知能システムレポート 3

三浦夢生

2020 年 12 月日

1　目的

gensim や MeCab といったライブラリを用いて文章の形態素解析やモデルの作成，作者の分類を行い，ライブラリの使い方を学ぶ.

2　用語

今回用いたライブラリ等について簡単に述べる.

2.1　形態素解析

文法等の追加情報がない自然言語のテキストを，対象とする言語の文法や辞書データに基づいて意味を持つ最小単位に分割し，品詞などの区別を行うことである．この最小単位のことを形態素という.

プログラミングにおける字句解析や文法解析に相当する.

2.2　MeCab

京大，NTT コミュニケーションズの共同研究プロジェクトを通じて開発されたオープンソースの形態素解析エンジンのこと．辞書やコーパスに依存しない汎用的な設計を基本指針としており，他のエンジンより平均的に高速に動作する.

今回入力テキストを形態素解析し，分かち書き形式にするために用いた.

2.3　gensim

gensim とは，オープンソースの Python 向け自然言語処理ライブラリのことで，単語のベクトル化やトピックモデルの提供をする.

今回は MeCab で作成した分かち書きをベクトル化し学習するために用いた.

3　方法

ソースコードを付録に示す.

融資が提供している青空文庫の非公式 API にリクエストを送信し，ステータスコード 200 が帰ってきたものの内，日本人著者の文章を抽出，MeCab で分かち書きに変換し gensim で学習を行った.

また，指定した作品を API から取得し，gensim を用いて類似度の評価を行う.

4　実行結果

学習プログラム及び予測プログラムの実行結果を以下に示す.

```
1 retrieve 2423 books
2 180 authors
3 authors 小栗虫太郎{'', 尾崎紅葉'', 戸坂潤'', 福田英子'', 逸見猶吉'', 北條民雄'', 松本泰
```

```
'', 大杉栄'', 幸田露伴'', 正岡子規'', 伊藤野枝'', 添田唖蟬坊'', 太宰治'', 水野仙子
'', 久坂葉子'', 九鬼周造'', 吉行エイスケ'', 野呂栄太郎'', 飯田平作'', 西田幾多郎
'', 河上肇'', 山村暮鳥'', 金子ふみ子'', 黒岩涙香'', 野上豊一郎'', 村山槐多'', 尾形
亀之助'', 薄田淳介'', 吉江喬松'', 内村鑑三'', 市島春城'', 水野葉舟'', 石川啄木
'', 高山樗牛'', 加能作次郎'', 樋口一葉'', 石原莞爾'', 富田倫生'', 泉鏡花'', 寺田寅
彦'', 田中貢太郎'', 鷹野つぎ'', 楠山正雄'', 野口米次郎'', 山中貞雄'', 与謝野晶子
'', 若山牧水'', 牧逸馬'', 中戸川吉二'', 森林太郎'', 綱島梁川'', 斎藤緑雨'', 橋本五
郎'', 島木健作'', 南方熊楠'', 素木しづ子'', 河東碧梧桐'', 小林多喜二'', 別所梅之助
'', 宮本百合子'', 國木田獨歩'', 佐左木俊郎'', 牧野信一'', 三木清'', 淡島寒月'', 宮
澤賢治'', 大阪圭吉'', 葛西善蔵'', 甲賀三郎'', 石橋忍月'', 木下尚江'', 梶井基次郎
'', 小泉節子'', 国木田独歩'', 阿部徳蔵'', 薄田泣菫'', 平林初之輔'', 田中英光'', 松
濤明'', 桑原隲蔵'', 南部修太郎'', 森鴎外'', 萩原朔太郎'', 大手拓次'', 長谷川時雨
'', 林芙美子'', 橋本進吉'', 平出修'', 八木重吉'', 武田麟太郎'', 黒島伝治'', 黒島傳
治'', 伊藤左千夫'', 鴨長明'', 桐生悠々'', 田澤稲舟'', 内藤湖南'', 幸徳秋水'', 松永
延造'', 新美南吉'', 葉山嘉樹'', 辻潤'', 伊丹万作'', 桑原隲藏'', 立原道造'', 若松賤
子'', 齋藤茂吉'', 徳冨蘆花'', 小島烏水'', 二葉亭四迷'', 泉鏡太郎'', 佐藤紅緑'', 上
田敏'', 中原中也'', 北一輝'', 岡本綺堂'', 長塚節'', 森田草平'', 夢野久作'', 鈴木三
重吉'', 有島武郎'', 北原白秋'', 三島霜川'', 高田力'', 岩野泡鳴'', 木内高音'', 素木
しづ'', 森本薫'', 無署名（夢野久作）'', 徳田秋聲'', 國木田独歩'', 海野十三'', 紀貫之
'', 尾崎放哉'', 宮武外骨'', 巌谷小波'', 與謝野晶子'', 徳田秋声'', 二十四世八観世左近
u3000', 押川春浪'', 夏目漱石'', 田畑修一郎'', 狩野亨吉'', 渡辺温'', 織田作之助
'', 折口信夫'', 小出楢重'', 横光利一'', 三文字屋金平'', 野口雨情'', 中島敦'', 石田
孫太郎'', 斎藤茂吉'', 漢那浪笛'', 菊池寛'', 巖谷小波'', 今井邦子'', 種田山頭火
'', 生田春月'', 宮原晃一郎'', 宮沢賢治'', 横瀬夜雨'', 池谷信三郎'', 加福均三'', 濱
田青陵'', 坂本竜馬'', 内田魯庵'', 塚原蓼洲'', 十一谷義三郎'', 島崎藤村'', 芥川龍之介
'', 岡本かの子'', 葛西善藏'', 小野浩'', 小山内薫'', 堺利彦'', 宮本百合子訳'', 田山
花袋'', 金田千鶴'', 木下杢太郎''}
```

```
1 authors 宮沢賢治{'', 芥川龍之介'', 太宰治'', 夏目漱石''}
2 title 古典竜頭蛇尾 ['', 後世'', 「自然を寫す文章」'', あけがた'']太宰治古典竜頭蛇尾
3
4 ---[:]--- と似た作品は宮本百合子鉛筆の詩人へ
5 [(':', 0.8146466016769409), 宮本百合子日本文化のために (':', 0.807538628578186), 宮本
      百合子科学の常識のため (':', 0.7985014319419861)]芥川龍之介後世
6
7 ---[:]--- と似た作品は辻潤惰眠洞妄語
8 [(':', 0.750522255897522), 夏目漱石子規の画 (':', 0.7126219272613525), 宮本百合子動物
      愛護デー (':', 0.7125130891799927)]夏目漱石「自然を寫す文章」
9
10 ---[:]--- と似た作品は内藤湖南平安朝時代の漢文學
11 [(':', 0.8074424266815186), 幸田露伴些細なやうで重大な事
      (':', 0.8049200773239136), 内藤湖南染織に關する文獻の研究
      (':', 0.7996359467506409)]宮沢賢治あけがた
12
13 ---[:]--- と似た作品は宮沢賢治さいかち淵
14 [(':', 0.6871287822723389), 宮沢賢治花椰菜 (':', 0.6742680072784424), 宮沢賢治土神と
      狐 (':', 0.6722469329833984)]
```

学習の段階で 180 の著者，2423 の作品を取得し学習を行った．

テストとして宮沢賢治，太宰治，芥川龍之介，夏目漱石の 4 作品に近い作品を予測させたが，宮沢賢治以外では芳しい結果は得られなかった．

5 考察

今回リクエストした際のパラメータが単純に 0 から 20000 までの総当りだったため各著者から数作品ずつではないため，特徴の抽出がうまくいかず期待した結果が得られなかったことが考えられる．

各著者から決まった作品数を抽出するか，外国人著者もトレーニングデータに追加するなどしたらもっと良いモデルが作れるかもしれない．

6 付録

今回用いたプログラムを以下に示す．

6.1 学習プログラム

```
import requests as req
from bs4 import BeautifulSoup
import MeCab
from gensim import models
from gensim.models.doc2vec import TaggedDocument
import threading

def get_books():
    booksInfo = []
    for book_id in range(20000):
        print(f'book id is {book_id}')
        res = req.get('http://pubserver2.herokuapp.com/api/v0.1/books/{}/content?format=html'.format(book_id))
        if res.status_code == 200:
            soup = BeautifulSoup(res.text, 'html.parser')
            if soup.find('title').__class__.__name__ != 'NoneType':
                if len(soup.find('title').get_text().split(' ')) == 2:
                    if soup.find('div', {'class': 'main_text'}).__class__.__name__ != 'NoneType':
                        title_list = soup.find('title').get_text().split(' ')
                        text = soup.find('div', {'class': 'main_text'}).get_text()
                        text = text.replace [#改ページ] ('', '')
                        text = text.replace('\u3000', '')
                        text = text.replace('\r', '')
                        booksInfo.append(title_list + [text])
    return booksInfo

def wakachi(_mecab, _text):
    node = _mecab.parseToNode(_text)
    wakachiWord = []
    while node is not None:
        hinshi = node.feature.split(',')[0]
        if hinshi in 名詞 ['']:
            wakachiWord.append(node.surface)
        elif hinshi in 動詞 ['', 形容詞'']:
            wakachiWord.append(node.feature.split(',')[6])
```

```
35          node = node.next
36      return wakachiWord
37
38  def make_doc(_mecab, _booksInfo):
39      documents = []
40      for i in range(len(_booksInfo)):
41          wakachiWord = wakachi(_mecab, _booksInfo[i][2])
42          document = TaggedDocument(wakachiWord, [_booksInfo[i][0] + ':' + _booksInfo[i
                ][1]])
43          documents.append(document)
44      return documents
45
46  def main():
47      mecab = MeCab.Tagger('-d /usr/lib/mecab/dic/mecab-ipadic-neologd/')
48      mecab.parse('')
49
50      booksInfo = get_books()
51      print(f'retrieve {len(booksInfo)} books')
52      author = []
53      for i in range(len(booksInfo)):
54          author.append(booksInfo[i][0])
55      author = set(author)
56      print(f'{len(author)} authors')
57      print(f'authors {author}')
58      documents = make_doc(mecab, booksInfo)
59
60      model = models.Doc2Vec(documents, dm=1, vector_size=300, window=5, min_count=1)
61      model.save('aozora.model')
62
63  if __name__ == '__main__':
64      main()
```

6.2 予測プログラム

```
1   import requests as req
2   from bs4 import BeautifulSoup
3   import MeCab
4   from gensim import models
5   from gensim.models.doc2vec import TaggedDocument
6
7   def get_books(_bookList):
8       booksInfo = []
9       for book_id in _bookList:
10          res = req.get('http://pubserver2.herokuapp.com/api/v0.1/books/{}/content?format
                =html'.format(book_id))
```

```
11          if res.status_code == 200:
12              soup = BeautifulSoup(res.text, 'html.parser')
13              if len(soup.find('title').get_text().split(' ')) == 2 and soup.find('div',
                     {'class': 'main_text'}).__class__.__name__ != 'NoneType':
14                  title_list = soup.find('title').get_text().split(' ')
15                  text = soup.find('div', {'class': 'main_text'}).get_text()
16                  text = text.replace [#改ページ] ('', '')
17                  text = text.replace('\u3000', '')
18                  text = text.replace('\r', '')
19                  booksInfo.append(title_list + [text])
20      return booksInfo
21
22  def wakachi(_mecab, _text):
23      node = _mecab.parseToNode(_text)
24      wakachiWord = []
25      while node is not None:
26          hinshi = node.feature.split(',')[0]
27          if hinshi in 名詞 ['']:
28              wakachiWord.append(node.surface)
29          elif hinshi in 動詞 ['', 形容詞'']:
30              wakachiWord.append(node.feature.split(',')[6])
31          node = node.next
32      return wakachiWord
33
34  def similar(_booksInfo, _mecab, _model):
35      for i in range(len(_booksInfo)):
36          wakachiWord = wakachi(_mecab, _booksInfo[i][2])
37          vector = _model.infer_vector(wakachiWord)
38          print('---[' + _booksInfo[i][0] + ':' + _booksInfo[i][1] + ']--- と似た作品は
                ')
39          print(_model.docvecs.most_similar([vector], topn=3))
40          print()
41
42  def main():
43      mecab = MeCab.Tagger('-d /usr/lib/mecab/dic/mecab-ipadic-neologd/')
44      mecab.parse('')
45      model = models.Doc2Vec.load('aozora.model')
46
47      bookList = [24457, 33202, 42297, 48198]
48      booksInfo = get_books(bookList)
49      print(f'retrieve {len(booksInfo)} books')
50      author = []
51      title = []
52      for i in range(len(booksInfo)):
53          author.append(booksInfo[i][0])
54          title.append(booksInfo[i][1])
```

```
55      author = set(author)
56      print(f'{len(author)} authors')
57      print(f'authors {author}')
58      print(f'title {title}')
59      print()
60      similar(booksInfo, mecab, model)
61
62  if __name__ == '__main__':
63      main()
```