MODUL 6: RELEVANCE FEEDBACK AND QUERY EXPANSION

6.1 Deskripsi Singkat

Dalam koleksi dokumen, suatu konsep yang sama dapat direpresentasikan dengan kata-kata yang berbeda, atau dikenal dengan synonymy. Biasanya pengguna mengatasi permasalahan tersebut dengan memperbaiki query yang diketikkan secara manual ketika mendapatkan hasil yang belum sesuai dengan kebutuhan informasinya.

Terdapat beberapa metode yang dapat digunakan untuk mengotomatisasi proses perbaikan query tersebut, diantaranya:

1. Relevance feedback dengan Rocchio Algorithm
2. Query expansion dengan thesaurus

6.2 Tujuan Praktikum

1. Dapat memperbaiki hasil perangkingan sistem information retrieval secara otomatis dengan menggunakan relevance feedback dan query expansion.

6.3 Material Praktikum

Tidak ada

6.4 Kegiatan Praktikum

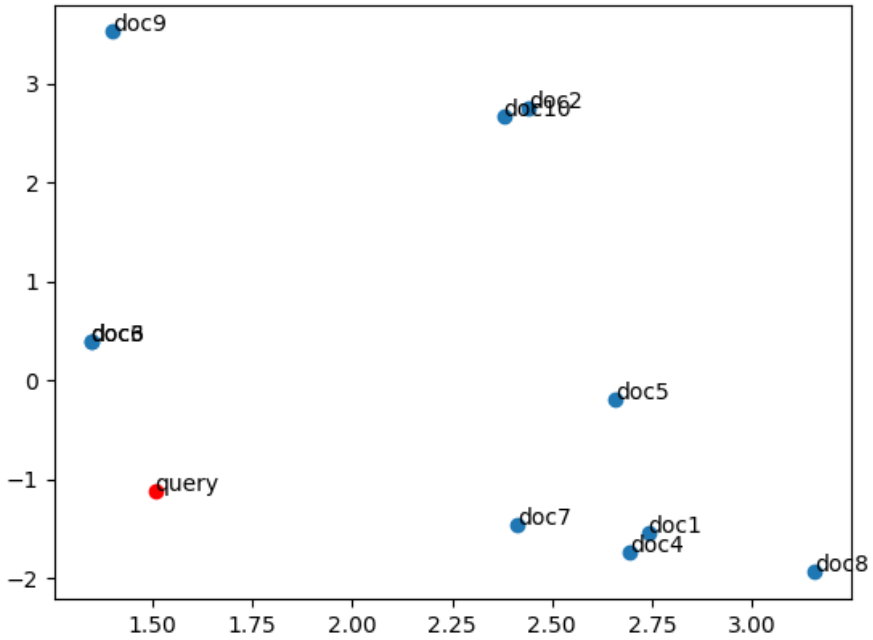
A. Relevance Feedback

Untuk mendapatkan gambaran mengenai kedekatan antar dokumen dan query, mari visualisasikan terlebih dahulu dalam suatu scatter plot. Gunakan matriks TD dan TQ pada praktikum modul 5 untuk divisualisasikan sebagai berikut.

```
from sklearn.decomposition import TruncatedSVD
import matplotlib.pyplot as plt

DT = TD.transpose()
print(DT)
model = TruncatedSVD(n_components=2, random_state=7).fit(DT)
DT_reduced = model.transform(DT)
QT_reduced = model.transform(TQ.transpose())
print(QT_reduced)
print(DT_reduced)
plt.scatter(DT_reduced[:, 0], DT_reduced[:, 1])
plt.scatter(QT_reduced[:, 0], QT_reduced[:, 1], color=["red"])
labels=list(doc_dict.keys())
for i, txt in enumerate(labels):
      plt.annotate(txt, (DT_reduced[i, 0], DT_reduced[i, 1]))
plt.annotate("query", (QT_reduced[0, 0], QT_reduced[0, 1]))
plt.show()
```

Pada kode di atas, TD dan TQ ditranspose terlebih dahulu untuk mendapatkan representasi vektor setiap dokumen dan query. Kemudian, digunakan SVD untuk mereduksi dimensi dari vektor setiap dokumen agar dapat ditampilkan dalam bentuk 2 dimensi. Scatter plot yang ditampilkan yaitu sebagai berikut.



Pada praktikum modul 5, Anda telah membuat fungsi utuk mengembalikan top k dokumen untuk query tertentu. Jalankan kembali fungsi tersebut untuk mendapatkan 5 dokumen teratas untuk query ”sistem informasi statistik”.

```
top_5 = exact_top_k(doc_dict, TD, TQ[:, 0], 5)
print(top_5)
```

Selanjutnya, Anda akan menggunakan informasi dari tabel relevance judgment pada praktikum modul 6 sebagai data relevance feedback dari pengguna. Pada prakteknya, hanya sebagian dokumen saja yang terlabeli dan dapat digunakan sebagai feedback. Dokumen yang relevan untuk query ”sistem informasi statistik” yaitu dokumen dengan id doc_1, doc_4, doc_5, doc_7, doc_8. Sedangkan dokumen yang tidak relevan yaitu doc_2, doc_3, doc_6, doc_9, dan doc_10.

```
rel_vecs_id = ["doc1", "doc4", "doc5", "doc7", "doc8"]
nrel_vecs_id = ["doc2", "doc3", "doc6", "doc9", "doc10"]

rel_vecs = []
for doc in rel_vecs_id:
      rel_vecs.append(DT[doc_ids.index(doc),:])

nrel_vecs = []
for doc in nrel_vecs_id:
      nrel_vecs.append(DT[doc_ids.index(doc),:])
```

Kemudian, informasi ini digunakan dalam Rocchio Algorithm untuk memperbaiki vektor query. Gunakan matriks Term-Query dari query ”sistem informasi statistik” pada modul 5 sebagai vektor awal. Gunakan alpha = 1, beta = 0.75, dan gamma = 0.15

```
query_vecs = TQ.transpose()
alpha = 1
beta = 0.75
gamma = 0.15

# Update query vectors with Rocchio algorithm
query_vecs = alpha * query_vecs + beta * np.mean(rel_vecs, axis=0) -
gamma * np.mean(nrel_vecs, axis=0)
query_vecs[query_vecs<0] = 0 #negative value => 0
```

Gunakan vektor query yang telah diperbaiki tersebut untuk menghitung kembali skor kemiripan antara query dan dokumen, dan mengembalikan 5 dokumen teratas.

```
top_5 = exact_top_k(doc_dict, TD, query_vecs[0, :].transpose(), 5)
print(top_5)
```

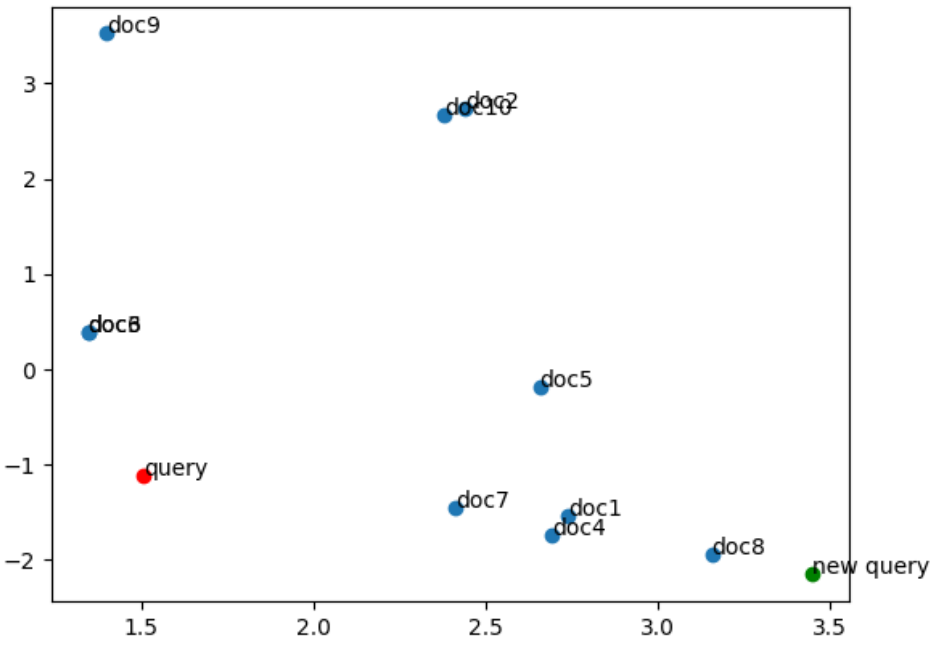
Bandingkan 5 dokumen teratas tersebut dengan hasil sebelumnya. Apakah ada perbedaan? Apa yang dapat Anda simpulkan?

Anda juga dapat menambahkan vektor query yang telah diperbaiki tersebut untuk digambarkan dalam scatter plot.

```
QT1_reduced = model.transform(query_vecs)

plt.scatter(DT_reduced[:, 0], DT_reduced[:, 1])
plt.scatter(QT_reduced[:, 0], QT_reduced[:, 1], color=["red"])
plt.scatter(QT1_reduced[:, 0], QT1_reduced[:, 1], color=["green"])
doc_ids=list(doc_dict.keys())
for i, txt in enumerate(doc_ids):
      plt.annotate(txt, (DT_reduced[i, 0], DT_reduced[i, 1]))
plt.annotate("query", (QT_reduced[0, 0], QT_reduced[0, 1]))
plt.annotate("new query", (QT1_reduced[:, 0], QT1_reduced[:, 1]))
plt.show()
```

Perhatikan posisi vektor query yang baru pada scatter plot tersebut.



B. Query Expansion dengan Thesaurus

Salah satu Thesaurus yang digunakan untuk memperluas query yaitu Wordnet. Wordnet untuk bahasa Inggris tersedia di library NLTK. Untuk menggunakannya, install terlebih dahulu NLTK dan download wordnet dengan kode berikut.

```
import nltk
nltk.download('wordnet')
nltk.download('omw-1.4')
```

Tulis kode berikut untuk memperluas query ”information system” dengan sinonimnya dari wordnet bahasa Inggris.

```
from itertools import chain
from nltk.corpus import wordnet
query = "information system"

expand_list = []
for term in query.split(" "):
      synonyms = wordnet.synsets(term)
      lemmas = set(chain.from_iterable([word.lemma_names() for word in
synonyms]))
      print(lemmas)
      expand_list = expand_list + list(lemmas)
print(expand_list)

query_expand = query + " " + (" ".join(expand_list)).replace("_", " ")
print(query_expand)
```

6.5 Penugasan

1. Tuliskan laporan praktikum yang merangkum kegiatan praktikum yang telah Anda lakukan pada kegiatan 6.4.
2. Buat kode untuk mengimplementasikan relevance feedback pada folder ”berita” dengan query ”vaksin corona jakarta” yang telah dikerjakan pada modul 5. Bandingkan top 5 dokumen yang dikembalikan sebelum dan sesudah dilakukan relevance feedback. Jelaskan kode tersebut dalam laporan praktikum.
   Note: Dokumen yang relevan yaitu berita2 dan berita3.