УДК 004.422

ВОЗМОЖНОСТИ ИНТЕГРАЦИИ НЕЙРОННОЙ СЕТИ В МОБИЛЬНОЕ ПРИЛОЖЕНИЕ С ИСПОЛЬЗОВАНИЕМ ФРЕЙМВОРКОВ МАШИННОГО ОБУЧЕНИЯ

Овсянников Александр Владимирович, инженер, преподаватель
Samsung Innovation Campus,
Макаров Константин Сергеевич, к.т.н., зав. кафедрой программного обеспечения и администрирования информационных систем,
Курский государственный университет
Ovsyannikov Alexander Vladimirovich, engineer, lecturer at Samsung Innovation Campus, it_school@kursksu.ru
Makarov Konstantin Sergeevich, Ph.D., head of the department of Software and Information Systems Administration, makarov_ks@kursksu.ru
Kursk State University

В работе рассмотрены основные подходы к интеграции нейронных сетей в мобильные приложения, отмечены их достоинства и недостатки. Представлен обзор наиболее популярных фреймворков машинного обучения, применяемых для интеграции нейронных сетей в мобильное приложение. В качестве примера рассмотрены основные этапы использования TensorFlow Lite для классификации аудиозаписей на мобильном устройстве.

Ключевые слова: *нейронная сеть, мобильная разработка, фреймворки машинного обучения, TensorFlow Lite, классификация аудиозаписей*

Сегодня мобильные устройства используются повсеместно – почти половина живущих на планете людей активно устанавливают и используют мобильные приложения. По данным одного из ведущих мировых порталов статистики Statista [1] активных пользователей мобильных устройств более трех миллиардов человек. Внедрение технологий машинного обучения в работу приложе-

ний способно расширить функционал приложений и улучшить качество их работы. Однако такая интеграция сопряжена с рядом существенных трудностей таких, как ограниченность производительности мобильных устройств, необходимость обеспечения работоспособности приложения в условиях отсутствия доступа к сети Интернет и др.

Для преодоления указанных выше недостатков и обеспечения интеграции технологий машинного обучения в работу мобильных приложений предлагается ряд технических решений [2], [3] и др. Вместе с тем в известных работах отсутствуют детальная классификация и обзор способов такой интеграции.

Рассмотрим способы интеграции технологий машинного обучения в работу мобильных приложений на платформе Android в силу открытости и распространенности данной операционной системы [4].

Существуют два основных подхода к интеграции нейронных сетей в мобильное приложение: с включением нейронной сети в код мобильного приложения (Tensorflow modile, Tensorflow lite, экспорт модели в C++ и др.) и с размещением на сервере (TF Serving & Docker, TF Serving & VDS, Google AI Platform, Flask & Keras и др.).

Основными достоинствами технологий, использующими первый подход, являются экономия ресурсов – отсутствие необходимости платить за облако, администрирование и т.д., скорость отклика (нет запроса по сети), возможность работы в условиях отсутствия доступа сети. Недостатком подобных способов интеграции нейронных сетей в мобильные приложения является оперативная память и размер внутреннего хранилища мобильных устройств, недостаточные для работы ряда моделей машинного обучения.

В этой связи выбор подхода к интеграции зависит от ряда факторов, таких как характер решаемой задачи (вычислительные ресурсы, необходимые для работы нейронной сети), параметров мобильных устройств, на которых будет разворачиваться приложения, доступа к беспроводным сетям передачи данных и др.

Существует несколько фреймворков машинного обучения, которые поддерживают мобильную разработку и совместимы с мобильными платформами. Рассмотрим наиболее распространённые библиотеки, включающие реализацию моделей машинного обучения.

1. TensorFlow Lite: TensorFlow Lite – версия фреймворка TensorFlow, оптимизированная для работы на мобильных устройствах. Он поддерживает множество архитектур нейронных сетей и форматов моделей, а также предоставляет API для выполнения вывода на мобильных устройствах.
2. PyTorch Mobile: PyTorch Mobile – портативная версия фреймворка PyTorch, предназначенная для мобильных устройств. Он поддерживает широкий спектр архитектур нейронных сетей и форматов моделей, а также предоставляет API для выполнения вывода на мобильных устройствах.
3. Core ML: Core ML – это фреймворк от Apple, предназначенный для разработки нейронных сетей и машинного обучения для iOS, iPadOS и macOS. Он поддерживает различные архитектуры нейронных сетей, такие как сверточные, рекуррентные и Transformer, и форматы моделей, такие как TensorFlow и PyTorch.
4. Caffe2: Caffe2 – фреймворк нейронных сетей, разработанный Facebook, который поддерживает мобильную разработку. Он предоставляет API для загрузки моделей нейронных сетей и выполнения вывода на мобильных устройствах.
5. ONNX Runtime: ONNX Runtime – открытая платформа для выполнения моделей нейронных сетей, разработанная Microsoft. Она поддерживает различные форматы моделей, такие как TensorFlow и PyTorch, и может быть использована для выполнения вывода на мобильных устройствах.

В качестве примера интеграции фреймворка машинного обучения в мобильное приложение рассмотрим использование TensorFlow Lite для классификации аудиозаписей. Применение данного фреймворка, разработанного компанией Google и относящегося к общей с операционной системой Android экосистеме, позволяет выполнять работать с моделями машинного обучения на заранее поставляемой среде выполнения, устанавливаемой вместе с сервисами Google Play, тем самым уменьшая размер мобильного приложения [5].

Рассмотрим основные этапы интеграции нейронной сети в мобильное приложение.

1) Добавим зависимость TensorFlow Lite в файл build.gradle вашего проекта, как я описывал в предыдущем ответе:

```
dependencies {
  implementation 'org.tensorflow:tensorflow-lite:2.5.0'
}
```

2) Скачаем предобученную модель для классификации аудиозаписей в формате TensorFlow Lite и добавим ее в папку assets вашего проекта.

3) Создадим пользовательский интерфейс для нашего приложения, добавим элементы управления, такие как кнопки и поля ввода, для загрузки аудиозаписей и вывода результатов классификации.

4) Напишем код для загрузки аудиозаписи и классификации ее с помощью модели TensorFlow Lite:

```
// Загрузка модели из файла в объект класса Interpreter
private MappedByteBuffer loadModelFile(String filename) throws IOException {
  AssetFileDescriptor fileDescriptor = getAssets().openFd(filename);
  FileInputStream inputStream = new FileInputStream(fileDescriptor.getFileDescriptor());
  FileChannel fileChannel = inputStream.getChannel();
  long startOffset = fileDescriptor.getStartOffset();
  long declaredLength = fileDescriptor.getDeclaredLength();
  return fileChannel.map(FileChannel.MapMode.READ_ONLY, startOffset, declaredLength);
}
// Классификация аудиозаписи с помощью модели TensorFlow Lite
private void classifyAudio() {
  // Получение входных параметров для модели
  int inputSize = interpreter.getInputTensor(0).numel();
  int bufferSize = recorder.getBufferSize() / 2;
  ByteBuffer inputBuffer = ByteBuffer.allocateDirect(inputSize * 2);
  inputBuffer.order(ByteOrder.nativeOrder());
  // Запись аудиозаписи в буфер
  recorder.startRecording();
  short[] buffer = new short[bufferSize];
  int bufferReadResult;
  while (recording) {
    bufferReadResult = recorder.read(buffer, 0, bufferSize);
    for (int i = 0; i < bufferReadResult; i++) {
      inputBuffer.putShort(buffer[i]);
    }
  }
  recorder.stop();
  // Подготовка входного тензора для модели
  interpreter.resizeInput(0, new int[]{1, inputSize});
  interpreter.allocateTensors();
  interpreter.getInputTensor(0).buffer().rewind();
  interpreter.getInputTensor(0).buffer().put(inputBuffer);
  // Запуск модели и получение выходных данных
  interpreter.run();
  float[][] output = new float[1][NUM_CLASSES];
  interpreter.getOutputTensor(0).buffer().rewind();
  interpreter.getOutputTensor(0).buffer().asFloatBuffer().get(output[0]);
  // Вывод результатов классификации на экран
  // ...
}
```

5) Добавим обработчик для кнопки, которая начинает запись аудиозаписи и классификацию:

```
recordButton.setOnClickListener(new View.OnClickListener() {
```

```
  @Override
  public void onClick(View view) {
    // Начать запись аудиозаписи и классификацию
    recording = true;
    classifyAudio();
  }
});
stopButton.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View view) {
    // Остановить запись аудиозаписи
    recording = false;
  }
});
```

6) Добавим код для вывода результатов классификации на экран в текстовое поле:

```
// Определение названий классов
private static final String[] CLASS_NAMES = {"Class 1", "Class 2", "Class 3"};
// Вывод результатов классификации на экран
private void displayResults(float[] output) {
  int maxIndex = 0;
  float maxValue = output[0];
  for (int i = 1; i < output.length; i++) {
    if (output[i] > maxValue) {
      maxIndex = i;
      maxValue = output[i];
    }
  }
  resultText.setText(CLASS_NAMES[maxIndex]);
}
```

Использование представленных выше способов позволяет решить задачу интеграции технологий машинного обучения в работу мобильных приложения для различных стеков технологий и фреймворков. Представлены практическая реализация способа на основе TensorFlow Lite. Разработанное мобильное приложение позволяет классифицировать аудиозаписи. В дальнейшем планируется разработка методических материалов, направленных на формирование у обучающихся навыков разработки мобильных приложений, использующих технологии машинного обучения.

Список использованных источников

1. Statista: insights and facts across 170 industries and 150+ countries. – URL: https://www.statista.com/ (дата обращения – 20.03.2023 г.)
2. Скрыльник Т. Выводим нейронную сеть в продакшин для Android приложения. – URL: https://www.statista.comhttps://habr.com/ru/post/467055/ (дата обращения – 20.03.2023 г.)
3. Острицова В.А. Нейросети в мобильных приложениях / В. А. Острицова, И. А. Бессмертный // Сборник трудов X Конгресса молодых ученых: материалы Конгресса. Том 1. – Санкт-Петербург: федеральное государственное автономное образовательное учреждение высшего образования "Национальный исследовательский университет ИТМО", 2021. – С. 52-55
4. Statcounter: GlobalStats. – URL: https https://gs.statcounter.com/os-market-share (дата обращения – 20.03.2023 г.)
5. Обзор анонсов TensorFlow на конференции Google I/O – 2021. – URL: https://habr.com/ru/company/google/blog/561694/ (дата обращения – 20.03.2023 г.)