به نام خدا

دانشکده علوم و فنون نوین

روش‌های یادگیری ماشین در پردازش زبان طبیعی (۸۳۰۴۳۶۸)
نیم‌سال اول ۱۴۰۲-۱۴۰۱

تمرین شماره ۳

تاریخ تحویل:
۱۴۰۱/۱۰/۰۴

---

۱. (۳۰٪) **[پیاده‌سازی- دسته‌بندی متون فارسی با شبکه عصبی MLP]** یک برنامه رایانه‌ای بنویسید که با استفاده از دادگان مجموعه *زبرا*، متون فارسی را با روش شبکه عصبی MLP دسته‌بندی کند. برای این کار، هر کدام از ۷ پوشه دادگان زبرا را به عنوان ۷ موضوع (دسته) در نظر بگیرید. دو فایل اول هر پوشه را به عنوان داده تست و هشت فایل دیگر را به عنوان داده آموزش به کار ببرید. در این تمرین تعداد ۱۵۰ واژه پرتکرار را (بعد از حذف ایست واژه‌ها) به عنوان ویژگی هر سند در نظر بگیرید. درصد دقت (Accuracy) را روی مجموعه آزمون و مجموعه آموزش گزارش کنید. مقادیر اولیه را به صورت تصادفی بین ۰.۵+ و ۰.۵- انتخاب کرده و از نرخ یادگیری ۰.۰۱ استفاده کنید.

الف) در این شبکه از ویژگی‌های TF (نرمال شده) و TF-IDF استفاده کنید. شبکه را برای هر کدام از ویژگی‌ها، برای حداقل دو تعداد مختلف از نرون‌های لایه مخفی آموزش داده و نتیجه را در هر حالت ارائه دهید. نمودار خطای شبکه در حین آموزش را برای هر حالت رسم کنید.

ب) بر روی داده تست، ماتریس درهم‌ریختگی (Confusion Matrix) را محاسبه کنید.

[نمره اضافی ۵۰٪ نمره این سوال] در پیاده‌سازی این الگوریتم می‌توانید از کدهای آماده استفاده کنید. در صورت پیاده‌سازی الگوریتم MLP توسط خود شما، نمره اضافی به این منظور در نظر گرفته می‌شود.

---

۲. (۲۵٪) **[پیاده‌سازی- دسته‌بندی متون فارسی با شبکه عصبی MLP و Word2Vec]** در این تمرین می‌خواهیم دسته‌بندی متون دادگان زبرا را با ویژگی‌های استخراج شده از کلمات با روش Word2Vec و دسته‌بند MLP انجام دهیم. برای این کار بردارهای از قبل آموزش داده شده را از سایت زیر دریافت کنید (با روش skip gram) و به ازای هر سند میانگین بردارهای کلمات را به عنوان نمایش بردار آن سند محاسبه کنید.

https://nlpdataset.ir/farsi/pre-trained_embeddings.html

یک شبکه MLP با یک لایه مخفی (با تعداد نرون‌های مخفی برابر با میانگین تعداد نرون‌های ورودی و خروجی) ایجاد کرده و با داده مذکور آموزش دهید و بر روی داده تست سوال قبل ارزیابی کنید و نتایج آن را با سوال قبل مقایسه کنید.

h.veisi@ut.ac.ir

به نام خدا

دانشکده علوم و فنون نوین

روش‌های یادگیری ماشین در پردازش زبان طبیعی (۸۳۰۴۳۶۸)
نیم‌سال اول ۱۴۰۲-۱۴۰۱

تمرین شماره ۳

تاریخ تحویل:
۱۴۰۱/۱۰/۰۴

---

۳. (۴۵٪) [**پیاده‌سازی: بدست آوردن بردار کلمات با شبکه عصبی**] در این سوال می‌خواهیم با استفاده از پیکره ایسنا بردار تعبیه کلمات آموزش دهیم. پیشنهاد ما این است که برای این تمرین از کتابخانه gensim[1] استفاده کنید، هرچند اگر مایلید می‌توانید از کتابخانه‌ها و کدهای آماده استفاده نکنید و خودتان پیاده‌سازی را انجام دهید و یا از کتابخانه‌های دیگر استفاده کنید. پیکره آموزش ایسنا را از آدرس زیر دانلود کنید.

https://sourceforge.net/projects/persica/files/persica.csv/download

همان‌طور که می‌بینید، هر نمونه پیکره ایسنا شامل شناسه خبر، عنوان خبر، متن خبر، تاریخ خبر، ساعت انتشار خبر و برچسب خبر است. برای این تمرین تنها به متن خبرها نیاز دارید. در ادامه مرحله به مرحله خواهیم دید برای آموزش بردار کلمات چه باید کرد.

- گام ۱ (خواندن خبرها از فایل): متن خبرها را از فایل بخوانید. برای این کار می‌توانید تابعی به نام read_isna_content بنویسید. ورودی این تابع فایل پیکره و خروجی آن یک رشته است؛ این رشته حاوی متن تمام خبرهای پیکره ایسنا است.
- گام ۲ (نرمال‌سازی): تابعی به نام normalize بنویسید. ورودی این تابع یک رشته و خروجی آن نیز یک رشته است. در این تابع، کدینگ‌های متفاوت یک حرف را یکسان‌سازی کنید («ئ/ی/...» را به «ی»، «ک» به «ک» و الخ) و نرمال‌سازی انجام دهید. برای نرمال‌سازی در این تمرین، بهتر است چیزی را از متن حذف نکنید؛ برای مثال نیازی نیست حروف لاتین را حذف کنید، اما نکته اساسی این است: ما می‌خواهیم هر واحد[2] از واحدهای دیگر فاصله داشته باشد. بنابراین حتما قبل و بعد هر کلمه، هر عدد و هر علامت نگارشی فاصله درج کنید. فراموش نکنید فاصله‌های اضافی را در انتها به یک فاصله تبدیل کنید.
- گام ۳ (تقطیع پیکره به جمله‌ها): پیکره را به جمله‌های تشکیل‌دهنده‌اش تقطیع کنید. مرز جمله را می‌توانید به کمک متد re.split و با علائم سجاوندی همچون «.:؛؟!» شناسایی کنید. برای این گام یک تابع به نام segment_corpus بنویسید. ورودی این تابع یک رشته (یعنی کل

[1] - https://radimrehurek.com/gensim/models/word2vec.html
[2] token

h.veisi@ut.ac.ir

پیکره به صورت یک رشته) و خروجی آن یک لیست از جمله‌های پیکره است. یعنی خروجی چنین چیزی باید باشد:

[جمله ۱ پیکره، جمله ۲ پیکره، جمله ۳ پیکره....، جمله n پیکره]

- گام ۴ (تقطیع جمله‌ها به کلمه‌ها): در گام ۳ متن پیکره را به جمله‌ها تقسیم کردیم و در یک لیست قرار دادیم. در این مرحله، هر جمله را به کلمات آن می‌شکنیم. برای این کار می‌توانید یک تابع به نام prepare_data بنویسید. ورودی این تابع یک لیست از جمله‌های تشکیل‌دهنده پیکره است؛ یعنی چنین چیزی:

[جمله ۱ پیکره، جمله ۲ پیکره، جمله ۳ پیکره....، جمله n پیکره]

و خروجی آن یک لیست تودرتو به شکل زیر است:

[کلمه ۱ جمله ۱ پیکره، کلمه ۲ جمله ۱ پیکره، کلمه ۳ جمله ۱ پیکره،...، کلمه *n* جمله ۱ پیکره]، [کلمه ۱ جمله ۲ پیکره، کلمه ۲ جمله ۲ پیکره، کلمه ۳ جمله ۲ پیکره،...، کلمه *n* جمله ۲ پیکره]،... [کلمه ۱ جمله *n* پیکره، کلمه ۲ جمله *n* پیکره، کلمه ۳ جمله *n* پیکره،...، کلمه *n* جمله *n* پیکره]

- گام ۵ (آموزش مدل): در این بخش باید یک مدل تعبیه‌سازی کلمات آموزش دهید؛ برای این کار از یکی از دو مدل CBOW و مدل Skipgram استفاده کنید. همان‌طور که می‌دانید هر مدل تعبیه کلمات پارامترهای گوناگونی دارد. مدل را با مقادیر زیر برای پارامترهای window، size و negative sampling آموزش دهید. (پارامتر min_count را همیشه برابر با ۲ در نظر بگیرید.)

size={50, 100}
window={5}
negative={5}

در گام ۴ یک لیست تودرتو از کلمات پیکره ساختیم:

[کلمه ۱ جمله ۱ پیکره، کلمه ۲ جمله ۱ پیکره، کلمه ۳ جمله ۱ پیکره،...، کلمه *n* جمله ۱ پیکره]،[کلمه ۱ جمله ۲ پیکره، کلمه ۲ جمله ۲ پیکره، کلمه ۳ جمله ۲

h.veisi@ut.ac.ir

پیکره،...، کلمه n جمله ۲ پیکره]،...[کلمه ۱ جمله n پیکره، کلمه ۲ جمله n پیکره، کلمه ۳ جمله n پیکره،...، کلمه n جمله n پیکره]

حال این لیست تودرتو را، همراه با پارامترهای مدل، به کلاس Word2Vec کتابخانه gensim بدهید و هر مدل را جداگانه ذخیره کنید؛ مانند کد زیر[3].

```
from gensim.models import Word2Vec

model = Word2Vec([['کتاب', 'خوب', 'است'],

['خیلی','خوابالود','هستم']], size=10, window=1,min_count=1, negative=2)

model.save("hw4_w1_n2_size10.model")
```

و بعد وقتی بخواهید، می‌توانید مدل موردنظرتان را لود کنید:

```
model = Word2Vec.load("hw4_w1_n2_size10.model.model")
```

توجه: فراموش نکنید که کلاس Word2Vec یک لیست تودرتو، به همان شیوه که توصیف شد، را به عنوان ورودی می‌پذیرد. پس به گام ۳ و ۴ دقت کنید و مشابه آن‌چه گفته شد، پیاده‌سازی کنید.

- گام ۶ (ارزیابی):

الف) با مدل‌های که آموزش داده شد، پنج نزدیک‌ترین کلمه به کلمات زیر را پیدا کنید و توضیح دهید: آیا تفاوتی بین مدل‌ها مشاهده می‌کنید؟ کدام مدل به نظر شما مدل بهتری است؟ چرا؟

[3] این کد مثال است و برای همین فقط دو جمله ورودی دارد. در مستندات کتابخانه gensim می‌توانید مثال‌های بیشتری ببینید. اگر نسخه gensim شما قدیمی باشد، ممکن است نام پارامترها تفاوت داشته باشد. مثلا در یکی از نسخه‌های قدیمی gensim پارامتر size داریم و در نسخه جدید آن پارامتر vector_size. اطمینان حاصل کنید که نسخه کتابخانه‌ای که نصب کرده‌اید، با مستنداتی که به آن رجوع می‌کنید، یکی باشد. (مستندات gensim4.0.0 و مستندات gensim3.8.3)

h.veisi@ut.ac.ir

به نام خدا

روش‌های یادگیری ماشین در پردازش زبان طبیعی (۸۳۰۴۳۶۸)

نیم‌سال اول ۱۴۰۲-۱۴۰۱

تمرین شماره ۳

تاریخ تحویل:
۱۴۰۱/۱۰/۰۴

ایران، دانشگاه، دولت، انقلاب، قانون

برای این کار می‌توانید از متد similar_by_word کتابخانه gensim استفاده کنید:

model.similar_by_word('دانشگاه')

size	w	n	کلمات نزدیک «ایران»	کلمات نزدیک «دانشگاه»	کلمات نزدیک «دولت»	کلمات نزدیک «انقلاب»	کلمات نزدیک «قانون»
۱۰۰	۵	۵					
۵۰	۵	۵					

ب) یکی از روش‌های ارزیابی تعبیه کلمات «تشابه کلمات» است. با استفاده از معیار شباهت کسینوسی، میزان شباهت بین جفت کلمه‌های موجود در فایل antonyms.txt را که همراه این تمرین ارائه شده است، محاسبه کنید و میانگین شباهت را محاسبه کنید. سپس جدول زیر را برای ۱۶ مدل خود پر کنید. آیا تفاوت قابل‌اعتنایی در نتایج مشاهده می‌کنید؟ کدام مدل بهتر عمل کرده است؟

size	w	n	میانگین شباهت کسینوسی در فایل **antonyms.txt**
۱۰۰	۵	۵	
۵۰	۵	۵	

h.veisi@ut.ac.ir