Разработка программного модуля извлечения структурированной информации в виде RDF триплетов из неструктурированных данных

Ключевые слова: неструктурированные данные, графы знаний, RDF триплеты

Аннотация

В работе рассмотрена проблема извлечения структурированной информации в виде RDF триплетов из неструктурированных данных. Исследованы методы для решения данной проблемы, такие, как использование RegEx, CoreNLP + Open IE, REBEL AI, SpaCy, Yandex GPT API. Проведен сравнительный анализ этих методов по выбранным 3-ем критериям. По результатам сравнения выявлены основные достоинства и недостатки каждого из аналогов, сделан вывод о том, какими качествами должно обладать разрабатываемое решение для последующего его использование в процессе разработки модели генерации графов знаний на основе неструктурированных данных.

Введение

В литературной сфере существуют неструктурированные данные [1], такие как рецензии, комментарии в социальных сетях и видеообзоры. Эти данные содержат информацию о произведениях, авторах и отзывах, но их разрозненность затрудняет обработку и использование. Графы знаний [2] позволяют упорядочить эту информацию, устанавливать связи между различными элементами и улучшать доступ к данным для анализа [3]. Их использование обеспечивает создание структурированных представлений данных, что обеспечивает организацию поиска информации, автоматизацию анализа и выявление новых связей.

Целью данной работы является исследование методов извлечения структурированной информации (в виде триплетов субъект-предикат-объект) из неструктурированных данных. Объектом исследования являются графы знаний, а предметом — методы извлечения структурированной информации (в виде триплетов субъект-предикат-объект) из неструктурированных данных.

В ходе проведения исследования были решены задачи, перечисленные ниже.

1. Обзор существующих методов извлечения структурированной информации (в виде триплетов субъект-предикат-объект) из неструктурированных данных.
2. Сравнение существующих методов на основе выбранных критериев.
3. Выводы о качестве существующих методов извлечения структурированной информации (в виде триплетов субъект-предикат-объект) из неструктурированных данных.

Обзор предметной области

Принцип отбора аналогов

В качестве аналогов для извлечения триплетов из неструктурированных данных были отобраны методы, которые применимы для обработки текстовых данных и извлечения информации из них. Поиск аналогов осуществлялся с использованием ресурсов Google Scholar, GitHub, а также специализированных библиотек и фреймворков для обработки текста, таких как SpaCy, CoreNLP, и другие. Для подбора аналогов использовались следующие запросы: “Information Extraction”, “OpenIE”, “Text Parsing”, “Relation Extraction”, “Entity Extraction”. Отбор осуществлялся по критериям: 1. Наличие публикаций или документации. 2. Наличие примеров использования. 3. Применимость для извлечения триплетов (в виде субъект-предикат-объект).

Использование регулярных выражений (RegEx)

Использование регулярных выражений (RegEx) [4] для поиска триплетов — метод, который использует шаблоны для поиска и извлечения информации из текста. Данный метод применяется для извлечения конкретных шаблонов из текста (электронные адреса, номера телефонов и т.д.), для извлечения структурированных данных из HTML-страниц и т.д.

REBEL AI

REBEL (Relation Extraction By End-to-End Language Generation) [5] — метод на основе модели BART [6], который извлекает триплеты (субъект-связь-объект) из текста. REBEL способен учитывать контекст для извлечения контекстуально зависимых триплетов.

SpaCy

SpaCy [7] — библиотека для обработки естественного языка, которая включает в себя ряд инструментов для извлечения информации из текста, такие как токенизация (процесс разбиения текста на слова, предложения и другие синтаксические единицы), разметка частей речи, синтаксический анализ и другие. Модели, предоставляемые библиотекой, могут быть использованы для извлечения триплетов из текста.

Stanford CoreNLP + Open IE

CoreNLP [8] — это библиотека для обработки естественного языка от Стэнфордского университета. Open Information Extraction (Open IE) [9] — семейство методов, которые выделяют связи и сущности из текста в свободной форме. Open IE ориентирован на извлечение фактов из предложений без предварительно заданной онтологии.

Yandex GPT API

Yandex GPT [10] API — это API (Application Programming Interface), предоставляющий доступ к модели Yandex GPT-lite, которая разработана для обработки и генерации текстов

на основе ИИ (Искусственного интеллекта). Эта модель относится к категории LLM (Large Language Model).

Критерии сравнения аналогов

Алгоритм извлечения

Данный критерий описывает детали алгоритма, используемого для извлечения триплетов (субъект-связь-объект). Инструмент может использовать различные методы машинного обучения, такие как нейронные сети, может использовать шаблоны и правила обработки текста.

Полученный результат

Данный критерий описывает, какой результат был получен (количество триплетов, дополнительные данные, и т.д.). В зависимости от выбранного метода, результат может быть различным.Для оценки критериев был использован текст, содержащий первые 5 предложений из текста, который был получен из WikiPedia по запросу “Dune (Frank Herbert)”.

Время работы

Данный критерий описывает время, затрачиваемое на извлечение триплетов в секундах. Время замерялось на системе с процессором AMD Ryzen 7 4800H, оперативной памятью 24 ГБ и 64-разрядной операционной системой. В качестве текста для извлечения использовался текст, описанный в предыдущем критерии.

Таблица сравнения аналогов

Таблица 1 — Сравнение аналогов по критериям.

Аналог	Алгоритм извлечения	Полученный результат	Время работы (с)
RegEx	Использование шаблонов для поиска конкретных паттернов в тексте, таких как имен, действия и объекты.	19 наборов по 3 слова, которые нельзя назвать триплетами. Данный метод бы выделил триплеты вида субъект-предикат-объект, если бы текст состоял из предложений по такому шаблону: “subject predicate object”.	0.05
REBEL AI	Использование этапов обработки текста, таких как: токенизация, pos tagging (маркировка частей речи для каждого слова) , обучение модели на размеченных данных для	26 триплетов, которые соответствуют виду субъект-предикат-объект,	63

Аналог	Алгоритм извлечения	Полученный результат	Время работы (с)
	предсказания триплетов, использование извлеченных признаков для классификации и выделения триплетов в новых текстах.		
SpaCy	Использование этапов обработки текста, таких как: токенизация, pos tagging (маркировка частей речи для каждого слова) , извлечение сущностей (выявление именованных сущностей), извлечение триплетов.	11 триплетов, которые соответствуют виду субъект-предикат-объект,	57
CoreNLP + Open IE	Использование этапов обработки текста, таких как: токенизация, pos tagging (маркировка частей речи для каждого слова) , синтаксический анализ (построение синтаксического дерева для определения зависимостей между словами), извлечение отношений с помощью Open IE.	100 триплетов, которые соответствуют виду субъект-предикат-объект, однако в результате присутствуют идентичные по содержанию триплеты, например Frank Herbert|have|Dune и Frank Herbert|have|Dune, it|be|although broadcast и it|be|broadcast. Однако в результате у каждого триплета был получен параметр confidence — критерий уверенности корректности триплета (содержимое триплета соответствует содержанию текста)	6.5
Yandex GPT API	Использование собственного механизма внимания, чтобы находить взаимосвязи между словами и сущностями в тексте. Этапы обработки: генерация контекста, выделение сущностей и отношений, использование обучения с подкреплением для того, чтобы точнее извлекать триплеты.	Получен ответный запрос, содержащий 9 триплетов, которые соответствуют виду субъект-предикат-объект	2.5

Выводы по итогам сравнения

На основе проведенного анализа и полученных результатов, представленных в таблице 1,

были сформулированы основные выводы.

- Использование регулярных выражений является самым быстрым и простым способом определения триплетов, но выделить их корректно можно только в тексте, который представляет собой набор предложений из 3 слов .
- Yandex GPT API извлек наименьшее количество триплетов из текста среди всех рассмотренных аналогов.
- REBEL AI извлек триплеты из текста за наибольшее время работы, это объясняется его ресурсоемкостью, так как он использует трансформерные модели [11], которые сами по себе являются ресурсоемкими из-за вычислений и своей архитектуры.
- SpaCy и CoreNLP + Open IE имели схожий алгоритм извлечения, однако CoreNLP + Open IE извлек наибольшее число триплетов из текста, к тому же CoreNLP + Open IE извлек триплеты за меньшее время работы, чем SpaCy(6.5 с против 57).

Выбор метода решения

Решение должно представлять собой программный модуль на Python 3.12 для автоматического извлечения триплетов из неструктурированных данных. Данный модуль в дальнейшем должен быть интегрирован в модель для генерации графа знаний из неструктурированных данных. Полученные триплеты должны быть представлены в формате [субъект, предикат, объект]. В качества опорного был выбран алгоритм для извлечения триплетов, который был использован при извлечении триплетов с помощью CoreNLP + Open IE, так как в процессе использования данного метода было получено наибольшее количество триплетов, к тому же время работы не превысило 10 секунд.

Описание метода решения

Программный модуль извлечения структурированной информации в виде RDF триплетов из неструктурированных данных будет опираться на алгоритм для извлечения RDF триплетов, реализованный с помощью CoreNLP + Open IE, таким образом будет обеспечено извлечение наибольшего количества RDF триплетов из неструктурированного текста за непродолжительное время (в случае для 5 предложений с помощью CoreNLP + Open IE извлеклось 100 триплетов за 6.5 секунд).Сама Программный модуль будет извлекать триплеты из текста поэтапно, после чего в дальнейшем будет проводиться оценка полученных триплетов, чтобы определить их корректность.

Описание этапов обработки текста:

1. Токенизация — разбиение текста на слова, предложения и другие синтаксические единицы;
2. Разметка частей речи (POS tagging) — определение частей речи для каждого слова;
3. Синтаксический анализ — построение синтаксического дерева для определения зависимостей между словами;
4. Извлечение отношений — анализ синтаксической структуры предложения для извлечение триплетов подобно тому, как это происходит с помощью Open IE: поиск отношений между сущностями на основе синтаксических зависимостей, например,

из дерева зависимостей, которое было получено на предыдущем этапе, выбирается подлежащее, глагол и дополнительное свойство, которое их связывает, таким образом получается RDF триплет.

В качестве технологий будут использоваться ЯП Python 3.12, так как он широко применяется для обработки естественного языка, и для него существует множество библиотек и фреймворков, предназначенных для выполнения таких задач. Также будет использоваться библиотека для обработки естественного языка NLTK, которая предоставляет набор инструментов для выполнения основных задач обработки текста. Так как данная библиотека является популярной, существует большое количество примеров кода для работы с текстом с её использованием.

Заключение

В рамках данного исследования был проведен обзор существующих методов для извлечения RDF триплетов. Было отобрано 5 аналогов и проведено их сравнение по 3 выбранным критериям: алгоритм извлечения, полученный результат, время работы. В ходе сравнения были выявлены основные недостатки и достоинства аналогов.

В результате было принято решение реализовать программный модуль, который будет решать задачу извлечения RDF триплетов из неструктурированных данных и будет написан на языке программирования Python, а его основный алгоритм будет опираться на алгоритм извлечения RDF триплетов, реализованный с помощью CoreNLP + Open IE.

Направление дальнейших исследований — разработка программного модуля для извлечения структурированной информации из неструктурированных данных, который планируется использовать при создании модели генерации графа знаний на тему литературы.

Список использованных источников

1. Buneman P. et al. Adding structure to unstructured data //Database Theory—ICDT’97: 6th International Conference Delphi, Greece, January 8–10, 1997 Proceedings 6. – Springer Berlin Heidelberg, 1997. – С. 336-350.
2. Hogan A. et al. Knowledge graphs //ACM Computing Surveys (Csur). – 2021. – Т. 54. – №. 4. – С. 1-37.
3. Hofer M. et al. Construction of knowledge graphs: current state and challenges //Information. – 2024. – Т. 15. – №. 8. – С. 509. URL: https://www.mdpi.com/2078-2489/15/8/509
4. Medeiros S., Mascarenhas F., Ierusalimschy R. From regexes to parsing expression grammars //Science of Computer Programming. – 2014. – Т. 93. – С. 3-18.
5. Хугет-Кабот П. Л., Навигли Р. REBEL: Relation Extraction By End-to-end Language generation // Findings of the Association for Computational Linguistics: EMNLP 2021. – 2021. – С. 2370–2381. URL: https://aclanthology.org/2021.findings-emnlp.204
6. Lewis M. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension //arXiv preprint arXiv:1910.13461. – 2019.
7. Vasiliev Y. Natural language processing with Python and spaCy: A practical introduction.

– No Starch Press, 2020.
8. Manning C. D. et al. The Stanford CoreNLP natural language processing toolkit //Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations. – 2014. – С. 55-60. URL: https://aclanthology.org/P14-5010.pdf
9. Stanovsky G. et al. Open ie as an intermediate structure for semantic tasks //Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers). – 2015. – С. 303-308. URL: https://aclanthology.org/P15-2050.pdf
10. Dmitrijev A. V., Krupnova E. S., Protopopova A. A. Metaphors and Analogies in the Context of Large Language Models //International Conference on Professional Culture of the Specialist of the Future. – Cham : Springer Nature Switzerland, 2024. – С. 326-341.
11. Михайлов Д. В., Емельянов Г. М. Трансформерные модели BERT, взаимное сходство смыслов коротких текстов и их ранжирование по близости эталону //Москва. – 2023. – Т. 12. – С. 159-161.