دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

درس:

بازیابی اطلاعات

تعریف پروژه

بهار ۱۴۰۱

مقدمه

در این پروژه می‌خواهیم بصورت عملی از مفاهیم تدریس‌شده در کلاس درس استفاده کنیم. پروژه در سه فاز تعریف می‌شود که انجام دو فاز اول الزامی و فاز سوم امتیازی می‌باشد. در دو فاز اول از شما می‌خواهیم یک موتور جستجو برای بازیابی اسناد متنی ایجاد کنید به گونه‌ای که کاربر پرسمان خود را وارد نموده و سامانه اسناد مرتبط را بازنمایی کند.

در انجام پروژه به نکات زیر توجه فرمایید:

- تنها در موارد ذکرشده در تمرین مجاز به استفاده از کتابخانه‌های آماده هستید.
- کدهای خود را در کوئرا بارگذاری نمایید (آدرس مربوطه در سایت درس قرار داده می‌شود).
- کدهای شما (به همراه کدهای دانشجویان ترم‌های گذشته) توسط کوئرا بررسی می‌شود. در صورت وجود شباهت، نمره‌ی طرفین صفر خواهد شد.
- ملاک اصلی انجام فعالیت ارائه گزارش مربوطه است و ارسال کد بدون گزارش فاقد ارزش است. سعی کنید گزارش شما دقیقا در راستای موارد خواسته‌شده باشد و از طرح موارد اضافی خودداری کنید.
- مهلت ارسال فاز اول پروژه تا پایان روز ۲۶ فروردین ماه و فاز دوم تا پایان روز ۱۴ خرداد و فاز سوم (اختیاری) تا پایان روز ۵ تیر ماه می‌باشد.
- انجام فازهای یک و دو پروژه الزامی بوده و هر کدام ۵۰ درصد از کل نمره‌ی پروژه درس را به خود اختصاص می‌دهند.
- انجام فاز سوم پروژه اختیاری است و نمره‌ی امتیازی برای آن در نظر گرفته شده‌است.
- به ازای هر روز تاخیر در در فاز اول ۵ درصد از نمره‌ی فاز مربوطه کسر می‌شود.
- موعد تحویل متعاقبا از طریق سایت درس اعلام خواهد شد.

راهنمایی:

در صورت نیاز می‌توانید سوالات خود در خصوص پروژه را از تدریسیاران درس، از طریق ایمیل زیر بپرسید.

IR.course1400@gmail.com

۱- فاز اول

در این فاز از پروژه به منظور ایجاد یک مدل بازیابی اطلاعات ساده نیاز است تا اسناد شاخص‌گذاری شوند تا در زمان دریافت پرسمان از شاخص مکانی برای بازیابی اسناد مرتبط استفاده شود. به طور خلاصه مواردی که در این فاز انجام شوند به شرح زیر می‌باشد.

- پیش‌پردازش داده‌ها
- ساخت شاخص مکانی
- پاسخ‌دهی به پرسمان کاربر

در ادامه هر مورد به صورت کامل شرح داده می‌شود.

۱-۱ پیش‌پردازش اسناد

قبل از ساخت شاخص مکانی لازم است متون را پیش‌پردازش کنید. گام‌های لازم در این قسمت به صورت زیر می‌باشد.

- استخراج توکن
- نرمال‌سازی متون
- حذف کلمات پر تکرار[۱]
- ریشه‌یابی

برای انجام پیش‌پردازش‌های لازم می‌توانید با صلاح‌دید خود یکی از کتابخانه‌های آماده را انتخاب و از آن استفاده کنید (راهنمایی: کتابخانه۱ و کتابخانه۲) و یا پیاده‌سازی شخصی خود را داشته باشید.

توجه: برای پیاده‌سازی شخصی بخش‌های مربوط به پیش‌پردازش اسناد نمره‌ی ارفاقی لحاظ نمی‌شود.

[۱] Stop Words

۱-۲ ساخت شاخص مکانی

با استفاده از اسناد پیش‌پردازش‌شده در گام قبل، شاخص مکانی را بسازید. در شاخص مکانی ساخته شده علاوه بر جایگاه کلمات در اسناد، باید به ازای هر کلمه از دیکشنری مشخص باشد که تعداد تکرار آن کلمه در کل اسناد چقدر است. همچنین باید مشخص باشد که در هر سند تعداد تکرار یک کلمه‌ی مشخص چقدر است. جزئیات کامل این قسمت در بخش ۲,۴,۲ از کتاب مرجع درس قابل مشاهده است. برای پیاده‌سازی این قسمت می‌توانید به اختیار خود یک ساختمان داده‌ی مناسب را انتخاب کنید. (دقت کنید که ساختمان داده‌ی انتخابی به‌گونه‌ای نباشد که در زمان جستجو و دیگر عملیات، سرعت مدل را پایین آورد.)

۱-۳ پاسخ‌دهی به پرسمان کاربر

در این بخش پرسمان کاربر در قالب یک متن آزاد دریافت می‌گردد. حداقل عملگرهای قابل استفاده در این بخش «!» بعنوان عملگرد NOT و "" برای تعین یک عبارت می‌باشد. پس از بازیابی، اسناد را بصورت رتبه‌بندی شده نمایش دهید. برای رتبه‌دهی به اسناد، سندی که تعداد بیشتری از کلمات پرسمان را در خود دارد مرتبط‌تر است.

۱-۴ مجموعه داده

مجموعه داده مورد استفاده در این پروژه مجموعه‌ای از خبرهای واکشی‌شده از چند وب‌سایت خبری فارسی است که در قالب یک فایل JSON در اختیار شما قرار خواهدگرفت. لازم است تنها محتوای "content" را بعنوان محتوای سند پردازش کنید. شماره‌ی هر خبر را به عنوان id آن سند (خبر) در نظر بگیرید و در زمان پاسخ به پرسمان، عنوان خبر و URL مربوط به سند بازیابی‌شده را نمایش دهید تا امکان بررسی صحت عملکرد سیستم وجود داشته‌باشد.

۱-۵ گزارش

۱. با ذکر مثال شرح دهید که در گام پیش‌پردازش چه عملیاتی انجام داده‌اید. همچنین دلیل انجام هر پردازش را ذکر کنید.

۲. صحت قانون Zipf را در دو حالت قبل و بعد از حذف کلمات پرتکرار از واژه‌نامه بررسی کنید (رسم نمودار برای هر حالت الزامی است.) در صورت برقراری/ عدم برقراری این قانون در هر حالت، علت را شرح دهید.

۳. صحت قانون heaps را در دو حالت قبل و بعد از ریشه‌یابی بررسی کنید. برای بررسی این قانون لازم است با استفاده از اندازه‌ی واژه‌نامه و تعداد توکن‌ها در ۵۰۰، ۱۰۰۰، ۱۵۰۰ و ۲۰۰۰ سند اول، اندازه‌ی واژه‌نامه مربوط به کل اسناد تخمین زده شود. در نهایت اندازه‌ی واقعی واژه‌نامه و اندازه‌ی تخمینی در هر دو حالت مقایسه و تحلیل شود. آیا در هر دو حالت قانون برقرار است؟ چرا؟ (رسم نمودار برای هر حالت الزامی است.)

۴. حداقل سه مورد از مواردی که در ریشه‌یابی با چالش روبرو بودید را ذکر کنید. (بطور مثال کلماتی که نیازی به ریشه‌یابی ندارند اما طبق روند ریشه‌یابی از دست می‌روند.)

۵. پاسخ به پرسمان در حالت‌های زیر:

الف) یک پرسمان از کلمات ساده و متداول (مانند تحریم‌های آمریکا علیه ایران، در نتایج بازیابی‌شده انتظار می‌رود اسنادی که کلمات تحریم، آمریکا، علیه و ایران را دارند در بالای لیست و اسنادی که برخی از کلمات را ندارند در رتبه‌های پایین‌تر لیست قرار داشته باشند.)

ب) یک پرسمان با عملگر NOT (مانند تحریم‌های آمریکا ! ایران، انتظار می‌رود اسنادی که شامل دو کلمه تحریم و آمریکا هستند اما کلمه‌ی ایران را ندارند در نتایج بازیابی‌شده وجود داشته باشند.)

پ) یک پرسمان با عملگر عبارت (مانند "کنگره ضدتروریست"، انتظار می‌رود اسنادی که شامل عبارت کنگره ضدتروریست در نتایج بازیابی‌شده وجود داشته باشند؛ بعبارت دیگر موقیت مکانی کلمات در این حالت مهم است.)

ت) یک پرسمان پیچیده (مانند "تحریم هسته‌ای" آمریکا ! ایران، انتظار می‌رود اسنادی که شامل عبارت تحریم هسته‌ای و کلمه‌ی آمریکا هستند اما کلمه‌ی ایران را ندارند در نتایج بازیابی‌شده وجود داشته باشد.)

ث) یک پرسمان کلمات نادر (مانند اورشلیم ! صهیونیست، خروجی مورد انتظار این قسمت مشابه با قسمت ب می‌باشد با این تفاوت که کلمات استفاده شده در پرسمان از کلمات نادر هستند.)

در هر مورد، تیتر خبر بازیابی شده را به همراه جمله(هایی) از هر سند بازیابی شده، که حاوی عبارت پرسمان بوده‌اند، گزارش کنید. همچنین در هر مورد با ذکر جزئیات شرح دهید که آیا سند بازیابی شده به پرسمان کاربر مرتبط هست یا خیر؟

توجه۱: در مواردی که تعداد اسناد بازیابی‌شده زیاد است، تنها ۵ سند اول را در گزارش وارد کنید.

توجه۲: تیتر اخبار را با فرمت مناسب و خوانا در گزارش خود بنویسید.

۲- فاز دوم

در این مرحله می‌خواهیم مدل بازیابی اطلاعات را گسترش و بازنمایی اسناد را به صورت برداری انجام دهیم تا بتوانیم نتایج جستجو را بر اساس ارتباط آن‌ها با پرسمان کاربر رتبه‌بندی کنیم. به این صورت که برای هر سند یک بردار عددی استخراج می‌شود که بازنمایی آن سند در فضای برداری است و این بردارها ذخیره می‌شوند. در زمان دریافت پرسمان، ابتدا بردار متناظر با آن پرسمان در همان فضای برداری ساخته و سپس با استفاده از یک معیار شباهت مناسب، شباهت بردار عددی پرسمان با بردار تمام اسناد در فضای برداری محاسبه می‌شود و در نهایت نتایج خروجی بر اساس میزان شباهت مرتب‌سازی می‌شوند. برای افزایش سرعت پاسخگویی مدل بازیابی اطلاعات می‌توان روش‌های مختلفی را به کار گرفت که به تفصیل در ادامه بیان می‌شود.

۲-۱ مدل‌سازی اسناد در فضای برداری

در مرحله قبل پس از استخراج توکن‌ها اطلاعات به صورت یک دیکشنری و شاخص مکانی ذخیره شدند. در این بخش هدف آن است که اسناد در فضای برداری بازنمایی شوند. با استفاده از روش وزن‌دهی $tf-idf$ بردار عددی برای هر سند محاسبه خواهد شد و درنهایت هر سند به صورت یک بردار شامل وزن‌های تمام کلمات آن سند بازنمایی می‌شود. محاسبه‌ی وزن هر کلمه t در یک سند d با داشتن مجموعه‌ی تمام اسناد D با استفاده از معادله‌ی زیر محاسبه می‌شود:

$$tfidf(t.d.D) = tf(t.d) \times idf(t,D) = (1 + \log(f_{t.d})) \times \log(\frac{N}{n_t})$$

که در آن $f_{t.d}$ تعداد تکرار کلمه‌ی t در سند d و n_t تعداد سندهایی است که کلمه‌ی t در آنها ظاهر شده است. توضیحات بیشتر این روش در فصل ۶ کتاب مرجع درس آمده است.

در نمایش برداری فوق برای کلمه‌ای که در یک سند وجود نداشته باشد وزن صفر در نظر گفته می‌شود و از این جهت بسیاری از عناصر بردارهای محاسبه شده صفر خواهد بود. برای صرفه جویی در مصرف حافظه به جای آن که برای هر سند یک بردار عددی کامل در نظر بگیرید که بسیاری از عناصر آن صفر هستند می‌توانید وزن کلمات در اسناد مختلف را در همان لیست‌های پست‌ها ذخیره کنید. در زمان پاسخ‌گویی به پرسمان کاربر که در ادامه

توضیح داده می‌شود نیز همزمان با جستجوی کلمات در لیست‌های پست‌ها می‌توانید وزن کلمات در اسناد مختلف را نیز واکشی کنید و به این شکل تنها عناصر غیر صفر بردارهای اسناد ذخیره و پردازش می‌شوند.

۲-۲ پاسخ‌دهی به پرسمان در فضای برداری

با داشتن پرسمان کاربر، بردار مخصوص پرسمان را استخراج کنید (وزن کلمات موجود در پرسمان را محاسبه کنید). سپس با استفاده از معیار شباهت سعی کنید اسنادی را که بیشترین شباهت (کمترین فاصله) را به پرسمان ورودی دارند پیدا کنید. سپس نتایج را به ترتیب شباهت نمایش دهید. معیارهای فاصله‌ی مختلفی می‌تواند برای این کار در نظر گرفته شود که ساده‌ترین آنها شباهت کسینوسی بین بردارها است که زاویه‌ی بین دو بردار را محاسبه می‌کند. این معیار به صورت زیر تعریف می‌شود:

$$similarity(a.b) = \cos(\theta) = \frac{a.b}{\|a\|\|b\|} = \frac{\sum_{i=1}^{N} a_i b_i}{\sqrt{\sum_{i=1}^{N} a_i^2}\sqrt{\sum_{i=1}^{N} b_i^2}}$$

توجه کنید که برای افزایش سرعت می‌توانید با استفاده از تکنیک $Index\ elimination$ شباهت کسینوسی را با اسنادی که امتیاز صفر خواهند گرفت محاسبه نکنید. در انتهای کار برای نمایش یک صفحه از نتایج پرسمان تنها کافیست K سندی انتخاب شوند که بیشترین شباهت را به پرسمان دارند.

۲-۳ افزایش سرعت پردازش پرسمان

با استفاده از تکنیک $Index\ elimination$ تا حدودی مشکل زیاد بودن زمان در مرحله قبل حل می‌شود اما همچنان زمان پاسخگویی برای بسیاری از کاربردها قابل قبول نمی‌باشد. برای آنکه سرعت پردازش و پاسخگویی افزایش یابد می‌توانید از $Champion\ lists$ استفاده کنید که قبل از آنکه پرسمانی مطرح شود و در مرحله پردازش اسناد، یک لیست از مرتبط‌ترین اسناد مربوط به هر $term$ در لیست جداگانه‌ای نگهداری شود. برای پیاده‌سازی این بخش پس از ساخت شاخص معکوس مکانی، $Champion\ list$ را ایجاد کنید و تنها بردار پرسمان را با بردار اسنادی که از طریق جستجو در $Champion\ list$ به دست آورده‌اید مقایسه کنید و K سند مرتبط را به نمایش بگذارید. توضیحات بیشتر این روش در فصل ۷ کتاب آمده است.

توجه: می‌توانید وزن دهی $tf-idf$ و ایجاد لیست $Champion$ را با استفاده از شاخص مکانی که در مرحله قبل پیاده‌سازی کردید، انجام دهید.

۲-۴ گزارش

۱. پاسخ به پرسمان در حالت‌های زیر:

الف) یک پرسمان از کلمات ساده و متداول تک کلمه‌ای
ب) یک پرسمان از عبارات ساده و متداول چند کلمه‌ای
پ) یک پرسمان دشوار و کم تکرار تک کلمه‌ای
ت) یک پرسمان دشوار و کم تکرار چند کلمه‌ای

در هر مورد، تیتر خبر بازیابی شده را به همراه جمله(هایی) که حاوی عبارت پرسمان بوده‌اند، گزارش کنید. همچنین در هر مورد با ذکر جزئیات شرح دهید که آیا سند بازیابی شده به پرسمان کاربر مرتبط هست یا خیر؟ تحلیل هر مورد الزامی است.

۲. موارد ب و ت را با روش مکانی فاز یک نیز تکرار کنید و نتایج دو حالت را با هم مقایسه و تحلیل کنید.

۳- فاز سوم

در این بخش از پروژه قصد داریم که موتور جستجوی خود را به کمک الستیک سرچ[۲]بسازیم. در ابتدا لازم است بدانیم که الستیک سرچ یک موتور جستجو و تجزیه و تحلیل توزیع شده است و پردازش‌های لازم را با سرعت بالایی برای انواع داده‌ها فراهم می‌کند. تفاوتی ندارد که متن ما ساختاریافته باشد یا بدون ساختار و یا داده‌ها از چه نوعی باشند، در هر صورت الستیک سرچ می تواند داده‌ها را به گونه‌ای کارآمد ذخیره و شاخص‌سازی کند که امکان جستجوهای سریع را داشته باشیم. در این بخش از پروژه قصد داریم تا با برخی از قابلیت‌های الستیک سرچ آشنا شویم. برای آشنایی بیشتر با ویژگی‌ها و قابلیت‌ها می‌توانید به مستندات موجود در سایت رسمی الستیک سرچ مراجعه نمایید.

۳-۱ ذخیره اسناد و ساخت شاخص در الستیک سرچ

مشابه دو بخش قبل ابتدا پیش‌پردازش‌های لازم را بر روی اسناد انجام دهید. در مرحله‌ی بعد برای استفاده از الستیک سرچ لازم است یک cluster ایجاد کرده و در آن شاخص خود را با نامی مناسب بسازید، سپس اسناد موجود در فایل json را که پیش‌پردازش بر روی آن‌ها صورت گرفته، به شاخص خود اضافه نمایید. برای اضافه کردن اسناد به شاخص، از bulk API استفاده می‌کنیم. که در مقایسه با تابع Index بسیار سریع‌تر عمل می‌کند.

Elasticsearch

۳-۲ پاسخ‌دهی به پرسمان کاربر

در این بخش پرسمان‌ها در قالب یک متن آزاد از کاربر دریافت می‌گردد. همانند فاز یک انتظار می‌رود استفاده از عملگرهای NOT و پرسمان عبارتی[۳] (که در فاز یک با " " مشخص شده بود) برای کاربر ممکن باشد. اسناد بازیابی شده به ترتیب و براساس رتبه‌بندی استخراج می‌شوند. اسنادی در رتبه بالاتر قرار می‌گیرند که تعداد بیشتری از کلمات پرسمان را در خود داشته باشند. برای مثال اگر یک کوئری چهار کلمه‌ای مثل "بررسی کاهش قیمت نفت "را داشته باشیم سندی که هر چهار کلمه را داشته باشد باید نسبت به سندی که تنها سه کلمه از این چهار کلمه را دارد، در رتبه بالاتری قرار بگیرد.

گزارش

۱. ساخت شاخص در این فاز (با استفاده از Bulk API) را با ساخت شاخص مکانی در فاز یک از نظر زمانی (پیاده سازی و زمان اجرا) مقایسه نمایید.

۲. پرسمان‌های زیر را در نظر بگیرید

الف) یک پرسمان دشوار (مانند "تحریم هسته‌ای" آمریکا ! ایران)

ب) یک پرسمان از کلمات نادر (مانند اورشلیم ! صهیونیست)

برای هر کدام از موارد فوق پرسمان مورد استفاده در فاز یک را تکرار کنید و عملکرد دو موتور بازیابی را از نظر سرعت بازیابی اسناد و کیفیت رتبه‌بندی اسناد مرتبط مقایسه نمایید.

۳. با ذکر علت بیان کنید شما به عنوان کاربر استفاده از کدام مدل را ترجیح می‌دهید.

توجه: برای بررسی دقت رتبه‌بندی، بررسی سه سند اول کافیست.

۳-۳ تصحیح املایی

یکی از کاربردهای الستیک سرچ، طراحی سیستم‌های تصحیح املایی است. در این بخش قصد داریم با کمک قابلیت‌های موجود در الستیک سرچ سیستمی طراحی کنیم که ورودی آن یک جمله بوده و خروجی آن آرایه‌ای از جملات با املای صحیح باشد. همانطور که در درس با مبحث تصحیح املایی آشنا شده‌اید می‌دانیم که یکی از روش‌های طراحی این سیستم‌ها استفاده از مدل‌های ngram است. برای طراحی یک سیستم تصحیح املایی

[۳] Phrase query

مبتنی بر ngram با کمک الستیک سرچ، ابتدا فایل Spelling_Correction.ipynb را دریافت کنید و سپس گام‌هایی که در ادامه شرح داده می‌شود را انجام دهید.

۳-۳-۱ ساخت شاخص trigram ها و bigram ها

برای طراحی این سیستم، از پیشنهاددهنده‌ها[4] استفاده خواهیم کرد. به کمک پیشنهاددهنده‌ها می‌توان برای هر کلمه، کلمات مشابه را پیدا کرده و آنها را با توجه به فاصله؛ ویرایش، امتیازدهی و مرتب کرد. هرچند این روش برای تصحیح املایی مبتنی بر محتوا کارآمد نمی‌باشد. برای اینکه بتوانیم غلط‌های املایی را با توجه به موقعیتی که در عبارت ورودی کاربر دارند تشخیص دهیم و پیشنهاددهنده‌های بهتری داشته باشیم، باید برای داده‌های متنی خود در سه قالب trigram، bigram و unigram شاخص بسازیم. برای این منظور در زمان ساخت شاخص مورد نظر تنها یک فیلد content_title تعریف کنید که داده‌های مربوط به فیلدهای content و title در مجموعه اسناد در آنها قرار می‌گیرد. الستیک سرچ این قابلیت را به ما می‌دهد که از یک فیلد چندین شاخص به شکل‌ها و انواع متفاوت داشته باشیم. با کمک analyzerها می‌توانیم عملیات‌های متفاوتی را بر روی اسنادی که می‌خواهیم وارد شاخص کنیم، یا پرسمان کاربر، انجام دهیم. برای فاز نگاشت[5] نکات زیر را در نظر بگیرید:

- برای این بخش نیازی به پیش پردازش مجموعه داده نیست، هرچند باید نیم فاصله‌های داخل متون به فاصله تبدیل شوند تا سیستم عملکرد بهتری داشته باشد. این کار باید با کمک char_filter ها انجام شود.
- در این بخش فیلد دیگری نباید تعریف شود.

پس از ساخت شاخص از طریق پارامترهایی که در درخواست ارسالی خود تعیین می‌کنیم، می‌توانیم عملکرد سیستم را کنترل کنیم. موارد زیر را در درخواست‌هایی که می‌فرستید لحاظ کنید (برای مطالعه در مورد هر پارامتر به این لینک مراجعه کنید) :

۱. روش هموارسازی[6]ای که استفاده می‌شود باید هموارسازی لاپلاس[7]باشد. مقداری که برای آلفا در نظر می‌گیرید را گزارش کنید. مقدار field را نیز برابر نامی که در هنگام نگاشت برای ngram خود تعریف

[4] suggester
[5] mapping
[6] smoothing
[7] laplace smoothing

کرده‌اید قرار دهید. برای مثال اگر نام آن را trigram گذاشته‌اید، مقدار field را content_title.trigram قرار دهید.

۲. تحقیق کنید که افزایش یا کاهش فیلد max_errors چه تاثیری بر روی کیفیت پیشنهاددهندهها و مدت زمان پاسخ درخواست می‌گذارد و نتیجه را در گزارش خود شرح دهید. برای این بخش مقدار آن را سه در نظر بگیرید.

۳. بررسی کنید که دو فیلد confidence و real_word_error_likelihood چه تاثیری بر امتیاز عبارت ورودی و عبارت‌های پیشنهادی دارند. مقادیری که برای این دو عبارت در نظر می‌گیرید را گزارش کنید و دلیل آن را شرح دهید.

۴. پیشنهاددهندهها از مجموعه‌ای از کلمات پیشنهادی که به ازای هر کلمه در عبارت ورودی کاربر تولید شده‌اند استفاده می‌کنند. این کار به کمک direct_generator انجام می‌شود. یک مولد[8] داخل direct_generator تعریف کنید که مقدار field در آن content_title و مقدار prefix_length آن دو باشد.

گزارش

عملکرد سیستم را به ازای جملات قرار داده شده در ژوپیتر نوت بوک بررسی کنید. برای هر عبارت ورودی، حداکثر ۵ عبارت پیشنهادی را نمایش دهید.

۳-۳-۲ ساخت ایندکس با توکن‌های وارون شده[9]

در صورت رعایت نکات گفته‌شده در بخش قبل، دیده می‌شود که سیستم تصحیح املایی طراحی شده قادر به تولید پیشنهادهای مناسب برای کلماتی که حروف ابتدایی آن‌ها به درستی تایپ نشده‌اند نیست. یک راه برای حل این مشکل این است که مقدار فیلد prefix_length را برابر با صفر قرار دهیم که بهینه نیست چرا که در این حالت السیتک سرچ تمامی کلمات داخل دیکشنری را مورد بررسی قرار می‌دهد. راه دیگر ایجاد یک شاخص دیگر است که در آن توکن‌ها به صورت وارون ذخیره شده‌اند. شاخص دیگری ایجاد کنید که در علاوه بر ذخیره‌سازی فیلد content_title به صورت ngram، آن را به صورت وارونه نیز ذخیره کند. برای این کار analyzerای تعریف کنید

[8] generator
[9] reverse

که از Reverse token filter برای وارونه کردن توکن‌ها استفاده می‌کند. پس از ساخت ایندکس، در بدنه درخواست ارسالی، در کنار مولدی که در قسمت قبل تعریف شد، مولد دیگری تعریف کنید که ورودی کاربر را به توکن‌های وارونه تبدیل کند و پس از تولید کلمات پیشنهادی، آن‌ها را قبل از رفتن به فاز امتیازدهی، وارونه کند. (راهنمایی: از فیلد های pre_filter و post_filter استفاده کنید). پس از اعمال تغییرات ذکر شده، بررسی کنید که آیا بهبودی در پیشنهادهای تولید شده ایجاد شده است یا خیر؟

۳-۳-۲ تولید کلمات مترادف

در این بخش می‌خواهیم قابلیت دیگری را به سیستم تصحیح املایی خود اضافه کنیم. می‌خواهیم در کنار پیدا کردن غلط‌های املایی، کلمات معادلی را هم برای برخی کلمات داخل جمله قرار دهیم که اگر کاربر آنها را به عنوان کوئری وارد کند، نتایج مرتبط‌تر و بهتری را دریافت کند. پیش نیاز این کار داشتن نگاشت بین مترادف‌ها است. این نگاشت‌ها در فایل synonyms.txt قرار داده شده است. این فایل را در مسیری که در آن الستیک سرچ را قرار داده‌اید ذخیر کنید. سپس شاخص جدیدی با نگاشت مشابه با شاخص بخش قبل ایجاد کنید با این تفاوت که باید یک analyzer دیگر برای تولید کلمات مترادف ایجاد کنید. برای این کار می‌توانید از این لینک کمک بگیرید. پس از این کار در درخواستی که ارسال می‌کنید، علاوه بر مولدهایی که در بخش قبل تعریف کردید، مولد دیگری در بخش direct_generator باید تعریف کنید که تولید کلمات مترادف را بر عهده بگیرد.

۳-۴ به کارگیری ویژگی مدولاسیون شباهت[1] در الستیک سرچ

یکی از ویژگی‌های الستیک سرچ امکان تغییر معیار شباهت برای پیدا کردن شبیه‌ترین اسناد به کوئری مورد نظر است. به این ویژگی در الستیک سرچ مدولاسیون شباهت می‌گویند. در درس با دو مورد از این الگوریتم‌ها آشنا شدیم. ساده‌ترین الگوریتم مورد استفاده همان روشی است که در فاز اول این پروژه از آن استفاده کردید. روش دیگر نیز الگوریتم شباهت tf-idf است که در فاز دوم پیاده سازی شد.

الگوریتم پیش فرض استفاده شده در الستیک سرچ الگوریتم BM25 است که پایه و اساس آن tf-idf میباشد. در این بخش از پروژه ابتدا الگوریتم پیش فرض الستیک سرچ را توضیح میدهیم سپس با استفاده از scripted similarity الگوریتم td-idf را پیاده‌سازی کنید.

[1] Similarity Modulation

۳-۴-۱ شاخص BM25

همانطور که گفته شد الگوریتم پیش فرض محاسبه شباهت در الستیک سرچ BM25 است. این الگوریتم که بر پایه tf-idf پیاده‌سازی شده، بسیاری از مشکلات ناشی از این روش را برطرف می‌نماید. این الگوریتم با استفاده از دو پارامتر k1 و B سعی دارد تا تاثیر طول داکیومنت و tf را کنترل کند و داکیومنت‌های بهتری را برگرداند. در بخش‌های قبلی پروژه این شاخص به صورت پیشفرض در رتبه بندی داکیومنت ها اعمال میشد. برای مطالعه بیشتر راجب دو پارامتر k1 و B می‌توانید به این لینک مراجعه کنید.

۳-۴-۲ساخت شاخص با TF-IDF

در این بخش قصد داریم شاخصی را با استفاده از تابع شباهت tf-idf پیاده‌سازی نماییم. مشابه فاز قبلی ابتدا پیش‌پردازش‌های لازم را بر روی اسناد انجام دهید. پس از ساخت شاخص و اضافه کردن اسناد، در بخش تنظیمات مربوط به شاخص‌ها میتوانید تابع شباهت خود را تغییر دهید. توجه نمایید که لازم است قبل از اعمال تغییرات شاخص موردنظر را بسته و بعد از اعمال تغییرات آن را مجدد باز نمایید تا تغییرات به درستی اعمال شود. در بخش از تنظیمات میتوانید الگوریتم پیش فرض شباهت را تغییر دهید. برای این منظور باید نوع مدولاسیون را scripted قرار دهید و در بخش source، کد مورد نظر مربوط به این الگوریتم را وارد کنید. برای آشنایی با متغیرهای مورد نیاز در نوشتن کد tf-idf می‌توانید از این لینک کمک بگیرید.

❋ لینک های کمکی مورد نیاز در این بخش:

- لینک ۱
- لینک ۲

گزارش

۱. به پرسمان‌ها در حالت‌های زیر پاسخ دهید:

توجه: query خود را به شکل match کوئری برای فیلد content اخبار بزنید.

الف) یک پرسمان دشوار و کم تکرار تک کلمه‌ای

ب) یک پرسمان دشوار و کم تکرار چند کلمه‌ای

۲. در هر حالت پرسمانی که در فاز۲ استفاده کردید را تکرار کنید. نتایج بازگردانده شده را از نظر میزان ارتباط مقایسه و تحلیل نمایید.

۳-۵ دسته‌بندی به روش نزدیکترین همسایه[۱۱]

در این بخش دو مجموعه سند در اختیار شما قرار گرفته است که یکی دارای برچسب و دیگری بدون برچسب می‌باشد. می‌خواهیم با استفاده از الستیک سرچ، الگوریتم دسته‌بندی نزدیکترین همسایه را به سیستم بازیابی اطلاعات خود اضافه نمایید تا بتوانید با استفاده از آن دسته‌ی اسنادی را که برچسب ندارند، مشخص کنید. مشابه فاز قبل ابتدا پیش‌پردازش‌های لازم را بر روی اسناد انجام داده و سپس اسناد را در یک فضای برداری مناسب بازنمایی کنید. بازنمایی اسناد به صورت برداری را می‌توانید با کمک ماژولی که در اختیار شما قرار می‌گیرد انجام دهید. برای فاز نگاشت باید یک فیلد dense_vector تعریف کرده تا امکان ذخیره کردن بردار تولید شده را داشته باشید. در کنار این فیلد لازم است بخش محتوا و برچسب اسناد را نیز ذخیره کنید. در ادامه برای برچسب‌زنی سعی کنید یک مقدار مناسب برای پارامتر K انتخاب و هر سند را با کمک الستیک سرچ برچسب بزنید. برای این کار باید در ابتدا سند را به فضای برداری برده و بردار حاصل را به الستیک بدهید تا اسناد با بالاترین شباهت را باز گرداند. سپس با توجه به دسته‌های اسناد بازگردانده شده، برچسب سند مورد نظر را مشخص کرده و به شاخص اضافه نمایید. این کار به تعداد اسناد بدون برچسب تکرار می‌کنیم تا دسته‌ی تمام اسناد مشخص شود.

گزارش

۱. در این قسمت، بازیابی نتایج پرسمان را بر روی اسنادی که توسط KNN برچسب زده‌ایم انجام خواهیم داد. برای هر پرسمان علاوه بر متن پرسمان، برچسب مورد نظر خود را نیز مشخص می‌کنیم تا تنها اسنادی که حاوی برچسب مد نظر ما هستند در نتایج مشاهده کنیم. در این قسمت سه پرسمان چند کلمه‌ای در حوزه ورزشی، اقتصادی و سلامت مشخش کنید و نتایج بازیابی را بررسی و تحیلی کنید. برای مثال یک پرسمان چند کلمه ای در حوزه ی اخبار ورزشی خواهیم داشت: "نتایج مسابقات لیگ برتر فوتبال ایران" برچسب: "ورزشی". جهت بررسی عملکرد سیستم ۵ سند اول بازیابی شده را باز کرده و مشخص کنید که آیا به پرسمان ارتباطی دارد؟ همچنین در حوزه‌ی مد نظر قرار دارد؟ در صورتی که هر دو شرط مذکور رعایت شود سیستم بازیابی عملکرد قابل قبولی دارد.

[۱۱] KNN

توجه:

محدودیتی برای پیاده سازی قسمت بازیابی اسناد ندارید می‌توانید این عمل را با استفاده از یک یا چند Index انجام دهید. توجه داشته باشید پیاده سازی شما به گونه‌ای باشد که قابلیت مشخص کردن متن پرسمان و دسته‌ی مد نظر را داشته باشد.