HW1

Прудников Евгений

Код программы: https://github.com/EvgeniyPrudnikov/Magistracy/blob/master/IR/main.ipynb

Нормализация текста

Для нормализации текстов документов было использовано, сначала, регулярное выражение «\W+» которое чистило текст от всех символов кроме букв чисел и знака нижнего подчеркивания (_), затем убирались стоп слова с помощью библиотеки nlkt и модуля stopwords для английского языка, далее проводился стэмминг с помощью все того же nlkt, для стемминга был выбран стеммер портера.

Такая же нормализация была применена к текстам запросов.

Построение инвертированного индекса

Был построен простой инвертированный индекс, который имеет вид:

```
{
      ‘__metadata__’: {
            ‘num_of_docs’: …,
            ‘index_len’: …,
            ‘avgdl’: ...,
            ‘avg_docs_len’: …,
            ‘max_docs_len’: …
      },
      ‘term1’: [(doc_id, term_frequency, doc_lenght), (…)],
      ‘term2’: [(doc_id, term_frequency, doc_lenght), (…)],
      …
}
```

В поле metadata содержатся общие характеристики индекса, такие как длина индекса (количество термов) (index_len), количество документов (num_of_docs), средняя длина документа (avgdl), средняя длина списка словопозиций (avg_docs_len), максимальная длина списка словопозиций (max_docs_len).

Для каждого терма индекс хранит список словопозиций, который включает идентификатор документа, частота этого терма в этом документе, длина документа (кол-во термов)

Статистики индексов:

Индекс по документам:

```
{
 'avg_docs_len': 18.416861329369294,
 'avgdl': 61.946428571428569,
 'index_len': 4709,
 'max_docs_len': 730,
 'num_of_docs': 1400
}
```

Индекс по заголовкам:


```
{
 'avg_docs_len': 8.5413819286256647,
 'avgdl': 8.0350000000000001,
 'index_len': 1317,
 'max_docs_len': 365,
 'num_of_docs': 1400
}
```


Алгоритм поиска

Был реализован алгоритм поиска, основанный на BM25. Для каждого найденного документа для каждого терма из запроса подсчитывалась RSV по формуле:

$$RSV(q,d) = \sum_{t \in q} \left(\log \left(1 + \frac{N - N_t + 0.5}{N_t + 0.5} \right) \times \frac{f_{t,d}(k_1 + 1)}{k_1((1-b) + b \cdot L_d/\bar{L}) + f_{t,d}} \right)$$

С параметрами *k1* = 1.2, *b* = 0.75 (так же я пробовал другие комбинации параметров, однако, на результаты поиска это почему-то не повлияло)
Далее, результаты сортировались по убыванию RSV и top-10 записывался в файл.

Результаты

Были получены следующие результаты (eval.py):

```
search by docs:
mean precision: 0.22024691358
mean recall: 0.306024492383
mean F-measure: 0.256145210109
MAP@10: 0.21955973307


search by titles:
mean precision: 0.173333333333
mean recall: 0.246569763465
mean F-measure: 0.203564866877
MAP@10: 0.170676459926
```

Видно, что поиск по заголовкам документов показывает чуть худшие результаты по сравнению с поиском по полному документу.

Так же я пробовал добавлять к формуле RSV множитель

$$\frac{(k_2 + 1) f_{t,q}}{k_2 + f_{t,q}}$$

И подбирать параметр *k2,* однако, это никак не улучшило результаты, а при значениях параметра 10, 100, 1000 даже значительно ухудшило.