به نام خدا

گزارش تمرین سوم درس پردازش زبان‌های طبیعی

امیررضا صدیقین
۹۹۳۶۱۴۰۲۴

تمرین ۲ :

بخش a:

در این بخش فایل متن مورد نظر خوانده شد. به دلیل آن‌که در این تمرین حروف بزرگ و کوچک برای ما مهم نیس ولی در پردازش ما بهتر است که همه‌ی حروف کوچک باشد متن را به صورت lowercase درآوردیم.

بخش b:

در این بخش متن به جملات آن تجزیه شده است. (با روش sent_tokenize)

بخش c:

در این بخش هر جمله به توکن‌های آن با روش RegexpTokenizer تبدیل شده است که ریجکس مربوط به آن به صورت "\w+" است و اول و آخر هر جمله نمادهای مربوط به ابتدا و انتهای آن را قرار داده ایم و متن مورد نظر تولید شده است.

بخش d:

در این بخش tokenها و تایپ‌ها دوباره از متن جدید به دست آمده است (این دفعه با استفاده از فاصله‌ی بین کلمات با استفاده از RegexpTokenizer("\S+")) و نمایش داده شده اند.

بخش e:

در این بخش دو پیکره به نام‌های with_stop_words_tokens و without_stop_words_tokens ساخته شده است که در پیکره‌ی دوم کلمات stop_words حذف شده است.

تمرین ۳:

در این بخش برای هر دو پیکره n_gram های مورد نظر استخراج شده است.

تمرین ٤:

در این بخش تعداد تکرار هر n_gram برای هر دو پیکره به دست آمده است (با استفاده از Counter(n_grams)) و ٥ تای اول هر کدام نمایش داده شده است.(با استفاده از most_common)

تمرین ٥:

در این بخش تابع n_gram_probability نوشته شده است که با توجه به فرمول داده شده در درس احتمال یک n_gram را متناسب با پیکره حساب می‌کند . برای هر مدل n_gram برای هر یک از پیکره‌ها یک دیتافریم ساخته شد و احتمال متناسب با n_gram ساخته شده است.

تمرین ٦:

بخش a:

در این بخش تابع predict_word نوشته شده است که دنباله کلمات و مدل مورد نظر را می‌گیرد و با توجه به بیشترین احتمال موجود کلمه‌ی بعد پیش بینی می‌شود. در این تابع ابتدا به ازای هر دنباله‌ی موجود در پیکره تشخیص می‌دهد که آیا این دنباله می‌تواند جواب صحیحی برای دنباله‌ی ورودی باشد یا خیر. آن گاه از بین آن‌هایی که مورد قبول هستند انتخاب شده و بر اساس probability آن‌ها مرتب می‌شود و بهترین آن‌ها انتخاب میشود و برگردانده می‌شود.

نکته ای که در این تمرین موجود است برای دنباله‌های ('beguile', 'the', 'world') و ('in', 'thy', 'noon') به دلیل آن‌که در این دنباله‌ها کلمات stop|_words وجود دارد پیکره‌ی without_stop_words قابل به حدس آن نیس .

بخش b:

در این بخش سه دنباله کلمه برای سه مدل گفته شده، به دست آمده است که پیش بینی در دو پیکره متفاوت باشد. (دلیل آن این است که کلمه‌ی پیش بینی شده در اولی خود یک stopwords است در حالی که در پیکره‌ی دوم کلمات stopwords حذف شده اند.)

sequence	for with stop_words	for without stop_words
('increase',)	that	thereby
('desire', 'increase')	that	thereby
('precious', 'friends', 'hid')	in	death

تمرین ٧ :

در این بخش تابع make_sentence نوشته شده است که در آن مدل زبانی گرفته شده و سایز جمله‌ی خواسته شده را نیز می‌گیرد و متناسب با جمله تولید می‌کند. در این تابع ابتدا n_gram اول انتخاب شده و هر دفعه به اندازه‌ی n تای آخر کلمات انتخاب شده در آن مدل کلمه‌ی بعدی را حدس میزند (مثل تابع predict ولی به طریق for) و با استفاده از این توکن‌ها متن جدید ساخته می‌شود و از بین آن توکن‌ها جملاتی که به اندازهsize توکن دارد جدا می‌کند و اولی را برمیگرداند.

```
for index,model in enumerate([unigrams1 , bigrams1,trigrams1,quadrigrams1]):
    print(index,"_gram         ",make_sentence(model, 20))
```

```
0 _gram         <s> be not self willed for thou art much too fair to be death s conquest and make worms thine heir </s>
1 _gram         <s> be not self willed for thou art much too fair to be death s conquest and make worms thine heir </s>
2 _gram         <s> be not self willed for thou art much too fair to be death s conquest and make worms thine heir </s>
3 _gram         <s> be not self willed for thou art much too fair to be death s conquest and make worms thine heir </s>
```

راه حل بهتر آن است که با استفاده از n_gram ها یک ماشین حالت ساخت و مسیری با طول size از <s> به <\s> انتخاب کرد و برگرداند (به دلیل کمبود وقت نشد پیاده سازی کنم)

تمرین ٨:

نحوه‌ی ساخت مدل زبانی با nltk در سایت https://www.kaggle.com/alvations/n-gram-language-model-with-nltk آمده است (به دلیل کمبود وقت)