УДК 004.42

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ ОБРАБОТКИ ВИДЕОФРАГМЕНТОВ С ПОМОЩЬЮ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Шинкевич М. А., Щербаков А.С., Федосеев В.И.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Ревинская И.И. – ассистент кафедры ЭТТ

Аннотация. Данная статья описывает приложение на основе микросервисов для обработки видеофрагментов записи движения грудной клетки и живота во время дыхания с использованием искусственного интеллекта. Приложение было разработано с использованием фреймворка Nest.js и GraphQL для бэкенда, а также Python для компонента искусственного интеллекта. Система обеспечивает точный и эффективный анализ биомеханики дыхания, что может быть полезно для медицинских профессионалов при диагностике заболеваний дыхательной системы и проведении научных исследований.

Ключевые слова: биомеханика дыхания человека, микросервисы, искусственный интеллект

Введение. Существует несколько подходов к обработке видео для анализа дыхания. Один из них – это использование алгоритмов компьютерного зрения для выделения области грудной клетки (живота) на видео и анализа изменений в этой области, связанных с дыханием.

Метод анализа дыхания на основе обработки видеофрагментов имеет ряд преимуществ перед другими методами анализа дыхания. В данной статье представлены возможности создания ПО с помощью искусственного интеллекта для обработки видеофрагментов и актуальности данного ПО.

Основная часть. Задачей исследования является разработка программного обеспечения для обработки видеофрагментов дыхания с помощью языка программирования Python и библиотеки OpenCV. Так же для взаимодействия с пользователями требуется разработать API на языке Javascript и GraphqQL.

Программное обеспечение должно обладать следующими основными функциями:

- создание сессии пользователя;
- прием видеофрагментов со внешних клиентов;
- анализ данных с помощью искусственного интеллекта;
- возможность отправки данных о результате исследования клиенту;
- сохранение данных о действиях пользователя.

Для реализации данного приложения была предложена микросервисная архитектура построения приложения, включающая в себе несколько основных модулей взаимодействия:

- модуль API;
- модуль обучения ИИ;
- модуль вычисления с помощью ИИ;
- модуль базы данных.

Все эти модули взаимодействуют друг с другом и соблюдают следующие принципы:

– отделение ответственности (Separation of Concerns): каждый микросервис должен иметь четко определённую ответственность и выполнять только ту функцию, для которой он предназначен. Это позволяет легко масштабировать и изменять каждый микросервис отдельно;

– компонуемость (Composability): микросервисы должны быть спроектированы таким образом, чтобы они могли быть легко комбинированы друг с другом и использованы в разных сочетаниях для создания более сложных приложений;

– независимость от технологий (Technology Independence): микросервисы должны быть независимы от конкретных технологий, используемых в других микросервисах, чтобы изменения в одном микросервисе не приводили к изменениям в других микросервисах;

– гибкость (Flexibility): микросервисы должны быть гибкими и способными адаптироваться к изменяющимся требованиям приложения. Например, если требуется добавить новый функционал, то можно создать новый микросервис и добавить его в приложение без необходимости изменения других микросервисов;

– надежность (Reliability): микросервисы должны быть надежными и устойчивыми к сбоям, так как сбой одного микросервиса может привести к сбоям в других микросервисах;

– мониторинг (Monitoring): микросервисы должны быть мониторимы и предоставлять информацию о своем состоянии, чтобы было возможно быстро обнаружить и устранить проблемы в работе приложения [1].

На диаграмме (рисунок 1) ниже для наглядности представлена схема взаимодействия отдельных модулей между собой:

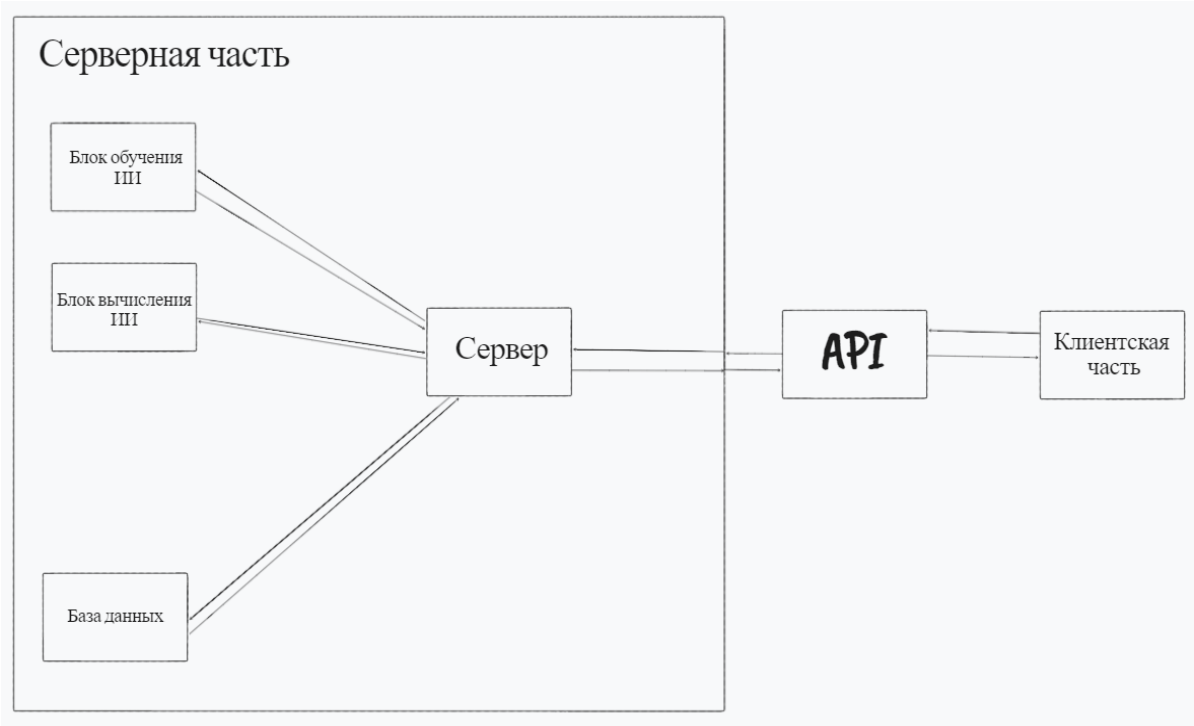

Рисунок 1 – Диаграмма взаимоотношения модулей ПО

Для разработки самого искусственного интеллекта нам понадобится язык программирования Python. *Python* – это интерпретируемый высокоуровневый язык программирования, который отличается простотой, удобством и выразительностью. Python используется во многих областях, в том числе в веб-разработке, научных вычислениях, анализе данных, машинном обучении и искусственном интеллекте.

Для написания серверной части ПО нам понадобятся следующие языки и библиотеки:

– *Nest.js*: это фреймворк для создания масштабируемых веб-приложений на Node.js. Он предоставляет множество инструментов для управления зависимостями, роутинга, логирования и других задач, связанных с разработкой веб-приложений;

– *GraphQL*: это язык запросов для API, который позволяет клиентам запрашивать только те данные, которые им нужны, и получать их в оптимальном формате. Nest.js предоставляет встроенную поддержку GraphQL, что делает его идеальным выбором для создания GraphQL API;

– *Apollo Server*: это библиотека для создания GraphQL API, которая может быть легко интегрирована с Nest.js. Она предоставляет ряд инструментов для валидации запросов, аутентификации и авторизации, кэширования и других задач;

– *TypeORM*: это ORM-фреймворк для работы с базами данных, который может быть использован с Nest.js. Он позволяет работать с различными базами данных, включая MySQL, PostgreSQL и SQLite, и предоставляет множество инструментов для управления таблицами, запросами и другими задачами, связанными с базами данных;

– *Flask*: это фреймворки для создания REST API на Python, которые могут быть использованы для интеграции модуля искусственного интеллекта с Nest.js через GraphQL;

– *TensorFlow и PyTorch*: это библиотеки машинного обучения, которые могут быть использованы для создания модели искусственного интеллекта;

– *numpy и pandas*: это библиотеки Python для работы с массивами и матрицами, которые могут быть полезны при обработке и анализе данных в модуле искусственного интеллекта;

– *scikit-learn*: это библиотека для машинного обучения, которая может быть полезна при реализации модели искусственного интеллекта;

– *gRPC*: это библиотека для создания распределенных систем на основе protobuf-сообщений, которая может быть использована для интеграции модуля искусственного интеллекта с бэкендом на Nest.js через GraphQL. gRPC предоставляет высокопроизводительный механизм для обмена сообщениями между приложениями на разных языках программирования и позволяет упростить процесс создания распределенных систем;

– *Docker*: это инструмент для создания, развертывания и запуска приложений в контейнерах. Он может быть использован для упрощения процесса развертывания приложения на разных окружениях и для обеспечения его переносимости;

– *Kubernetes*: это инструмент для оркестрации контейнеризированных приложений. Он предоставляет множество инструментов для автоматизации масштабирования, управления ресурсами и обеспечения отказоустойчивости приложения [2,3].

Данное приложение на основе микросервисной архитектуры представляет собой инновационное решение в области обработки видеофрагментов биомеханики дыхания. Оно имеет ряд преимуществ перед традиционными методами обработки данных, благодаря своей архитектуре, используемым технологиям и применению модуля искусственного интеллекта на Python.

Одним из главных преимуществ микросервисной архитектуры является масштабируемость, что позволяет гибко управлять приложением в зависимости от нагрузки и потребностей. Также, благодаря разбиению приложения на микросервисы, возможно изменять и расширять отдельные компоненты независимо друг от друга, что обеспечивает высокую гибкость и удобство в управлении.

Кроме того, микросервисное приложение обладает отказоустойчивостью, что обеспечивается его распределенной архитектурой. В случае отказа одного из компонентов, приложение продолжает работу, что гарантирует высокую доступность и надежность.

Использование библиотек и фреймворков, таких как TensorFlow или PyTorch, позволяет обрабатывать видеофрагменты быстрее, чем при использовании обычных алгоритмов обработки данных. Также, удобный интерфейс GraphQL упрощает взаимодействие с приложением и повышает его доступность для пользователей.

Использование Docker и Kubernetes упрощает процесс развертывания и масштабирования приложения на различных окружениях. Это позволяет ускорить процесс разработки и деплоя, а также уменьшить количество ошибок, связанных с различными конфигурациями и настройками.

Использование модели машинного обучения позволяет получать более точные результаты при обработке видеофрагментов биомеханики дыхания. Это возможно благодаря использованию алгоритмов обучения и оптимизации, которые позволяют извлекать максимально полезную информацию из видеофрагментов и делать более точные выводы о состоянии дыхательной системы [4].

Заключение. Разработка микросервисного приложения для обработки видеофрагментов дыхания человека с помощью искусственного интеллекта является эффективным решением для медицинской диагностики. Представленное приложение, основанное на бэкенде Nest.js и фронтенде GraphQL, позволяет быстро и эффективно обрабатывать большие объемы видеоданных, а также гибко настраивать и расширять его функциональность.

Применение искусственного интеллекта для анализа видеоданных дыхания позволяет получать более точные результаты, чем при использовании традиционных методов анализа. Также, микросервисная архитектура приложения позволяет масштабировать его в зависимости от потребностей, обеспечивать высокую отказоустойчивость и гибкость.

Данное приложение может быть полезным инструментом для медицинских учреждений и специалистов в области диагностики заболеваний дыхательной системы. Оно позволяет быстро и точно оценивать функциональное состояние дыхательной системы пациента, определять наличие патологий и эффективность лечения. Также, применение данного приложения может повысить эффективность и качество проводимых медицинских исследований в области дыхательной системы.

Список литературы

1. *BioMed Central [Электронный ресурс]. – Режим доступа:* https://www.biomedcentral.com/ *– Дата доступа: 24.11.2022.*
2. *ResearchGate [Электронный ресурс]. – Режим доступа:* https://www.researchgate.net/ *– Дата доступа: 24.11.2022.*
3. *ACM Digital Library [Электронный ресурс]. – Режим доступа:https://dl.acm.org/ – Дата доступа: 25.11.2022.*
4. *PubMed [Электронный ресурс]. – Режим доступа: https://pubmed.ncbi.nlm.nih.gov/ – Дата доступа: 25.11.2022.*

UDC 004.42

SOFTWARE FOR PROCESSING VIDEO FRAGMENTS USING ARTIFICIAL INTELLIGENCE

Shynkevich M. A., Scherbakov A.S., Fedoseev V.I.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Revinskaya I.I. – assistant of Department of ETT

Annotation. This article presents a microservices-based application for processing video fragments of human breathing mechanics using artificial intelligence. The application was built using the Nest.js framework and GraphQL for the backend, and Python for the AI component. The system provides accurate and efficient analysis of breathing patterns, which is useful for medical professionals in diagnosing respiratory conditions.

Keywords: human breathing mechanics, microservices, artificial intelligence.