МРНТИ 20.19.27 https://doi.org/10.58805/kazutb.v.3.20-151

АНАЛИТИКА НАУЧНЫХ ТЕКСТОВ НА ОСНОВЕ РАСПРЕДЕЛЕННЫХ ФРЕЙМВОРКОВ ПАРАЛЛЕЛЬНОЙ ОБРАБОТКИ ДАННЫХ

Г.Ж. Шуйтенов[1], С.А. Алтынбек[2], А.С. Тургинбаева[3*], С.Э. Сантеева[3]

[1]Esil University, Астана, Казахстан,

[2]Казахский университет технологии и бизнеса, Астана, Казахстан,

[3]Евразийский национальный университет им. Л. Н. Гумилева, Астана, Казахстан,

e-mail: tasheart@mail.ru

Статья рассматривает разработку интеллектуальной системы параллельного анализа неструктурированных данных на основе распределенного фреймворка Apache Spark. А также формирование математического аппарата для реализации быстрых алгоритмов анализа научных текстов на естественном языке с применением методов теории вероятностей и статистики, теории информации и машинного обучения. Apache Spark - это распределенный фреймворк для обработки больших данных и аналитики. Он обеспечивает быстрый и универсальный движок для крупномасштабной обработки данных, позволяющий пользователям выполнять параллельные вычисления в распределенных кластерах. Неструктурированные данные относятся к данным, которые не имеют предопределенной структуры, таким как текст, изображения, видео и публикации в социальных сетях. Анализ неструктурированных данных - сложная задача, поскольку для этого требуется извлекать значимую информацию из данных, которые нелегко организовать в строки и столбцы. Разработка интеллектуальной системы для параллельного анализа неструктурированных данных с использованием Apache Spark включает в себя несколько этапов. Во-первых, неструктурированные данные должны быть загружены в Spark framework. Это можно сделать с помощью различных источников данных, таких как распределенная файловая система Hadoop (HDFS), Amazon S3 или любая другая система хранения, поддерживаемая Spark. Интеллектуальная система может быть разработана с использованием таких языков программирования, как Scala, Java или Python, которые имеют привязки Spark и предоставляют API для взаимодействия с Spark framework. Эти API-интерфейсы позволяют разработчикам определять конвейеры передачи данных, настраивать параметры параллельной обработки и выполнять задачи анализа.

Ключевые слова: параллельный анализ, научный текст, большие данные, неструктурированные данные, обработка данных, Apache Spark, распределенные вычисления, математический аппарат.

ANALYTICS OF SCIENTIFIC TEXTS BASED ON DISTRIBUTED PARALLEL DATA PROCESSING FRAMEWORKS

G.Zh. Shuitenov[1], S.A. Altynbek[2], A.S. Turginbayeva[3*], С.Э. Сантеева[3]

[1]Esil University, Astana, Republic of Kazakhstan,

[2]Kazakh University of Technology and Business, Astana, Kazakhstan,

[3]L.N. Gumilyov Eurasian National University, Astana, Kazakhstan,

e-mail: tasheart@mail.ru

The article considers the development of an intelligent system for parallel analysis of unstructured data based on the Apache Spark distributed framework. As well as the formation of a mathematical apparatus for the implementation of fast algorithms for the analysis of scientific texts in natural language using methods of probability theory and statistics, information theory and machine learning. Apache Spark is a distributed framework for big data processing and analytics. It provides a fast and versatile engine for large-scale data processing, allowing users to perform parallel computing in distributed clusters. Unstructured data refers to data that does not have a predefined structure, such as text, images, videos, and social media posts. Analyzing unstructured data is a difficult task because it requires extracting meaningful information from data that is not easy to organize into rows and

columns. The development of an intelligent system for parallel analysis of unstructured data using Apache Spark involves several stages. First, unstructured data must be loaded into Spark framework. This can be done using various data sources, such as the Hadoop Distributed File System (HDFS), Amazon S3, or any other storage system supported by Spark. An intelligent system can be developed using programming languages such as Scala, Java or Python, which have Spark bindings and provide APIs for interacting with the Spark framework. These APIs allow developers to define data transfer pipelines, configure parallel processing parameters, and perform analysis tasks.

Keywords: parallel analysis, scientific text, big data, unstructured data, data processing, Apache Spark, distributed computing, mathematical apparatus.

ТАРТЫЛҒАН ПАРАЛЕЛЬДІ ДЕРЕКТЕРДІ ӨҢДЕУ ФРАММАЛАРЫНЫҢ НЕГІЗІНДЕГІ ҒЫЛЫМИ МӘТІНДІ ТАЛДАУ

Г.Ж. Шуйтенов[1], С.А. Алтынбек[2], А.С. Тургинбаева[3*], С.Э. Сантеева[3]

[1]Esil University, Астана, Қазақстан,

[2]Қазақ технология және бизнес университеті, Астана, Қазақстан,

[3]Л.Н. Гумилев атындағы Еуразия ұлттық университеті, Астана, Қазақстан,

e-mail: tasheart@mail.ru

Мақалада Apache Spark таратылған фраммаларға негізделген құрылымдалмаған деректерді параллель талдаудың интеллектуалды жүйесін әзірлеу қарастырылады. Сондай-ақ ықтималдық теориясы мен статистика әдістерін, ақпарат теориясын және машиналық оқытуды қолдана отырып, табиғи тілдегі ғылыми мәтіндерді талдаудың жылдам алгоритмдерін жүзеге асыруға арналған математикалық аппаратты қалыптастыру. Apache Spark - бұл үлкен деректерді өңдеуге және талдауға арналған таратылған құрылым. Ол пайдаланушыларға үлестірілген кластерлерде параллель есептеулер жүргізуге мүмкіндік беретін кең ауқымды деректерді өңдеу үшін жылдам және жан-жақты қозғалтқышты қамтамасыз етеді. Құрылымдалмаған деректер мәтін, суреттер, бейнелер және әлеуметтік медиа жазбалары сияқты алдын ала анықталған құрылымы жоқ деректерге жатады. Құрылымдалмаған деректерді талдау қиын мәселе болып табылады, өйткені ол жолдар мен бағандарға оңай ұйымдастырылмайтын деректерден маңызды ақпаратты алуды талап етеді. Apache Spark көмегімен құрылымдалмаған деректерді параллель талдауға арналған интеллектуалды жүйені әзірлеу бірнеше кезеңдерді қамтиды. Біріншіден, құрылымдалмаған деректер Spark framework-қа жүктелуі керек. Мұны таратылған Hadoop файлдық жүйесі (HDFS), Amazon S3 немесе Spark қолдайтын кез келген басқа сақтау жүйесі сияқты әртүрлі деректер көздерімен жасауға болады. Ақылды жүйені Scala, Java немесе Python сияқты бағдарламалау тілдерін қолдана отырып жасауға болады, олар Spark байланыстары бар және Spark Framework-пен өзара әрекеттесу үшін API ұсынады. Бұл API интерфейстері әзірлеушілерге деректер құбырларын анықтауға, параллельді өңдеу параметрлерін реттеуге және талдау тапсырмаларын орындауға мүмкіндік береді.

Түйін сөздер: параллельді талдау, ғылыми мәтін, үлкен деректер, құрылымдалмаған деректер, деректерді өңдеу, Apache Spark, бөлінген есептеулер, математикалық аппарат.

Введение. Совершенно очевидно, что в Казахстане идет большой интерес к научным исследованиям, Министерство науки и высшего образования ежегодно увеличивает грантовое финансирование научных проектов, в течение следующих лет ожидается увеличение количества выпускников бакалавриата, магистратуры и докторантуры, и поэтому разработка интеллектуальной информационной системы поиска нарушений в научных текстах крайне необходима. Статья описывает подход создания исследовательской системы для анализа научных текстов на естественном языке с применением алгоритмов параллельного машинного обучения.

Материалы и методы. Цифровизация стала важным фактором для развития системы высшего образования, а также для различных областей деятельности. Внедрение передовых цифровых технологий и информационных систем в учебный и административный процессы имеет несколько значимых преимуществ:

1. Улучшение эффективности и производитель-

ности: Использование цифровых инструментов позволяет автоматизировать рутинные задачи, упрощает процессы управления и администрирования, что в итоге улучшает производительность и позволяет сотрудникам и студентам более эффективно использовать свое время.

2. Анализ и принятие обоснованных решений: Собираемые и обрабатываемые цифровые данные предоставляют ценную информацию для анализа и принятия обоснованных управленческих решений. Агрегированные данные могут помочь учебным заведениям выявлять тенденции, прогнозировать потребности и оптимизировать свою деятельность.

3. Поддержка цифрового мышления и цифровых навыков: Использование цифровых технологий в образовательном процессе стимулирует студентов и сотрудников развивать цифровое мышление и осваивать цифровые навыки, которые становятся все более важными в современном обществе и на рынке труда.

4. Улучшение доступности и гибкости: Цифровые технологии позволяют предоставлять образование в онлайн-формате, что увеличивает его доступность для студентов из разных регионов и стран, а также позволяет обучаться в удобное время и темпе.

В целом, цифровизация высшего образования - это ключевой фактор для современного развития учебных заведений, что позволяет им более эффективно выполнять свои задачи и оставаться актуальными в современном образовательном и научном сообществе.

Обсуждение и результаты. Лавинообразное развитие информационных технологий в мире заставляет вузы идти в ногу с этими трендами, обновлять образовательные программы и вести работы по внедрению прогрессивных ИТ подходов в университетах. Так на текущий момент Казахстан уже активно применяет технологии дистанционного обучения, разработана нормативная база, на вебинарных площадках ведутся онлайн занятия, внедряются технологии смешанного обучения и новые подходы онлайн обучения [1] Но технологии не стоят на месте, так на текущий период быстро набирающая четвертая «образовательная» революция требует пересмотра традиционного образовательного подхода. Недалек тот момент, когда вузы будут испытывать потребность в таких технологиях, которые способны принимать «машинные решения» и делать выводы о необходимости изменения тех или иных подходов и действий в образовании. Одной из таких технологий может стать технология оперирования большими данными (Big Data) [2]. Оперирование большими данными (Big Data) в образовании - это технология аналитики образовательной системы, включающей измерение, сбор, анализ и представление структурированных и неструктурированных данных огромных объемов об обучающихся и образовательной среде с целью понимания особенностей функционирования и развития образовательной системы [3].

Уже предвидя такое положение дел, авторы статьи предлагают использование последних технологий параллельной обработки больших неструктурированных данных, с применением таких распределенных компонентов как HDFS, MapReduce [4], на языках высокого уровня Java, Python, где будут реализованы быстрые алгоритмы на вычислительном кластере.

Целью использования технологий обработки больших данных является обработка неструктурированных данных, к коим и относятся научные тексты, где помимо разработки методов математического анализа текста необходимо использовать передовые IT технологии, в частности системы параллельной обработки, реализация алгоритмов поиска «коротких маршрутов» с использованием распределенных фреймворков потоковой обработки данных. На основе базовых компонентов HDFS, MapReduce, компонентами управления балансировкой нагрузки, обеспечения безопасности системы, алгоритмами репликации данных и журналирования вычислений можно сформировать блочную архитектура обработки неструктурированных данных независимыми Tasker-ами на вычислительных нодах. Для организации самих параллельных алгоритмов обработки научных текстов можно использовать распределенные фреймворки, работающие на распространенном формате для обмена данных MPI (Message Passing Interface), а организация параллельных вычислений реализуется на широко используемой технологии использования распределенных вычислений [5]. Очевидными инновационными идеями такого подхода для анализа научных текстов является:

- построение математической модели для анализа научных текстов и ее реализация на распределенной платформе.
- создание технологии, использующей новый тренд в обработке данных, в частности параллельные вычисления, распределенное хранение данных и их обработка.

- возможность горизонтального масштабирования за счет подключения в вычислительный кластер недорогих вычислительных ресурсов;
- внедрение в организации образования системы аналитики и визуализации данных по научным исследованиям [6-7].

На текущий момент Apache Hadoop и Apache Spark являются двумя популярными инструментами для обработки и анализа больших объемов данных, включая анализ текстов. Резюмируя особенности каждого инструмента, статья имела целью определить их отличия, и понять, что может быть эффективнее для анализа текстов.

Hadoop: Hadoop является фреймворком для распределенной обработки больших объемов данных, основанным на модели MapReduce. Основные компоненты Hadoop - это Hadoop Distributed File System (HDFS) для хранения данных и MapReduce для обработки данных [8].

Преимущества:

Масштабируемость: Hadoop хорошо подходит для обработки и хранения очень больших объемов данных. Он может работать на кластере из десятков или сотен узлов.

Устойчивость к отказам: Hadoop обладает механизмами восстановления данных и обработки отказов узлов, что делает его надежным при обработке больших объемов данных.

Недостатки:

Относительно низкая производительность для небольших задач: Hadoop имеет высокую стоимость запуска задачи из-за необходимости записи на диск промежуточных результатов после каждого шага MapReduce.

Сложность программирования: Разработка задач в Hadoop на языке Java может быть сложной и требовать больше усилий.

Spark: Spark также является фреймворком для распределенной обработки данных, но, в отличие от Hadoop, использует более эффективную модель обработки данных, называемую «Resilient Distributed Dataset» (RDD). Это позволяет Spark обрабатывать данные в оперативной памяти, что делает его значительно быстрее для некоторых типов задач.

Преимущества:

Высокая производительность: Spark может быть значительно быстрее Hadoop для некоторых типов задач, благодаря обработке данных в памяти.

Простота использования: Spark предоставляет API на нескольких языках программирования, таких как Scala, Java, Python и SQL, что делает его более доступным для разработчиков.

Недостатки:

Потребление памяти: Использование оперативной памяти может быть проблемой для Spark при обработке очень больших объемов данных, что может привести к нехватке памяти и снижению производительности.

Низкая устойчивость к отказам: В отличие от Hadoop, Spark не имеет такой же уровень отказоустойчивости, что может быть важно при обработке критических данных.

Какой из них эффективнее для анализа текстов зависит от конкретных требований и характера вашей задачи. Если у вас есть огромные объемы данных, требующие высокой степени отказоустойчивости, Hadoop может быть предпочтительнее. Если же ваш анализ текстов фокусируется на относительно небольших объемах данных, и вы хотите достичь более высокой производительности, Spark может быть более подходящим выбором. Дополнительно Spark также имеет разнообразные библиотеки и инструменты для машинного обучения (MLlib), графовых вычислений (GraphX) и обработки потоков данных (Spark Streaming), делая его мощным инструментом для различных задач. Исходя из вышеизложенного вывода мы делаем заключение, что для анализа небольших по размеру научных текстов более предпочтительным выглядит использование Apache Spark.

Как мы отметили выше, в части использования фреймворков распределенной обработки данных мы будем использовать довольно популярный Big Data фреймворк с открытым исходным кодом для распределённой пакетной и потоковой обработки неструктурированных и слабоструктурированных данных, входящий в экосистему проектов Hadoop - Apache Spark [9-10].

В отличие от классического обработчика ядра Apache Hadoop с двухуровневой концепцией MapReduce на базе дискового хранилища, Spark использует специализированные примитивы для рекуррентной обработки в оперативной памяти. Благодаря этому многие вычислительные задачи реализуются на этом фреймворке значительно быстрее. Например, возможность многократного доступа к загруженным в память пользовательским данным позволяет эффективно работать с алгоритма-

ми машинного обучения (Machine Learning). Spark может работать как в среде кластера Hadoop под управлением YARN, так и без компонентов ядра Hadoop, например, на базе системы управления кластером Mesos. Spark поддерживает несколько популярных распределённых систем хранения данных (HDFS, OpenStack Swift, Cassandra, Amazon S3) и языков программирования (Java, Scala, Python, R), предоставляя для них API-интерфейсы.

С целью анализа научных текстов нам необходимо для начала провести процедуру установки распределенного фреймворка Apache Spark. Загрузить последнюю версию Spark, можно с официального сайта https://spark.apache.org/downloads.html. После скачивания файла его необходимо распаковать, следуя инструкциям, и если процедура прошла успешно, то введя в терминал команду $spark-shell вы получите следующий вывод на экран как показано на рисунке 1.

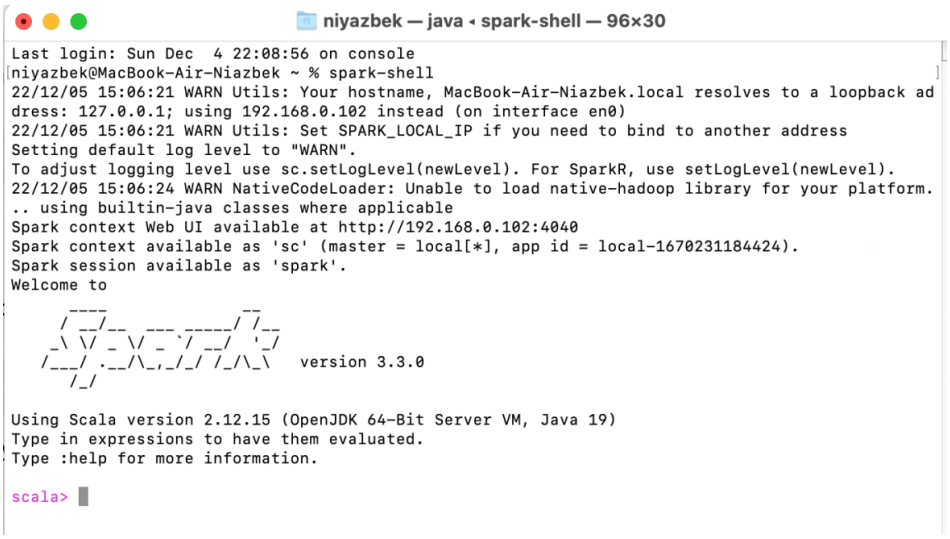

Рис. 1 - Интерфейс установки

Для реализации поставленной задачи обработки научных текстов будет проводиться контент-анализ с использование методов прикладной математики, в частности статистического анализа и теории вероятностей. Основные идеи проекта базировались на основе методов теории вероятностей и статистики, а также теории информации, в том числе Гауссовского распределения и Байесовской формулы вероятностей, которые позволили построить надежный алгоритм для определения методов сокрытия заимствований в научных текстах и нарушения принципов Академической Честности. Одним из методов исследование текстов является анализ энтропии языков, используемых при написании научных работ (казахский, русский и английский). Программа находит аномальное поведение символов в научном тексте и соответственно находит нарушение распределения символов в тексте с использованием систем параллельной обработки данных.

Spark и Scala тесно связаны друг с другом, и Scala является одним из основных языков программирования для разработки приложений на платформе Spark [11-12]. Scala - это статически типизированный язык программирования, который работает на платформе Java Virtual Machine (JVM) и совместим с Java. Он предоставляет множество функциональных и объектно-ориентированных возможностей. Scala является компилируемым языком программирования, что делает его быстрее по сравнению с интерпретируемыми языками, такими как Python. Scala предоставляет множество функциональных возможностей, таких как высокоуровневые функции, замыкания и паттерн-матчинг, которые позволяют разработчикам писать более компактный и выразительный код. И главное возможности параллелизма и распределенных вычислений в Spark хорошо сочетаются с мощью Scala при написании функциональных программ. Исходя из вышеизложенных выводов мы реализуем наш проект на этом языке с использованием специализирован-

ной среда разработки Intelij IDEA. Для облегчения установки необходимых библиотек используется Apache Maven - фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM, являющемся подмножеством XML. Проект Maven издаётся сообществом Apache Software Foundation, где формально является частью Jakarta Project. После создания и открытия проекта, импортируем библиотеки/зависимости в проект так как на рисунке 2, добавляя их в pom.xml, нам понадобятся библиотеки apache spark core и apache spark sql. Зависимости можно найти на официальном сайте maven https://mvnrepository.com/artifact/org.apache.spark.

```
<dependencies>
    <!-- https://mvnrepository.com/artifact/org.apache.spark/spark-core -->
    <dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-core_2.13</artifactId>
        <version>3.3.0</version>
    </dependency>

    <!-- https://mvnrepository.com/artifact/org.apache.spark/spark-sql -->
    <dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-sql_2.13</artifactId>
        <version>3.3.0</version>
        <scope>provided</scope>
    </dependency>
```

Рис. 2 - Импорт библиотек в проект

Создаем основной main класс, импортируем Spark и инициализируя/настраивая конфигурации (Рис 3).

```
public class TextAnalysis {

    public static void main(String[] args) {
        SparkConf sparkConf = new SparkConf();
        sparkConf.setAppName("Spark WordCount example using Java");
        sparkConf.setMaster("local");
        JavaSparkContext sc = new JavaSparkContext(sparkConf);
```

Рис. 3 - Импорт Spark

Инициализируем наш текстовой файл для дальнейшей работы.

Для дальнейшей работы переводим тип данных Spark RDD в привычную java Map. Далее считаем буквы и проценты повторений, выводим как видно на рисунке 4.

```
Map<String,Integer> c = counts.collectAsMap();
int counter = 0;
double per;
double per100 = 0;

for (var f : c.values()){
    counter = f + counter;
}

for (var f : c.entrySet()){
    System.out.print("Буква " + f.getKey() + " | Процент ");
    per = (double)(f.getValue()*100)/counter;
    per100 = per100 + per;
    System.out.println(String.format("%.2f",per) + " %");
}

System.out.println("---------------");
System.out.println(c);
System.out.println(counter);
System.out.println(per100);
System.out.println("---------------");
```

Рис. 4 - Вывод данных

```
Буква a | Процент 8,09 %
Буква b | Процент 1,36 %
Буква c | Процент 2,40 %
Буква d | Процент 4,66 %
Буква e | Процент 12,39 %
Буква f | Процент 2,16 %
Буква g | Процент 2,02 %
Буква h | Процент 6,59 %
Буква i | Процент 6,83 %
Буква j | Процент 0,10 %
Буква k | Процент 0,80 %
Буква l | Процент 3,80 %
Буква m | Процент 2,43 %
Буква n | Процент 7,25 %
Буква o | Процент 7,58 %
Буква p | Процент 1,77 %
Буква q | Процент 0,09 %
Буква r | Процент 5,82 %
Буква s | Процент 6,42 %
Буква t | Процент 8,89 %
Буква u | Процент 2,57 %
Буква v | Процент 1,06 %
Буква w | Процент 2,33 %
Буква x | Процент 0,16 %
Буква y | Процент 1,82 %
Буква z | Процент 0,09 %
---------------
```

Рис. 5 - Вывод итоговых данных

Запускаем программу и в консоли будут выведены все значения (также Map сразу сортирует наш список в алфавитном порядке). В качестве примера для анализа был взят текст художественного произведения «Война и мир», 1 том на английском языке. Скорость обработки неструктурированного текста с использованием подходов параллельной обработки на основе распределенного фреймворка Apache Spark оказалось очень быстрой и в течении нескольких секунд программный комплекс вывел следующие итоговые данные, как показывает наша программа на рисунке 5.

Выводы. Как видно из полученных данных, аномального поведения текста мы не наблюдаем, частотный анализ и распределение символов вполне коррелирует с характерным распределением символов для английских текстов [13]. Это только один из вариантов анализа, который будет внедрен нами и дальнейшее развитие предполагает реализацию нами следующих видов анализов:

- графематический анализ-выделение слов, цифровых комплексов, формул и т.д.;
- синтаксический анализ-построение дерева зависимостей всего предложения;
- семантический анализ-построение семантического графа текста, сопряженного с необходимостью обработки больших объемов распределенных данных;

Система будет обеспечивать:

- возможность параллельной загрузки и распределенного хранения анализируемых научных текстов;
- анализ текста посредством реализации математических методов и алгоритмов обработки научного текста
- реализацию экспертной системы с последующим просмотром итоговых результатов анализа текста;
- визуализацию результатов работы системы, ее компонентов и формирование отчетов;
- оптимальное и приемлемое время системы, за счет распараллеливания вычислений и быстрых алгоритмов;
- многоязычный интерфейс и анализ текста на основных мировых языках.

В части дальнейшего развития проекта авторы ориентируются на применение методов data mining для позиционирования идентифицированных сведений в многомерном пространстве аналитических измерений (предметная область технологии/прорывного направления; стадия жизненного цикла; уровень потенциала; использованные методы и средства и т.п.). Будут применяться морфологические и синтаксические анализаторы текста для поиска аномалий в тексте с помощью математического аппарата, функционального анализа, включая свойства метрических пространств, вычисление топологий для естественных языков. Будет применять-

ся нечеткая логика для поиска по нечеткому критерию, использования функций принадлежности и лингвистических переменных для нечеткого вывода. Будет создан новые next-generation быстрые поисковые алгоритмы на основе методов кластеризации и последующего синтеза.

Научно-исследовательская работа выполняется в рамках ГФ Министерством науки и высшего образования Республики Казахстан AR19677733 по теме «Разработка интеллектуальной распределенной системы параллельного анализа научных текстов» на 2023-2025 гг.

Литература

1. Abdymanapov, S., Muratbekov, M., Altynbek, S., Barlybayev, A. Fuzzy expert system of information security risk assessment on the example of analysis Learning Management Systems. // IEEE Access,2021, - 9.- pp. 156556-156565. DOI: 10.1109/ACCESS.2021.3129488.

2. Boranbayev, A., Shuitenov, G., Boranbayev, S. The Method of Analysis of Data from Social Networks Using Rapidminer //Advances in Intelligent Systems and Computing, 2020 - 1229 AISC, pp. 667-673.

3. Boranbayev, A., Shuitenov, G., Boranbayev, S. The method of data analysis from social networks using apache Hadoop // Advances in Intelligent Systems and Computing. - 2018. - 558. pp. 281-288.

4. Altynbek, S., Begehr, H. A pair of rational double sequences. //Georgian Mathematical Journal. - 2022. - 29(2), pp. 163-166. https://doi.org/10.1515/gmj-2021-2119.

5. A. Barlybayev; Z. Kaderkeyeva; G. Bekmanova; A. Sharipbay; A. Omarbekova; S. Altynbek. Intelligent System for Evaluating the Level of Formation of Professional Competencies of Students. // IEEE Access. - 2020. - 8, pp. 58829-58835, 9027836. DOI: https://doi.org/10.1109/ACCESS.2020.2979277

6. Silen D.: Fundamentals of Data Science and Big Data. Python and Data Science. - Peter-Trend Publishing House, 2018. - 336 p. ISBN 978-5-4461-0944-9

7. Paul Deitel, Harvey Deitel Python: Artificial Intelligence, Big Data and Cloud Computing - LitRes Publishing House, 2020. - 864 p. ISBN 978-5-4461-1432-0

8. Hope T., Resheff Y., Lieder I. Learning TensorFlow. // Boston: Oreilly. -2017. 242 pp.

9. Goodfellow I., Bengio Y., Courville A. Deep Learning. // Cambridge, MA: MIT Press, -2016. 800 pp.

10. Dua D. and Graff C. UCI Machine Learning Repository. // Irvine, CA: University of California, School of Information and Computer Science. - 2019.

11. Holden Karau, Andy Konwinski, Patrick Wendell, Matei Zaharia. Learning Spark: Lightning-Fast Big Data Analysis. // NYU: O'reilly. - 2015. 274 pp.

12. https://spark.apache.org/ Date of address - 25.08.2023

13. https://wiki5.ru/wiki/Letter_frequency/ Date of address - 25.08.2023

References

1. Abdymanapov, S., Muratbekov, M., Altynbek, S., Barlybayev, A. Fuzzy expert system of information security risk assessment on the example of analysis Learning Management Systems. // IEEE Access,2021, - 9.- pp. 156556-156565. DOI: 10.1109/ACCESS.2021.3129488.

2. Boranbayev, A., Shuitenov, G., Boranbayev, S. The Method of Analysis of Data from Social Networks Using Rapidminer //Advances in Intelligent Systems and Computing, 2020 - 1229 AISC, pp. 667-673.

3. Boranbayev, A., Shuitenov, G., Boranbayev, S. The method of data analysis from social networks using apache Hadoop // Advances in Intelligent Systems and Computing. - 2018. - 558. pp. 281-288.

4. Altynbek, S., Begehr, H. A pair of rational double sequences. //Georgian Mathematical Journal. - 2022. - 29(2), pp. 163-166. https://doi.org/10.1515/gmj-2021-2119.

5. A. Barlybayev; Z. Kaderkeyeva; G. Bekmanova; A. Sharipbay; A. Omarbekova; S. Altynbek. Intelligent System for Evaluating the Level of Formation of Professional Competencies of Students. // IEEE Access. - 2020. - 8, pp. 58829-58835, 9027836. DOI: https://doi.org/10.1109/ACCESS.2020.2979277

6. Silen D.: Fundamentals of Data Science and Big Data. Python and Data Science. - Peter-Trend Publishing House, 2018. - 336 p. ISBN 978-5-4461-0944-9

7. Paul Deitel, Harvey Deitel Python: Artificial Intelligence, Big Data and Cloud Computing - LitRes Publishing House, 2020. - 864 p. ISBN 978-5-4461-1432-0

8. Hope T., Resheff Y., Lieder I. Learning TensorFlow. // Boston: Oreilly. -2017. 242 pp.

9. Goodfellow I., Bengio Y., Courville A. Deep Learning. // Cambridge, MA: MIT Press, -2016. 800 pp.

10. Dua D. and Graff C. UCI Machine Learning Repository. // Irvine, CA: University of California, School of Information and Computer Science. - 2019.

11. Holden Karau, Andy Konwinski, Patrick Wendell, Matei Zaharia. Learning Spark: Lightning-Fast Big Data Analysis. // NYU: O'reilly. - 2015. 274 pp.

12. https://spark.apache.org/ Date of address - 25.08.2023

13. https://wiki5.ru/wiki/Letter_frequency/ Date of address - 25.08.2023

Сведения об авторах

Шуйтенов Г.Ж. – проректор по цифровизации, Esil University, Астана Казахстан, e-mail: g.shuitenov@mail.ru;

Алтынбек С.А. – PhD, проректор по науке, Казахский университет технологии и бизнеса, Астана, Казахстан, e-mail: serik_aa@bk.ru;

Сантеева С.Ә. – PhD, и.о. доцента, Евразийский национальный университет им. Л.Н. Гумилева, Астана, Казахстан, e-mail: saya_santeeva@mail.ru;

Тургинбаева А.С. – магистр, старший преподаватель, Евразийский национальный университет им. Л.Н. Гумилева, Астана, Казахстан, e-mail: tasheart@mail.ru.

Information about the authors

Shuitenov G. Zh. – Vice-Rector for Digitalization, Esil University, Astana, Kazakhstan, e-mail: g.shuitenov@mail.ru;

Altynbek S. A. – PhD, Vice-Rector for Science, Kazakh University of Technology and Business, Astana, Kazakhstan, e-mail: serik_aa@bk.ru;

Santeyeva S. A. – PhD, Acting Associate Professor, L.N. Gumilyov Eurasian National University, Astana, Kazakhstan, e-mail: saya_santeeva@mail.ru;

Turginbayeva A. S. – Master's degree, senior lecturer, L.N. Gumilyov Eurasian National University, Astana, Kazakhstan, e-mail: tasheart@mail.ru.