به نام آنکه جان را فکرت آموخت

فاز اول پروژه درس بازیابی پیشرفته اطلاعات

پیاده‌سازی یک سیستم پایه بازیابی اطلاعات

ترم اول سال تحصیلی ۹۳-۹۴

دانشکده مهندسی کامپیوتر

دانشگاه شریف

مدرس:

دکتر سلیمانی

طراحان:

صهبا عظامی

علی یادگاری

مقدمه

در این پروژه شما سیستم بازیابی اطلاعات ساده‌ای را پیاده‌سازی خواهید کرد. در این راه می‌بایست شاخص‌گذاری[1] اسناد، فشرده‌سازی شاخص‌ها[2] و ذخیره آن‌ها، بازیابی رتبه‌بندی شده[3] آنها بر حسب پرسمان ورودی و همچنین پردازش توزیع‌شده با استفاده از چهارچوب هدوپ[4] را پیاده‌سازی کنید. بطور کلی برای انجام پروژه دو انتخاب پیش‌رو دارید. یکی پیاده‌سازی توزیع‌شده[5] سیستم (که این انتخاب شامل ۲۰ درصد نمره اضافه می‌باشد) و دیگری پیاده‌سازی متمرکز[6] سیستم (توزیع نشده!) می‌باشد.

پروژه را میتوان به سه بخش کلی تقسیم کرد. بخش اول طراحی و پیاده‌سازی یک شاخص‌گذار که آدرس محلی اسناد متنی اولیه را دریافت می‌کند و بعنوان خروجی داده‌ساختاری قابل جستجو از اسناد براساس کلمات موجود در آنها تولید می‌کند. همچنین در این بخش می‌بایست فشرده‌سازی شاخص‌ها را نیز انجام دهید. بخش دوم اضافه کردن امکان جستجو بر روی داده‌ساختار خروجی بخش قبل براساس مدل فضای برداری[7] را شامل می‌شود. در ابتدای این بخش، پس از دریافت پرسمان ورودی باید از درستی املا آن اطمینان حاصل کنید و امکان اصلاح املای[8] پرسمان ورودی را به سیستم خود اضافه کنید. بخش سوم و انتهایی نیز به ارزیابی سیستم بازیابی اطلاعات پیاده‌شده می‌پردازد. این ارزیابی براساس معیارهای recall, precision, MAP[9] خواهد بود. دقت کنید که این سه بخش باید به گونه‌ای طراحی شوند که امکان اجرای هرکدام از بخش‌ها بصورت مستقل از دیگری وجود داشته باشد.

---

[1] Indexing
[2] Index compression
[3] Ranked retrieval
[4] Hadoop
[5] Distributed
[6] Centralized
[7] Vector space model
[8] Spell correction
[9] Mean average precision

مجموعه اسناد[10]

اسناد مورد نظر برای استفاده در این پروژه، از مجموعه‌ی اسناد Cranfield انتخاب شده است. این مجموعه‌ی داده، شامل ۱۴۰۰ سند است که در آن هر سند، در یک فایل قرار گرفته شده است. همچنین به منظور سهولت در ارزیابی و مقایسه‌ی اسناد بازیابی شده با اسناد مرتبط واقعی، تعداد ۲۲۵ پرسمان به همراه شماره ی اسناد مرتبط به هر یک از آنها در پوشه‌ای مجزا و در کنار مجموعه داده قرار گرفته است.

نحوه‌ی ارتباط پرسمان و اسناد مرتبط با آن در فایل relevance مشخص شده است که در آن، هر سطر نشان‌دهنده‌ی یک پرسمان است. عدد اول هر سطر شماره‌ی پرسمان را نشان می‌دهد و اعداد بعدی ن سطر شماره‌ی اسناد مرتبط با آن را (به ترتیب صعودی) نشان می‌دهد. دقت کنید که این اعداد متناظر به نام فایل‌ها در پوشه‌ی docs و Queries است.

نمره‌دهی

تقسیم نمرات بین سه بخش پروژه به شرح زیر است:

- بخش اول (تحلیل و شاخص‌گذاری اسناد، فشرده‌سازی شاخص‌ها): ۵۰ نمره + ۵ نمره اضافه
- بخش دوم (اصلاح پرسمان، جستجو و بازیابی): ۳۰ نمره
- بخش سوم (ارزیابی سیستم): ۲۰ نمره
- سیستم بازیابی اطلاعات توزیع‌شده (هدوپ): ۲۰ نمره اضافه

[10] Dataset

صورت پروژه

بخش اول (تحلیل و شاخص‌گذاری اسناد، فشرده‌سازی شاخص‌ها)

در این بخش، ابتدا باید مجموعه‌ای از فایل‌ها را بخوانید و آنها را به token تجزیه کنید. تجزیه‌ی اسناد به token[11] برای تشکیل Dictionary و Posting List استفاده می‌شود. حداقل قواعدی که باید برای تجزیه اسناد به token استفاده کنید عبارت است از:

- منظور از یک token، دنباله‌ای از کاراکترهاست که این کاراکترها می‌توانند حروف انگلیسی، اعداد و نقطه باشد.
- نقطه به تنهایی یک token محسوب نمی‌شود و در صورت token است که بین دو عدد قرار گرفته باشد (به طور مثال در اعداد اعشاری) و یا جزیی از آدرس اینترنتی و ایمیل باشد.
- آدرس‌های اینترنتی و ایمیل نباید تجزیه شوند و بعنوان یک token محسوب می‌شوند.
- (راهنمایی) برای سهولت کار خود میتوانید از Regular Expression استفاده کنید.

پس از تجزیه‌ی متن به tokenها، باید عملیات زیر انجام شود:

- از تکنیک حذف لغات پرکاربرد استفاده کنید. برای اینکار از stopwordهای ارائه شده در مجموعه اسناد استفاده کنید.
- در بخش نرمال‌سازی، می‌بایست ریشه کلمات را ذخیره کنید و برای این‌کار از stemmerهای آماده استفاده کنید. بعنوان پیشنهاد می‌توانید از porter stemmer که پیاده‌سازی آن به زبان‌های برنامه‌نویسی مختلف در این لینک آمده است، استفاده کنید. همچنین تمامی Tokenها را بصورت lowercase ذخیره کنید.

پس از این مرحله می‌بایست شاخص‌گذاری اسناد را انجام دهید. انتخاب داده‌ساختار مناسب برای ذخیره اسناد و tokenها بر عهده خودتان می‌باشد

- برای تشکیل posting list از شماره اسناد موجود در مجموعه اسناد اولیه استفاده کنید.
- (راهنمایی) با توجه به استفاده از فضای برداری برای محاسبه شباهت در بخش بعدی به تعداد دفعات تکرار لغات در اسناد (Term Frequency) نیاز دارید.

بنابراین در این بخش حداقل می‌بایست دو داده‌ساختار را در پایگاه‌داده ذخیره کنید. برای سهولت کار از ذخیره در پایگاه‌داده صرف نظر کنید و داده‌ساختارهای خود را در دیسک ذخیره کنید. حداقل انتظار کارکرد از ذخیره‌سازی سیستم شما، امکان Load/Save خواهد بود. درصورت تمایل به ذخیره داده در پایگاه‌داده می‌توانید از mongoDB استفاده کنید.

---

[11] Tokenize

فشرده سازی: در سیستم‌های بازیابی اطلاعات از تکنیک‌های مختلفی برای کاهش حجم شاخص‌ها استفاده می‌شود. در این قسمت می‌بایست یکی از تکنیک‌های معرفی شده در کلاس درس را پیاده‌سازی کنید. فشرده‌سازی باید در هر دو قسمت ,dictionary posting list باشد. برای فشرده‌سازی dictionary از تکنیک Dictionary-as-string استفاده کنید. برای فشرده‌سازی posting list تکنیک Gap-storing استفاده می‌شود. با توجه به نحوه ذخیره‌سازی integer در جاوا و تعداد اسناد موجود در مجموعه اسناد پروژه، استفاده از این تکنیک به کاهش حجم posting list کمکی نمی‌کند بنابراین از پیاده‌سازی این تکنیک صرف نظر کنید. اما درصوررتی که از byte در جاوا و تکنیک variable coding یا Gamma code استفاده کنید می‌توانید ۵ نمره اضافه کسب کنید. داده‌ساختارهای خود را به هردو صورت فشرده و ساده ذخیره کنید تا امکان مقایسه بین آنها وجود داشته باشد.

عملکرد نهایی این بخش می‌بایست به شکل زیر قابل مشاهده باشد:

۱) دریافت یک سند و نمایش تمام tokenهای آن بصورت ریشه‌یابی‌شده[۱۲]
۲) دریافت یک مجموعه سند و نمایش dictionary و posting list حاصل از آن
۳) دریافت یک سند و بروزرسانی dictionary و posting listها
۴) مقایسه حجم داده‌ساختار ذخیره شده قبل و بعد از فشرده‌سازی شاخص‌ها

[12] stem

بخش دوم (اصلاح پرسمان، جستجو و بازیابی اسناد)

در این بخش، سیستم شما یک پرسمان متنی از کاربر دریافت می‌کند. سیستم شما باید قابلیت اصلاح کلمات پرسمان ورودی را داشته باشد. در این حالت فرض کنید که کلمات موجود در اسناد، همگی درست و فاقد اشتباه املایی و تایپی هستند. سیستم شما باید قادر باشد تا با دریافت پرسمان ورودی اشتباهات املایی و تایپی را تشخیص داده و به صورت کارا مجموعه‌ای از کلمات درست را به کاربر پیشنهاد دهد. به منظور سهولت در پیاده‌سازی، اشتباهات را تنها به صورت مستقل از متن در نظر بگیرید، یعنی تنها حالاتی که کلمه‌ی پرسمان در دیکشنری موجود نیست را اصلاح کنید.

پس از دریافت پرسمان، سیستم براساس شباهت اسناد و پرسمان، سندها را بصورت رتبه‌بندی شده در خروجی می‌دهد. سیستم شما باید از مدل‌های بازیابی زیر در این بخش پشتیبانی کند. توجه کنید که مدل اول، بازیابی بولی و سایر مدل‌ها، بازیابی ترتیب‌دار هستند:

- بازیابی بولین به صورت and و به صورتی که کلمات پرسمان q همگی در سند d موجود باشند.
- تعداد کلمات از پرسمان q که در سند d وجود دارند.
- ضرب داخلی در فضای برداری tf-idf مدل lnn-ltn
- کسینوس زاویه بین q,d درفضای برداری tf-idf مدل lnc-ltc

واسط کاربری سیستم در این بخش باید امکانات زیر را فراهم کند:

۱) دریافت پرسمان و نوع مدل بازیابی از کاربر
۲) نمایش لیست اسناد مرتبط به ترتیب شباهت
۳) امکان انتخاب سند توسط کاربر و مشاهده محتویات آن

بخش سوم (ارزیابی سیستم)

در مجموعه اسناد موجود علاوه بر فایل اسناد، تعدادی پرسمان و نتیجه آنها در اختیار شما قرار گرفته است. این بخش می‌بایست مجموعه پرسمان‌ها و پاسخ‌های درست برای هر پرسمان را بعنوان ورودی دریافت کند و با مقایسه پاسخ سیستم به پرسمان‌ها و پاسخ‌های درست موجود معیارهای recall, precision, MAP را برای مدل‌های بازیابی مورد استفاده، محاسبه نموده و در خروجی نشان دهد.

سیستم بازیابی اطلاعات توزیع‌شده (هدوپ)

برای کسب اطلاعات بیشتر درباره هدوپ می‌توانید به کتاب Hadoop Definitive Guide (3rd edition) مراجعه کنید. در این پروژه بخش‌های یک و دو را بصورت توزیع‌شده پیاده‌سازی خواهد کرد. بعنوان طراحی انجام پروژه می‌توانید از طراحی زیر استفاده کنید:

بخش اول (شاخص‌گذاری)

- Map: یک سند را ورودی می‌گیرد و خروجی را بصورت (term, docID) تولید می‌کند.
- Reduce: (term, docID) خروجی فاز map را بعنوان ورودی دریافت می‌کند. دقت کنید که ورودی این فاز بصورت (term, list(doc)) خواهد بود. خروجی این فاز تجمیع اسناد برای هر واژه و یا به عبارتی دیگر، posting list خواهد بود.

بخش دوم (جستجو)

- Map: یک پرسمان را ورودی می‌گیرد و خروجی را بصورت (query, relevant-doc) برمی‌گرداند. در relevant-doc، شماره سند، آدرس محل ذخیره آن و میزان شباهت براساس مدل اولیه را درنظر بگیرید.
- Reduce: در این فاز، خروجی باید بصورت لیست مرتب‌شده اسناد مرتبط باشد. برای اینکار می‌توانید از تابع sort هدوپ استفاده کنید. به این منظور لازم است شیء relevant-doc را sortable پیاده‌سازی کنید.

برای اطمینان از درستی پیاده‌سازی سیستم برروی هدوپ می‌توانید سیستم خود را در حالت standalone اجرا کنید. تحویل نهایی پروژه در حالت pseudo-distributed انجام خواهد گرفت.

نکات پایانی

- برنامه ارسالی شما می‌بایست به زبان java باشد. پروژه باید به صورت انفرادی انجام شود.
- موارد تحویل‌دادنی را بصورت فایل فشرده در قالب zip و به آدرس sharif.fall93.mir@gmail.com ارسال کنید.
- موارد تحویل‌دادنی شامل کدها و گزارش مختصر می‌باشد. گزارش می‌بایست شامل توضیح طراحی پروژه، درصد کاهش حجم در فشرده‌سازی و نتایج ارزیابی باشد.
- نام فایل ارسالی و همچنین subject ایمیل ارسالی باید بصورت زیر باشد:

Project1-StudentNumber1

- موعد تحویل پروژه ساعت **۶:۰۰ صبح ۱۱ آبان‌ماه ۱۳۹۳** خواهد بود.
- به ازای **هر ساعت تاخیر یک درصد** از نمره پروژه شما کسر خواهد شد.
- سوالات خود را درمورد پروژه در گروه درس مطرح کنید.
- درصورت مشاهده تقلب، طبق قوانین دانشکده با شما برخورد خواهد شد.

موفق باشید ...