گزارش تمرین اول درس بازیابی اطلاعات

مینا فریدی، شماره دانشجویی: ۸۱۰۱۰۰۴۳۰

آماده سازی محیط:

ابتدا سیستم عامل ubuntu را نصب کردم و سپس در آن به ترتیب جاوا، maven و گالاگو را نصب نمودم و فایل متن ها را که به صورت زیپ هست باید از زیپ در بیاوریم تا به فرمت trectext در بیاید.

ایجاد شاخص:

برای ایجاد شاخص به ترتیب مراحل توکن کردن، نرمال کردن کلمات، حذف حروف اضافه و ریشه یابی انجام داده شد.

نمونه انجام مراحل:

- According to Wikipedia, Information Retrieval is the activity of obtaining information resources relevant to an information need from a collection of information resources.

- **Tokenization**,
  According, to, Wikipedia, Information, Retrieval, is, the, activity, of, obtaining, information, resources, relevant, to, an, information, need, from, a, collection, of, information, resources.
- **Normalization**,
  according, to, wikipedia, information, retrieval, is, the, activity, of, obtaining, information, resources, relevant, to, an, information, need, from, a, collection, of, information, resources.
- **Stopping**,
  according, -, wikipedia, information, retrieval, -, -, activity, -, obtaining, information, resources, relevant,-, -, information, need, -, -, collection, -, information, resources.

  "to", "is", "the", "of", "an", "a" are treated as stop words. Because they are frequent and uninformative.
- **Stemming**,
  accord, -, wikipedia, inform, retrieve, -, -, act, -, obtain, inform, resource, relate,-, -, inform, need, -, -, collect, -, inform, resource.

برای توکن کردن متن و تبدیل آن به توکن‌های سازنده اش از این تنظیمات استفاده می‌کنیم:

```
"tokenizer" : {

"fields" : ["text","head"],

"formats" : {
    "text"    : "string",
    "head"    : "string"
  }
},
```

الگوریتم های مختلفی برای ریشه یابی وجود دارد که یکی از رایج ترین آن ها porter است که در واقع از آخر کلمات قسمت های پر تکرار مثل ing را حذف می‌کند و در کل عملکرد مناسبی دارد. تنظیم کردن آن به این صورت انجام شد:

```
"stemmer" : ["porter"],
```

برای ایجاد شاخص تنظیمات ذکر شده را ذخیره کرده و با دستور زیر اجرا میکنیم:

Galago/galago-3.16/core/target/appassembler/bin/galago build /home/mina/Desktop/CA1-Resources/indexResult.json

```
{

  "server" : true,

  "indexPath" : "indexDir",
  "inputPath" : "./CA1-Resources/Corpus/corpus/Documents.trectext",

    "stemmer" : ["porter"],

    "tokenizer" : {

    "fields" : ["text","head"],

    "formats" : {
        "text"    : "string",
        "head"    : "string"
      }
    },
    "fileType":"trectext"
}
```

سوال اول: روش بازیابی BM25

ابتدا پرس و جو را روی کوئری های ۱۰۱ تا ۱۵۰ یا روش اصلی BM25 انجام می دهیم. و مقدار نتایج خروجی برای هر کوئری را ۱۰۰ قرار میدهیم (Requested = 100).

جدول زیر نتایج را برای معیارهای MAP، nDCG، Recall، P5 را در کوئری های ۱۰۱ تا ۱۵۰ با پارامترهای b,k مختلف نشان میدهد. بر اساس این جدول، مشاهده می‌کنیم که در b=0.4, k=2.5 حالت بهینه رخ داده است.

k	b	MAP	nDCG	Recall	P@5
1.2	0.75	0.168	0.331	0.899	0.368
1.2	0.50	0.171	0.337	0.843	0.400
1.2	0.60	0.167	0.334	0.870	0.384
1.2	0.50	0.171	0.337	0.843	0.400
1.6	0.50	0.174	0.343	0.833	0.412
1.9	0.50	0.178	0.347	0.850	0.408
2.5	0.50	0.179	0.349	0.831	0.420
2.5	0.60	0.177	0.346	0.865	0.400
2.5	0.30	0.181	0.346	0.848	0.408
2.5	0.45	0.180	0.348	0.829	0.420
2.5	0.40	0.182	0.349	0.831	0.420
2.0	0.45	0.180	0.345	0.880	0.392

در قسمت بعد برای کوئری‌های ۵۱ تا ۱۰۰ عمل پرس و جو را انجام می‌دهیم. یک بار به پارامترهای b, k مقدار دیفالت و یک بار مقدار بهینه قسمت قبل را می‌دهیم و مشاهده میکنیم که با بکار بردن پارامترهای بهینه نتیجه بهتر می شود. زیرا مقدار بهینه‌ی b , k به خود داکیومنت ربط دارند و چون داکیومنت ها ثابت هستند با کوئری های متفاوت بازم مقادیر بهینه نباید خیلی فرق بکنند و بهینه سازی در این حالت میتواند موئثر واقع شود.

k	b	MAP	nDCG	Recall	P@5
0.75	1.2	0.382	0.322	0.170	0.220
0.40	2.5	0.412	0.331	0.179	0.228

در قسمت بعدی نتایج پرس‌وجوها را با توابع ارزیابی مختلف بررسی می‌کنیم. مقادیر به دست آمده از هر یک از روش های ذکر شده در جدول زیر آمده است. مطابق جدول روش پیشنهادی اول به دلیل این که تنها از IDF استفاده کرده از معیارهای کارایی مطلوبی برخوردار نیست، همچنین از این جدول نتیجه می‌گیریم که مدل های ۴ و ۵ بر اساس معیارهای ارزیابی هم نتایج کلی بهتری داشتند و هم با به کار بردن پارامترهای b,k بهینه ی بدست آمده در قسمت های قبل بهبود بیشتری از خود نشان داده اند. همان‌طور که در جدول زیر مشاهده می‌کنید به دلیل این که در روش های یک و سه از پارامتر های b و k استفاده نشده است این دو روش نسبت به تغییرات این پارامترها حساس نبوده لذا بهینه سازی ای هم ممکن نیست.

	k	b	MAP	nDCG	Recall	P@5
suggested method 1	1.2	0.75	0.008	0.021	0.017	0.022
	2.5	0.40	0.008	0.021	0.017	0.022
	2.0	0.45	0.008	0.021	0.017	0.022
suggested method 2	1.2	0.75	0.147	0.287	0.201	0.402
	2.5	0.40	0.142	0.280	0.195	0.394
	2.0	0.45	0.144	0.284	0.200	0.400
suggested method 3	1.2	0.75	0.073	0.181	0.126	0.222
	2.5	0.40	0.073	0.181	0.126	0.222
	2.0	0.45	0.073	0.181	0.126	0.222
suggested method 4 BM25L	1.2	0.75	0.171	0.324	0.220	0.396
	2.5	0.40	0.178	0.330	0.227	0.416
	2.0	0.45	0.179	0.331	0.227	0.412
suggested method 5 BM25+	1.2	0.75	0.170	0.322	0.220	0.382
	2.5	0.40	0.179	0.331	0.228	0.412
	2.0	0.45	0.187	0.330	0.227	0.412

سوال دوم: روش بازیابی Pivoted Length Normalization

در این قسمت روش پیشنهادی پنجم و مدل اصلی به همراه مدل پایه ای BM25 و نیز مدل بدون مولفه لگاریتمی تودرتو با هم به مقایسه گذاشته شده اند. همانطور که مشاهده میشود بر روی مقدار های پیشفرض پارامترهای k و b مدل BM25+ توانسته است بهترین نتایج را داشته باشد.

MAP	nDCG	Recall	P@5	
0.125	0.257	0.198	0.194	مدل اصلی
0.112	0.236	0.195	0.188	مدل بدون مولفه لگاریتمی تو در تو
0.168	0.331	0.899	0.368	BM25L
0.170	0.322	0.220	0.382	suggested method 5 BM25+

*) به پیوست و درکنار گزارش فایل های مربوطه قابل دسترسی هستند.