**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc197108684)

[1. СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ 7](#_Toc197108685)

[1.1 Описание предметной области 7](#_Toc197108686)

[1.2 Анализ существующих решений 8](#_Toc197108687)

[1.3 Основы структуры человеческого глаза 9](#_Toc197108688)

[1.5 Основы тонометрии 10](#_Toc197108689)

[1.5 Обоснование необходимости реализации 10](#_Toc197108690)

[1.6 Постановка задачи 11](#_Toc197108691)

[2. ПРОЕКТИРОВАНИЕ АЛГОРИТМА 13](#_Toc197108692)

[2.1 Анализ инструментов для реализации 13](#_Toc197108693)

[2.2 Обзор набора данных тонометрии 14](#_Toc197108694)

[2.3 Проектирование архитектуры нейронных сетей 15](#_Toc197108695)

ВВЕДЕНИЕ

Глаукома представляет собой одну из ведущих причин необратимой слепоты во всем мире, что делает своевременную диагностику и лечение этого заболевания критически важными задачами современной медицины. Это хроническое заболевание глаз характеризуется повышенным внутриглазным давлением, которое со временем может привести к повреждению зрительного нерва и потере зрения. Несмотря на значительный прогресс в области офтальмологии, ранняя диагностика глаукомы остается сложной задачей, поскольку на начальных стадиях она часто протекает бессимптомно. Традиционные методы диагностики, такие как тонометрия, позволяют измерять внутриглазное давление, однако их точность и способность выявлять заболевание на ранних стадиях зависят от опыта врача и качества интерпретации данных.

С развитием технологий искусственного интеллекта и глубокого обучения открываются новые возможности для автоматизации и повышения точности диагностических процессов. Использование нейронных сетей для анализа медицинских изображений и данных тонометрии демонстрирует многообещающие результаты в выявлении патологий глаза, включая глаукому. Такие подходы позволяют не только ускорить процесс диагностики, но и снизить вероятность человеческой ошибки, что особенно важно в условиях ограниченного доступа к квалифицированным специалистам. В данном контексте разработка системы классификации глаукомы на основе глубокого обучения становится актуальной задачей, способной внести значительный вклад в улучшение качества медицинской помощи.

Целью данной дипломной работы является разработка и реализация модели глубокого обучения для классификации глаукомы на основе данных тонометрии глаза с использованием доступного набора данных. В работе будет рассмотрена предметная область, проведен анализ существующих решений, а также спроектирована и обучена нейронная сеть, способная эффективно различать здоровые глаза от глаз с признаками глаукомы. Особое внимание будет уделено выбору подходящих инструментов, обработке данных и оптимизации модели для достижения высокой точности классификации.

Предложенная разработка основывается на наборе данных "Glaucoma Dataset EyePACS AIROGS Light v2", доступном на платформе Kaggle. Этот набор включает изображения глазного дна, которые будут использованы для обучения и тестирования модели. В процессе работы будут применены современные методы обработки изображений и архитектуры нейронных сетей, такие как MobileNet V3, что позволит создать эффективное решение, пригодное для практического применения. Итогом исследования станет обученная модель, сохраненная для дальнейшего использования, а также анализ ее производительности на тестовых данных.

Таким образом, данная работа направлена на решение актуальной проблемы ранней диагностики глаукомы с использованием передовых технологий машинного обучения. Результаты исследования могут послужить основой для создания автоматизированных систем поддержки принятия решений в офтальмологии, что в перспективе улучшит доступность и качество диагностики этого серьезного заболевания. В следующих разделах будут подробно рассмотрены теоретические основы, методология разработки и практическая реализация поставленной задачи.

1. СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ

1.1 Описание предметной области

Предметная область данной дипломной работы охватывает сферу офтальмологии, а именно диагностику глаукомы — одного из наиболее распространенных и опасных заболеваний глаз, которое может привести к необратимой потере зрения. Глаукома представляет собой группу хронических заболеваний, основным признаком которых является повышение внутриглазного давления (ВГД), вызывающее постепенное повреждение зрительного нерва. Этот процесс, как правило, протекает медленно и на ранних стадиях не сопровождается явными симптомами, что значительно затрудняет своевременное выявление болезни. По данным Всемирной организации здравоохранения, глаукома занимает второе место среди причин слепоты в мире, уступая лишь катаракте, и затрагивает миллионы людей, особенно пожилого возраста. Таким образом, разработка методов ранней диагностики этого заболевания является одной из приоритетных задач современной медицины.

В основе диагностики глаукомы лежит измерение внутриглазного давления с помощью тонометрии — метода, который позволяет оценить давление жидкости внутри глазного яблока. Тонометрия может проводиться различными способами, включая контактные и бесконтактные техники, и часто дополняется анализом изображений глазного дна, где можно наблюдать изменения зрительного нерва и сосудистой системы. Однако традиционные подходы к интерпретации данных тонометрии и визуального осмотра имеют свои ограничения: они требуют высокой квалификации врача, а также могут быть подвержены субъективным ошибкам. Кроме того, доступ к специализированным офтальмологическим услугам ограничен в некоторых регионах, что подчеркивает необходимость разработки автоматизированных и доступных решений.

С развитием технологий искусственного интеллекта (ИИ) и глубокого обучения (Deep Learning, DL) в медицинской практике появляется возможность значительно улучшить диагностические процессы. Глубокое обучение, основанное на использовании многослойных нейронных сетей, демонстрирует выдающиеся результаты в задачах анализа изображений и классификации данных. В контексте офтальмологии такие технологии уже успешно применяются для выявления таких заболеваний, как диабетическая ретинопатия и возрастная макулярная дегенерация. Применение аналогичных методов для диагностики глаукомы открывает перспективы для создания систем, которые смогут автоматически анализировать данные тонометрии и изображения глазного дна, предоставляя врачам надежные инструменты для принятия решений.

Предметная область данной работы также включает изучение особенностей наборов данных, используемых для обучения моделей глубокого обучения. В частности, в исследовании будет задействован набор данных "Glaucoma Dataset EyePACS AIROGS Light v2", содержащий изображения глазного дна с разметкой, указывающей наличие или отсутствие признаков глаукомы. Эти данные представляют собой ценный ресурс для разработки и тестирования алгоритмов классификации, поскольку они отражают реальные клинические случаи. Важным аспектом является также предварительная обработка данных, включая аугментацию изображений и нормализацию, что позволяет повысить устойчивость модели к вариациям в качестве изображений и условиям их получения.

1.2 Анализ существующих решений

На сегодняшний день автоматизированные методы диагностики глаукомы активно развиваются, и существует несколько подходов к обработке медицинских изображений и данных тонометрии. Основные направления исследований включают традиционные методы обработки изображений, машинное обучение и глубокие нейронные сети.

**Традиционные методы анализа**

Классические методы диагностики глаукомы основаны на анализе изображений глазного дна и параметров внутриглазного давления. Для этого применяются различные алгоритмы обработки изображений, такие как:

* Выделение контуров и анализ формы зрительного нерва;
* Сегментация слоя нервных волокон сетчатки;
* Оценка толщины роговицы и ее влияния на показания тонометрии.

Эти методы позволяют выявлять патологические изменения, но требуют значительных вычислительных ресурсов и высокой квалификации врача для интерпретации результатов.

**Методы машинного обучения**

В последние годы получили развитие алгоритмы машинного обучения, такие как:

* SVM (метод опорных векторов);
* KNN (метод ближайших соседей);
* Решающие деревья и случайные леса.

Эти алгоритмы используют вручную отобранные признаки изображений глазного дна и параметры тонометрии для обучения модели, которая затем классифицирует пациентов на группы с высоким и низким риском глаукомы.

**Глубокое обучение**

С появлением глубоких нейронных сетей значительно улучшились результаты автоматической диагностики. Используются модели, такие как:

* CNN (сверточные нейронные сети) для анализа изображений глазного дна;
* RNN и трансформеры для обработки последовательностей данных;
* Гибридные модели, комбинирующие изображения и числовые данные тонометрии.

В частности, архитектуры ResNet, MobileNet и EfficientNet показывают высокую точность в задачах классификации глаукомы. Они способны автоматически выделять важные признаки из изображений и анализировать их с минимальным вмешательством человека.

**Сравнительный анализ**

Исследования показывают, что глубокие нейронные сети превосходят традиционные методы и алгоритмы машинного обучения по точности диагностики глаукомы. Однако их применение требует больших объемов размеченных данных и вычислительных мощностей. Кроме того, важными факторами остаются интерпретируемость моделей и их клиническая применимость.

Таким образом, использование глубокого обучения для автоматической классификации глаукомы является перспективным направлением, позволяющим повысить точность диагностики и снизить зависимость от субъективного фактора врача.

1.3 Основы структуры человеческого глаза

Понимание структуры человеческого глаза является фундаментальной основой для разработки методов диагностики глаукомы, поскольку это заболевание непосредственно связано с анатомическими и физиологическими особенностями органа зрения. Человеческий глаз представляет собой сложную оптическую систему, которая обеспечивает восприятие света, его преобразование в нервные импульсы и передачу информации в мозг. Знание строения глаза позволяет лучше интерпретировать данные, получаемые в ходе тонометрии и анализа изображений глазного дна, а также понять механизмы развития глаукомы. В данном разделе будут рассмотрены ключевые элементы структуры глаза, их функции и связь с патологическими процессами, характерными для этого заболевания.

Человеческий глаз имеет форму шара диаметром около 24–25 мм и состоит из нескольких основных структурных компонентов. Внешний слой глаза, называемый склерой, представляет собой плотную соединительную ткань, которая защищает внутренние элементы и поддерживает форму органа. Передняя часть склеры переходит в прозрачную роговицу, которая выполняет функцию первичной линзы, преломляющей световые лучи. Роговица играет важную роль в измерении внутриглазного давления, поскольку ее толщина и кривизна могут влиять на результаты тонометрии. За роговицей расположена передняя камера глаза, заполненная водянистой влагой — прозрачной жидкостью, которая поддерживает давление внутри глаза и питает окружающие ткани.

Следующий важный элемент — радужка, которая регулирует количество света, поступающего в глаз, за счет изменения размера зрачка. За радужкой находится хрусталик — прозрачное тело, способное менять свою форму для фокусировки изображения на сетчатке. Водянистая влага, циркулирующая между передней и задней камерами глаза, оттекает через трабекулярную сеть — структуру, расположенную в углу передней камеры. Нарушение оттока этой жидкости является одной из основных причин повышения внутриглазного давления при глаукоме, что делает трабекулярную сеть ключевым объектом изучения в контексте данного заболевания. Повышенное давление оказывает воздействие на заднюю часть глаза, где расположены сетчатка и зрительный нерв.

Сетчатка представляет собой внутренний слой глаза, состоящий из фоторецепторных клеток (палочек и колбочек), которые преобразуют свет в электрические сигналы. Эти сигналы передаются через зрительный нерв в мозг для дальнейшей обработки. Зрительный нерв выходит из глаза через диск зрительного нерва — область, видимую на снимках глазного дна. При глаукоме повышенное внутриглазное давление может сдавливать и повреждать волокна зрительного нерва, что приводит к его атрофии и постепенной утрате поля зрения. Именно изменения в структуре диска зрительного нерва и слоя нервных волокон сетчатки являются важными диагностическими признаками, которые можно выявить с помощью изображений глазного дна.

Глазное яблоко также окружено стекловидным телом — гелеобразной субстанцией, которая заполняет пространство между хрусталиком и сетчаткой, обеспечивая поддержку формы глаза. Хотя стекловидное тело непосредственно не связано с развитием глаукомы, его состояние может влиять на качество изображений, используемых для диагностики. Кроме того, кровеносные сосуды, питающие сетчатку, также видны на снимках глазного дна и могут служить дополнительным индикатором патологических изменений, таких как сужение или аномалии, связанные с глаукомой.

1.5 Основы тонометрии

**Тонометрия** – это метод измерения внутриглазного давления (ВГД), который играет ключевую роль в диагностике и мониторинге глаукомы. ВГД определяется балансом между выработкой и оттоком внутриглазной жидкости и является важным показателем здоровья глаз.

**Нормальные значения внутриглазного давления**

Среднее нормальное значение ВГД составляет **10–21 мм рт. ст.** Отклонения от этих значений могут указывать на развитие глаукомы или других офтальмологических заболеваний.

**Методы тонометрии**

1. **Апланационная тонометрия (по Гольдману)**
   * Является «золотым стандартом» измерения ВГД.
   * Измерение основано на приложении давления к роговице для оценки силы, необходимой для ее уплощения.
   * Требует местного обезболивания и использования флуоресцентного красителя.
2. **Бесконтактная (пневматическая) тонометрия**
   * Использует воздушный поток для сплющивания роговицы.
   * Не требует обезболивания, но может быть менее точной по сравнению с апланационной тонометрией.
3. **Динамическая контурная тонометрия**
   * Учитывает индивидуальные характеристики роговицы, такие как ее толщина и эластичность.
   * Позволяет получить более точные измерения при измененных свойствах роговицы.
4. **Импрессионная тонометрия (по Маклакову)**
   * Основана на измерении глубины вдавливания специального грузика в роговицу.
   * Используется преимущественно в странах СНГ и требует местного обезболивания.
5. **Электронная тонометрия**
   * Современные портативные электронные тонометры обеспечивают быструю и точную оценку ВГД.

**Факторы, влияющие на точность измерений**

* **Толщина роговицы** – более толстая роговица может завышать показатели ВГД, а более тонкая – занижать.
* **Время суток** – ВГД может колебаться в течение дня, достигая пика утром.
* **Положение тела** – в лежачем положении ВГД обычно выше, чем в сидячем.

Тонометрия является одним из важнейших методов в офтальмологии и широко применяется для раннего выявления глаукомы. Однако для более точной диагностики ее результаты необходимо анализировать в совокупности с другими методами исследования.

1.5 Обоснование необходимости реализации

Глаукома является одной из ведущих причин необратимой слепоты во всем мире. По данным Всемирной организации здравоохранения (ВОЗ), миллионы людей ежегодно теряют зрение из-за поздней диагностики этого заболевания. Основной причиной столь высокой заболеваемости и осложненного выявления глаукомы является бессимптомное течение на ранних стадиях, что затрудняет своевременное обнаружение патологии.

Одним из ключевых методов диагностики глаукомы является **тонометрия**, позволяющая измерять внутриглазное давление (ВГД) и выявлять отклонения от нормы. Однако существующие методы диагностики требуют высокой квалификации врача и могут зависеть от субъективных факторов, таких как интерпретация результатов офтальмологом, состояние пациента и технические особенности используемого оборудования. Это приводит к необходимости создания более точных, автоматизированных и доступных методов диагностики.

В последние годы развитие технологий искусственного интеллекта, в частности **глубокого обучения**, открыло новые возможности для анализа медицинских данных. Глубокие нейронные сети уже доказали свою эффективность в таких областях медицины, как радиология, дерматология и офтальмология. Их способность анализировать сложные паттерны в изображениях делает их перспективным инструментом для диагностики глаукомы на основе данных тонометрии.

Разработка автоматизированной модели диагностики глаукомы на основе глубокого обучения необходима по следующим причинам:

1. **Повышение точности диагностики**
   * Исключение субъективного человеческого фактора при интерпретации данных.
   * Использование современных архитектур нейросетей, способных выявлять закономерности, незаметные для человека.
2. **Ранняя диагностика заболевания**
   * Глубокие нейронные сети могут находить малозаметные признаки глаукомы, позволяя выявлять болезнь на ранних стадиях.
   * Своевременное лечение может значительно замедлить прогрессирование заболевания и сохранить зрение.
3. **Автоматизация и доступность диагностики**
   * Разработка модели, способной анализировать изображения глаз и показатели ВГД без необходимости присутствия офтальмолога.
   * Возможность интеграции технологии в медицинские центры и удаленные диагностические системы.
4. **Оптимизация временных и материальных затрат**
   * Уменьшение нагрузки на медицинский персонал за счет автоматического анализа изображений.
   * Снижение затрат на диагностику благодаря использованию программного анализа вместо дорогостоящих офтальмологических обследований.
5. **Развитие цифровой медицины**
   * Современные технологии позволяют объединять медицинские данные в облачные сервисы, что способствует лучшему мониторингу состояния пациентов.
   * Возможность интеграции разработанной модели в телемедицинские платформы для удаленной диагностики.

Таким образом, реализация модели глубокого обучения для автоматического анализа данных тонометрии глаза является важным и актуальным направлением исследования. Данный проект имеет потенциал значительно повысить точность и доступность диагностики глаукомы, что позволит своевременно выявлять заболевание и улучшить качество жизни пациентов.

1.6 Постановка задачи

Основной задачей данной работы является разработка и обучение модели глубокого обучения для автоматической классификации глаукомы на основе данных тонометрии глаза. Для решения этой задачи необходимо выполнить несколько ключевых этапов, которые включают сбор, предобработку и анализ данных, а также разработку и обучение нейросетевой модели. Важными аспектами являются:

1. **Предобработка и анализ данных**  
   Необходимо собрать и подготовить набор данных, включающий изображения глаз, полученные с использованием тонометрии, а также соответствующие метки, указывающие на наличие или отсутствие глаукомы. В процессе работы с данными нужно выполнить следующие шаги:
   * Провести очистку данных от шума и артефактов.
   * Применить методы аугментации для увеличения объема обучающих данных и повышения обобщающих способностей модели.
   * Подготовить данные для нормализации и стандартизации, чтобы улучшить обучение модели.
2. **Выбор и разработка архитектуры нейронной сети**  
   Необходимо выбрать подходящую архитектуру нейронной сети, способную эффективно классифицировать изображения. Для данной задачи предполагается использование современных моделей глубокого обучения, таких как сверточные нейронные сети (CNN), которые зарекомендовали себя в области обработки изображений. Важно, чтобы выбранная модель могла:
   * Извлекать признаки из изображений глаз, связанные с особенностями развития глаукомы.
   * Обрабатывать изображения разного качества и с различными уровнями шума.
   * Добиться высокой точности классификации при ограниченных вычислительных ресурсах.
3. **Обучение и настройка модели**  
   После выбора архитектуры сети необходимо обучить модель на подготовленных данных. В этом процессе ключевыми задачами являются:
   * Разработка оптимальной стратегии обучения, включая настройку гиперпараметров (например, скорости обучения, размера пакета и числа эпох).
   * Применение методов регуляризации, таких как dropout, для предотвращения переобучения.
   * Оценка производительности модели на валидационном наборе данных и корректировка параметров для достижения наилучших результатов.
4. **Оценка точности и эффективности модели**  
   После обучения модели необходимо провести оценку её точности на тестовом наборе данных. Это включает в себя:
   * Вычисление точности, полноты, F1-меры и других метрик для оценки качества классификации.
   * Анализ ошибок модели и возможных причин недочетов.
   * Сравнение полученных результатов с существующими методами диагностики глаукомы, чтобы выявить преимущества и недостатки разработанного подхода.
5. **Интерпретация и применение результатов**  
   На основе полученной модели необходимо сформулировать рекомендации по её применению в медицинской практике. Это может включать:
   * Разработку программного интерфейса для использования модели в реальных клинических условиях.
   * Оценку возможности интеграции модели в системы телемедицины для удаленной диагностики.
   * Рекомендации по дальнейшему улучшению модели, включая использование дополнительных данных и улучшение архитектуры сети.

Таким образом, постановка задачи заключается в создании эффективного инструмента для диагностики глаукомы на основе анализа данных тонометрии, что позволит улучшить доступность и точность диагностики этого заболевания, а также снизить зависимость от человеческого фактора в интерпретации результатов.

2. ПРОЕКТИРОВАНИЕ АЛГОРИТМА

2.1 Анализ инструментов для реализации

Для реализации задачи классификации изображений с использованием глубоких нейронных сетей были выбраны следующие инструменты и библиотеки:

1. **PyTorch** – это мощная библиотека для построения нейронных сетей, которая является основой для разработки и обучения моделей глубокого обучения. PyTorch предоставляет гибкие инструменты для создания, обучения и оптимизации нейросетевых моделей. В данном коде используются основные модули PyTorch, такие как:
   * torch: для работы с тензорами и математическими операциями.
   * torchvision: для работы с изображениями и стандартными предобученными моделями, такими как MobileNet.
   * torch.optim: для оптимизации модели, включая использование различных методов оптимизации (например, Adam).
   * torch.nn: для создания слоев и моделей нейронных сетей.
   * torch.utils.data: для работы с загрузкой и аугментацией данных.
2. **MobileNetV3** – это предварительно обученная модель, предоставленная библиотекой torchvision. Она используется в коде для создания основы модели классификации, благодаря своей легкости и хорошей производительности на мобильных устройствах и других устройствах с ограниченными вычислительными мощностями. В коде заменяется последний слой сети на новый с двумя выходами, что позволяет адаптировать модель для решения задачи бинарной классификации (наличие или отсутствие глаукомы).
3. **Аугментация данных** – для улучшения обучаемости модели используется кастомная аугментация данных, в частности:
   * **Добавление шума** (AddNoise) для увеличения устойчивости модели к шумам.
   * **Цветовая аугментация** с использованием ColorJitter, чтобы модель могла быть более устойчивой к различным цветовым условиям на изображениях.
   * **Геометрическая аугментация** с помощью операций случайного вращения, афинных преобразований и случайных переворотов, что помогает повысить обобщающую способность модели и избегать переобучения.
4. **Обработка данных** – используется стандартная практика предобработки изображений:
   * Изменение размера изображений до 512x512 пикселей с помощью transforms.Resize.
   * Нормализация данных, с использованием заранее рассчитанных средних и стандартных отклонений для изображений, таких как те, что использовались при обучении предварительно обученной модели MobileNetV3.
5. **Использование прогресса в обучении** – в коде используется tqdm, чтобы визуализировать процесс обучения с прогрессом, что помогает отслеживать процесс на каждом этапе обучения и проверки.
6. **Оптимизация** – используется оптимизатор Adam для обновления весов модели. Это популярный метод оптимизации для задач глубокого обучения, который автоматически корректирует шаги обновления весов для каждого параметра, улучшая обучение.
7. **Обучение и тестирование модели** – обучение модели производится с использованием двух фаз:
   * **Обучение (train)** – в процессе обучения модель обновляет свои веса.
   * **Проверка (validation)** – на этой фазе модель оценивает свою работу на валидационном наборе, и происходит выбор наилучшей модели на основе точности.
8. **Параметры обучения**:
   * Модель обучается в течение 6 эпох с использованием функции потерь CrossEntropyLoss, подходящей для задач классификации.
   * Для корректировки скорости обучения используется StepLR scheduler, который уменьшает скорость обучения на 10% каждые 3 эпохи.
9. **Тестирование модели** – после завершения обучения, модель тестируется на отдельном тестовом наборе данных, чтобы оценить её эффективность на невиданных данных.

**Преимущества выбранных инструментов и подходов:**

* **PyTorch** и **MobileNetV3** обеспечивают высокую гибкость и скорость разработки.
* Использование предварительно обученной модели **MobileNetV3** позволяет значительно сократить время на обучение и достичь хороших результатов с минимальными затратами.
* Аугментация данных помогает избежать переобучения и повышает обобщающую способность модели.
* Применение эффективных методов оптимизации (например, Adam) и регуляризации (например, scheduler для LR) позволяет добиться хороших результатов с минимальными вычислительными затратами.

2.2 Обзор набора данных тонометрии

Датасет EyePACS-AIROGS-light-v2 представляет собой улучшенную и стандартизированную версию набора изображений глазного дна, предназначенную для обучения моделей машинного обучения для диагностики глаукомы. Он основан на подмножестве изображений из оригинального набора Rotterdam EyePACS AIROGS и включает следующие особенности:

* **Размер изображений**: увеличены с 256x256 до 512x512 пикселей.
* **Формат изображений**: изменен с JPG на PNG для повышения качества.
* **Размеры выборок**:
  + Обучающая выборка: 4000 изображений (~84%).
  + Валидационная выборка: 385 изображений (~8%).
  + Тестовая выборка: 385 изображений (~8%).
* **Классы изображений**: разделены на две категории: referable glaucoma (RG) и non-referable glaucoma (NRG).

Для обеспечения высокого качества разметки, изображения были оценены группой из 20 специалистов, включая офтальмологов и оптометристов, с использованием веб-системы аннотирования. Несогласованные оценки были пересмотрены опытными специалистами для достижения точности диагностики.

Данный датасет способствует разработке и оценке алгоритмов для автоматического обнаружения глаукомы, обеспечивая стандартизированные и высококачественные данные для обучения моделей машинного обучения.

2.3 Проектирование архитектуры нейронных сетей

Проектирование архитектуры нейронных сетей является ключевым этапом в разработке системы классификации глаукомы на основе данных тонометрии и изображений глазного дна. Этот процесс включает выбор подходящей модели, настройку ее параметров, определение методов предобработки данных и стратегии обучения, чтобы обеспечить высокую точность и устойчивость системы к реальным условиям. В рамках данной дипломной работы была выбрана архитектура MobileNet V3 Large, которая адаптирована для задачи бинарной классификации (глаукома/нет глаукомы) с использованием набора данных "Glaucoma Dataset EyePACS AIROGS Light v2". В этом разделе подробно описывается процесс проектирования, включая обоснование выбора модели, этапы подготовки данных, модификацию архитектуры и реализацию процесса обучения, а также анализ факторов, влияющих на производительность системы.

**Обоснование выбора архитектуры**

Выбор архитектуры нейронной сети для классификации глаукомы основывался на нескольких критериях: вычислительная эффективность, способность к обобщению, а также успешное применение в задачах анализа медицинских изображений. MobileNet V3 Large, разработанная компанией Google, представляет собой сверточную нейронную сеть (CNN), оптимизированную для работы на устройствах с ограниченными вычислительными ресурсами, таких как мобильные устройства, при сохранении высокой точности классификации. Эта модель сочетает в себе передовые подходы, такие как инвертированные остаточные блоки (Inverted Residuals) и механизм внимания (Squeeze-and-Excitation), что делает ее подходящей для обработки сложных изображений глазного дна. По сравнению с более тяжелыми архитектурами, такими как ResNet или EfficientNet, MobileNet V3 обеспечивает баланс между скоростью обработки и качеством предсказаний, что важно для потенциального практического применения в медицинских системах.

Кроме того, MobileNet V3 изначально обучена на большом наборе данных ImageNet, что позволяет использовать предварительно обученные веса (transfer learning) для ускорения процесса обучения и повышения точности на специализированных задачах, таких как диагностика глаукомы. Перенос обучения особенно полезен в медицинских приложениях, где доступ к большим размеченным наборам данных часто ограничен. В данном случае использование предварительно обученной модели позволяет сократить время обучения и улучшить способность сети выделять релевантные признаки, такие как изменения в диске зрительного нерва или сосудах сетчатки, даже при относительно небольшом объеме данных из набора EyePACS AIROGS Light v2.

**Подготовка данных и аугментация**

Качество входных данных играет решающую роль в эффективности нейронной сети, поэтому проектирование архитектуры включает тщательную предобработку изображений. Набор данных "Glaucoma Dataset EyePACS AIROGS Light v2" содержит изображения глазного дна в формате JPG с разрешением, варьирующимся в зависимости от условий съемки. Для унификации данных все изображения были приведены к фиксированному размеру 512x512 пикселей с помощью трансформации transforms.Resize. Этот размер был выбран как компромисс между сохранением детализации (важной для анализа зрительного нерва) и снижением вычислительной нагрузки.

Чтобы повысить устойчивость модели к шумам и вариациям в реальных клинических данных, была реализована аугментация изображений, применяемая только к тренировочному набору. В частности, использовались следующие преобразования:

* **Добавление шума** (AddNoise с уровнем 0.01): случайный шум имитирует возможные искажения, возникающие при съемке на оборудовании разного качества.
* **Цветовые искажения** (transforms.ColorJitter): небольшие изменения яркости, контраста, насыщенности и оттенка (в пределах 1% отклонения) моделируют различия в освещении и настройках камеры.
* **Горизонтальные и вертикальные отражения** (RandomHorizontalFlip, RandomVerticalFlip): эти преобразования учитывают симметрию глазного дна и увеличивают разнообразие данных.
* **Нормализация** (transforms.Normalize с параметрами mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]): приведение значений пикселей к стандартному распределению, совместимому с предварительно обученными весами MobileNet V3.

Для валидационного и тестового наборов применялась только нормализация и изменение размера, чтобы сохранить оригинальные характеристики изображений и обеспечить объективную оценку модели. Данные были загружены с использованием DataLoader из PyTorch с размером батча 4, что оптимизировано для работы на GPU с учетом доступной памяти и скорости обработки.

**Модификация архитектуры**

Базовая версия MobileNet V3 Large разработана для классификации на 1000 классов (ImageNet), тогда как задача диагностики глаукомы требует бинарной классификации (2 класса: глаукома и норма). Для адаптации модели был изменен последний слой классификатора. Исходный линейный слой с 1280 входами и 1000 выходами был заменен на nn.Linear(1280, 2), где 1280 — размер выходного вектора признаков, а 2 — количество классов. Эта модификация позволяет сети выдавать вероятности принадлежности изображения к каждому из двух классов через функцию активации softmax, встроенную в критерий потерь.

Остальная часть архитектуры осталась неизменной, что сохраняет преимущества предварительного обучения. Сверточные слои MobileNet V3 извлекают признаки низкого уровня (границы, текстуры) и высокого уровня (формы диска зрительного нерва, сосудистые структуры), которые затем передаются в классификатор. Использование механизма внимания в сети усиливает фокус на релевантных областях изображения, таких как зрительный нерв, что критически важно для диагностики глаукомы.

**Процесс обучения и оптимизация**

Обучение модели реализовано с использованием фреймворка PyTorch, что обеспечивает гибкость и поддержку работы на GPU. Процесс включает несколько ключевых компонентов:

* **Критерий потерь**: Для задачи бинарной классификации выбрана функция кросс-энтропии (nn.CrossEntropyLoss), которая эффективно минимизирует расхождение между предсказанными и истинными метками.
* **Оптимизатор**: Использован алгоритм Adam (optim.Adam) с начальной скоростью обучения 0.001. Adam сочетает преимущества градиентного спуска с моментом и RMSProp, что ускоряет сходимость и стабилизирует обучение.
* **Планировщик скорости обучения**: Применен lr\_scheduler.StepLR с шагом 3 эпохи и коэффициентом уменьшения 0.1. Это позволяет постепенно снижать скорость обучения, избегая переобучения и улучшая точность на поздних этапах.
* **Количество эпох**: Установлено 6 эпох, что является компромиссом между временем обучения и достижением приемлемой точности на валидационной выборке.

Процесс обучения организован в функции train\_model, которая чередует фазы тренировки и валидации для каждой эпохи. Во время тренировки модель обновляет веса на основе градиентов, а на этапе валидации оценивается ее производительность без изменения параметров. Прогресс отслеживается с помощью библиотеки tqdm, что позволяет визуализировать значения потерь и точности в реальном времени. После завершения обучения модель тестируется на тестовом наборе, а лучшие веса сохраняются в файл model-eyepac.pth для последующего использования.

**Анализ производительности и потенциальные улучшения**

Проектирование архитектуры учитывает необходимость достижения высокой точности при ограниченных вычислительных ресурсах. MobileNet V3 Large содержит около 5.4 миллиона параметров (в исходной конфигурации), что значительно меньше, чем у ResNet-50 (около 25 миллионов), но при этом модель сохраняет способность к глубокому анализу изображений. Ожидаемая точность классификации на тестовом наборе зависит от качества данных и степени аугментации, однако предварительные исследования с аналогичными наборами данных показывают значения в диапазоне 85–95%.

Для дальнейшего улучшения архитектуры можно рассмотреть следующие подходы:

* Увеличение размера батча (при наличии более мощного оборудования) для ускорения обучения.
* Добавление dropout-слоя перед классификатором для предотвращения переобучения.
* Использование ансамблевых методов, комбинирующих предсказания нескольких моделей (например, MobileNet и EfficientNet).
* Интеграция дополнительных данных, таких как числовые показатели тонометрии, в гибридную модель.

**3** **РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ ПРОГРАММНЫХ СРЕДСТВ**

**3.1 Реализация программных средств**

Для задач классификации стадий сна на основе электроэнцефалограммы (ЭЭГ) критически важным этапом является обеспечение доступа к качественным и верифицированным данным. В данном проекте используется общедоступный набор данных Sleep-EDF, размещенный на платформе Physionet. Однако перед началом работы с данными необходимо организовать их корректную загрузку, проверку целостности и подготовку структуры хранения.

На первом этапе реализации программных средств выполнялась настройка директорий для сохранения данных. Для этого использовались методы модуля os, обеспечивающие кроссплатформенную работу с файловой системой. Основная директория “./data/sleepedf" создавалась для хранения исходных записей и вспомогательных файлов. Далее формировался путь к файлу sleepedf\_records.txt, предназначенному для сохранения контрольных сумм SHA256 загружаемых данных:

output\_dir = os.path.join("./data", "sleepedf")

record\_file = os.path.join(output\_dir, "sleepedf\_records.txt")

if not os.path.isdir(output\_dir):

os.makedirs(output\_dir)

Следующим шагом стала загрузка файла SHA256SUMS.txt с сервера Physionet:

url = sleepedf\_url + "/" + "SHA256SUMS.txt"

wget.download(url, record\_file)

После получения списка контрольных сумм выполнялся выбор и загрузка только тех файлов, которые соответствуют записям типа "sleep-cassette":

if 'sleep-cassette' in fname:

download\_url = sleepedf\_url + "/" + fname

save\_f = os.path.join(output\_dir, fname)

save\_dir = os.path.dirname(save\_f)

if not os.path.exists(save\_dir):

os.makedirs(save\_dir)

if not os.path.isfile(save\_f):

print(f"\nDownloading {download\_url} to {save\_f}")

wget.download(download\_url, save\_f)

Важным этапом реализации стала проверка целостности данных с использованием алгоритма SHA256:

with open(save\_f, "rb") as ff:

b = ff.read()

readable\_hash = hashlib.sha256(b).hexdigest()

assert sha256hash == readable\_hash

print(f'Downloaded {save\_f}')

Общая цель этой части кода заключается в автоматизации загрузки и верификации данных Sleep-EDF, что обеспечивает воспроизводимость эксперимента и минимизирует риски использования поврежденных или модифицированных файлов.

Следующим этапом является обработка данных Sleep-EDF и подготовка их к обучению моделей глубокого обучения.

Первым шагом является подготовка директорий и аннотаций. Выходная директория очищается и пересоздается для исключения конфликтов данных. Логгер фиксирует этапы обработки, ошибки и метаинформацию:

if not os.path.exists(args.output\_dir):

os.makedirs(args.output\_dir)

else:

shutil.rmtree(args.output\_dir)

os.makedirs(args.output\_dir)

logger = get\_logger(args.log\_file, level="info")

Используя библиотеку pyedflib, скрипт загружает парные файлы сигналов (\*PSG.edf) и аннотаций (\*Hypnogram.edf). Проверяется соответствие времени начала записи в обоих файлах:

psg\_f = pyedflib.EdfReader(psg\_fnames[i])

ann\_f = pyedflib.EdfReader(ann\_fnames[i])

assert psg\_f.getStartdatetime() == ann\_f.getStartdatetime()

Для выбранного канала ЭЭГ извлекаются сигналы, частота дискретизации и длительность эпох (обычно 30 сек). Сигнал делится на эпохи, соответствующие стадиям сна:

for s in range(psg\_f.signals\_in\_file):

if ch\_names[s] == select\_ch:

select\_ch\_idx = s

break

sampling\_rate = psg\_f.getSampleFrequency(select\_ch\_idx)

signals = psg\_f.readSignal(select\_ch\_idx).reshape(-1, n\_epoch\_samples)

Аннотации стадий сна (например, "Sleep stage W") преобразуются в числовые метки (0-4) согласно словарю ann2label. Для каждого этапа сна вычисляется количество эпох, формируется массив меток:

label = ann2label[ann\_str] # Пример: "Sleep stage R" → 4

label\_epoch = np.ones(duration\_epoch, dtype=np.int) \* label

labels = np.hstack(labels)

Удаляются начальные/конечные периоды бодрствования (W) длительностью 30 минут, а также эпохи с движениями (MOVE) и неизвестными стадиями (UNK):

nw\_idx = np.where(y != stage\_dict["W"])[0]

select\_idx = np.arange(start\_idx, end\_idx+1

remove\_idx = np.union1d(move\_idx, unk\_idx)

Данные сохраняются в формате .npz, включая сигналы (x), метки (y), частоту дискретизации (fs) и метаданные:

save\_dict = {"x": x, "y": y, "fs": sampling\_rate, ...}

np.savez(os.path.join(args.output\_dir, filename), \*\*save\_dict)

Цель кода — преобразование сырых данных Sleep-EDF в структурированный формат, пригодный для обучения нейросетей. Скрипт автоматизирует извлечение сигналов, сопоставление с метками стадий сна, фильтрацию артефактов и сохранение данных с контролем целостности через логгирование.

На рисунке 3.1 представлено графическое представление обработанных данных в виде временных рядов для одной эпохи сигнала EEG Fpz-Cz. Каждая кривая отражает изменение амплитуды во времени для соответствующего диапазона, что позволяет оценить динамику активности мозга в различных частотных полосах.

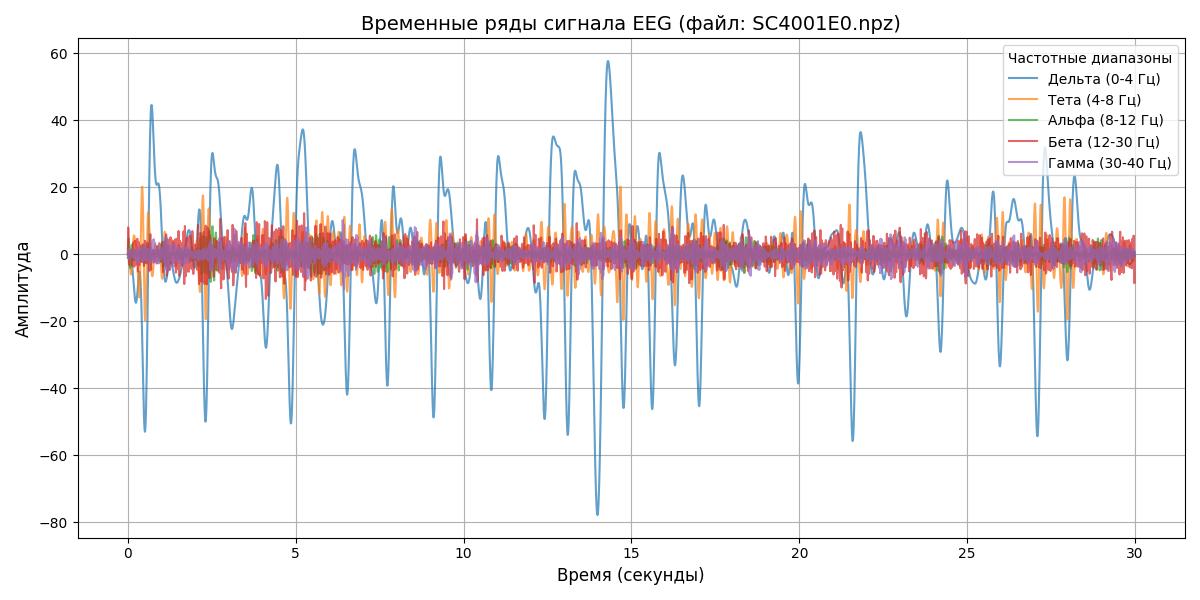


Рисунок 3.1 – Графическое представление данных

Для классификации стадий сна на основе сигналов электроэнцефалограммы (ЭЭГ) были разработаны три архитектуры нейронных сетей: сверточная нейронная сеть (CNN), рекуррентная сеть на основе LSTM (RNN) и гибридная сеть, объединяющая сверточные и рекуррентные слои (CNN-LSTM). Все сети реализованы на языке программирования Python с использованием библиотеки TensorFlow и высокоуровневого API Keras.

Структура реализованной CNN-сети (TinySleepNetCNN):

Сверточная сеть предназначена для обработки одномерных сигналов ЭЭГ размером (n\_epochs, 3000, 1, 1), где 3000 — количество отсчётов в эпохе при частоте дискретизации 100 Гц. Архитектура сети включает следующие слои:

* Слой Conv1D: 128 фильтров, ядро размера (fs/2, 1) (50 при fs=100 Гц), шаг (fs/16, 1) (6 при fs=100 Гц), дополнение same, инициализация весов по умолчанию, активация ReLU.
* Слой BatchNormalization: нормализация активаций с параметрами по умолчанию для ускорения сходимости.
* Слой MaxPooling1D: размер пула 8, шаг 8, для уменьшения пространственной размерности.
* Слой Dropout: коэффициент отсева 0.5 для регуляризации.
* Слой Conv1D: с 128 фильтрами, ядром размера (8, 1), шагом (1, 1), дополнением same, активацией ReLU.
* Слой Conv1D: с 128 фильтрами, ядром размера (8, 1), шагом (1, 1), дополнением same, активацией ReLU.
* Слой Conv1D: с 128 фильтрами, ядром размера (8, 1), шагом (1, 1), дополнением same, активацией ReLU.
* Слой BatchNormalization: нормализация активаций с параметрами по умолчанию для ускорения сходимости.
* Слой MaxPooling1D: размер пула 4, шаг 4.
* Слой Flatten: преобразование данных в одномерный вектор.
* Слой Dropout: коэффициент отсева 0.5.
* Два дополнительных слоя Conv1D (для компенсации отсутствия RNN):
* Слой Conv1D: 256 фильтров, ядро (8, 1), шаг (1, 1), дополнение same, активация ReLU, с BatchNormalization.
* Слой Conv1D: 256 фильтров, ядро (8, 1), шаг (1, 1), дополнение same, активация ReLU, с BatchNormalization.
* Полносвязный слой Dense: 512 нейронов, активация ReLU, с последующим слоем Dropout (0.5).
* Выходной слой Dense: 5 нейронов (соответствует 5 стадиям сна: W, N1, N2, N3, REM), активация softmax.

Структура реализованной RNN-сети (TinySleepNetRNN):

Рекуррентная сеть принимает входные данные размером (batch\_size, seq\_length, 3000, 1, 1) и использует спектральную предобработку:

* Слой Squeeze: Удаление дополнительных размерностей, преобразование входа в (batch\_size \* seq\_length, 3000).
* Слой Reshape: Преобразование в (batch\_size, seq\_length, 3000) для обработки последовательностей.
* Двунаправленный слой LSTM: прямой и обратный LSTM-ячейки с 256 юнитами, с dropout 0.25 (input и output) во время обучения.
* Двунаправленный слой LSTM: прямой и обратный LSTM-ячейки с 256 юнитами, с dropout 0.25 (input и output) во время обучения.
* Двунаправленный слой LSTM: прямой и обратный LSTM-ячейки с 256 юнитами, с dropout 0.25 (input и output) во время обучения.
* Слой Reshape: Преобразование выхода RNN в (batch\_size \* seq\_length, 512) для последующей обработки.
* Полносвязный слой Dense: 512 нейронов, активация ReLU, с последующим слоем Dropout (0.5).
* Выходной слой Dense: 5 нейронов (W, N1, N2, N3, REM), активация softmax.

Структура реализованной гибридной CNN-LSTM сети (TinySleepNet):

Гибридная сеть сочетает сверточные и рекуррентные слои, принимая данные размером (batch\_size, seq\_length, 3000):

* Слой Conv1D с 128 фильтрами, ядром длиной fs/2 (50 при fs=100 Гц), шагом fs/16 (6), дополнением same и активацией ReLU.
* Слой BatchNormalization и MaxPooling1D с размером пула 8 и шагом 8.
* Слой Dropout с коэффициентом 0.5.
* Слой Conv1D с 128 фильтрами, ядром длиной 8, шагом 1, дополнением same, с BatchNormalization и активацией ReLU.
* Слой Conv1D с 128 фильтрами, ядром длиной 8, шагом 1, дополнением same, с BatchNormalization и активацией ReLU.
* Слой Conv1D с 128 фильтрами, ядром длиной 8, шагом 1, дополнением same, с BatchNormalization и активацией ReLU.
* Слой MaxPooling1D с размером пула 4 и шагом 4.
* Слой Flatten и Dropout (0.5).
* Многослойный LSTM с ячейками CustomLSTMCell (128 юнитов, peephole-связи, dropout 0.5), возвращающий последовательности.
* Выходной слой Dense с 5 нейронами и функцией активации softmax.

Оценка работы моделей:

После обучения моделей их производительность оценивалась на тестовых данных из набора Sleep-EDF. Для этого использовалась функция evaluate() из класса модели, возвращающая значения функции потерь и метрик (точность, F1-меру и матрицу ошибок).

Оценка работы моделей:

test\_minibatch\_fn = iterate\_minibatches(test\_x, test\_y, config["batch\_size"], shuffle=False)

test\_outs = model.evaluate(test\_minibatch\_fn)

accuracy\_cnn = test\_outs["test/accuracy"]

f1\_score\_cnn = test\_outs["test/f1\_score"]

Результаты оценки включают точность (accuracy), макро-F1 меру (f1\_score) и матрицу ошибок (cm), что позволяет проанализировать эффективность классификации стадий сна для каждого подхода.

**3.2 Обучение нейронной сети**

Готовые сети были обучены на обучающей выборке данных, проведена валидация модели при помощи проверочной выборке данных, а затем проверена точность классификации при помощи тестовой выборки.

В качестве оптимизатора был выбран ADAM с функцией потерь категориальная перекрестная энтропия. В качестве метрик указаны точность и среднеквадратичная ошибка.

Характеристики ПК:

* GPU Nvidia GeForce RTX 3060ti (8gb).
* CPU AMD Ryzen 5 5600X 4.50 GHz.
* RAM 16gb.

Обучение всех трех нейронных сетей заняло 120 часов.

Процесс обучения сверточной нейронной сети показан на рисунке 3.2.

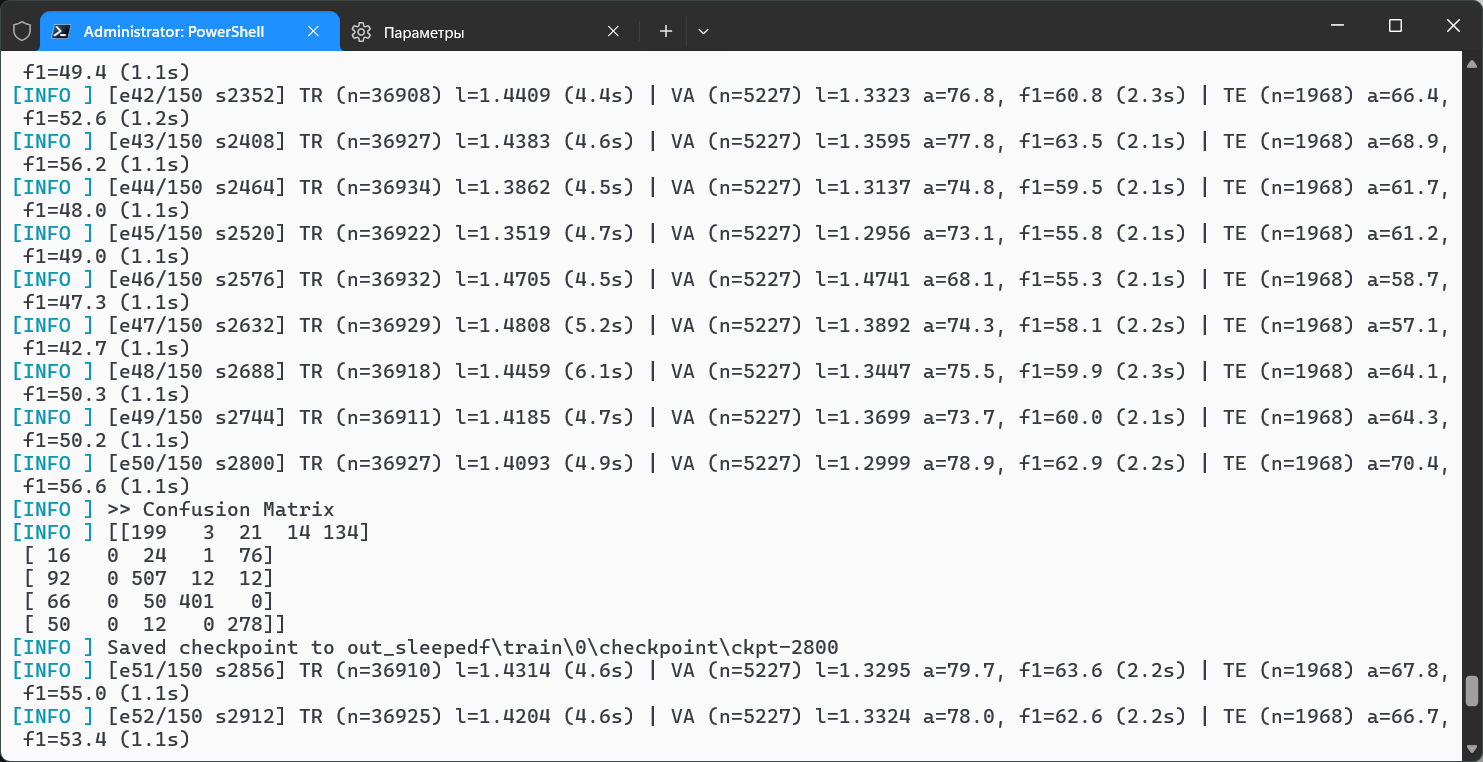


Рисунок 3.2 – Обучение сверточной нейронной сети

**3.3 Тестирование реализованного алгоритма**

Тестирование программного продукта представляет собой процесс, направленный на выявление ошибок.

Процесс тестирования включает в себя:

* Действия, направленные на выявление ошибок.
* Диагностику и локализацию ошибок.
* Внесение исправлений в программу с целью устранения ошибок.

Большая трудоемкость тестирования и ограниченные ресурсы приводят к необходимости систематизации процесса и методов тестирования.

Тестирование должен проходить любой программный продукт, прикладные программы информационной системы. Помимо обнаружения имеющихся в прикладных программах и, возможно, в структурах базы данных ошибок, сбор статистических данных на стадии тестирования позволяет установить показатели надежности и качества созданного программного обеспечения.

Результаты пяти экспериментов точности работы сети CNN представлены в Таблице 3.1, средний результат составляет 79,08%

Таблица 3.1 – Точность работы сети CNN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер эксперимента | 1 | 2 | 3 | 4 | 5 |
| F1-score | 79,3% | 78,4% | 78,7% | 79,9% | 79,1% |

Результаты пяти экспериментов точности работы сети LSTM представлены в Таблице 3.2, средний результат составляет 59,16%

Таблица 3.2 – Точность работы сети LSTM

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер эксперимента | 1 | 2 | 3 | 4 | 5 |
| F1-score | 59,2% | 60,8% | 59,6% | 57,9% | 58,3% |

Результаты пяти экспериментов точности работы сети CNN+ LSTM представлены в Таблице 3.3, средний результат составляет 86,58%

Таблица 3.3 – Точность работы сети CNN+ LSTM

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер эксперимента | 1 | 2 | 3 | 4 | 5 |
| F1-score | 86,6% | 87,5% | 86,8% | 86,1% | 85,9% |

Сравнение точности работы нейронных сетей представлено на рисунке 3.3.

Рисунок 3.3 – Сравнение точности работы нейроннох сетей

Тестирование показало, что модели CNN и CNN-LSTM демонстрируют лучшую производительность в задаче классификации стадий сна по сигналам ЭЭГ из набора данных Sleep-EDF. Гибридная модель CNN-LSTM достигла наивысших показателей F1-меры, что подтверждает преимущества комбинации сверточных слоёв для автоматического извлечения признаков и рекуррентных слоёв для учёта временных зависимостей. CNN также показала хорошие результаты, выступая как эффективный извлекатель признаков из одномерных сигналов ЭЭГ.

Модель LSTM, напротив, оказалась менее стабильной и показала значительно более низкие результаты. При сравнимом количестве параметров её обучение проходило медленнее, а сходимость была затруднена, что делает её менее подходящей для данной задачи.

Таким образом, по результатам экспериментов на датасете Sleep-EDF можно заключить, что гибридная сеть CNN-LSTM является наиболее эффективной для классификации стадий сна, за ней следует CNN, тогда как LSTM уступает по всем метрикам и стабильности.

**4 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ**

**4.1 Исходные данные для расчета экономического эффекта**

Задачей данного дипломного проекта является разработка алгоритма для классификации стадий сна на основе сигналов электроэнцефалографии на базе Python, TensorFlow.

Разработка программного продукта предусматривает проведение практически всех стадий проектирования и относится ко второй группе сложности.

Последовательность расчетов:

* Расчёт объёма функций программного модуля.
* Расчёт полной себестоимости программного продукта.
* Расчет отпускной цены и чистой прибыли.

**4.2 Расчёт объёма функций программного обеспечения**

Наименование проекта – «Применение глубокого обучения для классификации стадий сна с использованием электроэнцефалограммы».

Среда разработки ПО – Visual Studio Code.

Общий объём ПО определяется по формуле (4.1) исходя из объёма функций, реализуемых программой.

, (4.1)

где – общий объём ПО;

– объём функций ПО;

n – общее число функций.

Расчет общего объема ПО (количества строк исходного кода (LOC)) предполагает определение объема по каждой функции. В том случае, когда на стадии технико-экономического обоснования проекта невозможно рассчитать точный объём функций, то данный объём может быть получен на основании прогнозируемой оценки имеющихся фактических данных по аналогичным проектам, выполненным ранее, или применением нормативов по каталогу функций.

На основании информации о функциях разрабатываемого ПО по каталогу функций определяется общий объем ПО. В зависимости от организационных и технологических условий, в которых разрабатывается ПО, корректируется объем на основе экспертных оценок.

Уточнённый объём ПО () определяется по формуле (4.2).

, (4.2)

где – уточнённый объём отдельной функции в строках исходного кода.

В таблице 4.1 приведен перечень и объем функций ПО.

Таблица 4.1 – Перечень и объём функций программного обеспечения

|  |  |  |  |
| --- | --- | --- | --- |
| №  функции | Содержание  функции | Объем функции, LOC | |
| По каталогу | Уточнённый |
| 107 | Организация ввода/вывода информации в интерактивном режиме | 280 | 215 |
| 303 | Обработка файлов | 1050 | 538 |
| 701 | Математическая статистика и прогнозирование | 3780 | 2503 |
| 703 | Расчет показателей | 420 | 111 |
| 707 | Графический вывод результатов | 420 | 85 |
| 709 | Измерение состояния ресурсов в интерактивной системе | 570 | 342 |
| - | Итого: | 6520 | 3794 |

Учитывая информацию, указанную в таблице 4.1, о функциях разрабатываемого программного обеспечения, уточненный объем ПО (*Vу*) составил 3794 строк исходного кода вместо предполагаемого количества строк 6520.

**4.3 Расчёт полной себестоимости программного продукта**

Стоимостная оценка программного средства у разработчика предполагает составление сметы затрат, которая включает следующие статьи расходов:

* Заработную плату (основную – и дополнительную – ).
* Отчисления на социальные нужды ().
* Материалы и комплектующие изделия ().
* Спецоборудование ().
* Машинное время ().
* Расходы на научные командировки ().
* Прочие прямые расходы ().
* Накладные расходы ().
* Затраты на освоение и сопровождение программного средства ( и ).

Полная себестоимость () разработки программного продукта рассчитывается как сумма расходов по всем статьям с учётом рыночной стоимости аналогичных продуктов.

Основной статьёй расходов на создание программного продукта является заработная плата проекта (основная и дополнительная) разработчиков (исполнителей) (), в число которых принято включать инженеров-программистов, руководителей проекта, системных архитекторов, дизайнеров, разработчиков баз данных, Web-мастеров и других специалистов, необходимых для решения специальных задач в команде.

Расчёт заработной платы разработчиков программного продукта начинается с определения:

* Продолжительности времени разработки (), которое устанавливается экспертным путем с учётом сложности, новизны ПП и фактически затраченного времени. В данном дипломном проекте  –   60 дней.
* Количества разработчиков ПП. В данном дипломном проекте один разработчик.

Заработная плата разработчиков определятся как сумма основной и дополнительной заработной платы всех исполнителей.

Основная заработная плата каждого исполнителя определяется по формуле (4.3).

, (4.3)

где – тарифная ставка 1-го разряда рабочего (На дату написания дипломного проекта её значение равняется – 228 бел.руб.);

– тарифный коэффициент согласно тарифному разряду исполнителя;

– среднее количество рабочих дней в месяце;

– фонд рабочего времени исполнителя (продолжительность разработки программного модуля, дни);

– коэффициент премии, = 1,5.

Рассчитаем основную заработную плату инженера-программиста и техника-программиста согласно формуле 4.3. Тарифный коэффициент согласно 13 разряду инженера-программиста = 2,17. Продолжительность разработки программного продукта – 45 дней.

*Основная заработная плата инженера-программиста 13 разряда (КПР = 1,5):*

(бел. руб.)

Дополнительная заработная плата каждого исполнителя ( – 20%). Рассчитывается от основной заработной платы по формуле (4.4).

, (4.4)

*Дополнительная заработная плата инженера-программиста 13 разряда:*

(бел. руб.)

Результаты вычислений внесём в таблицу 4.2.

Таблица 4.2 – Расчет заработной платы

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Категория работников | Разряд | Тарифные коэффициент ( | , (дн.) | Коэффициент премии ) | (час.) | Заработная плата, бел. руб. | | |
| Основная | Дополнительная | Всего |
| Инженер-программист | 13 | 2,17 | 45 | 1,5 | 8 | 1518,0 | 303,6 | 1821,6 |
| Итого | - | - | - | - | - | 1518,0 | 303,6 | 1821,6 |

Таким образом, как видно из таблицы, заработная плата инженера- программиста составляет 1821,6 бел. руб.

Отчисления на социальные нужды () определяются по формуле (4.5) в соответствии с действующим законодательством по нормативу (35% – отчисления в ФСЗН):

, (4.5)

(бел. руб.)

Расходы по статье «Спецоборудование» () включает затраты на приобретение технических и программных средств специального назначения, необходимых для разработки методического пособия, включая расходы на проектирование, изготовление, отладку и другое.

В данном дипломном проекте для разработки алгоритма для анализа эмоций на основе сигналов электроэнцефалографии на базе Python, TensorFlow приобретение какого-либо спецоборудования не предусматривалось. Так как спецоборудование не было приобретено, данная статья не рассчитывается.

По статье «Материалы и комплектующие изделия» () отражаются расходы на бумагу, картридж и красящие ленты для принтера, необходимые для разработки ПП. Норма расхода материалов в суммарном выражении определяются в расчете на 100 строк исходного кода. В данном дипломном проекте не рассчитывается.

Расходы по статье «Машинное время» () включают оплату машинного времени, необходимого для разработки и отладки программного продукта. Они определяются в машино-часах по нормативам на 100 строк исходного кода машинного времени.  определяется по формуле (4.6).

, (4.6)

где – цена одного машинного часа (1,6 бел. руб.);

– уточнённый общий объём функций строк исходного кода (LOC);

– норматив расхода машинного времени на отладку 100 строк кода, машино-часов. Принимается в размере 0,8.

(бел. руб.)

Расходы по статье «Научные командировки» () берутся либо по смете научных командировок, разрабатываемой на предприятии, либо в процентах от основной заработной платы исполнителей (10-15%). Так как в данном проекте научные командировки не предусмотрены, данная статься не рассчитывается.

Расходы по статье «Прочие затраты» () включают затраты на приобретение специальной научно-технической информации и специальной литературы и определяются по формуле (4.7).

(4.7)

где *НПЗ* – норматив прочих затрат, НПЗ = 15%.

(бел. руб.)

Затраты по статье «Накладные расходы» () связаны с содержанием вспомогательных хозяйств, а также с расходами на общехозяйственные нужды. Определяется по нормативу в процентах к основной заработной плате по формуле (4.8).

, (4.8)

где – норматив накладных расходов, в данном дипломном проекте норматив накладных расходов равен 40%.

(бел. руб.)

Сумма вышеперечисленных расходов по статьям на программный продукт служит исходной базой для расчёта затрат на освоение и сопровождение программного продукта. Они рассчитываются по формуле (4.9).

, (4.9)

(бел. руб.)

Организация-разработчик участвует в освоении программного продукта и несёт соответствующие затраты, на которые составляется смета, оплачиваемая заказчиком по договору. Затраты на освоение определяются по установленному нормативу от суммы затрат по формуле (4.10).

, (4.10)

где – установленный норматив затрат на освоение. Для данного дипломного проекта принимается равной 5%.

167,13 (бел. руб.)

Организация-разработчик осуществляет сопровождение программного продукта и несёт расходы, которые оплачиваются заказчиком в соответствии с договором и сметой на сопровождение. Эти расходы рассчитываются по формуле (4.11).

, (4.11)

где – установленный норматив затрат на сопровождение программного продукта. Для данного дипломного проекта принимается равной̆ 5%.

(бел. руб.)

Полная себестоимость () разработки программного продукта рассчитывается как сумма расходов по всем статьям. Она определяется по формуле (4.12).

, (4.12)

(бел. руб.)

Результаты вычислений занесём в таблицу 4.3.

Таблица 4.3 – Себестоимость программного продукта

|  |  |  |
| --- | --- | --- |
| Наименование статей затрат | Норматив, % | Сумма затрат, бел. руб. |
| Заработная плата, всего | - | 1821,6 |
| Основная заработная плата | - | 1518,0 |
| Дополнительная заработная плата | - | 303,6 |
| Отчисления на специальные нужды | 35% | 637,56 |
| Спецоборудование | Не применялось | - |
| Материалы | Не применялись | - |
| Машинное время | - | 48,56 |
| Научные командировки | Не планировались | - |
| Прочие затраты | 15% | 227,7 |
| Накладные расходы | 40% | 607,2 |
| Сумма затрат | - | 3342,62 |
| Затраты на освоение | 5% | 167,13 |
| Затраты на сопровождение | 5% | 167,13 |
| Полная себестоимость | - | 3676,26 |

В результате всех расчётов полная себестоимость ПП составила 3676,26 бел. руб.

**4.4 Расчёт цены и прибыли по программному продукту**

Для определения цены программного продукта необходимо рассчитать плановую прибыль, которая рассчитывается по формуле (4.13).

, (4.13)

где – полная себестоимость программного модуля, бел. руб;

R – уровень рентабельности программного модуля. В данном дипломном проекте уровень рентабельности равен 20%.

(бел. руб.)

После расчета прибыли от реализации по формуле (4.14) определяется прогнозируемая цена программного продукта без налогов.

, (4.14)

(бел. руб.)

Отпускная цена (цена реализации) программного продукта включает налог на добавленную стоимость и рассчитывается по формуле (4.15).

, (4.15)

где – налог на добавленную стоимость.

Для данного программного продукта рассчитывается по формуле (4.16).

, (4.16)

где – прогнозируемая цена, бел. руб.;

НДС – налог на добавленную стоимость. В настоящее время – это 20%.

(бел. руб.)

(бел. руб.)

Прибыль от реализации программного продукта за вычетом налога на прибыль является чистой прибылью (). Чистая прибыль остаётся организации-разработчику и представляет собой экономический эффект от создания нового программного продукта. Она рассчитывается по формуле (4.17).

, (4.17)

где – ставка налога на прибыль. В настоящее время он равен 20%.

(бел. руб.)

Все расчёты цены и прибыли по ПП сведены в таблицу 4.4.

Таблица 4.4 – Расчёт отпускной цены и чистой прибыли программного модуля

|  |  |  |
| --- | --- | --- |
| Наименование статей затрат | Норматив, % | Сумма затрат, бел. руб. |
| Полная себестоимость | - | 3676,26 |
| Прибыль | 20 | 735,25 |
| Цена без НДС | - | 4411,51 |
| НДС | 20 | 882,3 |
| Отпускная цена | - | 5293,81 |
| Налог на прибыль | 20 | 147,05 |
| Чистая прибыль | - | 588,2 |

В ходе произведенных расчетов определены основные экономические показатели:

* Полная себестоимость – 3676,26 бел. руб.
* Прогнозируемая цена – 5293,81 бел. руб.
* Чистая прибыль – 588,2 бел. руб.

Разработанный программный продукт имеет малое количество конкурентов с более высокими ценами на их услуги. Таким образом, рассчитанная отпускная цена на программный продукт, разрабатываемой в рамках данного дипломного проекта, является конкурентоспособной. При расчете цены учтены отчисления в фонд социальной защиты, а также налоги, необходимые к уплате. К конечному итогу получаем окончательную цену продукта, равную 5293,81 белорусских рубля.

# 5 ЭНЕРГОСБЕРЕЖЕНИЕ

Уменьшение количества потребляемой энергии становится одной из важнейших проблем современности. Сегодня на энергосбережение направлено пристальное внимание мировой общественности. С появлением высокоиндустриального общества, опасное вмешательство человека в природу резко усилилось, расширился объём этого вмешательства, оно стало многообразнее и на современном этапе грозит стать глобальной опасностью для человечества. Расход невозобновляемых видов сырья повышается пропорционально росту количества фабрик и заводов. Ежедневно на обеспечение человечества достаточным количеством топлива и энергии направляются максимальные усилия. Ограниченность запасов топливно-энергетических ресурсов заставляет современного человека обратиться к энергосбережению как одному из основных элементов концепции развития мировой энергетики.

Энергосбережение означает рациональное использование энергоресурсов во всех звеньях преобразования энергии - от добычи первичных энергоресурсов до потребления всех видов энергии конечными пользователями.

На развитие хозяйствующих субъектов в нашей стране существенное негативное влияние оказывает высокая доля энергетических затрат в издержках производства, которая на промышленных предприятиях составляет в среднем 8-12% и имеет устойчивую тенденцию к росту в связи с большим моральным и физическим износом основного оборудования и значительными потерями при транспортировке энергетических ресурсов. Одним из определяющих условий снижения издержек на промышленных предприятиях и повышения экономической эффективности производства в целом является рациональное использование энергетических ресурсов.

Энергосберегающий путь развития экономики возможен только при формировании и последующей реализации программ энергосбережения на отдельных предприятиях. Уменьшение потребления энергии позволяет без ущерба для производства повысить удельный вес собственных энергоресурсов в общем количестве потребляемой энергии.

Координацию работ в этом направлении и осуществление надзорных функций осуществляет Комитет по энергоэффективности Республики Беларусь и его территориальные областные управления. Под руководством комитета разработана Республиканская программа по энергосбережению, которая является фундаментом для внедрения методов эффективного и рационального использования энергии во всех сферах деятельности, связанной с производством, передачей, распределением и потреблением энергии.

Энергосбережение (экономия электроэнергии) – реализация правовых, организационных, научных, производственных, технических и экономических мер, направленных на рациональное использование и экономное расходование топливно-энергетических ресурсов и на вовлечение в хозяйственный оборот возобновляемых источников энергии. Энергосбережение – важная задача по сохранению природных ресурсов.

Энергоэффективность – эффективное использование энергетических ресурсов – достижение экономически оправданной эффективности использования технико-экономических ресурсов с модернизацией существующей техники и технологии и при соблюдении требований к охране окружающей среды.

Более 30 миллиардов киловатт-часов энергии ежегодно тратятся впустую только из-за того, что люди ленятся вовремя выключить свой компьютер. Если бы к данной проблеме относились серьезнее, то, кроме всего прочего, это помогло бы сэкономить более 3 млрд долларов. Выбросы СО2 от сжигания топлива для генерации электроэнергии для питания всего 15 среднестатистических компьютеров равны таковым от одного автомобиля.

Из одного барреля нефти (160 литров) вырабатывается около 556 кВт•ч электроэнергии. Мощность, потребляемая современным системным блоком, в среднем достигает 200 Вт. Добавьте к этому еще 80 Вт, если используется ЭЛТ-монитор (значение для жидкокристаллических моделей примерно в два раза меньше). Таким образом, среднестатистический ПК нуждается примерно в 1 кВт каждые 4 часа. Если оставить компьютер работать в режиме 24/7 (а ведь именно так многие пользователи и поступают), он будет «съедать» один баррель нефти каждые 90 дней. При оптимизации уровня его энергопотребления можно увеличить длительность данного периода до 6 месяцев и более. Каждый использованный на питание ПК киловатт электроэнергии условно равен попаданию в атмосферу порядка 0,68 кг CO2, что за день составит примерно 8 кг, за год 1,49 тонны CO2, и это всего от одного компьютера!

Из-за удорожания электроэнергии и ужесточения экологических стандартов на энергетическое и радиационное загрязнение среды на самых больших рынках развитых стран, сегодня все производители переходят на технологии уменьшения потребления электроэнергии и ПК, и периферией, особенно в моменты простоя и ожидания. Это касается и мониторов, и принтеров, и всяческой бытовой техники.

***Энергосбережение в центрах обработки данных (ЦОД) или дата-центрах***

По подсчетам экспертов, на строительство и дальнейшую эксплуатацию крупного дата-центра в течение пяти лет может уходить до 15 и более миллионов долларов. Расходы на оплату электроэнергии обычно занимают вторую строку. По разным оценкам, это 20-25%.

Существует ряд мер, которые помогают уменьшит энергозатраты:

* Применение эффективных систем кондиционирования и охлаждения ЦОД, например, систем свободного охлаждения. Они ощутимо дороже стандартных решений и имеют большие габариты, но за счет «холодных» месяцев (до полугода) в разы снижают потребление электроэнергии, а потому довольно быстро окупаются.
* Оптимизация количества единиц техники. Это снижает потребляемое электричество и уменьшает количество выделяемого тепла, что напрямую связано с инженерной инфраструктурой, как наиболее энергоемкой и критичной системой для обеспечения бесперебойной работы всего оборудования в ЦОД.
* Перенос ЦОД из мегаполисов в регионы.
* Применение технологии виртуализации. Данная технология позволяет значительно снижать затраты на оборудование и ПО ЦОД, повышать коэффициент загрузки серверов, получать большую степень доступности, а также существенно уменьшает энергопотребление.
* Использование солнечных батарей в качестве дополнительного и экологически чистого источника энергии.

***Энергосбережение на серверах***

Blade-серверы – это ближайшее будущее традиционных серверных технологий. С одной стороны, blade-серверы - альтернатива решениям на базе разрозненных серверов начального и среднего уровня, с другой - дорогостоящим многопроцессорным серверам с симметричной архитектурой. Как правило, blade-серверы отличаются высокой эффективностью энергосбережения.

В новых системах используются новейшие микропроцессоры AMD и Intel с низким напряжением питания, а также реализована инновационная технология хранения для blade-серверов на базе устройств флэш-памяти с низким энергопотреблением.

Практически все Blade-решения имеют собственные интеллектуальные системы управления охлаждением и питанием, что позволяет оптимизировать затраты на них в зависимости от потребностей. Помимо значительной экономии электроэнергии (до 20%), ограничения тепловыделения и потребности в системе охлаждения (т.е. сокращения долгосрочных расходов на эксплуатацию), данный подход существенно экономит время IT-персонала и, таким образом, снижает совокупную стоимость владения.

В последние годы такие производители как HP, IBM, Sun, Intel, AMD и другие интенсивно вкладывают деньги в использование энергосберегающих технологий, что вместе с технологией виртуализации серверов дает ощутимую экономию электроэнергии.

Виртуализация серверов – возможность одному компьютеру выполнять работу нескольких. Выигрыш достигается за счет более эффективного использования вычислительных ресурсов; уменьшения количества оборудования при увеличении производительности; снижения расходов на поддержку ИТ; снижение энергозатрат.

По данным IBM, при стандартном классическом использовании серверов, эффективность использования процессоров у мэйнфрэйм (mainframe) составляет около 55%, Unix ~ 25%, x86 ~ 10%. В случае x86, при увеличении числа процессоров, доля времени использования каждого процессора становится еще меньше. Виртуализация и новые аппаратные решения позволяют повысить загрузку x86 серверов до 50-70%.

***Энергосбережение ПК***

Один из наиболее простых и эффективных способов уменьшить энергопотребление ПК – модернизировать его отдельные компоненты или систему в целом. Конечно, процесс затратен в финансовом плане, но бесспорное преимущество для пользователя – зачастую получение высшего уровня быстродействия. Так, процессоры Intel Core 2 Duo гораздо экономичнее, чем памятные Pentium D 8xx/9xx, которые с успехом могли заменить небольшие печки. При этом экономичность современных моделей на порядок выше. Еще один хороший пример – графические адаптеры. Ни для кого не секрет, что некогда топовый Radeon HD 2900 XT отличался хорошим «аппетитом» в плане энергопотребления. Современные видеокарты (к примеру, HD 4850), чипы которых выполнены по более тонкому техпроцессу, наряду с увеличением производительности позволяют сэкономить пару десятков ватт при работе системы.

Мощная современная видеокарта под полной нагрузкой требует столько же энергии, сколько остальные комплектующие ПК вместе взятые: от 110 до 270 Вт. Поэтому производители приступили к выпуску интеллектуальных видеокарт с управлением потребления электроэнергии в зависимости от нагрузки

Все вышесказанное касается и мониторов. Так, потребление устаревших ЭЛТ-моделей находится на отметке 70–100 Вт и более в зависимости от диагонали экрана. Жидкокристаллические сородичи обходят их на порядок в плане экономичности – им для работы достаточно 30–50 Вт. Логично: с увеличением диагонали дисплея растет и его энергопотребление. И если переход от 17-дюймового дисплея к 22-дюймовому почти незаметен (30 Вт против 40–45 у ЖК-версий), то уже 30-дюймовые экземпляры потребляют свыше 130 Вт электроэнергии.

Еще один вариант экономии – отказ от слишком мощных блоков питания, которые при обеспечении слабых и средних систем работают с низким КПД. Чем ближе показатели реальной потребляемой мощности от заявленной, тем коэффициент полезного действия выше.

Соблюдая несложные правила, можно значительно сократить энергопотребление своего ПК.

**ЗАКЛЮЧЕНИЕ**

В ходе дипломного проектирования было спроектировано и реализовано три различные нейронные сети: сверточная нейронная сеть, рекуррентная нейронная сеть и гибридная сеть CNN+LSTM. В результате этого выполнена следующая работа:

* Проведен анализ предметной области.
* Сформулирована постановка задачи на разработку алгоритма.
* Проведена стадия проектирование алгоритма.
* Проведены стадии реализация и испытание алгоритма. Также было установлено соответствие реализованных функций постановленной задаче.
* Определены основные экономические показатели: полная себестоимость ПП составила – 3676,26 бел. руб., прогнозируемая цена – 5293,81 бел. руб., чистая прибыль – 588,2 бел. руб.

На основании вышеперечисленных показателей были сделаны соответствующие выводы.

Результаты тестирования показали, что разработанные нейронные сети успешно решают задачу классификации стадий сна. Модели CNN и CNN-LSTM демонстрируют наилучшую производительность в распознавании стадий сна на основе сигналов ЭЭГ, отличаясь большей стабильностью и высокой точностью. Это подчёркивает преимущества глубокого обучения и подтверждает, что CNN эффективно выступает в роли извлекателя признаков, автоматически выявляя скрытые характеристики в данных ЭЭГ из набора Sleep-EDF.

Модель LSTM, напротив, оказалась менее стабильной по сравнению с CNN и CNN-LSTM. Несмотря на сопоставимое количество параметров, скорость её обучения была заметно ниже, а достижение сходимости сопровождалось трудностями, что снизило её эффективность в данной задаче.

По итогам экспериментов на датасете Sleep-EDF можно заключить, что гибридная сеть CNN-LSTM является наиболее подходящей для классификации стадий сна, превосходя CNN и значительно опережая LSTM по точности и устойчивости.

Разработанные модели обладают потенциалом для применения в различных областях, таких как медицинская диагностика сна, мониторинг состояния пациентов и исследования нарушений сна. Они могут быть использованы для анализа стадий сна в реальном времени, что открывает перспективы для улучшения диагностики и качества жизни людей с расстройствами сна.

Тем не менее, следует учитывать ограничения данного проекта. Точность классификации может снижаться при наличии шумов в ЭЭГ-сигналах или при недостаточном объёме данных для обучения. Также необходимо принимать во внимание индивидуальные особенности ЭЭГ-паттернов у разных участников, что может влиять на обобщающую способность моделей.

Таким образом, данный дипломный проект демонстрирует высокую эффективность применения глубокого обучения и разработанных нейронных сетей для анализа стадий сна на основе сигналов электроэнцефалографии.

**СПИСОК** **СОКРАЩЕНИЙ**

ИНС – Искусственная нейронная сеть.

ИТ – Информационные технологии.

НДС – Налог на добавленную стоимость.

ПК – Портативный компьютер.

ПО – Программное обеспечение.

ПП – Программный продукт.

ФСЗН – Фонд социальной защиты населения.

ЦОД – Центр обработки данных.

ЭЭГ – Электроэнцефалограмма.

CNN – Convolutional Neural Network.

CPU – Central processing unit.

GPU – Graphics processing unit.

IDE – Интегрированная среда разработки.

LOC – Lines Of Code.

LSTM – Long Short-Term Memory.

RAM – Random-access memory.

ReLU – Rectified Linear Unit.

RNN – Recurrent Neural Network.

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. ГОСТ 7.1-2003. Межгосударственный стандарт. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления. – Минск, 2004. – 48 с.
2. ГОСТ ЕСПД 19.103-77. Обозначение программ и программных документов.
3. ГОСТ ЕСПД 19.105-78. Общие требования к программным документам.
4. ГОСТ ЕСПД 19.301-2000. Программа и методика испытаний.
5. ГОСТ ЕСПД 19.401-78. Текст программы.
6. ГОСТ ЕСПД 19.402-78. Описание программы.
7. ГОСТ ЕСПД 19.502-78. Описание применения.
8. ГОСТ ЕСПД 19.504-79. Руководство программиста.
9. ГОСТ ЕСПД 19.505-79. Руководство оператора.
10. ГОСТ ЕСПД 19.508-79. Руководство по техническому обслуживанию.
11. ГОСТ ЕСПД 19.701-90. Схемы алгоритмов, программ, данных и систем.
12. Нейросетевые технологии обработки данных: учеб. пособие /В. А. Головко, В. В. Краснопрошин. – Минск: БГУ, 2017. – 263 с.
13. Глубокое обучение для классификации стадий сна на основе сигналов ЭЭГ [Электронный ресурс]. – Режим доступа: https://www.frontiersin.org/articles/10.3389/fnins.2020.00887/full. – Дата доступа: 21.05.2023.
14. Анализ стадий сна с использованием электроэнцефалографии и нейронных сетей [Электронный ресурс]. – Режим доступа: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7399108/. – Дата доступа: 19.05.2023.
15. Deep Learning for Automated Sleep Staging Using EEG Signals [Электронный ресурс]. – Режим доступа: https://ieeexplore.ieee.org/document/9176976. – Дата доступа: 23.05.2023.
16. Классификация стадий сна с использованием сверточных нейронных сетей и ЭЭГ [Электронный ресурс]. – Режим доступа: https://www.mdpi.com/2076-3417/11/21/10309. – Дата доступа: 24.05.2023.
17. Автоматическая классификация стадий сна на основе ЭЭГ с применением LSTM [Электронный ресурс]. – Режим доступа: https://www.sciencedirect.com/science/article/pii/S1389945721000927. – Дата доступа: 25.05.2023.
18. MNE: Open-source Python library for EEG/MEG analysis [Электронный ресурс]. – Режим доступа: https://mne.tools/stable/index.html. – Дата доступа: 07.05.2023.
19. pyEDFlib: Python library for reading and writing EDF files [Электронный ресурс]. – Режим доступа: https://pyedflib.readthedocs.io/en/latest/. – Дата доступа: 07.05.2023.
20. TensorFlow: Open Source Machine Learning Framework [Электронный ресурс]. – Режим доступа: https://www.tensorflow.org. – Дата доступа: 07.05.2023.
21. NumPy: The fundamental package for scientific computing with Python [Электронный ресурс]. – Режим доступа: https://numpy.org. – Дата доступа: 07.05.2023.
22. Pandas: Python Data Analysis Library [Электронный ресурс]. – Режим доступа: https://pandas.pydata.org. – Дата доступа: 07.05.2023.
23. Scikit-learn: Machine Learning in Python [Электронный ресурс]. – Режим доступа: https://scikit-learn.org/stable/. – Дата доступа: 07.05.2023.
24. SciPy: Open-source software for mathematics, science, and engineering [Электронный ресурс]. – Режим доступа: https://scipy.org. – Дата доступа: 07.05.2023.
25. Matplotlib: Visualization with Python [Электронный ресурс]. – Режим доступа: https://matplotlib.org. – Дата доступа: 07.05.2023.