فاز سوم(۲۰+۱۰۰):

سلام، این فاز از پروژه دو بخش جدا از هم رو پوشش می‌دهد. در بخش‌های ۱ تا ۵ از شما خواسته شده تا:

1. با پیاده‌سازی یک خزنده اطلاعات مقالات را از سایت researchgate.net جمع آوری کنید.
2. از ElasticSearch استفاده کنید و اطلاعاتی را که جمع‌آوری کردید بوسیله‌ی آن شاخص‌گذاری کنید.
3. Page rank را برای مقالاتی که بدست آوردید محاسبه کرده و آن را هم به ElasticSearch اضافه کنید.
4. با استفاده از امکانات ElasticSearch پرس‌وجوهای وزن‌دار را بر روی داده‌ها اجرا کنید.
5. با پیاده‌سازی الگوریتم HITS نویسندگان برتر را پیدا کنید.

همان‌طور که می‌بینید این ۵ بخش به هم پیوسته هستند.

در نهایت از شما می‌خواهیم که با یک مسئله‌ی learning to rank را بوسیله‌ی Ranking SVM حل کنید. برای این کار از مجموعه‌ی داده‌ی MQ2008 استفاده خواهیم کرد.

بخش اول: پیاده سازی خزنده و واکشی اطلاعات(۲+۴۰)

در این بخش هدف استخراج اطلاعات مورد نیاز از قالب HTML سایت researchgate.net برای مقالات و همچینین یافتن مقلات دیگر از طریق ارجاعات بین آنها و استخراج اطلاعات از آن مقالات و ادامه همین روند خزش است. برای این منظور از کتابخانه Scrapy استفاده کنید.

اطلاعات مورد نیاز برای برای ذخیره در موتور جستجو و بازیابی آنها برای هر مقاله به این شرح است:

1. چکیده مقاله (در صورت وجود ، برخی مقالات دارای چیکده نمی باشند ، همچنین برخی مقالات دارای قسمت مشترک چکیده و تصاویر مقاله می باشد که باید تنها چکیده استخراج گردد)
2. عنوان مقاله
3. سال انتشار

4. نام نویسنده (نویسندگان)
5. ارجاعات مقاله (از بین مقالاتی که در این بخش است تعداد 10 تا اولین را انتخاب کرده و سپس از طریق این 10 لینک عملیات استخراج سایر مقالات را انجام می دهید)

برای اینکه به اطلاعات بیان شده در هر لینک دسترسی پیدا کنیم باید تگ های HTML را به صورت بهینه کاوش کنیم ، به تگ حاوی اطلاعات رسیده (مثلا نام نویسنده) و سپس محتوای تگ (مثلا یک متن به عنوان نام مقاله یا لینک برای ارجاعات) را به دست آورده و خود تگ حذف گردد، ابزار مناسب برای این منظور CSS Selector است که باید موارد مورد نیاز برای این منظور را مطالعه بفرمایید.

برای شروع عملیات خزش نیاز به وجود چند لینک اولیه شروع می باشد که به عنوان start_urls در اختیار کلاس Spider کتابخانه Scrapy قرار می گیرد، برای این منظور از تعداد 3 مقاله اولیه که در ادامه لینک آنها قرار گرفته عملیات خزش را شروع می کنید و تا رسیدن به 2000 مقاله این فرایند را ادامه می دهید (این مقدار به عنوان ورودی به تابع داده میشود)، به 3 نکته در این قسمت توجه بفرمایید:

1. توجه کنید که مقالات (لینک ها) باید متمایز باشد و اگر در فرایند یافتن 2000 مقاله به مقالات تکراری برخورد کردید آنها را نادیده گرفته و ذخیره نکنید.
2. در این قسمت هدف تنها یافتن لینک مقالات به تعداد بیان شده است، پس از به دست آوردن لینک مقالات، لینک مقالات اولیه بیان شده را نیز به صف لینک های یافت شده اضافه کنید تا عملیات استخراج اطلاعات از قالب های HTML شروع گردد.
3. با توجه به ساختار صفحات سایت Reseacrh Gate مشاهده می‌شود که برخی اطلاعات مورد نیاز مانند ارجاعات در URL متفاوت با صفحه اصلی هر مقاله است، برای این منظور به این توضیحات توجه بفرمایید، لینک به یک مقاله به این صورت است: https://www.researchgate.net/publication/PaperID_PaperTitle که شناسه و نام مقاله ذکر به ترتیب PaperID و PaperTitle است. برای دسترسی به برخی اطلاعات مورد نیاز مانند ارجاعات نیاز است که در انتها لینک مقاله '/references را وارد کرده تا قالب HTML حاوی ارجاعات و لینک به سایر مقالات را دریافت کرده و سپس در این قالب HTML جدید عملیات خزش را صورت دهید.
4. ممکن است برخی از مقالات ارجاع نداشته و یا به حداقل تعداد 10 نرسد که در این صورت همان تعداد موجود واکشی شود.

در بالا به یافتن تگ ها به صورت بهینه اشاره شده، به این معنی که برای یافتن یک تگ خاص حاوی اطلاعات به صورت مشخص از طریق CSS selector همان تگ را واکشی کنید و مجبور به استفاده از for برای یافتن تگ مورد نظر از بین تمام تگ های موجود نباشد، برای مثال برای یافتن لینک ارجاعات تمام تگ های a را دریافت نکنید، بلکه به صورت مستقیم سراغ تگ مورد نظر بروید (برای این منظور مقداری برای آشنایی با فرم HTML سایت از طریق استفاده از Inspect وقت بگذارید)

لینک مقالات برای شروع به این ترتیب است :

https://www.researchgate.net/publication/323694313_The_Lottery_Ticket_Hypothesis_Training_Pruned_Neural_Networks
https://www.researchgate.net/publication/317558625_Attention_Is_All_You_Need
https://www.researchgate.net/publication/328230984_BERT_Pre-training_of_Deep_Bidirectional_Transformers_for_Language_Understanding

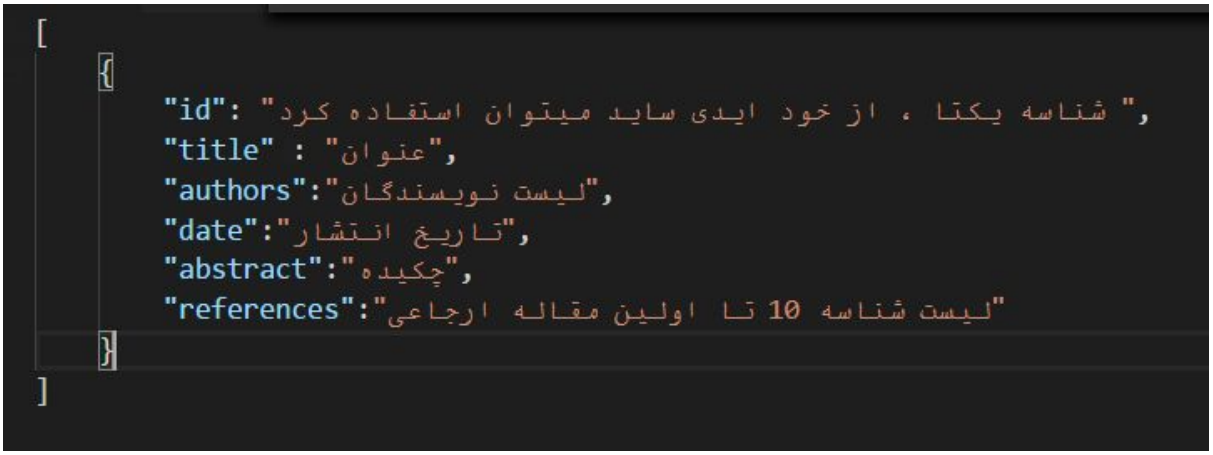
برای جمع بندی این قسمت ورودی و خروجی تابع مورد نظر که باید توسط شما نوشته شود را خلاصه می کنیم

ورودی :

1. آدرس مقالات شروع (3 لینک بالا)
2. تعداد مقالات برای استخراج اطلاعات (پیش فرض 2000)

خروجی :

1. فایل json تولید شده توسط Scrapy که دارای اطلاعات ذکر شده در بالا است(نویسندگان و ارجاعات به صورت لیست هستند)، فیلدها به این شرح می‌باشد :



علاوه بر خروجی ها بالا اطلاعات دیگری قابل استخراج است که در قسمت امتیازی قرار می گیرد

خروجی امتیازی :

1. نویسندگان مقالات با نقطه (.) از هم جدا شده اند ، در صورتی که تعداد نویسندگان بیشتر از تعدادی باشد برای دسترسی به سایرین باید از لینک show all n authors آنها را دریافت کنید.

بخش دوم: شاخص‌گذاری اطلاعات واکشی شده(۲۰)

در این بخش از پروژه از شما می‌خواهیم تا اطلاعاتی را که در بخش اول بدست آوردید را با استفاده از ElasticSearch شاخص‌گذاری کنید.

ElasticSearch یک موتور جستجو بر پایه‌ی Lucene است. ElasticSearch یک Restful web service راه‌اندازی می‌کند که عملیات کاربر با استفاده از درخواست به این وب‌سرور انجام می‌شود. این موتور جستجو قابلیت شاخص‌گذاری به صورت چندبخشی و نگه‌داری آن‌ها به صورت توزیع‌شده را داراست. به این معنی که شاخص‌ها را در قسمت‌های متفاوتی نگهداری می‌کند و این امکان را دارد که از هر قسمت تعدادی کپی نگه دارد.

برای ساخت شاخص لازم است که ابتدا یک نام برای شاخص انتخاب کرده و یک شاخص خالی ایجاد کنید سپس با ارسال درخواست‌های PUT یا POST به سرور ElasticSearch، اسناد را به شاخص اضافه کنید. جستجو کردن هم به راحتی و با ارسال درخواست به سرور انجام می‌شود.برای آشنایی بیشتر با این ابزار می‌توانید راهنمای آن را ملاحظه کنید.

برای سادگی ارتباط با ElasticSearch، می‌توانید از واسطهایی که در زبان‌های مختلف برای ارتباط با آن وجود دارد استفاده کنید. این واسطها به شما کمک می‌کند تا ارتباط با سرور ElasticSearch را از طریق فراخوانی کتابخانه‌ای انجام دهید.

ورودی:

- نتیجه‌ی عملیات خزش در قالب Json
- آدرسی که ElasticSearch روی آن در حال اجرا است(مثلا localhost:9200)

خروجی:

- تابعی جهت حذف تمام اطلاعات شاخص درون ElasticSearch
- تابعی که با فراخوانی آن اطلاعات مربوطه در ElasticSearch ذخیره شود.

اطلاعات مقالات را درون شاخص paper_index و حتما به این شکل ذخیره کنید:

```
{
  "paper": {
    "id": 'شناسه ی یکتا برای هر مقاله',
    "title": 'عنوان مقاله',
    "authors": ['نام اولین نویسنده' , ... ,'نام آخرین نویسنده'],
    "date": 'سال انتشار مقاله',
    "abstract": 'چکیده مقاله',
    "references": ['شناسه ی اولین مقاله ارجاعی' ,... ,'شناسه ی آخرین مقاله ارجاعی'],
  }
}
```

بخش سوم: ارزش گذاری مقالات(۱۵)

در این قسمت از شما می‌خواهیم تا مقدار page rank را برای صفحات موجود در ElasticSearch محاسبه کرده و نتیجه‌ی آن را به ElasticSearch اضافه کنید.

ورودی:

- اطلاعات ذخیره شده در بخش دوم
- α مورد نیاز در الگوریتم page rank

خروجی:

- محاسبه‌ی مقدار page rank برای تمامی مقالات و قرار دادن نتیجه در ElasticSearch به شکل زیر:

```
{
  "paper": {
    "page_rank": مقدار بدست آمده,
    ...
  }
}
```

بخش چهارم: جستجو (۲+۲۵)

در این بخش از شما خواسته شده است تا امکان جستجو در ElasticSearch را فراهم کنید به طوری که کاربر بتواند به صورت وزن‌دار بر اساس فیلدهای عمومی (عنوان مقاله، متن چکیده، سال انتشار) جستجو کند. امکان تاثیر دادن و یا ندادن page rank در مقدار امتیازی[1] که موجب مرتب‌سازی نتایج جستجو می‌شود، قسمت امتیازی این بخش خواهد بود. توجه کنید که تمامی این امکانات باید با شخصی‌سازی query های ارسالی به ElasticSearch پیاده‌سازی شود. به طور پیش‌فرض برای هر جستجو ۱۰ مقاله‌ی برتر را خروجی دهید.

در ارتباط با فیلدهای عمومی جستجو در نظر داشته‌باشید که جستجوی این موارد می‌تواند وزن‌های متفاوتی داشته باشد با این تعریف که ممکن است یافتن مقاله با عنوانی شامل کلمه‌ای خاص برای ما اهمیت بیشتری نسبت به یافتن مقاله‌ای با چکیده‌ای شامل کلمه‌ی دیگر داشته باشد. به علاوه منظور از جستجو بر اساس فیلد سال انتشار این است که با جستجوی سال ۲۰۱۵ قصد داریم مقالاتی که در سال‌های بعد از ۲۰۱۵ و یا در سال ۲۰۱۵ به چاپ رسیده‌اند امتیاز بالاتری بگیرند و این که مقاله دقیقا در سال ۲۰۱۵ چاپ شده باشد مدنظر نیست.

ورودی:

- آدرسی که ElasticSearch روی آن در حال اجراست.
- اطلاعات ذخیره‌شده در بخش‌های پیشین
- وزن مربوط به قسمت‌های عنوان مقاله، متن چکیده و سال انتشار
- مشخص کردن استفاده یا عدم استفاده از page rank در نتایج جستجو

خروجی:

- لیست مقالات با ترتیب صحیح و اطلاعات هر مقاله شامل عنوان مقاله، متن چکیده، نویسندگان مقاله و سال انتشار مقاله

خروجی امتیازی:

- تاثیر page rank در نتایج جستجو

[1] score

دقت کنید که همانطور که گفته شد تمامی امکانات باید با شخصی‌سازی query های ارسالی به ElasticSearch پیاده‌سازی شود. یعنی واکشی تمامی مقالات و مرتب‌سازی با امکانات زبان برنامه‌سازی قابل قبول نیست.

بخش پنجم: رتبه‌بندی نویسندگان بر اساس HITS[2] (+۸)

تعریف می‌کنیم فرد A به فرد B ارجاع دارد اگر فرد A مقاله‌ای نوشته باشد که به یکی از مقالات فرد B ارجاع داشته باشد. در چنین حالتی پیوندی از A به B در نظر می‌گیریم.

با این تعریف جدید می‌خواهیم با استفاده از روش HITS و محاسبه‌ی شاخص‌های hub و authority نویسندگان سایت را رتبه‌بندی کنیم. برای این کار لازم است از معیار authority برای امتیازدهی به افراد و تشخیص افراد شاخص (n نفر برتر) استفاده کنید.

ورودی:

- آدرسی که ElasticSearch روی آن در حال اجراست
- تعداد نویسندگان برتر موردنظر (n)

خروجی:

- لیست n نویسنده‌ی برتر اول به همراه معیار authority برای هر کدام.

برای پیاده‌سازی HITS عدد ۵ را برای تعداد اجرای حلقه در نظر بگیرید.

[2] Hyperlink_induced topic search

بخش ششم: رتبه‌بندی[3] (+۸)

این بخش ارتباطی با بخش‌های ۱ تا ۵ ندارد.

در این بخش از پروژه قصد داریم به صورت تحت‌نظارت[4] میزان ارتباط پرس‌وجوها با اسناد را بررسی کنیم. برای این کار از مجموعه‌ی داده‌های MQ2008 استفاده می‌کنیم. MQ2008 یکی از مجموعه‌ی داده‌های تحت نظارت در در دیتاست LETOR 4 است که برای پژوهش در حوزه‌ی یادگیری رتبه‌بندی[5] استفاده می‌شود.

دراین مجموعه‌ی داده هر سطر نشان‌دهنده‌ی یک زوج (پرس‌وجو، سند) است.

2 qid:10032 1:0.056537 2:0.000000 3:0.666667 4:1.000000 5:0.067138 6:0.000000 7:0.000000 8:0.000000 9:0.000000 10:0.000000 11:0.058781 12:0.000000 13:0.591833 14:1.000000 15:0.066747 16:0.003980 17:0.000000 18:0.296296 19:0.200000 20:0.004012 21:0.946170 22:0.732324 23:0.520967 24:0.562389 25:0.000000 26:0.000000 27:0.000000 28:0.000000 29:0.504600 30:0.616488 31:0.215857 32:0.723049 33:1.000000 34:0.000000 35:0.000000 36:0.000000 37:0.953885 38:0.910033 39:0.490034 40:0.843384 41:0.000000 42:0.125000 43:0.000000 44:0.000000 45:0.000000 46:0.076923 #docid = GX029-35-5894638 inc = 0.0119881192468859 prob = 0.139842

به عنوان نمونه به یکی از سطرهای این دیتاست نگاه می‌کنیم. عددی که در ستون اول (از سمت چپ) قرار دارد به معنی میزان ارتباط پرس‌وجو با سند است. این عدد می‌تواند ۰، ۱ یا ۲ باشد که عدد بزرگتر به معنی ارتباط بیشتر پرس‌وجو با آن سند است. ستون دوم شناسه‌ی پرس‌وجو را نشان می‌دهد. دقت کنید که در این مجموعه‌ی داده بازای هر پرس‌وجو تعدادی سند وجود دارد که پشت سر هر قرار گرفته است مثلا در داده‌های آموزشی که در اختیار شما قرار گرفته است ۸ سطر اول مربوط به یک پرس‌وجو با شماره‌ی ۱۰۰۰۲ است. ۴۶ ستون بعدی نشان‌دهنده‌ی ویژگی‌هایی است که به هر سند نسبت داده شده و سند با آن‌ها نمایش داده می‌شود. این ویژگی‌ها به ترتیب مربوط به این اطلاعات هستند:

[3] Ranking
[4] Supervised
[5] Learning to rank

Column in Output	Description
1	TF(Term frequency) of body
2	TF of anchor
3	TF of title
4	TF of URL
5	TF of whole document
6	IDF(Inverse document frequency) of body
7	IDF of anchor
8	IDF of title
9	IDF of URL
10	IDF of whole document
11	TF*IDF of body
12	TF*IDF of anchor
13	TF*IDF of title
14	TF*IDF of URL
15	TF*IDF of whole document
16	DL(Document length) of body
17	DL of anchor
18	DL of title
19	DL of URL
20	DL of whole document
21	BM25 of body
22	BM25 of anchor
23	BM25 of title
24	BM25 of URL

25	BM25 of whole document
26	LMIR.ABS of body
27	LMIR.ABS of anchor
28	LMIR.ABS of title
29	LMIR.ABS of URL
30	LMIR.ABS of whole document
31	LMIR.DIR of body
32	LMIR.DIR of anchor
33	LMIR.DIR of title
34	LMIR.DIR of URL
35	LMIR.DIR of whole document
36	LMIR.JM of body
37	LMIR.JM of anchor
38	LMIR.JM of title
39	LMIR.JM of URL
40	LMIR.JM of whole document
41	PageRank
42	Inlink number
43	Outlink number
44	Number of slash in URL
45	Length of URL
46	Number of child page

ستون بعدی شناسه‌ی سند است. دو ستون آخر را می‌توانید اصلا در نظر نگیرید.(inc و prob) برای آشنایی بیشتر با مجموعه‌ی داده می‌توانید به اینجا نگاه کنید.

برای راحتی بیشتر داده‌ها به صورت سه مجموعه‌ی آموزش، اعتبارسنجی[6] و تست در اختیار شما قرار می‌گیرد.

کاری که شما باید انجام دهید این است که داده‌ها را به شکل مناسب برای اجرای الگوریتم Ranking SVM در آورید. یعنی بازای هر پرس‌وجو q زوج‌های (d_i, d_j, q) را تشکیل دهید(همان‌طور که گفتیم اسناد بازیابی شده‌ی مربوط به یک پرس‌وجو در سطرهای پشت‌سر هم قرار گرفته است) .سپس SVM را بر روی داده‌ها آموزش دهید کنید. برای بدست آوردن بهترین پارامتر برای الگوریتم SVM از مجموعه‌ی اعتبارسنجی استفاده کنید. به این معنی که پارامتر بهینه الگوریتم SVM را بدون دیدن مجموعه‌ی تست و با استفاده از مجموعه‌ی اعتبارسنجی پیدا کنید. در نهایت کارایی سیستم را روی داده‌ی تست گزارش کنید برای این کار از معیار NDCG@5 استفاده کنید.

[6] Validation

نکات تکمیلی:

- برای اجرای برنامه طراحی یک واسط کاربری تحت کنسول که به صورت واضح قابلیت اجرای ۵ بخش ابتدایی را داشته باشد کفایت می‌کند.
- **بسیار مهم:** به دلیل اتفاقات اخیر و این که ممکن است تحویل حضوری نداشته باشیم نیاز است تا برای هر کدام از بخش‌های پروژه گزارش تهیه کنید و آن را همراه با کدهای خود در کوئرا بارگذاری کنید. دقت کنید که باید تمامی نکات خواسته شده در صورت پروژه را در گزارش نشان دهید. مثلا در بخش جستجو با تغییر وزن‌ها تغییراتی که در نتایج مشاهده می‌شود را در گزارش بیاورید. همچنین در بخش ششم گزارشی کامل از راه حل خود ارائه کنید و کارایی مدل را هم گزارش کنید.