Предобработка данных с датчиков мобильных устройств на базе Android для машинного обучения

Д. А. Забалуев

Санкт-Петербургский государственный электротехнический университет ЛЭТИ» им. В.И. Ульянова (Ленина)

dazabaluev@stud.etu.ru

И. И. Холод

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

iiholod@mail.ru

Аннотация. **Объектом исследования является работа с данными, получаемыми с датчиков мобильных устройств на базе Android. Предметом исследования являются способы предобработки данных с датчиков мобильных устройств на базе Android для машинного обучения. В ходе исследования были изучены существующие работы, связанные с использованием данных с датчиков мобильных устройств на базе Android в машинном обучении, и обоснована необходимость предобработки таких данных. На основе проведенного исследования было составлено несколько методов предобработки данных, получаемых с датчиков мобильных устройств на базе Android.**

Ключевые слова: машинное обучение, датчики, обработка данных, Android, мобильные устройства, потоковые данные

I. Введение

В настоящее время наблюдается тенденция к росту использования технологий искусственного интеллекта и, в частности, машинного обучения (МО) практически в каждой сфере человеческой деятельности. Благодаря МО появилась возможность не только оптимизировать существующие решения определенных проблем, но и найти новые решения для проблем, которые ранее казались трудно разрешаемыми или нерешаемыми вовсе.

При применении технологии МО для решения неоднородных задач могут использоваться самые разные данные, поэтому и источников данных существует очень много. На сегодняшний день одними из наиболее распространенных источников данных являются умные мобильные устройства.

Современные смартфоны и смарт-часы оборудованы множеством различных датчиков, таких как: акселерометр, гироскоп, датчик температуры окружающей среды, датчик света и т. д. При этом эти датчики могут непрерывно отдавать показания с заданными временными интервалами. Таким образом, такие устройства можно рассматривать как набор источников потоковых данных.

Данная статья организована следующим образом: в разделе 2 обоснована целесообразность применения данных с датчиков смартфона в машинном обучении; в разделе 3 изложено обоснование необходимости предобработки данных с датчиков мобильных устройств на базе Android; в разделе 4 приводятся формальные описания составленных методов; в разделе 5 кратко излагаются выводы и будущая работа.

Работа выполнена при поддержке гранта Российского научного фонда №25-11-20020 и Санкт-Петербургского научного фонда.

II. Применение данных с датчиков мобильных устройств в машинном обучении

Данные, получаемые с датчиков современных мобильных устройств, способны предоставить много полезной информации как об окружающей среде, так и о самом пользователе устройства. Обученные на таких данных нейронные сети можно использовать, например, в медицине или в сфере дорожного транспорта.

Ымери (Ymeri) и др. [1] исследовали возможности по применению данных, получаемых с датчиков смарт-часов для автоматической оценки тяжести симптомов болезни Паркинсона (тремор, замедленность движений, нарушения равновесия, непроизвольные движения и т. д.) в реальном времени. В результате на данных, собранных с акселерометра и гироскопа на устройствах 129 пациентов, была обучена нейронная сеть, которая продемонстрировала достаточно высокую точность оценки тяжести симптомов.

Феррейра (Ferreira) и др. [2] провели исследование, целью которого являлось выявление наиболее эффективной с точки зрения производительности комбинации датчиков смартфона и алгоритмов классификации, способной с высокой точностью определять наличие или отсутствие элементов агрессивного вождения у водителей. В результате, на данных, собранных с акселерометра, гироскопа и магнетометра смартфона в течение четырех заездов, было обучено несколько нейронных сетей, которые с достаточно высокой точностью классифицировали определенные элементы вождения как агрессивные.

Дей (Dey) и др. [3] предложили использовать МО совместно с датчиками смартфона для анализа состояния дорог и классификации дорожных участков по их состоянию. Авторы работы разработали Android-приложение, которое в реальном времени считывает показания с магнитометра, акселерометра и GPS, и собрали данные на разных участках дорог. В результате

было обучено несколько нейронных сетей, которые с достаточно высокой точностью классифицировали состояние дороги по данным с датчиков смартфона.

III. Необходимость предобработки данных с датчиков мобильных устройств на базе Android

Обычно любые данные, вне зависимости от их источника, необходимо подвергнуть обработке прежде, чем использовать их в МО, поскольку, в противном случае, может пострадать как эффективность обучения, так точности работы полученной модели [4]. Обработка данных для МО заключается в устранении пропущенных значений, ошибок и несоответствий в датасете [5][6]. При этом уже существует множество методов для обработки датасетов, демонстрирующих разную степень эффективности в зависимости от разных условий. Так или иначе, чаще всего задача по обработке больших данных ложится на плечи специалистов по науке о данных, в то время как простые наборы данных, теоретически, можно предобрабатывать ещё до формирования из них датасетов.

Основная причина, по которой данные, получаемые с датчиков мобильных Android-устройств, необходимо предобрабатывать, заключается в том, что задаваемая конкретная частота опроса датчиков является только лишь указанием для системы, а не гарантом того, что показания с датчиков будут отдаваться именно с этой частотой [7]. Например, нередки ситуации, когда при одновременном опросе N датчиков с частотой в t миллисекунд, показания будут приходить в разное время. В итоге, при получении показаний с каждого из датчиков будет получаться N векторов показаний с разными метками времени, которые нельзя объединить в один вектор показаний с общей меткой времени, поскольку, фактически, показания были получены неодновременно.

Таким образом, на рис. 1 изображено графическое представление существующей на данной момент проблемы по использованию показаний, получаемых с датчиков мобильных устройств на базе Android, непосредственно для формирования датасетов для последующего их применения в МО.

Допустим, необходимо сформировать датасет на основе показаний, получаемых с акселерометра и гироскопа с заданной частотой w (пункт А на рис. 1). В этом случае ОС Android (пункт Б на рис. 1) опрашивает эти датчики с частотой w, получает их показания и метку времени этих показаний, а затем передает их вызывающему Android-приложению (пункт В на рис. 1) в виде определенной структуры данных. Мобильное приложение, инициализировавшее сбор данных с датчиков, сохраняет полученные данные в определенном формате (например, в виде CSV файла), тем самым формируя набор «сырых» данных (пункт Г на рис. 1). Затем полученный набор «сырых» данных отправляется на некоторый сервер для последующей обработки и анализа вручную (пункты Г, Д и Е на рис. 1).

На практике чаще всего применяется именно подобный подход, однако у него есть несколько существенных недостатков:

- при формировании каждого нового датасета на основе собранных данных, нужно производить устранение пропущенных значений и приведение показаний, полученных с нескольких датчиков, к одной метке времени (если предполагается, что эти показания были получены практически одновременно);
- собранные одновременно с нескольких датчиков «сырые» данные плохо подходят для формирования датасета, поскольку обучение на таких данных может привести к низкой точности работы итоговой модели, а использование таких данных на уже обученной модели может привести к получению недостоверных результатов.

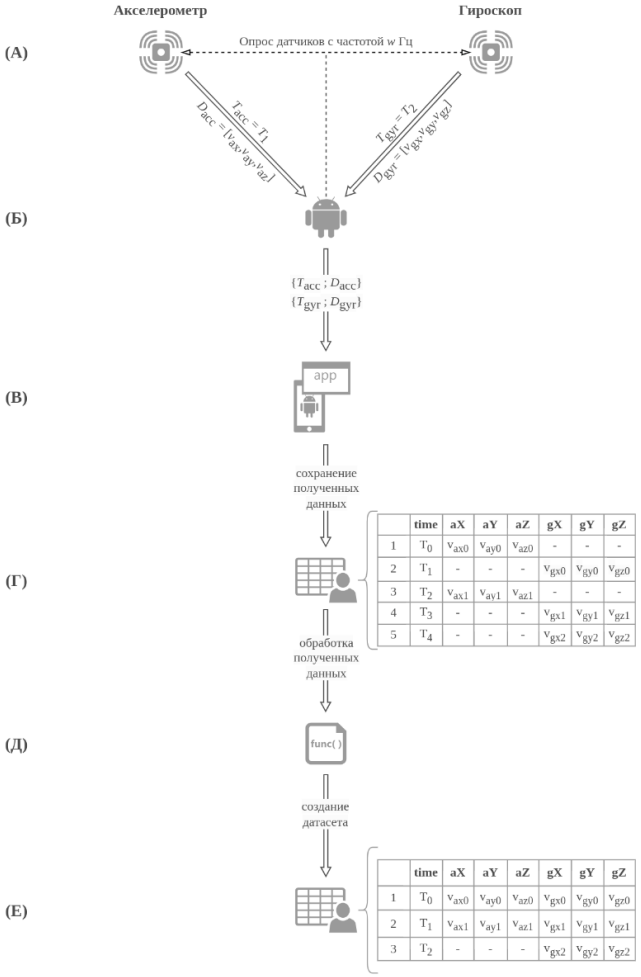

Рис. 1. Графическое представление существующей на данной момент проблемы по работе с данными, получаемыми с датчиков мобильных устройств на базе Android

Для решения этих проблем в данной работе предлагается иной подход, графическое представление которого изображено на рис. 2.

Стоит отметить, что пункты А и Б на рис. 2 повторяют аналогичные на рис. 1, а отличие этого подхода от существующего заключается в том, чтобы передавать полученные с датчиков показания приложению не напрямую, а через программный модуль-посредник (пункты В и Г на рис. 2).

Программный модуль-посредник должен будет реализовывать набор методов, позволяющих в реальном времени производить обработку как получаемых с датчиков показаний, так и меток времени этих показаний, чтобы формировать один целый вектор данных с единственной меткой времени. В свою очередь, Android-приложение получает от программного модуля эти вектора данных по запросу и может:

- использовать полученные векторы данных для создания датасета (пункт Д на рис. 2);
- использовать полученные векторы для обучения моделей методами МО в реальном времени, либо для классификации или прогнозирования посредством уже обученных моделей в реальном времени (пункт Е на рис. 2).

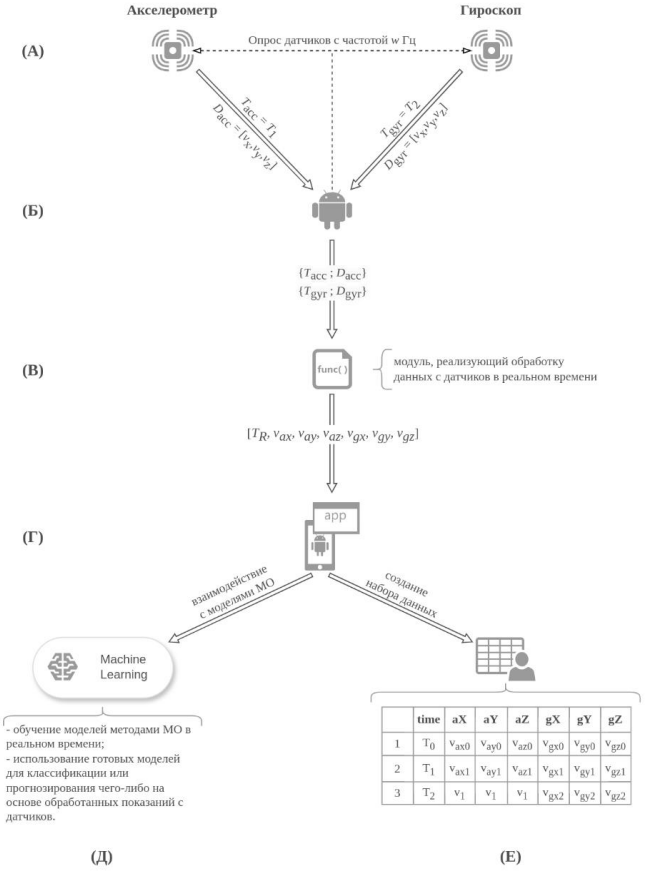

Рис. 2. Графическое представление предлагаемого решения

Для того чтобы данный подход можно было применить на практике, необходимо разработать несколько различных методов, способных обрабатывать данные с датчиков смартфона в реальном времени, устраняя ложные пропуски данных и синхронизируя показания, полученные с нескольких датчиков за короткий промежуток времени, к единой метке времени.

IV. Разработанные методы предобработки данных

В этом разделе изложены определения двух разработанных методов с пояснениями к ним.

A. Общие обозначения

Пусть имеется S датчиков, с которых необходимо получать вектор данных R по запросу. Данные с i-того датчика ($i = 1..S$) представлены в виде пары T_i и D_i, где T_i – метка времени получения данных с этого датчика (в UNIX формате), а D_i – набор данных, полученных с этого датчика.

Наборы данных D_i представляют собой массивы вещественных чисел вида:

$$[v_1, v_2, ..., v_{N_{i-1}}, v_{N_i}],$$

где N_i – количество значений, получаемых с i-того датчика.

В таком случае, результирующий вектор данных R имеет следующий вид:

$$R = \left[T_R, v_{1;1}, ..., v_{1;N_1}, ..., v_{i;1}, ..., v_{i;N_i}, ..., v_{S;1}, ..., v_{S;N_S} \right],$$

где T_R – общая метка времени всего вектора данных (в UNIX формате).

B. Метод «Сырые данные»

Суть данного алгоритма заключается в том, чтобы просто попытаться синхронизировать полученные за короткий промежуток времени данные с датчиков, создавая имитацию одновременного получения данных с каждого датчика по одному разу. То есть в результирующий вектор пойдут необработанные данные с датчиков (поэтому алгоритм и называется «Сырые данные»), а обработке подвергнутся только метки времени

В момент запроса вектора данных запускается процесс ожидания получения данных с каждого из S датчиков хотя бы по одному разу. Таким образом, будет получено S наборов данных D_i с их собственными временными метками T_i.

Наборы данных D_i пойдут в результирующий вектор R в том виде, в котором они были получены с датчиков. При этом значение метки времени T_R результирующего вектора может быть рассчитано несколькими разными способами:

- в качестве значения T_R принимается момент времени поступления запроса на получение вектора данных R (момент времени запуска процесса ожидания данных с каждого датчика по 1 разу);
- в качестве значения T_R принимается момент времени формирования вектора данных R (момент времени окончания процесса ожидания данных с каждого датчика по 1 разу);
- в качестве значения T_R принимается среднее арифметическое всех меток времени T_i.

Стоит учесть, что процесс ожидания получения данных со всех датчиков по одному разу может оказаться очень долгим в случаях, когда одновременно опрашивается много датчиков с низкой частотой опроса, поэтому стоит также ввести параметр максимального времени ожидания t_w. Когда $t_w > 0$, процесс ожидания получения данных с каждого из S датчиков завершается досрочно по истечении времени t_w. Если за это время не было получено данных с i-того датчика, то вместо набора данных D_i в результирующем векторе будет стоять пропуск данных.

Необходимо также отметить, что способ расчета метки времени T_R результирующего вектора R, также как и значение максимального времени ожидания t_w, являются параметрами данного метода, передаваемыми ему извне.

C. Метод «Аппроксимация по предварительно собранным данным»

В данном методе датчики непрерывно опрашиваются в реальном времени, а получаемые данные записываются в некоторый локальный кэш заданного размера. Когда кэш оказывается заполненным, то для того, чтобы записать в него новое значение, из него удаляется самое старое. При запросе вектора данных, набор данных для каждого датчика рассчитывается при помощи аппроксимации данных, хранящихся в кэше. Из полученных наборов данных и формируется результирующий вектор. При этом метка времени данных задается либо извне, либо принимается как момент времени поступления запроса на получение вектора данных.

Представим каждый датчик как функцию вида $f_i(t)$, позволяющую получить набор данных D_i с i-того датчика в определенный момент времени t. При запросе вектора данных R для определенного момента времени x, формирующие результирующий вектор наборы данных D_i получаются как значения функций $f_i(x)$.

Все датчики опрашиваются в реальном времени в независимых потоках с некоторой заданной периодичностью t_p. Полученный с i-того датчика набор данных D_i записывается в локальный кэш и удаляется из него только тогда, когда кэш оказывается полностью заполнен и этот набор данных является самым старым среди сохраненных в кэше. При этом для каждого датчика выделен собственный кэш заданного размера C.

Значением функции $f_i(t)$ в некоторой точке t является результат аппроксимации хранящихся в локальном кэше данных, полученных с i-того датчика:

- если t – некоторый момент времени, входящий в промежуток времени получения самых старых и самых новых данных, хранящихся в локальном кэше, то для аппроксимации применяются численные методы интерполяции;
- в остальных случаях для аппроксимации применяются численные методы экстраполяции.

Данный метод предполагает, что метка времени T_R результирующего вектора данных либо вычисляется как момент времени поступления запроса на формирование вектора данных R, либо принимается равной значению параметра t, переданного извне при запросе вектора. Из этого следует, что этот алгоритм позволяет получать вектора показаний датчиков не только в реальном времени, но и за какой-то прошедший промежуток времени.

Необходимо также отметить, что используемые при вычислении значений функций $f_i(t)$ численные методы интерполяции и экстраполяции тоже являются параметрами данного алгоритма, передаваемыми извне. Фактически, ответственность за подбор методов аппроксимации ложится на использующую алгоритм сторону, что позволяет управлять точностью получаемых результатов за счет возможности подбирать наиболее подходящие методы аппроксимации для каждой задачи, связанной с использованием данных с датчиков.

V. Заключение

В рамках данной работы были рассмотрены некоторые примеры применения технологии МО на данных, получаемых с датчиков мобильных устройств на базе Android. Было выяснено, что данная задача является актуальной на сегодняшний день. Помимо этого, была детально рассмотрена существующая проблема по использованию данных с датчиков мобильных устройств на базе Android без предобработки и предложен подход, способный решить эту проблему.

Предложенный подход включает в себя разработку нескольких гибких методов предобработки показаний с датчиков в реальном времени. Таким образом, было составлено два разных метода предобработки и дано их формальное описание.

В качестве дальнейшего развития работы в данном направлении планируется:

- составить еще один метод предобработки, который будет комбинировать оба предложенных в этой работе;
- написать псевдокод для каждого из методов;
- реализовать каждый метод на языке Kotlin и провести серию экспериментов на реальных Android-смартфонах, чтобы подтвердить или опровергнуть применимость разработанных методов на практике.

Благодарность

Авторы выражают благодарность коллегам из СПбГЭТУ «ЛЭТИ» за ценные советы и поддержку в ходе исследования. Отдельная благодарность выражается А. Азаревичу за консультации по обработке и синхронизации потоковых данных.

Список литературы

[1] Quantifying Parkinson's disease severity using mobile wearable devices and machine learning: the ParkApp pilot study protocol / Ymeri G., Salvi D., Olsson C. M., Wassenburg M. V., Tsanas A., Svenningson P. // BMJ Open. 2023. Vol. 13, no. 12. P. 10-17. doi: 10.1136/bmjopen-2023-077766.

[2] Driver behavior profiling: An investigation with different smartphone sensors and machine learning / Ferreira J., Carvalho E., Ferreira B.V., de Souza C., Suhara Y., Pentland A., Pessin G. // PLoS one. 2017. Vol. 12, no. 4. P. 201-216. doi: 10.1371/journal.pone.0174959.

[3] MagTrack: detecting road surface condition using smartphone sensors and machine learning / Dey M.R., Satapathy U., Bhanse P., Mohanta B.K., Jena D. // TENCON 2019-2019 IEEE Region 10 Conference (TENCON). IEEE. 2019. P. 2485-2489.

[4] Shalev-Shwartz S., Ben-David S. Understending machine learning: From theory to algorithms. Cambridge university press, 2014. 410 p.

[5] Misra P., Yadav A.S. Impact of preprocessing methods on healthcare predictions // Proceedings of 2nd International conference on Advanced Computing and Software Engineering (ICACSE). 2019. P. 144-150.

[6] Amato A., Di Lecce V. Data preprocessing impact on machine learning algorithm performance // Open Computer Science. 2023. Vol. 13, no 1. P. 100-116. doi: 10.1515/comp-2022-0278.

[7] Sensors Overview / Google Developers – URL: https://developer.android.com/develop/sensors-and-location/sensors/sensors_overview (дата обращения 10.03.2025).