Text Preprocessing

Topic 4

Dwi Intan Af'idah, S.T., M.Kom

Text Preprocessing

- Data teks merupakan data yang tidak beraturan karena terdapat perulangan kata dan munculnya banyak kata yang tidak berkontribusi pada analisis data.
- Prapengolahan teks atau preprocessing perlu dilakukan untuk membersihkan data teks terlebih dahulu sebelum dilakukan proses analisis sentimen.
- Prapengolahan teks menghilangkan data yang tidak konsisten, data yang duplikat, dan data yang tidak berpengaruh terhadap polaritas suatu dokumen.

Steps of Text Preprocessing

01 Case Folding

02 Filtering

03 Tokenization

04 Slangword Convertion

05 Stopword Removal

06 Stemming

07 Lemmatization

08 Padding

Case Folding

- Case folding merupakan proses dalam text preprocessing yang dilakukan untuk menyeragamkan karakter pada data.
- Proses case folding adalah proses mengubah seluruh huruf menjadi huruf kecil.
- Pada proses ini karakter-karakter 'A'-'Z' yang terdapat pada data diubah kedalam karakter 'a'-'z'.

Kampus Merdeka INDONESIA JAYA

Poltek Harber
POLITEKNIK HARAPAN BERSAMA

Coding of Case Folding

3.1. Casefolding

```
[ ]  1 #mengambil data ulasan
     2 data_content = data_dataset ['ulasan']
     3 print(data_content[:4])
```

```
0    Dalam beberapa Bulan terakhir saya sempat ke s...
1    Salah satu tempat yang harus dikunjungi saat m...
2    Recommended untuk yg ingin melihat keindahan o...
3    Air terjun Sekumpul Sekarang menjadi lebih bai...
Name: ulasan, dtype: object
```

```
[ ]  1 #casefolding
     2 data_casefolding = data_content.str.lower()
     3 data_casefolding.head()
```

```
0    dalam beberapa bulan terakhir saya sempat ke s...
1    salah satu tempat yang harus dikunjungi saat m...
2    recommended untuk yg ingin melihat keindahan o...
3    air terjun sekumpul sekarang menjadi lebih bai...
4    air terjun, pemandangan bagus dan penduduk yan...
Name: ulasan, dtype: object
```

Filtering

- Filtering merupakan langkah untuk menghilangkan karakter-karakter ilegal pada dokumen seperti tanda baca, simbol, angka, html, dan mention.
- Proses dalam menghilangkan karakter-karakter illegal dapat disebut filtering.
- Contoh karakter ilegal yang dihilangakan antara lain %, &, >, (, {,], 1-9, @uluwatu, http://tripadvisor.com.

Kampus Merdeka INDONESIA JAYA

Poltek Harber
POLITEKNIK HARAPAN BERSAMA

Coding of Filtering

3.2. Filtering

```
#filtering

#url
filtering_url = [re.sub(r'''(?i)\b((?:https?://|www\d{0,3}[.]|[a-z0-9.\-]+[.][a-z]{2,4}/)(?:[^\s()<>]+|
\(([^\s()<>]+|(\([^\s()<>]+\)))*\))+(?:\(([^\s()<>]+|(\([^\s()<>]+\)))*\)|[^\s`!()\[\]{};:'".,<>?«»“”‘’]))''',
 " ", tweet) for tweet in data_casefolding]
#cont
filtering_cont = [re.sub(r'\(cont\)'," ", tweet)for tweet in filtering_url]
#punctuatuion
filtering_punctuation = [re.sub('[!"”#$%&’()*+,-./:;<=>?@[\]^_`{|}~]', ' ', tweet) for tweet in filtering_cont]
#hapus simbol'[!#?,.:";@()-_/\']'
#  hapus #tagger
filtering_tagger = [re.sub(r'#([^\s]+)', '', tweet) for tweet in filtering_punctuation]
#numeric
filtering_numeric = [re.sub(r'\d+', ' ', tweet) for tweet in filtering_tagger]

# # filtering RT , @ dan #
# fungsi_clen_rt = lambda x: re.compile('\#').sub('', re.compile('rt @').sub('@', x, count=1).strip())
# clean = [fungsi_clen_rt for tweet in filtering_numeric]

data_filtering = pd.Series(filtering_numeric)

```

Tokenization

- Tokenisasi merupakan proses untuk memecah dokumen teks menjadi token.
- Tokenisasi memiliki kemampuan untuk memecah dokumen menjadi kata, frasa, simbol atau elemen lain yang memiliki makna.
- Pada proses tokenisasi, data teks ulasan dipecah menjadi token-token yang terdiri dari satu kata yang bermakna dan disimpan dalam array kata.

Coding of Tokenization

3.3. Tokenization

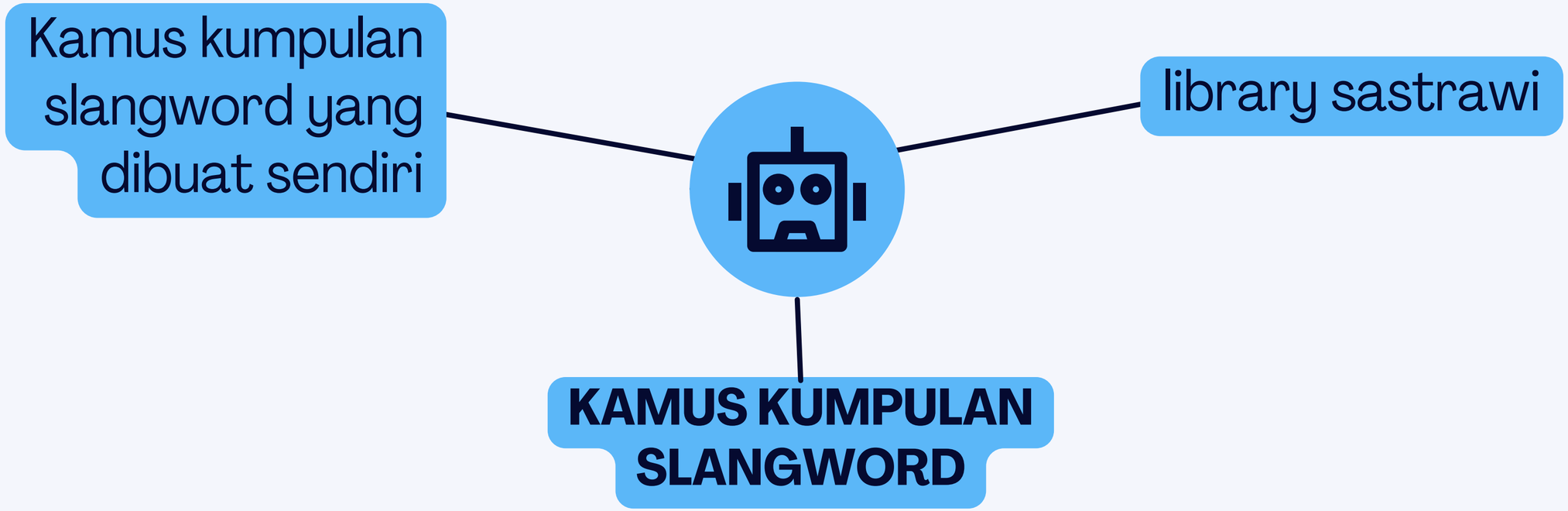
```
[ ]  1 #tokenization
     2
     3 data_tokens = [word_tokenize(line) for line in data_filtering]
     4 print(data_tokens)
```

```
[['dalam', 'beberapa', 'bulan', 'terakhir', 'saya', 'sempat', 'ke', 'sana', 'kali', 'sebelum', 'dan',
```

Slangword Convertion

- Dalam bahasa Indonesia, orang sering menulis kata yang tidak baku (slang word) daripada kata baku, seperti “cepat” dan bukan “cepat”.
- Jika hal ini terjadi, komputer akan mengartikan “cepet” dan “cepat” sebagai dua kata yang berbeda, padahal memiliki arti yang sama.
- Untuk mengatasinya, kita harus mengganti semua kata slang ke dalam bentuk standarnya dengan proses konversi slang word to standar word.

Kamus Slangword



Coding of Slang Word Convertion

3.4. Konversi Slangword

```
[ ]
#slang word
path_dataslang = open("kamus kata baku 1.csv")
dataslang = pd.read_csv(path_dataslang, encoding = 'utf-8', header=None, sep=";")

def replaceSlang(word):
  if word in list(dataslang[0]):
    indexslang = list(dataslang[0]).index(word)
    return dataslang[1][indexslang]
  else:
    return word

data_formal = []
for data in data_tokens:
  data_clean = [replaceSlang(word) for word in data]
  data_formal.append(data_clean)
len_data_formal = len(data_formal)
print(data_formal)
len_data_formal
```

```
[['dalam', 'beberapa', 'bulan', 'terakhir', 'saya', 'sempat', 'ke', 'sana', 'kali', 'sebelum', 'dan', 'sesudah', 'lahar'
41
```

Stopword Removal

- Stopword Removal merupakan tahap pengambilan kata-kata penting dan membuang kata-kata yang dianggap tidak penting.
- Cara untuk membuang kata yang tidak penting disebut stopword removal.
- Stopword removal bertujuan untuk menghilangkan kata-kata yang sering muncul namun tidak memiliki kontribusi dalam proses analisis data.
- Stopword removal berusaha memperkecil dimensi data dan mempercepat waktu komputasi (Symeonidis dkk., 2018).
- Contoh kata yang tidak penting di bahasa Indonesia seperti kata “dan”, ”yang”, ”di”, ”ke”.

Coding of Stopword Removal (1)

3.5. Stopword Removal

```
[ ]  1 nltk.download('stopwords')
     2 default_stop_words = nltk.corpus.stopwords.words('indonesian')
     3 stopwords = set(default_stop_words)
     4
     5 def removeStopWords(line, stopwords):
     6   words = []
     7   for word in line:
     8     word=str(word)
     9     word = word.strip()
    10     if word not in stopwords and word != "" and word != "&":
    11       words.append(word)
    12
    13   return words
    14 data_notstopword = [removeStopWords(line,stopwords) for line in data_formal]
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Coding of Stopword Removal (2)

```
[ ]  1 print(pd.DataFrame(default_stop_words)[:10])
```

```
          0
0       ada
1    adalah
2    adanya
3    adapun
4      agak
5   agaknya
6      agar
7      akan
8   akankah
9     akhir
```

```
[ ]  1 len(data_notstopword)
```

```
41
```

```
[ ]  1 data_notstopword
```

```
 'jorok',
 'kecewa',
 'staff',
 'nya'],
['mengunjungi',
```

Stemming

- Stemming merupakan proses memetakan variasi kata ke bentuk dasar.
- Proses stemming dilakukan dengan menghapus imbuhan, baik awalan maupun akhiran dari suatu kata untuk mendapatkan kata dasarnya.
- Stemming yang umum digunakan pada teks berbahasa Indonesia menggunakan library stemmer Sastrawi yang dikembangkan berdasarkan algoritma Nazief-Adriani.

Coding of Stemming

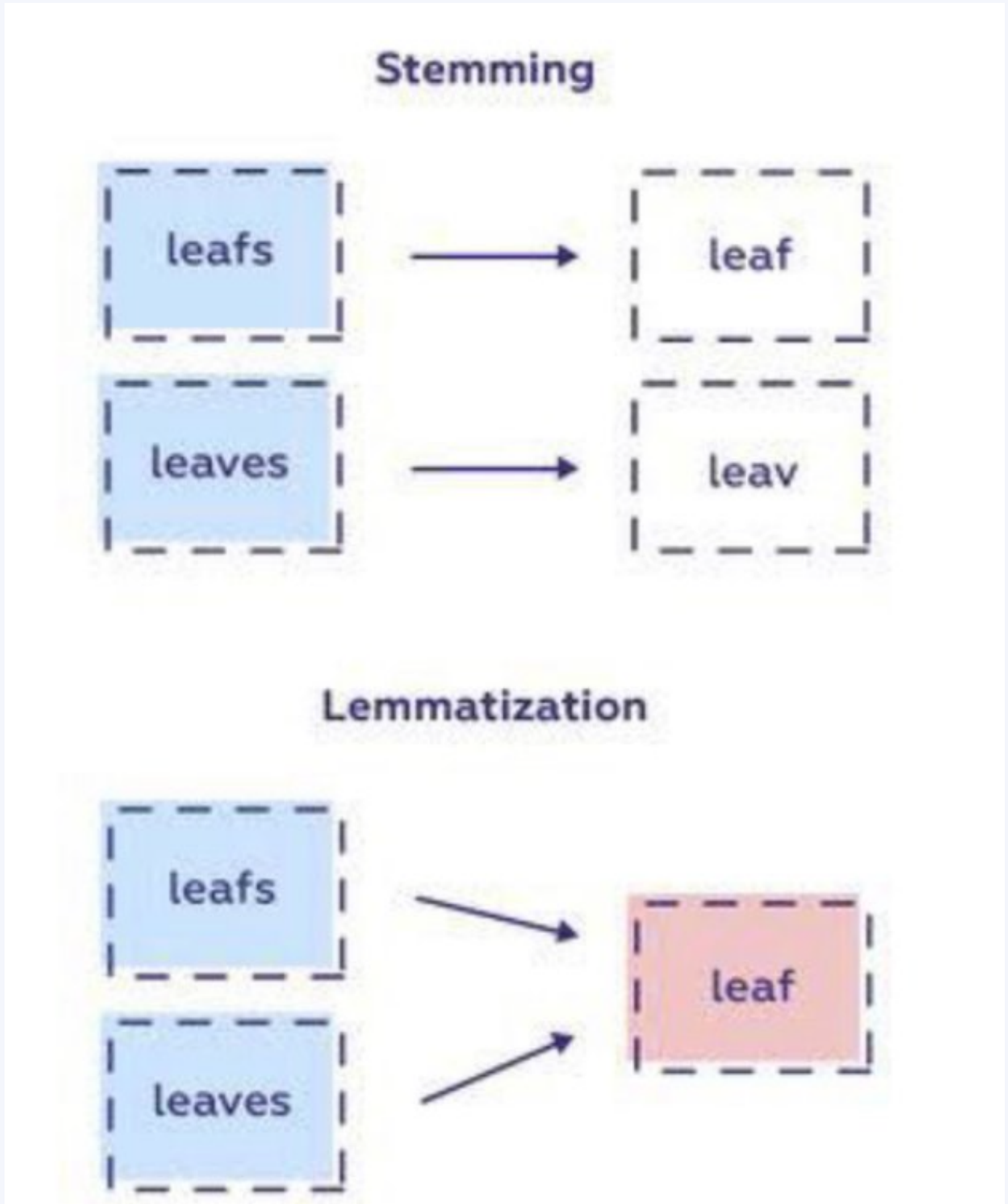
3.6. Stemming

```
[ ]  1 white_list = ["bali"] #ini perlu/tidak perlu diubah karena dianggap sastr
     2
     3 factory = StemmerFactory()
     4 ind_stemmer = factory.create_stemmer()
     5 def stemmer(line):
     6     temp = list()
     7     for word in line:
     8       if(word not in white_list):
     9         word = ind_stemmer.stem(word)
    10       if(len(word)>3):
    11         temp.append(word)
    12     return temp
    13
    14 reviews = [stemmer (line) for line in data_notstopword]
    15 print(reviews)
```

```
[['kali', 'lahar', 'dingin', 'pandang', 'kesan', 'pandang', 'desa', 'sidemang
```

Lemmatization

Di dalam suatu kata tentunya pasti mempunyai variasi misalnya saja kata yang mempunyai sifat kalimat aktif atau pasif, dan pada suatu kata pasti mempunyai akar kata. Oleh karena itu variasi kata tersebut perlu direduksi agar mendapatkan hasil analisis yang lebih baik lagi, proses mereduksi variasi kata menjadi akar kata disebut Lemmatization. Lemmatization adalah proses mengestrak akar kata dari kata yang ditempelkan dengan tujuan untuk mereduksi variasi kata menjadi akar kata (Arimbawa & ER, 2020) . Di dalam Bahasa Inggris misalnya kata runs, running, ran adalah semua bentuk kata run, oleh karena itu run adalah inti dari semua kata ini dan lemmatization mengembalikan kata yang sebenarnya dari bahasa tersebut. Lemmatization digunakan jika diperlukan untuk mendapatkan kata-kata yang valid (Jabeen, 2018).



Stemming vs Lemmatization

Stemming vs Lemmatization

- Stemming dan Lemmatization keduanya menghasilkan bentuk akar dari kata-kata infleksi. Perbedaannya adalah bahwa Stemming mungkin bukan kata yang sebenarnya sedangkan Lemmatization adalah kata bahasa yang sebenarnya.
- Stemming mengikuti algoritma dengan langkah-langkah yang harus dilakukan pada kata-kata yang membuatnya lebih cepat. Sedangkan, dalam Lemmatization menggunakan korpus WordNet dan korpus untuk stopword juga untuk menghasilkan lemma yang membuatnya lebih lambat daripada Stemming.

Stemming vs Lemmatization

Kapan Menggunakan Lemmatization dan Kapan Menggunakan Stemming ?

Jika kecepatan difokuskan maka Stemming harus digunakan, karena lemmatizers memindai korpus yang menghabiskan waktu dan pemrosesan. Jika kita sedang ingin membangun aplikasi bahasa dimana bahasa itu penting, maka kita harus menggunakan Lemmatization, karena menggunakan korpus untuk mencocokan bentuk akarnya. Jadi sebenarnya tergantung kasus apa yang sedang kita kerjakan.

Konversi Kalimat - Padding

- Proses pembelajaran yang dilakukan oleh neural network memerlukan masukan data dengan panjang yang sama.
- Padding merupakan proses yang dilakukan untuk membuat input mempunyai panjang yang sama dengan cara menambahkan kata “<pad>”.
- Dataset pada penelitian itu memiliki panjang teks yang berbeda-beda.
- Oleh karena itu, perlu dilakukan padding agar vektor memiliki panjang yang sama sebelum diproses pada neural network.

Kampus Merdeka INDONESIA JAYA

Poltek Harber
POLITEKNIK HARAPAN BERSAMA

4.Konversi Kalimat

```
[ ]
#Pembuatan Kamus kata
t  = Tokenizer()
fit_text = reviews
t.fit_on_texts(fit_text)

#Pembuatan Id masing-masing kata
sequences = t.texts_to_sequences(reviews)

#hapus duplikat kata yang muncul
list_set_sequence = [list(dict.fromkeys(seq)) for seq in sequences]

#mencari max length sequence
def FindMaxLength(lst):
    maxList = max((x) for x in lst)
    maxLength = max(len(x) for x in lst )
    return maxList, maxLength

# Driver Code
max_seq, max_length_seq = FindMaxLength(list_set_sequence)
jumlah_index = len(t.word_index) +1

print('jumlah index : ',jumlah_index,'\n')
print('word_index : ',t.word_index,'\n')
print('index kalimat asli     : ', sequences,'\n')
print('kalimat tanpa duplikat : ',list_set_sequence,'\n')
print('panjang max kalimat : ', max_length_seq,'kata','\n')
# print('kalimat terpanjang setelah dihapus duplikat : ', max_seq,'\n')

count_word = [len(i) for i in list_set_sequence]
print('list panjang kalimat : ', count_word)
max_len_word = max(count_word)
print(max_len_word)
```

Coding of Sentences Convertion

```
[ ]
min_len_word=min(count_word)
print(min_len_word)
```

```
6
```

```
[ ]
print ("Original list is : " + str(list_set_sequence[0]))
```

```
Original list is : [27, 131, 32, 3, 12, 13, 132, 133, 134, 135, 78, 136, 4]
```

Kampus Merdeka INDONESIA JAYA

Poltek Harber
POLITEKNIK HARAPAN BERSAMA

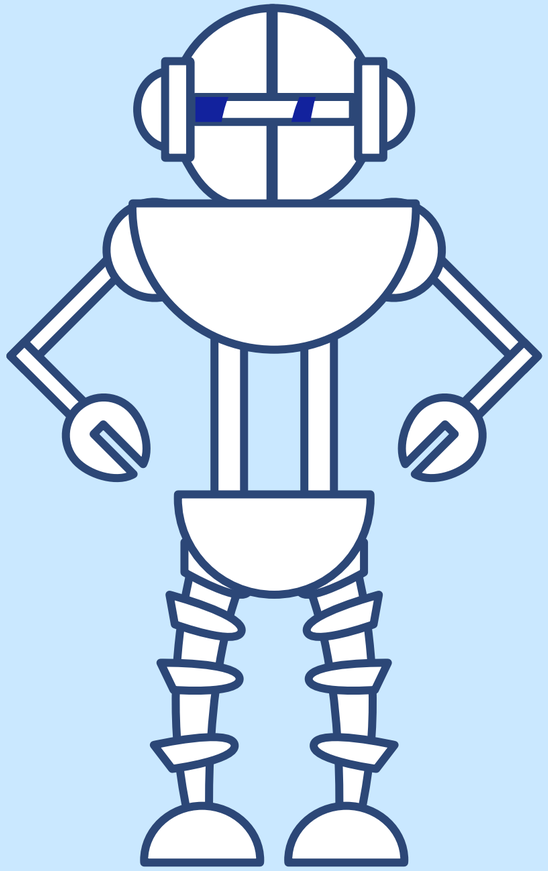
Coding of Padding

4.1. Padding

```
1 #Padding
2 from keras.preprocessing.sequence import pad_sequences
3 padding= pad_sequences([list(list_set_sequence[i]) for i in range(len(list_set_sequence))],
4 maxlen= max_len_word, padding='pre')
5 padding[:10]
```

```
array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,  27, 131,  32,   3,  12,  13,
        132, 133, 134, 135,  78, 136,   4],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  17,  18,
         28,  33,  49, 137,  50, 138, 139],
```

Reference



- Kedia, A., dan Rasu, M., 2020, Hands-On Python Natural Language Processing, Packt Publishing Ltd., Brimingham, UK
- Hidayatullah, A.F., dan Ma'arif, M.R., 2016, Pre-processing Tasks in Indonesia Twitter Messages, Journal of Physics 801, 1–6.

Thank You