Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 22.М07-мм

Автоматическая подстановка параметров текстовых макросов в корректной форме

Карими Хурматулла

Отчёт по учебной практике

Научный руководитель:
к. ф.-м. н.доцент Д. В. Луцив

Санкт-Петербург
2022

Оглавление

Введение

Достаточно много технических текстов пишутся с использованием языков разметки. Примеры — Википедия [1], разные справочники, документация открытых проектов (скажем, на GitHub [2]) с использованием Markdown, reStructuredText [3] и прочих. В некоторых из этих языков поддерживается повторное использование (DITA [4], DRL [5]) фрагментов текста. В некоторых (MediaWiki [6], Markdown [7]) — гиперссылки. Повторно используемый фрагмент может, в зависимости от контекста, быть в разных формах (падеж, число, спряжение). Большинство языков разметки исходно делались для английского языка, в котором разные формы пишутся одинаково. Это упрощает задачу автоматической подстановки: в английский документ можно вставить термин просто макропроцессором, потому что его не надо склонять. А в Википедии внутренняя ссылка автоматически сгенерируется по формату WikiWord.

С русским (и прочиим славянскими) и рядом других языков так не получится. Поэтому, например, в русской Википедии ссылки на русские страницы часто приходится писать в формате «`Подробнее об этом на [Русской Странице РусскаяСтраница]`» вместо просто «`Подробнее об этом на РусскаяСтраница`».

[1] https://ru.wikipedia.org/
[2] https://github.com/
[3] https://docutils.sourceforge.io/rst.html
[4] https://www.dita-ot.org/
[5] https://docs.drools.org/
[6] https://www.mediawiki.org/wiki/MediaWiki
[7] https://www.markdownguide.org/

1. Цель работы

Разработать и реализовать систему автоматической подстановки терминов в текст в нужной форме.Для достижения обозначенной цели были поставлены следующие задачи:

1. Изучить инструменты анализа тектов на естственных языках, позволяющие анализировать предложения.

2. Выбрать модели машинного обучения, которые позволят, обучившись на корректных текстах выбирать нужные формы для подстановки фрагментов в текст.

3. Провести эксперименты с этими моделями, выбрать наиболее подходящую.

4. Реализовать прототип инструмента, выполняющего макроподстановку словосочетаний в корректном падеже и числе.

2. Обзор

2.1. Обзор NLTK

Инструментарий естественного языка (NLTK) - это платформа, используемая для создания программ на Python, которые работают с данными человеческого языка для применения в статистической обработке естественного языка (NLP).Он содержит библиотеки обработки текста для токенизации, синтаксического анализа, классификации, выделения элементов, пометки и семантического обоснования. Он также включает графические демонстрации и примеры наборов данных, а также сопровождается кулинарной книгой и книгой, в которой объясняются принципы, лежащие в основе задач обработки языка, которые поддерживает NLTK [10].Итак, как мы знаем, NLTK слишком велик, чтобы его объяснять, но мы сосредоточимся на очень специфической области нашей работы: создании структур и внедрении различных анализаторов для наших структур.

2.1.1. Рекурсия в синтаксической структуре

Грамматика называется рекурсивной, если категория, встречающаяся в левой части произведения, также появляется в правой части произведения, как показано рис 1. Производственный Nom -> Adj Nom (где Nom - категория номиналов) включает прямую рекурсию по номинации категории, тогда как косвенная рекурсия по S возникает в результате комбинации двух производств, а именно S -> NP VP и VP -> V S. [4]

2.1.2. Синтаксический анализ рекурсивного спуска (метод сверху вниз)

Это один из самых простых методов синтаксического анализа, который мы можем использовать для наших грамматических структур. Этот метод делит цели на подцели, и каждая подцель подразделяется на другую подцель до тех пор, пока структура ваших предложений не будет завершена. [5]

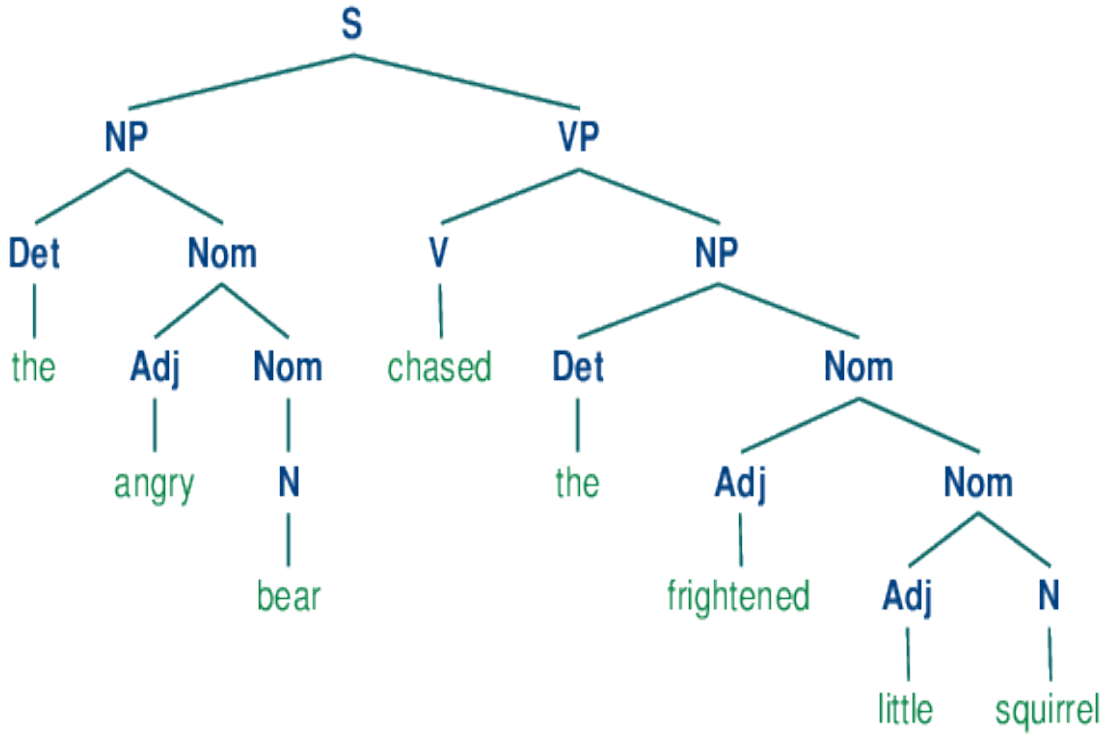

Рис. 1: Пример рекурсивности в синтаксической структуре)

2.2. PyMorphy2

Прежде чем мы начнем обсуждать, что такое pymorphy2, давайте обсудим морфологический анализ.Морфологический анализ - это анализ внутренних структур слов, поэтому, если мы увидим, что русский и украинский языки имеют богатую морфологию, поэтому, используя морфологический анализ, мы определим состояние наших слов, что это состояние, например, существительное или глагол [2]. Pymorphy2 это морфологический анализатор, который анализирует русские тексты с помощью словаря opencorpora. Алгоритм Pymorphy выполняет морфологическую обработку на основе грамматической характеристики типа (слова, лемматизация), но если слово не существует в словаре, поэтому предиктор в pymorphy2 объединит два алгоритма: 1 - по префиксу 2 - по окончанию слов. В то же время, когда мы анализируем слова, мы столкнемся с несколькими состояниями слов и оценками, затем, выбрав состояние слова с наивысшим баллом Мы можем исправить ваше предложение. [7]. Морфологический анализатор pymorphy2 умеет:

1. приводить слово к нормальной форме (например, “люди -> чело-

век", или "гулял -> гулять").

2. ставить слово в нужную форму. Например, ставить слово во множественное число, менять падеж слова и т.д.

3. возвращать грамматическую информацию о слове (число, род, падеж, часть речи и т.д.)[8]

2.3. LightGBM: Высокоэффективное повышение градиента Дерево решений

Это один из наиболее распространенных алгоритмов искусственного интеллекта, который использовался в многоклассовой классификации, прогнозировании кликов. В настоящее время перед GBP стоит задача получения информации по каждой функции и их расчета, поскольку это отнимает много времени. Итак, для решения этой проблемы GBPT внедрила два метода: 1 - ГОСС (односторонняя выборка на основе градиента): Итак, для наших экземпляров данных нет предопределенных весов, и мы говорим, что разные экземпляры данных с разным градиентом играют разную роль в вычислениях, и в результате мы можем сказать, что больше информации может быть получено из экземпляров больших данных с большие градиенты. 2 - EFB (эксклюзивный пакет функций) - Обычно в реальных приложениях дополнительные функции создают эффективный подход без потерь. [1].

2.4. LemmInflect

LemmInflect использует словарный подход для лемматизации английских слов и преобразования их в формы, заданные предоставленным пользователем тегом Universal Dependencies или Penn Treebank. Библиотека работает со словами, не входящими в словарный запас (OOV), применяя методы нейронных сетей для классификации словоформ и выбора соответствующих правил морфинга [3].

2.5. NLTK4Russian

Как было написано, проект направлен на создание лингвистического комплекса для изучения корпусов русскоязычных текстов, основанного на различных методах и алгоритмах в рамках наиболее популярных инструментов современной компьютерной лингвистики (NLTK, Pattern, GenSim и т.д.). Таким образом, в этом проекте они имеют работал над двумя основными задачами:

1. разработка морфологического анализатора для русского языка на основе NLTK и Pymorphy

2. Создание парсера для русского языка на основе категориальной грамматики и парсера, встроенного в nltk

В рамках проекта производится разработка синтаксического анализатора для русского языка. Модуль синтаксического анализа в NLTK позволяет исследователям самостоятельно создавать формальные грамматики различных типов для разных естественных языков и применять их в конкретных целях автоматической обработки текстов. [6].

2.6. Метод повторного использования документации семейств программных продуктов

Существуют существующие технологии разработки многоразовой технической документации с акцентом на возможность повторного использования, как указано в диссертации Романовского Константина Юрьевича. Существует два типа повторного использования: случайное и запланированное.

Доступное повторное использование предполагает, что разработчики подключают существующие компоненты, когда и если представится такая возможность. Планируемое повторное использование предполагает систематическую разработку компонентов, подготовленных для повторного использования, и дальнейшее применение таких компонентов в процессе разработки.

В методе SPP есть два прогрессивных метода: упреждающий и гибкий. Мы сможем начать с нуля в proactive, разработав семейство и добавив в него компоненты. Однако в flexible это похоже на то, как мы начнем использовать продукт, а затем добавим повторно используемые компоненты, но выпуск программного обеспечения отнимает много времени [9].

Заключение

На данный момент изучена предметная область и стек технологий, сделан обзор, сформулирована цель и поставлены задачию

Список литературы

[1] Guolin Ke Qi Meng Thomas Finley Taifeng Wang Wei Chen Weidong Ma Qiwei Ye Tie-Yan Liu. LightGBM: A Highly Efficient Gradient Boosting Decision Tree." Advances in Neural Information Processing Systems 30 (NIPS 2017), pp. 3149-3157.

[2] Korobov Mikhail. Morphological analyzer and generator for Russian and Ukrainian languages. — 2015. — P. 320–332.

[3] LemmInflect. LemmInflect.

[4] NLTK. Рекурсия в синтаксической структуре.

[5] NLTK. Синтаксический анализ рекурсивного спуска.

[6] P. V. Panicheva E. B. Enikeeva Protopopova A. P. Mirzagitova A. D. Moskvina O. A. Mitrofanova. Проект NLTK4Russian.

[7] P. V. Panicheva E. B. Protopopova O. A. Mitrofanova A. P. Mirzagitova. Разработка лингвистического комплекса для морфологического анализа русскоязычных корпусов текстов на основе Pymorphy и NLTK, Труды международной конференции "Корпусная лингвистика – 2015". СПб., 2015. С. 361-373.

[8] Pymorphy2. Морфологический анализатор pymorphy2.

[9] Романовский К. Ю. Метод повторного использования документации семейств программных продуктов / К. Ю. Романовский.

[10] Технопедия. Что означает Natural Language Toolkit (NLTK)?