كلية الهندسةالمعلوماتية
جامعة دمشق
قسم هندسة البرمجيات ونظُم المعلومات

مشروع نظم استرجاع المعلومات
(IR)

إشراف المهندسة :

لين قويدر

إعداد الطلاب :

لجين شاوي
محمد العلبي
ياسمين العقلة

- **ال data sets المستخدمة :**

لدينا 2 data sets (CISI , CACM)

CACM عبارة عن مجموعة من ملخصات المقالات ، نُشرت في مجلة ACM بين عامي 1958 و 1979. غالبًا ما يُزعم أنها أصغر من ملاحظة أي تأثير حقيقي.

محتوياتها:

في cacm.all يحتوي على 3204 مدخلات معنونة.

يتم تمييز كل تسمية على وجه التحديد بامتداد متبوعًا بحرف تظهر في إدخال بالترتيب التالي:

(.I) معرف (رقم الملف)

(.T) العنوان

(.W) الملخص

(.B) تاريخ نشر المقال

(.A) قائمة المؤلفين

(.N) المعلومات عند إضافة الإدخال

(.X) قائمة المراجع الترافقية لمستندات أخرى

تحوي على ملف للاستعلامات يحوي على :

أربعة وستين (64) جملة استعلام تستخدم أيضًا نفس علامات النص.

تظهر بالترتيب التالي:

(.I) معرف

(.W) استعلام

(.A) قائمة المؤلفين

(.N) اسم المؤلفين وبعض الكلمات الأساسية حول ما يبحث عنه الاستعلام

تقييمات الصلة:

في qrels.text ، يتبع معرف الاستعلام عن طريق معرف المستند متبوعًا بـ 0 int و 0.0 float. يحتوي كل مستند على صف خاص به ويمكن استخدامه كمتجه قابل للتدريب.

مجموعة CISI تشبه إلى حد بعيد مجموعة CACM وتستخدم نفس الرموز.

يحتوي الملف CISI.ALL على 1460 نصًا.

وتحتوي على:

(.I) معرف (رقم الملف)
(.T) العنوان
(.W) الملخص
(.A) قائمة المؤلفين
الاستعلامات:
تحوي على 112 استعلام في ملف CISI.QRY تستخدم نفس رموز CACM
تقييمات الصلة:
في CISI.REL ، يتبع معرف الاستعلام معرف المستند متبوعًا بـ 0 int و 0.0 float.
يحتوي كل مستند على صف خاص به ويمكن استخدامه كمتجه قابل للتدريب.

حيث سنقوم بما يلي

- **خطوات المشروع :**

قمنا في هذا المشروع ببناء وتحقيق نظام استرجاع المعلومات قادر على :

A. تجذير الكلمات :

قمنا ببناء ملف يحوي تابع لقراءة من الملف ذي اللاحقة (ALL.) ياخذ الملف المطلوب ويرد سلسلة من العناوين وسلسلة من محتويات الCorpus
ثم بنينا ملف لمعالجة النص يحوي على التوابع التالية:

- تابع convert_lower_case لتحويل حروف النص للشكل الصغير.
- تابع remove_stop_words لازالة ال(stop word)
- تابع remove_punctuation لإزالة علامات الترقيم من سلسلة
- تابع remove_apostrophe لإزالة single quotation

- قمنا بعمل Tokenization للملفات عن طريق ملف Python قمنا ببنائه (Tokenization) يحوي على :

_ قمنا بعمل Tokenization (Tokenize): حيث استخدمنا مكتبة الـ Nltk لعمل ذلك .

_ تابع لتحويل الأرقام المكتوبة للشكل العددي لشكل كتابي (Convert_ Numbers).
تابع يقوم بتنفيذ ماسبق على التسلسل (Work).
ثم قمنا بعمل Stemming للكلمات وذلك عن طريق خوارزمية Porter في ملف Python قمنا ببنائه Porter باستخدام تابع (Stem)

B. الوصول للأفعال الشاذة :

قمنا بالوصول للأفعال الشاذة ومعالجتها وردها وردها الى أصلها عن طريق عمل Lemmatizatio للكلمات عن طريق ملف Python قمنا ببنائه Porter باستخدام تابع اسمه Stemming الذي يستعمل WordNetLimmatizer .

C. الوصول إلى التعابير التي تملك أشكال مختلفة :
قمنا بمعالجة حالة التواريخ والعديد من الحالات الخرى كالإيميلات والكلمات التي تحوي (_و-) والكلمات المختصرة كأسماء البلدان (U.S.و...) داخلها عن طريق ملف Pythonقمنا ببنائه (Regex) والذي يعتمد على مكتبة Re.

D. اقتراح تصحيح للكلمات :
قمنا ببناء ملف spellchecking يحوي على تابع يطابق الجملة كلمة كلمة ويصححها علن طريق مكتبة ال TextBlob

E . الوصول للأسماء باختلاف طرق التمثيل الكتابي لها :
قمنا بمعالجة هذه الحالة عن طريق عمل Soundex ضمن ملف Python اسمه Soundex استعمل مكتبة Fuzzy ونقوم بتطبيقه على الـ Query لمعرفة الكلمات التي لها نفس التمثيل الصوتي واضافتها ل Tokens الاستعلام .

F .بناء فهارس مناسبة للمحتوى المعالج :
لبناء الفهرس قمنا ببناء Vector Space Model .
قمنا بداية بحساب TF-IDF لكل Term و Document عن طريق ملف Python قمنا ببنائه TF-IDF حيث يحتوي على :
- Tf: وهو عبارة عن Python dict يقوم بربط الـ Term . documentمع قيمة الـ Tf الخاصة بهم .

(Term,document) : tf

Tf(term , document)= term _ frequent/word _ count_in_doc

- Idf :هو عبارة أيضاً عن Python dict يربط كل term مع idf الخاصة به.

Term : idf

(count _ document/ count _document_has_term+1)

- idf مع tf أيضا هو Python dict يربط كل Term ,document مع الـ الخاصة بهم .

(term,document): tf_idf

Tf_idf(term,document)= tf * idf

ثم قمنا باستخدام Tf_ idf ببناء الـ Vector Space Modelضمن ملف Pythonقمنا ببنائه (Vector.py).

M . بناء تابع مطابقة بين الاستعلام والمحتوى المعالج :

للمطابقة بينهم قمنا باستخدام cosine similarity حيث قمنا بعمل تابع ضمن ملف الـ بأخذ Vector .pyويرد قيمة الـ tow vectors بينهم حسب القانون :

$$\text{Cosine} = \frac{V1*V2}{|V1|*|V2|}$$

ويتم استدعاء هذا التابع ضمن تابع (getRelevanceFiles) الذي يقوم بعمل مطابقة بينquery vector و documents vectors المحسوبة مسبقاً .

N . بالنسبة للـ query:

قمنا بتطبيق نفس الخطوات السابقة التي تم تطبيقها على الملفات وعمل vector لها جلب الملفات المقاربة للاستعلام .

O. معيار الـ Area under curve:

قمنا بحساب معيار الـ AUC ضمن ملف AUC.py .

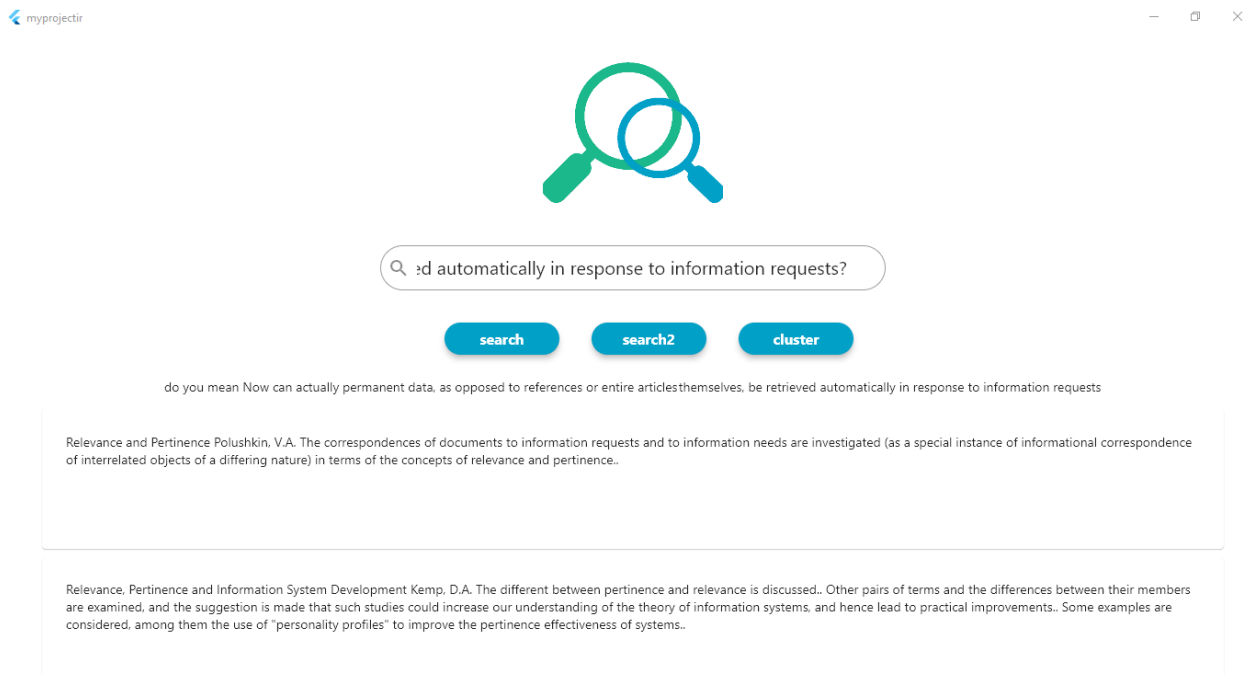
حيث يأخذ مصفوفة الملفات النتيجة وملفات الـ relevance التي قمنا بقرائتها من ملف الـ relevance.txt وحساب الـ precision,recall لكل استعلام عن طريق حساب

False position,true position,false negative,true negative

تم تطبيق cluster لمساعدة المستخدم اللذي لا يملك فكرة عن مايريده في البحث حيث انه استخدمنا معاني النصوص ووضعنا عنوان كل نص للدلالة عليه

- **توصيف لبنية النظامsystem architecture :**

قمنا ببناء واجهة فلاتر الآتية:



وايضاً قمنا ببناء برنامج بايثون واستخدمنا للAPI الFlask
حيث يوجد لدينا العديد من ال API’s
الاول لجلب ليست من النتائج الخاص بCISI ومعه تصحيح الاستعلام
الثاني لجلب ليست من النتائج الخاصة بال CACM ومعه ايضاً تصحيح للاستعلام
الثالث لجلب ال cluster

- **النتائج والتقييم على ال testing data :**

(حنحط هون صور للنتائج وللكود المطبق لحساب القيم المطلوبة يعني تحت كل وحدة بنحطهن)

- سنطبق مجموعة من المقاييس على نظامنا لنعلم مدى صحة نتائج هذا النظام ومدى مطابقتها للنتائج الصحيحة التي ستظهر عند الاستعلامات

1- Precision

هو نسبة النتائج ال relevant التي قام النظام بإرجاعها مقسوم على جميع النتائج التي جلبها النظام ويعطى بالقانون

Precision(q)=| Rq^ Aq|/ Aq

وكلما كانت هذه القيمة أكبر كان النظام أفضل .

2- Recall

هو نسبة عدد النتائج ال relevant التي استطاع النظام جلبها بالنسبة لعدد النتائج التي كان يتوجب على النظام إرجاعها ويطعى بالقانون

Recall= | Rq^ Aq|/ Rq

3- Precision@10

لحساب P@10 : ننظر إلى الوثائق ال relevant التي قام النظام باستردادها وننظر إلى عدد الوثائق التي استرجعها النظام

التنفيذ لهم:

```
Average Precision is :  32.833333333333336
Precision is :  0.6666666666666666
Average Recall is :  9.146091506073672
Recall is :  0.6666666666666666
F-score is :  14.306850189819203
Accuracy :  73.82465753424653
```

4- Mean Average Precision (MAP)

هو المتوسط الحسابي لمجاميع ال Ap

التنفيذ :

MAP=0.4619271392767632

5- Mean Reciprocal Rank (MRR)

- **تقسيم العمل بين أعضاء المجموعة**
  حيث تم تسليم الفلاتر والتقرير للزميلة ياسمين
  وتم تسليم الطلب الاضافي محمد
  وتم تسليم معالجة النصوص وبقية الطلبات والربط للزميلة لجين

- **المصادر**
- **https://medium.com/kuzok/news-documents-clustering-using-python-latent-semantic-analysis-b95c7b68861c**
- **https://www.pragmalingu.de/docs/guides/data-comparison**
- **https://anvil.works/blog/how-to-build-a-search-engine**
- **https://blog.dominodatalab.com/getting-started-with-k-means-clustering-in-python**
- **https://towardsdatascience.com/clustering-documents-with-python-97314ad6a78d**
- **https://www.kaggle.com/code/taranjeet03/search-clustering/notebook**
- **https://www.kaggle.com/code/vabatista/introduction-to-information-retrieval/notebook**

- **https://livebook.manning.com/book/essential-natural-language-processing/chapter-3/v-3/60**

THE END