МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное

образовательное учреждение высшего образования

«Самарский национальный исследовательский университет

имени академика С.П. Королева»

(Самарский университет)

Институт информатики, математики и электроники

Факультет информатики

Кафедра информационных систем и технологий

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

«АВТОМАТИЗИРОВАННАЯ СИСТЕМА НАВИГАЦИИ ВНУТРИ ПОМЕЩЕНИЙ С ПОМОЩЬЮ ИНЕРЦИАЛЬНОЙ ТЕХНОЛОГИИ ЛОКАЛЬНОГО ПОЗИЦИОНИРОВАНИЯ МОБИЛЬНЫХ УСТРОЙСТВ»

по направлению подготовки

09.04.01 Информатика и вычислительная техника

(уровень магистратуры)

профиль «Программное обеспечение мобильных устройств»

Обучающийся\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Д. Мавлютов

(подпись, дата)

Руководитель ВКР,

к.т.н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_О.К. Головнин

(подпись, дата)

Нормоконтролер\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Я.В. Соловьева

(подпись, дата)

Самара 2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное

образовательное учреждение высшего образования

«Самарский национальный исследовательский университет

имени академика С.П. Королева»

(Самарский университет)

Институт информатики, математики и электроники

Факультет информатики

Кафедра информационных систем и технологий

УТВЕРЖДАЮ

Заведующий кафедрой ИСТ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.А. Прохоров

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ МАГИСТРА

студенту Мавлютову Владимиру Дмитриевичу

группа № 6223-090401D .

Тема работы: «Автоматизированная система навигации внутри помещений с помощью инерциальной технологии локального позиционирования мобильных устройств»

утверждена приказом по университету №25-Т от «20» января 2020 г.

Исходные данные к работе:

* объект автоматизации: расчет интенсивности движения транспортных потоков;
* типы операционных систем: Windows 10, Mac OS X, iOS, Android;
* среда программирования: Visual Studio Code;
* техническое обеспечение: тип ЭВМ – IBM PC совместимый;
* манипулятор – мышь, клавиатура.

Перечень вопросов, подлежащих разработке в работе:

* анализ предметной области «Анализ транспортных потоков»;
* аналитический обзор систем-аналогов;
* разработка системы детектирования признаков акустического излучения транспортных средств;
* разработка и исследование алгоритмов классификации акустического излучения;
* разработка, отладка и тестирование автоматизированной системы.

Руководитель работы

доцент кафедры ИСТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_О.К. Головнин

(подпись)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_\_\_г.

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Д. Мавлютов

(подпись)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_\_\_г.

РЕФЕРАТ

Пояснительная записка 90 с, 42 рисунка, 4 таблицы, 72 источника, 3 приложения.

Презентация: 23 слайда Microsoft PowerPoint.

АВТОМАТИЗИРОВАННАЯ СИСТЕМА, ТРАНСПОРТНЫЙ ПОТОК, СВЕРТОЧНАЯ НЕЙРОННАЯ СЕТЬ, АКУСТИЧЕСКОЕ ИЗЛУЧЕНИЯ, ДЕТЕКТИРОВАНИЕ ПРИЗНАКОВ, ПАТТЕРН, ШАБЛОН.

Целью выпускной квалификационной работы магистра является разработка автоматизированной системы детектирования признаков акустического излучения транспортных средств на аудиозаписях с использованием сверточных нейронных сетей.

Проведено исследование предметной области «Анализ транспортных потоков», рассмотрены задачи детектирования признаков и особенности акустического излучения, проанализированы нейросетевые технологии и алгоритмы дискретного преобразования сигналов, применяемые в решении аналогичных задач. Выполнен аналитический обзор существующих систем-аналогов. Разработан метод классификации объектов по их акустическому излучению. Построены диаграммы по методологии UML, создана модель данных, разработаны алгоритмы функционирования и описана архитектура системы. Разработана автоматизированная система детектирования признаков акустического излучения транспортных средств на аудиозаписях с использованием сверточных нейронных сетей. Проведены исследования эффективности разработанной системы для решения поставленной задачи. Программное обеспечение системы разработано на языках Python и JavaScript в среде Visual Studio Code.

СОДЕРЖАНИЕ

[Определения, обозначения и сокращения 8](#_Toc41306003)

[Введение 9](#_Toc41306004)

[1 Анализ исследуемой задачи и направления ее решения 12](#_Toc41306005)

[1.1 Анализ предметной области 12](#_Toc41306006)

[1.1.1 Транспортный поток 12](#_Toc41306007)

[1.1.2 Состав транспортного потока 13](#_Toc41306008)

[1.1.3 Классификация транспортных средств 13](#_Toc41306009)

[1.1.4 Задача детектирования признаков 16](#_Toc41306010)

[1.1.5 Акустическое излучение 18](#_Toc41306011)

[1.2 Нейросетевые технологии 22](#_Toc41306012)

[1.2.1 Классификация нейронных сетей 22](#_Toc41306013)

[1.2.2 Модель нейрона 23](#_Toc41306014)

[1.2.3 Функция активации нейрона 24](#_Toc41306015)

[1.2.4 Сверточные нейронные сети и их особенности 25](#_Toc41306016)

[1.3 Алгоритмы дискретного преобразования сигналов 26](#_Toc41306017)

[1.3.1 Дискретное преобразование Фурье 26](#_Toc41306018)

[1.3.2 Быстрое преобразование Фурье 28](#_Toc41306019)

[1.3.3 Дискретное вейвлет-преобразование 29](#_Toc41306020)

[1.4 Мел-частотные кепстральные коэффициенты 31](#_Toc41306021)

[1.4.1 Мел 31](#_Toc41306022)

[1.4.2 Кепстр 33](#_Toc41306023)

[1.4.3 Вычисление мел-частотных кепстральных коэффициентов 34](#_Toc41306024)

[1.5 Описание систем-аналогов 36](#_Toc41306025)

[1.5.1 ShotSpotter 36](#_Toc41306026)

[1.5.2 Shazam 38](#_Toc41306027)

[1.5.3 AudioAnalytics 40](#_Toc41306028)

[1.6 Постановка задачи 41](#_Toc41306029)

[1.7 Описание методологии проектирования системы 42](#_Toc41306030)

[1.8 Выбор комплекса программных средств 42](#_Toc41306031)

[1.8.1 Выбор среды проектирования 43](#_Toc41306032)

[1.8.2 Выбор языка программирования 43](#_Toc41306033)

[1.8.3 Выбор среды программирования 43](#_Toc41306034)

[1.8.4 Выбор системы управления базами данных 44](#_Toc41306035)

[2 Проектирование и программная реализация автоматизированной системы 46](#_Toc41306036)

[2.1 Разработка метода детектирования признаков (паттернов) в аудиофайлах 46](#_Toc41306037)

[2.2 Архитектура системы 49](#_Toc41306038)

[2.3 Проектирование информационно-логической модели системы 51](#_Toc41306039)

[2.3.1 Диаграмма вариантов использования 51](#_Toc41306040)

[2.3.2 Диаграмма классов 53](#_Toc41306041)

[2.3.3 Диаграмма состояний 54](#_Toc41306042)

[2.3.4 Диаграмма взаимодействий 55](#_Toc41306043)

[2.3.5 Диаграмма деятельности 58](#_Toc41306044)

[2.3.6 Разработка логической модели данных 59](#_Toc41306045)

[2.3.7 Разработка физической модели данных 59](#_Toc41306046)

[2.4 Выбор комплекса технических средств 60](#_Toc41306047)

[2.4.1 Расчет объема оперативной памяти 60](#_Toc41306048)

[2.4.2 Расчет объема дискового пространства 61](#_Toc41306049)

[2.4.3 Рекомендованные характеристики технических средств 61](#_Toc41306050)

[2.5 Разработка алгоритмов функционирования системы 62](#_Toc41306051)

[2.6 Разработка и описание интерфейса пользователя 62](#_Toc41306052)

[3 Экспериментальные исследования автоматизированной системы 69](#_Toc41306053)

[3.1 Исследование и визуализация данных 69](#_Toc41306054)

[3.1.1 Набор данных 69](#_Toc41306055)

[3.1.2 Преобразование данных 71](#_Toc41306056)

[3.2 Проведение исследования 72](#_Toc41306057)

[3.2.1 Методика исследования 72](#_Toc41306058)

[3.2.2 Инструменты получения признаков 73](#_Toc41306059)

[3.2.3 Обучение многослойного персептрона 74](#_Toc41306060)

[3.2.4 Обучение сверточной нейронной сети 75](#_Toc41306061)

[3.3 Результаты исследования 77](#_Toc41306062)

[3.4 Описание контрольного примера работы системы 77](#_Toc41306063)

[3.4.1 Назначение 77](#_Toc41306064)

[3.4.2 Исходные данные 78](#_Toc41306065)

[3.4.3 Результаты расчета 79](#_Toc41306066)

[Заключение 81](#_Toc41306067)

[Список использованных источников 83](#_Toc41306068)

[Приложение А Функции активации нейронов 91](#_Toc41306069)

[Приложение Б Руководство пользователя 92](#_Toc41306070)

[Приложение В Листинг программы 97](#_Toc41306071)

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД – база данных;

ИТС – интеллектуальная транспортная система;

МЛП – многослойный персептрон;

МЧКК – мел-частотные кепстральные коэффициенты;

СНС – сверточная нейронная сеть;

СУБД – системы управления базами данных;

ТП – транспортный поток;

ТС – транспортное средство;

CD – Compact Disc;

GPS – Global Positioning System;

IBM – International Business Machines;

PDF – Portable Document Format;

RBF – Radial Basis Function;

ReLU – Rectified Linear Units;

REST – Representational State Transfer;

SVM – Support Vector Machine;

UML – Unified Modeling Language.

ВВЕДЕНИЕ

Актуальность. На сегодняшний день можно с уверенностью сказать, что информационные технологии «умного города» изменили многое в нашей жизни, и эти положительные изменения с каждым днем все заметней. Одним из примеров развития технологий в сфере управления транспортом является интеллектуальная транспортная система (ИТС), которая позволяет строить безопасную, эффективную и экологичную инфраструктуру современной городской агломерации [1]. Для построения ИТС необходим глубокий анализ различных характеристик транспортного потока (ТП).

Сбор и анализ характеристик ТП возможен лишь при наличии таких технических возможностей [2]. Сбор информации о текущем состоянии ТП осуществляется с помощью различных технических устройств: петлевых датчиков, видеокамер, детекторов. Существует множество алгоритмов обработки видеоинформации в мониторинге транспортных потоков [3]. Такой подход не лишен недостатков, поскольку обработка видеофайлов обладает огромной избыточностью информации, которая требует значительных затрат для обработки, хранения и анализа данных. Стоит отметить, что огромную часть расходов включают в себя такие пункты, как монтаж, эксплуатация и поддержка сети видеонаблюдения. Таким образом, анализ менее избыточного по информации аудиосигнала видится рациональным.

В задаче сбора и анализа аудиоданных накоплен огромный опыт. Больших успехов в области преобразования и анализа сигналов, в том числе и аудиосигналов, достигли такие ученые, как К.Ф. Гаусс, С.А. Прохоров, В.А Котельников. Выполненные исследования в области детектирования признаков в сигналах опираются на разработки ученых и практиков: Е.М. Миркеса, С. Дэвиса, П. Мермельшейтна. Значительный опыт накоплен в области интеллектуальных информационных технологий и разработки систем: Д. Кнут, Д.А. Поспелов и др. Однако данные исследования и труды не концентрировали внимание на применении нейронной сети в качестве классификатора для признаков, полученных из аудиосигналов.

Объектом исследования является процесс извлечения данных из аудиосигналов.

Предметом исследования является методы, модели, алгоритмы и средства детектирования и классификации транспортных средств (ТС) на аудиозаписях.

Целью выпускной квалификационной работы магистра является разработка автоматизированной системы детектирования признаков акустического излучения ТС на аудиозаписях с использованием сверточных нейронных сетей (СНС).

Для достижения цели в работе поставлены и решены следующие задачи:

* анализ исследуемой задачи и современных направлений ее решения;
* проектирование и программная реализация автоматизированной системы, позволяющей осуществлять подсчет интенсивности транспортных потоков за счет детектирования отдельных ТС на аудиозаписях;
* экспериментальные исследования разработанной автоматизированной системы детектирования признаков акустического излучения ТС.

Научная новизна содержится в следующих результатах:

* новый метод детектирования признаков (паттернов) акустического излучения транспортных средств в аудиофайлах путем получения мел-частотных кепстральных коэффициентов в качестве признаков, классифицируемых с помощью СНС;
* алгоритмы функционирования и программное обеспечение автоматизированной системы, реализующие предложенный метод детектирования признаков.

Основной практический результат работы – разработанная автоматизированная система детектирования акустического излучения ТС на аудиофайлах с использованием СНС, которая может применяться для решения задачи анализа характеристик ТП.

Теоретической и методологической основой исследования методы обработки сигналов и распознавания речи, методы и технологии искусственного интеллекта.

Положения, выносимые на защиту:

* новый метод детектирования признаков (паттернов) акустического излучения транспортных средств в аудиофайлах, использующий СНС для классификации по признакам, формируемым мел-частотными кепстральными коэффициентами;
* алгоритмы функционирования и программное обеспечение автоматизированной системы, реализующие предложенный метод детектирования признаков.

Апробация работы. Основные результаты работы докладывались и обсуждались на международных, всероссийских и региональных конференциях: XXVI Международная конференция «Математика. Компьютер. Образование» (Пущино, 2019); VII Всероссийская научная конференция с международным участием «Информационные технологии интеллектуальной поддержки принятия решений» (Уфа, 2019); LXIX Молодежная научная конференция, посвященная 85-летию со дня рождения первого космонавта Земли Ю.А. Гагарина (Самара, 2019); Международная научно-техническая конференция «Перспективные информационные технологии» (Самара, 2019); Международная мультидисциплинарная конференция по промышленному инжинирингу и современным технологиям (Владивосток, 2019); Международная молодёжная научная конференция «XV Королёвские чтения», посвящённая 100-летию со дня рождения Д.И. Козлова (Самара, 2019); Всероссийская научно-техническая конференция «Новые информационные технологии в научных исследованиях» (Рязань, 2019).

Публикации по теме работы. Основные положения и результаты работы изложены в 8 научных публикациях, включающих 2 статьи в изданиях, индексируемых Scopus и Web of Science.

1. Анализ исследуемой задачи и направления ее решения
   1. Анализ предметной области
      1. Транспортный поток

Транспортный поток – это упорядоченное транспортной сетью движение транспортных средств (ТС). Транспортный поток (ТП) состоит из отдельных автомобилей, обладающих различными динамическими характеристиками и управляемых разными по квалификации водителями, поэтому он не является однородным.

ТП можно классифицировать по следующим признакам.

По состоянию транспортных средств:

* груженый поток, обусловленный движением транспортных средств с грузом, это производительный пробег транспорта;
* порожний поток, обусловленный движением транспортных средств без груза, это непроизводительный пробег транспорта.

По направлению движения:

* односторонний поток, обусловленный движением транспортных средств в одном направлении;
* двусторонний поток, обусловленный движением транспортных средств в прямом и обратном направлении.

По объекту перевозки:

* грузовой, обусловленный перевозками грузов конкретным видом транспорта;
* пассажирский, обусловленный перевозкой пассажиров;
* комбинированный, обусловленный перевозкой грузов и пассажиров в одном транспортом средстве.

По виду транспорта:

* железнодорожный поток, в том числе вагонопоток и контейнеропоток на железнодорожном транспорте;
* автомобильный поток (автомобилепоток);
* воздушный поток (образованный перемещением воздушного транспорта – самолетов, вертолетов);
* водный (образованный перемещением водного транспорта, морского или речного).
  + 1. Состав транспортного потока

В состав транспортного потока входят различные типы ТС. Типы ТС могу быть следующими: легковой автомобиль, грузовик до 2т, грузовик до 6т, грузовик до 8т, грузовик до 14т, грузовик более 14т, автобус, сочлененный автобус, троллейбус, сочлененный троллейбус, пассажирская газель, велосипед, трехколесный, мотоцикл / мопед, автопоезд, трактор, трамвай, трамвай с 2 трамваями.

Существуют различные способы описания состава транспортного потока , предложенные в нормативных или отраслевых распорядительных документах и актах. Для формирования набора атрибутов, описывающих состав транспортного потока в соответствии с применяемым способом, обязательна привязка к юридическому документу. Для каждого метода описания состава транспортного потока формируется набор типов ТС и коэффициентов преобразования в легковой автомобиль . Такой подход позволяет нам последовательно обрабатывать данные о составе транспортного потока, а также показать, насколько динамические показатели конкретного транспортного средства отличаются от динамических показателей легкового автомобиля.

* + 1. Классификация транспортных средств

Интеллектуальные системы мониторинга ТрП являются эффективным инструментом для снижения пробок на дорогах. Данные системы являются неотъемлемым компонентом в построении интеллектуальных транспортных систем (ИТС), которые используются для сбора данных о дорожном движении, таких как: количество ТС, типы ТС и скорость ТС. На основе собранных данных ИТС выполняют анализ дорожного движения, чтобы лучше использовать дорожные системы, прогнозировать будущие потребности в перевозках и повышать безопасность перевозок [4]. Транспортные агентства во многих странах тратят огромные деньги на разработку, развертывание и обслуживание систем мониторинга трафика [5].

На рисунке 1.1 представлена схема, на которой показана схема устройств, выполняющие детектирование и классификацию ТС.

Классификации ТС подразделяется на три класса в зависимости от того, где развернута система мониторинга: на проезжей части, над проезжей частью, и возле проезжей части. Далее, эти системы классифицируются на основе типов датчиков, а также на основе того, как данные, получаемые с датчиков, используются для классификации ТС.

Системы классификации ТС, находящиеся на проезжей части, размещают датчики на дорожном покрытии или под ним. Различные типы датчиков используются для систем классификации транспортных средств на дорогах, таких как пьезоэлектрические датчики [6], магнитометры [7], датчики вибрации [8], петлевые датчики [9]. Из датчика извлекаются различные виды информации, включая длину транспортного средства, количество осей и уникальные характеристики сигнала или формы волны. Такие системы обладают высокой точностью классификации ТС, поскольку датчики имеют непосредственный контакт с проезжающими ТС, эффективно фиксируя характеристики кузова и движения транспортных средств. Однако основным недостатком является высокая стоимость монтажа и технического обслуживания, поскольку для укладки датчиков под проезжей частью необходима специальная техника и материалы для размещения устройства.

Системы, располагающиеся на обочине проезжей части решают проблему стоимости схем классификации ТС на проезжей части. В данных системах тоже используются различные типы датчиков, включая: магнитометры [10], акселерометры [11] и акустические датчики [12].

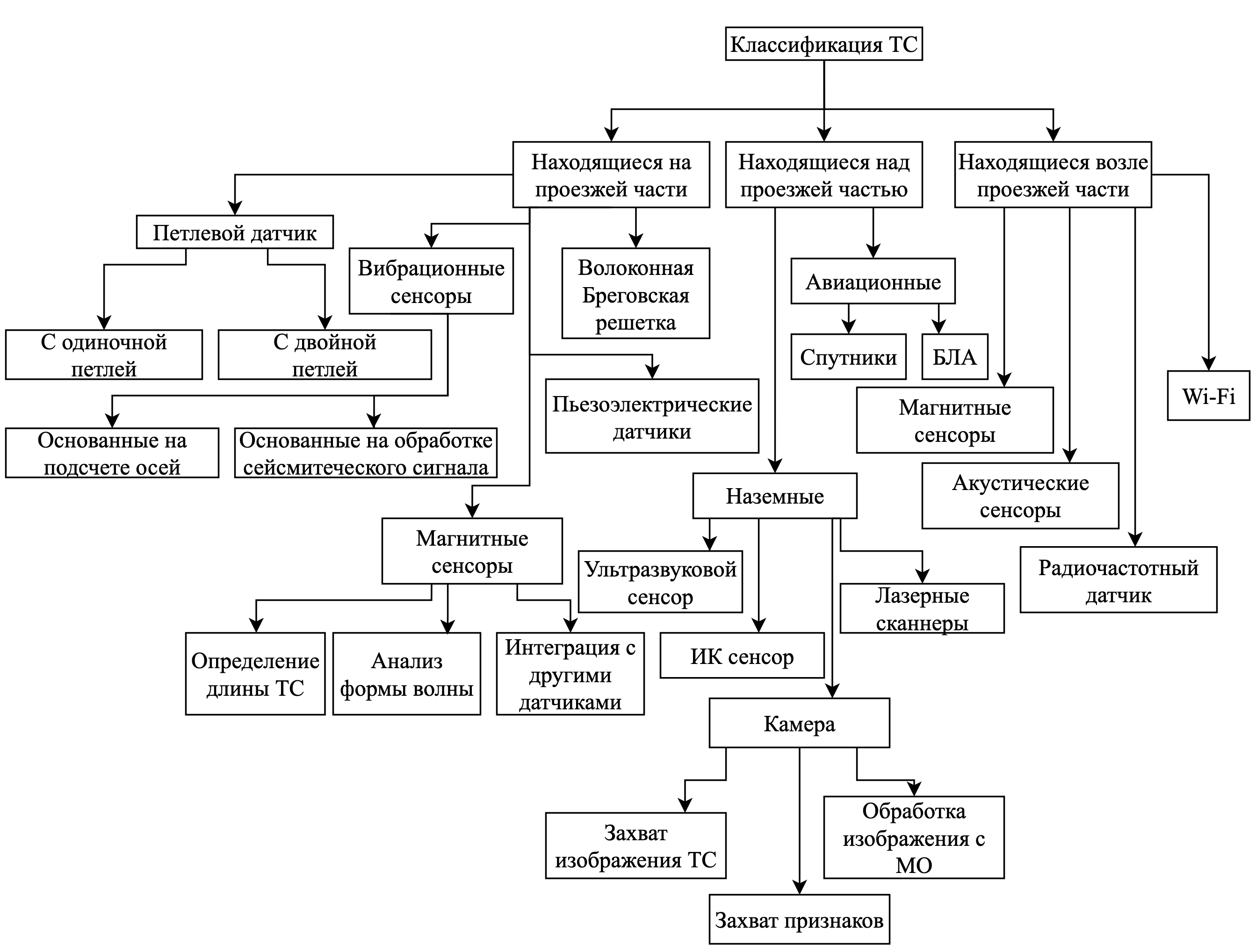


Рисунок 1.1 – Схема устройств классификации ТС

В последнее время были использованы современные датчики, такие как лазерное инфракрасное обнаружение и определение дальности (LIDAR) [13], инфракрасные датчики [14] и приемопередатчики Wi-Fi [15]. Несмотря на преимущества в виде более простой установки и более низкой стоимости, данные системы требуют точной настройки и правильного расположения датчиков. Друга проблема заключается в том, что большинство систем не могут точно классифицировать ТС, которые были перекрыты другим ТС. Кроме того, необходим алгоритм для обработки данных датчика, чтобы уменьшить влияние шума и повысить точность классификации.

В системах, располагающихся над проезжей частью, имеется возможность покрывать сразу несколько полос движения. Например, в этих системах используются беспилотные летательные аппараты и спутники [16]. Наиболее распространенной технологией в этой категории являются системы на основе камер [17]. Хотя основанные на камерах системы имеют высокую точность классификации, на производительность влияют погодные условия и условия освещения. Еще одна важная проблема — это проблемы с конфиденциальностью водителя, поскольку это может доставлять дискомфорт водителям, а также в некоторых странах подобная съёмка может быть запрещена. Решить проблему конфиденциальности можно используя различные типы датчиков, такие как инфракрасные датчики и лазерный сканер [18].

* + 1. Задача детектирования признаков

Детектирование или выделение признаков – это разновидность абстрагирования, процесс снижения размерности, в котором исходный набор исходных переменных сокращается до более управляемых групп (признаков) для дальнейшей обработки, оставаясь при этом достаточным набором для точного и полного описания исходного набора данных [19]. Выделение признаков используется в машинном обучении, распознавании образов и при обработке изображений. Выделение признаков начинает с исходного набора данных, выводит вторичные значения ([признаки](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%B7%D0%BD%D0%B0%D0%BA_(%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD))), для которых предполагается, что они должны быть информативными и не быть избыточными, что способствует последующему процессу обучения машины и обобщению шагов, а в некоторых случаях ведёт и к лучшей человеческой интерпретацией данных. Выделение данных также необходимо для сокращений итераций переобучения нейронной из-за большого большого числа входных данных.

Когда входные данные алгоритма слишком большие для обработки и есть подозрение, что данные избыточные (например, измерения проведены как в футах, так и в метрах, или повторяемость изображений представлена пикселами), то они могут быть преобразованы в сокращённый набор признаков (называемый вектором признаков). Определение подмножества начальных признаков называется отбором признаков [20]. Отобранные признаки проверяются на содержание необходимой информации во входных данных, так что желаемая задача может быть выполнена с помощью этого сокращённого набора вместо исходных полных данных.

Методы выделения признаков делятся на несколько категорий [21], схема методов представлена на рисунке 1.2:

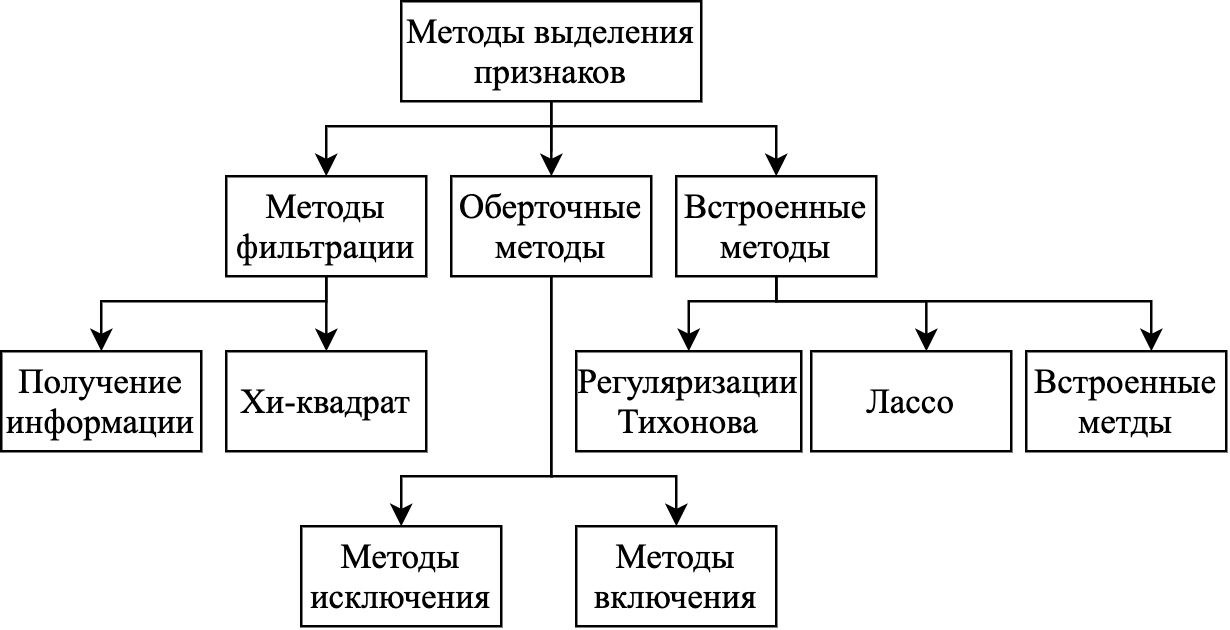


Рисунок 1.2 – Методы выделения признаков

* методы фильтрации: они применяют статистическую меру для присвоения оценки каждому объекту. Объекты ранжируются по выбранной величине, и либо сохраняются, либо удаляются из набора данных. Методы часто являются одномерными и рассматривают функцию независимо или с учетом зависимой переменной;
* оберточные методы: рассматривают выбор набора признаков как проблему поиска, где различные комбинации подготавливаются, оцениваются и сравниваются с другими комбинациями. Процесс поиска может быть методическим (например, как «лучший сначала»), может быть стохастическим (алгоритм случайного набора высоты), или эвристическим (прямой и обратные проходы для удаления и добавления объектов);
* встроенные методы определяют, какие функции лучше всего влияют на точность во время создания самой модели. Наиболее распространенными встроенными методами определения признаков являются методы регуляризации. Методы регуляризации также часто называют штрафующими методами, потому что они вводят дополнительные ограничения в оптимизацию алгоритма прогнозирования, который смещает модель в сторону меньшей сложности.
  + 1. Акустическое излучение

Само по себе излучение представляет собой передачу энергии в форме волн или частиц через пространство или материальную среду [22]. Звуковое излучение – это физическое явление, представляющее собой распространение в виде упругих волн механических колебаний в твердой, жидкой или газообразной среде. В некоторым смысле под звуком рассматривают данные колебания в связи с тем, как они воспринимаются органами чувств различных животных [23].

Как и любая волна, звуковое излучение характеризуется амплитудой и частотой. Амплитуда отвечает за громкость звука, а частота в свою очередь определяет тон и высоту звука. Человеческие способен воспринимать колебания от 16-20 Гц до 15-20 кГц. Звуком ниже диапазона слышимости человека называют инфразвуком, а выше и до 1 ГГц – ультразвуком. Звуком с частотой выше 1 ГГц называют гиперзвук. Громкость звука сложным образом зависит от эффективного звукового давления, частоты и формы колебаний, а высота звука – не только от частоты, но и от величины звукового давления.

Когда говорим о звуковом излучении, то стоит отметить методы, как можно данное звуковое излучение конвертировать в аудиофайл. Аудиофайл – компьютерный файл, состоящий из информации об амплитуде, сохраненной для дальнейшего воспроизведения или анализа.

Цифровой звук – результат преобразования аналогового сигнала звукового диапазона в цифровой аудиоформат. Формат файла определяет структуру и особенности представления звуковых данных при хранении на запоминающем устройстве вычислительной машины. Группы звуковых форматов файлов показаны на рисунке 1.3.



Рисунок 1.3 – Классификация форматов файлов

Касательно звукового излучения ТС, основными источниками шума являются:

* силовая установка – двигатель (корпус двигателя, процесс сгорания, шум механизма газораспредетилея, системы впуска и выпуска двигателя;
* топливоподающая аппаратура;
* вентилятор системы охлаждения двигателя;
* трансмиссия (коробка передач и задний мост);
* шины;
* вспомогательное оборудование;
* музыка из салона автомобиля;
* дополнительные звуки при неисправном состоянии ТС.

В таблице 1.1 представлено сравнение популярных форматов аудиозаписей.

Таблица 1.1 – Виды цифрового звука

|  |  |  |  |
| --- | --- | --- | --- |
| Название формата | Разрядность, бит | Частота дискретизации, кГЦ | Число каналов |
| CD | 16 | 44,1 | 2 |
| AC3 | 16-24 | 48 | 6 |
| DTS | 20-24 | 48;96 | до 8 |
| DVD-Audio | 16;20;24 | 44,1; 48; 88,2; 96 | 6 |
| MP3 | 16-24 | до 48 | 2 |
| AAC | 16-24 | до 96 | до 48 |
| AAC+ | 16-24 | до 48 | 2 |
| Ogg Vorbis | до 32 | до 192 | до 255 |
| WMA | до 24 | до 96 | до 2 |
| WAV | 8; 16; 24; 32 | любая | 1;2;3;4;6 |

Зависимость шума, издаваемого ТС, от скорости движения описывается сложной функцией. При небольших скоростях (в пределах 50-60 км/ч) основным источником звукового излучения является двигатель и смежные агрегаты [24]. В то же время, при высоких скоростях наибольшую интенсивность акустического излучения генерируют шины ТС при взаимодействии с дорожным покрытием [25].

Хотя шины и двигатели грузовых и легковых автомобилей сильно отличаются, как видно из спектрограмм, что проезжающие мимо звукозаписывающего устройства грузовой (рисунок 1.4) и легковой (рисунок 1.5) автомобили имеют идентичную форму звукового сигнала. Данная схожесть усложняет процесс классификации.

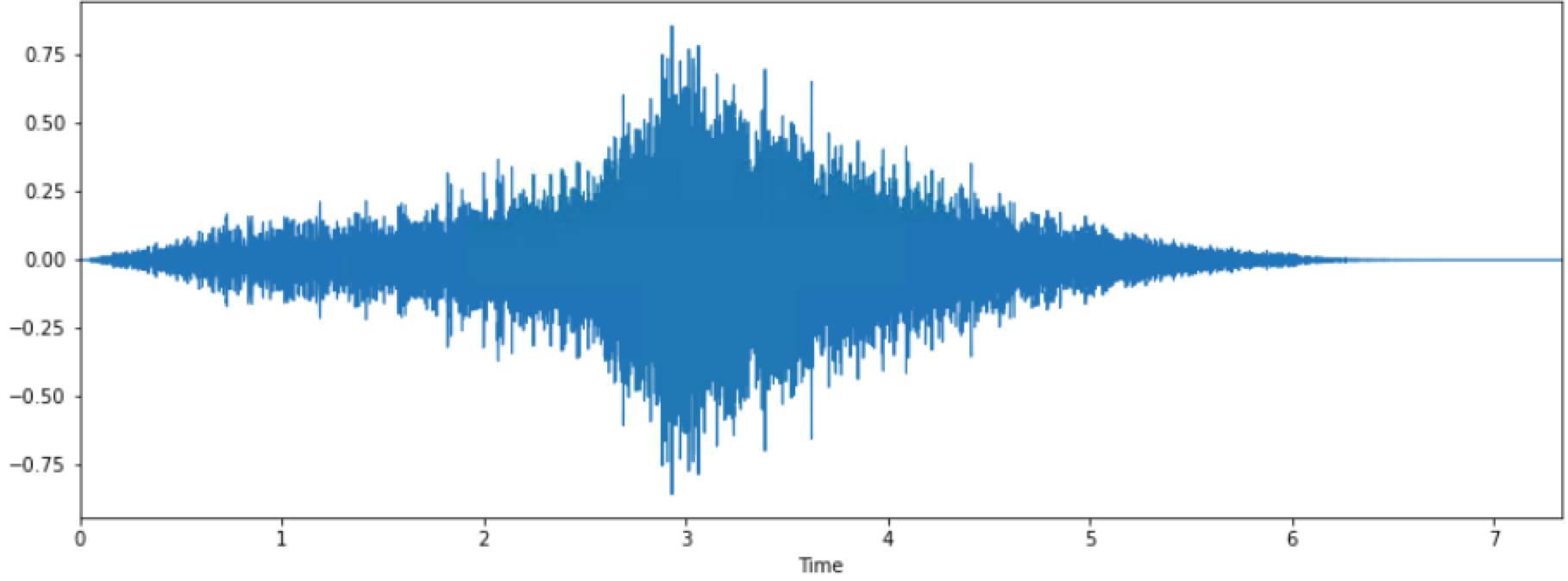


Рисунок 1.4 – Спектрограмма акустического излучения   
проезжающего грузового автомобиля

Анализ звукового излучения является важнейшим инструментом для моделирования и дальнейшего вычленения необходимых характеристик из аудиофайла [26, 27].

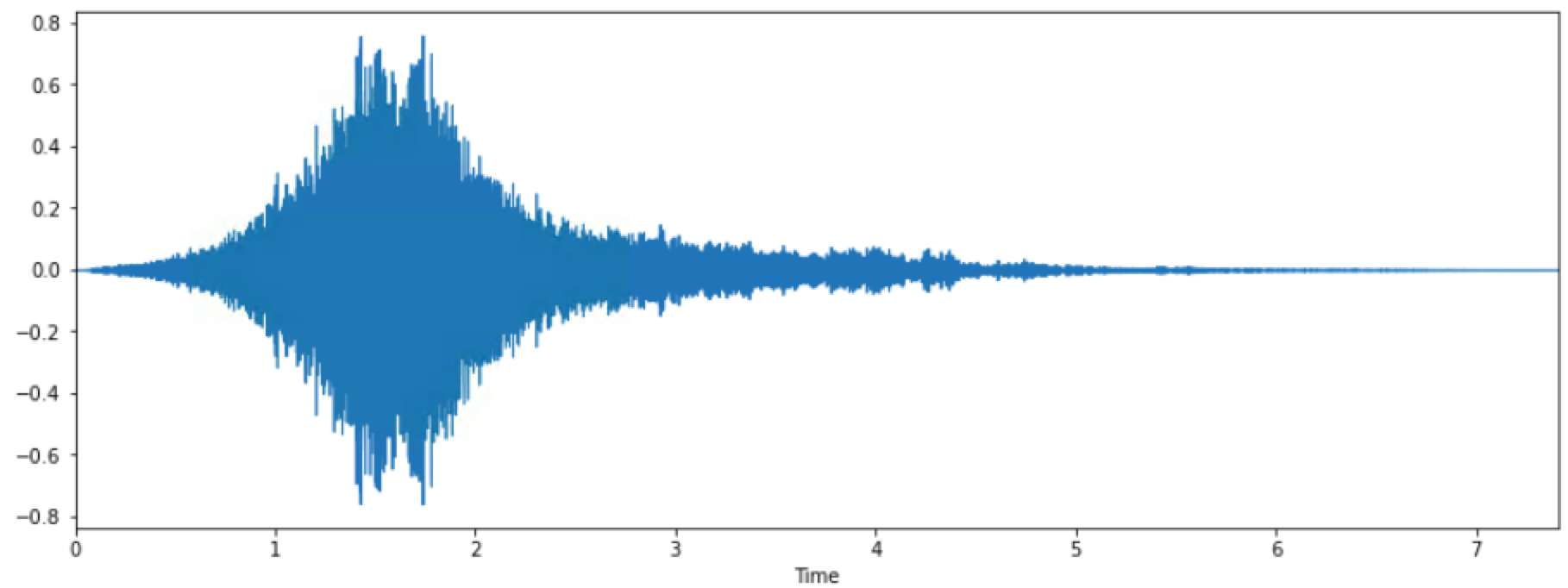


Рисунок 1.5 – Спектрограмма акустического излучения   
проезжающего легкового автомобиля

* 1. Нейросетевые технологии
     1. Классификация нейронных сетей

Нейронная сеть – вычислительная сеть, основанные на биологических нейронных сетях, которые составляют головной мозг различных животных [28, 29]. Подобные системы обучаются, чтобы выполнять задачи аналогичные тем, на которых они обучались [30, 31]. Например, в задаче распознавании образов нейронные сети могут научиться распознавать изображения, которые содержат кошек, анализируя примеры, которые были размечены как «содержащие кошку» или «не содержащие кошку». Нейронная сеть решает задачу распознавания без предварительных знаний о кошках, например, что у них есть хвост, шерсть, усы, особая форма лица и прочие идентификаторы кошек. Вместо этого сеть сама генерирует необходимые идентифицирующие признаки из примеров, на которых они обучаются.

Нейронные сети применяются для решения следующих задач:

* распознавание образов и классификация;
* принятие решений и управление;
* кластеризация;
* прогнозирование;
* аппроксимация;
* сжатие данных и ассоциативная память;
* анализ данных;
* хемоинформатика;
* оптимизация.

Различные архитектуры нейронных сетей предназначены для решения определенного класса задач. Сети классифицируются по типу входной информации, характеру обучения, характеру настройки синапсов, времени передачи сигнала и характеру связей. Далее представлены самые распространённые топологии нейронных сетей:

* нейронные сети прямого распространения и персептроны;
* сети радиально-базисных функций;
* нейронная сеть Хопфилда;
* цепи Маркова;
* машина Больцмана;
* автокодировщик;
* сверточные нейронные сети
* глубинные сверточные нейронные сети;
* рекуррентные нейронные сети;
* самоорганизующиеся карты;
* развертывающие нейронные сети.
  + 1. Модель нейрона

На рисунке 1.6 представлена модель нейрона.

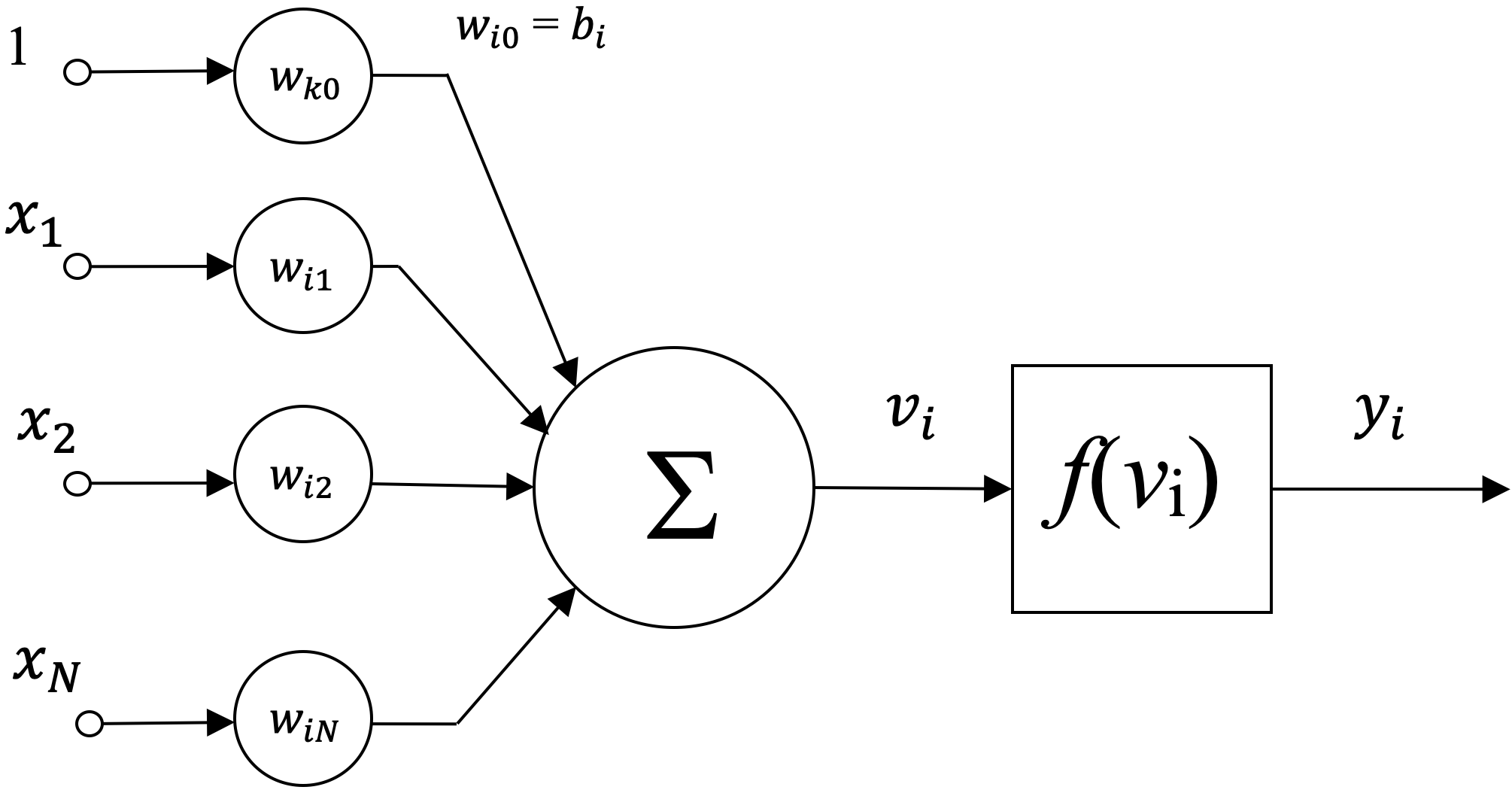


Рисунок 1.6 – Модель нейрона

Нейроны делятся на три основных типа [32]: входной, скрытый и выходной. Также существуют нейроны смещения и контекстные нейроны. В том случае, когда нейросеть состоит из большого количества нейронов, вводят термин слоя. Соответственно, есть входной слой, который получает информацию, N скрытых слоев, которые ее обрабатывают и выходной слой, который выводит результат. У каждого из нейронов есть 2 основных параметра: входные данные и выходные данные. В случае входного нейрона входные данные равны выходным. В остальных, во входные данные попадает суммарная информация всех нейронов с предыдущего слоя, после чего, она нормализуется, с помощью функции активации и попадает в выходные данные.

* + 1. Функция активации нейрона

Нейрон представляет собой математическую функцию, построенную на основе модели биологического нейрона [33]. Нейрон является элементарной единицей в нейронной сети, которая получает один или несколько входных сигналов и суммирует их для получения выходного сигнала.

Обычно каждый вход взвешивается отдельно, а сумма передается через нелинейную функцию более известную как функция активации. Функция активации f ограничивает амплитуду выходного сигнала нейрона. Можно выделить три основных типа функций активации:

* кусочно-линейная функция, описывается формулой (1):

; (1)

* гиперболическая функция активации:

; (2)

* функция единичного скачка, или пороговая функция, которая описывается следующим образом:

;(3)

* сигмоидальная функция. Она является самой распространенной функцией, используемой для обучения нейронных сетей, так как позволяет как усиливать слабые сигналы, так и не насыщаться от сильных сигналов. Производная сигмоиды может быть легко выражена через саму функцию, что позволяет существенно сократить вычислительную сложность методов обучения сети. Примерами сигмоидальной функции могут служить логистическая функция активации:

(4)

где − параметр наклона сигмоидальной функции.

Вид функции активации во многом определяет функциональные возможности нейронной сети и метод обучения этой сети. В таблице А.1 (Приложение А) представлен перечень наиболее известных функций активации [34].

* + 1. Сверточные нейронные сети и их особенности

Сверточная нейронная сеть (СНС) является специальной архитектурой нейронных сетей, предложенная Яном Лекуном [35], изначально ориентированной на эффективное распознавание образов в изображениях [36], входит в состав технологий глубокого обучения. За основу архитектуры были взяты особенности зрительной коры [37], в которой были обнаружены так называемые простые клетки, которые реагирует на прямые линии под разными углами, и сложные клетки, реакция которых связана с активацией набора простых клеток. Идея сверточной СНС заключается в том, что чередуются между собой сверточные слои и субдискретизирующие слои.

Структура у сети является однонаправленной, а это значит, что в ней нет обратных связей, также она является принципиально многослойной. При обучении используются различные методы, однако чаще всего выбирают метод обратного распространения ошибки. СНС не ограничивает разработчика в выборе функции активации нейрона, а это значит, что можно выбрать абсолютно любую функцию.

Архитектура сети получила такое название из-за наличия операции свертки, суть которой заключается в том, что каждый фрагмент рассматриваемого изображения перемножается на ядро свертки поэлементно, а результат умножения суммируется и записывается в аналогичную позицию выходного изображения. Схема архитектуры представлена на рисунке 1.7.

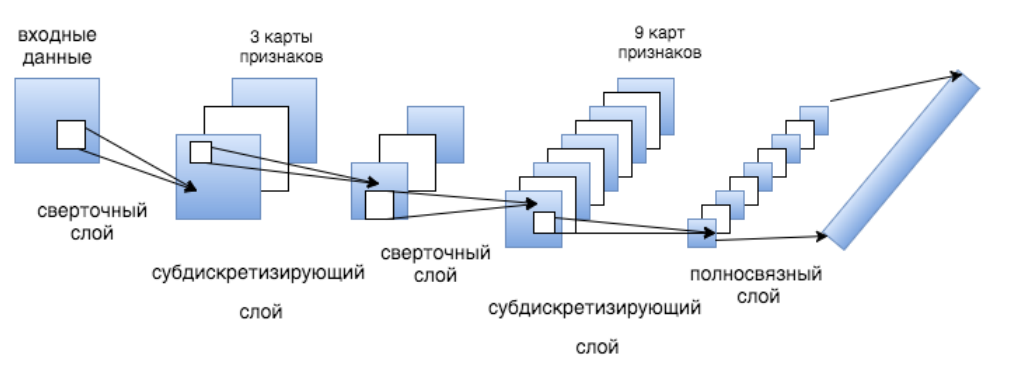


Рисунок 1.7 – Архитектура сверточной нейронной сети

* 1. Алгоритмы дискретного преобразования сигналов

Для того, чтобы можно было совершать анализ аудиофайлов, необходимо их преобразовать из аналогового формата в дискретный. Для это существуют алгоритмы дискретного преобразования.

* + 1. Дискретное преобразование Фурье

Дискретное преобразование – это одно из преобразований Фурье, широко применяемых в алгоритмах цифровой обработки сигналов (его модификации применяются в сжатии звука в MP3, сжатии изображений в JPEG и множества других форматов), а также в других областях, связанных с анализом частот в дискретном (к примеру, оцифрованном аналоговом) сигнале. Дискретное преобразование Фурье требует в качестве входа дискретную функцию. Такие функции часто создаются путём дискретизации. Дискретные преобразования Фурье помогают решать дифференциальные уравнения в частных производных и выполнять операцию свертки. Дискретные преобразования Фурье также активно используются в статистике, при анализе временных рядов. Существуют многомерные дискретные преобразования Фурье [38, 39].

Прямое преобразование выглядит следующим образом:

(5)

при

Обратное преобразование:

(6)

при Вторая часть выражения следует из первой по формуле Эйлера.

Для обоих преобразований справедливы следующие обозначения:

* – количество значений сигнала, измененных за период, а также компонент разложения;
* , – измененные значения сигнала, которые являются входными данными для прямого преобразования и выходными для обратного;
* – N комплексных амплитуд синусоидальных сигналов, слагающих исходный сигнал; являются выходными данными для прямого преобразования и входными для обратного; поскольку амплитуды комплексные, то по ним можно вычислить одновременно амплитуду и фазу;
* – вещественная амплитуда -го синусоидального сигнала;
* – фаза -го синусоидального сигнала;
* – индекс частоты. Частота -го сигнала равна , где – период времени, в течение которого брались входные данные.

Из последнего видно, что преобразование раскладывает сигнал на синусоидальные составляющие (которые называются гармониками) с частотами колебаний за период до одного колебания за период. Поскольку частота дискретизации сама по себе равна N отсчётов за период, то высокочастотные составляющие не могут быть корректно отображены – возникает муаровый эффект. Это приводит к тому, что вторая половина комплексных амплитуд, фактически, является зеркальным отображением первой и не несёт дополнительной информации.

* + 1. Быстрое преобразование Фурье

Быстрое преобразование Фурье представляет собой алгоритм ускоренного вычисления дискретного преобразования Фурье, позволяющий получить результат за время, меньшее чем . Иногда под быстрым преобразованием Фурье понимается один из алгоритмов, называемый алгоритмом прореживания по частоте – времени, имеющий сложность .

При применении основного алгоритма дискретное преобразование Фурье [40] может быть выполнено за действий при , в частности, при понадобится действий.

Дискретное преобразование Фурье преобразует набор чисел , в набор чисел , такой, что , где – первообразный корень из единицы, то есть и при 0<k<n. Основной шаг алгоритма состоит в сведении задачи для чисел к задаче с меньшим числом. Для над полем комплексных чисел вводятся: , , где – любое число.

Таким образом, дискретное преобразование Фурье может быть представлено в виде:

. (7)

Далее вычисляется каждое , где , здесь по-прежнему требуется совершить действий, то есть на этом этапе производится операций.

Алгоритм быстрого преобразования Фурье логично применять для , потому как при малом числе отсчётов он даёт небольшой выигрыш в скорости по отношению к прямому расчёту дискретного преобразования Фурье. Для

*.* (8)

Обозначая , , получается, если положить , при :

. (9)

* + 1. Дискретное вейвлет-преобразование

Дискретное вейвлет-преобразование является реализацией вейвлет-преобразования с использованием дискретного набора масштабов и переносов вейвлета, подчиняющихся некоторым определённым правилам. Другими словами, это преобразование раскладывает сигнал на взаимно ортогональный набор вейвлетов, что является основным отличием от непрерывного вейвлет-преобразования, или его реализации для дискретных временных рядов, иногда называемой непрерывным вейвлет-преобразованием дискретного времени.

Вейвлет может быть сконструирован из функции масштаба, которая описывает свойства его масштабируемости. Ограничение состоит в том, что функция масштаба должна быть ортогональна к своим дискретным преобразованиям, что подразумевает некоторые математические ограничения на них, которые везде упоминаются, т.е. уравнение гомотетии:

*,* (10)

где S – фактор масштаба (обычно выбирается как 2).

Более того, площадь под функцией должна быть нормализована и функция масштабирования должна быть ортогональна к своим численным переносам, т. е.

*.* (11)

После введения некоторых дополнительных условий (поскольку вышеупомянутые ограничения не приводят к единственному решению), получается результат всех этих уравнений, т. е. конечный набор коэффициентов , которые определяют функцию масштабирования, а также вейвлет. Вейвлет получается из масштабирующей функции как N, где N – четное число. Набор вейвлетов затем формирует ортонормированный базис, который используется для разложения сигнала. Следует отметить, что обычно только некоторые коэффициент , будут ненулевыми, что упрощает вычисление.

Существует несколько видов реализации алгоритма дискретного вейвлет-преобразования. Самый старый и наиболее известный – алгоритм Малла (пирамидальный). В этом алгоритме два фильтра – сглаживающий и несглаживающий составляются из коэффициентов вейвлета и эти фильтры рекуррентно применяются для получения данных для всех доступных масштабов. Если используется полный набор данных и длина сигнала равна , сначала рассчитываются данные для масштаба , затем данные (D/2)/2 для масштаба ,… пока в конце не получится 2 элемента данных для масштаба L/2. Результатом работы этого алгоритма будет массив той же длины, что и входной, где данные обычно сортируются от наиболее крупных масштабов к наиболее мелким.

Дискретное вейвлет-преобразование может использоваться для простого и быстрого удаления шума с зашумлённого сигнала. Если брать только ограниченное число наиболее высоких коэффициентов спектра дискретного вейвлет-преобразования, и провести обратное вейвлет-преобразование (с тем же базисом) получится сигнал очищенный от шума. Есть несколько способов как выбрать коэффициенты, которые нужно сохранить. Для определения порога в этих методах определяется оценка дисперсии шума, заданную следующей формулой:

*,* (12)

где соответствует всем коэффициентам наиболее высокого поддиапазона масштаба разложения (где должно располагаться наибольшее количество шума). Или же дисперсия шума может быть получена независимым путем, например, как дисперсия сигнала.

* 1. Мел-частотные кепстральные коэффициенты

Для повышения эффективности анализа аудиосигналов резонно рассмотреть мел-частотные кепстральные коэффициенты (МЧКК) сигналов [29, 41].

* + 1. Мел

Мел – психофизическая единица высоты звука, применяемая в основном в музыкальной акустике. Название происходит от слова мелодия [42]. В связи с тем, что график амплитудно-частотная характеристика человеческого уха не похож на прямую, поэтому амплитуда не является точной мерой громкости звука. Поэтому и была введена эмперически подобранная единица громкости, например, как фон. Контур равных громкостей представлен на рисунке 1.8.

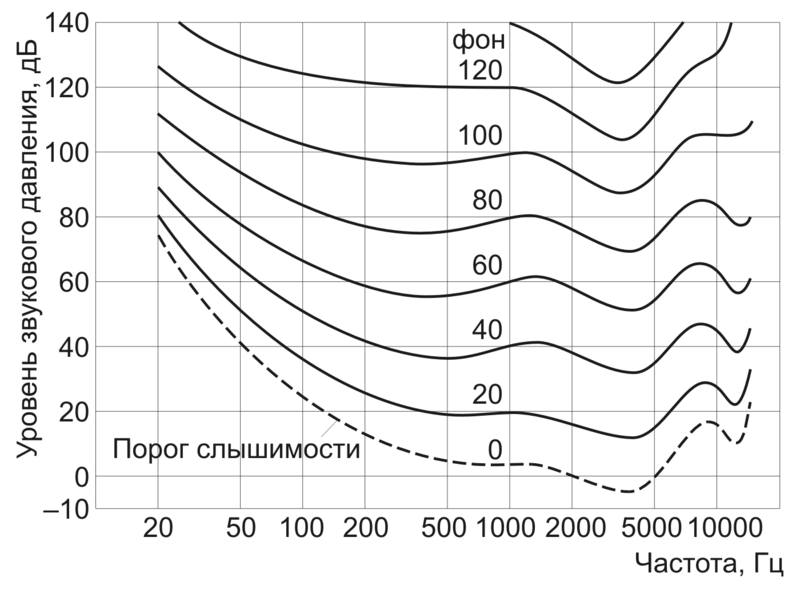


Рисунок 1.8 – Контур равных громкостей

Количественная оценка звука по высоте основана на статистической обработке большого числа данных о субъективном восприятии высоты звуковых тонов. Результаты исследований показывают, что высота звука связана главным образом с частотой колебаний, но зависит также от уровня громкости звука и его тембра. Звуковые колебания частотой 1000 Гц при эффективном звуковом давлении 2⋅10−3 Па (то есть при уровне громкости 40 фон), воздействующие спереди на наблюдателя с нормальным слухом, вызывают у него восприятие высоты звука, оцениваемое по определению в 1000 мел. Звук частоты 20 Гц при уровне громкости 40 фон обладает по определению нулевой высотой (0 мел). Зависимость нелинейна, особенно при низких частотах (для «низких» звуков). На рисунке 1.9 представлен график зависимости высоты звука в мелах от частоты колебаний.

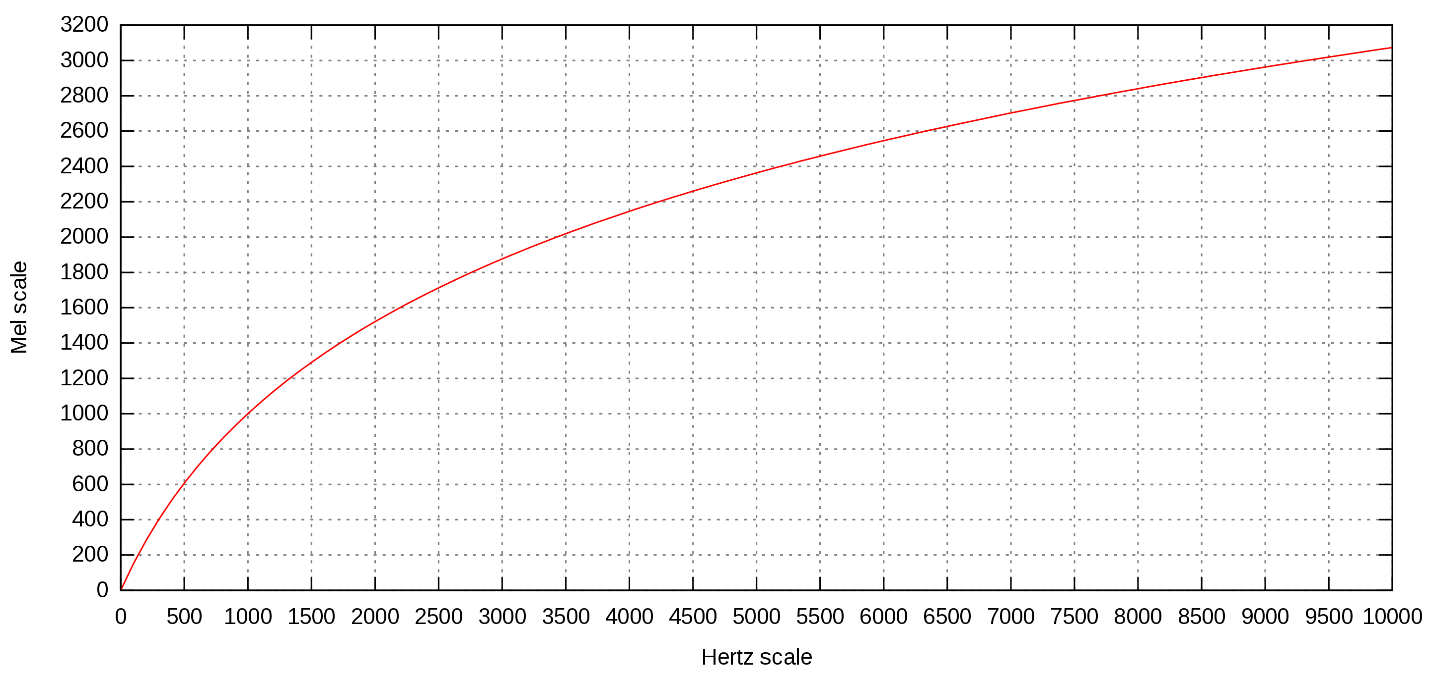


Рисунок 1.9 – График зависимости высоты звука в мелах   
от частоты колебаний

Существуют различные варианты формул для перехода между частотой в Гц и высотой звука в мелах [43]. Распространенный вариант данного преобразования выглядит следующим образом [44]:

*.* (13)

* + 1. Кепстр

Кепстр – один из видов гомоморфной обработки сигналов [45], функция обратного преобразования Фурье от логарифма спектра мощности сигнала [46]. Кепстр можно представить следующим выражением:

,(14)

где – спектр входного сигнала.

Аргумент имеет размерность времени, но оно является кепстральным временем, поскольку в любой момент зависит от функции исходного сигнала со спектром , заданной при .

* + 1. Вычисление мел-частотных кепстральных коэффициентов

В обработке звука мел-частотный кепстр представляет собой кратковременный спектр мощности звука, основанный на линейном косинусном преобразовании логарифмического спектра мощности в нелинейной мел шкале от частоты.

МЧКК это коэффициенты, которые в комбинации представляют собой мел-частотный кепстр [41]. Коэффициенты получаются из типа кепстрального представления аудиосигнала (нелинейный «спектр спектра»). Разница между кепстром и мел-частотным кепстром заключается в том, что диапазон частот мел-частотного кепстра равномерно распределен по мел шкале, которая соотносится со слуховой системой человека больше, чем равномерно распределенные частотные диапазоны в обычном кепстре. Данное искажение частоты может обеспечить более качественное представление звука при компрессии аудиофайла.

Алгоритм получения МЧКК строится следующим образом [47]: необходимо для начала получить спектр исходного сигнала ( с помощью преобразования Фурье:

(15)

Полученный спектр раскладывается на мел-шкале. Для этого используются окна, расположенные на мел-оси:

, (16)

частоты для которой получаются из равенства:

.(17)

Далее рассчитываем по формуле энергию каждого окна:

,(18)

где M – количество фильтров, которые мы хотим получить. После этого применяется дискретное косинусное преобразование для получения набора МЧКК:

.(16)

Пример полученной последовательности показан на рисунке 1.10.

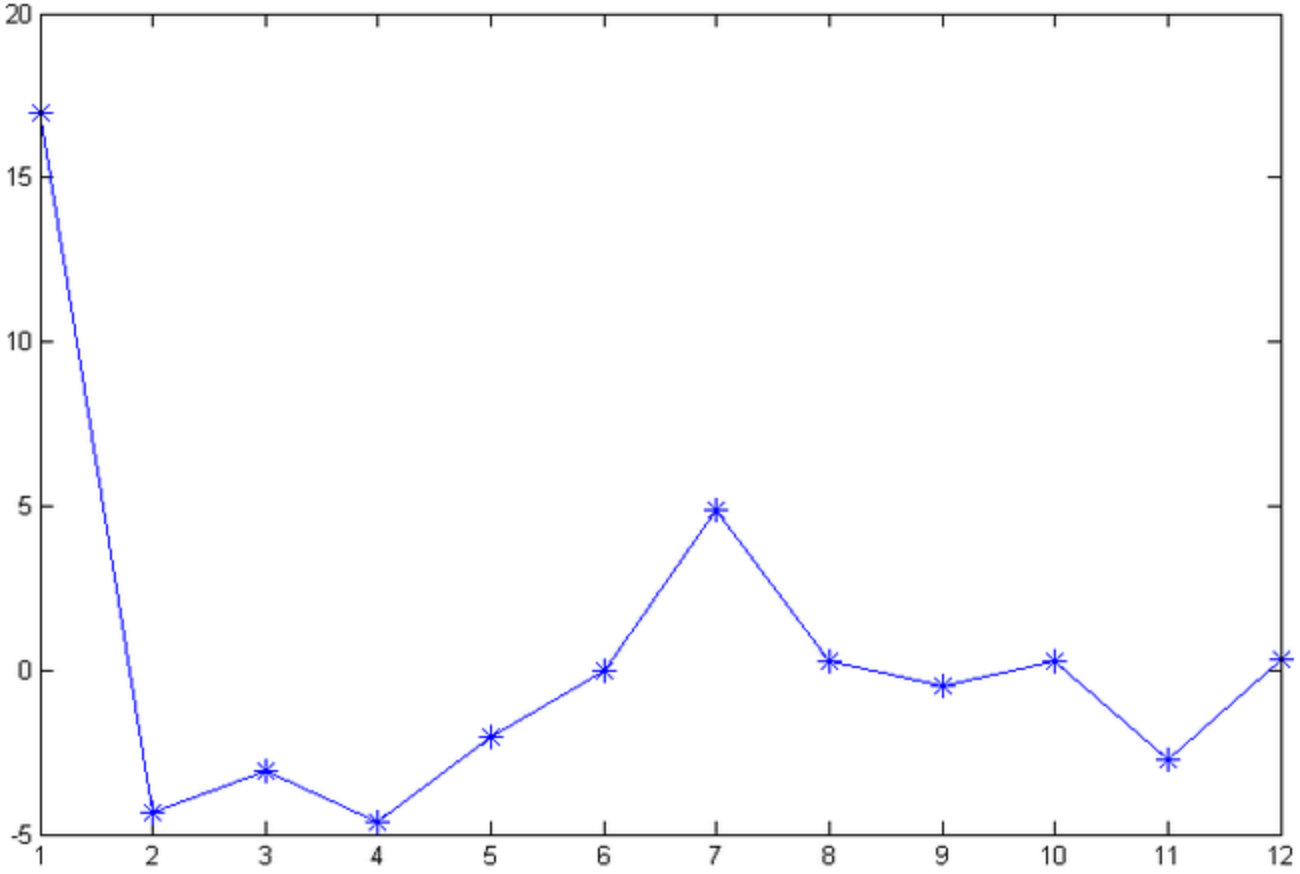


Рисунок 1.10 – Последовательность мел-частотных кепстральных коэффициентов

* 1. Описание систем-аналогов

Задача определения детектирования сигнала возникала еще до начала Второй мировой войны. Как частный случай этой задачи можно выделить определение выстрела, а также определение его источника. Лишь в 90-ых годах удалить разработать и внедрить подобную систему.

* + 1. ShotSpotter

В полиции утверждают, что далеко не всегда люди звонят в полицию, когда где-то стреляют. Если же и звонят, то может пройти достаточно много времени, пока человек решится позвонить, а также пока полиция приедет на место выстрела. В качестве решения проблемы используется система акустического наблюдения в режиме реального времени — ShotSpotter.

Система начала свою работу в Вашингтоне, где её установили еще в 2006 году. ShotSpotter хорошо себя зарекомендовал и позволил обнаружить огромное количество выстрелов из огнестрельного оружия, что позволило полиции быстро отреагировать в каждом конкретном случае. Сама система устроена следующим образом: на домах, столбах и прочих высоких сооружениях устанавливаются направленные микрофоны, которые улавливают все звуки. Пример размещения устройства изображен на рисунке 1.11.

Акустическая точка, если так можно выразиться, оснащается соответствующим программным обеспечением, проводящим предварительную идентификацию резких звуков, а также GPS, что позволяет точно указать место, где производился выстрел. В случае положительной идентификации выстрела, информация передается на центральный компьютер, где проводится дополнительный анализ звука (чтобы отсеять возможные ложные срабатывания, вроде пролетающего вертолета, взорвавшейся петарды, что сложнее, и всяких прочих помех). Если выстрел подтверждается, патруль выезжает на место. Все это занимает всего несколько минут. Кроме возможности быстрого реагирования на инцидент, система приносит и пользу иного рода [48].

Также данная система имеет мобильный клиент, интерфейс которого представлен на рисунке 1.12.

Данная система обладает следующими недостатками: высокая стоимость, закрытый исходный код системы, кампания, владеющая системой, работает только на территории США.



Рисунок 1.11 – Размещение микрофонов для системы ShotSpotter

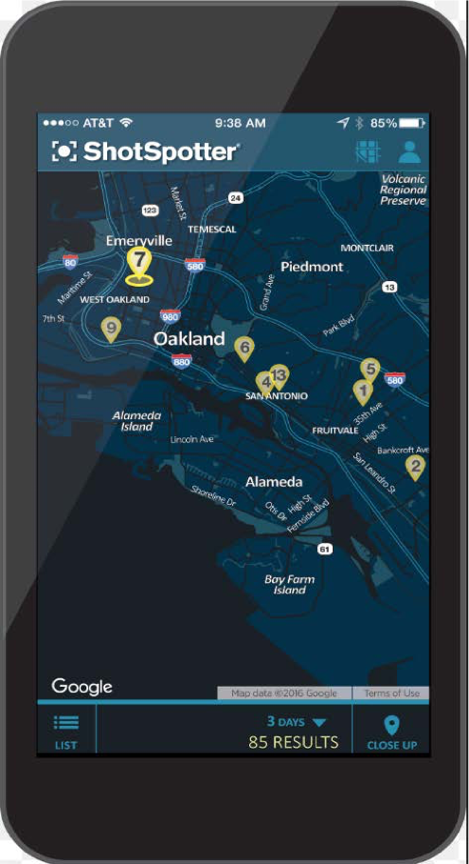


Рисунок 1.12 – Мобильный клиент системы ShotSpotter

* + 1. Shazam

Shazam представляет собой бесплатный кроссплатформенный проект, который позволяет пользователю определить, какая играет в данный момент композиция. На данный момент существует приложение для Mac OS, Windows 8/10, Android, iOS.

Приложение использует микрофон, устройства для записи фрагмента записи музыки, которая играет где-либо. В базе данных Shazam хранятся фингерпринты для каждой аудиозаписи, которые позволяют выполнять сравнения точно и быстро.

Чтобы осуществить фингерпринтинг аудиозаписи необходимо преобразовать файл в спектрограмму, пример которой показан на рисунке 1.13, где по оси ординат представляется частота, а на оси абсцисс представляется время, а плотность затенения представляет амплитуду [49].

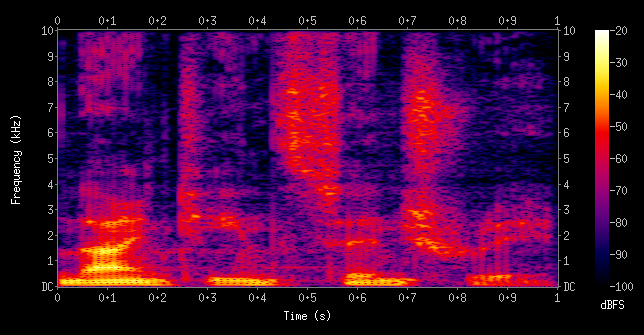


Рисунок 1.13 – Пример спектрограммы

Для каждой секции аудиофайла выбираются самые сильные пики, и спектрограмма сводится к диаграмме рассеяния. Пример которой представлен на рисунке 1.14. После этого происходит комбинаторное хеширование, на диаграмме рассеяния проставляются точки-якоря, которые потом помогут в поиске по базе данных. Данный подход позволяет точно определить проигрываемую композицию, даже несмотря на внешние шумы или низкое качество записи.

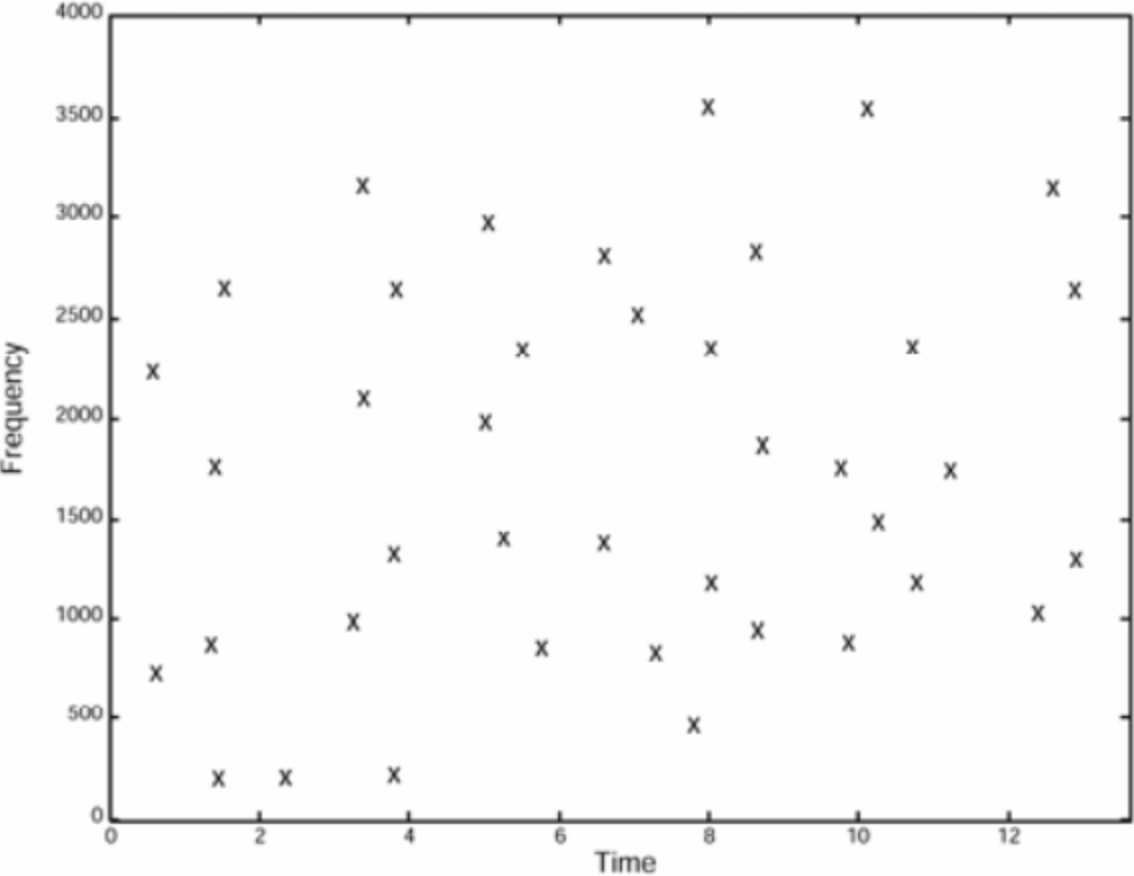


Рисунок 1.14 – Пример диаграммы рассеивания

На рисунке 1.15 представлено мобильное приложение Shazam.

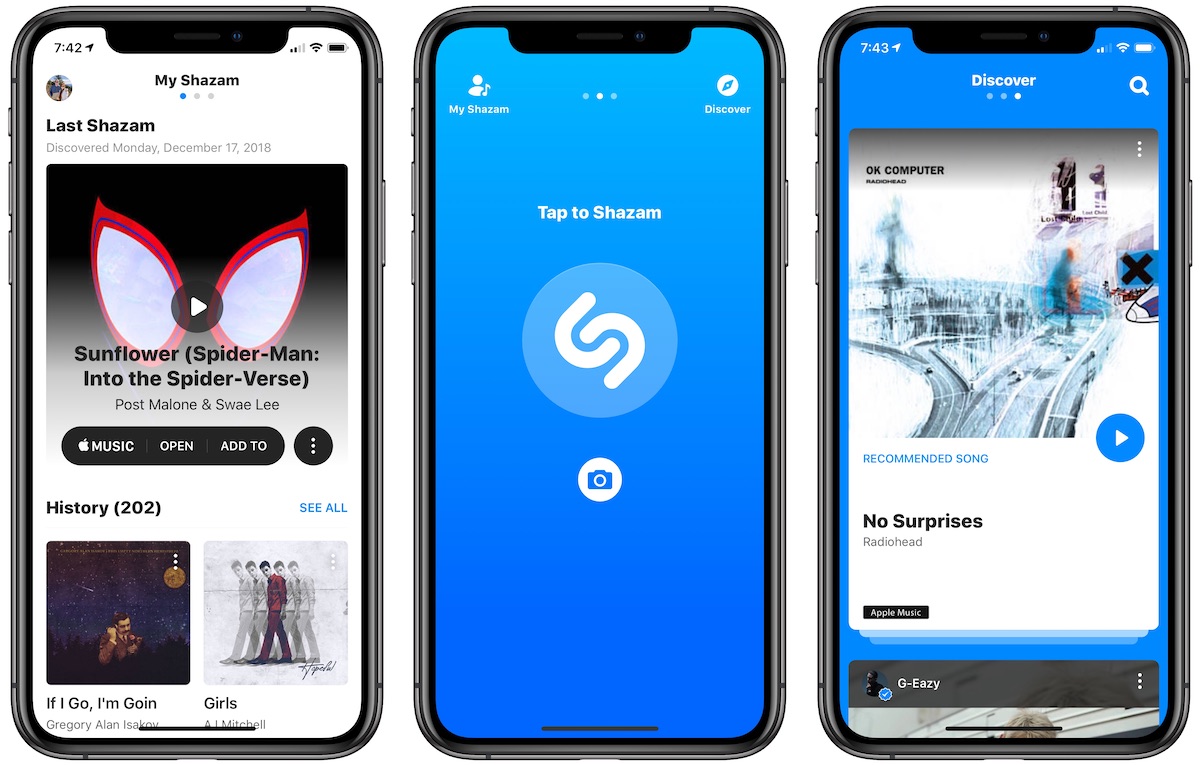


Рисунок 1.15 – Мобильный клиент Shazam

Данная система правильно определяет музыкальные произведения, однако её невозможно применить для классификации звуков окружающей среды, потому что они не настолько строго определены, в отличие от музыкальных композиций. Также в свободном доступе нет исходного кода алгоритмов, которые используются для создания фингерпринтов и дальнейшего сравнения с аналогами в базе данных.

* + 1. AudioAnalytics

Проект «AudioAnalytics» [50], базирующийся в Великобритании, предоставляет сразу несколько решений для различных вариантов использования. Архитектура предлагаемых решений имеет следующий вид: программа CoreLogger, работающая на устройстве конечного пользователя, позволяет принимать и отображать/сохранять тревожные события. На рисунке 1.16 представлен пример устройства этой компании, который осуществляет детектирование паттернов в аудиосигналах.



Рисунок 1.16 – Устройcтво по обнаружению звука от Audio Analytics

Работает в связке с другой частью общей системы – Sound Packs – которая представляет не что иное как набор различных модулей аудиоаналитики. Основные возможности данных модулей представляют собой детектирование следующих аудио событий:

* агрессия (разговор на повышенных тонах, крик);
* сигнализация автомобилей;
* разбивающееся стекло;
* поиск ключевых слов («полиция», «помогите» и т. д.);
* выстрелы;
* крик/плач ребенка.

Дополнительно предоставляется часть системы под названием Core Trainer, которая на основании поданного ей на вход набора аудио сигналов, выделит наиболее отличающиеся (выделяющиеся) части и сформирует новый паттерн для SoundPacks.

* 1. Постановка задачи

В рамках данной ВКР магистра поставлена задача разработать автоматизированную систему детектирования признаков акустического излучения транспортных средств на аудиозаписях с использованием сверточных нейронных сетей. Необходимо, чтобы на основе полученной аудиозаписи пользователь мог получить нужные характеристики ТП.

В соответствии с поставленной целью необходимо решить следующие задачи:

* проанализировать существующие методы сбора акустического излучения транспортных средств;
* проанализировать существующие методы анализа аудиофайлов;
* выполнить обзор существующих подходов в детектировании признаков (паттернов) в аудиофайлах;
* определить возможность применения нейронных сетей к решению данной задачи;
* разработать автоматизированную систему детектирования признаков акустического излучения транспортных средств на аудиозаписях с использованием сверточных нейронных сетей.

Для разработки автоматизированной системы определим и систематизируем подзадачи, которые необходимо решить:

* пересылка аудиозаписи, содержащей звуки ТП;
* обработка полученной аудиозаписи;
* детектирование сигнатур в аудиофайле;
* подсчет количества ТС и его классификация.
  1. Описание методологии проектирования системы

UML (сокр. от англ. Unified Modeling Language – унифицированный язык моделирования) – язык графического описания для объектного моделирования в области разработки программного обеспечения. UML является языком широкого профиля, это открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML моделью. UML был создан для определения, визуализации, проектирования и документирования в основном программных систем [51].

В рамках языка UML все представления о модели сложной системы фиксируются в виде специальных графический конструкций – диаграмм. Процесс объектно-ориентированного анализа и проектирования неразрывно связан с процессом построения диаграмм. При этом совокупность построенных диаграмм является самодостаточной в том смысле, что в них содержится вся информация, которая необходима для реализации проекта сложной системы.

В процессе разработки UML-проекта осуществляется постепенный переход от анализа и проектирования к описанию реализации программы.

* 1. Выбор комплекса программных средств

Комплекс программных средств для разработки системы включает в себя среду проектирования, язык программирования, интегрированную среду разработки, набор тестовых и обучающих данных.

* + 1. Выбор среды проектирования

В качестве среды проектирования информационно-логической модели выбран инструмент drawio. Он является векторным графическим редактором диаграмм и блок-схем и поддерживает большое количество различных нотаций. Основное преимущество данного инструмента является его открытый исходный код, что делает инструмент бесплатным и надежным.

* + 1. Выбор языка программирования

Для обработки данных, обучения и структуризации системы был выбран язык программирования Python. Python представляет из себя скриптовый язык, который предназначен для решения огромного спектра задач. Основным преимуществом данного языка является то, что он включает в себя большое количество готовых библиотек и модулей, которые являются аналогами систем автоматизированного проектирования. Язык является интерпретируемым, а это значит, что он не компилируется [52]. До запуска исходный код представляет из себя обычный текстовый файл. Программировать можно практически на всех платформах, язык хорошо спроектирован, логичен и лаконичен.

Для построения графического интерфейса в браузере использовался язык JavaScript. Это мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией языка ECMAScript. JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам. На сегодняшний день с развитием языка и инструментов, построенных вокруг JavaScript, даже можно разрабатывать нативные приложения для мобильных устройств.

* + 1. Выбор среды программирования

В качестве среды программирования был выбран редактор исходного кода Visual Studio Code. Данный редактор является разработкой компании Microsoft, однако распространяется под лицензией MIT [53], а его исходный код опубликован на GitHub, что делает данный продукт полностью бесплатным для конечного пользователя. Visual Studio Code основан на Electron [54] – [фреймворк](https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA), позволяющий с использованием [Node.js](https://ru.wikipedia.org/wiki/Node.js) разрабатывать настольные приложения, которые работают на движке Blink. Несмотря на то, что редактор основан на Electron, он не использует редакто Atom. Вместо него реализуется веб-редактор Monaco, разработанный для Visual Studio Online.

Данный редактор исходного кода, несмотря на свою легковесность, обладает огромными возможностями для рефакторинга, IntelliSense, навигацию по коду, подсветку синтаксиса, поддержку системы контроля версий, отладку. Visual Studio Code поддерживает большую часть существующих языков программирования, а также вспомогательных инструментов, благодаря системе плагинов. Имеется возможность кастомизации редактора, часть функций доступна через графический интерфейс, однако зачастую используется палитра команд или редактирование конфигурационных JSON файлов.

* + 1. Выбор системы управления базами данных

Выбор базы данных (БД) является важным этапом в проектировании системы, потому что БД предопределяет множество нюансов на процессе разработки, а также дальнейшем

В качестве системы управления базами данных (СУБД) была выбрана MongoDB. MongoDB – документоориентированная (СУБД) с открытым исходным кодом, не требующая описания схемы таблиц. Классифицирована как NoSQL, использует JSON-подобные документы и схему базы данных. Написана на языке C++. В основе данной СУБД лежит концепция коллекций и документов [55].

Коллекция – это группа документов MongoDB. Является эквивалентом простой таблицы в реляционной базе данных. Коллекция помещена внутри одной БД. Документ в коллекции моет иметь различные поля. Чаще всего, все документы в коллекции созданы для одной, либо относящихся друг ко другу целей.

Документ – это набор пар «ключ – значение». Документ имеет динамическую схему. Это означает, что документ в одной и той же коллекции не обязан иметь один одинаковый набор полей или структуру, а общие поля в коллекции могут иметь различные типы данных.

Данная СУБД упрощает разработку, потому что нет строгой схемы, которая хранится в БД, а также MongoDB позволяет проще оперировать с байтовыми данными.

.

1. Проектирование и программная реализация автоматизированной системы
   1. Разработка метода детектирования признаков (паттернов) в аудиофайлах

Для того, чтобы эффективно и точно распознавать необходимые аудиособытия, был предложен следующий метод, состоящий из нескольких шагов, схема которого представлена на рисунке 2.1.



Рисунок 2.1 – Алгоритм детектирования признаков

На первом шаге происходит преобразование исходного аудиосигнала в набор фреймов с перекрытием. На рисунке 2.2 представлен пример буферизации с перекрытием.

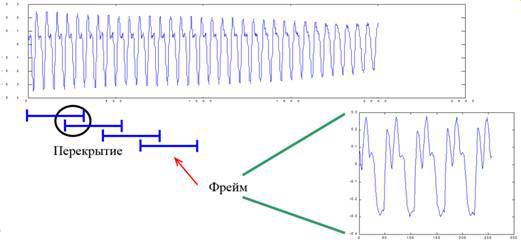


Рисунок 2.2 – Буферизация с перекрытием

Второй шаг представляет из себя предобработку. Обычно, эта стадия включает в себя фильтрацию и оконное взвешивание. Данный шаг необходим для спектрального сглаживания сигнала. В таком случае сигнал становится менее восприимчивым к различным шумам, возникающим в процессе обработки.

В разработанной системе в качестве предобработки производится передискретизация аудиозаписи, а также преобразование к одному к каналу. Также происходит нормализация разрядности сигнала, так что значения варьируются от -1 до 1. Это убирает сложность обработки данных, потому что различные аудиофайлы могут иметь различный диапазон разрядности.

Третий шаг – извлечение необходимых признаков. Существует множество подходов к извлечению признаков из аудиосигнала. Главная цель данных методов – уменьшение избыточности сигнала и выделение наиболее релевантной информации, и избавление от нерелевантной. Обычно, признаки, описывающие аудиосигнал для получения той или иной характеристики комбинируются в один вектор, на основе которого и происходит процесс обучения и дальнейшая классификация с использованием выбранной обученной модели. Далее, на рисунке 2.3 представлены наиболее популярные признаки, выделяемые из аудиосигнала:

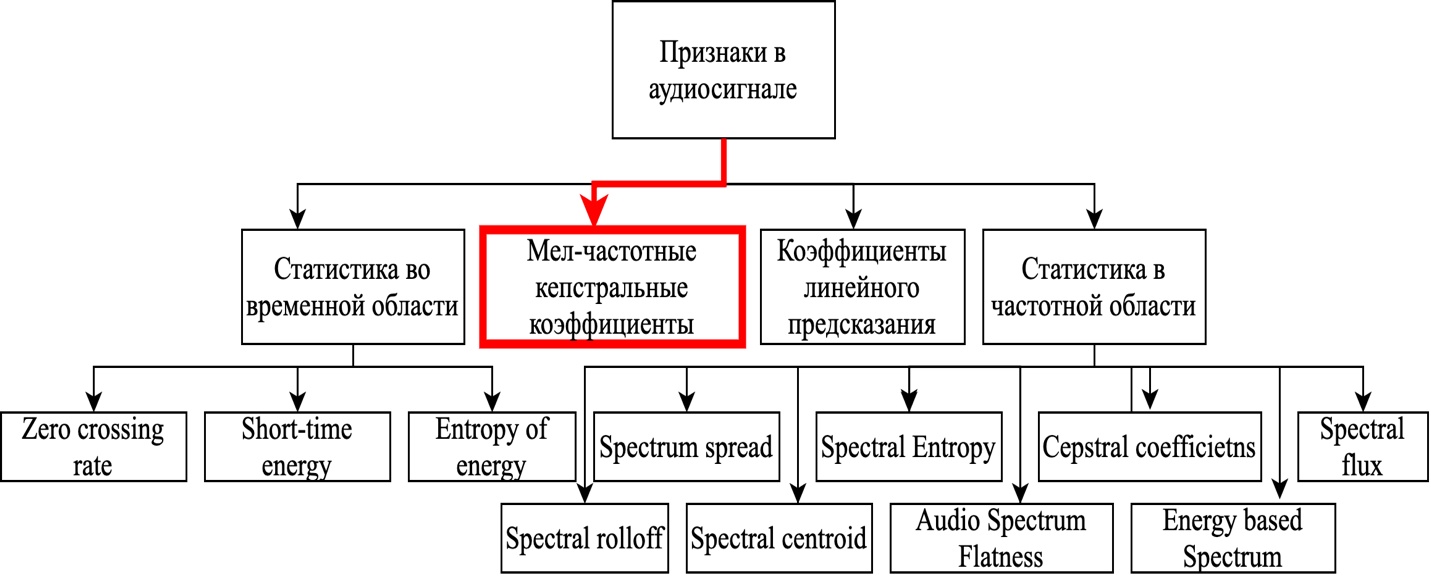


Рисунок 2.3 – Классификация методов получения   
признаков в аудиосигнале

В качестве инструмента извлечения признаков предложено использовать извлечение МЧКК из аудиофайлов. МЧКК суммируют распределение частот по размеру окна, поэтому можно анализировать как частотные, так и временные характеристики звука. Эти аудио представления позволят нам определить признаки, необходимые для классификации.

На четвертом шаге происходит постобработка признаков. После извлечения необходимых признаков сигнала для их дальнейшего использования производится нормализация признаков так, чтобы каждый компонент вектора признаков имел среднее значение и стандартное отклонение 1.

Одним из преимуществ использования методов сокращения размерности является способность существенно увеличить скорость и точность процесса обучения, и точность работы алгоритмов машинного обучения, за счет избавления от избыточного количества признаков и выделения существенных признаков. В разработанной системе на данном шаге используется метод главных компонент, так как он позволяет сократить размерность вектора признаков за счет выделения независимых компонент, максимальным образом покрывающий разброс по всем событиям.

На последнем пятом шаге предложенного метода происходит выбор обучающей модели. Для различных типов аудиособытий стоит подбирать конкретный классификатор, потому что это может обеспечить большой прирост точности в процессе классификации.

На рисунке 2.4 представлена схема классификаторов, применяемых для распознавания аудиособытий.

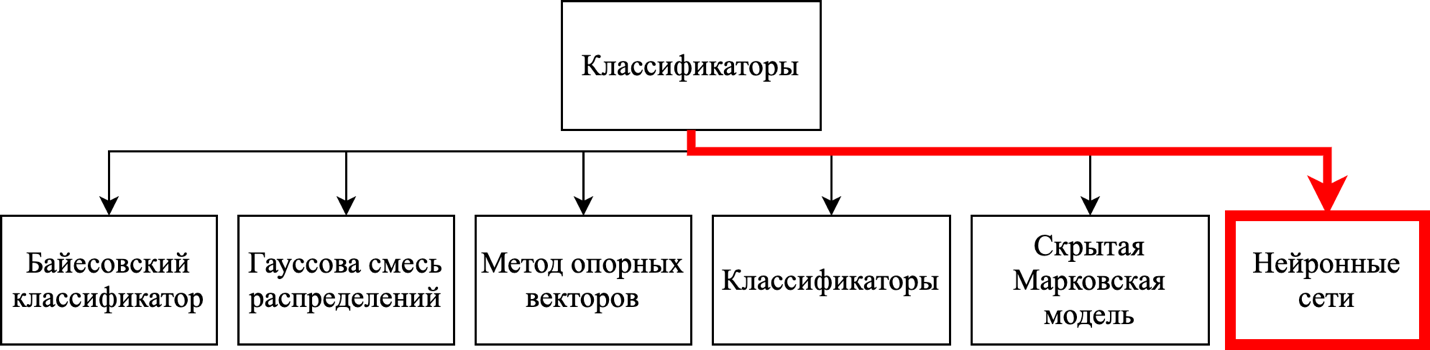


Рисунок 2.4 – Виды классификаторов

В качестве классификатора используются СНС. Слои сети организованы в трех измерениях: ширина, высота и глубина. Узлы в одного слоя не обязательно соединяются со всеми узлами следующего слоя, а зачастую лишь с какой-то его частью. Оптимизатором модели выступает Adam (adaptive moment estimation) – оптимизационный алгоритм, сочетающий в себе и идею накопления движения и идею более слабого обновления весов для типичных признаков [58]. Сверточные слои разработаны для обнаружения признаков. Алгоритм работы заключается в скольжении окна фильтра по входным данным, выполнении умножения матриц и сохранении результата в карте объектов. Данная операция называется операцией свертки. В качестве функции активации для сверточных слоев используется функция ReLU.

Каждый сверточный слой связан с субдискретизирующим слоем. Субдикретизирующий слой уменьшает размерность модели, а также сокращает количество параметров и последовательных вычислений. Данная оптимизация сокращает количество времени на обучение, а также сокращает переобучение.

* 1. Архитектура системы

Основой архитектуры системы являются микросервисы. Микросервисы – это новый архитектурный стиль, который пытается преодолеть недостатки централизованной монолитной архитектуры [56], в которой логика приложения заключена в большие развертываемые части файлы. Сами по себе микросервисы это подход к разработке отдельного приложения в виде набора небольших сервисов, каждый из которых работает в своем собственном процессе и обменивается данными с легковесными механизмами, часто под собой подразумевающий RESTful API [57]. В отличие от монолитов, микросервисы способствуют независимому развертыванию и масштабируемости и могут быть разработаны с использованием различных стеков технологий [58].

Хотя микросервисы можно рассматривать как эволюцию сервис-ориентированной архитектуры, однако они являются архитектурно различными. В отношении совместного использования и повторного использования: с учетом того, что повторное использование сервиса часто оказывалось не столь ожидаемого, вместо повторного использования уже существующего микросервиса для новых задач или вариантов использования, ведь они должны быть небольшими и достаточно независимыми, чтобы можно было быстро разработать новый, который может сосуществовать, развивать или заменять предыдущий в соответствии с потребностями задачи.

Схема архитектуры системы представлена на рисунке 2.5.

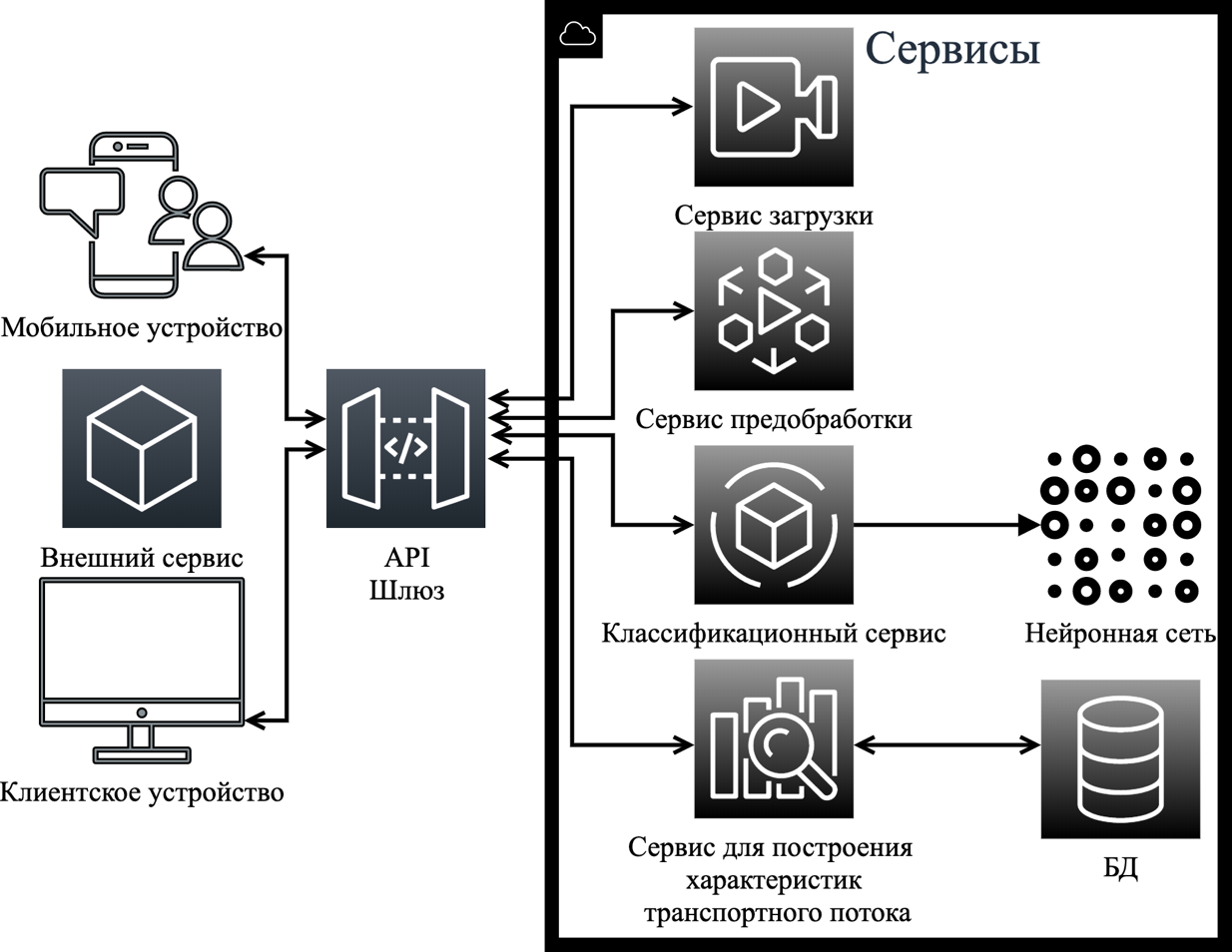


Рисунок 2.5 – Архитектура системы

Таким образом, микросервисная архитектура обеспечивает следующие преимущества:

* независимая разработка: маленькие независимые компоненты могут создаваться маленькими независимыми командами. Объём времени, необходимого для изучения компонента, значительно снижается, и разрабатывать новые функции становится проще.
* независимое развёртывание: каждый отдельный компонент можно разворачивать независимо. это позволяет выпускать новый функционал быстро и с меньшими рисками.
* независимая масштабируемость: каждый компонент можно масштабировать независимо от другого. Это позволяет обеспечить необходимую устойчивость системы меньшими затратами.
* возможность повторного использования: компоненты реализуют свою маленькую конкретную функцию. Это означает, что их проще адаптировать для использования в других системах, сервисах или приложениях.
  1. Проектирование информационно-логической модели системы
     1. Диаграмма вариантов использования

Диаграмма вариантов использования описывает функциональное назначение системы, то есть то, что система будет делать в процессе своего функционирования. Диаграмма вариантов использования является исходным концептуальным представлением или концептуальной моделью системы в процессе ее проектирования и разработки.

Разработка диаграммы вариантов использования преследует цели:

* определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;
* сформулировать общие требования к функциональному поведению проектируемой системы;
* разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
* подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества сущностей или актантом (actor), взаимодействие которых с системой отображается в виде так называемых вариантов использования. При этом актантом (actor) или действующим лицом называется любая сущность, взаимодействующая с системой извне. Это может быть человек, техническое устройство, программа или любая другая система, которая служит источником воздействия на моделируемую систему так, как определит сам разработчик. В свою очередь, вариант использования (use case) служит для описания сервисов, которые система предоставляет актеру или прецедентов использования системы. Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемый системой при диалоге с актантом.

Помимо актантов и вариантов использования, данная диаграмма содержит отношения – описывающие взаимодействия экземпляров одних актантов и вариантов использования с экземплярами других актантов, и вариантов использования. В языке UML имеется несколько стандартных видов отношений между актантами и вариантами использования, такие как: ассоциации, расширения и включения.

Диаграмма вариантов использования представлена на рисунке 2.6

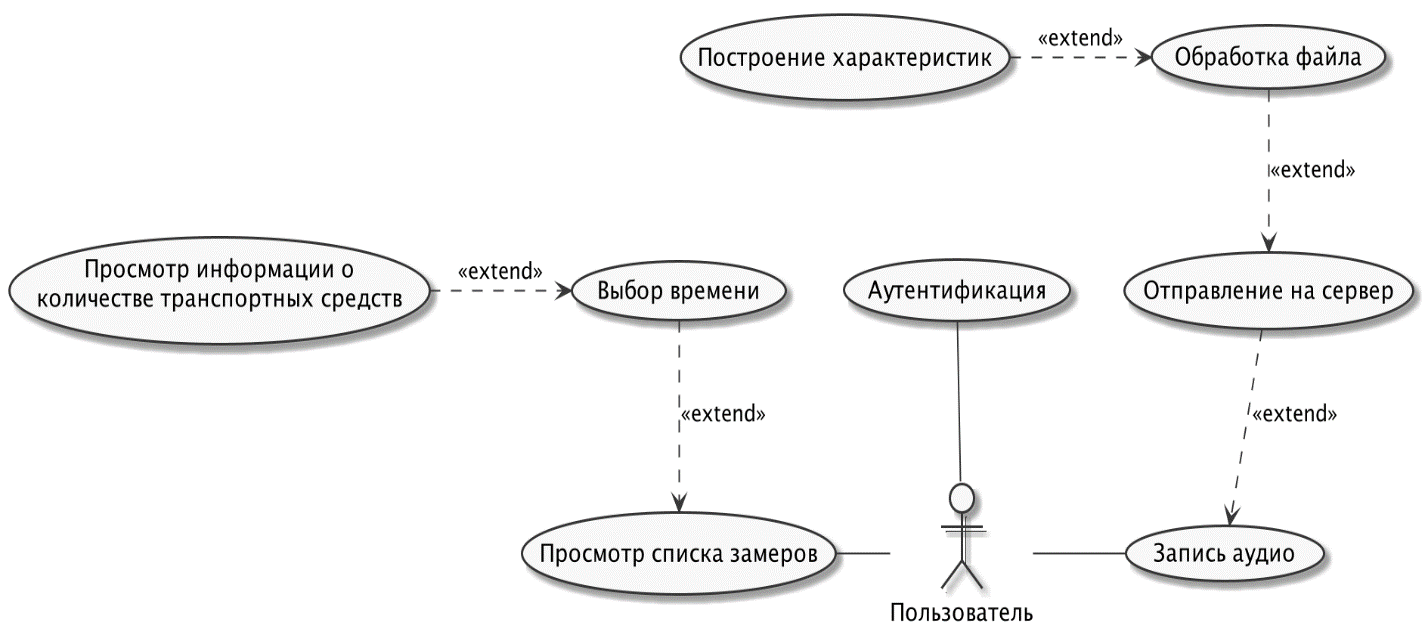


Рисунок 2.6 – Диаграмма вариантов использования

Пользователь имеет возможность загружать аудиозапись, чтобы далее она была обработана на сервере, с последующим вычленением всех необходимых характеристик и сохранении информации в базе данных. Пользователю также доступен выбор точек на карте, где происходила запись аудиофайла. При просмотре данных точек доступна информация о перечне ТС, которые проезжали в указанный промежуток времени.

* + 1. Диаграмма классов

Базовыми элементами автоматизированной системы являются классы. Класс – это шаблон, который описывает множество однотипных объектов, имеющих определенный набор параметров и характеристик – атрибуты, операции, отношения и семантику.

В нотации UML классы системы отображаются на диаграмме классов. Диаграмма классов представляет собой набор классов, интерфейсов, их атрибутов и методов, а также взаимосвязей между этими классами [59]. Данная диаграмма предназначена для определения типов объектов разрабатываемой системы и для описания их отношений.

Диаграмма классов системы представлена на рисунке 2.7.

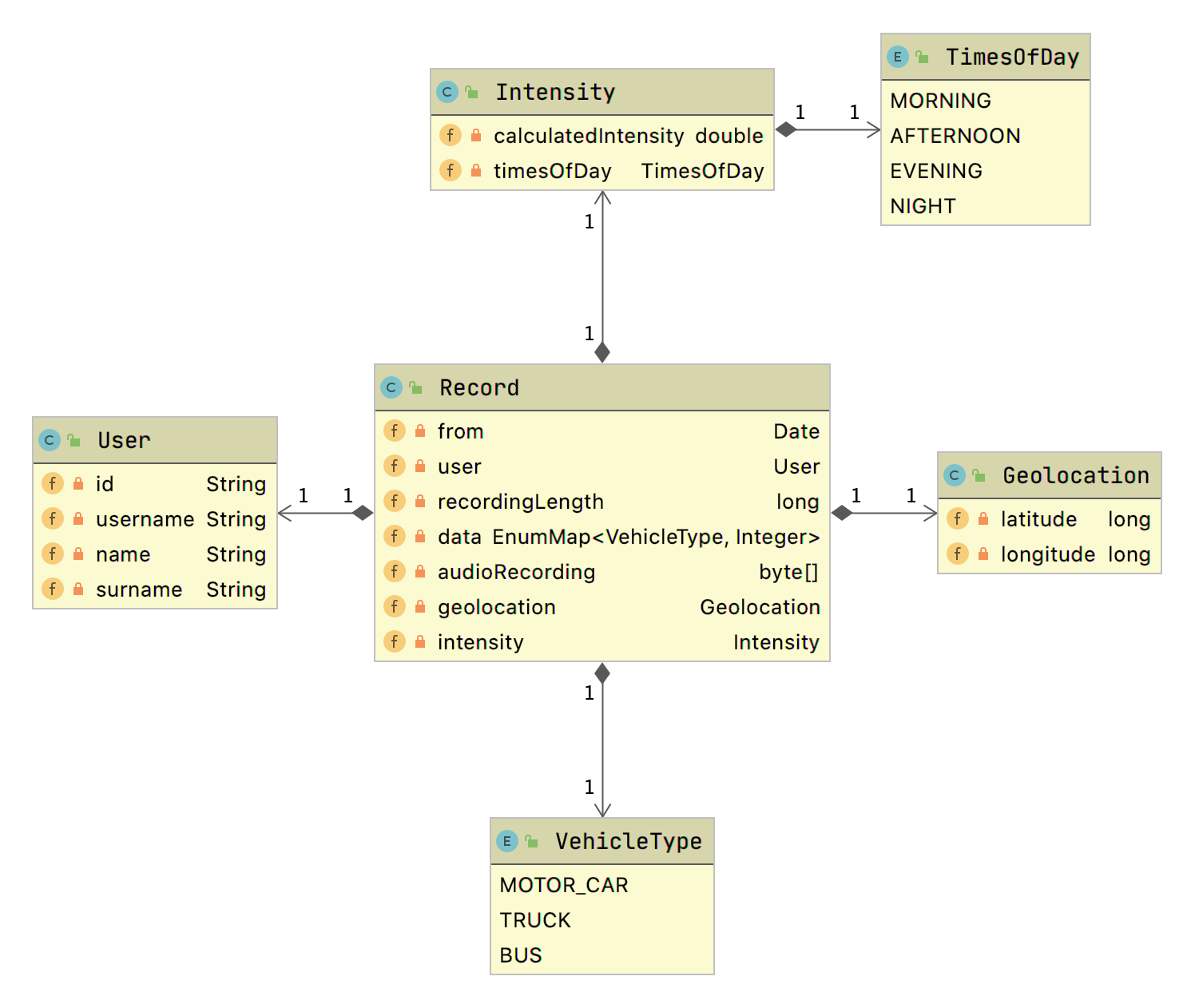


Рисунок 2.7 – Диаграмма классов

Наиболее важными среди связей диаграммы классов являются следующие:

* зависимость – отношение использования, при котором изменение объекта одного класса влияет на использующий его объект другого класса;
* обобщение, представляющее собой отношение между родительской сущностью и ее потомком;
* ассоциация, показывающая какую-либо связь между объектами различных классов;
* агрегация – это разновидность ассоциации, представляющая собой отношение типа «часть-целое»;
* композиция, являющаяся типом агрегации, при которой объект-часть принадлежит только единственному целому.
  + 1. Диаграмма состояний

Главное предназначение этой диаграммы – показать поведение одного объекта в течение его жизни, начиная от создания объекта и заканчивая его уничтожением. Каждая диаграмма состояний представляет некоторый автомат. С ее помощью представляется поведение сущностей во времени основываясь на восприятии ими каких-либо событий. Системы, которые реагируют на внешние действия от других систем или от пользователей, иногда называют реактивными. Если такие действия инициируются в произвольные случайные моменты времени, то говорят об асинхронном поведении модели.

Диаграмма состояний по существу является графом специального вида, который представляет некоторый автомат. Вершины этого графа являются состояния (или псевдосостояния). Дуги графа служат для обозначения переходов из состояния в состояние. Диаграммы состояний могут быть вложены друг в друга, образуя вложенные диаграммы более детального представления отдельных элементов модели. Диаграмма состояний разработанной системы представлена на рисунке 2.8.

Под состоянием понимается абстрактный метакласс, используемый для моделирования отдельной ситуации; при этом имеет место выполнение некоторого условия. Состояние может быть задано в виде набора конкретных значений атрибутов класса или объекта, которые отражают динамический или функциональный аспект его поведения. При этом изменение их отдельных значений будет отражать изменение состояния моделируемого класса или объекта.

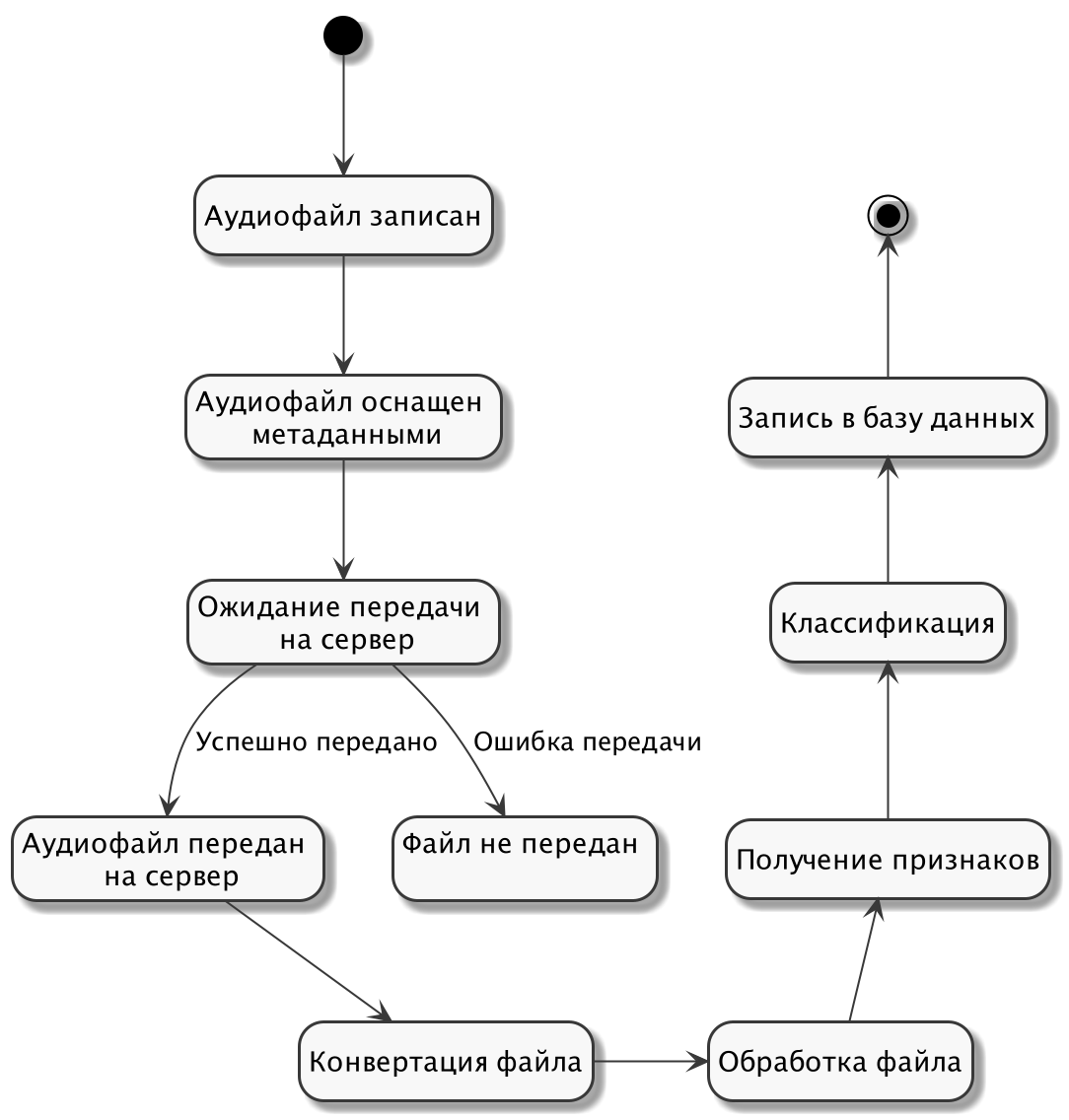


Рисунок 2.8 – Диаграмма состояний

* + 1. Диаграмма взаимодействий

Для моделирования взаимодействия объектов в языке UML используются советующие диаграммы взаимодействия. Так, взаимодействия объектов можно рассматривать во времени, и тогда для представления временных особенностей передачи и приема сообщений между объектами используется диаграмма последовательности.

На диаграмме последовательности изображаются объекты, которые непосредственно участвуют во взаимодействии и не показываются возможные статические ассоциации с другими объектами. Для диаграммы последовательности ключевым моментом является именно динамика взаимодействия объектов во времени. При этом диаграмма последовательности имеет как бы два измерения.

Одно – слева направо в виде вертикальных линий, каждая из которых изображает линию жизни отдельного объекта, участвующего во взаимодействии. Данный вид отображений называют swimlane или плавающая дорожка. Второе – вертикальная временная ось, направленная сверху вниз. При этом взаимодействие объектов реализуется посредством сообщений, которые посылаются от одних объектов к другим и также образуют порядок по времени своего возникновения. Удобно на диаграмме последовательности демонстрировать динамику сообщений и вызовов. При это видно: какой объект создается и в какой момент времени, к каким объектам адресуются вызовы и как они располагаются во времени [60]. На рисунке 2.9 по казан пример диаграммы последовательности.

* Для того, чтобы пользователь совершил анализ звуковой записи ТП, ему необходимо загрузить файл на сервер, где развернуто приложение. Если сервер работает в нормальном режиме и может принять аудиозапись, он посылает запрос пользователю и начинается загрузка файла. По истечении загрузки файла, происходит сохранение записи со всем необходимыми данными в базу данных, далее происходит предобработка и постобработка файла, получение необходимых признаков. После происходит классификация с использованием нейронной сети. Результаты классификации передаются в сервис подсчета интенсивности движения в соответствии с алгоритмами расчета интенсивности движения. Подсчитанная интенсивность сохраняется в базе данных, а полученная информация посылается пользователю. Пользователю представлена информация об интенсивности движения на том участке движения, где была произведена запись звука ТП.

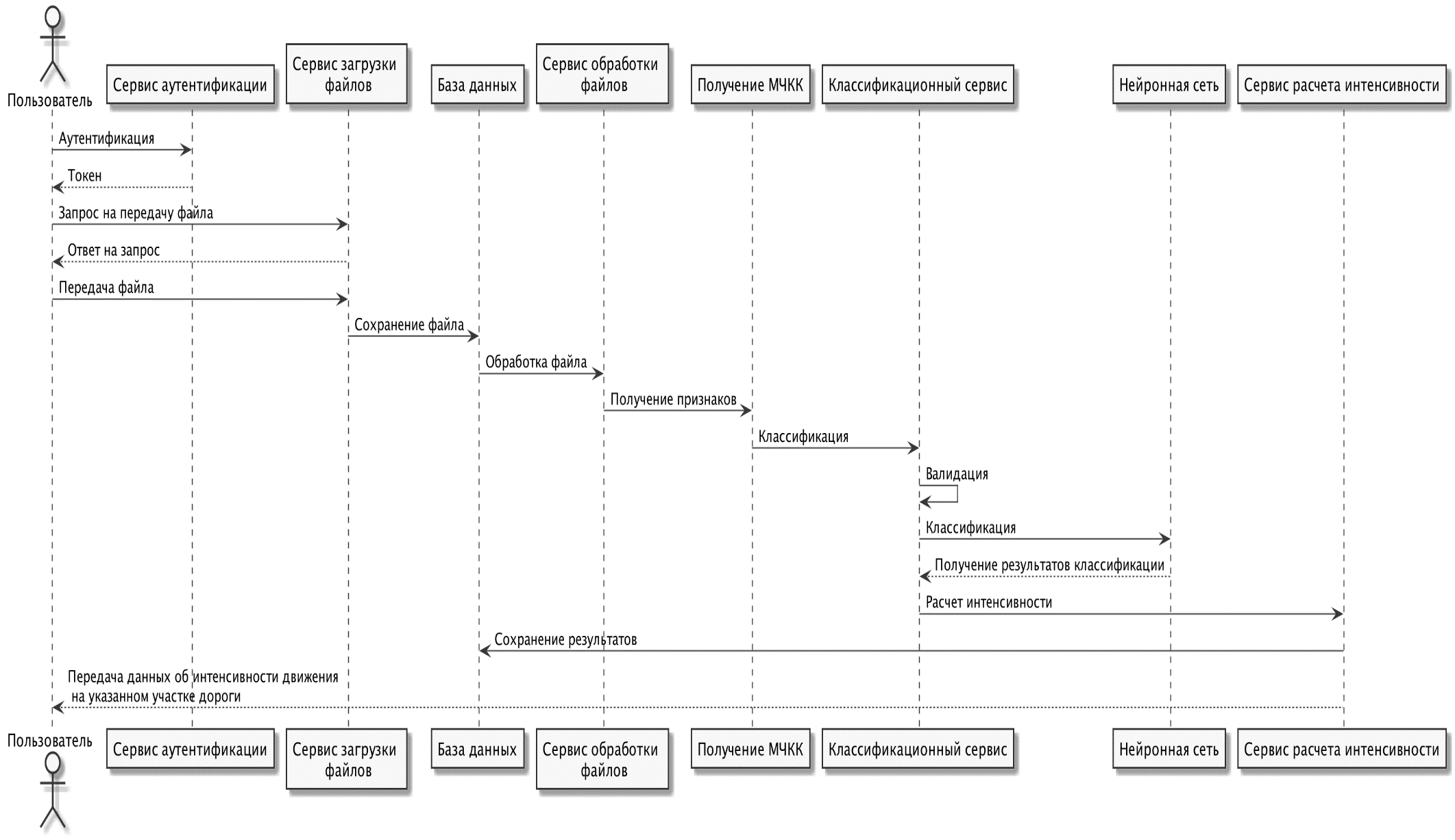


Рисунок 2.9 – Диаграмма последовательности

* + 1. Диаграмма деятельности

Диаграммы деятельности используются для моделирования процесса выполнения операций. Каждое состояние на диаграмме деятельности соответствует выполнению некоторой элементарной операции, а переход в следующее состояние срабатывает только при завершении этой операции в предыдущем состоянии. Состояние действия – специальный случай состояния с некоторым входным действием и, по крайней мере, одним выходящим из состояния переходом.

На рисунке 2.10 представлена диаграмма деятельности для разработанной системы.



Рисунок 2.10 – Диаграмма деятельности

Этот переход неявно предполагает, что входное действие уже завершилось. Состояние действия не может иметь внутренних переходов, поскольку оно является элементарными. Обычное использование состояния действия заключается в моделировании одного шага выполнения алгоритма (процедуры) или потока управления. Каждая диаграмма активностей должна иметь единственное начальное значение и единственное конечное состояния. При этом каждая деятельность начинается в начальном и заканчивается в конечном состоянии [61].

* + 1. Разработка логической модели данных

Логическая модель базы данных является моделью базы данных системы, представленной в виде совокупности сущностей, атрибутов и связей. Логическая модель позволяет графически представить структуру базы данных, которая не зависит от аппаратной платформы и конечной реализации базы данных.

В процессе проектирования системы была разработана логическая модель базы данных, для описания которой была выбрана методология IDEF1X. На рисунке 2.11 представлена логическая модель, содержащая основные сущности базы данных системы.

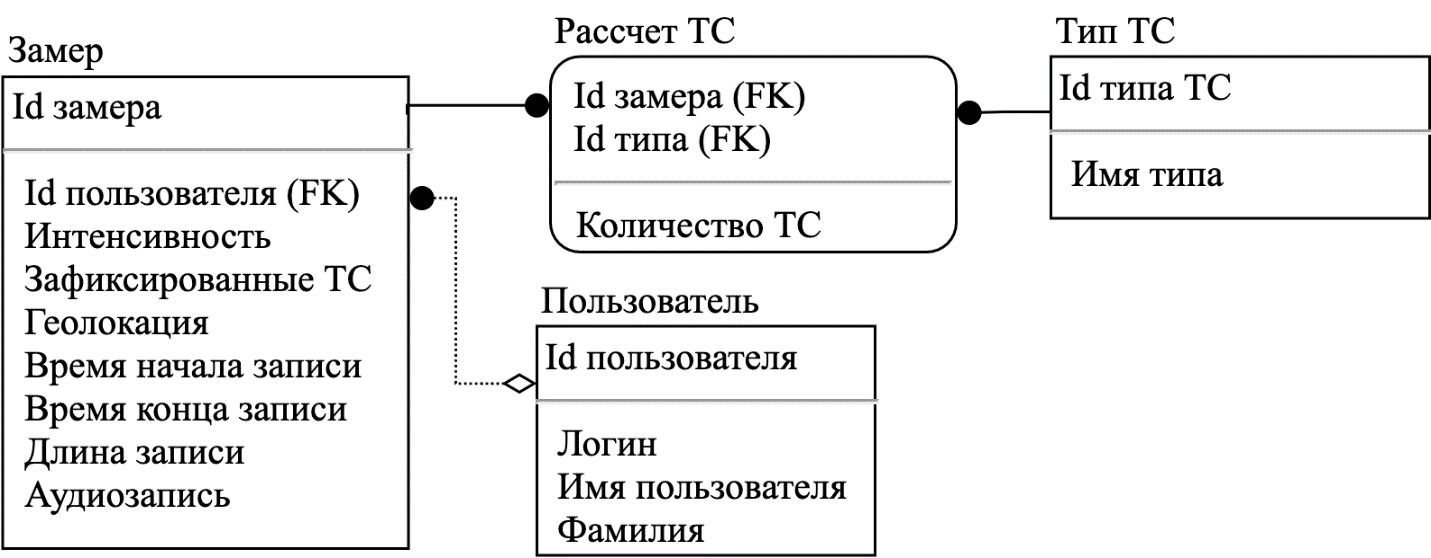


Рисунок 2.11 – Основные сущности логической модели данных

* + 1. Разработка физической модели данных

Под физической моделью базы данных понимают модель, определяющую способы размещения данных в среде хранения и способы доступа к этим данным. Физическая модель позволяет описывать все детали, которые необходимы для создания базы данных в конкретной СУБД: названия таблиц и столбцов, определения первичных и внешних ключей типы данных, и т.д.

Физическая модель базы данных, представленная на рисунке 2.12, построена на основе логической модели из п. 2.3.6.

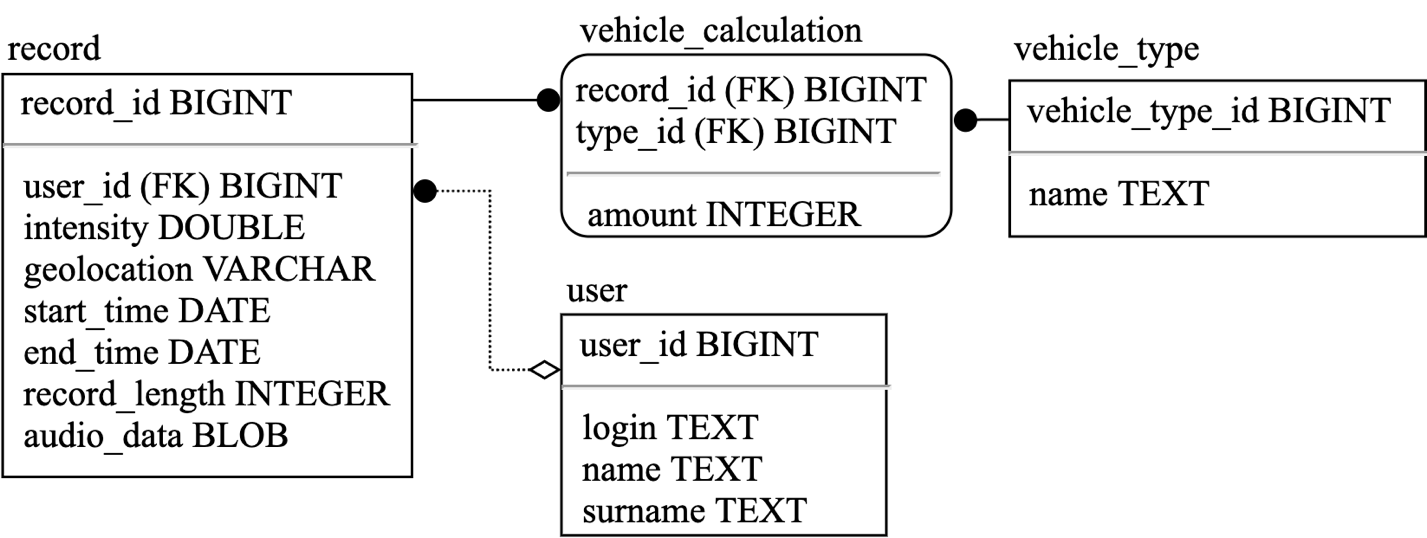


Рисунок 2.12 – Физическая модель данных

* 1. Выбор комплекса технических средств
     1. Расчет объема оперативной памяти

Объем оперативной памяти, необходимый для нормальной работы системы, рассчитывается в соответствии со следующей формулой:

,(19)

где – объем оперативной памяти, занимаемый операционной системой;

– объем оперативной памяти, необходимый для нормальной работы системы, определяемый объемом памяти запущенной системы, равный 40Мб;

– объем данных БД, который может быть одновременно загружен в оперативную память.

Расчет объема оперативной памяти проведен, исходя из предположения, что в качестве операционной системы сервера (ОС) используются наиболее распространенные в настоящее время ОС Windows 8 и Windows 10, таким образом: = 750 Мб. Общий объем ОЗУ составляет = 750 + 25 + 100 = 875 Мб.

* + 1. Расчет объема дискового пространства

Для расчета объема дискового пространства, необходимого для нормальной работы системы, использована следующая формула:

,(20)

где – объем оперативной памяти, необходимый для нормальной работы операционной системы;

– объем оперативной памяти, необходимый для нормальной работы системы (включает объем памяти для работы с базой данных);

–­ объем памяти, занимаемый всем необходимым сопутствующим программным обеспечением;

– объем памяти, занимаемый базой данных (всеми таблицами) при ее максимальном заполнении;

– объём памяти необходимый для хранения файла справки.

Расчет необходимого дискового пространства проведен, исходя из того, что в качестве ОС используются наиболее распространенные в настоящее время ОС Windows 8 и Windows 10, т.е. = 8.5 Гб.

Согласно формуле, объем памяти, необходимый для хранения программ определяется объемом памяти, который занимает запущенное приложение в памяти, а также объём необходимый для хранений файла справки. Для хранения системы необходимо 25 Мб. Для хранения БД необходимо 1024 Мб.

Таким образом, получаем, что = 8500 + 25 + 0 + 1024 + 0 = 9549 Мб.

* + 1. Рекомендованные характеристики технических средств

Для работы серверной части разработанной системы детектирования признаков акустического излучения ТС на аудиозаписях необходимы следующие технические средства:

* IBM-совместимый компьютер;
* не менее 875 Мбайт оперативной памяти;
* не менее 9549 Мбайт на жестком диске;
* поддерживаемая Windows 8 графическая карта;
* поддерживаемая Windows 8 мышь и клавиатура;
* операционная система Windows 8 и выше.

Для работы клиентской части разработанной системы необходимо одно из следующих технических средств:

* мобильное устройство под управлением операционной системы iOS не ниже 12 версии;
* мобильное устройство под управлением операционной системы Android не ниже 6 версии.
  1. Разработка алгоритмов функционирования системы

На рисунке 2.13 представлена схема алгоритма, соответствующего предложенному в 2.1 методу детектирования признаков (паттернов). Алгоритм учитывает особенности архитектуры системы, а именно использование мобильных устройств для сбора информации.

* 1. Разработка и описание интерфейса пользователя

Пользовательский интерфейс представляет собой совокупность правил и средств, которая регламентирует и обеспечивает взаимодействие пользователя с системой [62].

Перед началом работы пользователю нужно будет ввести логин и пароль в форме аутентификации. Экран формы входа в систему представлен на рисунке 2.14.

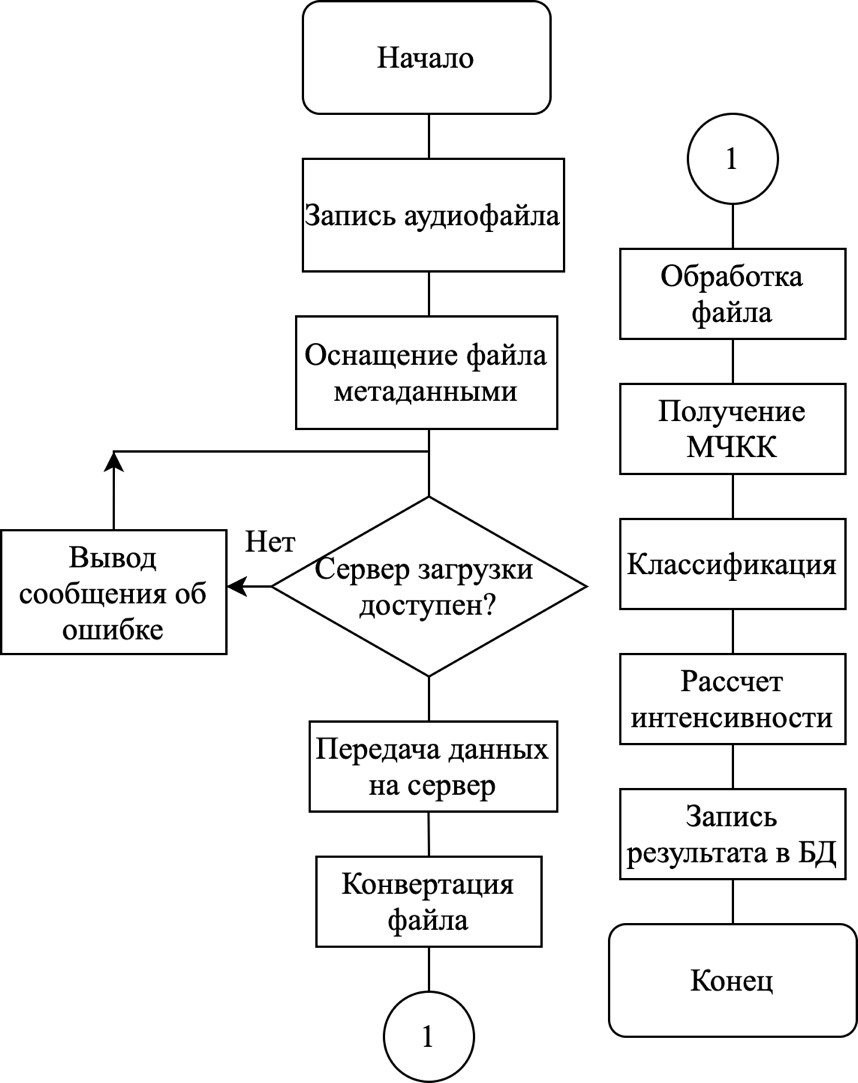


Рисунок 2.13 – Алгоритм классификации ТС на аудиозаписи

Если у пользователя нет учетной записи, то он может её создать, перейдя в форму создания учетной записи. Для создания учетной записи пользователю необходимо указать свое имя, логин и пароль. Экран создания учетной записи представлен на рисунке 2.15.

Первая форма после аутентификации – это карта, на которой расположены пункты, где был произведена запись и подсчитано количество ТС. Данная форма представлена на рисунке 2.16.

Чтобы сделать новую запись, необходимо нажать на кнопку «+», таким образом, пользователь перейдет в форму записи аудиофайла. Для того, чтобы отфильтровать точки замеров по каким-либо признаком, необходимо нажать на кнопку «Фильтры» и установить необходимую конфигурацию для последующего отображения.

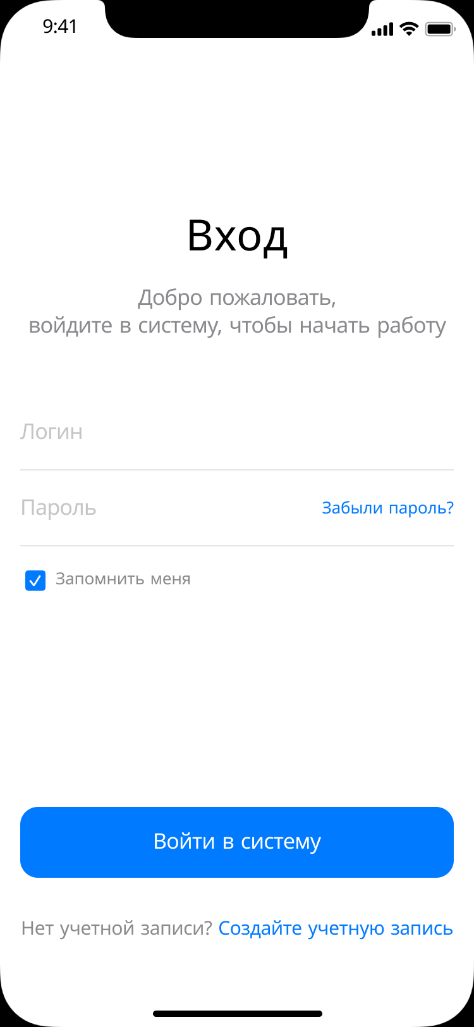


Рисунок 2.14 – Вход в систему

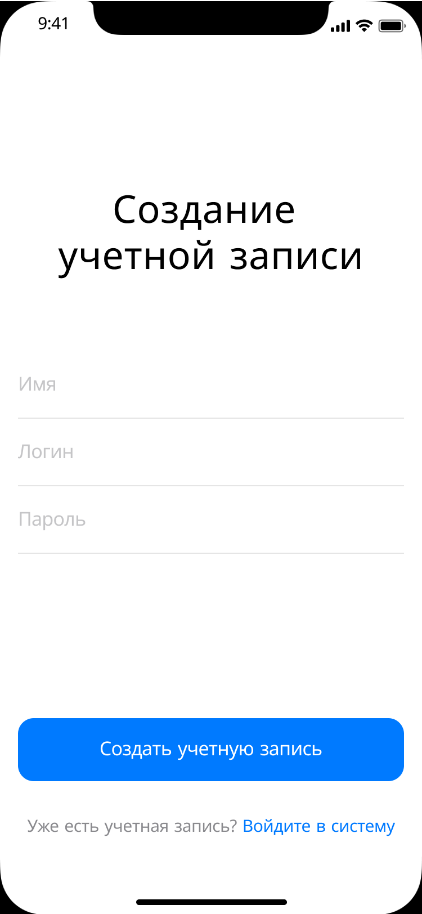


Рисунок 2.15 – Создание учетной записи

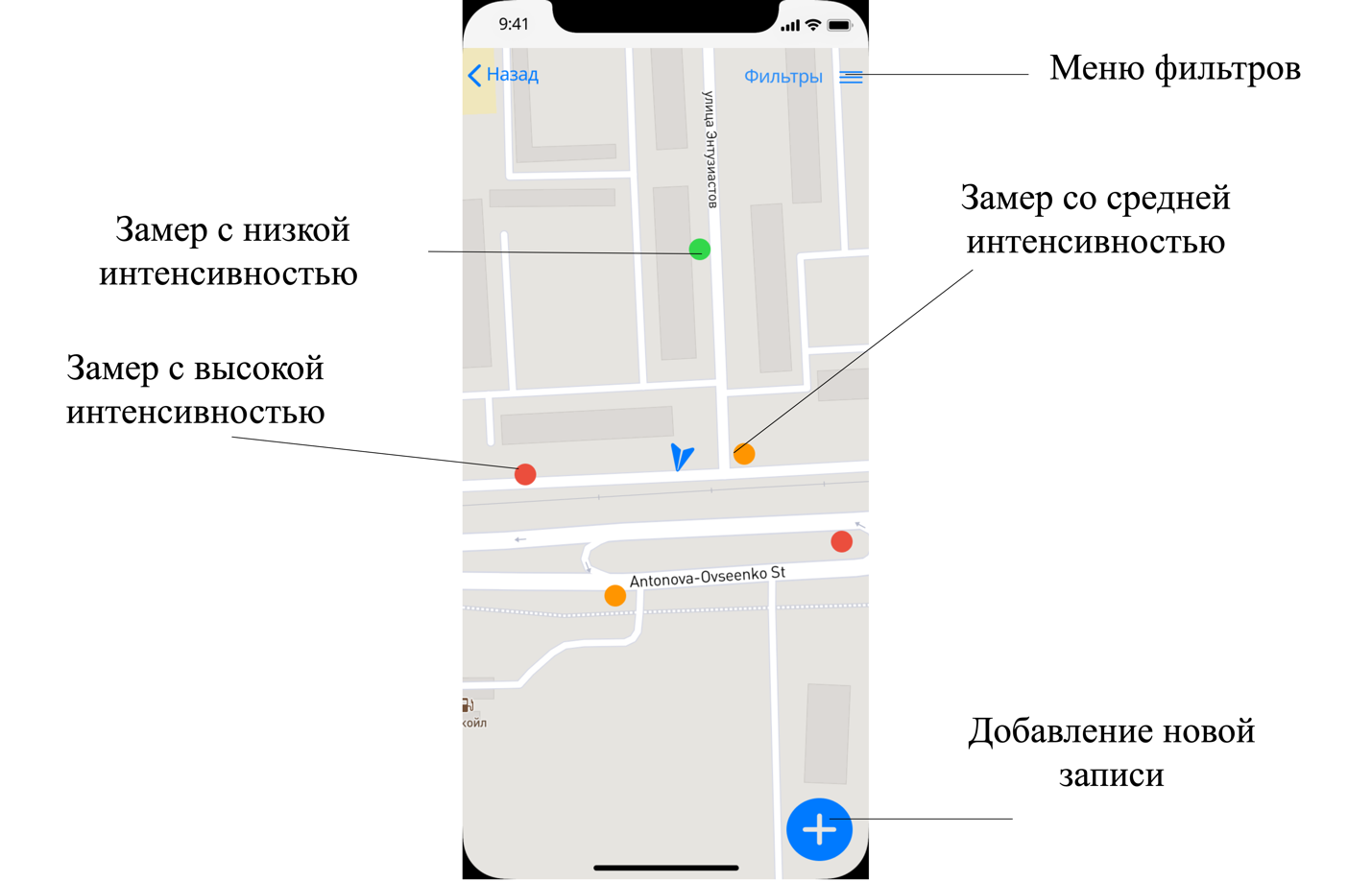


Рисунок 2.16 – Просмотр карты со списком замеров

На форме записи пользователю представлена возможность сделать запись. После того, как аудиозапись сделана, пользователю будет представлена спектрограмма, при нажатии на которую можно будет прослушать сделанную аудиозапись. При нажатии на кнопку «Отправить запись» происходит отправка сделанного аудиофайла на сервер. Также на сервер отправляются данные о местоположении, где была совершена запись, а также логин пользователя.

На рисунке 2.17 представлена форма, на которой можно сделать аудиозапись ТП, а также отправить данную запись на сервер.

При выборе замера на карте или на обзорном списке замеров, пользователю будет представлена форма обзора записи, пример которой представлен на рисунке 2.18. На данной форме содержится информация о дате, когда была совершена запись, точное время начала и конца записи, спектрограмма аудиосигнала и перечень ТС, который был определен. Также имеется возможность экспорта данной записи в PDF формат при выборе данной функции из контекстного меню.

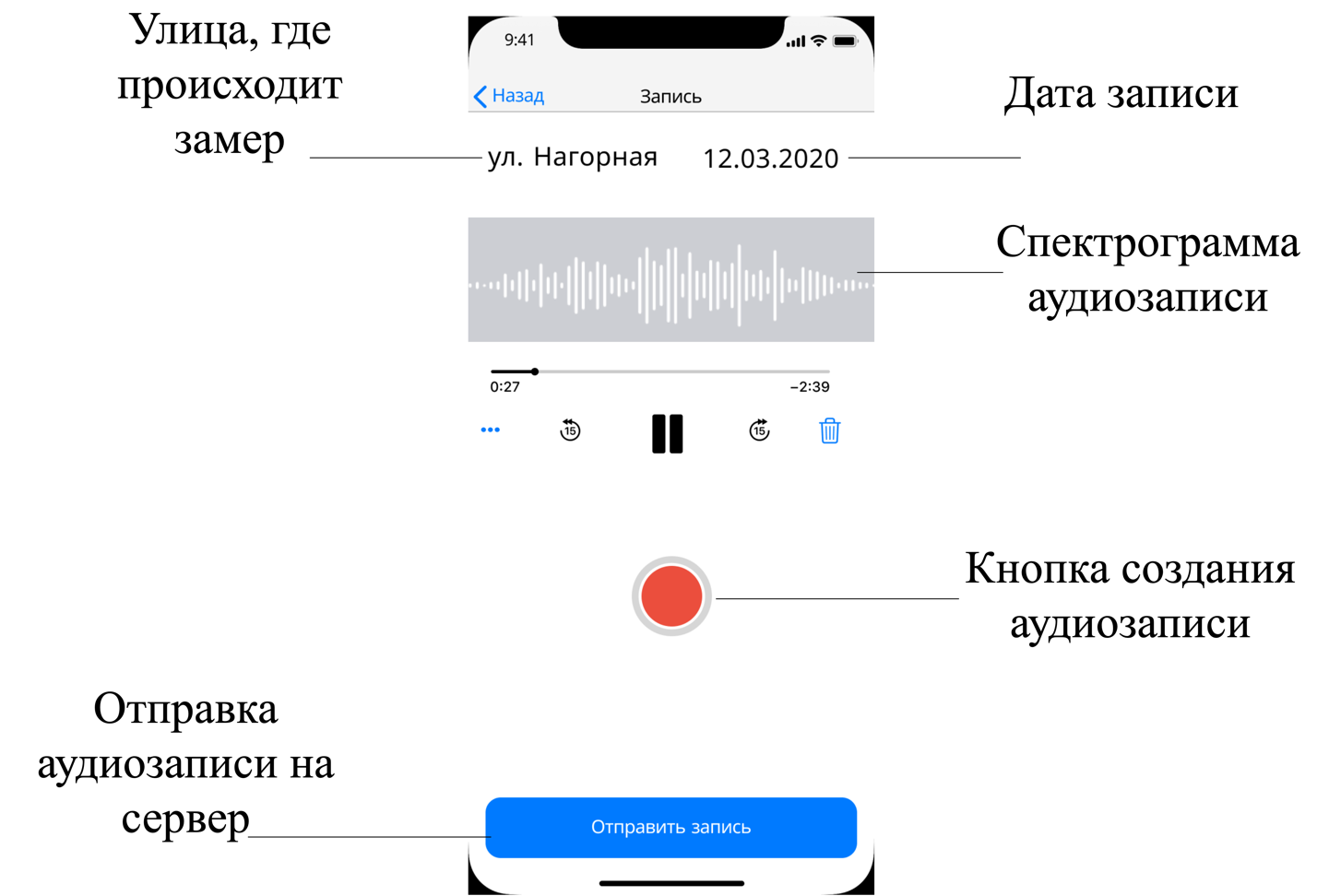


Рисунок 2.17 – Запись аудиофайла



Рисунок 2.18 – Обзор записи

На рисунке 2.19 представлен список замеров для более удобной навигации. Возможна фильтрация, для ограничения количества списков замеров.

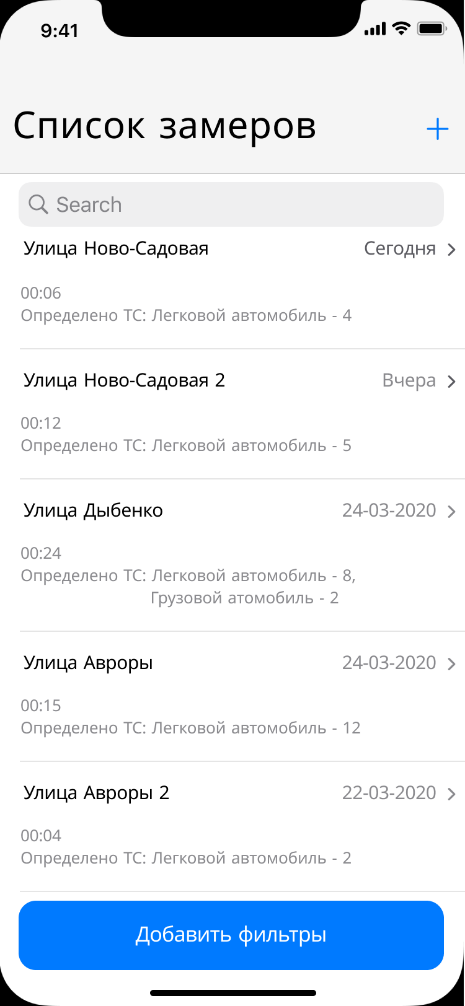


Рисунок 2.19 – Список замеров

Основными фильтрами являются: дата записи, время, в которое это запись была совершена, удаленность замера от текущей геолокации пользователя. Пример данной формы представлен на рисунке 2.20.

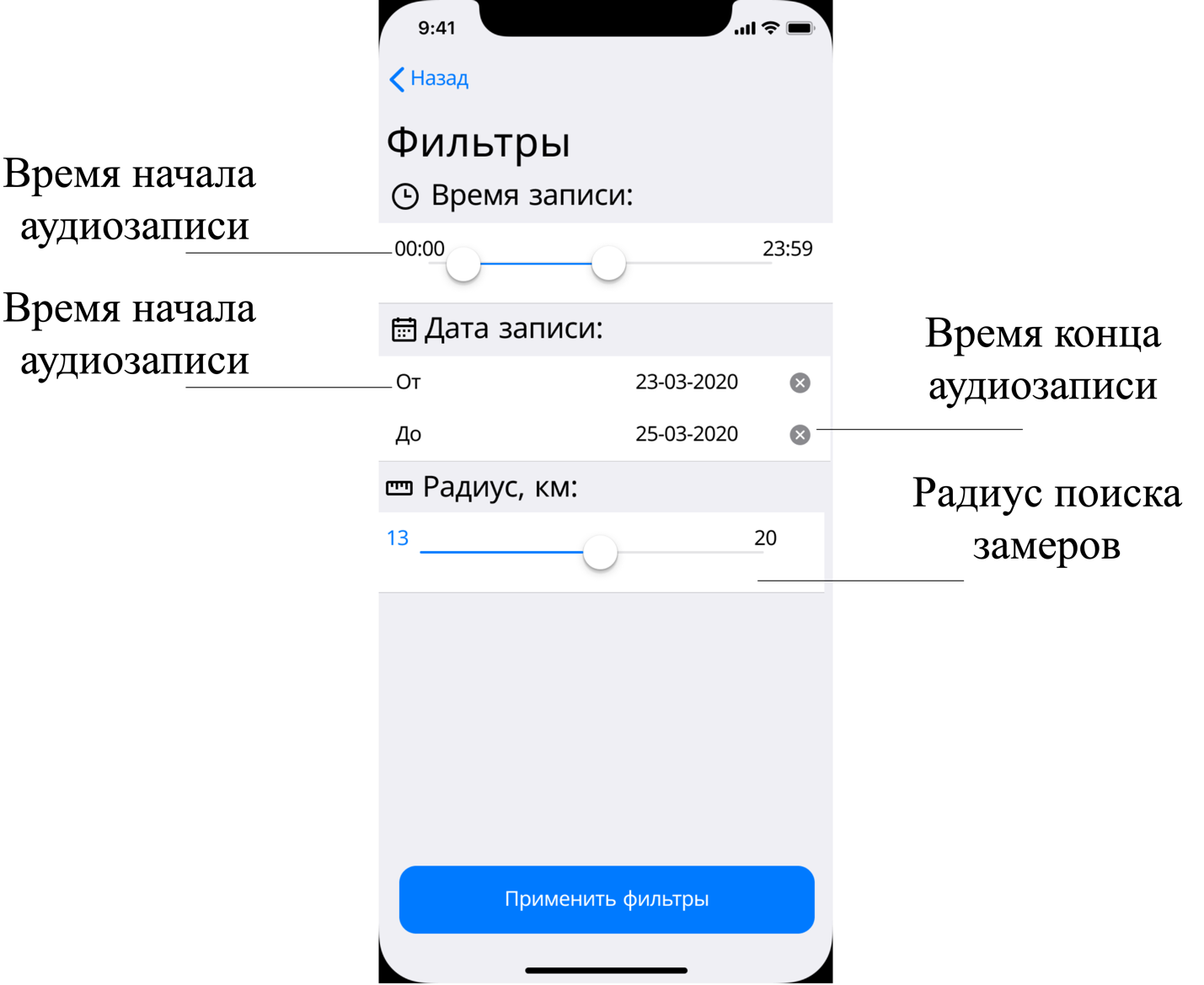


Рисунок 2.20 – Форма добавления фильтров

1. Экспериментальные исследования автоматизированной системы
   1. Исследование и визуализация данных
      1. Набор данных

Для разработки данной системы используется набор данных Urbansound8K [63]. Набор данных UrbanSound8K состоит из 8732 коротких (менее 4х секунд) фрагментов звуков города, которые разделены в 10 классов, которые состоят из:

* звук сигнала автомобиля;
* звук играющих детей;
* собачий лай;
* звук сверления;
* звук холостого хода двигателя;
* звук выстрела;
* звук пневматического бурильного молотка;
* звук сирены;
* звуки проезжающей машины;
* уличные шумы/ уличная музыка.

Обучающая и тестируемая выборки состоят из .wav файлов и описывающими их метаданными, хранящиеся в .csv таблице. Распределение классов представлено в таблице 3.1.

Как можно видеть, метки классов не сбалансированы. Хотя 7 из 10 классов имеют ровно 1000 образцов, звук сирены имеет примерно такое же значение (929), оставшиеся две метки (звук сигнала автомобиля и звук выстрела) имеют 43% и 37% относительно количества других образцов соответственно. Звуковые волны оцифровываются путем их дискретизации с дискретными интервалами, более известными как частота дискретизации (как правило, 44,1 кГц для звука с качеством для CD диска означает, что выборки берутся 44 100 раз в секунду).

Таблица 3.1 – Распределение классов звуков в наборе данных

|  |  |
| --- | --- |
| Имя класса | Количество записей в наборе |
| Звук играющих детей | 1000 |
| Собачий лай | 1000 |
| Уличные шумы/ уличная музыка | 1000 |
| Звук отбойного молотка | 1000 |
| Звук сверления | 1000 |
| Звук холостого хода двигателя | 1000 |
| Звук проезжающего автомобиля | 1000 |
| Звук сирены | 929 |
| Звук сигнала автомобиля | 429 |
| Звук выстрела | 374 |

Каждая выборка представляет собой амплитуду волны в конкретном интервале времени, где глубина в битах определяет, насколько детализированная выборка будет также известна как динамический диапазон сигнала (обычно 16 бит, что означает, что выборка может варьироваться от 65 536 значений амплитуды).

Пример дискретизации 16 битного сигнала представлен на рисунке 3.1.

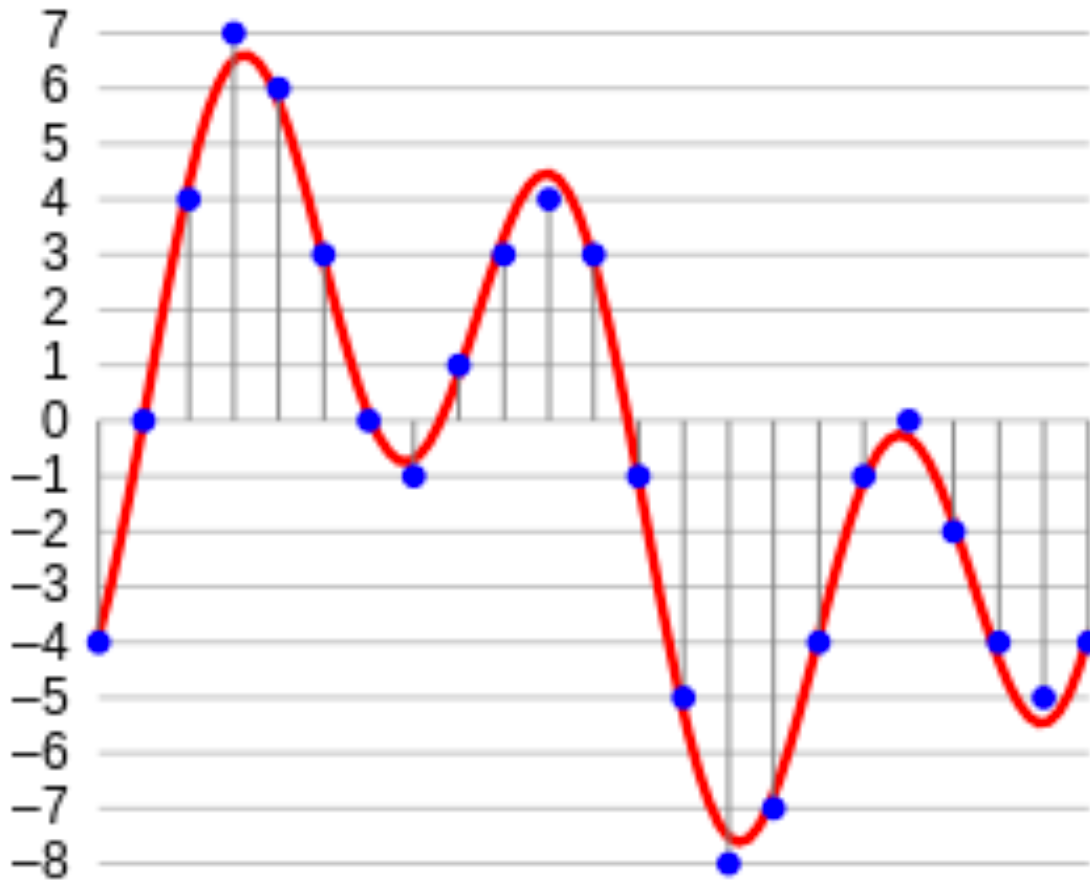


Рисунок 3.1 – Пример дискретизации сигнала

Поэтому данные, которые мы будем анализировать для каждого звукового фрагмента, по сути, представляют собой одномерный массив или вектор значений амплитуды.

* + 1. Преобразование данных

Сперва, происходит буферизация с перекрытием в исходном файле. Далее следует предобработка сигнала. Чтобы преобразовать данные в представления спектрограммы, была использована библиотека LibROSA, который представляет собой пакет с открытым исходным кодом, реализованный на Python. Данная библиотека предоставляет возможности для анализа звуковых и музыкальных данных, а также для визуализации спектрограммы.

На рисунке 3.2 представлен пример формы звукового сигнала, проезжающего мимо одного легкового автомобиля, а на рисунке 3.3 представлена спектрограмма выстрела из пистолета.

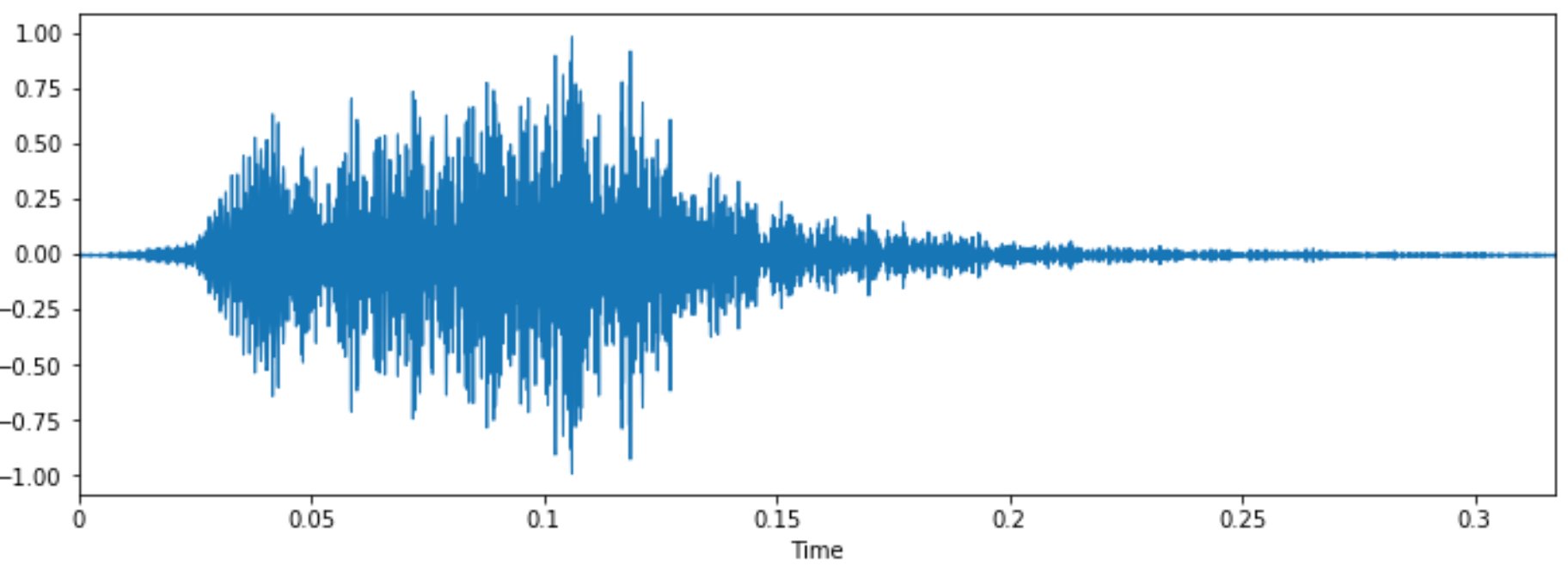


Рисунок 3.2 – Спектрограмма сигнала проезжающего легкового автомобиля

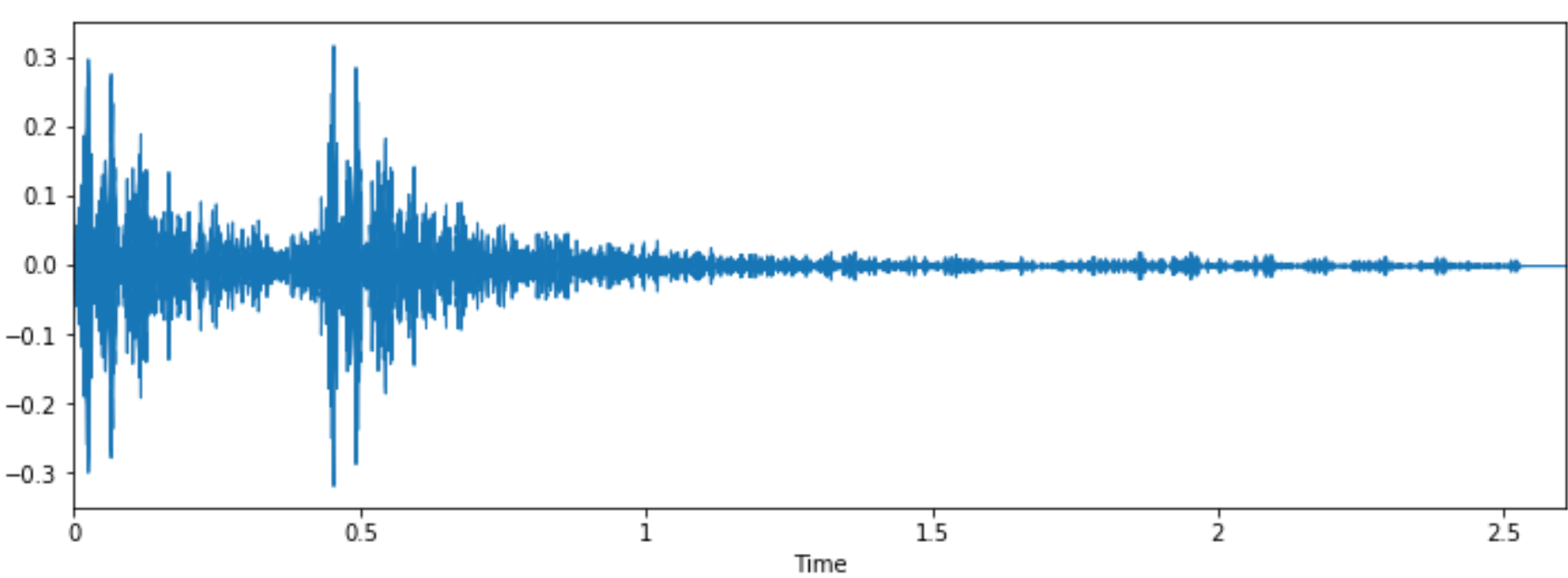


Рисунок 3.3 – Спектрограмма сигнала при выстреле из пистолета

Из визуального анализа видно, что тяжело идентифицировать разницу между некоторыми классами. Примером являются схожие формы сигналов повторяющихся звуков сверления, холостого хода двигателя и отбойного молотка. Аналогично, пики в образце проезжающего автомобиля похож по форме на образец выстрела из пистолета (хотя образцы отличаются тем, что для двух выстрелов имеется два пика по сравнению с одним пиком для проезжающего автомобиля). Также у формы автомобиля есть схожести с уличной музыкой, и звуком играющих детей.

Человеческое ухо может естественным образом обнаружить разницу между гармониками, будет интересно посмотреть, насколько хорошо модель глубокого обучения сможет извлечь необходимые функции для различения этих классов.

Тем не менее, легко отличить от формы волны, разницу между определенными классами, такими как лай собаки и отбойный молоток.

Также стоит отметить, что большинство экземпляров выборки, имеют два аудиоканала (стереозвук), хотя у некоторых всего лишь один аудиоканал. Самый простой способ избежать данной проблемы это объединить два канала в один, путем усреднения двух каналов.

* 1. Проведение исследования
     1. Методика исследования

Методика исследования заключается в подборе наиболее оптимальной и точной архитектуры нейронной сети. Для сравнения выбраны две архитектуры: МЛП и СНС. МЛП был выбран потому, что он является классической архитектурой и является первым выбором в качестве архитектуры сети, когда рассматривается новая область задач для решения которых может примениться нейронная сеть.

Архитектура СНС предлагается в качестве альтернативы МЛП, потому что архитектура СНС широко применяется в задача классификации образов, а данная область смежная с задачей, которая рассматривается в данной работе.

Для сравнения эффективностей данных архитектур, было произведено обучение сетей с данными архитектурами на одинаковом наборе данных. После обучения был произведен замер точности классификации и выбиралась сеть с наибольшей точностью.

Большую точность классификации показала архитектура СНС.

* + 1. Инструменты получения признаков

Для решения проблемы детектирования паттернов а аудиосигналах, было предложено применить методы глубоко обучения, которые оказались крайне успешными в решении задач классификации изображений.

В качестве инструмента извлечения признаков используется МЧКК. Сперва извлекаем МЧКК из экземпляров для каждого кадра с размером окна в несколько миллисекунд. МЧКК суммируют распределение частот по размеру окна, поэтому можно анализировать как частотные, так и временные характеристики звука. Подобное представление аудио позволит нам определить признаки для классификации. Далее следует постобработка полученных признаков.

На рисунке 3.4 показан график вычленения МЧКК для аудиофайла длинной в 60 секунд.

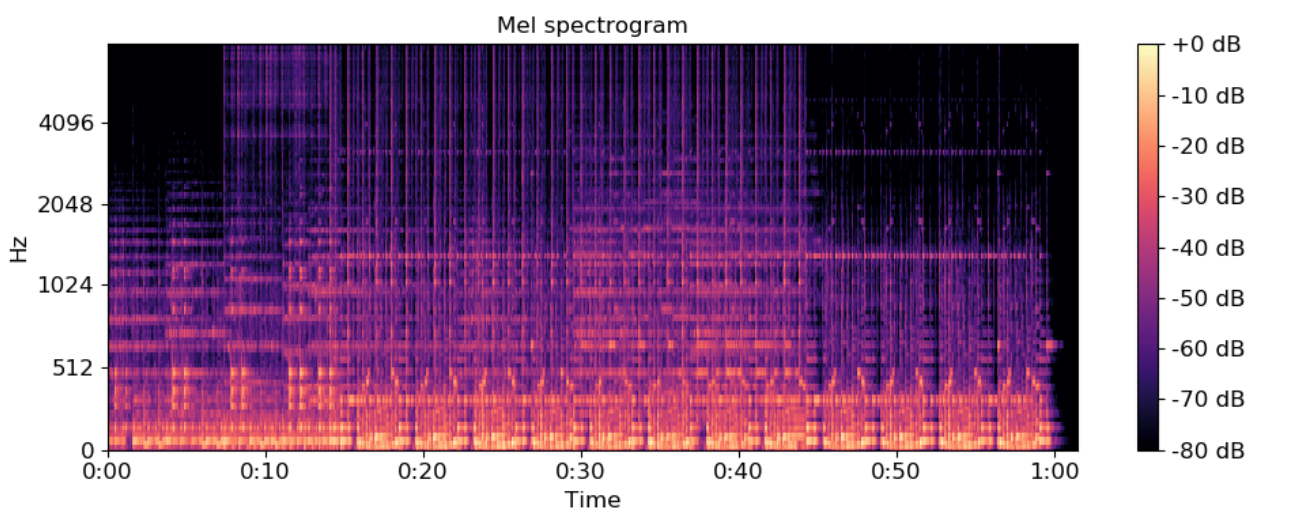


Рисунок 3.4 – Спектрограмма проезжающего потока автомобилей

* + 1. Обучение многослойного персептрона

Для решения задачи классификации мы будет использовать нейронную сеть. Для начала был рассмотрен простая структура нейронной сети – многослойный персептрон (МЛП). Построение МЛП выполнено с помощью Keras и TensorFlow.

Модель сети является последовательной, что позволяет нам размещать слои последовательно. Архитектура разработанного многослойного персептрона состоит из трех слоев: входной, скрытый и выходной слои.

Первый слой является входным. Каждый образец содержит 40 МЧКК, поэтому слой имеет форму 1х40. Первые два слоя имеют 256 нейронов, а функция активации – ReLU, потому что это является оптимальным выбором для нейронных сетей с подобной архитектурой. Уровень исключения равен 50%, так как это позволит нам регуляризовать нейронную сеть во время обучения, что позволяет получить сеть с более качественным предсказаниями.

Выходной слой будет иметь 10 нейронов, которые соотносятся с количеством классов объектов наборе данных. Функцией активации для этого класса выбрана softmax. Softmax делает выходную сумму близкой к 1, поэтому выходные значения могут быть интерпретированы как вероятности. Затем модель сделает свой прогноз на основе того, какой вариант имеет наибольшую вероятность.

Для тестирования модели существуют различные алгоритмы. Эффективность алгоритмов тестирования представлена в таблице 3.2 [64].

Таблица 3.2 – Точность классификации алгоритмов тестирования

|  |  |
| --- | --- |
| Имя алгоритма | Точность классификации, % |
| RBF SVM | 68 |
| Random Forest 500 | 66 |
| IBk5 | 55 |
| J48 | 48 |
| ZeroR | 10 |

Результаты тестирования точности распознавания модели на тестовой и обучающих выборках получись равными 87,6% и 92,5% соответственно. Значения получились довольно высокие, а также небольшая разница между ними (около 5%). Это нам говорит о том, что при обучении модель не была подвержена переобучению. График зависимости точности классификации от количества эпох представлен на рисунке 3.5.

Рисунок 3.5 – График зависимости точности классификации  
 от количества эпох обучения МЛП

* + 1. Обучение сверточной нейронной сети

Обучение МЛП показало достаточно хороший результат. Однако стоит выяснить, может ли другая архитектура сети получить еще более высокую точность. СНC показывает хороший результат в задачах классификации изображений, поэтому данная архитектура была выбрана для проверки предположения.

Так как СНC требует, чтобы количество входов было равно, мы обнуляем векторы, чтобы они все стали одинакового размера.

Модель сети также остается последовательной, с 4 сверточными слоями и плотным выходным слоем. Количество фильтров для сверточных определяется как 16, 32, 64 и 128. Размер ядра равен матрице 2х2, потому что размер окна в данном случае равен 2.

Первый слой получает форму 40, 174 на 1, где 40 – количество МЧКК, 174 – количество кадров с учетом заполнения и 1 – количество каналов. Функция активации для сверточных слоев будет, как и в прошлой модели – ReLU, а уровень исключения – 20%.

Каждый сверточный слой имеет субдискретизирующий слой размерностью 2 на 2. Все субдискретизирующие слои соединены с одни с усредняющим слоем, который подает усредненные данные на выходной слой. Субдискретизирующие слои позволяют снизить размерность модели (путем уменьшения количества параметров и последовательных вычислений), что ведет к уменьшению времени обучения и снижает переобучаемость.

Выходной слой идентичен выходному слою персептрона.

На рисунке 3.6 представлен график зависимости точности классификации от количества эпох обучения.

Рисунок 3.6 – График зависимости точности классификации  
 от количества эпох обучения СНС

Результаты тестирования точности распознавания модели на тестовой и обучающих выборках получись равными 98,2% и 92,0% соответственно. Точность обучения повысилось на 6% для обучающей и около 4% для тестирующей. Хотя разница между ними увеличилась (до 6%), данное значение не столь велико, что является индикатором того, при обучении модель не была подвержена переобучению.

* 1. Результаты исследования

На этапе разработки модели данные валидации использовались для оценки модели. СНС была выбрана в качестве архитектуры потому, что она показала более высокую точность классификации. Точность классификации у СНС составила 92,0% против 87,6% у МЛП.

Результирующая модель достигла точности классификации 92,0%, что превышает ожидания по точности классификации тестирующей модели RBF SVM.

Валидация модели проводилась c использованием аудиофайлов, которые были взяты из свободного аудио банка, а также с данными, собранными самостоятельно.

В качестве контрольной проверки работоспособности системы рассмотрены аудиофайлы, которые были собраны.

* 1. Описание контрольного примера работы системы
     1. Назначение

Контрольный пример предназначен для проверки правильности работы следующих функций автоматизированной системы детектирования акустического излучения ТС:

* выгрузка аудиофайла на сервер приложения;
* обработка и классификация переданного файла;
* классификация;
* сохранение результатов классификации в базу данных;
* отображение результатов классификации на клиентском устройстве.
  + 1. Исходные данные

Для выполнения контрольного примера 12.03.2020 была сделана аудиозапись ТП на пересечении улиц Нагорная и Майская. Запись сделана с помощью разработанного приложения, которое установили на мобильный телефон iPhone XS под управление iOS 13.

На данной аудиозаписи длиной 160 секунд должно быть зафиксировано 23 автомобиля. На рисунке 3.5 представлена аудиозапись перед отправлением на сервер.



Рисунок 3.5 – Спектрограмма проезжающего потока автомобилей

* + 1. Результаты расчета

В ходе обработки файла, отправленного на сервер, было обнаружено 21 ТС. Пример отображения результата после обработки показан на рисунке 3.6.

Система верно определила большую часть ТС, хотя не смогла 2 ТС. Это связано с тем, что они проехали в один и тот же момент времени с другим ТС, что вызывало перекрытие их акустического излучения. Однако этот факт не повлиял на общую точность классификации, которая составила 92.0% и 87,6%.

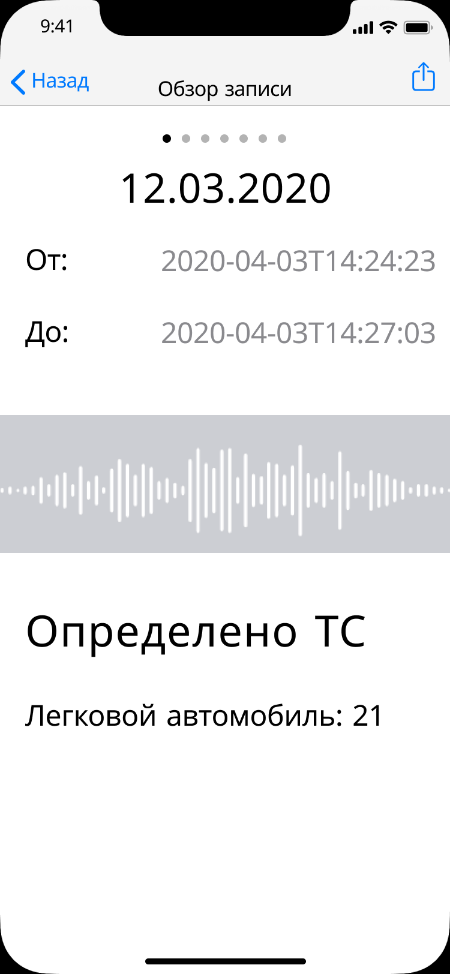


Рисунок 3.6 – Спектрограмма проезжающего потока автомобилей

В таблице 3.3 представлена точность классификации контрольного набора данных, который был собран вручную и взят с открытых звуковых банков.

Таблица 3.3 – Точность классификации алгоритмов тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование аудиозаписи | Количество подсчитанных вручную ТС | Количество определенных системой ТС | Точность классификации, % |
| Запись 1 | 68 | 65 | 92,5 |
| Запись 2 | 43 | 43 | 100,0 |
| Запись 3 | 10 | 10 | 100,0 |
| Запись 4 | 14 | 13 | 92,9 |
| Запись 5 | 33 | 30 | 90,9 |
| Запись 6 | 68 | 65 | 92,5 |
| Запись 7 | 22 | 18 | 81,8 |
| Запись 8 | 3 | 103 | 100,0 |
| Запись 9 | 55 | 50 | 90,9 |
| Запись 10 | 46 | 53 | 86,8 |
| Личная запись 1 | 15 | 13 | 86,7 |
| Личная запись 2 | 7 | 7 | 100,0 |
| Личная запись 3 | 33 | 30 | 90,9 |
| Личная запись 4 | 46 | 35 | 76,1 |
| Личная запись 5 | 18 | 9 | 50,0 |
| Личная запись 6 | 13 | 7 | 53,8 |
| Личная запись 7 | 18 | 15 | 83,3 |
| Личная запись 8 | 24 | 20 | 83,3 |
| Личная запись 9 | 68 | 55 | 80,9 |
| Личная запись 10 | 34 | 17 | 50,0 |

ЗАКЛЮЧЕНИЕ

В соответствии с заданием на выпускную квалификационную работу магистра проведен анализ предметной области, рассмотрены задачи детектирования признаков и особенности акустического излучения, проанализированы нейросетевые технологии и алгоритмы дискретного преобразования сигналов, применяемые в решении аналогичных задач. Выполнен аналитический обзор существующих систем-аналогов. Разработан метод классификации объектов по их акустическому излучению. Построены диаграммы по методологии UML, создана модель данных, разработаны алгоритмы функционирования и описана архитектура системы. Разработана автоматизированная система детектирования признаков акустического излучения транспортных средств на аудиозаписях с использованием сверточных нейронных сетей. Проведены исследования эффективности разработанной системы для решения поставленной задачи. Программное обеспечение системы разработано на языках Python и JavaScript в среде Visual Studio Code.

По теме выпускной квалификационной работы опубликовано 8 научных работ [65-72]. Результаты работы докладывались на различных конференциях международного, областного и регионального уровней, отмечены дипломами о лучших докладах.

Таким образом, основные результаты работы:

* разработан новый метод детектирования признаков (паттернов) акустического излучения транспортных средств в аудиофайлах путем получения мел-частотных кепстральных коэффициентов в качестве признаков, классифицируемых с помощью СНС;
* разработаны алгоритмы функционирования, а также разработано программное обеспечение автоматизированной системы, реализующие предложенный метод детектирования признаков. Проведенное с использованием разработанной системы исследование на наборе данных UrbanSound8k, состоящем из 8732 помеченных аудиозаписей, показало результирующую точность классификации с использованием СНС в 92%. Так, в сравнении с МЛП, СНС обладает более высокой точностью классификации: 92,0% у СНС против 87,6% у МЛП. Кроме этого, проведено исследование на авторском наборе данных, включающем 200 аудиофайлов, записанных на улично-дорожной сети г.о. Самара с использованием разработанного мобильного приложения, и дополненного аудиофайлами, взятыми из открытых банков данных. Результирующая точность на авторском наборе данных с применением СНС составила 84,5%. Некоторое сокращение точности на авторском наборе относительно эталонного UrbanSound8k связано с тем, что в авторских аудиозаписях присутствует наслоение звукового излучения от нескольких ТС в один и тот же момент времени.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Головнин, О.К. Управляемый данными анализ транспортных потоков в различных дорожных условиях [Текст] / О.К. Головнин // V Междунар. конф. и молодёжная школа «Информационные технологии и нанотехнологии» (ИТНТ-2019) : сборник трудов (Самара, 21–24 мая 2019 г.). – Самара : Новая техника, 2019. – Т. 4. Науки о данных. – С. 533-542.
2. Гасников, А.В. Введение в математическое моделирование транспортных потоков [Текст] / А.В. Гасников, С.Л. Кленов, Е.А. Нурминский, Я.А. Холодов [и др.]; под ред. А.В. Гасникова. – М.: МФТИ, 2010. – 360 с.
3. Кузмин, С.А. Алгоритмы обработки видеоинформации в системе мониторинга транспортных потоков [Текст] / С.А. Кузмин. – Электротехнические и информационные комплексы и системы. – № 3. – 2015. – 138 с. – C. 90-96.
4. DeepWiTraffic: Low cost WiFi-based traffic monitoring system using deep learning [Электронный ресурс] // URL: https://arxiv.org/abs/1812.08208 (дата обращения: 23.05.2020).
5. Lee, H. Using LIDAR to validate the performance of vehicle classification stations [Текст] / H. Lee, B. Coifman. – Journal of Intelligent Transportation Systems. – Vol. 19. – 2015. – P. 355-369.
6. Rajab, S.A. Vehicle classification and accurate speed calculation using multi-element piezoelectric sensor [Текст] / S.A. Rajab, A. Mayeli, H.H. Refai. – IEEE Intelligent Vehicles Symposium Proceedings. IEEE, 2014. – P. 894–899.
7. Xu, C. Vehicle classification using an imbalanced dataset based on a single magnetic sensor [Текст] / C. Xu, Y. Wang, X. Bao, F. Li. – Sensors, Vol. 18, №. 6, 2018. – 1690 p.
8. Stocker, M. Situational knowledge representation for traffic observed by a pavement vibration sensor network [Текст] / M. Stocker, M. Ronkko, and M. Kolehmainenm. – IEEE Transactions on Intelligent Transportation Systems, Vol. 15, № 4, 2014. – P. 1441–1450.
9. Meta, S. Vehicle-classification algorithm based on component analysis for single-loop inductive detector [Текст] / S. Meta, M. G. Cinsdikici. – IEEE Trans- actions on Vehicular Technology, Vol. 59, №. 6, 2010. – P. 2795–2805.
10. Yang, B. Vehicle detection and classification for low-speed congested traffic with anisotropic magnetoresistive sensor [Текст] / B. Yang, Y. Lei. – IEEE Sensors Journal, Vol. 15, № 2, 2015. – P. 1132–1138.
11. Ma, W. wireless accelerometer-based automatic vehicle classification prototype system [Текст] / W. Ma, D. Xing, A. McKee, R. Bajwa, C. Flores, B. Fuller, P. Varaiya A. – IEEE Transactions on Intelligent Transportation Systems, Vol. 15, № 1, 2014. – P. 104–111.
12. Georg, J. Vehicle detection and classification from acoustic signal using ANN and KNN in Control Communication and Computing (ICCC) [Текст] / J. George, L. Mary, K. Riyas. – International Conference on. IEEE, 2013. – P. 436–439.
13. Lee, H. Side-fire LIDAR-based vehicle classification [Текст] / H. Lee, B. Coifman. – Transportation Research Record: Journal of the Transportation Research Board, № 2308, 2012. – P. 173–183
14. Odat, E. Vehicle classification and speed estimation using combined passive infrared/ultrasonic sensors [Текст] / E. Odat, J. S. Shamma, C. Claudel. – IEEE Transactions on Intelligent Transportation Systems, 2017. 1241 p.
15. Won, M. WiTraffic: Low-cost and non-intrusive traffic monitoring system using WiFi [Текст] / M. Won, S. Zhang, S. H. Son // 26th International Conference on Computer Communication and Networks (ICCCN). – IEEE, 2017. – P. 1–9.
16. Tang, T. Arbitrary-oriented vehicle detection in aerial imagery with single convolutional neural networks [Текст] / T. Tang, S. Zhou, Z. Deng, L. Lei, H. Zou. – Remote Sensing, Vol. 9, № 11, 2017. – 1170 p.
17. Chen, Z. Vehicle detection, tracking and classification in urban traffic [Текст] / Z. Chen, T. Ellis, S. A. Velastin // 15th International IEEE Conference on Intelligent Transportation Systems, IEEE, 2012. – P. 951–956.
18. Chidlovskii, B. Vehicle type classification from laser scans with global alignment kernels [Текст] / B. Chidlovskii, G. Csurka, J. Rodriguez-Serrano // 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). – IEEE, 2014. – P. 2840–2845.
19. What is a feature extraction? [Электронный ресурс] // URL: https://deepai.org/machine-learning-glossary-and-terms/feature-extraction (дата обращения: 23.05.2020).
20. Alpaydin, E. Introduction to Machine Learning [Текст] / E. Alpaydin. – London: The MIT Press, 2010. – 110 p.
21. An introduction to feature selection [Электронный ресурс]. – https://machinelearningmastery.com/an-introduction-to-feature-selection/
22. World of Physics. Wolfram [Электронный ресурс] // URL: http://scienceworld.wolfram.com/physics/Radiation.html (дата обращения: 23.05.2020).
23. Прохоров, М.А. Физическая Энциклопедия. Звук [Текст] / М.А. Прохоров, А.М. Бонч-Бруевич, А. М. Балдин. – Физическая энциклопедия: [в 5 т.] / Гл. ред. А. М. Прохоров. — М.: Советская энциклопедия (т. 1 – 2); Большая Российская энциклопедия (т. 3—5), 1988—1999. – 1463 c.
24. Васильев, А.В. Исследование уровня акустического излучения системы «шины автотранспортного средства – дорожное покрытие» [Текст] / А. В. Васильев, Е.А. Комлик // Noise Theory And Practise, 2015. – С. 38-34.
25. Васильев, А.В. Акустическая экология города: учеб. пособие для студентов вузов [Текст] / А.В. Васильев. – Федеральное агентство по образованию, Тольяттинский гос. ун-т Тольятти, 2007. – 166 с.
26. Прохоров, С.А. Структурно-спектральный анализ случайных процессов [Текст] / С. А. Прохоров, В. В. Графкин. – Самарский национальный исследовательский университет им. акад. С.П. Королева, 2010. – 128 c.
27. Прохоров, С.А. Прикладной анализ случайных процессов [Текст] / С.А. Прохоров, В.В. Графкин, О.А. Дягтерева. – Cамарский государственный аэрокосмический университет им. акад. С.П. Королева, 2007. – 586 c.
28. Chen, Y.Y. Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes [Текст] / Y. Y. Chen, Y. H. Lin, C. C. Kung, M.H. Chung, I.H. Yen // Sensors. 19 (9), 2019. – 2047 p.
29. Davis, S. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences [Текст] / S. Davis, P. Mermelstein // IEEE Transactions on Acoustics, Speech, and Signal Processing. – 1980. – Vol. 28. – P. 357-366.
30. Миркес, Е.М. Нейрокомпьютер [Текст] / Е. М. Миркес. – Новосибирск: Наука, 1999. – 337 с.
31. Из истории искусственного интеллекта: история искусственного интеллекта до середины 80-х годов [Электронный ресурс] // URL: https://refdb.ru/look/1538888.html (дата обращения: 23.05.2020).
32. Hoppensteadt, F.C. Weakly connected neural networks [Текст] / F. C. Hoppensteadt, E. M. Izhikevich. – Springer, 1997. – 402 p.
33. Maan, A.K. Survey of Memristive Threshold Logic Circuits [Текст] / A.K. Maan, D. A. Jayadevi, A. P James // IEEE Transactions on Neural Networks and Learning Systems, 2016. – P. 1734–1746.
34. Солдатова, О.П. Основы нейроинформатики [Текст] / О.П. Солдатова. – Самара:СГАУ, 2006. – 131 с.
35. LeCun, Y. Backpropagation Applied to Handwritten Zip Code Recognition [Текст] / Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel // Neural Computation, Vol. 1(4), 1989. – P. 541-551.
36. DeepLearning 0.1. LISA Lab [Электронный ресурс] // URL: http://deeplearning.net/tutorial/lenet.html (дата обращения: 23.05.2020).
37. Matusugu, M. Subject independent facial expression recognition with robust face detection using a convolutional neural network [Текст] / M. Matusugu, M. Katsuhiko, M. Yusuke, K. Yuji // Neural Networks: journal. Vol. 16, № 5, 2003. – P. 555—559.
38. Федоренко, С.В. Модификация алгоритма Грецеля-Блейхута [Текст] / С.В. Федоренко. – Статья. - Журнал Приборостроение, 2013. – С 17–20.
39. Гаусс, К.Ф. Арифметические исследования [Текст] / К.Ф. Гаусс // Труды по теории чисел. – М.: Изд-во АН СССР. – 1959. – 703 с.
40. Котельников, В.А. Собрание трудов в пяти томах [Текст] / В.А. Котельников. – Т. 5, Часть 2. Основы радиотехники. – 486 с.
41. Min, X. HMM-based audio keyword generation [Текст] / X. Min, A. Kiyoharu, N. Yuichi, S. Shin'ichi // Advances in Multimedia Information Processing. – PCM 2004: 5th Pacific Rim Conference on Multimedia. Springer, 2004. – 544 p.
42. Stevens, S. S. Scale for the Measurement of the Psychological Magnitude Pitch [Текст] / S. S. Stevens, J. Volkmann, E. B. Newman. – 1937. – 188 p.
43. Dixon Ward, W. Musical Perception [Текст] / W. Dixon Ward. – Foundations of Modern Auditory Theory / Jerry V. Tobias. — Academic Press, 1970. – 412 p.
44. O'Shaughnessy, D. Speech communication: human and machine [Текст] / D. O'Shaughnessy. – Addison-Wesley, 1987. – 150 p.
45. Оппенгейм, А.В. Цифровая обработка сигналов [Текст] / А. В. Оппенгейм, Р. В. Шафер. – М.: Связь, 1979. – 361 c.
46. Рабинер, Л.Р. Цифровая обработка речевых сигналов [Текст] / Л.Р. Рабинер, Р.В. Шафер. – Рипол Классик, 1981.
47. Sahidullah, M. Design, analysis and experimental evaluation of block-based transformation in MFCC computation for speaker recognition [Текст] / M. Sahidullah, G. Saha // Speech Communication. – Vol. 54 (4), 2012. – P. 543–565.
48. ShotSpotter: The review of the system [Электронный ресурс] // URL: https://www.shotspotter.com/technology/ (дата обращения: 23.05.2020).
49. An Industrial-Strength Audio Search Algorithm [Электронный ресурс] // URL: https://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf (дата обращения: 23.05.2020).
50. Audio Analytics – enabling intelligent products through sound recognition [Электронный ресурс] // URL: https://www.audioanalytic.com/ (дата обращения: 23.05.2020).
51. Иващенко, А.В. Теоретические основы проектирования автоматизированных систем обработки информации и управления [Текст] / А.В. Иващенко, И.А. Лёзин, И.В. Лёзина. – Самара: СНЦ РАН, 2007. – 94 с.
52. Knuth, D.E. Mathematical writing [Электронный ресурс] // URL: http://jmlr.csail.mit.edu/reviewing-papers/knuth\_mathematical\_writing.pdf (дата обращения: 23.05.2020).
53. Visual Studio Code source code [Электронный ресурс] // URL: https://github.com/microsoft/vscode (дата обращения: 23.05.2020).
54. Microsoft’s new Code editor is built on Google’s Chromium [Электронный ресурс] // URL: https://arstechnica.com/information-technology/2015/04/microsofts-new-code-editor-is-built-on-googles-chromium/ (дата обращения: 23.05.2020).
55. What Is MongoDB? [Электронный ресурс] // URL: https://www.mongodb.com/what-is-mongodb (дата обращения: 23.05.2020).
56. Hassan, S. Microservices and their design trade-offs: A self-adaptive roadmap [Текст] / S. Hassan, R Bahsoon // IEEE International Conference on Services Computing (SCC), IEEE, 2016. – P. 813–818.
57. Lewis, J.M. Microservices: a definition of this new architectural term [Электронный ресурс] // URL: martinfowler.com/articles/microservices.html (дата обращения: 23.05.2020).
58. Fowler, M. Monolith first [Электронный ресурс] // URL: http://martinfowler.com/bli- ki/MonolithFirst.html (дата обращения: 23.05.2020).
59. Теория и практика UML. Диаграмма состояний [Электронный ресурс] // URL: http://www.it-gost.ru/articles/view\_articles/97 (дата обращения: 23.05.2020).
60. Дерябкин, В.П. Методические указания к лабораторному практикуму по UML [Текст] / В.П. Дерябкин, О.К. Либерзон. – Самара, 2008. – 120 c.
61. Буч, Г. Язык UML Руководство пользователя [Текст] / Г. Буч, Д. Рамбо, А. Джекобсон – 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс, 2006. – 496 с.
62. Пользовательский интерфейс [Электронный ресурс] // URL: http://atworks.ru/polzovatelskij-interfejs-chto-eto.html (дата обращения: 23.05.2020).
63. Dataset UrbanSound8k [Электронный ресурс] // URL: https://urbansounddataset.weebly.com/urbansound8k.html (дата обращения: 23.05.2020).
64. A dataset and taxonomy for urban sound research [Электронный ресурс] // URL: www.justinsalamon.com/uploads/4/3/9/4/4394963/salamon\_urbansound\_ acmmm14.pdf (дата обращения: 23.05.2020).
65. Головнин, О.К. Анализ характеристик транспортных потоков в интеллектуальной транспортной системе на основе аудиофайлов [Текст] / О.К. Головнин, А.С. Привалов // Актуальные проблемы автотранспортного комплекса: межвузовский сборник научных статей (с международным участием). – Самара: СамГТУ, 2019. – С. 126-130.
66. Привалов, А.С. Детектирование транспортных средств на аудиозаписи с использованием сверточных нейронных сетей [Текст] / А.С. Привалов // Новые информационные технологии в научных исследованиях : материалы XXIV Всероссийской научно-технической конференции студентов, молодых ученых и специалистов (Рязань, 13–15 ноября 2019 г.). – Рязань : Book Jet, 2019. – С. 221.
67. Golovnin, O.K. A Web-Oriented Approach for Urban Road Traffic Simulation [Электронный ресурс] / O.K. Golovnin, K.V. Pupynin, A.S. Privalov // 2019 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon) (1-4 Oct. 2019, Vladivostok, Russia), IEEE, 2019. – URL : https://ieeexplore.ieee.org/document/8934302 (дата обращения: 23.05.2020).
68. Golovnin, O.K. Vehicle Detection in Audio Recordings by Machine Learning [Электронный ресурс] / O.K. Golovnin, A.S. Privalov, K.V. Pupynin, // 2019 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon) (1-4 Oct. 2019, Vladivostok, Russia), IEEE, 2019. – URL : https://ieeexplore.ieee.org/document/ 8933842 (дата обращения: 23.05.2020).
69. Привалов, А.С. Мобильное приложение для определения характеристик транспортных потоков [Электронный ресурс] / А.С. Привалов, О.К. Головнин // Математика. Компьютер. Образование : тезисы докладов двадцать шестой международной конференции (Пущино, 28 января – 2 февраля 2019 г.). – URL : http://mce.su/rus/archive/mce26/doc333051/ (дата обращения: 23.05.2020).
70. Привалов, А.С. Автоматизированная система извлечения информации о транспортном потоке из аудиозаписи [Текст] / А.С. Привалов, О.К. Головнин // XV Королёвские чтения, посвященные 100-летию со дня рождения Д.И. Козлова : сборник трудов междунар. молодежной научной конф.(Самара, 8–10 октября 2019 г.). – Самара : АНО «Издательство СНЦ», 2019. – Т. 1. – С. 540.
71. Головнин, О.К. Алгоритм детектирования элементарных признаков акустического излучения транспортных средств [Текст] / О.К. Головнин, А.С. Привалов // Перспективные информационные технологии (ПИТ 2019) : труды Международной научно-технической конференции (Самара, 24–26 июня 2019 г.) ; под ред. С.А. Прохорова. – Самара : Издательство Самарского научного центра РАН, 2019. – С. 430–432.
72. Privalov, A.S. System of audio mining for obtaining traffic flow characteristics [Текст] / A.S Privalov, O.K. Golovnin // Information Technologies for Intelligent Decision Making Support : Proceedings of the 7th All-Russian Scientific Conference, May 28–30, 2019, Ufa. – Ufa, 2019. – Vol. 1. – P. 7–10.

ПРИЛОЖЕНИЕ А   
Функции активации нейронов

В таблице А.1 представлен перечень наиболее известных функций активации.

Таблица А.1 – Примеры функций активации

|  |  |  |
| --- | --- | --- |
| Название | Формула | Область значений |
| Линейная |  | (-∞, ∞) |
| Полулинейная |  | (0, ∞) |
| Линейная с насыщением |  | (-1, 1) |
| Логистическая |  | (0, 1) |
| Гиперболический тангенс |  | (-1, 1) |
| Рациональная |  | (-1, 1) |
| Синусоидальная |  | (-1, 1) |
| Экспоненциальная |  | (0, ∞) |
| Гаусса |  | (-∞, ∞) |
| Пороговая |  | (0, ∞) |

ПРИЛОЖЕНИЕ Б   
Руководство пользователя

Б.1 Назначение системы

Автоматизированная система детектирования признаков акустического излучения ТС на аудиозаписях с использование СНС. Система позволяет вести учет количества проезжающих ТС на совершенной аудиозаписи, а также сохранять данные результаты в базу данных.

Б.2 Требования к обеспечению

Минимальные требования к техническому и программному обеспечению серверной части системы:

* тактовая частота процессора – не менее 1,5 ГГц;
* число ядер процессора — не менее 2;
* объем оперативной памяти – не менее 2 Гб;
* дисковая подсистема – не менее 5 Гб свободного места на диске;
* операционная система – любая;
* скорость сетевого адаптера – не менее 100 Мбит/сек.

Для клиентской части понадобятся следующее аппаратное и программное обеспечение:

* мобильный телефон под управление операционной системы Android не ниже версии 8.0;
* мобильный телефон под управление операционной системы iOS не ниже версии 13.0

Б.3 Описание функционала пользователя

На рисунке Б.1 представлена форма аутентификации пользователя. Для входа в систему необходимо ввести логин и пароль и нажать на кнопку «Войти в систему». Учетная запись нужна для того, чтобы можно было идентифицировать отправителя аудиозаписи.

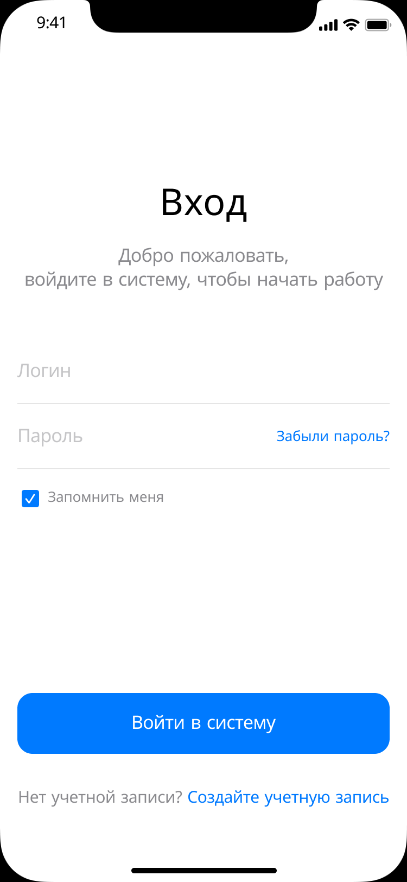


Рисунок Б.1 – Форма аутентификации

Если пользователя еще не существовало в системе, он может создать учетную запись при нажатии на кнопку «Создайте учетную запись» (рисунок Б.2).



Рисунок Б.2 – Форма создании учетной записи

После прохождения аутентификации пользователь будет перенаправлен на карту, где расположена карта с произведенными замерами (рисунок Б.3).

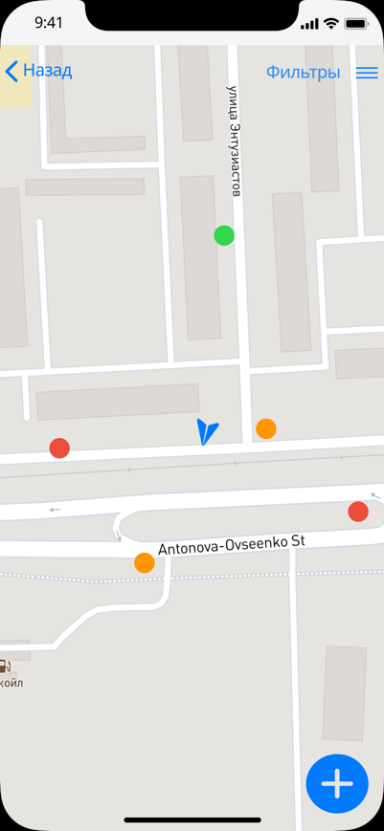


Рисунок Б.3 – Карта с произведенными замерами

При нажатии на «+» пользователь будет перенаправлен на форму записи аудиофайла. Данная форма представлена на рисунке Б.4.



Рисунок Б.4 – Форма записи аудиофайла

После отправки файла и получения ответа с сервера пользователь перейдет на форму, которая содержит информацию о ТП, аудиозапись которого он сделал. Пример данной формы представлен на рисунке Б.5.



Рисунок Б.5 – Форма обзора аудиозаписи

При нажатии на копку «Назад» на форме с картой, пользователь перейдет на списковый обзор замеров, совершенных с помощью системы. Пример данной формы представлен на рисунке Б.6.

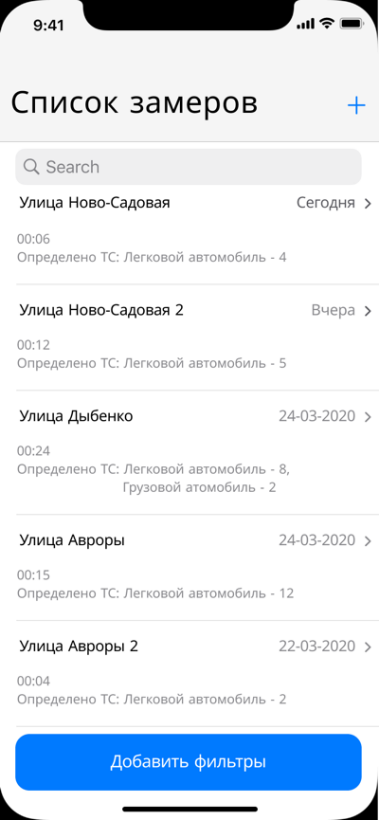


Рисунок Б.6 – Список замеров

В системе предусмотрена фильтрация данных. При нажатии на кнопку «Добавить фильтры» пользователь перейдет на форму выбора фильтров, которые применяются для фильтрации данных. Пример данной формы показан на рисунке Б.7.

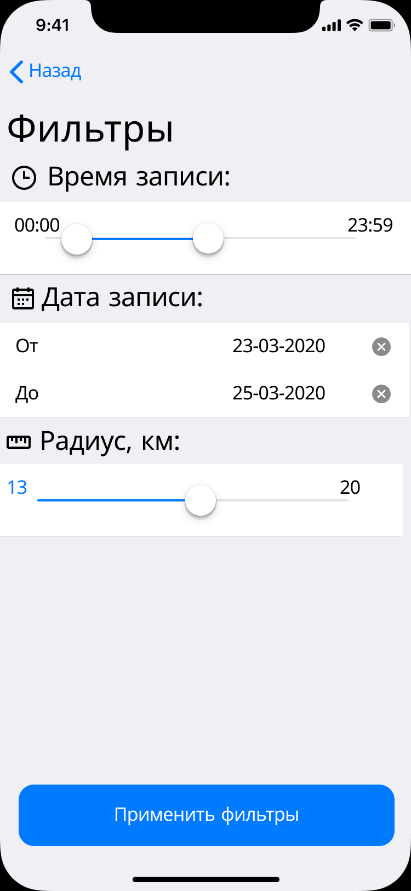


Рисунок Б.7 – Фильтры

ПРИЛОЖЕНИЕ В   
Листинг программы

import pandas as pd

metadata = pd.read\_csv('../UrbanSound Dataset sample/metadata/UrbanSound8K.csv')

metadata.head()

print(metadata.class\_name.value\_counts())

import pandas as pd

import os

import librosa

import librosa.display

from helpers.wavfilehelper import WavFileHelper

wavfilehelper = WavFileHelper()

audiodata = []

for index, row in metadata.iterrows():

file\_name = os.path.join(os.path.abspath('/Volumes/Untitled/ML\_Data/Urban Sound/UrbanSound8K/audio/'),'fold'+str(row["fold"])+'/',str(row["slice\_file\_name"]))

data = wavfilehelper.read\_file\_properties(file\_name)

audiodata.append(data)

audiodf = pd.DataFrame(audiodata, columns=['num\_channels','sample\_rate','bit\_depth'])

print(audiodf.num\_channels.value\_counts(normalize=True))

import librosa

from scipy.io import wavfile as wav

import numpy as np

filename = '../UrbanSound Dataset sample/audio/100852-0-0-0.wav'

librosa\_audio, librosa\_sample\_rate = librosa.load(filename)

scipy\_sample\_rate, scipy\_audio = wav.read(filename)

print('Original sample rate:', scipy\_sample\_rate)

print('Librosa sample rate:', librosa\_sample\_rate)

print('Original audio file min~max range:', np.min(scipy\_audio), 'to', np.max(scipy\_audio))

print('Librosa audio file min~max range:', np.min(librosa\_audio), 'to', np.max(librosa\_audio))

import matplotlib.pyplot as plt

# Original audio with 2 channels

plt.figure(figsize=(12, 4))

plt.plot(scipy\_audio)

plt.figure(figsize=(12, 4))

plt.plot(librosa\_audio)

mfccs = librosa.feature.mfcc(y=librosa\_audio, sr=librosa\_sample\_rate, n\_mfcc=40)

print(mfccs.shape)

import librosa.display

librosa.display.specshow(mfccs, sr=librosa\_sample\_rate, x\_axis='time')

def extract\_features(file\_name):

try:

audio, sample\_rate = librosa.load(file\_name, res\_type='kaiser\_fast')

mfccs = librosa.feature.mfcc(y=audio, sr=sample\_rate, n\_mfcc=40)

mfccsscaled = np.mean(mfccs.T,axis=0)

except Exception as e:

print("Error encountered while parsing file: ", file)

return None

return mfccsscaled

import pandas as pd

import os

import librosa

# Set the path to the full UrbanSound dataset

fulldatasetpath = '/Volumes/Untitled/ML\_Data/Urban Sound/UrbanSound8K/audio/'

metadata = pd.read\_csv('../UrbanSound Dataset sample/metadata/UrbanSound8K.csv')

features = []

# Iterate through each sound file and extract the features

for index, row in metadata.iterrows():

file\_name = os.path.join(os.path.abspath(fulldatasetpath),'fold'+str(row["fold"])+'/',str(row["slice\_file\_name"]))

class\_label = row["class\_name"]

data = extract\_features(file\_name)

features.append([data, class\_label])

# Convert into a Panda dataframe

featuresdf = pd.DataFrame(features, columns=['feature','class\_label'])

print('Finished feature extraction from ', len(featuresdf), ' files')

from sklearn.preprocessing import LabelEncoder

from keras.utils import to\_categorical

X = np.array(featuresdf.feature.tolist())

y = np.array(featuresdf.class\_label.tolist())

le = LabelEncoder()

yy = to\_categorical(le.fit\_transform(y))

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, yy, test\_size=0.2, random\_state = 42)

import numpy as np

from keras.models import Sequential

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D, MaxPooling2D

from keras.optimizers import Adam

from keras.utils import np\_utils

from sklearn import metrics

num\_labels = yy.shape[1]

filter\_size = 2

# Construct model

model = Sequential()

model.add(Dense(256, input\_shape=(40,)))

model.add(Activation('relu'))

model.add(Dropout(0.5))

model.add(Dense(256))

model.add(Activation('relu'))

model.add(Dropout(0.5))

model.add(Dense(num\_labels))

model.add(Activation('softmax'))

model.compile(loss='categorical\_crossentropy', metrics=['accuracy'], optimizer='adam')

model.summary()

score = model.evaluate(x\_test, y\_test, verbose=0)

accuracy = 100\*score[1]

print("Pre-training accuracy: %.4f%%" % accuracy)

from keras.callbacks import ModelCheckpoint

from datetime import datetime

num\_epochs = 100

num\_batch\_size = 32

checkpointer = ModelCheckpoint(filepath='saved\_models/weights.best.basic\_mlp.hdf5',

verbose=1, save\_best\_only=True)

start = datetime.now()

model.fit(x\_train, y\_train, batch\_size=num\_batch\_size, epochs=num\_epochs, validation\_data=(x\_test, y\_test), callbacks=[checkpointer], verbose=1)

duration = datetime.now() - start

print("Training completed in time: ", duration)

score = model.evaluate(x\_train, y\_train, verbose=0)

print("Training Accuracy: ", score[1])

score = model.evaluate(x\_test, y\_test, verbose=0)

print("Testing Accuracy: ", score[1])

import librosa

import numpy as np

def extract\_feature(file\_name):

try:

audio\_data, sample\_rate = librosa.load(file\_name, res\_type='kaiser\_fast')

mfccs = librosa.feature.mfcc(y=audio\_data, sr=sample\_rate, n\_mfcc=40)

mfccsscaled = np.mean(mfccs.T,axis=0)

except Exception as e:

print("Error encountered while parsing file: ", file)

return None, None

return np.array([mfccsscaled])

def print\_prediction(file\_name):

prediction\_feature = extract\_feature(file\_name)

predicted\_vector = model.predict\_classes(prediction\_feature)

predicted\_class = le.inverse\_transform(predicted\_vector)

print("The predicted class is:", predicted\_class[0], '\n')

predicted\_proba\_vector = model.predict\_proba(prediction\_feature)

predicted\_proba = predicted\_proba\_vector[0]

for i in range(len(predicted\_proba)):

category = le.inverse\_transform(np.array([i]))

print(category[0], "\t\t : ", format(predicted\_proba[i], '.32f') )

filename = '../UrbanSound Dataset sample/audio/100852-0-0-0.wav'

print\_prediction(filename)

filename = '../UrbanSound Dataset sample/audio/103199-4-0-0.wav'

print\_prediction(filename)

filename = '../UrbanSound Dataset sample/audio/101848-9-0-0.wav'

print\_prediction(filename)

filename = '../UrbanSound Dataset sample/audio/100648-1-0-0.wav'

print\_prediction(filename)

filename = '../Evaluation audio/dog\_bark\_1.wav'

print\_prediction(filename)

filename = '../Evaluation audio/drilling\_1.wav'

print\_prediction(filename)

filename = '../Evaluation audio/gun\_shot\_1.wav'

print\_prediction(filename)

filename = '../Evaluation audio/siren\_1.wav'

print\_prediction(filename)

import numpy as np

max\_pad\_len = 174

def extract\_features(file\_name):

try:

audio, sample\_rate = librosa.load(file\_name, res\_type='kaiser\_fast')

mfccs = librosa.feature.mfcc(y=audio, sr=sample\_rate, n\_mfcc=40)

pad\_width = max\_pad\_len - mfccs.shape[1]

mfccs = np.pad(mfccs, pad\_width=((0, 0), (0, pad\_width)), mode='constant')

except Exception as e:

print("Error encountered while parsing file: ", file\_name)

return None

return mfccs

import pandas as pd

import os

import librosa

# Set the path to the full UrbanSound dataset

fulldatasetpath = '/Volumes/Untitled/ML\_Data/Urban Sound/UrbanSound8K/audio/'

metadata = pd.read\_csv('../UrbanSound Dataset sample/metadata/UrbanSound8K.csv')

features = []

# Iterate through each sound file and extract the features

for index, row in metadata.iterrows():

file\_name = os.path.join(os.path.abspath(fulldatasetpath),'fold'+str(row["fold"])+'/',str(row["slice\_file\_name"]))

class\_label = row["class\_name"]

data = extract\_features(file\_name)

features.append([data, class\_label])

# Convert into a Panda dataframe

featuresdf = pd.DataFrame(features, columns=['feature','class\_label'])

print('Finished feature extraction from ', len(featuresdf), ' files')

from sklearn.preprocessing import LabelEncoder

from keras.utils import to\_categorical

# Convert features and corresponding classification labels into numpy arrays

X = np.array(featuresdf.feature.tolist())

y = np.array(featuresdf.class\_label.tolist())

# Encode the classification labels

le = LabelEncoder()

yy = to\_categorical(le.fit\_transform(y))

# split the dataset

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, yy, test\_size=0.2, random\_state = 42)

import numpy as np

from keras.models import Sequential

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D, Conv2D, MaxPooling2D, GlobalAveragePooling2D

from keras.optimizers import Adam

from keras.utils import np\_utils

from sklearn import metrics

num\_rows = 40

num\_columns = 174

num\_channels = 1

x\_train = x\_train.reshape(x\_train.shape[0], num\_rows, num\_columns, num\_channels)

x\_test = x\_test.reshape(x\_test.shape[0], num\_rows, num\_columns, num\_channels)

num\_labels = yy.shape[1]

filter\_size = 2

# Construct model

model = Sequential()

model.add(Conv2D(filters=16, kernel\_size=2, input\_shape=(num\_rows, num\_columns, num\_channels), activation='relu'))

model.add(MaxPooling2D(pool\_size=2))

model.add(Dropout(0.2))

model.add(Conv2D(filters=32, kernel\_size=2, activation='relu'))

model.add(MaxPooling2D(pool\_size=2))

model.add(Dropout(0.2))

model.add(Conv2D(filters=64, kernel\_size=2, activation='relu'))

model.add(MaxPooling2D(pool\_size=2))

model.add(Dropout(0.2))

model.add(Conv2D(filters=128, kernel\_size=2, activation='relu'))

model.add(MaxPooling2D(pool\_size=2))

model.add(Dropout(0.2))

model.add(GlobalAveragePooling2D())

model.add(Dense(num\_labels, activation='softmax'))

model.compile(loss='categorical\_crossentropy', metrics=['accuracy'], optimizer='adam')

model.summary()

# Calculate pre-training accuracy

score = model.evaluate(x\_test, y\_test, verbose=1)

accuracy = 100\*score[1]

print("Pre-training accuracy: %.4f%%" % accuracy)

from keras.callbacks import ModelCheckpoint

from datetime import datetime

#num\_epochs = 12

#num\_batch\_size = 128

num\_epochs = 72

num\_batch\_size = 256

checkpointer = ModelCheckpoint(filepath='saved\_models/weights.best.basic\_cnn.hdf5',

verbose=1, save\_best\_only=True)

start = datetime.now()

model.fit(x\_train, y\_train, batch\_size=num\_batch\_size, epochs=num\_epochs, validation\_data=(x\_test, y\_test), callbacks=[checkpointer], verbose=1)

duration = datetime.now() - start

print("Training completed in time: ", duration)

score = model.evaluate(x\_train, y\_train, verbose=0)

print("Training Accuracy: ", score[1])

score = model.evaluate(x\_test, y\_test, verbose=0)

print("Testing Accuracy: ", score[1])

def print\_prediction(file\_name):

prediction\_feature = extract\_features(file\_name)

prediction\_feature = prediction\_feature.reshape(1, num\_rows, num\_columns, num\_channels)

predicted\_vector = model.predict\_classes(prediction\_feature)

predicted\_class = le.inverse\_transform(predicted\_vector)

print("The predicted class is:", predicted\_class[0], '\n')

predicted\_proba\_vector = model.predict\_proba(prediction\_feature)

predicted\_proba = predicted\_proba\_vector[0]

for i in range(len(predicted\_proba)):

category = le.inverse\_transform(np.array([i]))

print(category[0], "\t\t : ", format(predicted\_proba[i], '.32f') )

filename = '../UrbanSound Dataset sample/audio/100852-0-0-0.wav'

print\_prediction(filename)

filename = '../UrbanSound Dataset sample/audio/103199-4-0-0.wav'

print\_prediction(filename)

filename = '../UrbanSound Dataset sample/audio/101848-9-0-0.wav'

print\_prediction(filename)

filename = '../UrbanSound Dataset sample/audio/100648-1-0-0.wav'

print\_prediction(filename)

filename = '../Evaluation audio/dog\_bark\_1.wav'

print\_prediction(filename)

filename = '../Evaluation audio/drilling\_1.wav'

print\_prediction(filename)

filename = '../Evaluation audio/gun\_shot\_1.wav'

print\_prediction(filename)

from helpers.wavfilehelper import WavFileHelper

wavfilehelper = WavFileHelper()

data = wavfilehelper.read\_file\_properties('../Urban Data sample set/audio/30204.wav')

print(data)

export default class App extends Component {

render() {

return (

<View style={styles.container}>

<Geolocation>

<Context.Consumer>

{({loading}) =>

loading ? <Loading visible={true} /> : <AppWithRoute />

}

</Context.Consumer>

</Geolocation>

</View>

);

}

}

const styles = StyleSheet.create({

container: {

width: '100%',

height: '100%',

},

});

class ViewRoad extends Component {

static navigationOptions = ({navigation}) => ({

title: 'View Road',

headerRight: () => (

<Button title="Edit" onPress={() => navigation.navigate('EditRoad')} />

),

});

state = {

modalVisible: false,

damaged\_segments: [],

selected\_segment: {},

latitude: null,

longitude: null,

};

async updateCoordinate(current\_coordinate) {

const distance = getDistanceFrom(current\_coordinate, this.state);

if (this.state.latitude === null || distance > 50) {

await this.setCoordinate(current\_coordinate);

this.loadDamagedSegments();

}

}

setCoordinate(coordinate) {

this.setState({

...coordinate,

});

}

loadDamagedSegments() {

const {latitude, longitude} = this.state;

axios

.get(webservice + '/damaged\_road/' + latitude + '/' + longitude)

.then((response) => {

this.setState({damaged\_segments: response.data});

});

}

componentDidUpdate(prevProps) {

if (this.isScreenFocused(prevProps)) {

this.loadDamagedSegments();

}

}

isScreenFocused(prevProps) {

const isPreviouslyFocused = prevProps.isFocused;

const isCurrentFocused = this.props.isFocused;

return isCurrentFocused && !isPreviouslyFocused;

}

showModal(selected\_segment) {

this.setState({modalVisible: true, selected\_segment});

}

hideModal() {

this.setState({modalVisible: false});

}

render() {

return (

<Context.Consumer>

{({latitude, longitude}) => {

this.updateCoordinate({latitude, longitude});

return (

<Fragment>

<Map>

<Polylines

damaged\_segments={this.state.damaged\_segments}

onPress={(segment) => this.showModal(segment)}

/>

</Map>

<Modal

visible={this.state.modalVisible}

onRequestClose={() => this.hideModal()}

animationType="slide">

<SegmentDetail segment={this.state.selected\_segment} />

</Modal>

</Fragment>

);

}}

</Context.Consumer>

);

}

}

export default withNavigationFocus(ViewRoad);

class SegmentDetail extends Component {

renderItem(label, value, icon) {

return (

<View style={styles.item}>

<View style={styles.itemIcon}>

<Icon name={icon} size={30} />

</View>

<View style={styles.itemData}>

<Text style={styles.itemLabel}>{label}</Text>

<Text style={styles.itemValue}>{value}</Text>

</View>

</View>

);

}

render() {

const {segment} = this.props;

return (

<View style={styles.container}>

<Text style={styles.title}>Segment Detail</Text>

{this.renderItem(

'Damage Type',

segment.damage\_type.name,

'road-variant',

)}

{this.renderItem(

'Damage Level',

segment.damage\_level.name,

'signal-cellular-2',

)}

{this.renderItem('Information', segment.information, 'information')}

</View>

);

}

}