

Семантическая дедупликация текстов с использованием библиотеки Fast Semantic Text Deduplication в среде Google Colab

Анишкова Анастасия Сергеевна

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

Цель данного исследования заключается в разработке и анализе эффективного метода семантической дедупликации текстов с применением современных NLP-моделей и облачных технологий, обеспечивающего высокую точность и скорость обработки данных. Для реализации исследования использовалась облачная среда Google Colab и язык программирования Python. Семантическая дедупликация текстов может быть использована везде, где требуется работа с большими объемами текстовых данных.

Ключевые слова: Семантическая дедупликация, Python, Google Colab, Fast Semantic Text Deduplication.

Semantic text reduplication using the Fast Semantic Text Deduplication library in the Google Colab environment

Anishkova Anastasia Sergeevna

Sholom-Aleichem Priamurskiy State University

Student

Abstract

The purpose of this study is to develop and analyze an effective method of semantic deduplication of texts using modern NLP models and cloud technologies that ensure high accuracy and speed of data processing. The Google Colab cloud environment and the Python programming language were used to implement the study. Semantic text reduplication can be used wherever it is required to work with large amounts of text data.

Key words: Semantic Deduplication, Python, Google Colab, Fast Semantic Text Deduplication.

1 Введение

1.1 Актуальность

Семантическая дедупликация текстов с использованием библиотеки Fast Semantic Text Deduplication в Google Colab актуальна в условиях роста объемов текстовых данных. Традиционные методы не учитывают смысловое сходство, что снижает их эффективность. Современные подходы, такие как модели на основе BERT, обеспечивают высокую точность, а Google Colab делает метод доступным для широкого круга пользователей. Это особенно

важно для анализа больших данных, машинного обучения, обработки новостей, отзывов и других задач, где требуется качественная очистка текстов от дубликатов.

1.2 Обзор исследований

В. Л. Дашонок, А. Д. Хомоненко, В. А. Боброва рассмотрели подход к устранению избыточности семантически близкой текстовой информации в хранилищах и базах данных [1], подход к обработке текста на естественном языке описал Р. А. Вартанян [2], В. Девараджан, Р. Субраманьян провели эффективное изучение данных в веб-масштабе посредством семантической дедупликации [3], анализ семантического сходства текстовых документов с целью выявления практически совпадений провели В. Девараджан, Р. Субраманьян [4], И. М. Нгуэна протестировали методы быстрого обнаружения семантических дубликатов в базах данных [5].

1.3 Цель исследования

Целью исследования является разработка и анализ метода семантической дедупликации текстов с использованием библиотеки Fast Semantic Text Deduplication и современных NLP-моделей в среде Google Colab, направленного на эффективное устранение дубликатов на основе их смыслового сходства при обработке текстовых данных.

2 Материалы и методы

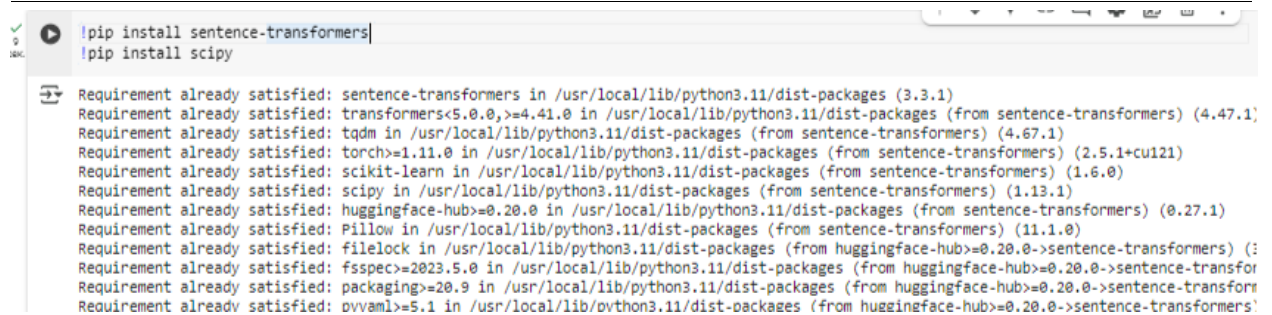
В данном исследовании используется среда google colab и язык программирования питон.

3 Результаты

В современном мире работа с большими данными стала неотъемлемой частью многих процессов, однако их ручной анализ часто оказывается трудоемким и недостаточно качественным, поэтому использование методов семантической дедупликации текстов с помощью программного кода становится эффективным решением для автоматизации этой задачи.

Ссылка на ноутбук
https://colab.research.google.com/drive/1AgyFy2d_T7t9pWbM8AT82CMFj9J2AH-b?usp=sharing.

Первым делом необходимо установить необходимые библиотеки (рис.1): sentence-transformers, которая предоставляет удобный интерфейс для работы с моделями преобразования текста в векторные представления, и scipy, которая предлагает множество функций для математических вычислений, статистики, оптимизации и работы с линейной алгеброй.



```

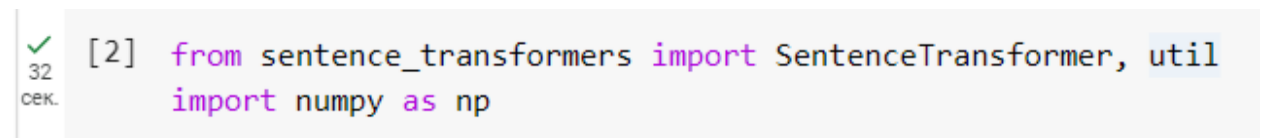
!pip install sentence-transformers
!pip install scipy

Requirement already satisfied: sentence-transformers in /usr/local/lib/python3.11/dist-packages (3.3.1)
Requirement already satisfied: transformers<5.0.0,>=4.41.0 in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (4.47.1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (4.67.1)
Requirement already satisfied: torch>=1.11.0 in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (2.5.1+cu121)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (1.6.0)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (1.13.1)
Requirement already satisfied: huggingface-hub>=0.20.0 in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (0.27.1)
Requirement already satisfied: Pillow in /usr/local/lib/python3.11/dist-packages (from sentence-transformers) (11.1.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.20.0->sentence-transformers) (1.16.0)
Requirement already satisfied: fspec>=2023.5.0 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.20.0->sentence-transformers) (2023.5.0)
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.20.0->sentence-transformers) (24.1)
Requirement already satisfied: pvvaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.20.0->sentence-transformers) (5.1.0)

```

Рисунок 1. Установка библиотек

Далее выполняет импорт необходимых модулей и библиотек для работы с семантическим анализом текста и математическими операциями (рис.2).



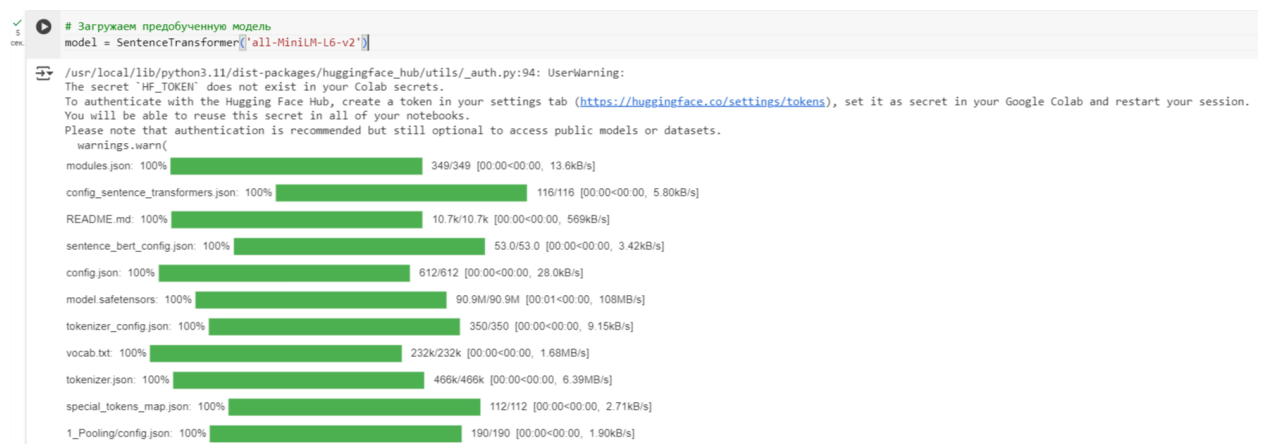
```

[2] from sentence_transformers import SentenceTransformer, util
import numpy as np

```

Рисунок 2. Импорт библиотек

Загрузим предобученную модель all-MiniLM-L6-v2, которая преобразует тексты в векторные представления. Эти векторы затем можно использовать для семантического анализа, поиска сходства, дедупликации и других задач NLP (рис.3).



```

# Загружаем предобученную модель
model = SentenceTransformer('all-MiniLM-L6-v2')

/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
warnings.warn(
modules.json: 100% 349/349 [00:00<00:00, 13.6kB/s]
config_sentence_transformers.json: 100% 116/116 [00:00<00:00, 5.80kB/s]
README.md: 100% 10.7k/10.7k [00:00<00:00, 569kB/s]
sentence_bert_config.json: 100% 53.0/53.0 [00:00<00:00, 3.42kB/s]
config.json: 100% 612/612 [00:00<00:00, 28.0kB/s]
model.safetensors: 100% 90.9M/90.9M [00:01<00:00, 108MB/s]
tokenizer_config.json: 100% 350/350 [00:00<00:00, 9.15kB/s]
vocab.txt: 100% 232k/232k [00:00<00:00, 1.68MB/s]
tokenizer.json: 100% 466k/466k [00:00<00:00, 6.39MB/s]
special_tokens_map.json: 100% 112/112 [00:00<00:00, 2.71kB/s]
1_Pooling/config.json: 100% 190/190 [00:00<00:00, 1.90kB/s]

```

Рисунок 3. Запуск предобученной модели

Напишем код, который создает список текстов, который используется как входные данные для семантического анализа, дедупликации, поиска сходства (рис.4).

```
✓ 0 [4] # Пример списка текстов
сек.  texts = [
        "The cat sits on the mat.",
        "A cat is sitting on the mat.",
        "I love programming in Python.",
        "Python is my favorite programming language.",
        "The dog barks loudly.",
        "Cats are sitting on the mat."
    ]
```

Рисунок 4. Список текстов

Далее преобразуем тексты в векторные представления и вычислим матрицу косинусного сходства между всеми парами текстов (рис.5).

```
✓ 0 [5] # Преобразуем тексты в векторные представления
сек.  embeddings = model.encode(texts, convert_to_tensor=True)

✓ 0 [6] # Вычисляем матрицу косинусного сходства
сек.  cosine_scores = util.cos_sim(embeddings, embeddings)
```

Рисунок 5. Преобразования и вычисления

Реализуем дедупликацию текстов на основе их семантического сходства. Проверим, являются ли тексты дубликатами, используя матрицу косинусного сходства, и сохраняет только уникальные тексты (рис.6).

```
✓ 0 # Порог для определения дубликатов
сек. threshold = 0.8

# Список для хранения уникальных текстов
unique_texts = []

# Проходим по всем текстам и проверяем дубликаты
for i in range(len(texts)):
    is_duplicate = False
    for j in range(i + 1, len(texts)):
        if cosine_scores[i][j] > threshold:
            is_duplicate = True
            break
    if not is_duplicate:
        unique_texts.append(texts[i])
```

Рисунок 6. Проверка дубликатов

Выводим результаты на экран (рис.7), демонстрируя оригинальные тексты (до дедупликации) и уникальные тексты (после дедупликации).



```
print("Оригинальные тексты:")
print(texts)
print("\nУникальные тексты после дедупликации:")
print(unique_texts)
```

Оригинальные тексты:
['The cat sits on the mat.', 'A cat is sitting on the mat.', 'I love programming in Python.', 'Python is my favorite programming language.', 'The dog barks loudly.', 'Cats are sitting on the mat.']

Уникальные тексты после дедупликации:
['Python is my favorite programming language.', 'The dog barks loudly.', 'Cats are sitting on the mat.']

Рисунок 7. Вывод результатов

Результат показывает, что программа успешно удалила тексты с высоким семантическим сходством, оставив только уникальные тексты.

Библиографический список

1. Дашонок В. Л., Хомоненко А. Д., Боброва В. А. Подход к устранению избыточности семантически близкой текстовой информации в хранилищах и базах данных // Бюллетень результатов научных исследований. 2013. №3 (8). С. 14-22.
2. Вартанян Р. А. Подход к обработке текста на естественном языке // Наукосфера. 2020. №11-2. С. 199-203.
3. Abbas A. et al. Semdedup: Data-efficient learning at web-scale through semantic deduplication // arXiv preprint arXiv. 2023. №2303. С. 25-30.
4. Devarajan V., Subramanian R. Analyzing semantic similarity amongst textual documents to suggest near duplicates // Indonesian Journal of Electrical Engineering and Computer Science. 2022. №3. С. 1703-1711.
5. Nguena I. M. et al. Fast semantic duplicate detection techniques in databases // Journal of Software Engineering and Applications. 2017. №6. С. 529.