MODUL 2: TERM-VOCABULARY DAN TEXT PREPROCESSING

2.1 Deskripsi Singkat

Term adalah sebutan untuk unit yang diindeks, biasanya berupa kata. Dokumen adalah sebutan untuk unit yang dikembalikan sebagai output dari suatu *query*, sedangkan sekumpulan dokumen tersebut disebut dengan *Collection* atau *Corpus*. Sekumpulan *term* unik dalam suatu *corpus* dikenal dengan *Vocabulary*.

Terdapat beberapa struktur data pada Python yang sering digunakan dalam information retrieval, diantaranya *List* dan *Dictionary*. List sama seperti array pada bahasa pemrograman lain. Namun, anggota List pada python tidak selalu harus bertipe data sama. Sedangkan Dictionary sama seperti map, yang setiap elemennya terdiri dari sepasang key dan value.

Text preprocessing adalah suatu proses pengubahan bentuk data yang belum terstruktur menjadi data yang terstruktur sesuai kebutuhan. Beberapa teknik *text preprocessing* diantaranya tokenisasi, *capitalization/case-folding*, eliminasi s*topword*, normalisasi, dan *stemming*.

1. Tokenisasi
   Tokenisasi adalah proses memotong suatu text menjadi bagian-bagian yang disebut token.
2. *Capitalization*/*Case-folding*
   Capitalization adalah proses mengubah teks menjadi seluruhnya huruf kapital (*uppercase*), sedangkan *case-folding* adalah proses mengubah teks menjadi seluruhnya huruf kecil (*lowercase*). Python merupakan bahasa pemrograman *case sensitive* yang membedakan huruf besar dan huruf kecil, sehingga sering membutuhkan *capitalization/case-folding*.
3. Eliminasi Stopword
   Stopword merupakan kata-kata yang dianggap hanya sedikit berarti dalam pemrosesan teks, sehingga dapat dieliminasi. Setiap bahasa memiliki daftar *stopword* yang berbeda. Strategi lain penentuan *stopword* adalah dengan menghitung frekuensi kemunculan kata dan menganggap kata-kata yang jarang muncul dalam dokumen sebagai *stopword* yang dapat dieliminasi.
4. Normalisasi
   Normalisasi adalah mengelompokkan token yang penulisan hurufnya berbeda tetapi termasuk ke kata yang sama menjadi satu *term* yang sama. Contohnya yaitu kata anti vaksin dapat dituliskan dengan beberapa variasi seperti berikut, antivaksin : {anti vaksin, antivaksin, anti-vaksin}. Selain itu, normalisasi dapat juga digunakan untuk

mengelompokkan bahasa gaul ke bahasa bakunya, contohnya, saya: {gue, gua, aku, aq, sy}

5. *Stemming*

   Stemming adalah mengubah kata-kata berimbuhan ke bentuk dasarnya.

2.2 Tujuan Praktikum

Setelah praktikum pada modul 2 ini diharapkan mahasiswa mempunyai kompetensi sebagai berikut.

1) Dapat memahami konsep dasar term, document, vocabulary, corpus, dan index.
2) Dapat memahami struktur data pada python yang biasa digunakan sistem information retrieval.
3) Dapat mengimplementasikan term, document, vocabulary, dan corpus menggunakan python.
4) Dapat mengimplementasikan teknik preprocessing teks.

2.3 Material Praktikum

Tidak ada.

2.4 Kegiatan Praktikum

A. Struktur Data List dan Dictionary pada Python

Suatu list ditandai dengan penggunaan "[]". Buat beberapa list berikut, lalu tampilkan elemennya dengan mengakses indeks listnya.

```
list1 = ["Wilayah", "Kamu", "Sudah", "Bebas", "COVID-19", "?"]
print(list1[1])

list2 = [1, 2, "tiga", "empat", 5]

#menampilkan semua anggota list
for i in range(len(list2)):
      print(list[i])

#atau
for list in list2:
      print(list)
```

Sedangkan suatu *dictionary* ditandai dengan penggunaan "{}". Buat beberapa *dictionary* berikut, lalu tampilkan elemennya dengan mengakses *key*-nya.

```
dict1 = {1: "Wilayah", 2: "Kamu", 3: "Sudah", 4: "Bebas", 5:
"COVID-19", 6: "?"}
print(dict[1])

list2 = {"januari": 1, "februari": 2, "maret": 3, "april": 4,
"mei": 5}
for key in list2:
      print(dict2[key])
```

B. Implementasi Struktur Data Python untuk Information Retrieval

Misalnya terdapat tiga dokumen pada *corpus,* yang setiap *term* pada dokumennya dipisahkan oleh spasi. Buat list *term* untuk setiap dokumen, kemudian gabungkan list dokumen ke dalam corpus dan seluruh term ke corpus_term.

```
doc1 = "pengembangan sistem informasi penjadwalan"
doc1_term = ["pengembangan", "sistem", "informasi", "penjadwalan"]
doc2 = "pengembangan model analisis sentimen berita"
doc2_term = ["pengembangan", "model", "analisis", "sentimen",
"berita"]
doc3 = "pengembangan analisis sistem input output"
doc3_term = ["pengembangan", "analisis", "sistem", "input",
"output"]

corpus = [doc1, doc2, doc3]
corpus_term = [doc1_term, doc2_term, doc3_term]
```

Gabungkan semua *term* ke dalam *vocabulary*. Term yang ada dalam *vocabulary* harus unik. Dan hitung berapa kali *term* tersebut disebutkan dalam dokumen. Gunakan *dictionary* untuk struktur data *vocabulary*, dengan *term* sebagai *key* dan frekuensi kemunculannya sebagai value.

```
vocabulary = {}

for d in corpus_term:
      for term in d:
            if term not in vocabulary:
                  vocabulary[term] = 1
            else:
                  vocabulary[term] = vocabulary[term]+1
print(vocabulary)
```

Output yang ditampilkan adalah anggota dari *vocabulary* yang terdiri dari list *term* beserta frekuensi kemunculannya.

```
{'pengembangan': 3, 'sistem': 2, 'informasi': 1, 'penjadwalan': 1,
'model': 1, 'analisis': 2, 'sentimen': 1, 'berita': 1, 'input': 1,
'output': 1}
```

C. Tokenisasi

Buat fungsi tokenisasi dengan memotong suatu teks secara otomatis berdasarkan spasi.

```
def tokenisasi(text):
      tokens = text.split(" ")
      return tokens
```

Kemudian panggil fungsi tersebut untuk memotong tiga dokumen pada latihan 2B menjadi sekumpulan token.

```
doc1 = "pengembangan sistem informasi penjadwalan"
doc2 = "pengembangan model analisis sentimen berita"
doc3 = "pengembangan model analisis sentimen berita"
corpus = [doc1, doc2, doc3]

for d in corpus:
      token_kata = tokenisasi(d)
      print(token_kata)
```

maka output yang akan dihasilkan adalah daftar token dari setiap dokumen di dalam list *corpus*

```
['pengembangan', 'sistem', 'informasi', 'penjadwalan']
['pengembangan', 'model', 'analisis', 'sentimen', 'berita']
['pengembangan', 'analisis', 'sistem', 'input', 'output']
```

Fungsi tokenisasi juga terdapat pada beberapa *library text processing* seperti **nltk** dan **spacy**. Untuk menggunakan *library* tersebut, *install library* yang dibutuhkan dengan perintah:

```
pip install spacy
```

Kemudian import library tersebut, dan panggil fungsi tokenisasinya. Spacy memiliki *library* pemrosesan teks khusus untuk bahasa Indonesia. Setelah menjalankan kodingan seperti blok kode di bawah, akan dihasilkan output yang sama dengan tokenisasi dokumen secara manual (pada bagian awal).

```
from spacy.lang.id import Indonesian
import spacy
nlp = Indonesian()  # use directly
nlp = spacy.blank('id')  # blank instance'
# Bisa pilih salah satu bentuk variabel nlp di atas, atau boleh
juga menggunakan keduanya pada blok program berbeda

for d in corpus:
      spacy_id = nlp(d)
      token_kata = [token.text for token in spacy_id]
      print(token_kata)
```

D. *Capitalization/Case-Folding*

Ubah semua huruf pada dokumen menjadi huruf besar dengan fungsi upper() dan menjadi huruf kecil dengan fungsi lower().

```
text = "Wilayah Kamu Sudah 'Bebas' COVID-19? Cek 34 Kab/Kota Zona
Hijau Terbaru"
text_capital = text.upper()
text_lower = text.lower()
```

E. Eliminasi Stopword

Misalnya list `stopwords` berisi daftar kata-kata yang termasuk dalam stopword. Lakukan eliminasi stopword untuk suatu teks dengan kode berikut.

```
stopwords = ["yang", "dari", "sudah", "dan"]
text = "Wilayah Kamu Sudah 'Bebas' COVID-19? Cek 34 Kab/Kota Zona
Hijau Terbaru"
tokens = ["wilayah", "kamu", "sudah", "bebas", "covid-19", "?",
"cek", "34", "kab", "/", "kota", "zona", "hijau", "terbaru"]
tokens_nostopword = [w for w in tokens if not w in stopwords]
print(tokens_nostopword)
```

Selain itu, sudah ada beberapa *library* yang menyediakan list stopword untuk bahasa Indonesia, diantaranya spacy.

```
from spacy.lang.id import Indonesian
import spacy
nlp = Indonesian()  # use directly
nlp = spacy.blank('id')  # blank instance'
#Bisa menggunakan salah satu bentuk variabel nlp di atas, atau bisa
menggunakan keduanya pada blok program berbeda

stopwords = nlp.Defaults.stop_words
tokens_nostopword = [w for w in tokens if not w in stopwords]
print(tokens_nostopword)
```

F. Normalisasi

Misalkan terdapat *dictionary* yang berisi daftar kata yang perlu dinormalisasi. Berikut kode yang digunakan untuk mengganti kata-kata tersebut ke bentuk normalnya.

```
normal_list = {"gue": "saya", "gua": "saya", "aku": "saya", "aq":
"saya", "lagi": "sedang"}
text = "aq lagi di jalan nih"
text_normal = []
for t in text.split(" "):
    text_normal.append(normal_list[t] if t in normal_list else t)
print(text_normal)
```

G. Stemming

Salah satu *library* bahasa Indonesia yang menyediakan fungsi stemming bahasa Indonesia yaitu sastrawi. Lakukan instalasi *library* sastrawi terlebih dahulu dengan menggunakan pip.

```
pip install sastrawi
```

Lalu buat kode python berikut.

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
# create stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()
# stemming process
text = "Wilayah Kamu Sudah 'Bebas' COVID-19? Cek 34 Kab/Kota
Zona Hijau Terbaru"
output   = stemmer.stem(text)
print(output)
```

2.5 Penugasan

1. Diketahui suatu dokumen berikut terdiri dari beberapa paragraf dan setiap paragraf terdiri dari beberapa kalimat. Paragraf yang berbeda dipisahkan dengan Enter, sedangkan kalimat dipisahkan dengan titik, tanda tanya, atau tanda seru. Buat kode fungsi python untuk memisahkan dokumen sehingga menghasilkan variabel list_paragraf (nama fungsi: `paragraph_parsing`), dan masing-masing paragraf menjadi variabel list_kalimat (nama fungsi: `sentence_parsing`). Contoh input dokumen:

```
Mobilitas warga bakal diperketat melalui penerapan
PPKM level 3 se-Indonesia di masa libur Natal dan
tahun baru (Nataru). Rencana kebijakan itu dikritik
oleh Epidemiolog dari Griffith University Dicky
Budiman.
Dicky menyebut pembatasan mobilitas memang akan
memiliki dampak dalam mencegah penularan COVID-19.
Tapi, kata dia, dampaknya signifikan atau tidak akan
bergantung pada konsistensi yang mendasar yakni
testing, tracing, treatment (3T) hingga vaksinasi
COVID-19.
```

Output yang diharapkan tercetak setelah memanggil `paragraph_parsing`:

```
List paragraf:

p1: Mobilitas warga bakal diperketat melalui penerapan PPKM
level 3 se-Indonesia di masa libur Natal dan tahun baru
(Nataru). Rencana kebijakan itu dikritik oleh Epidemiolog dari
Griffith University Dicky Budiman.

p2: Dicky menyebut pembatasan mobilitas memang akan memiliki
dampak dalam mencegah penularan COVID-19. Tapi, kata dia,
dampaknya signifikan atau tidak akan bergantung pada konsistensi
yang mendasar yakni testing, tracing, treatment (3T) hingga
vaksinasi COVID-19.
```

Output yang diharapkan tercetak setelah memanggil `sentence_parsing` untuk paragraf 1:

```
List kalimat pada paragraf 1:

s1: Mobilitas warga bakal diperketat melalui penerapan PPKM
level 3 se-Indonesia di masa libur Natal dan tahun baru
(Nataru).

s2: Rencana kebijakan itu dikritik oleh Epidemiolog dari
Griffith University Dicky Budiman.
```

2. Lakukan *case-folding* (*upper case* dan *lower case*), tokenisasi, eliminasi stopword dan *stemming* pada dokumen di folder “berita” menggunakan *library* yang sudah tersedia (nltk, spacy, sastrawi, etc).