DOI: 10.15514/ISPRAS-2023-35(2)-5

Ферма данных: информационная система сбора, хранения и обработки неструктурированных данных из разнородных источников

С.П. Левашкин, ORCID: 0000-0001-8687-2704 <s.levashkin@psuti.ru>
К.Н. Иванов, ORCID: 0000-0002-6248-9564 <k.ivanov@psuti.ru>
С.В. Кушуков, ORCID 0009-0004-0270-5853 <s.kushukov@psuti.ru>

Поволжский государственный университет телекоммуникаций и информатики, 443086, Россия, г. Самара, Московское ш., 77

Аннотация. Представлена оригинальная информационная система «ферма данных». Сегодня успешное применение алгоритмов искусственного интеллекта, прежде всего глубокого обучения на основе нейронных сетей, практически полностью зависит от наличия данных. И чем больше объем этих данных, тем лучше результаты работы алгоритмов. Хорошо известны примеры таких алгоритмов от Facebook, Google, Microsoft, Yandex и др. Данные должны содержать как обучающую выборку, так и тестируемую. Причем, данные должны быть хорошего качества и обладать определенной структурой, в идеале быть размеченными, чтобы алгоритмы обучения работали адекватно. Это представляет серьезную проблему, требующую огромных вычислительных и человеческих ресурсов. Именно решению этой проблемы посвящена данная статья. На сегодня ферма данных представляет из себя довольно сложную информационную систему, построенную по модульному принципу, схожую с известным конструктором «Лего». Отдельными модулями системы являются различные современные алгоритмы, технологии и целые библиотеки искусственного интеллекта, а все вместе они призваны автоматизировать процесс получения и структурирования качественных больших данных в различных предметных областях. Система была протестирована на данных по COVID-19 в регионах России и странах мира. Кроме того, был разработан удобный интерфейс визуализации данных, собранных и обработанных на ферме. Это дает возможность проводить наглядные численные эксперименты компьютерного моделирования и сравнивать их с реальными данными, превращая ферму в интеллектуальную информационную систему поддержки принятия решений.

Ключевые слова: интеллектуальная информационная система; ферма данных; большие данные; обработка данных; визуализация данных; компьютерное моделирование

Для цитирования: Левашкин С.П., Иванов К.Н., Кушуков С.В. Ферма данных: информационная система сбора, хранения и обработки неструктурированных данных из разнородных источников. Труды ИСП РАН, том 35, вып. 2, 2023 г., стр. 57-72. DOI: 10.15514/ISPRAS–2023–35(2)–5

Благодарности: Ферма данных была разработана в рамках мегагранта Минобрнауки. Авторы благодарят коллег РФЯЦ – ВНИИТФ (Снежинск) и ИПМ им. М.В. Келдыша (Москва) за сотрудничество.

Data farm: Information system for collecting, storing and processing unstructured data from heterogeneous sources

S.P. Levashkin, ORCID: 0000-0001-8687-2704 <s.levashkin@psuti.ru>
K.I. Ivanov, ORCID: 0000-0002-6248-9564 <k.ivanov@psuti.ru>
S.V. Kushukov, ORCID: 0009-0004-0270-5853 <s.kushukov@psuti.ru>

Povolzhskiy State University of Telecommunications and Informatics, 77, Moskovskoe sh., Samara, 443086, Russia

Abstract. The original information system «data farm» is presented. Today, the successful application of artificial intelligence algorithms, primarily deep learning based on artificial neural networks, almost completely depends on the availability of data. And the larger the amount of these data (big data), the better are the results of the algorithms execution. There are well-known examples of such algorithms from Facebook, Google, Microsoft, Yandex, etc. The data must contain both the training sample and the test one. Moreover, the data must be of good quality and have a certain structure, ideally, be labeled in order for the learning algorithms to work adequately. This is a serious problem requiring huge computational and human resources. This paper is dedicated to solve this problem. Today data farm is a rather complex information system built on a modular basis, similar to the well-known Lego constructor. Separate modules of the system are various modern algorithms, technologies and entire libraries of artificial intelligence, and all together they are designed to automate the process of obtaining and structuring high-quality big data in various subject domains. The system has been tested on data of COVID-19 in regions of Russia and countries around the world. In addition, a user-friendly interface for visualizing collected and processed on the farm data was developed. This makes it possible to conduct visual numerical experiments of computer simulation and compare them with real data, turning the farm into an intelligent decision support information system.

Keywords: intelligent information system; data farm; big data; data processing; data visualization; computer modeling

For citation: Levashkin S.P., Ivanov K.I., Kushukov S.V. Data farm: Information system for collecting, storing and processing unstructured data from heterogeneous sources. Trudy ISP RAN/Proc. ISP RAS, vol. 35, issue 2, 2023. pp. 57-72 (in Russian). DOI: 10.15514/ISPRAS-2023-35(2)-5

Acknowledgements: The data farm was developed as part of a mega-grant from the Ministry of Science and Education. The authors are grateful to the colleagues of VNIITF of Rosatom (Snezhinsk) and Keldysh Institute of Applied Mathematics of Russian Academy of Sciences (Moscow) for collaboration.

1. Введение

Автоматизированный сбор, хранение и структурирование разнородных данных является одной самых сложных проблем в так называемом «слабом искусственном интеллекте» (СИИ), то есть интеллекте, основанном на данных. Ее решение представляет из себя серьезную проблему, требующую огромных вычислительных и человеческих ресурсов. До сих пор для структурирования и разметки данных применяется ручной или интерактивный режим. Между тем, успешное применение алгоритмов машинного обучения, которые являются основными в СИИ, невозможно без процесса предварительной подготовки данных [1]. Этой задачей занимаются специалисты в науке о данных (data scientists) [2]. Цель, которую преследует представленная в этой статье ферма данных, максимально автоматизировать данный процесс.

Хорошо известна концепция «фермерство данных» (data farming) и его различные определения. Например: *Фермерство данных — это процесс использования разработанных вычислительных экспериментов для «выращивания» данных, которые затем можно анализировать с использованием статистических методов и методов визуализации, чтобы получить представление о сложных системах. Эти методы могут быть применены к любой вычислительной модели* [3] или *Фермерство данных — это метод, в котором используется междисциплинарный подход, включающий высокопроизводительные вычисления,*

моделирование и симуляцию, а также статистический анализ, позволяющий глубоко изучить множество вопросов с многочисленными альтернативами. Это инновационный метод тщательного изучения неопределенных событий с множеством потенциальных результатов. Проводя многочисленные эксперименты, фермерство данных позволяет нам понять как очевидные, так и неясные результаты, тем самым предоставляя содержательные ответы лицам, принимающим решения [4].

Наша концепция фермы данных заключается в том, чтобы на ферме выращивание данных, их хранение, анализ и визуализация были сведены в единую автоматизированную систему, построенную по модульному принципу конструктора «Лего». Причем, следуя принципу «Лего», мы можем как собрать эти модули, так и разобрать, переделав их, если результаты работы системы не будут удовлетворительными. А именно:

M1) модуль выращивания (больших) данных заключается в сборе данных в какой-то предметной области из гетерогенных источников (в основном из сети Интернет) в режиме реального времени 24/7;

M2) модуль хранения данных – это хранилище данных в NoSQL базе MongoDB;

M3) модуль анализа данных – это предварительная обработка данных, их разметка, агрегация и выдача в формате удобном для применения машинного обучения или иных алгоритмов искусственного интеллекта;

M4) модуль визуализации данных, собранных и обработанных на ферме – это дашборд, который дает возможность проводить наглядные численные эксперименты компьютерного моделирования и сравнивать их с реальными данными, что позволяет максимально точно вычислить или оценить параметры той или иной модели какой-либо предметной области.

Эффективность данного подхода была протестирована на данных по COVID-19 и модели, разработанной РФЯЦ – ВНИИТФ Росатома в мегапроекте «Моделирование эпидемий вирусных инфекций». Стоит также отметить, что предложенная архитектура фермы данных универсальна и легко может быть адаптирована к любой предметной области.

Проделанный анализ открытых источников информации по методам и программам не позволил выявить программное средство, удовлетворяющее всем основным требованиям к разрабатываемому в данном научном исследовании проекту «ферма данных», в котором осуществляется сбор информации из открытых источников, а также прогнозирование развития событий на основе собранных данных при помощи математической модели.

Наибольший интерес из рассмотренных систем представляет программный комплекс автоматического интеллектуального сбора данных из различных интернет источников [5] (Экспериментальный образец программного комплекса «Автоматическая интеллектуальная система сбора данных из различных интернет источников» (ЭО ПК)), разработанный в Национальном центре когнитивных разработок (НЦКР) — центре компетенции Национальной технологической инициативы (НТИ) на базе Университета ИТМО. Он обеспечивает функции создания задач сбора данных, тестирования задач, создания сценариев сбора, запуск сценариев локально и в распределенном режиме. Он может быть использован для создания различных систем мониторинга. ЭО ПК предназначен для решения следующих задач:

– сбор данных из различных источников;
– обеспечение мониторинга в сети интернет.

Сильной частью этого комплекса является удобный интерфейс и мультиканальное использование со встроенными механизмами управления и квотирования (различные интерфейсы доступа). Взаимодействие оператора с ЭО ПК осуществляется посредством графического пользовательского интерфейса, загружаемого в приложение Telegram путем манипуляций отображаемыми графическими элементами. С помощью этих манипуляций

пользователь может осуществлять работу по созданию и запуску сценариев сбора данных, в рамках которых обеспечивается функциональность ЭО ПК.

К недостаткам рассмотренного пакета программ можно отнести:

– строго интерактивный режим работы системы, который предполагает высокий уровень ручной работы оператора и требует его соответствующей квалификации;
– весьма ограниченное использование современных информационных технологий и искусственного интеллекта: непосредственно интерфейс системы представляет собой telegram-приложение (по сути это telegram-бот, ведущий «диалог» с оператором);
– отсутствие каких-либо примеров собранных системой данных;
– отсутствие средств обработки (агрегация, разметка) собранных данных;
– отсутствие возможности применения средств для прогнозирования на основе собранных данных.

Заметим, что все эти недостатки отсутствуют в ЭО ПК «ферма данных».

Среди зарубежных разработок стоит отметить BrandWatch – это платформа мониторинга социальных сетей, которая просматривает в Интернете отзывы, статьи и обсуждения. Компания задействует анализ настроений на основе правил [6]. В общем, подход компании – это сбор и обработка данных, их фильтрация, обработка и визуализация для конечного пользователя.

Упомянутые выше компании не раскрывают информацию о том, какие программные комплексы они используют, если речь идет о применении машинного обучения для анализа данных. В своем большинстве, используются опросы как через специальные формы, так и с помощью телефонных звонков. Наиболее близкой концепцией в плане сбора данных обладает отечественный университет ИТМО, а по своей общей концепции наша система имеет больше общего с зарубежной BrandWatch.

Кроме того, нам неизвестны и другие аналоги разработанной системы, что подтверждается проведенным нами патентным поиском [7] и отзывами экспертов на наши публикации [8-10].

Другие разделы статьи устроены следующим образом. Разд. 2 содержит технические требования к реализации программного комплекса. В разд. 3 описаны архитектура фермы данных, ее модули и функционалы. Статью завершают выводы, сделанные на основании исследования, список литературы и информация об авторах.

2. Требования к реализации программного комплекса «ферма данных»

При реализации программного комплекса ферма данных (ПКФД) в качестве основных ключевых условий были сформулированы следующие:

- разрабатываемый ПКФД должен позволять осуществлять поиск, извлечение и обновление требующейся информации в задаваемых форматах и с задаваемой периодичностью;
- разрабатываемый ПКФД необходимо реализовать на принципах максимального использования открытого программного обеспечения;
- хранилище данных создаваемого ПКФД должно обеспечиваться программной средой системы управления базой данных (СУБД);
- ПКФД должен быть масштабируемым, иметь модульную схему, может быть примененным в любых предметных областях.

В следующем разделе статьи представлены описание ПКФД, его компоненты и применяемые методы, позволяющие собирать, обрабатывать и хранить данные из открытых источников в автоматическом режиме. А также результирующая визуализация средствами интерактивной аналитической панели.

3. Архитектура фермы данных

Цель разработки программно-инструментальной системы «ферма данных» направлена на создание технических решений (см. раздел 2) для формирования динамических баз данных. Система осуществляет поиск и извлечение информации из структурированных и неструктурированных источников, используя технологии искусственного интеллекта и обработки больших данных. Это позволяет получить достаточный объем релевантной информации для поддержки моделей, создаваемых в рамках комплексного проекта в определенной предметной области, например, в сфере COVID-19 [9]. Одно из главных преимуществ разработанной системы является возможность быстрого развертывания сбора, хранения и обработки данных на основе Open-Source компонентов, работающих в режиме реального времени 24/7. Помимо этого, система позволяет получать оповещения о работе через мессенджеры. Она является модульной и позволяет легко заменять компоненты. Благодаря этому, спектр применения системы очень широк, так как ее можно адаптировать для структурирования больших данных в любой предметной области.

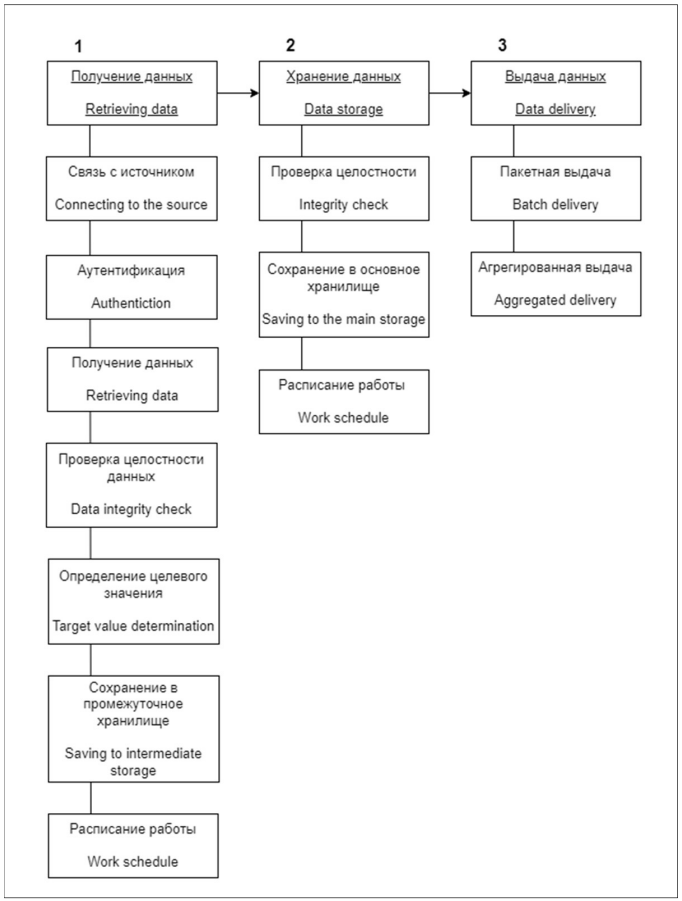

Рис. 1: Принципиальная схема работы программного комплекса ферма данных (ПКФД)
Fig. 1: Operational scheme of the data farm software package (DFSP)

В основу ПКФД легли следующие программные компоненты:

- Ubuntu server 20.4.01 LTS;
- MongoDB 4.4.1;
- Conda 4.9.0;
- Jupyter-notebook 6.0.3;
- Python 3.9.5;
- PyTorch;
- TensorFlow;
- NLTK;
- и другие [8-10].

Отметим некоторые преимущества MongoDB, которые обусловили выбор в пользу этого NoSQL хранилища данных:

- легкая масштабируемость;
- гибкая схема добавления документа;
- хранение данных в формате JSON;
- возможность глубоких запросов.

Главные процессы, представленные на рис. 1:

- подключение к ресурсу – обеспечивает функционал подключения к ресурсу данных по различным протоколам как в интернете, так и подключение к другим базам данных;
- аутентификация – обеспечивает механизмы безопасности при подключении, такие как хранение и предоставление паролей и ключей;
- получение данных – механизм опроса для синхронных и асинхронных источников данных, пакетная обработка;
- проверка данных на целостность – обеспечивает качество полученных данных.

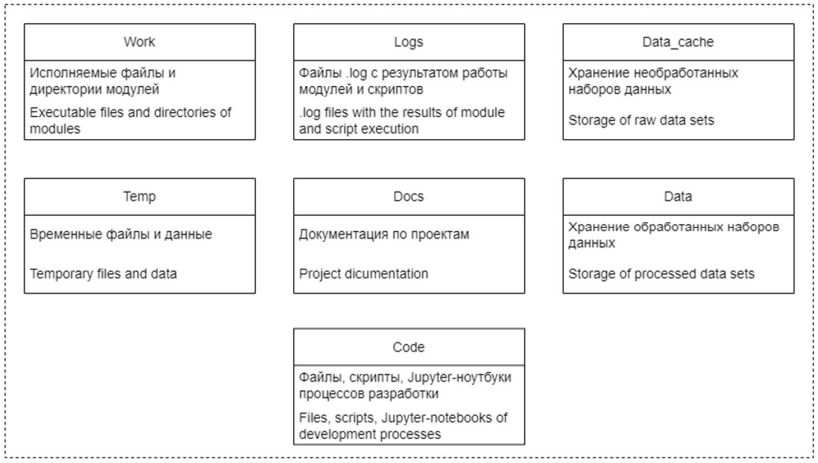

Рис. 2: Структура директорий программного комплекса ферма данных (ПКФД)
Fig. 2: Directory structure of the data farm software package (DFSP)

Кроме того, в работе ПКФД учитываются:

- информация в комментариях о ресурсе данных;
- импорт библиотек Python;
- специальные преобразования данных (очистка, понижение размерности, восстановление и т.п.);
- сохранение данных в директорию временного размещения;
- вызов функций и указание параметров для обработки данных перед тем, как поместить их в основное хранилище MongoDB.

Структура директорий ПКФД на удаленном сервере, представлена на рис. 2:
На рис. 3 показана детальная модульная структура ПКФД. Принципы организации моделей и их функциональное назначение описываются в следующих подразделах статьи.

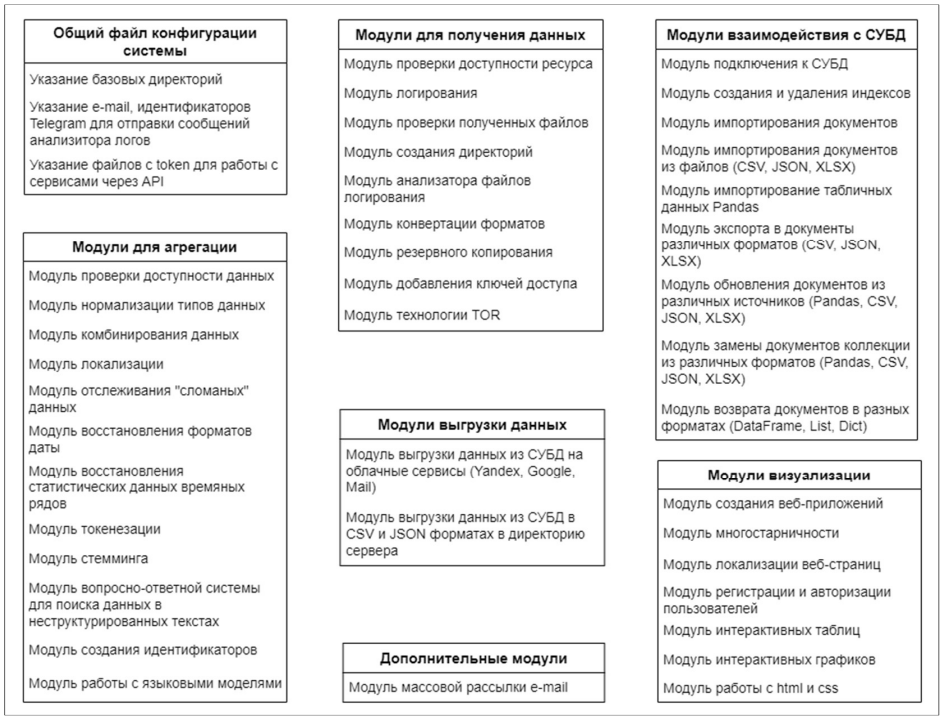

Рис. 3: Модули программного комплекса ферма данных (ПКФД)
Fig. 3: Modules of the data farm software package (DFSP)

3.1 Модуль сбора данных

Мы используем единый подход к сбору данных. Он заключается в стандартизации используемых функций, которые могут быть применимы к различным источникам. Такой подход позволяет быстро добавлять скрипты для ранее не рассматриваемых ресурсов, а также не вызывает затруднений в работе команды, так как в процессе используются одинаковые наименования и конструкции кода. Кроме того, применяются общие файлы настройки для определения директорий хранения данных, что позволяет упростить код скриптов, а также при необходимости ускорить процесс переноса системы на другие платформы (Рис. 4):


```
# Common path download data
main_path = '/home/user/Project/'
download_path = main_path + 'Data_cache/'
download_path_fast_test = download_path + 'Fast_test/'
logs_path = main_path + 'Logs/'
token_vk_download = main_path + 'Work/download_data/module_for_scripts/access_token.txt'

# StopCoronaVirus - download separately
StopCoronaVirus = download_old_path + 'StopCoronaVirus/'

# GOGOV.RU - download separately
Vactinations = download_old_path + 'Vactinations/'

# Covid-19 test data St. Petersburg - download separately
Covid_test_RU_SPE = download_old_path + 'Covid_test_RU-SPE/'

# --- Actual data acquisition scripts (as well as reworked old versions of scripts) ---
# Statistical information (Stop corona virus / Gogov and some more)

Stop_corona_virus = download_path + 'Stop_corona_virus/'
Gogov = download_path + 'Gogov/'
Yanstat = download_path + 'Yanstat/'
Stop_corona_vaccination = download_path + 'Stop_corona_vaccination/'
Stop_corona_restrict = download_path + 'Stop_corona_restrict/'
Stop_corona_russia = download_path + 'Stop_corona_russia/'

# --- VK operational headquarters ---
# --- Information in unstructured text form ---

# Самарская область (Жигулёвск, Новокуйбышевск, Самара, Сызрань, Тольятти, Чапаевск)
# Samara region (Zhigulevsk, Novokuibyshevsk, Samara, Syzran, Tolyatti, Chapaevsk)
VK_RU_SAM = download_path + 'VK_RU_SAM/'

# Алтайский край (Барнаул, Бийск, Новоалтайск, Рубцовск, Белогорск)
# Altai Territory (Barnaul, Biysk, Novoaltaysk, Rubtsovsk, Belogorsk)
VK_RU_ALT = download_path + 'VK_RU_ALT/'
```

Рис. 4: Фрагмент файла конфигурации директорий
Fig. 4: Fragment of the directories configuration file

На рис. 5 представлены ресурс данных Novel Coronavirus (COVID-19) Cases Data университета Джона Хопкинса и QR код ссылки на источник. На листинге 1 приведен скрипт получения данных из этого источника.

Рис. 5: Источник данных: Novel Coronavirus (COVID-19) Cases Data университета Джона Хопкинса
Fig. 5: Source data: Novel Coronavirus (COVID-19) Cases Data from John Hopkins University

```
from hdx.api.configuration import Configuration
from hdx.data.dataset import Dataset
from connect import connect
from makedir import makedir
from clogs import set_log
from copyrate import cops
from check_size import checking
from config import *
from rewriting import rewrite
from check import checks

log = log_conf

director = jhu
directory = director + 'Data/'
```

```
16 directory_backup = director + 'Backup/'
17
18 Configuration.create(hdx_site='prod', user_agent='A_Quick_Example',
hdx_read_only=True)
19
20 def save(source, direct):
21     dataset = Dataset.read_from_hdx(source)
22     resources = dataset.get_resources()
23     for res in resources:
24         url, path = res.download(folder=direct)
25         print('Resource URL %s downloaded to %s' % (url, path))
26
27 query_dataset = 'https://data.humdata.org/dataset/'
28 query_dataset_url = 'novel-coronavirus-2019-ncov-cases'
29 name_source = 'jhu'
30 set_log(log)
31
32 if connect(query_dataset+query_dataset_url, name_source, log) == True:
33     makedir(directory)
34     makedir(directory_backup)
35     before_size = checking.size_change_before(directory_backup)
36     rewrite(directory)
37     save(source=query_dataset_url, direct=directory)
38     after_size = checking.size_change_after(directory)
39     if checking.final_comparison(before_size, after_size, name_source)
== True:
40         if cops(directory, directory_backup, name_source) == True:
41             checks(director, directory_backup)
```

Листинг 1: Скрипт получения данных с Novel Coronavirus (COVID-19) Cases Data университета Джона Хопкинса

Listing 1: The script for obtaining data from Novel Coronavirus (COVID-19) Cases Data of Johns Hopkins University

Для отслеживания ошибок в модуле сбора данных задействовано логирование. В лог файл поступает вся ключевая информация о работе того или иного скрипта (рис. 6).

```
3847 2023-04-11 ⇒ 00:40:02 ⇒ WARNING ⇒ stop_corona_russia ⇒ source is working
3848 2023-04-11 ⇒ 00:40:02 ⇒ WARNING ⇒ stop_corona_russia ⇒ successful download of updated data
3849 2023-04-11 ⇒ 00:40:02 ⇒ WARNING ⇒ stop_corona_russia ⇒ backup completed successfully
3850
3851 2023-04-11 ⇒ 00:50:02 ⇒ WARNING ⇒ stop_corona_vaccination ⇒ source is working
3852 2023-04-11 ⇒ 00:50:02 ⇒ WARNING ⇒ stop_corona_vaccination ⇒ downloading data without updating
3853 2023-04-11 ⇒ 00:50:02 ⇒ WARNING ⇒ stop_corona_vaccination ⇒ backup completed successfully
3854
3855 2023-04-11 ⇒ 01:00:01 ⇒ WARNING ⇒ stop_corona_virus ⇒ source is working
3856 2023-04-11 ⇒ 01:03:50 ⇒ WARNING ⇒ stop_corona_virus ⇒ successful download of updated data
3857 2023-04-11 ⇒ 01:03:50 ⇒ WARNING ⇒ stop_corona_virus ⇒ backup completed successfully
3858
3859 2023-04-11 ⇒ 10:12:04 ⇒ WARNING ⇒ jhu ⇒ source is working
3860 2023-04-11 ⇒ 10:14:46 ⇒ WARNING ⇒ jhu ⇒ downloading data without updating
3861 2023-04-11 ⇒ 10:14:46 ⇒ WARNING ⇒ jhu ⇒ backup completed successfully
```

Рис. 6: Фрагмент файла логирования скриптов получения данных
Fig. 6: Fragment of the log file for data acquisition scripts

Анализатор файлов логирования оповещает об ошибках при загрузке данных, что весьма актуально при сборе интернет-данных. Сообщения отправляются как на электронную почту администратора ПКФД, так и в его Telegram канал. Поскольку система предусматривает возможности расширения и добавления новых разработчиков к проекту, в анализатор можно добавлять неограниченное число лиц для получения информации о работоспособности системы сбора данных (Рис. 7).

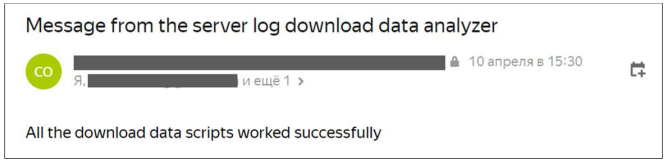

Рис. 7: Поступление сообщения от анализатора файлов логирования системы
Fig. 7: Message from the system of log file analyzer

3.2 Модуль хранения данных

Для организации хранилища данных (data warehouse), используется MongoDB – документно-ориентированная система управления базами данных с открытым исходным кодом, не требующая описания схемы таблиц. Это NoSQL база данных, которая использует JSON-подобные документы и предоставляет драйверы и инструменты для взаимодействия с хранилищем данных MongoDB, используя различные языки программирования, такие как Python, JavaScript, Java, Go и C#. Пример документа, который может сохранять MongoDB, показан на листинге 2. Также используется Pymongo – официальная библиотека Python для взаимодействия с MongoDB, которая работает на Python 3.9.5.

```
Dashboard_ver_3> db.russia_lang_eng.find().sort({"Date": -1}).limit(1)
[
  {
    _id: '13719700186478753932',
    Date: ISODate("2023-03-27T00:00:00.000Z"),
    'Confirmed WHO': 0,
    'Deaths WHO': 0,
    'Recovered JHU': 0,
    'Daily confirmed WHO': 0,
    'Daily deaths WHO': 0,
    'Daily recovered JHU': 0,
    'Confirmed': 22595199,
    'Deaths': 397109,
    'Mortality_percent': 0.01,
    'Recovered': 21946453,
    'Infected on date': 251637,
    'Daily confirmed': 8989,
    'Daily deaths': 31,
    'Daily recovered': 8160,
    hash_row: '14139009034644559760'
  }
]
```

Листинг 2: Пример документа в MongoDB
Listing 2: An example of a document stored in the MongoDB

Работа с базой MongoDB предусматривает не только использование встроенной консоли mongoshell, но и модульные скрипты для получения данных с различных ресурсов (пример скрипта приведен на листинге 3.

```
1  from config import *
2  from makedir import makedir
3  from tools_mongo import *
4  from tools_aggregation import *
5  from clogs import set_log
```

```
6  import logging
7
8  log = log_conf_agg
9  set_log(log)
10 type_agg = name_db_add_world
11 database = dashboard_database_ver_3
12 address = dashboard_address
13
14 list_csv_path = find_final_csv(aggregated_final + world_path)
15 start_quick_save_2(type_agg, list_csv_path, database, address,
                        key_for_update='_id')
```

Листинг 3: Пример скрипта для добавления данных в MongoDB
Listing 3: Example of a script for adding data to MongoDB

3.3 Модуль обработки данных

Процесс агрегации данных рассматривается как предварительная обработка. Он заключается в следующем:

- агрегация данных из предварительного хранилища, сбор одинаковых данных из разных источников;
- проверка целостности данных – проверка и обработка недостающих, дублирующийся данных, проверка ошибки формата данных,
- приведение значений к общим наименованиям, локализация значений, добавление координат;
- сохранение в MongoDB – размещение в основном хранилище;
- расписание работы – запуск скриптов для получения данных;

На листинге 4 продемонстрирован пример подобного скрипта.

```
1  from config import *
2  from makedir import makedir
3  pd.options.mode.chained_assignment = None
4  from tools_aggregation import *
5  from tools_mongo import *
6  from clogs import set_log
7  import logging
8  log = log_conf_agg
9  set_log(log)
10
11 covid_city_name = 'covid_city.csv'
12 covid_city = raw_data + 'covid_city/'
13 covid_city_final = raw_data  + 'covid_city/Final/'
14
15 os.chdir(covid_city)
16 extension = 'csv'
17 filenames_gogov_data = [i for i in
glob.glob('*.{}'.format(extension))]
18 subset_columns_covid_city = ['date', 'city_name']
19
20 comb_file(name_combined_file=covid_city_name,
directory_rebuild=covid_city, directory_final=covid_city_final,
subset_columns = subset_columns_covid_city, key='date')
21
22 df0 = read_from_source(path_file=covid_city_final + covid_city_name)
23
```



```
24 df0 = df0.drop_duplicates(subset=['date', 'city_name'], keep='first')
25
26 ru_coordinate = raw_data + 'city_coordinate/koord_russia.csv'
27 df1 = pd.read_csv(ru_coordinate, encoding='cp1251', sep=';')
28
29 df1 = df1.rename({'Город': 'city_ru_name'}, axis=1)
30
31 st = df0.merge(df1, left_on='city_ru_name', right_on='city_ru_name',
how='left')
32
33 st = st.drop(columns=['region_ru_name', 'region_name', 'city_name'])
34 st = st.rename({'Регион': 'region', 'Федеральный округ':
'federal_district', 'lat': 'latitude_dd', 'lng': 'longitude_dd',
'city_ru_name': 'city_name'}, axis=1)
35 st[['area', 'latitude_dd', 'longitude_dd']] = st[['area',
'latitude_dd', 'longitude_dd']].astype('string')
36 st['area'] = st['area'].str.replace(',', '.')
37 st['area'] = st['area'].str.replace(' ', '')
38 st['latitude_dd'] = st['latitude_dd'].str.replace(',', '.')
39 st['longitude_dd'] = st['longitude_dd'].str.replace(',', '.')
40
41 st[['city_name', 'region_code', 'region', 'federal_district']] =
st[['city_name', 'region_code', 'region',
'federal_district']].astype('string')
42 st[['area', 'latitude_dd', 'longitude_dd']] = st[['area',
'latitude_dd', 'longitude_dd']].astype('float')
43 st[['mask', 'gloves']] = st[['mask', 'gloves']] .astype('bool')
44 st['date'] = pd.to_datetime(st['date'], format='%Y-%m-%d')
45
46 hash_rows = pd.util.hash_pandas_object(st, index=True).values
47 hash_rows = pd.DataFrame(hash_rows, columns =['hash_row'])
48 st = st.join(hash_rows)
49 st['hash_row'] =  st['hash_row'].astype('string')
```

Листинг 4: Пример скрипта агрегации для объединения CSV данных COVID-19 по городам Российской Федерации, добавлением координат, настройкой типов данных в столбцах и добавлением идентификаторов
Listing 4: An example of an aggregation script for combining COVID-19 CSV data for cities in the Russian Federation, adding coordinates, setting up data types in columns, and adding identifiers

Кроме того, при запуске скриптов используется системный сервис Cron для контроля времени их работы (рис. 8):

```
# Aggregation
0 3 */1 * * /home/user/anaconda3/bin/python3 /home/user/Project/Work/aggregation_data/RU_AMU.py >
10 3 */1 * * /home/user/anaconda3/bin/python3 /home/user/Project/Work/aggregation_data/RU_ALT.py
20 3 */1 * * /home/user/anaconda3/bin/python3 /home/user/Project/Work/aggregation_data/RU_KEM.py
30 3 */1 * * /home/user/anaconda3/bin/python3 /home/user/Project/Work/aggregation_data/RU_CHE.py
```

Рис. 8: Запуск скриптов агрегации по ежедневному расписанию
Fig. 8: Running aggregation scripts on a daily schedule

Модуль обработки естественного языка собирает, сохраняет и обрабатывает текстовые данные из социальных сетей, таких как VKontakte, Twitter, Telegram и др. Этот модуль настроен таким образом, чтобы вычислить или оценить параметры модели, разработанной в конкретной предметной области. В нашем случае, предметной областью были данные по COVID-19. Пример исходной записи обрабатываемых данных представлен далее (рис. 9):

7 октября в регионе подтверждено 644 случая новой коронавирусной инфекции: - 248 в г. самара - 118 в г. тольятти - 26 в г. новокуйбышевск - 23 в г. отрадный - 21 в г. кинель - 16 в нефтегорском районе - 16 в г. сызрань - 12 в кинель-черкасском районе - 12 в сергиевском районе - 11 в волжском районе - 10 в г. чапаевск - 9 в красноармейском районе - 9 в ставропольском районе - 8 в приволжском районе - 7 в алексеевском районе - 7 в кинельском районе - 7 в клявлинском районе - 7 в кошкинском районе - 6 в похвистневском районе - 6 в большеглушицком районе - 6 в богатовском районе - 6 в г. похвистнево - 5 в исаклинском районе - 5 в пестравском районе - 5 в борском районе - 5 в г. жигулевск - 5 в г. октябрьск - 5 в хворостянском районе - 4 в большечерниговском районе - 4 в челно-вершинском районе - 3 в красноярском районе - 3 в шигонском районе - 3 в елховском районе - 2 в камышлинском районе - 2 в безенчукском районе - 1 в шентали нском районе - 1 в сызранском районе 📌612 человек обследовано с диагнозом внебольничная пневмония и орви; 📌32 человека выявлено при профилактическом обследовании лиц, не имеющих клинических проявлений. 📉нарастающим итогом в регионе зафиксировано 109622 случая коронавируса. скончались 4048 человек , в том числе женщины 60, 60, 60, 81, 87, 87, 84, 84 лет; мужчины 73, 78 лет; страдавшие заболеванием сердечно-сосудистой системы; женщины 58, 62, 84 лет; мужчина 81 года, страдавшие заболеваниями сердечно-сосудистой и эндокринной систем; женщины 72, 78, 78, 79, 79, 79, 81, 83 лет; мужчины 55, 59 лет; женщина 67 лет, страдавшая заболеванием сердечно-сосудистой системы, онкологическим заболеванием; женщины 72, 85 лет, страдавшие заболеваниями сердечно-сосудистой системы и органов дыхания; мужчина 58 лет, страдавший инфекционным заболеванием; мужчина 67 лет, страдавший заболеванием пищеварительной системы; показатель смертности на 100 тыс. населения в области - 126,78, по стране - 146,27. коэффициент летальности в области - 3,69, в стране - 2,78. 🧬проведено лабораторных исследований на кви за сутки – 14432, нарастающим итогом 3741448.

Рис. 9: Пример полученной записи из группы оперативного штаба ВКонтакте по Самарской области
Fig. 9: Example of a record received from the VKontakte operational headquarters group for the Samara region

date	city	infected_city	pneumonia_ARVI	death_rate_per_100_thousand_population_region	death_rate_per_100_thousand_population_country	without_clinical_manifestation
01-01-2021	самара	127	258	19.35	39.42.	45
01-04-2021	самара	80	169	34.67	67.97.	30
01-05-2021	самара	74	145	47.35	75.69.	15
01-06-2021	самара	18	64	61.35	83.47.	8
01-07-2020	самара	41	32	1.75	6.53.	19

Рис. 10: Фрагмент полученного набора данных с использованием методов обработки естественного языка
Fig. 10: Fragment of the obtained dataset using natural language processing methods

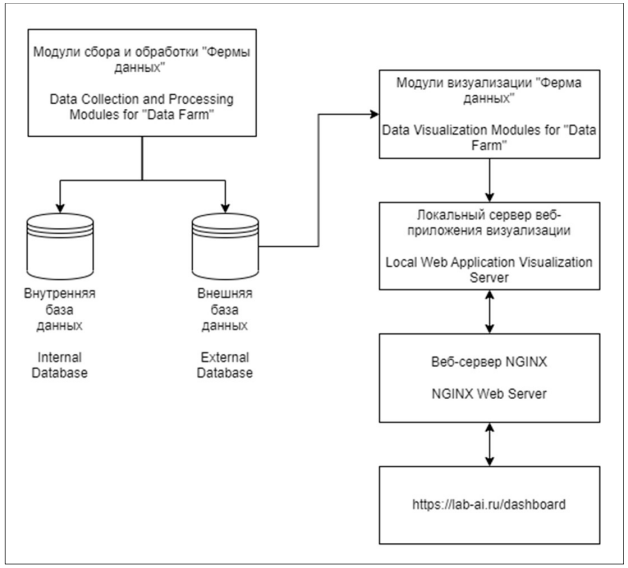

Рис. 11: Архитектура визуализации данных
Fig. 11: Data visualization architecture

3.4 Модуль визуализации данных

Визуализация данных представлена в виде дашборда, позволяющего анализировать и сравнивать собранные и обработанные данные о ходе заболевания COVID-19 в странах мира и регионах РФ. Уникальность визуализации состоит в том, что представленные на дашборде результаты исследования позволяют точную настройку ряда управляющих параметров моделей распространения коронавирусной эпидемии. Все размещенные данные доступны для скачивания в формате CSV. На Рис. 11 показана архитектура визуализации данных:
На Рис. 12 представлен скриншот страницы дашборда, расположенного по адресу: https://lab-ai.ru/dashboard

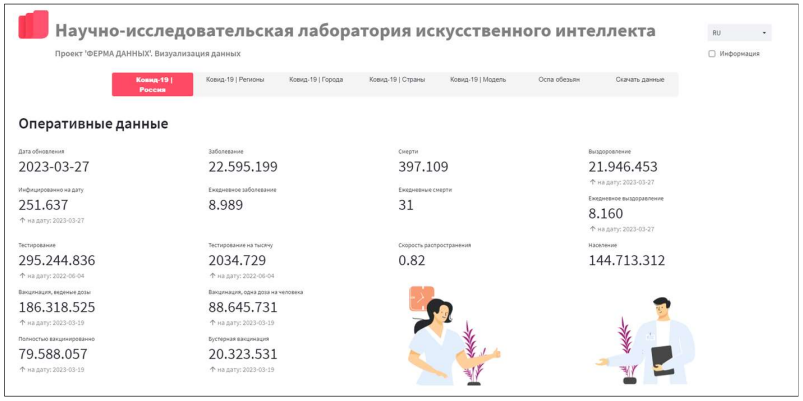

Рис. 12: Страница дашборда визуализации данных по COVID-19
Fig. 12: COVID-19 data visualization dashboard page

4. Заключение

В статье представлена оригинальная разработка автоматизированной системы сбора, хранения и обработки данных из разнородных источников («ферма данных»). Данная система призвана вычислить или оценить параметры модели в любой предметной области. Подчеркнем, что наша задача, как разработчиков, была добиться максимальной автоматизации всех процессов, используемых на ферме.

Описана архитектура фермы, ее модули и функционалы, а также их реализация на языке программирования Python. Ферма была развернута и протестирована на удаленном сервере. В качестве предметной области использовались источники данных по COVID-19 в России и других странах мира. Моделью этой предметной области служила модель распространения вирусных инфекций, разработанная РФЯЦ – ВНИИТФ Росатома. На выходе системы – файлы временных рядов в формате CSV и JSON, т.е. большие данные готовые для машинного обучения и анализа. Данные по количеству инфицированных, умерших, выздоровевших, сделанных тестов, положительных тестов, вакцинированных, случаев применения ИВЛ и т.п. представлены для городов России с населением свыше 50,000 человек.

Ферма работает в режиме реального времени 24/7, собирая, сохраняя и обрабатывая большие объемы данных. При ее разработке, широко применяются самые современные методы, технологии, среды, библиотеки и алгоритмы искусственного интеллекта и науки о данных. Вот некоторые из них: NoSQL база данных MongoDB; Anaconda; Jupyter; Pandas; Scikit-learn; NLTK; интеллектуальная оптимизация; обработка естественного языка и многие другие.

Отметим, что ферма данных находится в стадии активной разработки: добавляются новые модули и она адаптируется к другим предметным областям. В зависимости от сложности и

объема решаемых задач, а также размера данных предметной области, ферма может быть преобразована в фабрику или даже концерн данных.

Список литературы / References

[1] Müller A.C., Guido S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, 2016, 398 p.

[2] Куцев Р. Разметка данных в машинном обучении: процесс, разновидности и рекомендации / Kutsev R. Data labeling in machine learning: process, variations and recommendations. Available at: https://habr.com/ru/company/ods/blog/327242/, accessed March 14, 2023 (in Russian).

[3] Lucas T.W., Kelton W.D. et al, Changing the Paradigm: Simulation, Now a Method of First Resort. Naval Research Logistics, vol. 62, issue 4, 2015, pp. 293–305.

[4] A. Kusiak, Data Farming: A Primer. International Journal of Operations Research, vol. 2, issue 2, 2005, pp. 48-57.

[5] Экспериментальный образец программного комплекса «Автоматическая интеллектуальная система сбора данных из различных интернет источников» / Experimental sample of the software complex «Automatic intelligent system for collecting data from various Internet sources». Available at: https://actcognitive.org/files/aicrawler_2_rukovodstvo_operatora.pdf, accessed April 14, 2023 (in Russian).

[6] Bannister K. Understanding Sentiment Analysis: What It Is & Why It's Used. Available at: https://www.brandwatch.com/blog/understanding-sentiment-analysis/, accessed April 14, 2023.

[7] Отчет о патентных исследованиях по тематике «ферма данных» / Patent Research Report on Data Farm. Available at: https://ai.psuti.ru/docs/Patent_search.pdf, accessed April 14, 2023 (in Russian).

[8] Левашкин С.П., Агапов С.Н. и др, Исследование адаптивно-компартментной модели распространения КОВИД-19 в некоторых регионах РФ методами оптимизации, Математическая биология и биоинформатика, том 16, вып. 1, 2021 г., стр. 136-151 / Levashkin S.P., Agapov S.N. et al. Study of SEIRD Adaptive-Compartmental Model of COVID-19 Epidemic Spread in Russian Federation Using Optimization Methods. Mathematical Biology and Bioinformatics, vol. 16, issue 1, 2021, pp. 136-151.

[9] Проект 'ФЕРМА ДАННЫХ'. Визуализация данных. Научно-исследовательская лаборатория искусственного интеллекта / Project 'DATA FARM'. Artificial Intelligence Research Laboratory. Available at: https://lab-ai.ru/dashboard, accessed April 14, 2023 (in Russian).

[10] Левашкин С.П., Захарова О.И. и др. Модульная система сбора данных. Свидетельство о регистрации программы для ЭВМ, № 2022617725. Дата государственной регистрации в реестре программ для ЭВМ 25.04.2022 / Levashkin S.P., Zakharova O.I. et al. Modular data collection system. Certificate of registration of a computer program, № 2022617725. Date of state registration in the register of computer programs 25.04.2022 (in Russian).

Информация об авторах / Information about authors

Сергей Павлович ЛЕВАШКИН – профессор, кандидат физико-математических наук, PhD in Computer Science, действительный член Академии наук Мексики, заведующий научно-исследовательской лабораторией искусственного интеллекта. Ученый с более чем 20-летним опытом работы в университетах и компаниях России, Северной Америки и Европы в области искусственного интеллекта и машинного обучения.

Sergey Pavlovich LEVASHKIN is a Professor, a Candidate of Physical and Mathematical Sciences, PhD in Computer Science, a Full Member of the Academy of Sciences of Mexico, and the head of the Artificial Intelligence Research Laboratory. He is a scientist with more than 20 years of experience working in universities and companies of Russia, North America, and Europe in the field of artificial intelligence and machine learning.

Константин Николаевич ИВАНОВ – магистрант, инженер НИЛ ИИ ПГУТИ. Научные интересы включают: сбор данных, обработку естественного языка, построение веб-сервисов и приложений, визуализация данных.

Konstantin Nikolaevich IVANOV is a Master's Student and an engineer at the Artificial Intelligence Research Laboratory. His research interests include data collection, natural language processing, building web services and applications, data visualization.

Сергей Владимирович КУШУКОВ – магистрант, инженер НИЛ ИИ ПГУТИ. Научные интересы включают: информационные системы и технологии, сбор и обработка информации, написание веб-приложений.

Sergey Vladimirovich KUSHUKOV is a Master's Student and an engineer at the Artificial Intelligence Research Laboratory. His scientific interests include data collection and processing, as well as web application development.