Софийски университет “Св. Климент Охридски”

Факултет по математика и информатика

Проект

по

“Подходи за обработка на естествен език”

на тема

“Резюмиране на текст и намиране на ключови думи с TextRank”

Изготвен от: Иван Арабаджийски, ФН: 5MI3400052,
спец. Извличане на информация и обработка на знания
10.02.2022

Съдържание

1. Идея на проекта

Текст майнинг е процесът на извличане на важна информация от текст. Най-често срещани задачи са извличане на информация, лексикален анализ, разпознаване на шаблони (pattern recognition), резюмиране, разпознаване на емоции и други. Всички са обединени под шапката на обработката на естествен език. В днешно време с оглед на голямото количество текстови данни автоматичното резюмиране на текст намира много приложения в сфери като системи за провеждане на разговор, анотиране на изображения и най-вече търсачки.

Методите за резюмиране на текст попадат в две категории: extractive (извличащи) и abstractive (абстрактни). Абстрактните са базирани на идеята да се генерират резюмета от текст като се извлече техния смисъл, докато извличащите се съсредоточават върху отделяне на конкретни изречения от текста, които са най-важни според алгоритъма. В миналото абстрактните алгоритми не са се справяли много добре, но в последно време с навлизането на дълбокото обучение резултатите им са се подобрили значително. В този проект ще разгледаме извличащият графов алгоритъм използващ обучение без учител – TextRank. Моделът и алгоритмът ще бъдат имплементирани специално за български език и всички примери ще бъдат на български език.

2. Задачата

Целта на този проект е да анализираме съдържанието на даден текстови документ чрез автоматично резюмиране на текста. Както предполага името задачата се изпълнява на ниво документ, а не на ниво изречение или дума. Най-ранните опити за решаване на задачата разчитат на честота на фразите, позицията им в текста и ключовите фрази като важни за анализа елементи.

Можем да обобщим задачата така:

> *По даден текстов документ с дължина l (т.е. l на брой изречения) като вход, върнете текст с дължина n, по-малка от l, който предава най-важната информация от оригиналния текст.*

Тук е важно да спомена, че “най-важната информация” е субективно понятие, което често зависи от интерпретацията на анализатора.

Ще се опитаме да решим задачата за резюмиране на един документ (съществува и задача за резюмиране на много документи в едно резюме). Методите за създаване на резюме се уповават двата подхода extractive и abstractive, обаче някои методи успяват

да ги комбинират. Абстрактният метод се опитва да генерира резюмето като разбере смисъла на статията и създаде изречения, които не се срещат в оригиналния текст. Извличащият метод генерира новия текст изцяло от изречения намиращи се в документа.

2.1 Извличащи методи за резюмиране на текст

Извличащите методи се стараят да намерят подмножество от изреченията на оригиналния текст за да създадат резюмето. Това се случва в три основни стъпки:

- представяне на данните във формат, който най-добре отговаря на техните свойства
- оценка на изреченията базиранана това представяне
- построяване на резюмето от оценените изречения

2.2 Алгоритъмът TextRank

TextRank е графов алгоритъм. Той представя изреченията като върхове на граф. Сходството между две изречения се изчислява на базата на общите им думи и дължината на изреченията. Ребрата на графа между две изречения имат тегло равно на сходството на двете изречения. Естествено се интересуваме само от тези ребра с ненулева тежест. След построяването на графа можем да раздаден тегла на всеки връх използвайки следната формула:

$$WS(V_i) = (1-d) + d * \sum_{V_j \in \mathrm{In}(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j)$$

където:

- $WS(V_i)$ е претегленото сходство (weighted similarity) на върха V_i
- $Out(V_j)$ е множеството от наследниците на V_j
- $\mathrm{In}(V_j)$ е множеството от предшествениците на V_j
- w_{ji} е теглото от j до
- d е амортизиращ фактор (damping factor), който играе има за цел да симулира вероятността, с която се прескача от един връх на произволен друг връх, и обикновено приема стойност 0.85

3. Имплементация на алгоритъма

Първоначално извличаме тескта от подаден файл, всяка статия се намира в отделен файл на файловата система. Предварително сме ги подготвили в папка “*.articles*”. Разделяме текста на

изречения, след това преобразуваме изреченията във вид, с който можем да работим лесно. Прилагаме основни подходи за работа с естествени езици за да преработим изреченията. Преобразуваме думите до малки букви, премахваме пунктоационни знаци, цифри, стоп думи и накрая стемираме. След като сме подготвили изреченията можем да образуваме графа и да зададем тегла на ребрата му. Пресмятаме разстоянието между два върха като отношение на броя думи, които се срещат и в двете изречения и сбор от логаритмите на дължините им. Премахваме изолираните върове в графа, ако има такива. Ако няма общи думи измежду изреченията (всички ребра са с тежест 0) построяваме граф, в който всички тегла са равни на 1. Така резултатът ще бъде произволен. Това е очаквано, като се вземе предвид, че изреченията не са свързани помежду си. След това прилагаме TextRank и даваме тегла на всеки един връх. Правим 100 итерации с максимална грешка 10e-5 и амортизиращ фактор 0.85. Сортираме ги в нарастващ ред и избираме най-добрите. Дефинираме изреченията с най-висок резултат. За да сглобим резюме от най-добрите изречения имаме 2 подхода – фиксиран брой думи или процентно да намалил броят изречения. С други думи изходния ни текст може или да бъде с максимален брой думи maxWords или просто можем да вземем първите 0.2 * l изречения, където l е първоначалния брой на изреченията. Записваме резултатите в папка “.summary_articles”.

3.1 Резултати

На пръв поглед алгоритъмът се справя добре с резюмирането на всякакъв тип статии (от вестници, новини, уикипедия), но за жалост резултатите не са толкова впечатляващи що се отнася до художествена литература. При извличане на резюмета на приказки, разкази и други художествени текстове голяма пречка се оказва пряката реч. Части от текста като “— Все едно. Като сте такива магарета! — рече Лазо.” се тълкуват като 3 отделни изречения и “— рече Лазо.” е едно от тях. Заради името на главния герой то често се тълкува като важно (получава висока тежест) и алгоримът го включва в резюмето.

3.2 Подобрения

Има още интересни неща, които могат да се променят в алгоритъма. Може да се експериментира с различни стойности на параметрите или дори с различни подходи за пресмятане на разстоянието между изреченията. Най-важното и лесно подобрение на първо четене би било разширяване на списъка със стоп думи.

Друго очевидно подобрение е представянето на начин за тестване на резултатите. Събиране на корпус от подходящи за резюмиране документи заедно с “перфектното” им резюме и оценка на алгоритъма спрямо това.

4. Извличане на ключови думи

Друга интересна задача, която може да се реши отново с TextRank е извличане на ключови думи от текст. Алгоритъмът не се променя особено, като основната разлика е, че вече не даваме тежест на изреченията, а на самите думи. Важно в този случай е, че не позволяваме на всякакви

думи да бъдат ключови, а само съществителни и прилагателни. За да направим това разграничение обаче трябва да приложим друг известен подход за работа с естествения език наречен POS (Part-of-speech) tagging. Имплементацията на такъв алгоритъм сама по себе си е трудна задача и затова в този проект реших да използвам вече готов такъв. За жалост единствения, който изобщо намерих, не работеше. Без POS tagger крайните резултати станаха значително по-лоши, до такава степен, че не ги намирам за показателни.

5. Използвани технологии

- Python3 – Проектът е реализиран на езика Python
- nltk – Python библиотека за работа с естествени езици
- bulstem - Python библиотека за stemming
- bulgarian-nlp – POS Tagger

6. Използвани материали

1. An Introduction to Text Summarization using the TextRank Algorithm
2. BulStem-py
3. bulgarian-nlp
4. TextRank: Bringing Order into Texts
5. Guide to NLP’s TextRank Algorithm
6. Evaluation of LSA and TextRank Methods for Automatic Text Summarization
7. Text summarization using TextRank in NLP
8. Searching Strategies for the Bulgarian Language - stop words