Отчет по заданию “Inv Index”

Кукуев Михаил, АИСОБОИ-2

Для запуска использовать файл hw1.py, вызвать справку можно параметром -h.

Примеры:

python hw1.py index cran.all.1400
python hw1.py search cran.qry answer -k1 1.81 -b 0.78 -k2 100

1-5. Парсинг файлов осуществляется методами load_data и load_queries, возвращающими список объектов документов и запросов. За построение обратного индекса отвечает метод create_inverted_index, он возвращает объект класса InvertedIndexData, который хранит в одном поле словарь документов с их длинами, а в другом словарь терминов, в котором каждому термину ставится в соответствие словарь документов с частотой данного термина.

При нормализации слова приводились к нижнему регстру, затем применялся стеммер, он показал немного лучший результат по сравнению с лемматизатором; после исключались стоп-слова и знаки пунктуации, и потом с концов терма удалялись цифры и символ ‘/’, чтобы убрать слова с цифрами вначале и исправить слова вида ‘/word’. Также удалялось слово нулевой длины, т.к. немного ухудшало метрики качества.

Реализация поиска по запросу на основе BM25 находится в функции bm25_search: строится множество релевантных запросу документов как объединение множеств документов, релевантных каждому терму запроса, затем для каждого терма из запроса вычисляются tf и df, которые потом используются при расчете RSV. Построенный индекс сохраняется в бинарный файл путем сериализации объекта класса InvertedIndexData.

Статистика по построенному индексу и значения метрик качества поиска:

	Поиск по заголовкам	Поиск по аннотации
Размер словаря	1427	5940
Средняя длина документа	7	89
Средняя длина списка словопозиций	7.4456	14.114
Наибольшая длина списка словопозиций	359	714
mean precision	0.2462	0.2929
mean recall	0.3613	0.4256
mean F-measure	0.2929	0.3470
MAP@10	0.2859	0.3642

Метрики качества для поиска по аннотациям, как и ожидалось, оказались лучше, вероятнее всего потому что слова из запроса чаще присутствовали в тексте аннотации релевантного документа и отсутствовали в его заголовке. Также навряд ли слово из релевантного документа будет встречаться несколько раз в заголовке в отличии от текста аннотации, что напрямую влияет на величину RSV.

6. Для подбора параметров k1 и b был написан дополнительный скрипт grid_search.py, выполняющий вызов hw1.py в режиме поиска, передавая параметры k1 и b. Поиск оптимальных параметров выполнялся по сетке возможных значений k1 и b с шагом 0.01. Затем выводились первые 10 значений коэффициентов k1 и b с наибольшим значением функции MAP@10 из eval.py.

Топ-10 значений коэффициентов:

k1=1.81, b=0.78, MAP@10=0.375094
k1=1.80, b=0.78, MAP@10=0.374946
k1=1.77, b=0.79, MAP@10=0.374931
k1=1.73, b=0.79, MAP@10=0.374925
k1=1.83, b=0.78, MAP@10=0.374886
k1=1.75, b=0.78, MAP@10=0.374862
k1=1.74, b=0.78, MAP@10=0.374847
k1=1.70, b=0.79, MAP@10=0.374845
k1=1.71, b=0.80, MAP@10=0.374807
k1=1.71, b=0.79, MAP@10=0.374793

Метрики качества поиска для k1=1.81 b=0.78:

mean precision: 0.302666666667
mean recall: 0.436063405533
mean F-measure: 0.357320927832
MAP@10: 0.375093807564

Коэффициенты оказались довольно близки к рекомендуемым на Википедии (https://ru.wikipedia.org/wiki/Okapi_BM25#Функция_ранжирования) k1=2, b=0.75, возможно потому что длина запросов из выборки, аннотаций документов, и их отношение (длина документа на порядок больше длины запроса) близки к значениям, которые чаще всего встречаются на практике.

Для дальнейших исследований использовались коэффициенты k1=1.81, b=0.78.

7. В формуле RSV изменим вычисление IDF, учитывая количество релевантных запросу документов S и количество релевантных, содержащих терм t – St, как на слайде 50 лекции. Так как для нашей булевой формы запроса поиска St=Nt, то формула будет следующей:

$$IDF = \log \frac{(Nt + 0.5) * (N - S + 0.5)}{0.5 * (S - Nt + 0.5)}$$

Получились следующие значения метрик качества:

mean precision: 0.223111111111
mean recall: 0.326737659769
mean F-measure: 0.265159462651
MAP@10: 0.247851858851

Метрики сильно ухудшились, т.к. видимо такой подход скорее применяется для булевых запросов вида “term1 AND term2 ... AND termK”.

8. После нормирования RSV(q,d) на сумму IDF термов запроса значения ни одной из метрик не изменились.

9. В grid_search.py находится функция подбора значений коэффициента k2, реализованная аналогично подбору k1 и b с максимизацией функции MAP@10.

Топ-10 значений коэффициента k2:

Шаг 100	Шаг 10	Шаг 1
k2=0.000000, MAP@10=0.375094 k2=100.000000, MAP@10=0.362255 k2=200.000000, MAP@10=0.362255 k2=300.000000, MAP@10=0.362206 k2=400.000000, MAP@10=0.362206 k2=500.000000, MAP@10=0.361924 k2=600.000000, MAP@10=0.361924 k2=700.000000, MAP@10=0.361924 k2=800.000000, MAP@10=0.361924 k2=900.000000, MAP@10=0.361924	k2=0.000000, MAP@10=0.375094 k2=10.000000, MAP@10=0.364526 k2=20.000000, MAP@10=0.362998 k2=30.000000, MAP@10=0.362636 k2=40.000000, MAP@10=0.362403 k2=50.000000, MAP@10=0.362266 k2=90.000000, MAP@10=0.362255 k2=60.000000, MAP@10=0.362243 k2=70.000000, MAP@10=0.362243 k2=80.000000, MAP@10=0.362206	k2=0.000000, MAP@10=0.375094 k2=1.000000, MAP@10=0.371098 k2=2.000000, MAP@10=0.368816 k2=3.000000, MAP@10=0.367900 k2=4.000000, MAP@10=0.366435 k2=5.000000, MAP@10=0.365416 k2=6.000000, MAP@10=0.364738 k2=8.000000, MAP@10=0.364533 k2=7.000000, MAP@10=0.364510 k2=9.000000, MAP@10=0.364501

Наилучшим оказалось значение по умолчанию k2=0, скорее всего потому что в запросах слова редко повторяются более одного раза.

10. Да, можно. При увеличении порога precision будет расти, а recall уменьшаться, т.к. в результаты уже не будут попадать нерелевантные документы, у которых RSV меньше порогового значения, но также не попадут и релевантные документы, у которых RSV по каким-то причинам (недостаточное качество поиска, использование простого алгоритма) не превзошло порог. Поэтому стоит выбирать порог в зависимости от важности значения той или иной метрики в конкретном случае, например можно максимизировать F-меру, в которой точность и полнота учтены с разными весами.

Значения метрик при разных пороговых значениях RSV:

Метрика \ Пороговое значение RSV	10	15	20	30
Mean precision	0.3300	0.4807	0.6920	0.8620
Mean recall	0.4292	0.4007	0.3432	0.2701
Mean F-measure	0.3731	0.4370	0.4589	0.4113
MAP@10	0.3946	0.5192	0.7016	0.8568

Итог:

Наилучшие параметры: k1=1.81, b=0.78, k2=0.

Метрики:

mean precision: 0.302666666667
mean recall: 0.436063405533
mean F-measure: 0.357320927832
MAP@10: 0.375093807564