تشخیص کلمات غیرقانونی (تمرین دوم)

پردازش زبان‌های طبیعی

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف

محمدرضا دویران

نونا قاضی‌زاده

سارا آذرنوش

بهار ۱۴۰۲

مقدمه

در این پروژه هدف ما پیاده‌سازی یک سامانه‌ای است که قابلیت تشخیص کلمات غیرقانونی فارسی که ممکن است با طرق خاصی تغییر کرده باشند، را داشته باشد. لازم به ذکر است که بات‌هایی وجود دارد که وظیفه‌شان تشخیص کلمات غیرقانونی است، ولی بعضی از این بات‌ها نمی‌توانند این نوع کلمات را زمانی که بین حروفشان کاراکترهای غیرمرتبط بیاید، تشخیص دهند. بنابراین ما در این سامانه می‌خواهیم کلمات غیرقانونی را تشخیص دهیم که ممکن است بین حروف‌شان، حروف غیر فارسی (از جمله حروف انگلیسی)، اعداد و کاراکترهای خاص و ... آمده باشد.

همچنین ما در این سامانه دو حالت دیگر را نیز پشتیبانی می‌کنیم. اولی، حالتی است که کاربر به جای وارد کردن کل کلمه تنها مخفف آن را وارد کند به طور مثال به جای آنکه کلمه "جمهوری اسلامی" را بنویسد کلمه "جا" که مخفف جمهوری اسلامی است را بنویسد. دومی، حالتی است که در آن از کلمات هم‌سیاق معادل استفاده کند به طور مثال به جای کلمه "سلام" از کلمه "درود" استفاده کند.

پیاده‌سازی

استفاده از عبارات منظم و نرمالایز کردن و lemmatize کردن

برای بخش اول که می‌بایست کلمات غیرقانونی را که میان حروف‌شان، حروف غیر فارسی (از جمله حروف انگلیسی)، اعداد و کاراکترهای خاص را تشخیص دهیم. به همین منظور از عبارات منظم استفاده می‌کنیم. بدین صورت که با استفاده از تابع زیر حروف فارسی در هر کلمه را پیدا می‌کنیم و آن‌ها را به یکدیگر متصل می‌کنیم تا کلمه‌ای عاری از هرگونه حروف غیر فارسی (از جمله حروف انگلیسی)، اعداد و کاراکترهای خاص میان حروفش باشد.

```
def find_persian(string: str):
  return "".join(re.findall(r"[\u0600-\u06FF]+", string))
```

لازم به ذکر است که همچنین ممکن است کاربر برای دور زدن از اعداد فارسی استفاده کند. برای ساپورت کردن این حالت ما تمام اعدادی که در کلمات به فارسی آمده است را به صورت زیر (ابتدا با استفاده از maketrans یک translator از زبان فارسی به انگلیسی می‌سازیم سپس با استفاده از translate اعداد فارسی را به انگلیسی تبدیل می‌کنیم و سپس ادامه کار را می‌دهیم. برای ساخت translator به صورت زیر عمل می‌کنیم:

```
persian_numbers = '۱۲۳۴۵۶۷۸۹۰'
english_numbers = '1234567890'
english_trans = str.maketrans(persian_numbers, english_numbers)
```

سپس برای ترجمه اعداد از فارسی به انگلیسی به صورت زیر عمل می‌کنیم.

```
str(word).translate(english_trans)
```

لازم به ذکر است که نیاز است همچنین بررسی شود که نرمالایز شده یک کلمه و یا lemmatize شده کلمه جز کلمات غیرقانونی نباشد زیرا ممکن است به طور برای حالت نرمالایز کاربر کلمه دیگران را به صورت دیــگران بنویسد و یا برای حالت lemmatize شده کلمه را به صورت جمع آن بنویسد. به همین منظور از lemmatizer و normalizer استفاده می‌کنیم و به صورت زیر کلمه را نرمالایز و lemmatize می‌کنیم و بررسی می‌کنیم این کلمه در میان کلمات غیر قانونی نباشد.

```
normalized_word = normalizer.normalize(transfered_word)
lemmatized_word = lemmatizer.lemmatize(transfered_word)
```

بررسی وجود فاصله در میان کلمه غیرقانونی

گاها ممکن است که کاربر برای رد گم کردن میان حروف کلمه علاوه بر کاراکتر و این موارد از فاصله استفاده می‌کند. برای هندل کردن این مورد از تابع زیر استفاده می‌کنیم که عملا فاصله میان حروف کلمه را ایگنور می‌کند و سپس بررسی می‌شود که آیا این کلمه جز کلمات غیرقانونی است یا نه.

```
def find_words_ignoring_spaces(splitted_input, transfered_words):
  found_spaced_ill_words = {}
   text = "".join(transfered_words)
  for word in illegal_words:
    range = []
    word = word.strip()
    f = text.find(word)
    if f > 0:
      flag = 0
      for counter, index in enumerate(indices):
        if index[0] == f:
          f = f + len(word)
          range.append(counter)
```

```
            flag = 1
        if flag == 1 and index[1] == f:
            range.append(counter)
            break
    if range[0] != range[-1]:
        found_spaced_ill_words[word] = input[input.find(splitted_input[range[0]]):
input.find(splitted_input[range[-1] + 1])]
 return found_spaced_ill_words
```

بررسی استفاده کلمات مخفف با استفاده از دیتاست ساخته شده

حال برای بخش دوم که مربوط به کلمات مخفف است ابتدا یک خزنده برای ساخت یک دیتاست برای کلمات مخفف فارسی به صورت زیر پیاده‌سازی می‌کنیم.

```
import json
import os
import requests
from bs4 import BeautifulSoup
import lxml
URL =
"https://www.mokhafaf.com/word/%DA%A9%D9%84%D9%85%D8%A7%D8%AA-%D9%81%D8%A7
%D8%B1%D8%B3%D9%8A/"
DIR_NAME = '../datasets'
os.makedirs(DIR_NAME, exist_ok=True)
f = requests.get(URL)
soup = BeautifulSoup(f.content, 'lxml')
abbreviations = soup.findAll('div', {'class': 'short-content'})
data = {}
for abbreviation in abbreviations:
    abbreviated = abbreviation.find('h2').getText().replace('مخفف ', '').replace('.', '\u200C')
    real = abbreviation.find('h3').getText()
```

```
    data[abbreviated] = real
with open(f"{DIR_NAME}/abbreviation.json", "w", encoding="utf-8") as file:
    json.dump(data, file, ensure_ascii=False)
```

در نتیجه یک دیتاست که در این لینک وجود دارد را می‌سازیم. سپس این دیتاست را در پروژه لود می‌کنیم و سپس با استفاده از کلمه‌ای که در بخش قبل از متصل کردن حروف فارسی می‌سازیم بررسی می‌کنیم که آیا این کلمه مخفف شده یکی از کلمات غیرقانونی است یا نه. (بخش bold شده به همین منظور بررسی کلمات مخفف است)

```
if transfered_word in illegal_words or (transfered_word in abbreviation and
abbreviation[transfered_word] in illegal_words) or normalized_word in illegal_words or
lemmatized_word in illegal_words:
```

بررسی استفاده از کلمات هم‌سیاق با استفاده از fasttext

برای بخش سوم که برای یافتن کلمات هم‌سیاق است از fasttext استفاده می‌کنیم. بدین صورت که از مدل fasttext فارسی استفاده می‌کنیم سپس با استفاده از این مدل که train شده است با دادن یک کلمه، کلماتی که با همان سیاق هستند را به دست می‌آوریم. لازم به ذکر است که در این حالت ممکن است کلماتی نظیر جمع کلمه (مانند گل و گل‌ها)، نوع دیگر نوشتار (مانند بانک و بانك) و ... به ما توسط این مدل پیشنهاد بشود حال ما برای کنترل این مورد که دیگر این موارد را بررسی نکنیم زیرا در نرمالایز کردن و lemmatize کردن آن‌ها را بررسی کردیم از فاصله levenshtein استفاده می‌کنیم و برای آن یک threshold به مقدار ۳ انتخاب می‌کنیم که در این صورت کلماتی که فاصله levenshtein آن‌ها بیش از سه هست و همچنین فاصله کسینوسی آن‌ها با کلمه داده شده بیشتر از ۰.۵ است را به تعداد ۳۰ تا انتخاب می‌کنیم و در صورتی که کلمه‌ای در جمله آمده باشد و یکی از هم‌سیاق‌های آن جز کلمات غیرقانونی باشد آن کلمه را در نظر می‌گیریم. برای این کار به صورت زیر عمل می‌کنیم.

ابتدا مدل فارسی fasttext را به صورت زیر لود می‌کنیم:

```
model = fasttext.load_model('cc.fa.300.bin')
```

سپس با استفاده از تابع زیر کلمات هم‌سیاق را با توجه به توضیحاتی که در پاراگراف قبل دادیم به دست می‌آوریم:

```
def find_similar_words_in_context(given_word, k_num=30):
    similar_words = model.get_nearest_neighbors(given_word, k=k_num)
    founded_words = []
    for similarity, word in similar_words:
```

```
        if given_word not in word and Levenshtein.distance(word, given_word) > 3     and similarity >
0.5:
            founded_words.append(word)
    return founded_words
```

حال در تابع اصلی‌مان یعنی run به صورت زیر کلماتی که هم سیاق کلمه داده شده هستند را به دست می‌آوریم و بررسی می‌کنیم که آیا جز کلمات غیرقانونی هست یا نه.

```
founded_words = find_similar_words_in_context(transfered_word)
        for founded_word in founded_words:
            normalized_founded_word = normalizer.normalize(founded_word)
            lemmatized_founded_word = lemmatizer.lemmatize(founded_word)
            if founded_word in illegal_words or (founded_word in abbreviation and
abbreviation[founded_word] in illegal_words) or normalized_founded_word in illegal_words or
lemmatized_founded_word in illegal_words:
                found_ill_words[transfered_word] = word
```

تابع اصلی (run)

تمام توضیحات داده شده را باید در یک تابع اصلی هندل کنیم به همین منظور این تابع بدین صورت است که یک ورودی که جمله ماست به همراه کلمات غیرقانونی به عنوان ورودی می‌گیرد و با در نظر گرفتن تمام موارد فوق در صورت وجود کلمه غیرقانونی در جمله، آن کلمه و محل قرارگیری آن را نشان می‌دهیم. این تابع به صورت زیر پیاده می‌شود.

```
def run(input: str, illegal_words: list):
    status = 1
     splitted_input = input.split()

    found_ill_words = {}
    transfered_words = []
    actual_words = []
    for word in splitted_input:
        modified_word = str(word).translate(english_trans)
        transfered_word = find_persian(modified_word)
```

```
        transfered_words.append(transfered_word)
        normalized_word = normalizer.normalize(transfered_word)
        lemmatized_word = lemmatizer.lemmatize(transfered_word)
        if transfered_word in illegal_words or (transfered_word in abbreviation and
abbreviation[transfered_word] in illegal_words) or normalized_word in illegal_words or
lemmatized_word in illegal_words:
            found_ill_words[transfered_word] = word
        founded_words = find_similar_words_in_context(transfered_word)
        for founded_word in founded_words:
            normalized_founded_word = normalizer.normalize(founded_word)
            lemmatized_founded_word = lemmatizer.lemmatize(founded_word)
            if founded_word in illegal_words or (founded_word in abbreviation and
abbreviation[founded_word] in illegal_words) or normalized_founded_word in illegal_words or
lemmatized_founded_word in illegal_words:
                found_ill_words[transfered_word] = word

    print_illegal_words(input, found_ill_words)

    if status == 1: #ignoring space
        found_spaced_ill_words = find_words_ignoring_spaces(splitted_input, transfered_words)
        print_illegal_words_space(found_spaced_ill_words)
```

توابع برای نشان دادن کلمات غیرقانونی به همراه محل قرارگیری کلمات (با فاصله و بدون فاصله)

لازم به ذکر است که دو تابع print_illegal_words و print_illegal_words_space به منظور نمایش کلمه غیر قانونی به همراه محل قرارگیری آن است. این دو تابع به صورت زیر پیاده می‌شوند.

برای کلمات فاقد هرگونه فاصله در میان حروف آن:

```
def print_illegal_words(input, ill_words):
    if len(ill_words) > 0:
        print('Without any space between each word: \n')
    for w in ill_words:
        complete_word = ill_words[w]
```

```
        index = input.find(complete_word)
        print('word: \'' + w + '\'')
        print(f'span: ({index}, {index + len(complete_word)})\n')
```

برای کلمات دارای فاصله در میان حروف آن:

```
def print_illegal_words_space(found_spaced_ill_words):
    if len(found_spaced_ill_words) > 0:
        print('\nWith white space between each word: \n')
    for w in found_spaced_ill_words:
        complete_word = found_spaced_ill_words[w]
        index = input.find(complete_word)
        print('word: \'' + w + '\'')
        print(f'span: ({index}, {index + len(complete_word)})\n')
```

ارزیابی

تست بدون وجود فاصله در میان کلمه

تست ۱:

```
input = "این &تف...ن۸گ# را فروختم"
illegal_words = ['تفنگ']
run(input, illegal_words)
```

در این تست کلمه تفنگ جزء کلمه غیرقانونی است که ابتدا، وسط و انتهای آن کاراکترهایی آمده بنابراین انتظار داریم سامانه این کلمه و محل قرار گیری آن را به ما بدهد. که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:


word: "تفنگ"
span: (4, 14)
```

تست ۲:

```
input = "چ^!نگ4ال غذا خوردمf شق وalb*با ق."
illegal_words = ['قاشق','چنگال']
run(input, illegal_words)
```

در این تست کلمه قاشق و چنگال جزء کلمه غیرقانونی است که ابتدا، وسط و انتهای آن‌ها کاراکترهایی آمده بنابراین انتظار داریم سامانه این کلمات و محل قرار گیری آن‌ها را به ما بدهد. که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:

word: "قاشق"
span: (3, 10)

word: "چنگال"
span: (13, 22)
```

تست با وجود فاصله در میان کلمه

تست ۱:

```
input = "این &تف...ن۸# گ را به اح* مد تو سنگ7%#3!8..اپور   فروختم"
illegal_words = ['سنگاپور','تفنگ','احمد']
run(input, illegal_words)
```

در این تست کلمه سنگاپور، تفنگ و احمد جزء کلمه غیرقانونی است که ابتدا، وسط و انتهای آن‌ها کاراکترهایی آمده همچنین در میان حروف کلمات تفنگ و احمد فاصله آمده است. بنابراین انتظار داریم سامانه این کلمات و محل قرار گیری آن‌ها را به ما بدهد. که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:

word: "سنگاپور"
span: (32, 47)
```

With white space between each word:

word: "تفنگ"

span: (4, 15)

word: "احمد"

span: (22, 28)

تست ۲:

input = "من دیروز به l8یت۳۲ ا ل یا رفتم و با رض l49 دیدار کردم"

illegal_words = ['رضا' ,'ایتالیا']

run(input, illegal_words)

در این تست کلمات ایتالیا و رضا جزو کلمات غیرقانونی هستند. کلمه‌ی ایتالیا به صورت ۴ بخش مختلف نوشته شده‌است. در این حالت نیز با در نظر نگرفتن فاصله بین این ۴ بخش و کاراکترهای غیرفارسی آن، کلمه ایتالیا توسط سامانه به درستی بازیابی و پیدا می‌شود. همچنین کلمه‌ی رضا نیز به درستی به صورت زیر همراه با span متناظر با آن چاپ می‌شود:

With white space between each word:

word: "ایتالیا"

span: (12, 26)

word: "رضا"

span: (36, 43)

تست با استفاده از نرمالایزر

```
input = "این &دیـــ...#گران را فروختم"
illegal_words = ['دیگران']
run(input, illegal_words)
```

در این تست کلمه دیگران جزء کلمه غیرقانونی است که در اینجا در جمله ما کلمه دیــگران به کار رفته است و ابتدا، وسط و انتهای آن کاراکترهایی آمده بنابراین انتظار داریم سامانه این کلمه و محل قرار گیری آن را به ما بدهد. که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:


word: "دیــگران"
span: (4, 17)
```

تست با استفاده از lemmatizer

```
input = "این &گ..ل.ه#ا را فروختم"
illegal_words = ['گل']
run(input, illegal_words)
```

در این تست کلمه گل جزء کلمه غیرقانونی است که در اینجا در جمله ما کلمه گلها به کار رفته است و ابتدا، وسط و انتهای آن کاراکترهایی آمده بنابراین انتظار داریم سامانه این کلمه و محل قرار گیری آن را به ما بدهد. که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:


word: "گلها"
span: (4, 13)
```

تست با استفاده از وجود مخفف کلمه غیرقانونی در جمله

تست ۱:

```
input = "این &آ...ج#ا آپ فروختم"
illegal_words = ['ارتش جمهوری اسلامی ایران']
run(input, illegal_words)
```

در این تست کلمه ارتش جمهوری اسلامی ایران جزء کلمه غیرقانونی است که در اینجا در جمله ما کلمه آجا (مخفف آن) به کار رفته است و ابتدا، وسط و انتهای آن کاراکترهایی آمده بنابراین انتظار داریم سامانه این کلمه و محل قرار گیری آن را به ما بدهد. همچنین کلمه آپ (مخفف کلمه آسان پرداخت آمده است اما از آنجا که جز کلمات غیرقانونی نیست در نظر نمی‌گیریم) که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:

word: "آجا"
span: (4, 12)
```

تست ۲:

```
input = "این &آ...ج#ا آ..پ# فروختم"
illegal_words = ['آسان پرداخت', 'ارتش جمهوری اسلامی ایران']
run(input, illegal_words)
```

در این تست کلمات ارتش جمهوری اسلامی و آسان پرداخت ایران جزء کلمه غیرقانونی است که در اینجا در جمله ما کلمه آجا و آپ (مخفف آن‌ها) به کار رفته است و ابتدا، وسط و انتهای آن کاراکترهایی آمده بنابراین انتظار داریم سامانه این کلمه و محل قرار گیری آن را به ما بدهد. که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:

word: "آجا"
span: (4, 12)

word: "آپ"
span: (13, 18)
```

تست با استفاده از وجود هم‌سیاق کلمه غیرقانونی در جمله

```
input = "این &س...ل#ام را فروختم"
illegal_words = ['درود']
run(input, illegal_words)
```

در این تست کلمه درود جزء کلمه غیرقانونی است که در اینجا در جمله ما کلمه سلام (هم سیاق آن) به کار رفته است و ابتدا، وسط و انتهای آن کاراکترهایی آمده بنابراین انتظار داریم سامانه این کلمه و محل قرار گیری آن را به ما بدهد. که مطابق با انتظارمان خروجی‌اش به صورت زیر می‌شود:

```
Without any space between each word:

word: "سلام"
span: (4, 13)
```