به نام خدا

تمرین اول درس پردازش زبان طبیعی

«آشنایی با روش‌های بازنمایی کلمات و ابزار تشخیص اجزای سخن»

استاد درس: دکتر ممتازی

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی امیرکبیر

بهار ۱۴۰۱

برای ارسال تمرین به نکات زیر توجه کنید.

۱- در جدول زیر نحوه اعمال نمره منفی برای تاخیر در ارسال تمرین‌ها ذکر شده‌است.

میزان تاخیر (روز)	میزان جریمه
۱ الی ۲ روز	هر روز ۵٪
۲ الی ۶ روز	هر روز ۱۰٪

در صورتی که برای ارسال تمرین‌ها بین ۷ تا ۱۴ روز تاخیر داشته باشید، نمره شما از ۵۰٪ محاسبه می‌شود و پس از این بازه به تمرین ارسالی نمره‌ای تعلق نمی‌گیرد.

۲- هرگونه کپی‌برداری در انجام تمرین‌ها موجب کسر نمره خواهد شد.

۳- آخرین مهلت ارسال تمرین، **ساعت ۲۳:۵۵ روز شنبه ۲۴ اردیبهشت** می‌باشد.

۴- فایل های ارسالی خود شامل فایل‌های پیاده‌سازی و گزارش را فشرده کنید و با عنوان «**شماره دانشجویی_HW2**» مانند `HW2_97131022` ارسال کنید.

۵- زبان برنامه‌نویسی برای انجام تمرین‌ها، پایتون یا جاوا در نظر گرفته شده‌است.

۶- کدهای ارسالی خود را برای افزایش خوانایی و درک بهتر به صورت مناسب کامنت‌گذاری کنید.

۷- نحوه انجام پیش‌پردازش بر روی داده‌ها شامل کتابخانه مورد استفاده و مراحل انجام‌شده را در گزارش خود مکتوب کنید.

۸- برای انجام این تمرین می‌توانید از کتابخانه‌های آماده استفاده کنید.

۹- در صورت هرگونه سوال یا مشکل می‌توانید با تدریس‌یاران درس از طریق ایمیل زیر در ارتباط باشید.

محمدجواد ظهرابی – رضا زادکمالی nlp.aut.1401@gmail.com

بخش اول: آشنایی با روش‌های بازنمایی کلمات

در این بخش قصد داریم بازنمایی متن را به چندین روش مختلف محاسبه کنیم. مجموعه دادگان مورد استفاده در این تمرین بخشی از مجموعه‌داده‌ی همشهری است. برای این تمرین دو فایل، train.csv به عنوان مجموعه داده آموزش و test.csv به عنوان مجموعه داده آزمون در اختیار شما قرار گرفته شده است. این مجموعه داده دارای دو ستون است. ستون article متن سند و ستون id که مربوط به شناسه‌ی سند است.

```
!gdown --id 1-86CqCHek-U1iH5nW30RfnFU0PYmdhKB
!gdown --id 1YzRlYyye_KoEw7_q9NARiCwl3Cn-EH3J
```

برای این بخش استفاده از کتابخانه‌های آماده مجاز است. برای ایجاد مدل‌های word2vec و doc2vec استفاده از کتابخانه genism توصیه می‌شود. بردار هر متن را ۳۰۰ بعد در نظر بگیرید.

گام اول: ایجاد بازنمایی کلمات

با استفاده از مجموعه داده آموزش، بازنمایی بردار کلمات را با استفاده از مدل word2vec – skip-gram به دست بیاورید.

گام دوم: ایجاد بازنمایی اسناد

با استفاده از مجموعه داده آموزش، مدل‌های بازنمایی زیر را ایجاد کنید.

۱) ایجاد بازنمایی سند از طریق میانگین وزن‌دار بازنمایی‌های کلمات به دست آمده از گام اول و استفاده از TF-IDF هر یک از کلمات به عنوان وزن بردار مربوطه.

۲) آموزش بردار اسناد روی مجموعه داده با استفاده از مدل doc2vec.

گام سوم: یافتن اسناد مشابه

برای هر یک از اسنادی که در ادامه شناسه آن‌ها آمده است، شبیه‌ترین سند به آنها را از مجموعه‌ی آموزش بیابید. برای محاسبه شباهت از معیار شباهت کسینوسی استفاده کنید و همچنین برای محاسبه بردار از هر دو روش مطرح شده در گام دوم استفاده کنید.

- برای محاسبه معیار شباهت کسینوسی، پیشنهاد می‌شود که از کتابخانه scikit-learn استفاده کنید.
- در خروجی احتمال به دست آمده برای میزان شباهت را نیز چاپ کنید.

اسناد: Doc1, Doc3, Doc5, Doc25, Doc36

گام چهارم: بررسی کلمات مشابه

برای هر یک از کلمات زیر سه شبیه‌ترین کلمه را از میان تمام کلماتی که در بازنمایی word2vec از داده آموزش استخراج شده است، بیابید. برای این کار می‌توانید از تابع most_similar مدل word2vec ایجاد شده با کتابخانه gensim استفاده کنید.

در ادامه با استفاده از الگوریتم PCA بعد بردارهای کلمات مشابه و کلمات اصلی را به دو و سه بعد کاهش دهید. نقاط به دست آمده را همراه با کلمه‌ی مرتبط با آن، در دو فضای دوبعدی و سه‌بعدی نمایش دهید. نتیجه‌گیری و تحلیل خود را بیان کنید.

کلمات: تهران، بهداشت، دفاع، رودخانه، سرد، فرهنگ، استقلال

بخش دوم: تشخیص اجزای سخن (POS)[1]

در این بخش قصد داریم یک ابزار تشخیص اجزای سخن تولید کنیم. این ابزار می‌تواند در ابزارهای پردازش زبان طبیعی نظیر ترجمه ماشینی، پارس کردن متن، استخراج اطلاعات، سنتز گفتار و ... مورد استفاده قرار گیرد. مجموعه دادگان مورد استفاده در این بخش، UPC_2016 است. این مجموعه‌داده به سه بخش train، validation و test تقسیم شده است که از طریق لینک‌های زیر می‌توانید آن‌ها را دانلود کنید. هر کلمه از هر نمونه‌ی داده (جمله) در یک سطر قرار گرفته و نمونه‌های داده با استفاده از \n از یکدیگر جدا شده‌اند. در مقابل هر کلمه برچسب POS آن قرار گرفته است.

```
!gdown --id 1Px1lQhMgkdeFigxwmqvk1ojJPZpq40u5
!gdown --id 1WHbpY1Ydqq7yqtQ2YFavR78zsD7nDwbg
!gdown --id 1pMJQk75R3898sUzFKUMQTPPE_QlllvAY
```

گام اول: ایجاد شبکه‌ی عصبی حافظه کوتاه-مدت بلند دوطرفه[2]

شبکه‌ی عصبی بازگشتی (RNN) نوعی شبکه عصبی است که حافظه‌ی داخلی دارد؛ به‌عبارت دیگر، این شبکه یک شبکه‌ی عصبی معمولی است که در ساختارش حلقه‌ای دارد که ازطریق آن در هر گام (Step) خروجی گام قبلی، به‌همراه ورودی جدید، به شبکه وارد می‌شود. شبکه‌ی LSTM یا Long-Short Term Memory نوع خاصی از شبکه‌ی RNN است که مشکل حافظه‌ی بلندمدت شبکه‌ی RNN را حل می‌کند. در این گام قصد داریم که از شبکه‌ی LSTM دو طرفه (BiLSTM) برای حل مسئله‌ی تشخیص اجزای سخن استفاده کنیم. برای پیاده‌سازی این شبکه می‌توانید از کتابخانه‌های Tensorflow (Keras) و یا PyTorch استفاده کنید.

- برای ورودی شبکه، از بردارهای تعبیه پیش آموزش دیده‌ی Glove استفاده کنید. این بردارها را می‌توانید از لینک زیر دانلود کنید.

```
!gdown --id 1NMesvM67oiJ6-PpSbtErFanj0JMl_Z50
```

[1] Part Of Speech (POS)
[2] BiLSTM

گام دوم: ایجاد شبکه‌ی عصبی حافظه کوتاه-مدت بلند دوطرفه[3]

برای ارزیابی مدل گام قبل، از معیار رابطه‌ی ۱ استفاده کنید و مقدار این معیار را روی داده‌های آزمون، برای هر برچسب **POS** به دست آورید. نمودار خطای مدل انتخاب شده را روی مجموعه‌داده‌های train و validation در هر ایپاک[4] رسم کنید. همچنین با رسم کردن ماتریس درهم‌ریختگی[5] بررسی کنید که کدام یک از برچسب‌ها بیشترین خطا را با یکدیگر دارند. تحلیل خود را درباره‌ی ماتریس درهم‌ریختگی بیان کنید.

$$Accuravy = \frac{\#correctly\ tagged\ words}{\#total\ word\ token} \qquad (۱)$$

موفق باشید

[3] BiLSTM
[4] Epoch
[5] Confusion Matrix