

Санкт-Петербургский государственный университет

Кафедра информационно-аналитических систем

Группа 23.М07-мм

Разработка web-приложения для разметки датасетов

Коновалов Илья Олегович

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
Ассистент кафедры ИАС Г.А. Чернышев

Санкт-Петербург
2023

Оглавление

Введение	3
1. Постановка задачи	5
2. Обзор	6
2.1. Технологии	6
3. Реализация	8
3.1. Серверная часть	8
3.2. Клиентская часть	9
3.3. Функциональность	9
3.4. Внешний вид приложения	10
3.5. Тестирование и качество кода	11
4. Эксперимент	12
4.1. Датасет для тестирования	12
4.2. Аппаратно-программная конфигурация	12
4.3. Методика эксперимента	13
4.4. Результаты эксперимента	13
4.5. Интерпретация результатов	14
Заключение	15
Список литературы	16

Введение

Искусственный интеллект и, в частности, машинное обучение (ML) активно развиваются и проникают во все сферы жизни общества, одним из актуальных направлений является автоматическая суммаризация текстов. Одна из важных и сложных задач в этой области — суммаризация чатов. Работа с чатами требует особого подхода, поскольку тексты в них зачастую неполные, фрагментированные и содержат большое количество сленга или ошибок. Суммаризация предполагает создание сжатого варианта исходного текста с сохранением ключевых фактов и идей. Основная цель — предоставить читателю концентрированное содержание обсуждения без необходимости вникать в каждое сообщение.

Для суммаризации чатов необходимы размеченные данные, которые встречаются в открытом доступе довольно редко и зачастую являются специфичными для определенной задачи. В работах [8] и [6] было представлено desktop-приложение под названием «Chat Corpora Annotator»¹, предназначенное для разметки чатов и реализованное на языке C#. В нем была реализована следующая функциональность:

- Загрузка датасетов;
- Разметка датасетов;
- Продвинутый поиск по датасетам;
- Выгрузка размеченных датасетов.

Однако, такое решение оказалось неприемлимым по следующим причинам:

- Невозможность поддержки C# приложения;
- Отсутствие кроссплатформенности;

¹<https://github.com/yakovypg/Chat-Corpora-Annotator>

- Невозможность нативной интеграции будущей функциональности связанной с машинным обучением и реализуемой на языке Python.

Таким образом возникла цель создания веб-версии данного приложения на языке Python, которое потенциально могло бы разрешить все обозначенные проблемы.

1. Постановка задачи

Целью работы является создание web-разметчика на языке Python, реализующего основную функциональность «Chat Corpora Annotator». Для её выполнения были поставлены следующие задачи:

1. Выбрать подходящие технологии, библиотеки и фреймворки для реализации приложения;
2. Реализовать следующую функциональность:
 - Загрузка датасетов в формате CSV;
 - Просмотр датасетов;
 - Разметка датасетов;
 - Базовый поиск по отдельному датасету;
 - Выгрузка размеченных датасетов в форматах CSV и JSON.

2. Обзор

2.1. Технологии

2.1.1. Flask

В качестве основной технологии на backend-стороне приложения использовался Flask [2] — легковесный веб-фреймворк для языка Python, который предоставляет минимально необходимый набор инструментов для создания веб-приложений.

Преимущества Flask заключаются в следующем:

- Простота — Flask минималистичен и в этой связи довольно прост в использовании;
- Легковесность — по-умолчанию Flask содержит только самые необходимые для работы компоненты;
- Расширяемость — для Flask существует большое количество расширений и плагинов;
- Обширная база знаний и поддержка комьюнити — фреймворк существует с 2010 года и пользуется большой популярностью.

2.1.2. jQuery

Для реализации части frontend-функциональности использовался jQuery [10] — кроссбраузерная JavaScript библиотека, созданная для упрощения веб-разработки по сравнению со стандартным JavaScript. jQuery решает такой круг задач, как манипуляция DOM деревом, обработка событий, AJAX и прочее.

2.1.3. Bootstrap 5

В качестве источника визуальных компонентов использовалась библиотека Bootstrap 5 [1], которая является набором инструментов для создания веб-приложений и включает в себя HTML- и CSS-шаблоны

оформления для типографики, веб-форм, кнопок, блоков навигации и прочих компонентов веб-интерфейса, включая JavaScript-расширения.

3. Реализация

3.1. Серверная часть

Как было сказано ранее, на серверной стороне приложения используется фреймворк Flask.

Для работы с датасетами используется Pandas [4] — библиотека для анализа и обработки данных. Для загрузки датасета на сервер необходимо, чтобы он соответствовал определенному формату, а именно имел следующие колонки:

- username — имя пользователя, отправившего сообщение;
- sent — время и дата отправки сообщения в формате ISO 8601;
- text — текст сообщения;
- label — метка сообщения (применимо в случае загрузки уже размеченного датасета).

Размер загружаемого датасета ограничен 2 Мб, поскольку данный размер соответствует около 10000 записей, что вполне достаточно для разметки одним ассессором (более подробно в разделе 4).

Все датасеты хранятся в отдельной папке с данными. При загрузке датасета на сервер происходит сохранение его CSV файла в этой папке, затем к нему по необходимости применяются дополнительные преобразования, такие как добавление колонки индекса (именуемой id) и переформатирование даты и времени.

Для реализации индексации и поиска используется Whoosh [9] — библиотека, реализующая движок для полнотекстового поиска, написанная на языке Python. При загрузке датасета происходит индексация колонок id и text. Индекс датасета помещается в отдельную папку, соответствующую названию датасета.

При поиске по датасету в качестве результата возвращается найденное сообщение с выделенными совпадениями в тексте сообщения. Выделение происходит с помощью HTML тега <mark>.

При переименовании (удалении) датасета переименоывается (удаляется) соответствующая ему папка с индексом.

В качестве хранилища информации о датасетах и метках, принадлежащих каждому датасету, используется база данных SQLite3 [7] в связке с ORM-решением Flask-SQLAlchemy [3]. Между датасетами и метками установлено отношение один-ко-многим, то есть один датасет может иметь несколько меток, в то время как каждая метка принадлежит единственному датасету. В рамках одного датасета все метки должны быть уникальны.

3.2. Клиентская часть

В Flask реализуется стратегия server-side rendering, которая подразумевает, что HTML страница формируется на стороне сервера и отправляется клиенту в готовом виде. Для форматирования страниц используется встроенный во Flask шаблонизатор Jinja2.

Для манипуляций с DOM деревом, выполнения AJAX запросов и дополнительной логики отрисовки компонентов используется библиотека jQuery.

3.3. Функциональность

В приложении была реализована следующая функциональность:

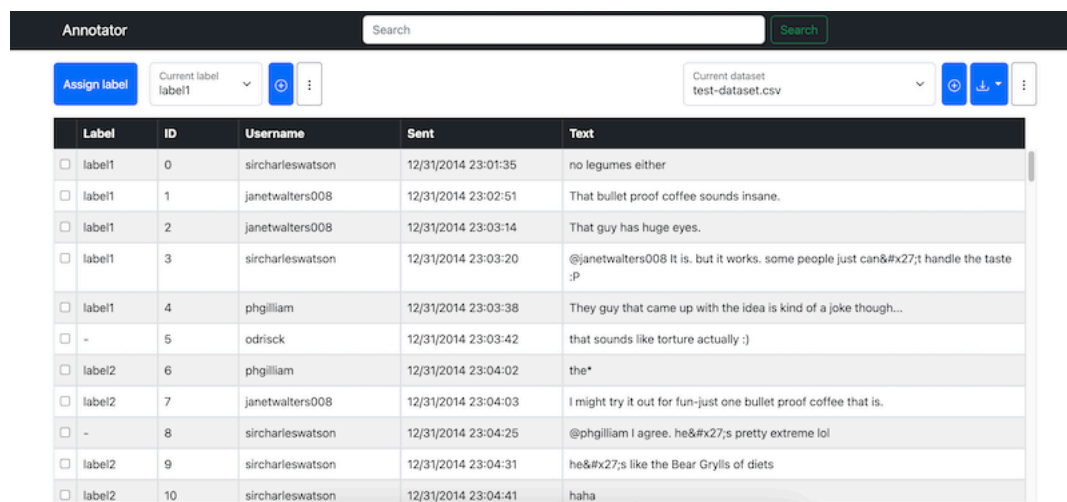
- Работа с датасетами
 - Загрузка
 - Разметка
 - Переименование
 - Удаление
 - Выгрузка в форматах CSV и JSON
- Работа с метками

- Создание
 - Переименование
 - Удаление
- Поиск по фразам

На данный момент приложение не подразумевает совместной работы нескольких пользователей. В будущем планируется добавить возможность разметки датасетов для отдельных пользователей.

3.4. Внешний вид приложения

Глобально приложение состоит из одной станицы, на которой отображается датасет и производится разметка. При выполнении поиска на этой же странице отображаются полученные результаты. Пример отображаемой страницы можно увидеть на Рисунках 1, 2, 3.



Label	ID	Username	Sent	Text
<input type="checkbox"/> label1	0	sircharleswatson	12/31/2014 23:01:35	no legumes either
<input type="checkbox"/> label1	1	janetwalters008	12/31/2014 23:02:51	That bullet proof coffee sounds insane.
<input type="checkbox"/> label1	2	janetwalters008	12/31/2014 23:03:14	That guy has huge eyes.
<input type="checkbox"/> label1	3	sircharleswatson	12/31/2014 23:03:20	@janetwalters008 It is. but it works. some people just can't handle the taste :P
<input type="checkbox"/> label1	4	phgilliam	12/31/2014 23:03:38	They guy that came up with the idea is kind of a joke though...
<input type="checkbox"/> -	5	odrisk	12/31/2014 23:03:42	that sounds like torture actually :)
<input type="checkbox"/> label2	6	phgilliam	12/31/2014 23:04:02	the*
<input type="checkbox"/> label2	7	janetwalters008	12/31/2014 23:04:03	I might try it out for fun-just one bullet proof coffee that is.
<input type="checkbox"/> -	8	sircharleswatson	12/31/2014 23:04:25	@phgilliam I agree. he's pretty extreme lol
<input type="checkbox"/> label2	9	sircharleswatson	12/31/2014 23:04:31	he's like the Bear Grylls of diets
<input type="checkbox"/> label2	10	sircharleswatson	12/31/2014 23:04:41	haha

Рис. 1: Просмотр и разметка датасета

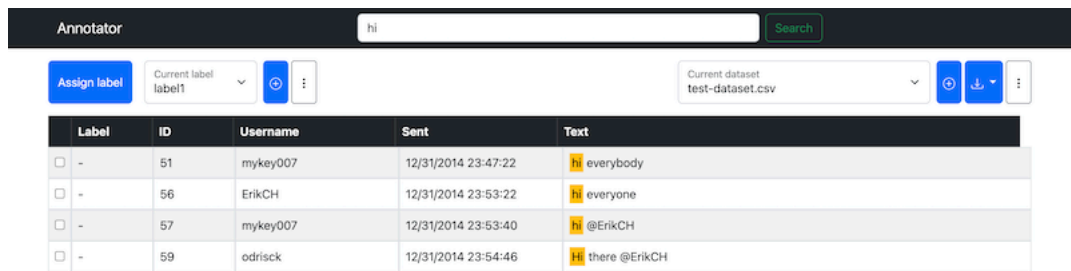


Рис. 2: Поиск по датасету

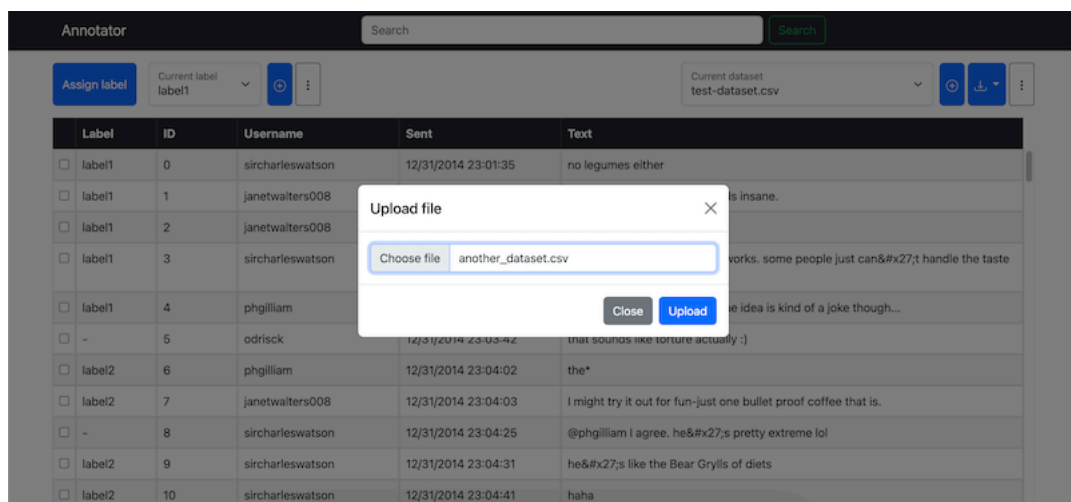


Рис. 3: Загрузка нового датасета

3.5. Тестирование и качество кода

Был написан пакет из 12 тестов, проверяющих логику работы с индексами и датасетами. Тестирование UI проводилось вручную. Также используется CI, в котором происходит проверка с помощью линтера flake8. Форматирование кода осуществляется с помощью black. К проекту написана документация по локальному запуску.

4. Эксперимент

Проводилось тестирование приложения на предмет времени индексации загруженного датасета (как упоминалось ранее, для этого используется библиотека Whoosh). Данный эксперимент проводился с целью выяснения возможностей масштабируемости приложения для работы с более крупными датасетами (как упоминалось в Разделе 3.1, в данный момент поддерживаются датасеты размерностью не более 2 Мб).

4.1. Датасет для тестирования

Для тестирования использовался датасет, содержащий сообщения онлайн-курса по обучению программированию².

Предварительно была произведена следующая обработка:

1. Удаление лишних столбцов;
2. Удаление записей, содержащих нулевые значения хотя бы в одном из столбцов;
3. Переименование столбцов для приведения к требуемому формату,

По итогу был получен датасет размером 500 Мб, содержащий 5037827 записей.

Средняя длина сообщения составила 60 символов ($\sigma = 100$). При таких условиях часть датасета размером 1 Мб будет включать приблизительно 10000 записей.

4.2. Аппаратно-программная конфигурация

Тестирование производилось на устройстве Macbook Air 13, обладающем следующими характеристиками:

- Процессор: Apple Silicon M1
- ОЗУ: 8 Гб

²<https://www.kaggle.com/datasets/freecodecamp/all-posts-public-main-chatroom>

- ОС: MacOS Ventura 13.4
- Размер SSD диска: 256 Гб

Эксперимент запускался в виртуальной машине со следующими характеристиками:

- Кол-во ядер процессора: 6
- ОЗУ: 4 Гб
- ОС: Ubuntu 22.04
- Дисковое пространство: 50 Гб
- Версия Python: 3.11

4.3. Методика эксперимента

Для эксперимента использовались 4 датасета размерами 1 Мб, 10 Мб, 100 Мб, 500 Мб, полученных из исходного за счет усечения.

Датасеты передавались программе для индексации, в ходе которой измерялись время индексирования и размер полученного индекса. Для каждого датасета выполнялось 5 запусков, итоговые результаты усреднялись.

4.4. Результаты эксперимента

По итогам эксперимента были получены результаты, представленные в Таблице 1.

Dataset size (MB)	Indexing time (s)	Index size (MB)
1	3.2 ± 0.2	4.6
10	38.5 ± 2.0	33.2
100	402.6 ± 23.3	298.7
500	3978.0 ± 199.3	12632

Таблица 1: Производительность whoosh

4.5. Интерпретация результатов

По полученным результатам можно судить о том, что Whoosh плохо подходит для индексации больших датасетов. Для улучшения производительности индексации можно применить дополнительные оптимизации структуры индекса, либо использовать альтернативные инструменты, например PyLucene [5].

Заключение

По итогам работы было реализовано веб приложение для разметки датасетов, а именно:

- Были выбраны технологии для реализации;
- Были реализованы визуальное представление и прикладная логика;
- Получен минимально рабочий продукт.

Направления для дальнейшей работы:

- Добавление поддержки многопользовательской разметки;
- Адаптация приложения для работы с большими датасетами;
- Улучшение UI и UX.

Исходный код приложения можно найти в репозитории [TurboGoose/CCA-web](#).

Список литературы

- [1] Bootstrap 5 Documentation. — URL: <https://getbootstrap.com/docs/5.0> (дата обращения: 24 декабря 2023 г.).
- [2] Flask Documentation. — URL: <https://flask.palletsprojects.com/en/latest/> (дата обращения: 24 декабря 2023 г.).
- [3] Flask-SQLAlchemy Documentation. — URL: <https://flask-sqlalchemy.palletsprojects.com/en/3.1.x/> (дата обращения: 24 декабря 2023 г.).
- [4] Pandas Documentation. — URL: <https://pandas.pydata.org/docs/> (дата обращения: 24 декабря 2023 г.).
- [5] PyLucene Documentation. — URL: <https://lucene.apache.org/pylucene/> (дата обращения: 24 декабря 2023 г.).
- [6] Query Processing and Optimization for a Custom Retrieval Language / Yakov Kuzin, Anna Smirnova, Evgeniy Slobodkin, George Chernishev // Proceedings of the First Workshop on Pattern-based Approaches to NLP in the Age of Deep Learning / Ed. by Laura Chiticariu, Yoav Goldberg, Gus Hahn-Powell et al. — Gyeongju, Republic of Korea : International Conference on Computational Linguistics, 2022. — . — P. 61–70. — URL: <https://aclanthology.org/2022.pandl-1.8>.
- [7] SQLite Documentation. — URL: <https://www.sqlite.org/docs.html> (дата обращения: 24 декабря 2023 г.).
- [8] Smirnova Anna, Slobodkin Evgeniy, Chernishev George. [Situation-Based Multiparticipant Chat Summarization: a Concept, an Exploration-Annotation Tool and an Example Collection](#) // Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop / Ed. by Jad Kabbara, Haitao Lin, Amandalynne Paullada, Jannis Vamvas. — Online :

Association for Computational Linguistics, 2021. — . — P. 127–137. — URL: <https://aclanthology.org/2021.acl-srw.14>.

- [9] Whoosh Documentation. — URL: <https://whoosh.readthedocs.io/en/latest/> (дата обращения: 24 декабря 2023 г.).
- [10] jQuery Documentation. — URL: <https://api.jquery.com/> (дата обращения: 24 декабря 2023 г.).