которая работает на мобильном устройстве, обучается вне этого устройства. После переобучения модели приложение необходимо обновить с учетом новой модели, и в зависимости от размера модели это может стать проблемой для пользователя.

Apple Core ML использует модель машинного обучения, которая была обучена где-то вне (на локальном компьютере или в облачной платформе), а затем преобразованная в подходящий формат Core ML. Для разработки приложения, включая создание исходного кода и интеграцию модели, используется IDE для Apple-совместимых платформ – Xcode. Компания Apple предоставила несколько готовых моделей Core ML с описанием каждой, основанным на том, что она способна прогнозировать [3]. Распознавание изображений в реальном времени, распознавание лиц, прогнозирование текста и идентификация голоса человека представляют собой несколько из различных применений, которые стали возможными благодаря машинному обучению с использованием моделей Core ML.

Основной целью разработки мобильного приложения «Camera Pro» является демонстрация интеллектуальных возможностей классификации объектов на изображениях и в реальном времени с камеры мобильного устройства с помощью моделей машинного обучения разработанных для Core ML. Тестирование работоспособности функций производилось на iPhone 12 под операционной системой iOS 14.8. В качестве моделей машинного обучения Core ML было выбрано три: Resnet50 – обнаруживает доминирующие объекты, присутствующие на изображении из набора 1000 категорий (деревья, животные, еда, автомобили, люди и другое); SqueezeNet – выполняет такую же функцию как и модель Resnet50, но имеет свой набор 1000 категорий и другую точность; YOLOv3Tiny – предназначена для быстрого обнаружения объектов, обнаруживающая 80 различных классов объектов для изображения RGB с размерами 416 на 416 пикселей, модель выводит два массива (по одному для каждого слоя) произвольной длины, каждый из них содержит оценки достоверности для каждой ячейки объекта и нормализированные координаты ограничивающие прямоугольник вокруг обнаруженного объекта.

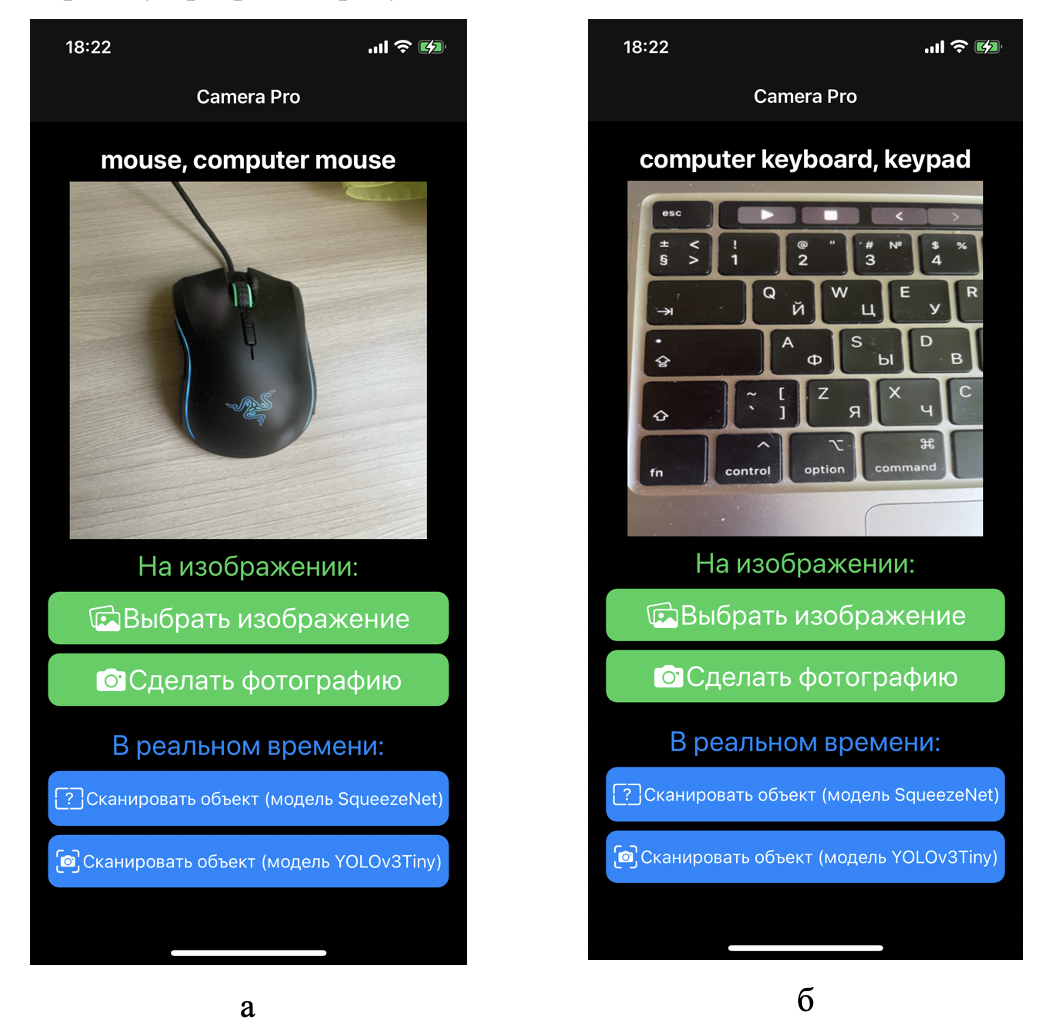


Рис. 1 – Результат работы приложения (a – выбрать фото, б – сделать фото)

В приложении «Camera Pro» модель Resnet50 выполняет функцию классификации объекта на статическом изображении, выбранном из галереи фотографий на устройстве или сделанном снимком в самом приложении.

Модель SqueezeNet [4] сканирует объект в реальном времени при помощи камеры мобильного устройства и выводит информацию о классификации обнаруженного объекта в текстовое поле, расположенное под камерой.

Модель машинного обучения YOLOv3Tiny используется для нахождения и классификации 80 различных типов объектов, присутствующих в кадре камеры. Применение данной модели помогает выделять область объекта и классифицировать его в режиме реального времени работы камеры мобильного устройства.

Интеграция моделей машинного обучения Apple Core ML в проект Xcode – не трудный процесс. После загрузки готовых моделей с расширением .mlmodel требуется скопировать файл в каталог проекта. Если открыть файл с моделью в Xcode можно будет ознакомиться со все доступной информацией о данной модели. На первой вкладке будет представлена информация о слоях внутри данной нейронной сети. На второй вкладке можно протестировать работу модели на своих примерах. На третьей вкладке можно проверить типы и размеры входных и выходных данных. Также важнейшим инструментом является автоматически сгенерированный класс модели Swift, где можно ознакомиться со всеми определениями, входными и выходными данными для выбранной модели, сгенерированной для класса. Эта информация помогает при интеграции модели в код.

Машинное обучение на iOS представляет собой систему инструментов и фреймворков, где Core ML находиться в качестве ядра. Core ML тесно интегрирован и поддерживает фреймворки Vision, Natural Language, Speech и Sound Analysis.

Фреймворк Vision [5] позволяет применять алгоритмы компьютерного зрения для выполнения множественных задач с входными изображениями и видео. Возможности этого фреймворка ориентированы на обнаружение лиц и ориентиров, обнаружение текста, распознавание штрих кодов, регистрирование изображений и отслеживание общих функций. Vision позволяет использовать пользовательские модели Core ML для таких задач, как классификация или обнаружение объектов. Этот фреймворк выполняет всю тяжелую работу от формата AVFoundation (полнофункциональная платформа для работы с временными аудиовизуальными носителями на устройствах Apple) до CVPixelBuffer (буфер изображений, содержащий пиксели в основной памяти), которого ожидает на вход или выход модель Core ML. Данный фреймворк использовался при разработке мобильного приложения «Camera Pro».

Фреймворк Natural Language предлагает функции распознавания речи. Он использует машинное обучение для глубокого понимания текста с помощью таких функций, как идентификация языка (автоматически определяет язык отрывка текста), токенизация (разбиение фрагмента текста на языковые единицы или токены), разметка частей речи (разметка отдельных слов их частью речи), лемматизация (вывод основы слова на основе его морфологического анализа) и определение имен в тексте (идентификация слов как имен людей, мест или организаций).

Фреймворк Speech выполняет функции распознавания речи в прямом эфире или в предварительно записанном аудио, получение транскрипций, альтернативные интерпретации и уровни достоверности результатов.

Фреймворк Sound Analysis классифицирует различные звуки, анализируя аудиофайлы или потоки. Определяет конкретные звуки, такие как смех или аплодисменты, в приложении, создав файл для анализа аудиофайла или потока.

Сам Core ML основан на примитивах низкого уровня, таких как Accelerate and BNNS, а также на Metal Performance Shaders. Библиотека BNNS фреймворка Accelerate – это набор функций, которые используются для построения нейронных сетей для обучения и вывода. Фреймворк Metal Performance Shaders содержит набор высоко оптимизированных вычислительных и графических шейдеров, разработанных для простой и эффективной интеграции в приложение Metal. Эти примитивы параллельного обмена данными специально настроены для использования уникальных характеристик оборудования каждого семейства графических процессоров для обеспечения оптимальной производительности.

В заключении следует отметить средства Apple Core ML – мощный инструмент, позволяющий внедрять модели машинного обучения любой сложности в мобильные приложения. Успешное внедрение предварительно обученных моделей машинного обучения в приложение «Camera Pro» позволило реализовать функции распознавания объектов на изображениях и в реальном времени. Apple Core ML позволяет разработчикам локально использовать возможности машинного обучения, от простой линейной регрессии до глубоких нейронных сетей прямо на iPhone. В то время как аналогичные средства для мобильных приложений использую облачные сервисы, требующие подключение к сети или работающие очень медленно на мобильных устройствах. Core ML позволяет оптимизировать работу модели машинного обучения для относительно быстрой работы на устройствах компании Apple. Компания Apple развивает направление машинного обучения, выпустив набор инструментов с открытым исходным кодом, который позволяет разработчикам преобразовывать различные модели машинного обучения в оптимизированную модель формата Core ML.

**Литература**

1. *We are social [Электронный ресурс]. – Режим доступа: https://wearesocial.com/us/. – Дата доступа: 11.02.2022.*
2. *Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2016. Tensorflow: large-scale machine learning on heterogeneous distributed systems . arxiv:1603.04467.*
3. *Apple Developer. Machine learning [Электронный ресурс]. – Режим доступа: https://developer.apple.com/machine-learning/models/. – Дата доступа: 11.02.2022.*
4. *Deep Learning Reading Group: SqueezeNet [Электронный ресурс]. – Режим доступа: https://www.kdnuggets.com/2016/09/deep-learning-reading-group-squeezenet.html. – Дата доступа: 12.02.2022.*
5. *Apple Developer. Documentation [Электронный ресурс]. – Режим доступа: https://developer.apple.com/documentation/vision/. – Дата доступа: 11.02.2022.*