ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

РАЗРАБОТКА ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ ДЛЯ АНАЛИЗА ДАННЫХ ГЕМОДИНАМИЧЕСКОЙ АКТИВНОСТИ МОЗГА

Выпускная квалификационная работа
обучающегося по направлению подготовки 38.03.05 «Бизнес-информатика»
очной формы обучения, группы 12001506
Хабибуллаева Алишера Зокиржон Угли

Научный руководитель:
к.т.н., доцент
Асадуллаев Р.Г.

БЕЛГОРОД 2019

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Одним из наиболее актуальных направлений науки на сегодняшний день является разработка и совершенствование различных управляющих систем, а также информационно-измерительных систем, использующих сигналы центральной нервной системы человека. Такого рода системы выявляют сигналы различного характера, которые непрерывно генерируются в мозге человека и в других отделах центральной нервной системы (ЦНС), производят обработку и преобразование полученных сигналов и на основе полученных сигналов передают управляющие команды на различные устройства.

Основная часть подобных управляющих систем создаётся на основе нейрокомпьютерного интерфейса (НКИ). Нейрокомпьютерный интерфейс является средством взаимодействия мозга и устройств, способных обрабатывать внешние управляющие сигналы. НКИ – это система, измеряющая сигналы активности мозга при помощи датчиков установленных инвазивным (с хирургическим вмешательством) или неинвазивным способом, обрабатывающая их и формирующая управляющие сигналы, соответствующие конкретным распознанным образам.

В настоящее время во всем мире активно ведутся исследования, направленные на создание неинвазивных НКИ, которые основаны на воображении движений. Главной проблемой при создании таких НКИ, которые используют сигналы активности мозга, является повышение точности при классификации воображаемых движений, а также усовершенствование процесса анализа получаемых сигналов.

Объектом данной работы являются НКИ для управления протезами.

Предмет – методы и алгоритмы анализа сигналов активности мозга.

Целью работы является повышение эффективности процесса классификации паттернов движений посредством нейрокомпьютерного интерфейса.

Для достижения указанной цели в работе решаются следующие основные задачи:

1) исследовать современные способы получения сигналов активности мозга при движении и существующие подходы к анализу и классификации сигналов активности мозга;

2) выбрать способ получения сигнала мозговой активности и метод анализа и классификации сигнала;

3) произвести сбор и предварительную обработку данных гемодинамической активности мозга;

4) реализовать модель машинного обучения для классификации паттернов движения кисти руки.

1 Исследование современных методов получения и классификации сигналов активности мозга

1.1 Современное состояние подходов и технических устройств получения сигналов активности мозга

Электроэнцефалография — метод регистрации электрической активности мозга с помощью электродов, располагаемых на коже волосистой части головы [1]. Результаты ЭЭГ используются как для клинической диагностики, так и в научных целях. Существует интракраниальная, или внутричерепная ЭЭГ (intracranial EEG, icEEG), также называемая субдуральной ЭЭГ (subdural EEG, sdEEG) и электрокортикографией (ЭКоГ, или electrocorticography, ECoG). При проведении таких видов ЭЭГ регистрация электрической активности осуществляется непосредственно с поверхности мозга, а не с кожи головы. ЭКоГ характеризуется более высоким пространственным разрешением по сравнению с поверхностной (чрескожной) ЭЭГ, поскольку кости черепа и кожа головы несколько «смягчают» электрические сигналы [1]. При проведении ЭЭГ измеряют суммарные постсинаптические токи. Потенциал действия (ПД, кратковременное изменение потенциала) в пресинаптической мембране аксона вызывает высвобождение нейромедиатора в синаптическую щель. Нейромедиатор, или нейротрансмиттер, — химическое вещество, осуществляющее передачу нервных импульсов через синапсы между нейронами. Пройдя через синаптическую щель, нейромедиатор связывается с рецепторами постсинаптической мембраны. Это вызывает ионные токи в постсинаптической мембране. В результате во внеклеточном пространстве возникают компенсаторные токи. Именно эти внеклеточные токи формируют потенциалы ЭЭГ. ЭЭГ нечувствительна к ПД аксонов [1].

Хотя за формирование сигнала ЭЭГ ответственны постсинаптические потенциалы, поверхностная ЭЭГ не способна зафиксировать активность одного дендрита или нейрона. Правильнее сказать, что поверхностная ЭЭГ представляет собой сумму синхронной активности сотен нейронов, имеющих одинаковую ориентацию в пространстве, расположенных радиально к коже головы. Токи, направленные по касательной к коже головы, не регистрируются. Таким образом, во время ЭЭГ регистрируется активность радиально расположенных в коре апикальных дендритов. Поскольку вольтаж поля уменьшается пропорционально расстоянию до его источника в четвертой степени, активность нейронов в глубоких слоях мозга зафиксировать гораздо труднее, нежели токи непосредственно около кожи.

Токи, регистрируемые на ЭЭГ, характеризуются различными частотами, пространственным распределением и взаимосвязью с различными состояниями мозга (например, сон или бодрствование) [1].

Рисунок 1.1 – Аппарат ЭЭГ

Наш мозг нуждается в постоянном притоке кислорода, снабжение которым происходит через кровь. Этот процесс называется оксигенацией, и

происходит при помощи клеток гемоглобина, отвечающих за перенос кислорода в крови [25].

Различают два типа гемоглобина: оксигемоглобин (HbO2) – с кислородом и дезоксигемоглобин (HHb) – без кислорода.

Ближняя инфракрасная спектроскопия (NIRS) – современная, неинвазивная технология для измерения концентрации окси-, дезоксигемоглобина и общего гемоглобина [11].

Технология основывается на двух главных принципах:

- ткани человека относительно прозрачны для света в ближнем ИК-диапазоне;

- гемоглобин – самый крупный абсорбент света в ближнем ИК-диапазоне. В этом диапазоне HbO2 и HHb демонстрируют кислородзависимую абсорбцию, при этом она отличается для волн разной длины.

Для NIRS необходимы по крайней мере один приемник и один передатчик, чтобы образовать канал. Передатчик передает свет с двумя разными длинами волн. Измерения происходят на уровне капилляров, там, где происходит кислородный обмен. Данные отображают относительную концентрацию, то есть начальное значение неизвестно, но измеряется изменение в концентрации [13].

Функциональная NIRS (fNIRS) – использование NIRS в целях функциональной нейровизуализации. С помощью fNIRS активность мозга измеряется через гемодинамические реакции, связанные с нейроактивностью.

Индекс насыщения тканей (ИНТ), Tissue Saturation Index (TSI), параметр, являющийся прямым индикатором процентного содержания, насыщенного кислородом Hb в ткани непосредственно под сенсором. Это достоверный и быстрый параметр, позволяющий вам измерять у испытуемых оксигенацию тканей головного мозга.

Рисунок 1.2 – Шлем fNIRS

В начале 1980-х годов в американском Университете Джонса Хопкинса группа под руководством Апостолоса Георгопулоса (Apostolos P. Georgopoulos) стала проводить опыты по регистрации активности одиночных нейронов. После двух с лишним лет в экспериментах на моторной коре головного мозга макак было обнаружено, что активность некоторых нейронов меняется, когда обезьяна двигает рукой в определенном направлении. Каждый нейрон настроен на свое направление, вызывающее у него максимальную активность [25]. При отклонении от этого направления активность клеток снижается пропорционально косинусу угла. Стало ясно, что можно с большой точностью расшифровать сигналы группы нейронов, отвечающих за движение конечности. Применяемые в то время электроды внешне напоминали швейные иглы. Они могли эффективно работать лишь несколько часов, пока у их кончиков не скапливались химические компоненты клеток, из-за чего чувствительность катастрофически падала. Кроме того, острие электрода с относительно большим диаметром повреждало нейроны даже при незначительных смещениях головы. В 1990-х годах в Университете Ханеманна Мигель Николелис (Miguel A. L. Nicolelis) и Джон Чэпин (John K. Chapin) применили гибкие электроды с тефлоновым покрытием и диаметром острия около 50 микрон. Результат: удалось снять данные сразу с 48 нейронов

головного мозга крысы. Причем одновременно регистрировалось как восприятие сенсорной информации, так и ответная регуляторная активность.

Рисунок 1.3 – Мозговой имплантат

Развитие технологии позволило отказаться от электродов, внедрённых в мозг, и даже просто от контактов (например, которыми снимают энцефалограмму).

В 2006 году команда доктора Юкиясу Камитани (Yukiyasu Kamitani) из международного института передовых телекоммуникационных исследований (ATR), расположенного близ Киото, совместно с компанией Honda, разработала и продемонстрировала в действии новый тип связи между человеком и машиной - интерфейс мозг-машина (Brain Machine Interface — BMI) [27].

Благодаря этому интерфейсу происходит ежесекундный анализ активности участков мозга, получаемый через магнитно-резонансную томографию (fMRI), после чего компьютерная программа по этим данным анализирует нервные сигналы в мозге, распознавая по ним выполняемые человеком движения (кисти и пальцев) [15]. Задержка между жестом человека и повторением движения робототехническим манипулятором составляла около 7 секунд, а точность распознавания достигла 85%.

Рисунок 1.4 – Аппарат МРТ

На основании исследования подходов и технических устройств получения сигнала мозга было установлено следующее. ЭЭГ не является достаточно портативным устройством получения сигнала, что делает его не применимым в рамках создания нейрокомпьютерного интерфейса, который предполагает свободное использование в повседневной жизни. ЭЭГ имеет большой недостаток в виде большого количества «лишних» сигналов, что усложняет задачу извлечения необходимого сигнала для классификации. Устройство fNIRS подходит для использования в качестве элемента получения сигнала активности мозга нейрокомпьютерного интерфейса, так как является портативным, не мешает при повседневном использовании, сигнал получаемый при помощи устройства позволяет классифицировать паттерны с минимальным использованием фильтров. Мозговой имплантат является инвазивным способом получения информации, он требует хирургического вмешательства. Данный факт не позволяет использовать устройство в рамках разрабатываемого нейрокомпьютерного интерфейса. В

ходе исследования МРТ выяснилось, что сигнал получаемый при помощи устройства, является очень точным, однако, время его получения превышает 3 секунды, МРТ является громоздким аппаратом, что не позволяет использовать его в повседневной жизни, поэтому МРТ не может быть применен в качестве элемента разрабатываемого нейрокомпьютерного интерфейса.

1.2 Анализ методов классификации сигналов в форме временных рядов

Дерево решений - это инструмент поддержки принятия решений, который использует древовидную диаграмму или модель решений и их возможные последствия, включая случайные исходы событий, затраты ресурсов и полезность. Это один из способов отображения алгоритма, который содержит только условные операторы управления.

Дерево решений представляет собой структуру, похожую на блок-схему, в которой каждый внутренний узел представляет «тест» для атрибута (например, подбрасывает ли монета головы или хвосты), каждая ветвь представляет результат теста, а каждый конечный узел представляет метка класса (решение принимается после вычисления всех атрибутов). Пути от корня к листу представляют правила классификации.

Древовидные алгоритмы обучения считаются одними из лучших и наиболее часто используемых контролируемых методов обучения. Древовидные методы расширяют возможности прогнозных моделей с высокой точностью, стабильностью и простотой интерпретации. В отличие от линейных моделей, они довольно хорошо отображают нелинейные отношения. Они адаптируются при решении любых проблем (классификация или регрессия). Алгоритмы дерева решений называются CART (деревья классификации и регрессии) [21].

Такие методы, как деревья решений, случайный лес широко используются во всех видах проблем с данными.

Деревья решений имеют естественную конструкцию «если… то… еще…», которая позволяет легко вписаться в программную структуру. Они также хорошо подходят для задач категоризации, где атрибуты или функции систематически проверяются для определения окончательной категории [22]. Например, дерево решений может эффективно использоваться для определения вида животного.

В результате дерево принятия решений является одним из наиболее популярных алгоритмов классификации, используемых в интеллектуальном анализе данных и машинном обучении [36]. Примеры приложений включают в себя:

1) оценка возможностей расширения бренда для бизнеса с использованием исторических данных о продажах;

2) определение вероятных покупателей продукта с использованием демографических данных, чтобы обеспечить таргетирование ограниченного рекламного бюджета;

3) прогнозирование вероятности дефолта для заемщиков-заявителей с использованием прогнозных моделей, созданных на основе исторических данных;

4) помощь в определении приоритетов лечения пациентов в отделении неотложной помощи с использованием прогностической модели, основанной на таких факторах, как возраст, артериальное давление, пол, локализация и выраженность боли и другие измерения;

5) деревья решений обычно используются в операционном исследовании, особенно в анализе решений, чтобы помочь определить стратегию, которая с наибольшей вероятностью может достичь цели.

Дерево решений - это тип контролируемого алгоритма обучения (с заранее определенной целевой переменной), который в основном

используется в задачах классификации. Он работает как для категориальных, так и для непрерывных входных и выходных переменных [36].

Типы дерева решений основаны на типе целевой переменной, которую мы имеем. Это может быть двух типов:

1) Дерево решений с категориальной переменной: дерево решений, которое имеет категориальную целевую переменную, тогда оно называется деревом решений с категориальной переменной.

2) Дерево решений с непрерывной переменной: дерево решений имеет непрерывную целевую переменную, которая называется деревом решений с непрерывной переменной. Пример: допустим, у есть проблема, чтобы предсказать, заплатит ли клиент свою премию за продление страховой компании (да / нет). Здесь известно, что доход клиента является существенной переменной, но у страховой компании нет информации о доходах для всех клиентов. Теперь, когда известно, что это важная переменная, возможно построить дерево решений для прогнозирования доходов клиентов на основе профессии, продукта и различных других переменных. В этом случае прогнозируются значения для непрерывной переменной.

Алгоритм дерева решений пытается решить проблему, используя представление дерева. Каждый внутренний узел дерева соответствует атрибуту, а каждый листовой узел соответствует метке класса [37].

1) Поместите лучший атрибут набора данных в корень дерева.

2) Разделите тренировочный набор на подмножества. Подмножества должны создаваться таким образом, чтобы каждое подмножество содержало данные с одинаковым значением для атрибута.

3) Повторяйте шаги 1 и 2 для каждого подмножества, пока не найдете листовые узлы во всех ветвях дерева.

В деревьях решений для прогнозирования метки класса для записи начинают с корня дерева. Сравнивают значения корневого атрибута с атрибутом записи. На основе сравнения следуют за веткой, соответствующей этому значению, и переходят к следующему узлу.

Сравнение значения атрибутов записи с другими внутренними узлами дерева продолжается пока не будет достигнут конечный узел с предсказанным значением класса. Моделируемое дерево решений может использоваться для прогнозирования целевого класса или значения [36].

Некоторые предположения, которые делаются при использовании дерева решений:

1) в начале весь тренировочный набор рассматривается как корень;

2) значения признаков предпочтительнее иметь категориальными; если значения являются непрерывными, то они дискредитируются до построения модели;

3) записи распределяются рекурсивно на основе значений атрибутов;

4) порядок размещения атрибутов в качестве корневого или внутреннего узла дерева выполняется с использованием некоторого статистического подхода.

Преимущества дерева решений:

1) легко понять: вывод дерева решений очень легко понять даже для людей не аналитического происхождения. Это не требует никаких статистических знаний, чтобы читать и интерпретировать их. Его графическое представление интуитивно понятно, и пользователи могут легко связать свою гипотезу;

2) полезно при исследовании данных. Дерево решений - это один из самых быстрых способов определения наиболее значимых переменных и связи между двумя или более переменными. с помощью деревьев решений можно создавать новые переменные / функции, которые лучше предсказывают целевую переменную. Алгоритм также может быть использован на этапе исследования данных. например, мы работаем над проблемой, когда у нас есть информация, доступная в сотнях переменных, поэтому дерево решений поможет определить наиболее значимую переменную;

3) деревья решений неявно выполняют скрининг переменных или выбор объектов;

4) деревья решений требуют относительно небольших усилий от пользователей для подготовки данных;

5) требуется меньше очистки данных: требуется меньше очистки данных по сравнению с некоторыми другими методами моделирования. На него не влияют выбросы и пропущенные значения в значительной степени;

6) тип данных не является ограничением: он может обрабатывать как числовые, так и категориальные переменные. Может также обрабатывать проблемы с несколькими выходами;

7) непараметрический метод: дерево решений считается непараметрическим методом. Это означает, что деревья решений не имеют предположений о распределении пространства и структуре классификатора;

8) нелинейные отношения между параметрами не влияют на производительность дерева;

9) количество гиперпараметров для настройки почти равно нулю.

Недостатки дерева решений:

1) избыточная подгонка: учащиеся по дереву решений могут создавать слишком сложные деревья, которые плохо обобщают данные. это называется переоснащением. Переоснащение является одной из наиболее практичных трудностей для моделей дерева решений. Эта проблема решается путем установки ограничений на параметры модели и сокращения;

2) не подходит для непрерывных переменных: при работе с непрерывными числовыми переменными дерево решений теряет информацию, когда оно классифицирует переменные в разных категориях;

3) деревья решений могут быть нестабильными, поскольку небольшие изменения в данных могут привести к созданию совершенно другого дерева. это называется дисперсией, которая должна быть снижена с помощью таких методов, как пакетирование и повышение;

4) жадные алгоритмы не могут гарантировать возвращение глобально оптимального дерева решений. Это может быть смягчено путем

обучения нескольких деревьев, где элементы и образцы выбираются случайным образом с заменой;

5) ученики дерева решений создают необъективные деревья, если доминируют некоторые классы. Поэтому рекомендуется сбалансировать набор данных до согласования с деревом решений;

6) получение информации в дереве решений с категориальными переменными дает необъективный ответ для атрибутов с большим числом «нет» категорий;

7) как правило, это дает низкую точность прогнозирования для набора данных по сравнению с другими алгоритмами машинного обучения;

8) расчеты могут стать сложными, когда есть много меток классов.

Идея бустинга возникла из идеи о том, можно ли изменить слабого «ученика», чтобы он стал лучше. Майкл Кернс сформулировал цель как «проблему бустинга гипотезы», заявив цель с практической точки зрения как: «... эффективный алгоритм для преобразования относительно плохих гипотез в очень хорошие гипотезы» [21].

Слабая гипотеза или слабый ученик определяется как тот, чья производительность, по крайней мере, немного лучше, чем случайный шанс. Эти идеи были основаны на работе Лесли Валианта над обучением без распространения или вероятностным подходом, основой для изучения сложности проблем машинного обучения [21].

Идея гипотезы бустинга заключается в фильтрации наблюдений, оставляя те наблюдения, которые слабый «ученик» может обрабатывать, и сосредотачиваясь на разработке новых слабых обучающихся для обработки оставшихся сложных наблюдений. Идея состоит в том, чтобы несколько раз использовать слабый метод обучения, чтобы получить ряд гипотез, каждая из которых переориентируется на примеры, которые предыдущие считали трудными и неправильно классифицированными.

Первой реализацией бустинга, которая имела большой успех в применении, был Adaptive Boosting или AdaBoost.

Бустинг относится к этой общей проблеме создания очень точного правила предсказания путем сочетания грубых и умеренно неточных эмпирических правил. Слабыми учениками в AdaBoost являются деревья решений с одним разделением, называемые пеньками решений за их краткость. AdaBoost работает, взвешивая наблюдения, делая больший вес на трудно классифицируемых экземплярах и меньше на тех, которые уже хорошо обработаны. Новые слабые «ученики» добавляются последовательно, что фокусирует их обучение на более сложных схемах. Это означает, что выборки, которые трудно классифицировать, получают увеличивающиеся веса, пока алгоритм не определит модель, которая правильно классифицирует эти выборки [37].

Прогнозы делаются большинством голосов прогнозов слабых учеников, взвешенных по их индивидуальной точности. Наиболее успешная форма алгоритма AdaBoost была для задач двоичной классификации и называлась AdaBoost.M1.

Arcing - это сокращение от Adaptive Reweighting and Combining. Каждый шаг в алгоритме дуги состоит из взвешенной минимизации, за которой следует повторное вычисление классификаторов и взвешенного ввода. Эта структура была доработана Фридманом и получила название «Машина бустинга градиента». Позже получила название «градиентный бустинг» или «Градиентный бустинг дерева» [38].

Статистические рамки рассматривают бустинг как проблему численной оптимизации, где цель состоит в том, чтобы минимизировать потерю модели путем добавления слабых учеников, используя процедуру, подобную градиентному спуску. Этот класс алгоритмов был описан как поэтапная аддитивная модель. Это связано с тем, что за один раз добавляется один новый слабый учащийся, а существующие слабые учащиеся в модели замораживаются и остаются без изменений. Следует обратить внимание, что эта поэтапная стратегия отличается от поэтапных подходов, которые корректируют ранее введенные термины при добавлении новых. Обобщение

позволило использовать произвольные дифференцируемые функции потерь, расширив метод за пределы задач бинарной классификации, чтобы поддержать регрессию, мультиклассовую классификацию и многое другое.

Градиентный бустинг включает в себя три элемента:

1) функция потерь для оптимизации;

2) слабый ученик, чтобы делать прогнозы;

3) аддитивная модель для добавления слабых учеников для минимизации функции потерь.

1) Функция потери. Используемая функция потерь зависит от типа решаемой проблемы. Она должна быть дифференцируемой, но поддерживаются многие стандартные функции потерь, и можно определить свои собственные. Например, регрессия может использовать квадратичную ошибку, а классификация может использовать логарифмическую потерю [38].

Преимущество структуры градиентного бустинга состоит в том, что новый алгоритм повышения не должен быть выведен для каждой функции потерь, которую может потребоваться использовать, вместо этого это достаточно универсальная структура, которая может использовать любую дифференцируемую функцию потерь.

2) Слабый ученик. В методе градиентного бустинга в качестве слабого ученика используются деревья решений. В частности, используются деревья регрессии, которые выводят реальные значения для расщеплений и чьи выходные данные могут быть добавлены вместе, что позволяет добавлять последующие выходные данные моделей и «корректировать» остатки в прогнозах.

Деревья строятся жадным образом, выбирая лучшие точки разделения, основываясь на показателях чистоты, таких как Джини, или чтобы минимизировать потери [38].

Первоначально, как в случае с AdaBoost, использовались очень короткие деревья решений, которые имели только одно разделение, называемое пнем принятия решения. Большие деревья могут быть использованы с 4-8 уровнями.

Обычно слабые ученики ограничиваются определенными способами, такими как максимальное количество слоев, узлов, расщеплений или листовых узлов. Это должно гарантировать, что ученики остаются слабыми, но все же могут быть построены жадным способом.

3) Аддитивная модель. Деревья добавляются по одному, а существующие деревья в модели не изменяются. Процедура градиентного спуска используется для минимизации потерь при добавлении деревьев. Традиционно, градиентный спуск используется для минимизации набора параметров, таких как коэффициенты в уравнении регрессии или веса в нейронной сети. После вычисления ошибки или потери веса обновляются, чтобы минимизировать эту ошибку. Вместо параметров есть слабые подмодели учеников или, более конкретно, деревья решений. После расчета потерь, чтобы выполнить процедуру градиентного спуска, необходимо добавить в модель дерево, которое уменьшает потери (т.е. следовать градиенту). Делается это путем параметризации дерева, затем изменяются параметры дерева и таким образом получается движение в правильном направлении (уменьшая остаточные потери). Обычно этот подход называется функциональным градиентным спуском или градиентным спуском с функциями. Вывод для нового дерева затем добавляется к выводу существующей последовательности деревьев, чтобы исправить или улучшить окончательный вывод модели. Добавляется фиксированное количество деревьев или обучение прекращается, когда потери достигают приемлемого уровня или больше не улучшаются в наборе данных внешней проверки [39].

Улучшения в базовом градиентном бустинге. Повышение градиента является жадным алгоритмом и может быстро дополнить набор данных для обучения. Он может извлечь выгоду из методов регуляризации, которые штрафуют различные части алгоритма и, как правило, улучшают производительность алгоритма за счет уменьшения переоснащения.

Четыре способа улучшения базового градиентного бустинга:

1) ограничения дерева. Важно, чтобы слабые ученики обладали навыками, но оставались слабыми;

2) взвешенные обновления. Прогнозы каждого дерева складываются последовательно;

3) стохастическое повышение градиента. Большое понимание ансамблей, бэггинга и случайного леса позволяло жадно создавать деревья из подвыборок учебного набора данных. Это же преимущество можно использовать для уменьшения корреляции между деревьями в последовательности в моделях градиентного бустинга. Этот вариант усиления называется стохастическим градиентным бустингом;

4) повышение наказания за градиент. Дополнительные параметры могут быть наложены на параметризованные деревья в дополнение к их структуре. Значения веса листьев деревьев могут быть упорядочены с помощью популярных функций регуляризации.

Машина опорных векторов, сокращенно МОВ, может использоваться для задач регрессии и классификации. Но метод широко используется в целях классификации [35].

Цель алгоритма машины опорных векторов состоит в том, чтобы найти гиперплоскость в N-мерном пространстве (N - число признаков), которая четко классифицирует точки данных. Чтобы разделить два класса точек данных, есть много возможных гиперплоскостей, которые могут быть выбраны. Цель алгоритма МОВ - найти плоскость, которая имеет максимальный запас, то есть максимальное расстояние между точками данных обоих классов. Максимизация расстояния запаса обеспечивает некоторое подкрепление, так что будущие точки данных могут быть классифицированы с большей уверенностью.

Гиперплоскости - это границы принятия решений, которые помогают классифицировать точки данных. Точки данных, падающие по обе стороны от гиперплоскости, можно отнести к разным классам. Кроме того, размерность гиперплоскости зависит от количества признаков [43]. Если количество

входных объектов равно 2, то гиперплоскость - это просто линия. Если количество входных объектов равно 3, то гиперплоскость становится двумерной плоскостью. Становится трудно представить, когда количество функций превышает 3.

Опорные векторы - это точки данных, которые находятся ближе к гиперплоскости и влияют на положение и ориентацию гиперплоскости [38]. Используя эти векторы поддержки, происходит максимизация маржи классификатора. Удаление опорных векторов изменит положение гиперплоскости. Аспекты, которые помогают создавать МОВ:

1) Большая маржинальная интуиция. В логистической регрессии берутся выходные данные линейной функции и сдавливается значение в диапазоне [0,1], используя сигмовидную функцию. Если сжатое значение больше порогового значения (0,5), происходит присвоение ему метки 1, в противном случае присваивается метка 0. В МОВ используются выходные данные линейной функции, и если этот результат больше 1, то определяется к одному классу, и если результат равен -1, идентифицируется с другим классом. Поскольку пороговые значения изменяются на 1 и -1 в МОВ, то получается этот диапазон значений усиления ([- 1,1]), который действует как маржа.

2) Функция стоимости и обновления градиента. Алгоритм МОВ стремится максимизировать разницу между точками данных и гиперплоскостью. Функция потерь, которая помогает максимизировать маржу - это шарнирная потеря. Стоимость равна 0, если прогнозируемое значение и фактическое значение имеют один и тот же знак. Если это не так, рассчитывается величина убытка. Также добавляется параметр регуляризации в функцию стоимости. Задача параметра регуляризации - сбалансировать максимизацию маржи и потери.

3) Функция потерь для МОВ. Затем, когда есть функция потерь, используются частные производные по весам, чтобы найти градиенты. Используя градиенты, можно обновить веса.

4) Градиенты. Когда нет неправильной классификации, то есть модель правильно предсказывает класс точки данных, нужно только обновить градиент из параметра регуляризации. Когда есть неправильная классификация, то есть модель делает ошибку в предсказании класса точки данных, происходит включение потери вместе с параметром регуляризации, чтобы выполнить обновление градиента.

В машинном обучении часто приходится иметь дело с многомерными данными. Но не все фитчи, которые используются в модели, являются значимыми. Добавление многих фитчей в надежде, что модель будет лучше учиться и даст точные результаты, часто приводит к проблеме, которую обычно называют «Проклятием размерности», которая гласит: по мере роста числа элементов или измерений объем данных, который необходимо точно обобщать, растет в геометрической прогрессии. Чтобы преодолеть эту проблему, нужно определить наиболее важные функции в наборе данных [44]. Одним из таких методов, позволяющих идентифицировать основные признаки из набора данных, тем самым уменьшая размерность набора данных, является анализ главных компонентов (PCA).

PCA принимает большой набор переменных, использует зависимости между этими переменными, чтобы представить его в более управляемой, низкоразмерной форме, не теряя слишком много информации. PCA служит хорошим инструментом для исследования данных и часто проводится как часть исследовательского анализа данных (EDA) [43].

Предположим, у есть n наблюдений и d переменных в наборе данных, и необходимо изучить связь между различными переменными в рамках EDA. Для большего значения d, например, 60, получается d (d-1) / 2 двумерные диаграммы рассеяния. Такое огромное количество графиков (1770, в данном случае) затрудняет выявление взаимосвязи между особенностями. Кроме того, эти 2D-графики содержат только часть общей информации, представленной в наборе данных [43].

PCA - это метод извлечения признаков, поэтому он комбинирует входные переменные особым образом, а затем избавляется от «наименее важных» переменных, сохраняя при этом наиболее ценные части (или главные компоненты) всех переменных [39].

Основным компонентом является нормализованная линейная комбинация исходных объектов в наборе данных. Предположим, что, начиная с d-мерных векторов и желая суммировать их, проецируя вниз в k-мерное подпространство так, чтобы оси нового подпространства указывали в направлениях наибольшей дисперсии данных. Нашим конечным результатом будет проекция исходных векторов на k направлений, называемых основными (главными) компонентами (ГК).

Выбор правильного количества основных компонентов важен для обеспечения эффективности PCA. Набор данных, содержащий n наблюдений и d признаков, учитывает мин (n - 1, d) различных основных компонентов. Но интерес представляют только первые несколько важных компонентов, которых достаточно для объяснения большого количества изменений в наборе данных. Один из способов определить необходимое количество компонент - взглянуть на совокупный коэффициент дисперсии, который является функцией числа компонентов.

PCA широко используется во многих областях, таких как компьютерное зрение и сжатие изображений [39]. В основном используется для следующих приложений:

1) визуализация данных: PCA позволяет визуализировать объекты больших размеров в более низкие измерения;

2) метод частных наименьших квадратов: объекты PCA могут использоваться в качестве основы для линейной модели в методе частных наименьших квадратов;

3) уменьшение размерности: уменьшает размерность элементов, теряя лишь небольшое количество информации;

4) обнаружение выбросов (улучшение качества данных): проецирует набор переменных в меньшем количестве измерений и выделяет посторонние значения;

Учитывая матрицу X, которая соответствует n наблюдениям с d характеристиками, и входу k, основная цель PCA состоит в том, чтобы разложить матрицу X на две меньшие матрицы, Z и W, так что X = ZW, где Z имеет размеры n * k и W имеет размеры k * d (см. Рисунок 4). Каждый ряд Z является фактором загрузки. Каждая строка W называется главным компонентом.

Важные моменты при использовании метода главных компонент представлены ниже.

Перед началом PCA данные должны быть нормализованы. Это важно, так как разные переменные в наборе данных могут измеряться в разных единицах. PCA в ненормализованном наборе данных приводит к более высоким собственным значениям для переменной, имеющей максимальную дисперсию, соответствующую собственному вектору ее первого ГК [43].

PCA может применяться только на числовых данных. Таким образом, если данные имеют категориальные переменные, они должны быть преобразованы в числовые значения. Такие переменные могут быть представлены с использованием схемы кодирования 1-из-N без наложения искусственного упорядочения. PCA не должен проводиться, когда большинство независимых функций являются категориальными. Вместо этого можно использовать категориальный анализ главных компонент для преобразования категорий в числовые значения посредством оптимального масштабирования.

Рекуррентные нейронные сети (RNN) являются мощным и надежным типом нейронных сетей и относятся к наиболее перспективным алгоритмам на данный момент, потому что они являются единственными с внутренней памятью. RNN относительно старые, как и многие другие алгоритмы глубокого обучения. Они были первоначально созданы в 1980-х годах, но

могут показать свой реальный потенциал только через несколько лет из-за увеличения доступной вычислительной мощности, огромного количества данных, которые мы имеем в настоящее время, и изобретения LSTM в 1990-х годах [36].

Из-за их внутренней памяти, RNN в состоянии помнить важные вещи о входе, который они получили, что позволяет им быть очень точными в предсказании того, что будет дальше. Именно по этой причине они являются предпочтительным алгоритмом для последовательных данных, таких как временные ряды, речь, текст, финансовые данные, аудио, видео, погода и многое другое, потому что они могут сформировать гораздо более глубокое понимание последовательности и ее контекста по сравнению с другими алгоритмами [43].

Рекуррентные нейронные сети дают предсказательные результаты в последовательных данных, которые другие алгоритмы не могут. Последовательные данные - это в основном просто упорядоченные данные, где связанные вещи следуют друг за другом. Примерами являются финансовые данные или последовательность ДНК. Наиболее популярным типом последовательных данных являются, возможно, данные временных рядов, которые представляют собой просто ряд точек данных, перечисленных во временном порядке.

Нейронные сети RNN и сети прямого распространения названы в честь того, как они передают информацию. В нейронной сети с прямой связью информация перемещается только в одном направлении, от входного слоя, через скрытые слои, к выходному слою. Информация движется прямо через сеть. Из-за этого информация никогда не касается узла дважды. Нейронные сети прямого распространения, не имеют памяти о входных данных, которые они получили ранее, и поэтому плохо предсказывают, что будет дальше. Поскольку сеть прямой связи учитывает только текущие входные данные, она не имеет понятия порядка во времени. Они просто не могут вспомнить ничего о том, что произошло в прошлом, кроме их обучения. В RNN информация

циклически проходит через цикл. Когда он принимает решение, он принимает во внимание текущие входные данные, а также то, что он узнал из входных данных, полученных ранее. Обычный RNN имеет кратковременную память. В сочетании с LSTM они также имеют долгосрочную память [43]. Рекуррентные нейронные сети добавляют непосредственное прошлое к настоящему. Поэтому рекуррентная нейронная сеть имеет два входа: настоящее и недавнее прошлое. Это важно, потому что последовательность данных содержит важную информацию о том, что будет дальше, поэтому RNN может делать то, что другие алгоритмы не могут. Прямая нейронная сеть назначает, как и все другие алгоритмы глубокого обучения, матрицу веса своим входам, а затем производит выход. Следует обратить внимание, что RNN применяет веса к текущему, а также к предыдущему входу. Кроме того, они также настраивают свои веса как для градиентного спуска, так и для обратного распространения во времени. В то время как прямые нейронные сети отображают один вход на один выход, RNN может отображать один ко многим, многие ко многим (перевод) и многие к одному (классификация голоса). Сети Long Short-Term Memory (LSTM) являются расширением для рекуррентных нейронных сетей, что в основном расширяет их память. Поэтому LSTM хорошо подходит, чтобы учиться на важном опыте, который очень долго задерживается в памяти между эпохами. Блоки LSTM используются в качестве строительных единиц для слоев RNN, которые затем часто называют сетью LSTM. LSTM позволяют RNN помнить свои входы в течение длительного периода времени [38]. Это связано с тем, что LSTM содержат свою информацию в памяти, что очень похоже на память компьютера, потому что LSTM может читать, записывать и удалять информацию из своей памяти. Эта память может рассматриваться как закрытая ячейка, где закрытая означает, что ячейка решает, следует ли хранить или удалять информацию, основываясь на важности, которую он присваивает информации. Присвоение важности происходит через веса, которые также изучаются алгоритмом [21]. Это просто означает, что со временем он узнает, какая информация важна, а какая нет. В LSTM у вас есть три гейта: вход,

забыть и выходной гейт. Гейты в LSTM являются аналоговыми, в виде сигмоидов, что означает, что они варьируются от 0 до 1. Тот факт, что они аналоговые, позволяет им делать обратное распространение. Проблемные вопросы исчезающих градиентов решаются через LSTM, потому что он держит градиенты достаточно крутыми, и поэтому обучение относительно короткое и точность высокая.

Сверточная нейронная сеть (ConvNet / CNN) - это алгоритм глубокого обучения, который может принимать входное изображение, присваивать важность (усваиваемые веса и смещения) различным аспектам / объектам в изображении и может отличать один от другого. Предварительная обработка, требуемая в ConvNet, значительно ниже по сравнению с другими алгоритмами классификации [43]. В то время как в примитивных методах фильтры сконструированы вручную, при достаточном обучении, ConvNets имеют возможность изучать эти фильтры / характеристики. Архитектура ConvNet аналогична архитектуре связности нейронов в человеческом мозге и была вдохновлена организацией визуальной коры. Отдельные нейроны реагируют на раздражители только в ограниченной области поля зрения, известной как рецептивное поле. Коллекция таких полей перекрывается, чтобы покрыть всю визуальную область. В случае очень простых двоичных изображений метод может показывать среднюю оценку точности при выполнении прогнозирования классов, но он почти не имеет конкурентов, когда речь идет о сложных изображениях, имеющих повсеместную зависимость от пикселей.

ConvNet может успешно захватывать пространственные и временные зависимости в изображении с помощью соответствующих фильтров. Архитектура обеспечивает лучшее согласование с набором данных изображения благодаря уменьшению количества задействованных параметров и возможности повторного использования весов [43]. Другими словами, сеть можно научить лучше понимать сложность изображения.

Роль ConvNet состоит в том, чтобы преобразовать изображения в форму, которую легче обрабатывать, не теряя при этом функций, которые имеют

решающее значение для получения хорошего прогноза. Это важно, когда необходимо спроектировать архитектуру, которая не только хороша в обучении, но и масштабируема для массивных наборов данных [37].

Цель операции Convolution - извлечь из входного изображения высокоуровневые объекты, такие как края. ConvNets не обязательно должен быть ограничен только одним сверточным слоем. Традиционно, первый ConvLayer отвечает за захват низкоуровневых функций, таких как края, цвет, ориентация градиента и т.д. Благодаря добавленным слоям архитектура также адаптируется к высокоуровневым функциям, что дает сеть, которая имеет хорошее понимание изображений в наборе данных, как и человек.

Операция имеет два типа результатов: один, в котором свернутый элемент имеет меньшую размерность по сравнению с вводом, а другой - размерность либо увеличивается, либо остается неизменной [37]. Это делается путем применения Valid Padding в случае первого или Same Padding в случае последнего. Подобно сверточному слою, слой пула отвечает за уменьшение пространственного размера свернутого элемента. Это должно уменьшить вычислительную мощность, необходимую для обработки данных, за счет уменьшения размерности. Кроме того, это полезно для извлечения доминирующих признаков, которые являются инвариантными относительно вращения и позиционирования, таким образом поддерживая процесс эффективного обучения модели.

Существует два типа пулинга: максимальный и средний. Max Pooling возвращает максимальное значение из части изображения, покрытой ядром. С другой стороны, Average Pooling возвращает среднее значение всех значений из части изображения, покрытой ядром. Max Pooling также выступает в роли шумоподавителя. Он полностью исключает шумовые активации, а также выполняет шумоподавление наряду с уменьшением размерности. С другой стороны, Среднее Объединение просто выполняет уменьшение размерности как механизм подавления шума. Следовательно, можно сказать, что Max Pooling работает намного лучше, чем Average Pooling [44].

Сверточный слой и объединяющий слой вместе образуют i-й слой сверточной нейронной сети. В зависимости от сложности изображений количество таких слоев может быть увеличено для еще более детального захвата деталей низкого уровня, но за счет большей вычислительной мощности.

После прохождения вышеуказанного процесса модель сможет выявить особенности. Двигаясь дальше, следует сгладить окончательный результат и передать его в обычную нейронную сеть для целей классификации.

Классификация - Полностью связанный слой (слой FC). Добавление полностью связанного слоя - это (обычно) дешевый способ изучения нелинейных комбинаций высокоуровневых характеристик, представленных выходными данными сверточного слоя. Слой Fully-Connected изучает, возможно, нелинейную функцию в этом пространстве [37]. Теперь, когда входное изображение преобразовано в подходящую форму для многоуровневого персептрона, изображение сводится в вектор-столбец. Сглаженный вывод подается в нейронную сеть с прямой связью, и обратное распространение применяется к каждой итерации обучения. В течение ряда эпох модель способна различать доминирующие и определенные низкоуровневые элементы изображений и классифицировать их с помощью техники классификации Softmax.

Существуют различные архитектуры доступных CNN, которые были ключевыми в построении алгоритмов, которые обеспечивают и должны обеспечивать работу ИИ в целом в обозримом будущем. Список некоторых из них: LeNet, AlexNet, VGGNet, GoogLeNet, RESNET, ZFNet [43].

Исследование методов классификации позволило выявить достоинства и недостатки различных подходов. Рассмотрены такие подходы как: деревья решений, метод опорных векторов, градиентный бустинг, метод анализа главных компонент, рекуррентные и сверточные нейронные сети. Каждый метод был проанализирован на предмет возможности работы с данными в виде временного ряда, а также данными, которые поступают непрерывно и в

большом количестве. Исследование показало, что большинство методов не способны работать с данными в виде временного ряда, некоторые подходы имеют низкую эффективность при работе с большим объемом данных.

2 Синтез методов получения и обработки данных мозговой активности

2.1 Выбор способа получения сигнала

Для создания нейрокомпьютерного интерфейса необходимо использовать один из способов получения сигнала активности мозга. Среди всех способов выделяются основные технологии, описанные в главе 1.1. Для получения данных активности мозга в рамках данной выпускной квалификационной работы необходимо использовать оборудование, которое способно: работать в автономном режиме, у пользователя должна быть возможность использовать оборудование в повседневной жизни, скорость считывания информации должна позволять получать данные с задержкой не более 3 секунд, возможность использования устройства любым пользователем вне зависимости от наличия конечностей и срока их потери, если такой факт присутствовал. В ходе проведенного исследования были выявлены преимущества и недостатки рассмотренных способов получения данных активности мозга при использовании их для разработки нейрокомпьютерного интерфейса.

При исследовании способа получения данных мозговой активности при помощи электроэнцефалографии было выявлено, что данный способ позволяет получать сигналы с высокой скоростью (до 500Ггц) и фиксировать факт и намерение совершить движение. Но при этом ЭЭГ имеют большое количество «лишних» сигналов из-за которых сложно выделить полезный сигнал, который необходим для реализации нейрокомпьютерного интерфейса. Также существенным недостатком способа получения сигнала при помощи ЭЭГ является тот факт, что устройство ЭЭГ практически невозможно применять в повседневной жизни из-за габаритов аппарата и чувствительности к внешним факторам.

Изучив устройство fNIRS работающее на технологии функциональной спектроскопии в ближнем инфракрасном свете обнаружено, что данное устройство анализирует гемодинамическую активность мозга, что позволяет решить проблему управления протезом при атрофии мышц или отсутствии конечности. Устройства, разработанные на основе данного метода можно использовать в повседневной жизни, т.к. они являются портативными и при использовании не подвергаются серьезному внешнему воздействию, которое провоцирует появление большого количества «лишних» сигналов.

Нейрохирургические имплантаты являются инвазивным способом получения данных активности мозга, т.е. требуют хирургического вмешательства для установки в голове у пользователя. При изучении данного метода было выявлено, что при тестирования нейрохирургических имплантатов для управления бионическими протезами не было получено положительных результатов, не удалось считать сигнал активации мозга и установили, что использование инвазивной технологии внелабораторных условиях оказалось невозможным.

Исследуя способ получения сигналов активности мозга при помощи магнито-резонансного сканирования выявлено, что данный способ обладает возможностью получить данные с высокой точностью, но время отклика устройства составляет порядка 7с. Помимо данного недостатка МРТ не возможно использовать в повседневной жизни и у пользователей зачастую имеются противопоказания для частого использования МРТ.

Таблица 1 – Сравнение подходов и устройств для получения сигнала активности мозга

Свойства Устройство	Необходимость хирургического вмешательства	Возможность портативного использования	Высокая скорость получения сигнала	Качество получаемого сигнала
ЭЭГ	-1	0	1	0
МРТ	-1	-1	-1	1
Нейроимлантат	1	1	0	0
fNIRS	-1	1	1	0

Проанализировав все способы получения данных мозговой активности и сопоставив возможности рассмотренных устройств и требований к оборудованию для разработки информационного обеспечения в рамках данной работы был выбран способ получения данных при помощи устройства fNIRS.

На основании исследования подходов и технических устройств получения сигнала активности мозга было проведено сравнение исследованных способов, которое представлено в таблице 1. Рассмотрев достоинства и недостатки каждого из подходов выбран способ для получения сигнала активности мозга, который будет использоваться для получения данных для классификации.

2.2 Выбор метода анализа и классификации данных

Данные активности мозга, получаемые с устройства fNIRS представляют из себя непрерывную последовательность в виде временного ряда. Для обработки и классификации данных необходимо подобрать метод, который способен: работать с большим количеством данных без потери качества выполнения функций, имеет возможность быстрой и неэнергозатратной перенастройки, способен работать с данными которые представлены в виде временного ряда.

При анализе методов обработки данных представленных в главе 1.2 были выявлены отличительные особенности каждого представленного метода. Деревья решений обладают такими положительными свойствами как: доступная интерпретация вывод дерева решений, что позволяет очень легко понять правила вывода даже для людей не аналитического происхождения. Графическое представление дерева решений интуитивно понятно, и пользователи могут легко связать свою гипотезу; дерево решений - это один из самых быстрых способов определения наиболее значимых переменных и

связи между двумя или более переменными; деревья решений требуют относительно небольших усилий от пользователей для подготовки данных; тип данных не является ограничением. Однако данный метод имеет значительные недостатки в виде избыточной подгонки, модели, обучающиеся по дереву решений, могут создавать слишком сложные деревья, которые плохо обобщают данные. Также данный метод не подходит для непрерывных переменных, деревья решений могут быть нестабильными. Как правило, это дает низкую точность прогнозирования для набора данных по сравнению с другими алгоритмами машинного обучения. Расчеты могут стать сложными, когда есть много меток классов.

Метод градиентного бустинга является мощным алгоритмом распознавания. Данный эффект достигается благодаря адаптивной технике композиции. Преимуществом бустинга является то, что он позволяет использовать много вариаций, таких как: использовать различные функции потерь, возможность выбора произвольной функции потерь, возможность использования любого набора базовых алгоритмов, благодаря математическому обоснованию бустинга не сложно провести некоторые математические и алгоритмические оптимизации. В свою очередь бустинг является трудоемким методом, что приводит к его медленной работе. Для работы с большим количество данных необходимо применение множества дополнительных модификаций, бустинг не показывает хороших результатов при комбинировании с другими мощными алгоритмами. При сложном алгоритме модели градиентного бустинга результат работы сложно интерпретировать.

Машина опорных векторов сложна в применении в задачах, где данные имеют большую размерность. В этом случае необходимо предварительно, используя другой метод снижения размерности данных, понизить размерность данных и лишь потом приступать к использованию машины опорных векторов. Главным недостатком машины опорных векторов является то факт, что метод использует не полный набор данных для классификации, а только

данные, которые находятся в непосредственной близости от границы класса. Преимуществом данного метода является возможность обучения машины опорных векторов на небольшом наборе данных.

Метод анализа главных компонент относительно прост в реализации, производит классификацию с высокой скоростью, не забивает оперативную память для хранения данных. В данном методе присутствует возможность быстрого дообучения модели. Метод основан на подборе лучшей аппроксимации данных на входе, что приводит к тому, что необходимо подавать только чистые данные для корректной работы. Метод анализа главных компонент ищет только линейные зависимости, что зачастую является существенным недостатком.

Рекуррентные и сверточные нейронные сети в отличии от других методов обнаруживают не только линейную зависимость, но также и нелинейную зависимость, что является огромным преимуществом нейронных сетей. Рекурентные нейронные сети могут обрабатывать временной ряд произвольной длины, использовать встроенную память для предсказания и определения контекста. Рекуррентные нейронные сети обладают большим количеством преимуществ по сравнению с другими методами анализа сигнала. Единственным недостатком является то, что для корректного обучения сети необходимо большое число данных, при маленькой обучающей выборке точность классификации или предсказания не будет очень высокой.

Сверточные нейронные сети позволяют с высокой точностью классифицировать изображения и видео. При свертке входных данных появляется возможно кластеризовать каждый элемент с высокой точностью. Недостатком сверточной нейронной сети является ее низкие показатели при работе с временными рядами.

Таблица 2 – Сравнение методов классификации данных

Свойства Метод	Работа с большим кол-вом данных	Возможность дообучения	Работа с временным рядом	Возможность ретроспективы
Дерево решений	-1	-1	-1	-1
Модель градиентного бустинга	0	-1	0	-1
Метод опорных векторов	0	0	0	-1
Метод главных компонент	1	1	0	-1
Рекурентные нейронные сети LSTM	1	1	1	1
Сверточные нейронные сети	1	1	-1	-1

Исследовав все методы анализа и классификации данных, был сделан выбор в пользу использования рекуррентных нейронных сетей, т.к. они в наибольшей степени способны используя функции и возможности архитектуры сети классифицировать и предсказывать данные, при этом учитывая предыдущие состояния системы.

На основании проведенного исследования методов классификации проведен анализ каждого из рассмотренных способа, в таблице 2 представлено сравнение методов в соответствии с критериями необходимыми в рамках проводимой работы. По результатам анализа и сравнения выбран метод для классификации сигналов активности мозга.

3 Разработка архитектуры и обучение нейронной сети для классификации паттернов движения кисти руки

3.1 Разработка плана эксперимента и сбор данных

Эксперимент для сбора данных производился в Научно-проектном центр когнитивных нейронаук и нейротехнологий (НейроНет центр). При проведении исследований использовался портативный томограф NIRSport Model 88. В томографе используется 16 датчиков, с помощью которых регистрируется гемодинамическая активность в 20 каналах [10]. Используемые датчики делятся на два вида: оптоды и детекторы. Оптоды и детекторы представляют собой специальные LED лампочки, которые излучают инфракрасный свет в определенном диапазоне, который способен проникать сквозь череп человека на расстояние 30мм. Во время работы устройства, оптоды излучаю свет, который попадая в мозг отражается от кровяных тел, а детектор сканирует пространство и получает информацию о том какое количество излученного света отражается от кровяных тел, обогащенных кислородом и необогащенных.

В используемом томографе 8 датчиков являются оптодами и имеют красную маркировку (рисунок 3.1), оставшиеся 8 датчиков – детекторы, имеют маркировку синего цвета. Соединения датчиков между собой являются каналами получения данный активности мозга.

Рисунок 3.1 – Оптоды и детекторы устройства

Данные на каналах отображают сигнал о количестве насыщенного кислородом гемоглобина (HbO) и ненасыщенного кислородом гемоглобина (Hb). Для записи активности мозга с использованием устройства fNIRS применяется специализированное программный продукт NIRSLab. Интерфейс программы представлен на рисунке 3.2.

Пакет NIRSLab представляет собой универсальную среду анализа, разработанную для поддержки изучения изменяющихся во времени измерений в ближней инфракрасной области тканей с использованием устройств fNIRS. Он состоит из модулей для:

1) Импорт данных измерений NIRS вместе с соответствующей информацией, касающейся выполненных измерений (например, расположение оптических датчиков на коже головы, время применения стимулов или экспериментальные задания).

2) Создание файлов, содержащих информацию о положении датчиков и времени измерения, вместе с наборами инструментов для редактирования этой информации.

3) Предварительная обработка данных измерений с использованием наборов инструментов, которые исключают избыточные каналы данных (шум), удаление нерелевантных временных интервалов эксперимента, удаление артефактов из данных и фильтрация, позволяющая исключить нерелевантные полосы частот.

4) Вычисление гемодинамических состояний с использованием настроек параметров, зависящих от длины волны и ее амплитуды.

5) Визуализация реакции временных рядов данных во время измерений и вычисленных гемодинамических состояний.

6) Извлечение динамических характеристик из гемодинамических состояний с использованием метода статистического параметрического картирования (SPM) и сопоставление этой информации с визуальной моделью области головы.

Эти модули связаны с автоматическим файловым менеджером и регистром метаданных, что делает этот программный продукт удобным и мощным средством работы с устройствами fNIRS [28].

При исследовании литературы было выявлено, что активация мозга человека при выполнении движения происходит в диапазоне от 0,01Hz до 0,2 Hz [29].

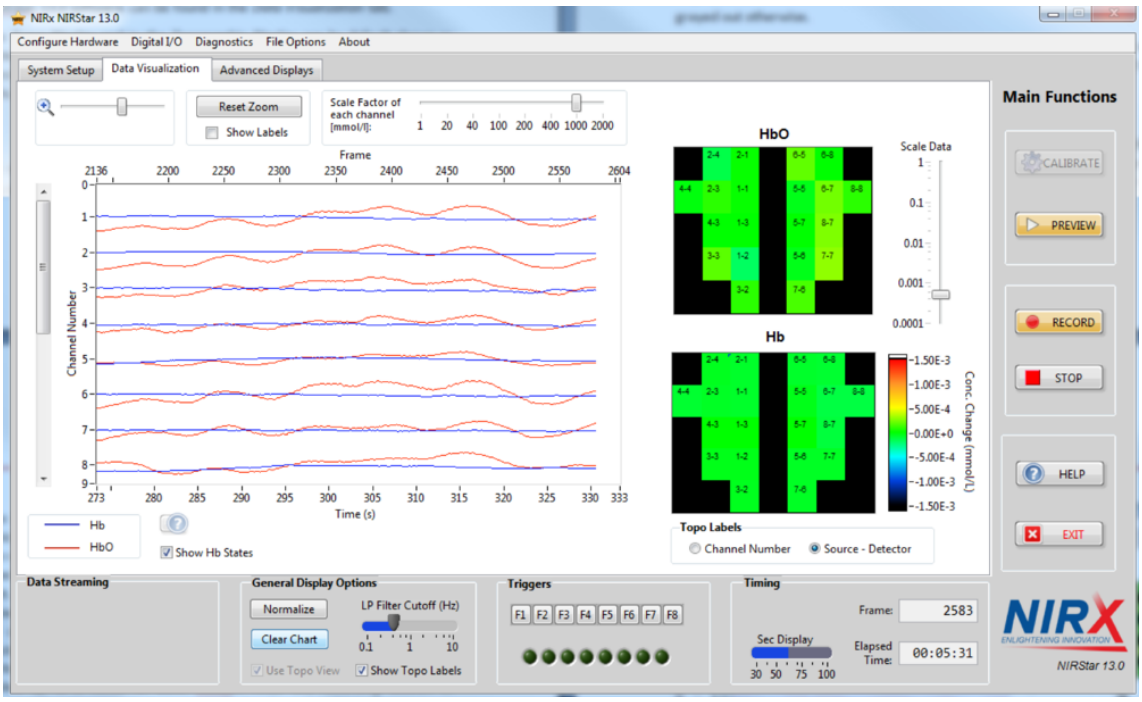

Рисунок 3.2 – Интерфейс NIRSLab

Датчики расположены в области Моторной коры, которая отвечает за планирование, контроль и выполнение произвольных движений. Схема расположения датчиков представлена на рисунке 3.1.

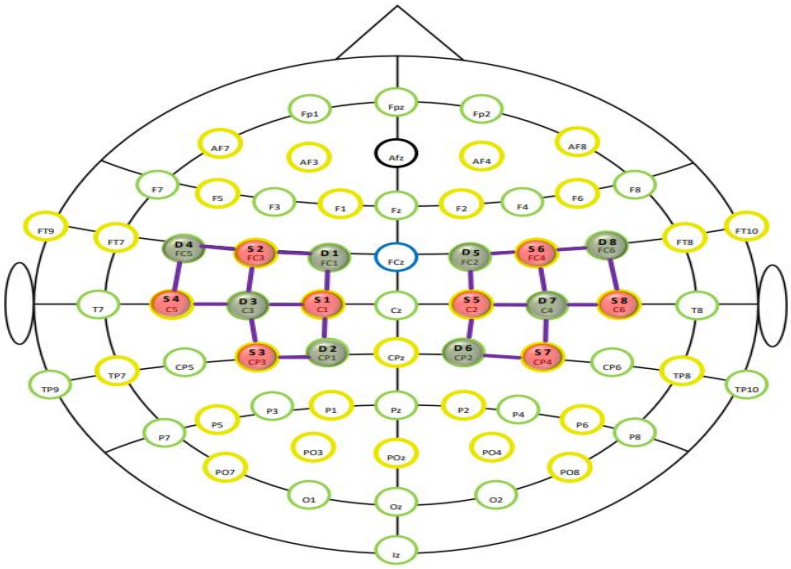

Рисунок 3.3 – Схема расположения датчиков

До начала эксперимента был разработан план чередовании действий, который представлен в таблице 3.1.

Таблица 3.1 - План эксперимента

№	Время, с.	Действие
1	0-20	Расслабление (шум)
2	21-40	Сжатие кисти руки
3	41-60	Расжатие кисти руки
4	61-80	Сжатие кисти руки
5	81-100	Расжатие кисти руки
6	101-120	Сжатие кисти руки
7	121-140	Расжатие кисти руки
8	141-160	Сжатие кисти руки
9	161-180	Расжатие кисти руки
10	181-200	Сжатие кисти руки
11	201-220	Расжатие кисти руки

В исследовании приняли участие 9 человек различного пола в возрасте от 18 до 30 лет. После установки и каллибровки оборудования на испытуемом (рисунок 3.4), при помощи специализированного программного обеспечения NIRSLab был произведен сбор данных гемодинамической активности мозга, в момент сбора данных испытуемый следовал плану эксперимента (рисунок 3.5).

Рисунок 3.4 – Калибровка устройства

Рисунок 3.5 – Запись данных активности мозга

Проанализировав принцип работы устройства fNIRS и программного средства NIRSLab были проведены эксперименты с участием 9 человек для получения данных активности мозга при выполнении сжатия и разжатия кисти руки. Программное средство NIRSLab позволяет настраивать необходимую частоту, на которой происходит активация мозга при выполнении движений. Для проведения эксперимента разработан план эксперимента, которому следовали все испытуемые.

3.2 Предварительная обработка данных

После того как все испытуемые прошли эксперимент, при помощи программы NIRSLab были получены данные о гемодинамической активности мозга. Фрагмент исходных данных, полученных после их снятия, представлен на рисунке 3.6.

5.1015544e-05	-1.4532299e-04	2.4023722e-04	9.1173351e-05	2.3905632e-04	7.1808230e-04	-2.1259465e-04	4.2496147e-0
4.1544772e-05	-1.1207145e-04	2.3763394e-04	7.5515694e-05	2.2881253e-04	7.0371490e-04	-2.6044331e-04	6.5360473e-0
-6.9506413e-08	-1.7221241e-04	1.9356873e-04	5.5392695e-05	1.8805097e-04	6.9323316e-04	-2.7118550e-04	2.0747924e-0
-1.4217806e-04	-3.1524422e-04	5.4675187e-05	-1.8440285e-05	1.3935374e-04	5.9127872e-04	-4.2827099e-04	-9.4693438e-0
-3.0600766e-04	-5.2205345e-04	-9.2943948e-05	-9.4283205e-05	1.0020373e-04	5.0485518e-04	-5.0583511e-04	-2.1729230e-0
-8.2578394e-05	-2.5165276e-04	1.0387584e-04	4.0048824e-05	2.2083132e-04	6.4680225e-04	-3.3531775e-04	-2.0809295e-0
-3.4859539e-05	-1.4105045e-04	1.7397186e-04	9.0951682e-05	2.7805226e-04	6.8920694e-04	-2.7727510e-04	1.1669852e-0
-9.9295488e-05	-1.9039055e-04	1.2480131e-04	8.0506885e-05	2.9078618e-04	6.7091989e-04	-3.1755489e-04	-1.0359799e-0
-2.2919834e-04	-3.1812492e-04	1.0512206e-05	5.1461735e-05	2.3547074e-04	6.0390169e-04	-4.1657989e-04	-7.0819173e-0
-3.6629935e-04	-5.0670485e-04	-1.5308234e-04	-4.0777311e-05	1.0718995e-04	4.6509886e-04	-5.5503723e-04	-2.4566535e-0
-6.3620078e-05	-2.4744176e-04	4.8625305e-05	1.0404579e-04	2.0139031e-04	6.3607498e-04	-2.8643938e-04	-1.7388249e-0
1.0348960e-04	-4.2042513e-05	2.4007200e-04	2.3919549e-04	3.3514289e-04	7.6499337e-04	-1.9933203e-04	6.5758970e-0
6.7772766e-05	-9.3545081e-05	1.7535147e-04	2.4175616e-04	3.6764654e-04	7.7160844e-04	-2.5380729e-04	1.9887870e-0
-1.3030454e-05	-1.9567017e-04	1.0672484e-04	1.8910089e-04	3.1078863e-04	7.0542534e-04	-3.4116412e-04	-1.8699633e-0
-1.6418086e-04	-3.5556842e-04	-2.9284402e-05	1.0896892e-04	2.0782837e-04	5.9331948e-04	-4.4094320e-04	-1.1848241e-0
-1.7459939e-05	-2.3900907e-04	7.3394694e-05	1.9223518e-04	2.9172550e-04	7.1838323e-04	-2.5235835e-04	5.2907468e-0
2.6138709e-04	7.6911150e-05	3.4379598e-04	3.7520690e-04	4.7832230e-04	8.6401793e-04	-1.1185233e-04	1.8675467e-0
2.4287781e-04	6.2147563e-05	3.1158363e-04	3.9302887e-04	4.7155496e-04	8.4867893e-04	-1.2859121e-04	1.5290319e-0
2.0897426e-04	-1.5279058e-05	2.6117128e-04	3.6980503e-04	4.5276459e-04	7.8749505e-04	-2.2949967e-04	1.0181391e-0
7.4859840e-05	-1.6146297e-04	1.4124181e-04	2.9278762e-04	3.8167075e-04	6.8484724e-04	-3.0762158e-04	3.7518093e-0
-4.5396255e-06	-3.0912444e-04	1.3551616e-05	2.5908920e-04	3.1544623e-04	6.8166910e-04	-3.0833940e-04	-1.4873698e-0
3.7627338e-04	1.3985782e-04	3.9493694e-04	4.8702606e-04	5.5085710e-04	8.9909142e-04	-3.1965228e-05	2.6774703e-0
4.0555862e-04	1.8514493e-04	4.2959524e-04	5.5419398e-04	5.8226584e-04	9.0495002e-04	-5.3551834e-05	2.6919995e-0

Рисунок 3.6 – Исходные данные активности мозга

На рисунке 3.7 представлен визуализированный сигнал одного канала в исходном виде.

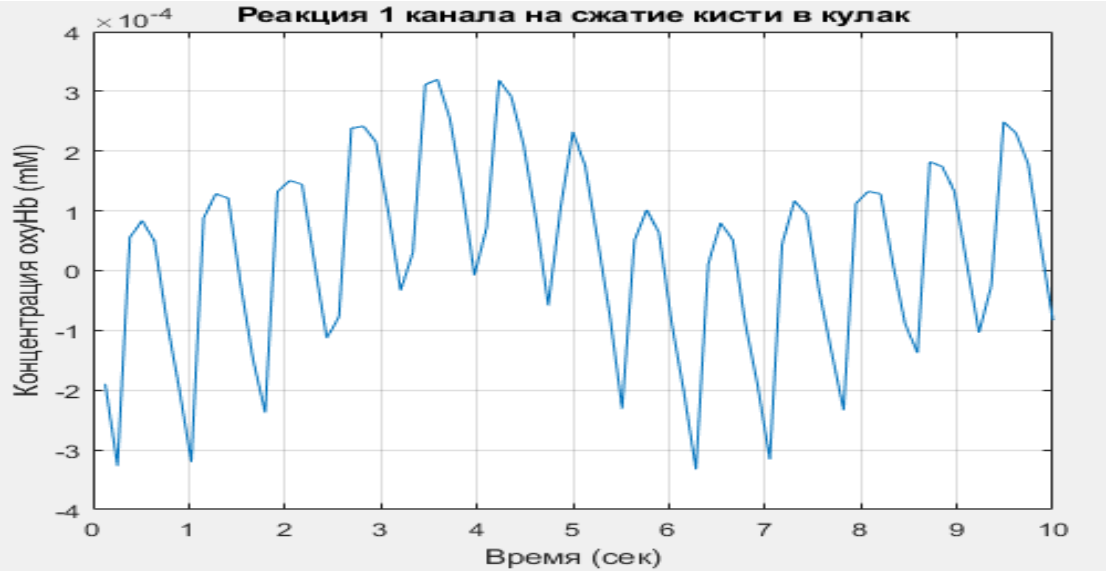

Рисунок 3.7 – Полученный сигнал одного канала

На рисунке 3.8 представлен сигнал, зарегистрированный на всех 20 каналах при сжатии кисти руки (на верхнем графике каналы 1-10, на нижнем 11-20).

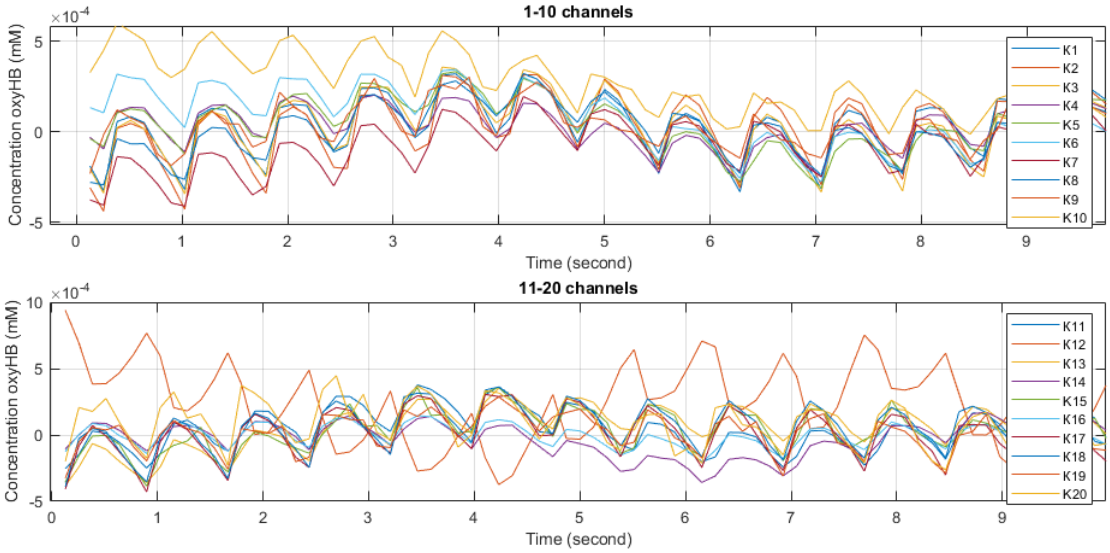

Рисунок 3.8 – Регистрируемые данные на 20 каналах

Для того, чтобы определить момент активации гемодинамической активности мозга необходимо сгладить изначально регистрируемый сигнал, тем самым избавившись от так называемых артефактов. Для этого необходимо использовать фильтр, который позволит избавиться от шума в виде краткосрочных колебаний и будет корректно отражать поведение временного ряда. С учетом этих целей был подобран фильтр, использующий метод скользящего среднего. Данный фильтр не затрачивает много ресурсов для работы и может быть реализован при помощи языка программирования Python. Для реализации скользящего среднего использовалась библиотека Pandas. В листинге 1 представлена функция скользящего среднего.

```
def rolling(self, window, min_periods=None, center=False,
            win_type=None, on=None, axis=0, closed=None):
    axis = self._get_axis_number(axis)
    return rwindow.rolling(self, window=window,
                           min_periods=min_periods,
                           center=center, win_type=win_type,
                           on=on, axis=axis, closed=closed)
data=data.rolling(data [:, 0], 5)
```

Листинг 1 – Функция скользящего среднего

После применения фильтра был получен сглаженный сигнал (рисунок 3.9).

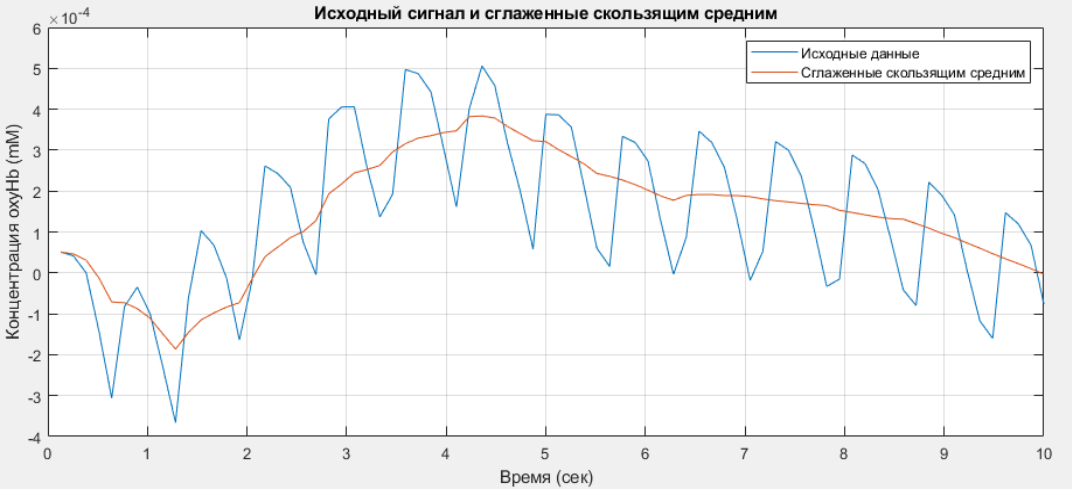

Рисунок 3.9 – Исходный и сглаженный сигнал одного канала

Сглаженный сигнал по всем каналам, получившийся после применения фильтра, представлен на рисунке 3.10.

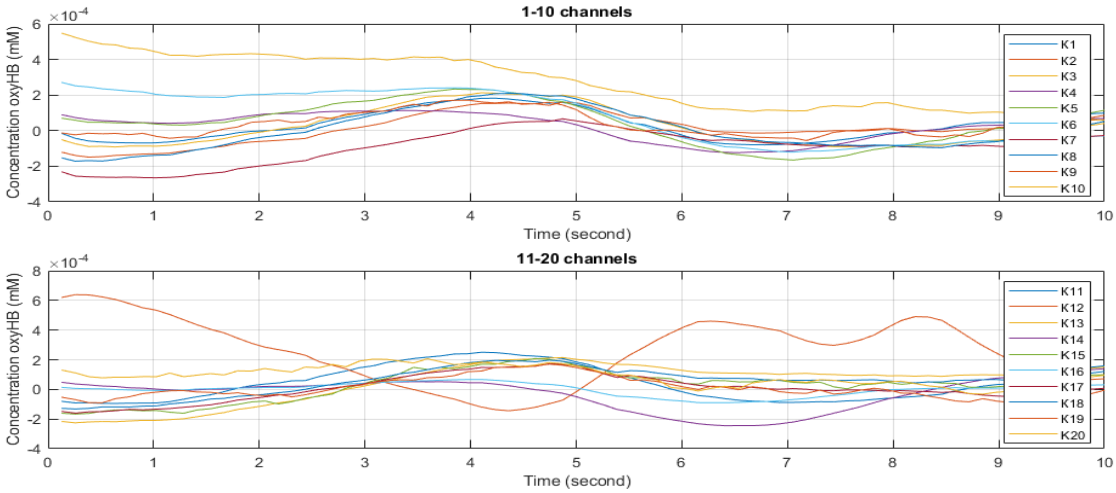

Рисунок 3.10 – Исходный и сглаженный сигнал всех 20 каналов

При обзоре данных было выявлено, что они имеют значительный разброс значений. Методы обработки сигнала чувствительные к шкалированию данных. Для того, чтобы избежать негативных последствий, необходимо произвести нормализацию данных. Для нормализации данных

гемодинамической активности мозга был выбран метод minmax нормализации.

Данный метод был реализован при помощи языка программирования Python с использованием библиотеки scikit-learn (листинг 2). Библиотека Scikit-learn является просты и эффективным инструментом сбора и обработки данных, находится в свободном доступе.

После нормализации значения «сырых данных» оказались в диапазоне [-1; 1].

```
def minmax_scale(X, feature_range=(-1, 1), axis=0, copy=True):
      X = check_array(X, copy=False, ensure_2d=False,
                  dtype=FLOAT_DTYPES, force_all_finite='allow-nan')
   original_ndim = X.ndim
   if original_ndim == 1:
      X = X.reshape(X.shape[0], 1)
   s = MinMaxScaler(feature_range=feature_range, copy=copy)
   if axis == 0:
      X = s.fit_transform(X)
   else:
      X = s.fit_transform(X.T).T
   if original_ndim == 1:
      X = X.ravel()
   return X
data=data.minmax_scale(data [:, 0])
```

Листинг 2 – Функция minmax нормализации

После экспорта полученных в результате проведения эксперимента данных активности мозга при движении кисти руки была проведена их предварительная обработка. Обработка включает в себя применение фильтров для сглаживания сигнала и нормализации значений исходных данных. Применены фильтры скользящего среднего и minmax нормализации при помощи языка программирования Python и библиотек Pandas и Scikit-learn.

3.3 Разработка модели машинного обучения в виде нейронной сети глубокого обучения

Из полученных данных, обработки и нормализации, был сформирован обучающий набор данных. Набор данных был разделен на 2 выборки в соотношении 80/20: (X_train, Y_train) –обучающая выборка и (X_test, Y_test) – тестовая выборка (листинг 3).

```
values1 = (data[:, 0])
y = (data[:, 1])
X_train, X_test, y_train, y_test = train_test_split(values1, y, test_size=0.2)
```

Листинг 3 – Формирование обучающей и тестовой выборки

Для распознавания паттернов движения кисти руки была разработана рекуррентная нейронная сеть LSTM. Разработанная нейронная сети была реализована при помощи языка программирования Python с использованием библиотеки Keras (листинг 4).

Keras является API для создания моделей глубокого обучения. Благодаря своей модульности позволяет удобно создавать прототипы, поддерживает сверточные и рекуррентные нейронные сети, работает как на обычных процессорах, так и на графических процессорах.

```
model = Sequential()
model.add(Embedding(20, 20, input_length=1))
model.add(LSTM(80, activation='ReLu')
model.add(Dropout(0.2))
model.add(Dense(5))
model.add(Activation('softmax'))
```

Листинг 4 – Создание нейронной сети глубокого обучения

Реализованная нейронная сеть представлена на рисунке 3.11. Нейронная сеть состоит из 5 слоев:

1) входной слой - состоит из 20 нейронов, принимая на вход данные с каждого канала;

2) LSTM-слой - реализует долгую краткосрочную память, 80 нейронов;

3) полносвязный слой – предназначен для классификации, 3 нейрона;

4) слой softmax - предназначенный для усиления уверенности в результатах классификации полносвязного слоя, 3 нейрона;

5) выходной слой сети.

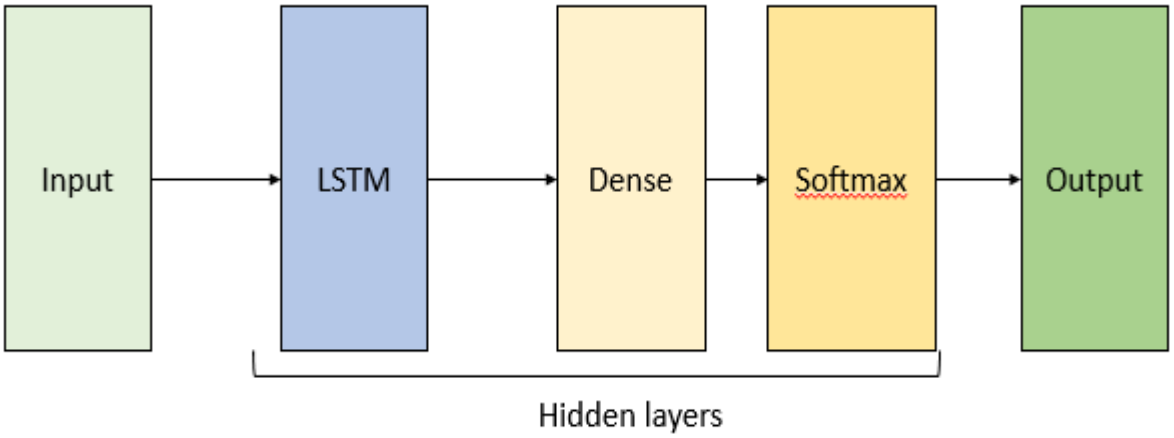

Рисунок 3.11 – Архитектура нейронной сети

Готовую нейронную сеть глубокого обучения обучили на обучающей выборке данных (X_train, Y_train), а затем проверили точность классификации при помощи тестовой выборки (X_test, Y_test) (листинг 5).

```
model.compile(loss='categorical_crossentropy',optimizer='rmsprop', metrics='categorical_accuracy')
model.fit(X_train, y_train, nb_epoch=11 , batch_size=34)
score = model.evaluate(X_test, y_test, batch_size=34)
print(score)
model_pred = model.predict(X_test)
print(accuracy_score(y_test, model_pred))
```

Листинг 5 – Обучение и тестирование модели

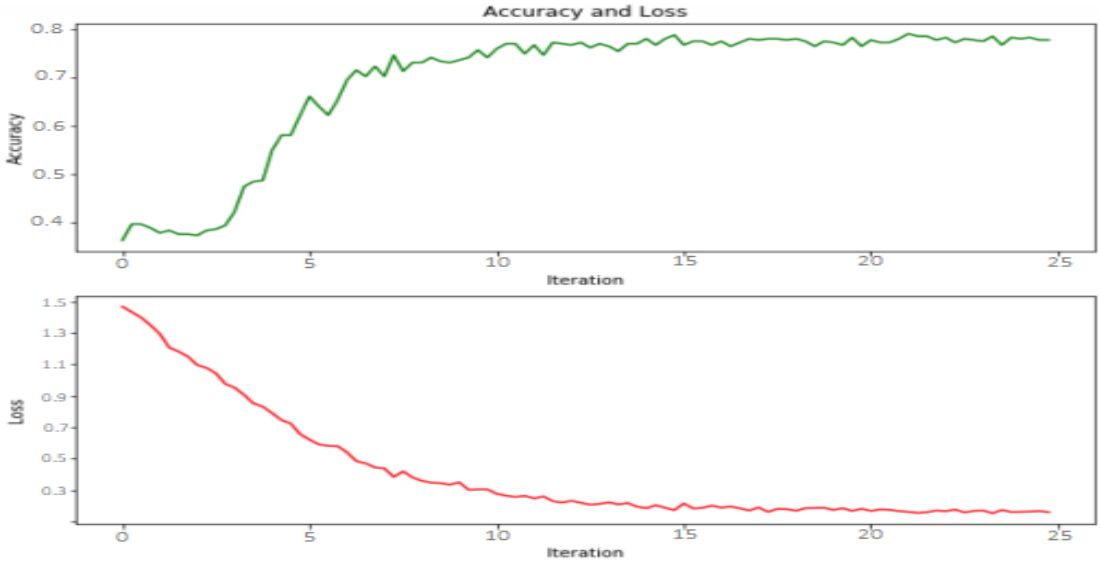

Рисунок 3.12 – График обучения нейронной сети

График точности обучения и тестирования модели представлен на рисунке 3.12. При тестировании модели, были получены результаты с точностью классификации равные 78%.

Используя язык программирования Python, библиотеки Keras и Scikit-learn была сформирована обучающая и тестовая выборки и разработана архитектура рекуррентной нейронной сети глубокого обучения. Разработанная нейронная сеть состоит из 5 слоев, 3 из которых являются скрытыми. Нейронная сеть обучена и протестирована на подготовленных наборах данных. Разработанная нейронная сеть позволила классифицировать движения кисти руки с точностью 78%.

ЗАКЛЮЧЕНИЕ

Передовым направлением научных исследований на сегодняшний день являются нейронауки. Нейронауки аккумулируют большой объем данных регистрируемой активности головного мозга человека. Для обработки столь масштабного объема данный необходимо использовать современные инструменты анализа данных. При использовании новых технологий регистрации мозговой активности, устаревшие методы обработки данных зачастую не справляются со своей задачей. Одной из главных задач в области нейронаук является получение сведений о различных состояниях человека на основании обработки его мозговой активности. Поэтому для получения качественных результатов исследований необходимо постоянно совершенствовать имеющиеся инструменты обработки данных.

В настоящее время российских инструментов обработки данных мозговой активности крайне мало, основными инструментами работы с такими данными являются зарубежные программные продукты. Однако ввиду некоторых ограничений, многие компании вынуждены отказываться от использования зарубежного программного продукта. Данный факт является стимулом для разработки отечественных программ для работы с данными мозговой активности.

В ходе выполнения выпускной квалификационной работы выполнялись задачи, необходимые для достижения поставленной цели - повышение эффективности процесса классификации мысленно воображаемых движений посредством нейрокомпьютерного интерфейса.

Проведено исследование существующих способов получения сигнала мозговой активности, рассмотрены технологии, основанные на электроэнцефалографии, функциональной спектроскопии в ближнем инфракрасном свете, магнито-резонансное сканирование. Перечисленные технологии являются неинвазивными способами регистрации активности мозга. Также исследована технология, основанная на внедрении нейрохирургических имплантатов, которая относится к инвазивному способу регистрации мозговой активности.

Изучены современные методы анализа и обработки данных, которые способны работать с большим объемом данных и имеют высокие показатели при применении в смежных областях. Рассмотрены методы деревьев решений, машина опорных векторов, сверточные нейронные сети, рекуррентные нейронные сети, метод градиентного бустинга, метод анализа главных компонент.

В ходе исследования методов получения и обработки данных мозговой активности выявлены достоинства и недостатки каждого из методов.

Для разработки информационного обеспечения анализа данных активности мозга, проанализировав результаты исследования, был выбран способ получения данных мозговой активности, основанный на использовании функциональной спектроскопии в ближнем инфракрасном свете (fNIRS), а для анализа и классификации данных был выбран метод использующий рекуррентные нейронные сети.

Используя оборудование NIRSport Model 88 и программу NIRSLab получены данные гемодинамической активности мозга 9 испытуемых, данные были отфильтрованы и нормализированы, что позволило сформировать обучающий набор данных. Полученные данные разделили на обучающую выборку и на тестовую выборку. Реализована и обучена на обучающей выборке модель машинного обучения в виде рекуррентной нейронной сети глубокого обучения. Проверка модели на корректность классификации на тестовой выборке показала точность в 78%, что является очень высоким результатом.

В результате выполнения выпускной квалификационной работы было получено информационное обеспечение в виде модели машинного обучения с высокой точностью классификации.

В дальнейшем разработанное информационное обеспечение возможно интегрировать с бионическим протезом кисти руки и разработав нейрокомпьютерный интерфейс управлять протезом «силой мысли».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Электроэнцефалография [Электронный ресурс] / Электрон. текстовые дан – URL: http://cnsinfo.ru/encyclopaedia/diagnostics/eeg/ (дата обращения 1.10.2018).

2. Schmidhuber, J. Deep learning in neural networks: an overview [Текст] / J. Schmidhuber // Neural Networks. - 2015. - № 61. -С. 85-117.

3. Grossberg, S. Towards solving the hard problem of consciousness: the varieties of brain resonances and the conscious experiences that they support [Текст]/ S. Grossberg // Neural Networks. - 2017. - № 87. - С. 38-95.

4. Каплан, А.Я. Изучение возможности управления отдельными пальцами фантома кисти руки человека в контуре интерфейса мозг-компьютер на волне P300 [Текст]/ Каплан А.Я., Жигулъская Д.Д., Кирьянов Д.А. // Вестник Российского государственного медицинского университета. - 2017. - Т. 2. - С. 26-33.

5. Будко, Р.Ю. Создание классификатора мимических движений на основе анализа электромиограммы[Текст]/ Будко Р.Ю., Старченко И.Б // Труды СПИИРАН – 2016 – № 46 – С. 76-89.

6. Pchelintseva, S.V. Recognition and classification of oscillatory patterns of electric brain activity using artificial neural network approach [Текст] / Pchelintseva S.V., Runnova A.E., Musatov V.Y., Hramov A.E. // Proc. SPIE. - 2017. - № 10063. - DOI: 10.1117/12.2250001

7. Афонин, А.Н Разработка и реализация макета бионического протеза кисти руки [Текст] / Афонин А.Н., Алейников А.Ю., Гладышев А.Р., Попова А.В. // Робототехника и техническая кибернетика. – 2016. - № 3(12). - С. 68-71.

8. Станкевич, Л.А. Классификация электроэнцефалографических паттернов воображаемых движений пальцами руки для разработки интерфейса мозг-компьютер [Текст] / Станкевич Л.А., Сонькин К.М., Нагорнова Ж.В. и др. // Труды СПИИРАН. – 2015. - № 3. - С. 163-182.

9. Maksimenko, V.A. Nonlinear analysis of brain activity, associated with motor action and motor imaginary in untrained subjects [Текст] / Maksimenko V.A., Pavlov A.N., Hramov A.E. et al. // Nonlinear Dynamics. - 2018. - № 91. - С. 2803.

10. Ситникова, М.А. Функциональная оптическая томография: надежный метод измерения мозговой активации в процессе решения различных математических задач [Электронный ресурс] / Ситникова М.А., Нюрк Г.Х. Современные проблемы науки и образования. – Электрон. журн.– Москва: Академия Естествознания, 2016. - URL: http://www.science-education.ru/ru/article/view?id=24640 (дата обращения: 05.10.2018).

11. Ferrari, M. A brief review on the history of human functional near-infrared spectroscopy (fNIRS) development and fields of application [Текст] / Ferrari M., Quaresima V. // Neuroimage. - 2012. - № 63. – С. 921–935.

12. Naseer, N. Classification of functional near-infrared spectroscopy signals corresponding to the right- and left-wrist motor imagery for development of a brain-computer interfaces [Текст] / Naseer N.; Hong K.-S. // Neuroscience Letters. – 2013. - № 553. - С. 84–89.

13. Robinson, N. Real-Time Subject-Independent Pattern Classification of Overt and Covert Movements from fNIRS Signals [Текст] / Robinson N., Zaidi A.D., Rana M. et al. // PloS ONE. – 2016. - № 11. – С. 7.

14. Календер, В. Компьютерная томография. Основы, техника, качество изображений и области клинического использования [Текст] / Календер, В. – Москва: Техносфера, 2010. - 344 с.

15. Scheinberg, K. An Efficient Implementation of an Active Set Method for SVMs [Текст] / K. Scheinberg. // The Journal of Machine Learning Research. - 2006. - № 7. - С. 2237-2257.

16. Cherkassky, V. Practical selection of SVM parameters and noise estimation for SVM regression [Текст] / V. Cherkassky, Yunqian Ma. // Neural Networks. - 2004. - № 17. - С. 113-126.

17. Chuanhou Gao. A comparative analysis of support vector machines and extreme learning machines [Текст] / Xueyi Liu, Chuanhou Gao, Ping Li. // Neural Networks. - 2012. - № 33. - С. 58-66.

18. Horata, P. Robust extreme learning machine [Текст] / Punyaphol Horata, Sirapat Chiewchanwattana Khamron Sunat. // Neurocomputing. - 2013. - № 102. - С. 31-44.

19. Звёздочкина, Н.В. Исследование электрической активности головного мозга [Текст] / Н.В.Звёздочкина. – Казань: Казан. ун-т, 2016. – 59 с.

20. Рупнова, М.Ю. Классификация паттернов двигательной активности на ЭЭГ-данных [Текст] / Рупнова М.Ю., Мусатов В.Ю., Пчелинцева С.В. // Вестник. - 2017. - № 8. – С. 186.

21. Галушкин, А.И. Нейронные сети: основы теории. [Текст] / А.И. Галушкин. - Москва: РиС, 2015. - 496 с.

22. Дизайн нейронной сети [Текст] / Мартин Т. Хейган, Говард Б. Демут, Марк Х. Бил, Орландо де Хесус. - Москва: Аист, 2017. – 1012с.

23. Ожиганов, И. Классификация сигналов головного мозга для нейрокомпьютерного интерфейса [Электронный ресурс] / И. Ожиганов – Электрон. текстовые дан. – 2015. - URL: http://www.azoft.ru/blog/ klassifikaciya-signalov-golovnogo-mozga-dlya-nejrokompyuternogo-interfejsa/ (дата обращения 18.04.2019).

24. Функциональная спектроскопия в ближнем инфракрасном свете [Электронный ресурс] / Электрон. текстовые дан – URL: http://usabilityin.ru/fnirs/ (дата обращения 10.10.2018).

25. Федюкович, Н.И. Анатомия и физиология: учебное пособие [Текст] /Н.И. Федюкович. - Ростов-н/Д.: «Феникс», 2010. - 416с.

26. Ткаченко, Б.И. Основы физиологии человека [Текст] / Б.И. Ткаченко. - СПб: Международный фонд истории науки, 2004. - 505 с.

27. Александров, Ю.И. Регистрация импульсной активности нервных клеток [Электронный ресурс] / Ю.И. Александров. – Электрон. текстовые дан.

– Москва: Инфра-М, 2015 – URL: https://pedlib.ru/Books/3/0337/ (дата обращения 21.04.2019).

28. Yong, Xu. NIRSLab User’s manual [Текст] / Yong Xu, Harry Graber, Randall Barbour // Germany – 2017. – 234 с.

29. Soltanlou, M. Reduction but no shift in brain activation after arithmetic learning in children: a simultaneous fNIRS-EEG study [Текст] / Mojtaba Soltanlou, Christina Artemenko, Ann-Christine Ehlis, Stefan Huber, Andreas J Fallgatter, Thomas Dresler, Hans-Christoph Nuerk. // Scientific reports. – 2018. – № 8. – С.1701.

30. Миркин, Б.Г. Введение в анализ данных [Текст] / Б.Г. Миркин. – Москва: Юрайт, 2015. – 178 с.

31. Мхитарян, В.С. Анализ данных [Текст] / Мхитарян В.С. - Москва: Юрайт, 2017. – 492 с.

32. Загоруйко, М.Г. Прикладные методы анализа данных и знаний [Текст] / М.Г. Загоруйко. - Новосибирск: ИМ СО РАН, 2009. — 270 с.

33. Мостлер, Ф. Анализ данных и регрессия [Текст] / Ф. Мостлер, Д. Тьюки. – Москва: ФиС, 2011. – 570 с.

34. Уэс, М. Python и анализ данных [Текст] / М. Уэс. – Москва: ДМК Пресс, 2015. – 482 с.

35. Ромальо, Л. Python. К вершинам мастерства [Текст] / Л. Ромальо. – Москва: ДМК Пресс, 2016. – 768 с.

36. Флах, П. Машинное обучение [Текст] / П. Флах. – Москва: ДМК Пресс, 2015. – 400 с.

37. Рашка, С. Python и машинное обучение [Текст] / С. Рашка. – Москва: ДМК Пресс, 2017. – 418 с.

38. Феверолф, М. Машинное обучение [Текст] / М. Феверолф, Д. Ричардс, Х. Бринк. – Санкт-Петербург: Питер, 2017. – 465 с.

39. Марманис, Х. Алгоритмы интеллектуального Интернета [Текст] / Х. Марманис, Д. Бабенко. – Санкт-Петербург: Символ-Плюс, 2011. – 480 с.

40. Мерков, А.Б. Распознавание образов: введение в методы статистического обучения [Текст] / А.Б. Мерков. – Москва: Urss, 2011. – 256 с.

41. Гелиг, А.Х. Введение в математическую теорию обучаемых распознающих систем и нейронных сетей [Текст] / А.Х. Гелиг, А.С. Матвеев. – Санкт-Петербург: издательство СПбГУ, 2015. – 224 с.

42. Мерков, А.Б. Распознавание образов: построение и обучение вероятностных моделей [Текст] / А.Б. Мерков. – Москва: Urss, 2014. – 240 с.

43. Ричарт, В. Построение систем машинного обучения на языке Python [Текст] / В. Ричарт, П.Л. Коэльо. – Москва: ДМК Пресс, 2015. – 302 с.

44. Мюллер, А. Введение в машинное обучение с помощью Python [Текст] / А. Мюллер, С. Гвидо. – Москва: Вильямс, 2017. – 480 с.