گزارش انجام فاز اول پروژه درس بازیابی پیشرفته اطلاعات

اعضای تیم:

سیدعلیرضا حسینی 96105701

سجاد شهابی 96105875

مهدی جوانمردی 96105683

پاییز ۹۹

ساختار کلی پروژه:

پروژه تشکیل شده از چندین مولفه است:

Parser: وظیفه خواندن مجموعه‌های داده و پارس کردن آن را دارد.

Engine: این package شامل موتور جستجو است و الگوریتم‌های جستجو در این package تعریف شده اند.

Normalizer: این package دارای کلاس‌هایی است که برای نرمال سازی متون مورد استفاده قرار می‌گیرد.

Index: این package شامل کلاس‌هایی برای index کردن و ذخیره index بر روی فایل می باشد.

Correction: این package شامل کلاس‌هایی برای تشخیص خطا و تصحیح متون می‌باشد.

User-interface: رابط کاربری تحت کنسول برنامه در اینجا پیاده شده است.

برای اجرای پروژه ابتدا لازم است pre-execution.py اجرا شود و سپس main.py در پوشه src اجرا شود. داده ها نیز باید در پوشه /data در مسیر root پروژه قرار بگیرند.

گزارش انجام قسمت اول (نرمال سازی):

برای متون انگلیسی:

از ابزار nltk برای نرمال سازی متون استفاده شده است. ابتدا توکن‌های متن استخراج می‌شوند و سپس حروف زائد حذف می‌شوند و در ادامه lemmatization و stemming بر روی کلمات اعمال می‌شود و در آخر حروف تکراری یافته و حذف می‌شوند.

نمونه‌ای از اجرای برنامه:

```
Write 1 for english or 2 for persian:
1
Write your query:
That said, there's no reason to reformat the corpus. You can just have your code make these
Normalized text:
that,said,there,no,reason,to,reformat,the,corpus,you,can,just,have,your,code,make,these
Stop words:

Part 1:
1- Show the Normalized Form of a Query and stop words:
2- Exit
1
Write 1 for english or 2 for persian:
1
Write your query:
The decision about whether to lowercase everything is really dependent of what you plan to do. For some purposes, lowercasing everything is
Normalized text:
decis,about,whether,to,lowercas,everyth,is,realli,depend,of,what,you,plan,to,do,for,some,purpos,lowercas,everyth,is,better,becaus,it,lower,
Stop words:
the
```

برای متون فارسی:

دقیقا همان مراحل بالا ولی با استفاده از کتابخانه هضم اعمال می‌شوند.

نمونه‌ای از اجرای برنامه:

```
Part 1:
1- Show the Normalized Form of a Query and stop words:
2- Exit
1
Write 1 for english or 2 for persian:
2
Write your query:
ش به اندازه‌ی فاصله داده شده باشد. بنابراین ابتدا تمامی اسنادی را که بین کلمات کوئری مشترک هستند را می‌گیریم و سپس از میان آنها اسنادی که شرط فاصله‌ی کلمات را داشته‌باشند بدست می‌آوریم؛
Normalized text:
میشود,تمام,کوئر,در,آن,سند,بود#باش,و,حداکثر,هر,کلمه,از,کلمه,بعد,به,اندازه,دادهشدهباشد,بنابراین,ابتدا,تمام,اسناد,بین,کوئر,مشترک,#هست,میگیر,و,سپس,از,م,آن,اسناد,شرط,داشتهباشند,بدس,میآور
Stop words:
فاصله,که,را,کل
```

گزارش انجام قسمت دوم (نمایه‌سازی):

برای ایجاد نمایه‌ها از سه کلاس Indexer، WordIndex، DocumentIndex استفاده شده‌است. که توضیح هر کلا می‌پردازیم.

کلاس DocumentIndex: این کلاس وظیفه‌ی نگه‌داری نمایه‌های مکانی مروبط به یک کلمه در یک مستند را دارد.

کلاس WordIndex: این کلاس وظیفه‌ی نگه‌داری لیستی از DoumentIndex های مربوط به یک کلمه را دارد.

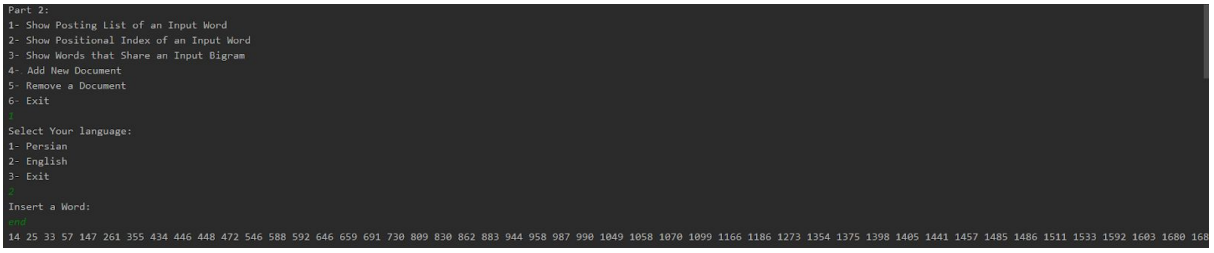
کلاس Indexer: این کلاس وظیفه‌ی نگه‌داری دیکشنری برای نگه‌داری WordIndexهای مربوط به کلمات مختلف را دارد علاوه بر این bigramها نیز در این کلاس ذخیره می‌شوند. کلمات نرمال شده به عنوان ورودی به این کلاس داده می‌شوند و بر اساس این کلمات اشیا مرتبط با آن کلمه ساخته می‌شوند.

نمونه‌ی اجرای بزنامه:

نمایش posting list یک کلمه:

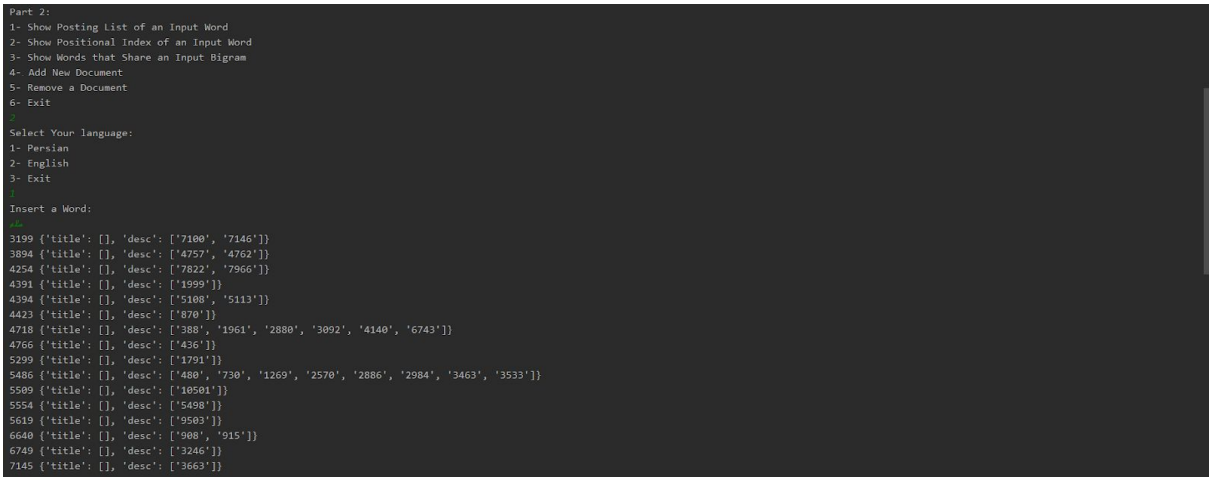
برای متون فارسی:

برای متون انگلیسی:



نمایش positional index یک کلمه:

برای متون فارسی:



برای متون انگلیسی:

```
Part 2:
1- Show Posting List of an Input Word
2- Show Positional Index of an Input Word
3- Show Words that Share an Input Bigram
4- Add New Document
5- Remove a Document
6- Exit
2
Select Your language:
1- Persian
2- English
3- Exit
2
Insert a Word:
end
14 {'title': [], 'desc': ['17']}
25 {'title': [], 'desc': ['33']}
33 {'title': ['3'], 'desc': []}
57 {'title': ['7'], 'desc': []}
147 {'title': ['5'], 'desc': ['5']}
261 {'title': [], 'desc': ['25']}
355 {'title': [], 'desc': ['30']}
434 {'title': [], 'desc': ['46']}
446 {'title': [], 'desc': ['6']}
448 {'title': [], 'desc': ['36']}
472 {'title': [], 'desc': ['34']}
546 {'title': [], 'desc': ['24']}
588 {'title': [], 'desc': ['29']}
592 {'title': [], 'desc': ['32']}
646 {'title': ['3'], 'desc': []}
659 {'title': [], 'desc': ['31']}
691 {'title': ['3'], 'desc': []}
730 {'title': [], 'desc': ['10']}
809 {'title': ['3'], 'desc': []}
830 {'title': [], 'desc': ['32']}
```

(که این بخشی از خروجی مربوط به این بخش است.)

نمایش کلماتی که شامل یک bigram هستند:

برای متون فارسی:

```
Part 2:
1- Show Posting List of an Input Word
2- Show Positional Index of an Input Word
3- Show Words that Share an Input Bigram
4- Add New Document
5- Remove a Document
6- Exit
3
Select Your language:
1- Persian
2- English
3- Exit
1
Insert a Bigram:
$ا
[illegible]
```

برای متون انگلیسی:

```
Part 2:
1- Show Posting List of an Input Word
2- Show Positional Index of an Input Word
3- Show Words that Share an Input Bigram
4- Add New Document
5- Remove a Document
6- Exit
3
Select Your language:
1- Persian
2- English
3- Exit
2
Insert a Bigram:
$a
['a', 'aakash', 'aala', 'aamodt', 'aaron', 'aaronson', 'ababa', 'abalon', 'abandon', 'abani', 'abdelmagi', 'abdi', 'abduct', 'abdul', 'abe', 'abha', 'abhor', 'abigail', 'abil', 'abl', 'aboard', 'abod', '
```

اضافه کردن یک مستند به مجموعه‌ی مستندها:

برای متون فارسی:

```
Part 2:
1- Show Posting List of an Input Word
2- Show Positional Index of an Input Word
3- Show Words that Share an Input Bigram
4- Add New Document
5- Remove a Document
6- Exit
4
Select Your language:
1- Persian
2- English
3- Exit
1
Insert Your New Document Location
../data/new_per.xml
```

که مستندی که آدرس آن در دستور بالا استفاده شده است، به شکل زیر است:

```
<mediawiki xmlns="http://www.mediawiki.org/xml/export-0.10/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocati
  <page>
    <title>سلام </title>
    <ns>0</ns>
    <id>1</id>
    <revision>
      <id>12439263</id>
      <parentid>10035767</parentid>
      <timestamp>2014-04-18T11:36:47Z</timestamp>
      <contributor>
        <username>Fatemibot</username>
        <id>355566</id>
      </contributor>
      <minor />
      <comment></comment>
      <model>wikitext</model>
      <format>text/x-wiki</format>
      <text xml:space="preserve">{{Coord|49.2611|-123.2614|type:landmark|display=title}}
{{بدون منبع}}
سلام
      </text>
      <sha1>oyvayrctay1y9br0yfmbdnitjznnbh1</sha1>
    </revision>
  </page>
</mediawiki>
```

همانطور که مشاهده می‌کنید مستند 1 به posting list کلمه‌ی "سلام" اضافه شده‌است.

```
Part 2:
1- Show Posting List of an Input Word
2- Show Positional Index of an Input Word
3- Show Words that Share an Input Bigram
4- Add New Document
5- Remove a Document
6- Exit
1
Select Your language:
1- Persian
2- English
3- Exit
1
Insert a Word:
سلام
1 3199 3894 4254 4391 4394 4423 4718 4766 5299 5486 5509 5554 5619 6640 6749 7145
```

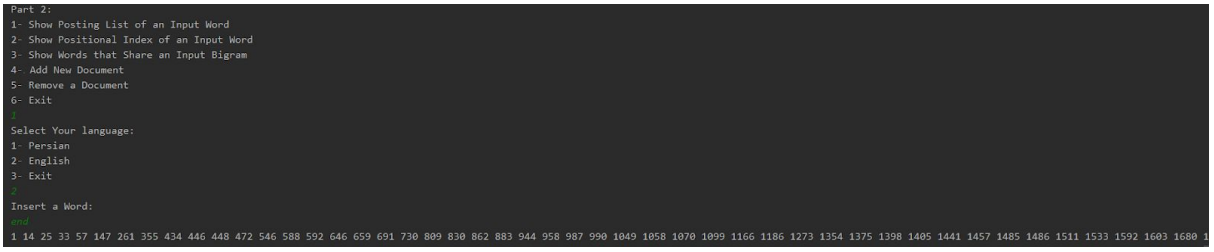
برای متون انگلیسی:

```
Part 2:
1- Show Posting List of an Input Word
2- Show Positional Index of an Input Word
3- Show Words that Share an Input Bigram
4- Add New Document
5- Remove a Document
6- Exit
4
Select Your language:
1- Persian
2- English
3- Exit
2
Insert Your New Document Location
../data/new_eng.csv
```

که مستندی که آدرس آن در دستور بالا استفاده شده است، به شکل زیر است:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	comment	descriptio	duration	event	film_date	languages	main_spe	name	num_spea	published	ratings	related_t	speaker_c	tags	title	url	views
2	end	end	end	end	end	end	end	end	end	end	end	end	end	end	end	end	end
3																	
4																	
5																	
6																	

همانطور که مشاهده می‌کنید مستند 1 به posting list کلمه‌ی "end" اضافه شده‌است.



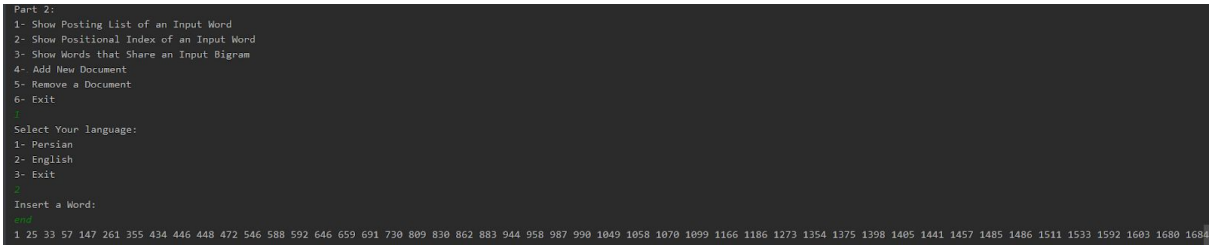
حذف یک مستند:

برای متون فارسی:

همانطور که مشاهده می‌کنید مستند 3199 از posting list کلمه‌ی "end" حذف شده‌است.

برای متون انگلیسی:

همانطور که مشاهده می‌کنید مستند 14 به posting list کلمه‌ی "end"حذف شده‌است.



گزارش انجام قسمت سوم (فشرده سازی):

در این بخش، دو نوع الگوریتم فشرده‌سازی پیاده‌سازی شده‌اند که الگوریتم‌های variable byte و gamma می‌باشند و برای هر کدام encoding خاص خودشان را در کلاس‌های مخصوص زده شده‌اند و چون از الگوریتم gamma برای ذخیره‌سازی استفاده کرده‌ایم دیکود را برای gamma زده‌ایم؛ از طرفی این فشرده‌سازی را از این جهت به indexer متصل کرده‌ایم که برای ذخیره‌ی posting list و position list این لیست که از Document Index تشکیل شده است استفاده کردیم و برای ذخیره‌سازی آنها در فایل به جای استفاده از اعداد از gamma encoding استفاده کرده‌ایم که در غیر این صورت فایل ذخیره‌سازی حجم بسیار بیشتری به خود می‌گرفت.

```
Please Select the Part of Project You Want To Test:
1- Part 1 Pre_Processing
2- Part 2 Indexing
3- Part 3 Compressing
4- Part 4 Spell Correction
5- Part 5 Search & Retrieval
6- Exit
3
Part 3:
1- Show The Saved Memory By Using Variable Byte Encoding
2- Show The Saved Memory By Using Gamma Code Encoding
3- Exit
1
english, before:  2493398 after: 2241793.0 bytes
persian, before:  22845663 after: 17394359.0 bytes
Part 3:
1- Show The Saved Memory By Using Variable Byte Encoding
2- Show The Saved Memory By Using Gamma Code Encoding
3- Exit
2
english, before:  2493398 after: 2239752.25 bytes
persian, before:  22845663 after: 18011132.5 bytes
```

حال برای اینکه ذخیره‌سازی را انجام دهیم position index ها را فشرده می‌کنیم و با حالت فشرده ذخیره می‌کنیم در حالت معمول میزان حجم دیتاهای فارسی 22,845,663 بایت می‌باشد که در ذخیره‌سازی با استفاده از فشرده‌سازی گاما 41,437,408 بایت می‌شود و در حالت معمول دیتاهای انگلیسی 2,493,398 بایت و حالت ذخیره شده 3,242,691 بایت می‌باشد؛ این از آن جهت است که اعداد بایتی را به صورت string در فایل ذخیره کرده‌ایم. در صورتی که میزان ذخیره‌سازی اصلی اگر بایت‌ها را با اعداد position index ذخیره شده مقایسه کنیم ، در فشرده‌سازی variable byte برای فارسی 43,610,432 بیت و برای انگلیسی 2,012,840 بیت حجم کم می‌شود که به ترتیب حجم فایل‌ها 17,394,359 بایت و 2,241793 بایت می‌شوند و در فشرده‌سازی گاما برای فارسی 38,676,244 بیت برای انگلیسی 2,029,166 بیت حجم کم می‌شود که به ترتیب حجم فایل‌ها 18,011,132 بایت و 1,759,194 بایت می‌شوند.

گزارش انجام قسمت چهارم (اصلاح پرسمان):

در این بخش، ابتدا کلمات درخواست وارد شده از کاربر را نرمال‌سازی می‌کنیم، سپس با توجه به معیار جاکارد 10تا از نزدیک‌ترین کلمات به هر یک از کلمات درخواست را مشخص کرده و از بین این کلمات آن کلمه‌ای که کمترین میزان edit distance را دارد به عنوان کلمه‌ی جایگزین انتخاب می‌کنیم. در صورتی که کلمه بعد از اعمال اصلاح تغییر نکرده باشد کلمه‌ی آمده در متن را تغییر نمی‌دهیم و در صورتی که کلمه دچار تغییر شده باشد حالت تغییر داده شده‌ی آن را جایگزین می‌کنیم.

نمایش اصلاح شده‌ی یک درخواست:

برای متون فارسی:

برای متون انگلیسی:

محاسبه‌ی فاصله‌ی جاکارد بین دو کلمه:

برای متون فارسی:

برای متون انگلیسی:

محاسبه‌ی edit distance بین دو کلمه:

برای متون فارسی:

برای متون انگلیسی:

گزارش انجام قسمت پنجم (جستجو):

برای جستجو به روش TF-IDF:

در این قسمت ابتدا از indexer برای یافتن تمامی داکیومنت‌های شامل عبارات درخواست استفاده می‌شود. پس از یافتن لیست مستندات، برای هر مستند یک وکتور بر اساس کلمات normalized شده عبارت درخواست ساخته می‌شود. سپس با محاسبه tf-idf برای درخواست و کوئری به روش lnc-ltc، بردارهای مستندات در بردار کوئری ضرب شده و سپس مستندات بر اساس حاصل این ضرب داخلی مرتب می‌شوند و خروجی داده می‌شوند.

نمونه‌ای از اجرای کوئری به روش TF-IDF:

```
5
Part 5:
1- Show Result of an Input Query Using tf-idf
2- Show Result of an Input Query Using Proximity Search
3- Exit
1
Write 1 for english or 2 for persian:
2
Write 1 for title or 2 for description or 3 for both:
3
write your query:
من کتاب را دوست دارم
your top 10 result is:
4718,3901,5554,5381,5553,6768,6375,4838,6749,7008
```

بعد از بررسی انجام شده مشخص شد که اکثر این مستندات دارای کلمات دوست، کتاب و داشتن بودند.

برای جستجو به روش Proximity:

برای این قسمت یک کوئری و میزان فاصله‌ای که مدنظر است گرفته می‌شود؛ سپس با استفاده از نرمال‌ساز مربوط به آن زبان کوئری ورودی را نرمال کرده سپس به تابعی می‌دهیم تا اسناد مربوط را بدهد. سند مربوط در این قسمت اینگونه تعریف می‌شود که تمامی کلمات کوئری در آن سند باشد و حداکثر فاصله‌ی هر کلمه از کلمه‌ی بعدش به اندازه‌ی فاصله داده شده باشد. بنابراین ابتدا تمامی اسنادی را که بین کلمات کوئری مشترک هستند را می‌گیریم و سپس از میان آنها اسنادی که شرط فاصله‌ی کلمات را داشته‌باشند بدست می‌آوریم؛ بعد از اینکه این اسناد را بدست آوردیم، آنها را امتیازدهی می‌کنیم؛ این امتیازدهی به شکل امتیازدهی قسمت TF-IDF می‌باشد. بعد از اینکه امتیازدهی صورت گرفت و رتبه هر سند معلوم شد، ۱۰ نتیجه‌ی بالا را خروجی می‌دهیم.

نمونه‌هایی از اجرای کوئری به روش Proximity:

```
Welcome to Our Search Engine,
Please Select the Part of Project You Want To Test:
1- Part 1 Pre_Processing
2- Part 2 Indexing
3- Part 3 Compressing
4- Part 4 Spell Correction
5- Part 5 Search & Retrieval
6- Exit
5
Part 5:
1- Show Result of an Input Query Using tf-idf
2- Show Result of an Input Query Using Proximity Search
3- Exit
2
Write 1 for english or 2 for persian:
2
Write 1 for title or 2 for description or 3 for both:
3
write your query:
دهستان شرقی
Enter your distance:
3
your top 10 result is:
3039,6459,5731,5354,5380,6036,3665,4339,3016
```

```
Part 5:
1- Show Result of an Input Query Using tf-idf
2- Show Result of an Input Query Using Proximity Search
3- Exit
2
Write 1 for english or 2 for persian:
1
Write 1 for title or 2 for description or 3 for both:
2
write your query:
david takes offenders
Enter your distance:
10
your top 10 result is:
3
```