References

1. Dmitriev A.S., Panas A.I. *Dynamic Chaos: Novel Type of Information Carrier for Communication Systems*. Moscow: Izd-vo fiziko-matematicheskoj literatury, 2002, 252 p. (In Russ.)
2. Varakin L.E. *Communication Systems with Noise-Like Signals*. Moscow: Radio i svyaz', 1985, 384 p. (In Russ.)
3. Dmitriev A.S. et al. Direct chaotic ultra-wideband wireless communications in the very high frequency and ultra high frequency radio bands. *Radiotekhnika i elektronika*, 2022, vol. 67, pp. 1013–1021. DOI: 10.31857/S0033849422080046 (In Russ.)
4. Kuz'min L.V., Efremova E.V. Experimental estimation of the propagation time of chaotic ultra-wide-band RF pulses through multipath channel. *Pis'ma v Zhurnal tekhnicheskoj fiziki*, 2020, vol. 46, pp. 23–27. DOI: 10.21883/PJTF.2020.16.49849.18352 (In Russ.)
5. Loginov S.S., Zuev M.Y. Testing of generators of pseudo-random signals based on a Lorenz system, realized over a Galois finite field. *2018 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO)*. Minsk, 2018, pp. 1–4. DOI: 10.1109/SYNCHROINFO.2018.8457039
6. Zhang L. System generator model-based FPGA design optimization and hardware co-simulation for Lorenz chaotic generator. *2017 2nd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*. Wuhan, 2017, pp. 170–174. DOI: 10.1109/ACIRS.2017.7986087
7. Loginov S.S., Afanasiev V.V. Poly-Gaussian models in describing the signals of Lorenz dynamic system. *2018 Systems of Signals Generating and Processing in the Field of on Board Communications*. Moscow, 2018, pp. 8350616–4. DOI: 10.1109/SOSG.2018.8350616
8. Chabdarov Sh.M., Trofimov A.T. Polygaussian representations of arbitrary noise and reception of discrete signals. *Radiotekhnika i elektronika*, 1975, vol. 20, no. 4, pp. 734–735. (In Russ.)
9. Nadeev A.F. et al. Optimal reception of multi-position signals at a complex of noise and impulse interference with arbitrary fluctuations. *Radiotekhnika*, 1990, no. 12, pp. 32–35. (In Russ.)
10. Kafarov K.M., Loginov S.S., Bobina E.A. Digital signal generators based on the Lorentz system implemented using fixed-point numbers. *Systems of Signals Generating and Processing in the Field of on Board Communications*. Moscow, 2023, vol. 6 (1), pp. 197–200. DOI: 10.1109/IEEECONF56737.2023.10092093

Reseived 23.04.2024

УДК 004.89

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ БИБЛИОТЕК VAEX И DASK

Пальмов С.В.[1,2], Шаталов Н.В.[1]
[1]Поволжский государственный университет телекоммуникаций и информатики, Самара, РФ
[2]Самарский государственный технический университет, Самара, РФ
E-mail: s.palmov@psuti.ru, nickit.schatalow@yandex.ru

Цель исследования заключалась в сравнении производительности библиотек Vaex и Dask, предназначенных для повышения эффективности процесса обработки данных. Для решения поставленной задачи были проведены эксперименты, связанные с оценкой временных затрат на выполнение различных классов операций. Исследование включало подготовку датасетов, формирование выборок данных, настройку исполнительных сред, установку и настройку указанных выше модулей, написание скриптов на языке Python, тестирование производительности и последующий анализ результатов. Было установлено, что Vaex демонстрирует высокое быстродействие в случае обработки больших наборов данных, состоящих из миллиона объектов, на одном локальном компьютере; показатели Dask уступают первой библиотеке. Сей факт указывает на то, что Vaex является более эффективным инструментом для обработки крупных датасетов в условиях, аналогичных использованным в настоящей работе. Результаты и выводы исследования подчеркивают значимость выбора оптимальной библиотеки при обработке данных большого объема, а также подтверждают преимущества библиотеки Vaex в данном контексте.

Ключевые слова: *Vaex, Dask, Python, большие данные, обработка данных*

Введение

В современном мире каждую секунду генерируется огромное количество данных, и их анализ становится все более важным в контексте принятия обоснованных решений. Для эффективной обработки и исследования подобных объемов информации необходимы специализированные инструменты.

Среди множества разнообразных решений, предназначенных для работы с данными, выделяются Vaex [1] и Dask [2] – две популярные библиотеки, способные обрабатывать большие датасеты и выполнять с ними сложные операции.

Целью настоящей работы являлась проверка гипотезы о том, что Vaex окажется предпочтительнее для обработки и анализа крупных объемов данных по сравнению с Dask при работе на локальной машине. В ходе исследования было проведено сравнение функциональности, особенностей и производительности обеих библиотек, а также рассмотрены сценарии, в которых каждая из них может продемонстрировать наибольшую эффективность.

Dask и Vaex

Dask [3] реализует параллельные вычисления. Она способна обрабатывать массивы данных, превышающие объем оперативной памяти компьютера. Одним из ключевых преимуществ Dask является масштабируемость для работы с кластерами.

Vaex [5], с другой стороны, также предоставляет высокопроизводительные инструменты для работы с данными, но фокусируется на эффективной обработке и анализе больших наборов данных на одиночных компьютерах. Она предлагает альтернативные структуры, оптимизированные для работы с большими объемами информации в оперативной памяти, а также обладает функционалом для быстрой визуализации и анализа данных.

Противопоставление Pandas: различия в работе Dask и Vaex

Датафрейм Dask не полностью совместим с Pandas [4], но указанные библиотеки довольно близки к этому: первая содержит в себе часть функционала, присущего второй.

Vaex в большей степени отличается от Pandas (хотя в плане выполнения базовых операций, таких как чтение данных и вычисление сводной статистики, они очень похожи).

При этом если сравнивать три указанные библиотеки между собой, [6], у каждой можно выделить своим преимущества.

Практическое сравнение библиотек при работе с данными

Технические характеристики компьютера для проводимых экспериментов представлены в таблице 1.

Проведено исследование процесса чтения и преобразования файлов размером 48 Гб на локальном компьютере с использованием библиотек Vaex и Dask. Чтобы повысить удобство процесса чтения CSV-файлов больших размеров [7], было выполнено их конвертирование в формат HDF5. В исследовании задействовано 52 CSV-файла, содержащие различные типы данных (текст, числа, даты и числовые массивы) для тренировочных проектов по машинному обучению [8].

Таблица 1. Технические характеристика локального компьютера

CPU	AMD Ryzen 5 1400
RAM, Гб	16
GPU	NVIDIA Pascal GeForce GTX 1050 Ti GDDR5 4 ГБ
HDD	2 ТБ, 190 Мбайт/сек

HDF5 (Hierarchical Data Format version 5) – это открытый стандарт для хранения и организации больших объемов данных. Он обеспечивает эффективное хранение различных типов разнородных данных, включая числовые массивы, текстовые данные, изображения, аудиофайлы и многое другое. HDF5 представляет собой иерархическую структуру данных, которая позволяет организовывать информацию в виде древовидной структуры, состоящей из групп и датасетов. Группы могут содержать в себе подгруппы и датасеты, чем обеспечивается гибкая организация данных.

На первом шаге эксперимента данные были преобразованы в формат HDF5 с использованием Vaex; временные затраты составили 578 с. Затем выполнялись чтение и обработка этих файлов при помощи той же библиотеки, что заняло 15 с. и 79 с. соответственно. Обработка заключалась в выводе в консоль всех значений переменных, которые оказывались больше 75.

Далее процесс был произведен повторно с использованием библиотеки Dask. Преобразование в HDF5 потребовало 1343 с., чтение и обработка – 49 и 283 с.

Таким образом, в результате данного эксперимента было показано, что с использованием Vaex на локальной машине процесс обработки данных происходил быстрее по сравнению с Dask.

Второй этап эксперимента заключался в исследовании операций над двумя датасетами с суммарным размером 33,86 ГБ. Наборы состояли из случайных чисел от 0 до 100, и содержали миллион строк и тысячу столбцов каждый (рисунок 1).

Код для преобразования формата с помощью Dask взят из официальной документации [11] (рисунок 2). Чтобы преобразовать два CSV (33,86 ГБ) в два файла HDF5 (14,9 ГБ), Vaex затратил 1106,63 с. (рисунки 3–5).

В ходе эксперимента выяснилось, что Dask не имеет возможности прочитать файлы HDF5 созданные через Vaex.

Неспособность второй библиотеки прочитать результаты конвертации первой может быть объяснена различиями в структуре файлов и способе их записи. Vaex использует оптимизированный формат данных HDF5, что может привести к возникновению специфических особенностей, несовместимых с Dask. Таким образом, для обе-

```
import pandas as pd
import numpy as np
from os import path


n_rows = 1000000
n_cols = 1000
for i in range(1, 3):
    filename = 'analysis_%d.csv' % i
    file_path = path.join('csv_files', filename)
    df = pd.DataFrame(np.random.uniform(0, 100, size=(n_rows, n_cols)), columns=['col%d' % i for i in range(n_cols)])
    print('Saving', file_path)
    df.to_csv(file_path, index=False)
```

Рисунок 1. Листинг программы создания файлов датасета

```
import datetime
import vaex

# фиксируем и выводим время старта работы кода
start = datetime.datetime.now()
df = vaex.open('csv_files/analysis_1.csv', convert='csv_files/analysis_1.hdf5')
# фиксируем и выводим время окончания работы кода
finish = datetime.datetime.now()
# вычитаем время старта из времени окончания
print('Время работы: ' + str(finish - start))
```

Рисунок 2. Листинг программы для конвертирования Vaex

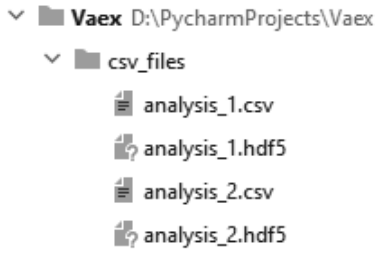

Рисунок 3. Результаты преобразования файлов

analysis_1	22.04.2024 0:08	Исходный файл ...	17 757 308 КБ
analysis_1.hdf5	22.04.2024 1:07	Файл "HDF5"	7 813 891 КБ
analysis_2	19.04.2024 23:38	Исходный файл ...	17 757 290 КБ
analysis_2.hdf5	22.04.2024 1:07	Файл "HDF5"	7 813 891 КБ

Рисунок 4. Информация о преобразованных файлах

```
D:\PycharmProjects\Vaex\venv\Scripts\python.exe D:/PycharmProjects/Vaex/vaex.py
Время работы: 0:18:26.631318
```

Рисунок 5. Время, затраченное на конвертирование файлов Vaex

спечения корректной обработки данных было выполнено их повторное преобразование посредством использования функционала второй библиотеки.

Код для конвертирования данных с помощью Dask приведен на рисунке 6.

Конвертирование данных заняло 1837,88 секунд (рисунок 7).

После этого используя различные операции над данными для анализа и визуализации, проверяем производительность данных библиотек. Результаты и названия операций представлены в таблице 2.

Таблица 2. Временные затраты (второй шаг эксперимента)

Операция	Vaex, с.	Dask, с.
Конвертация из CSV в HDF5	1106,63	1837,88
Подсчет всех данных	14,07	Ошибка
Фильтрация(отбор) данных	75,19	138,09
Среднее арифметическое чисел	64,53	133,01
Отфильтрованное среднее	18,17	Ошибка
Отфильтрованный параметр по колоннам	45,97	503,49
Отфильтрованная сумма столбцов	226,08	Ошибка
Вывод 20 строк	440,22	12,55
Добавить колонну	16,65	Ошибка
Ленивая оценка	33,13	221,00
Чтение файла	156,01	5,02
Количество всех значений	120,00	Ошибка

«Ошибка» в таблице 2 означает невозможность выполнения операции, ввиду «торможения» системы и полную остановку ядра.

Выводы

В результате проведенного исследования и сравнительного анализа библиотек Vaex и Dask при работе с данными на локальном компьютере, можно сделать следующие выводы.

Необходимость преобразования CSV в формат HDF5 [9] в Vaex может потребовать дополнительного времени, однако это может быть оправдано высокой оптимизацией и возможностью проведения интерактивного анализа на наборах данных, превышающих объем основной памяти компьютера.

Выявилась особенность преобразования файлов Vaex с использованием функции конвертирования. Файлы, преобразованные подобным образом, могут использоваться только с Vaex; попытка их чтения с помощью Dask приводит к возникновению ошибки формата, источником которой может являться использование оптимизированного варианта HDF5 в Vaex.

Dask работает медленнее на локальной машине по сравнению с первой библиотекой, но остается более эффективным решением для вычислительных кластеров. Также стоит отметить, что Dask основан на библиотеке Pandas [12, 13]; это позволяет ему интегрироваться с другими популярными Python-библиотеками, такими как Scikit-learn, NumPy и Scipy, значительно облегчая тем самым анализ данных, процесс машинного обучения и проверку статистических гипотез [10, 14].

Таким образом, выбор между Vaex и Dask зависит от конкретных задач и условий использования: Vaex может быть более предпочтителен при работе с большими объемами данных на одной машине, в то время как Dask может иметь приоритет при использовании вычислительных кластеров. Следовательно, можно утверждать, что проверяемая гипотеза является истинной.

```
import datetime
import dask.dataframe as df

# фиксируем и выводим время старта работы кода
start = datetime.datetime.now()
ds = df.read_csv('csv_files/*.csv')
ds.to_hdf('hdf5_files_dask/analysis_03.hdf5', key='table')
# фиксируем и выводим время окончания работы кода
finish = datetime.datetime.now()
# вычитаем время старта из времени окончания
print('Время работы: ' + str(finish - start))
```

Рисунок 6. Листинг программы для конвертирования Dask

```
D:\PycharmProjects\Vaex\venv\Scripts\python.exe D:/PycharmProjects/Vaex/dask.py
Время работы: 0:30:37.881006
```

Рисунок 7. Затраченное время на конвертирование файлов Dask

Литература

1. What is Vaex? URL: https://vaex.readthedocs.io/en/latest/index.html (дата обращения: 15.04.2024).
2. Dask – Dask documentation. URL: https://docs.dask.org/en/stable/ (дата обращения: 15.04.2024).
3. GitHub – dask/dask: Parallel computing with task scheduling. URL: https://github.com/dask/dask (дата обращения: 16.04.2024).
4. NumPy. URL: https://numpy.org/ (дата обращения: 16.04.2024).
5. GitHub – vaexio/vaex. URL: https://github.com/vaexio/vaex (дата обращения: 17.04.2024).
6. Dask vs Vaex – a qualitative comparison. URL: https://vaex.io/blog/dask-vs-vaex-a-qualitative-comparison (дата обращения: 17.04.2024).
7. Как использовать HDF5-файлы в Python. URL: https://habr.com/ru/companies/otus/articles/416309/ (дата обращения: 17.04.2024).
8. 52 датасета для тренировочных проектов. URL: https://habr.com/ru/companies/edison/articles/480408/ (дата обращения: 18.04.2024).
9. Vaex и Dask: когда Pandas не может обработать большие данные. URL: https://python-school.ru/blog/analiz-dannyh/vaex-vs-dask/ (дата обращения: 18.04.2024).
10. Использование библиотеки Vaex для обработки больших объемов данных. URL: https://newtechaudit.ru/ispolzovanie-biblioteki-vaex-dlya-obrabotki-bolshih-obyomov-dannyh/ (дата обращения: 19.04.2024).
11. Анализ данных с использованием библиотеки Dask. URL: https://habr.com/ru/companies/otus/articles/759552/ (дата обращения: 19.04.2024).
12. Груздев А.В., Хейдт М. Изучаем pandas / пер. с англ. А.В. Груздева. М.: ДМК, 2019. 682 с.
13. Уэс М. Python и анализ данных. Первичная обработка данных с применением pandas, Numpy и Jupiter / пер. с англ. А.А. Слинкина, 3-е изд. М.: ДМК, 536 с.
14. Васильев Ю.А. Python для data science. СПб.: Питер, 272 с.

Получено 23.04. 2024

Пальмов Сергей Вадимович, к.т.н., доцент, доцент кафедры информационных систем и технологий (ИСТ) Поволжского государственного университета телекоммуникаций и информатики (ПГУТИ). 443090, Российская Федерация, г. Самара, Московское шоссе, 77; доцент кафедры информационных технологий Самарского государственного технического университета. 443100, Самара, ул. Молодогвардейская, 244. Тел. +7 927 706-61-21. E-mail: s.palmov@psuti.ru

Шаталов Никита Владимирович, студент кафедры ИСТ ПГУТИ. 443090, Российская Федерация, г. Самара, Московское шоссе, 77. Тел. +7 999 170-27-28. E-mail: nickit.schatalow@yandex.ru

PERFORMANCE COMPARISON OF THE VAEX AND DASK LIBRARIES

Palmov S.V. [1,2], Shatalov N.V. [1]

[1]Povolzhskiy State University of Telecommunications and Informatics, Samara, Russian Federation

[2]Samara State Technical University, Samara, Russian Federation

E-mail: s.palmov@psuti.ru, nickit.schatalow@yandex.ru

The purpose of the study was to compare the performance of the Vaex and Dask libraries, designed to enhance data processing efficiency. In this regard, experiments involving the assessment of time consumption for various classes of operations were conducted. The research included dataset preparation, data sampling, environment configuration execution, installation and setup of the aforementioned modules, Python script development, performance testing and subsequent analysis of the results obtained. It was observed that Vaex exhibits high performance when processing large datasets comprising of million objects on a single local machine; Dask's metrics performance is inferior to the former library. This fact indicates that Vaex is a more efficient tool for processing large datasets under conditions similar to those used in this study. The results and conclusions of the study emphasize the importance of choosing the optimal library when processing large volumes of data, and also confirm the advantages of the Vaex library in this context.

Keywords: *Vaex, Dask, Python, big data, data processing*

DOI: 10.18469/ikt.2024.22.1.12

Palmov Sergey Vadimovich, Povolzhskiy State University of Telecommunications and Informatics, 77, Moscovskoye shosse, Samara, 443090, Russian Federation; Associated Professor of Information

Systems and Technologies Department, PhD in Technical Science. Samara State Technical University, 244, Molodogvardeyskaya Street, Samara, 443100, Russian Federation; Associated Professor of Technologies Department. Tel. +7 927 706-61-21. E-mail: s.palmov@psuti.ru

Shatalov Nikita Vladimirovich, Povolzhskiy State University of Telecommunications and Informatics, 77, Moscovskoye shosse, Samara, 443090, Russian Federation; Student of Information Systems and Technologies Department. Tel. +7 999 170-27-28. E-mail: nickit.schatalow@yandex.ru

References

1. What is Vaex? URL: https://vaex.readthedocs.io/en/latest/index.html (accessed: 15.04.2024).
2. Dask – Dask documentation. URL: https://docs.dask.org/en/stable/ (accessed: 15.04.2024).
3. GitHub – dask/dask: Parallel computing with task scheduling. URL: https://github.com/dask/dask (accessed: 16.04.2024).
4. NumPy. URL: https://numpy.org/ (accessed: 16.04.2024).
5. GitHub – vaexio/vaex. URL: https://github.com/vaexio/vaex (accessed: 17.04.2024).
6. Dask vs Vaex – a qualitative comparison. URL: https://vaex.io/blog/dask-vs-vaex-a-qualitative-comparison (accessed: 17.04.2024).
7. How to use HDF5 files in Pytho. URL: https://habr.com/ru/companies/otus/articles/416309/ (accessed: 17.04.2024). (In Russ.)
8. 52 datasets for training projects. URL: https://habr.com/ru/companies/edison/articles/480408/ (accessed: 18.04.2024). (In Russ.)
9. Vaex and Dask: when Pandas cannot process big data. URL: https://python-school.ru/blog/analiz-dannyh/vaex-vs-dask/ (accessed: 18.04.2024). (In Russ.)
10. Using the Vaex library for processing large amounts of data. URL: https://newtechaudit.ru/ispolzovanie-biblioteki-vaex-dlya-obrabotki-bolshih-obyomov-dannyh/ (accessed: 19.04.2024). (In Russ.)
11. Data analysis using the Dask library. URL: https://habr.com/ru/companies/otus/articles/759552/ (accessed: 19.04.2024). (In Russ.)
12. Gruzdev A.V., Heidt M. *Studying Pandas*. Transl. From English by A.V. Gruzdev. Moskow: DMK, 2019, 682 p. (In Russ.)
13. Ues M. *Python and Data Analysis. Primary Data Processing Using Pandas, Numpy and Jupiter*. Transl. From English by A.A. Slinkin. 3nd ed. Moscow: DMK, 536 p. (In Russ.)
14. Vasiliev Yu.A. *Python for Data Science.* Saint Petersburg: Piter, 272 p. (In Russ.)

Received 23.04.2024

УДК 004.89

РАЗРАБОТКА ИНСТРУМЕНТА ОБРАБОТКИ ЕСТЕСТВЕННОГО ЯЗЫКА ДЛЯ РЕШЕНИЯ ПРИКЛАДНОЙ ЗАДАЧИ ИЗВЛЕЧЕНИЯ СТАТИСТИЧЕСКИХ ДАННЫХ ИЗ ТЕКСТА

Захарова О.И., Бедняк С.Г.

Поволжский государственный университет телекоммуникаций и информатики, Самара, РФ

E-mail: o.zaharova@psuti.ru

Текстовая аналитика используется для изучения текстового содержимого и получения новых переменных из необработанного текста, которые можно использовать в качестве входных данных для моделей прогнозирования или других статистических методов, в том числе при решении фундаментальных задач. Цель исследования: проанализировать алгоритмы машинного обучения, практические наработки в этой области и разработать интегрируемый программный инструмент обработки текста, используя структуру алгоритма, на основе библиотек BasicStats, ReadabilityStats, SovChLit, позволяющий извлекать статистику из текстов большого объема на русском языке. Реализован метод извлечения статистических данных из необработанных текстов больших объемов на основе машинного обучения и обработки естественного языка на языке Python, с возможностью встраивания в другие проекты. Разработан программный инструмент, использующий функционал адаптированной для русского языка