بسم الله الرحمن الرحیم

دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

بازیابی پیشرفته‌ی اطلاعات

نیم‌سال دوم تحصیلی ۰۱-۰۰

تمرین اول: عبارات منظم
ناشناس کردن اطلاعات شخصی متن

محمدمهدی ابوترابی ۹۸۱۰۵۵۵۷
یاسمن زلفی موصلو ۹۸۱۰۵۷۹۵
فاطمه عسگری ۹۸۱۰۵۹۲۱

مقدمه

با توجه به افزایش روزافزون داده‌های دیجیتالی و نگرانی‌هایی در باب حفاظت از حریم شخصی افراد، هدف ما در این تمرین ناشناس کردن اطلاعات شخصی همچون نام افراد، نام شرکت ها، تاریخ تولد، آدرس، شماره تلفن و... بود که برای تشخیص چنین اطلاعاتی، علاوه بر جمع‌آوری داده‌ی کافی و مورد نیاز، از عبارات منظم کمک گرفتیم. در ادامه ساختار کلی پروژه به همراه توضیحات هر بخش و همچنین نمونه‌های ورودی و خروجی آورده شده است.

بررسی ساختار و نحوه‌ی کارکرد پروژه

بخش اصلی منطق پروژه فایل‌های detector هستند که در پوشه‌ی perdonal_information_hider قرار دارند. هر یک از کلاس‌ها وظیفه‌ی ناشناس کردن یک نوع اطلاعات را بر عهده دارند؛ به طور مثال number_detector با گرفتن متن ورودی، شماره‌های شخصی مانند شماره تلفن، شماره حساب و کد ملی را تشخیص داده و آن‌ها را پنهان می‌کند.
تمامی detectorها به ترتیب در تابع run که در فایل main قرار دارد صدا زده می‌شوند و اطلاعات را پنهان و با تگ مناسب جایگزین می‌کنند.

- فایل address_detector: این فایل حاوی کلاس AddressDetection است که در آن از کتابخانه‌ی parsi.io بخش تشخیص آدرس کمک گرفتیم اما از آنجایی که منطق و رجکس‌های این کتابخانه مشکلاتی داشت، رجکس آن را کمی تغییر دادیم و بهبود بخشیدیم. همچنین دیتاستی که برای بخش تشخیص آدرس این کتابخانه بود را کامل‌تر کردیم تا در نهایت با صحت بیش‌تری آدرس‌ها را تشخیص داده و پنهان کنیم.
در تصویر زیر نمونه‌ی ورودی و خروجی این کلاس را مشاهده می‌کنید:

```
In[5]: print(AddressDetection().hide_address('من در خیابان آزادی تهران زندگی می‌کنم.'))
من در <#address#> زندگی می‌کنم.
```

- فایل birth_date_detector: این فایل حاوی کلاس BirthDateDetection است که تاریخ تولد را تشخیص و آن را پنهان می‌کند. این کلاس ابتدا به کمک کتابخانه‌ی parstdex تاریخ را به هر فرمتی که باشد تشخیص می‌دهد. سپس اگر در ۲۰ حرف قبلی تاریخ، کلمات کلیدی‌ای مثل تولد، متولد یا ولادت یا در ۲۰ حرف بعدی کلمات کلیدی‌ای مثل به دنیا، زاده یا متولد شد آمده باشد، آن تاریخ را به عنوان تاریخ تولد تشخیص داده و با تگ <#birth_date#> جایگزین می‌کند. تاریخ‌های دیگر چون اطلاعات شخصی به حساب نمی‌آیند پنهان نمی‌شوند.
در تصویر زیر نمونه‌ی ورودی و خروجی این کلاس را مشاهده می‌کنید:

```
In[2]: from personal_information_hider import *
In[3]: print(BirthDateDetection().hide_birth_dates('سلام تولد من روز یکشنبه است.'))
سلام تولد من <#birth_date#> است.
```

- فایل company_detector: این فایل حاوی کلاس CompanyDetection است که وظیفه‌ی تشخیص و پنهان کردن نام شرکت‌ها را بر عهده دارد. در این بخش ما ابتدا از سایتی که متعلق به برنامه توسعه زیست بوم شرکت های خلاق است و اسامی شرکت‌های ایرانی در آن ثبت شده است، نام حدودا ۱۱۰۰ شرکت ایرانی را کراول کردیم. به علاوه نام تعداد زیادی از شرکت‌های مطرح دنیا را از سایت ویکی‌پدیا کراول کردیم. تمام این اسامی را به عنوان دیتاست قرار دادیم. کار این کلاس به دو بخش تقسیم می‌شود. اولا اگر هریک از نام‌های جمع‌آوری شده در متن آمده باشد، آن را تشخیص داده و پنهان می‌کند. ثانیا اگر هر یک از کلمات شرکت، موسسه، سازمان یا کمپانی آمده باشد، از POS tagging استفاده کرده و اگر ترکیب مضاف و مضاف‌الیه بود، آن مضاف‌الیه را که در واقع

با احتمال بسیار بالایی نام شرکت است اما ممکن است در آن دیتاست نیامده باشد به عنوان نام شرکت تشخیص داده و پنهان می‌کند.
در تصویر زیر نمونه‌ی ورودی و خروجی این کلاس را مشاهده می‌کنید:

```
In[4]: print(CompanyDetection().hide_companies_names('سلام من در شرکت اسنپ کار می‌کنم. این شرکت خیلی خوب است.'))
سلام من در شرکت <#company_name#> کار می‌کنم. این شرکت خیلی خوب است.
```

توجه شود که همانطور که توضیح داده شد در نمونه‌ی بالا، نام «اسنپ» چه در دیتاست باشد چه نباشد با مکانیزم گفته‌شده به عنوان نام شرکت تشخیص داده می‌شود اما کلمه‌ی «خیلی» که بعد از شرکت آمده به عنوان نام شرکت تشخیص داده نمی‌شود.
نکته‌ی دیگری که قابل ذکر است این است که تابع run موجود در فایل main یک ورودی flag می‌گیرد که با آن معلوم می‌شود از این مدل POS tagger استفاده بشود یا خیر؛ چراکه این مدل که از مدل POS tagger کتابخانه‌ی hazm گرفته شده است ممکن است در برخی سیستم‌ها برای اجرا شدن کاربر را با مشکلاتی روبرو کند اما اگر استفاده شود قدرت تشخیص مدل ما بیش‌تر و بهتر خواهد بود.

- فایل email_detector: این فایل حاوی کلاس EmailDetection است که از email detector موجود در کتابخانه‌ی parsi.io استفاده و آدرس ایمیل‌های تشخیص داده شده را با تگ <#email_address#> جایگزین می‌کند.
در تصویر زیر نمونه‌ی ورودی و خروجی این کلاس را مشاهده می‌کنید:

```
In[6]: print(EmailDetection().hide_emails('سلام ایمیل من mahdi.aboots@gmail.com است.'))
سلام ایمیل من <#email_address#> است.
```

- فایل url_detector: این فایل حاوی کلاس UrlDetection است که از url detector موجود در کتابخانه‌ی parsi.io استفاده می‌کند اما از آنجایی که این کتابخانه آدرس ایمیل‌ها را نیز به عنوان url تشخیص می‌دهد، ما در کلاس UrlDetection با رجکس مناسب برای آدرس‌های ایمیل، آن‌ها را از لیست urlها حذف کردیم. در نهایت این کلاس urlهای تشخیص داده شده را با تگ <#url_address#> جایگزین می‌کند.
در تصویر زیر نمونه‌ی ورودی و خروجی این کلاس را مشاهده می‌کنید:

```
In[8]: print(UrlDetection().hide_url('سلام آدرس سایت من google.com است.'))
سلام آدرس سایت من <#url_address#> است.
```

- فایل name_detector: این فایل حاوی کلاس NameDetection است که وظیفه‌ی تشخیص نام و نام خانوادگی را بر عهده دارد. در این بخش نیز از سایت‌های مختلف اسامی فارسی و نام‌های خانوادگی را کراول کردیم و به عنوان دیتاست قرار دادیم. مراحل crawling با جزئیات کامل در نوتبوک name_part.ipynb واقع در فولدر crawlers موجود است. کلاس NameDetection با استفاده از این دیتاست و عبارات منظم مناسب، اسامی را تشخیص داده و پنهان می‌کند. توجه شود که در این بخش به دلیل وجود حالات خاص بسیار زیاد، استفاده از POS tagger نیز کمک چندانی نمی‌کرد لذا از آن استفاده نکردیم و سعی کردیم رجکس‌ها و دیتاست را تقویت کنیم.
در تصویر زیر نمونه‌ی ورودی و خروجی این کلاس را مشاهده می‌کنید:

```
In[9]: print(NameDetection().hide_person_name('.من محمدمهدی امیری هستم'))
.من <#person_name#> هستم
```

- فایل number_detector: این فایل حاوی کلاس NumberDetection است که با استفاده از عبارات منظم مناسب اعداد انگلیسی و فارسی به تشخیص انواع شماره‌ها می‌پردازد. این کلاس از tokenize کردن کلمات استفاده می‌کند و به سه کلمه‌ی قبل و دو کلمه‌ی بعد هر شماره نگاه می‌کند. سپس بر اساس کلمات کلیدی معینی تشخیص می‌دهد شماره از چه نوعی است؛ به طور مثال اگر کلماتی مثل شماره منزل، موبایل، زنگ، تلفن یا تماس استفاده شده بود، آن شماره را به عنوان شماره تلفن تشخیص می‌دهد و با تگ <#phone#> جایگزین می‌کند. هم‌چنین اگر کلمات کلیدی بانک یا حساب آمده بود شماره را به عنوان شماره حساب پنهان می‌کند و اگر کلمات کلیدی کد ملی یا ملی آمده بود شماره را به عنوان کد ملی تشخیص می‌دهد. اگر هیچ یک از کلمات ذکرشده به کار نرفته بود، شماره را پنهان و تگگذاری نمی‌کند چراکه اطلاعات شخصی به حساب نمی‌آید.
در تصویر زیر نمونه‌ی ورودی و خروجی این کلاس را مشاهده می‌کنید:

```
In[10]: print(NumberDetection().hide_personal_numbers('.شماره تلفن من ۰۲۱۷۷۸۶۶۶ است'))
.است <#phone#> شماره تلفن من
In[11]: print(NumberDetection().hide_personal_numbers('.شماره حساب من ۰۲۱۷۷۸۶۳۳۳۶۶ است'))
.است <#account_number#> شماره حساب من
```

- فایل main: در این فایل تابع run قرار دارد که به عنوان ورودی یک address می‌گیرد که متن موجود در آن فایل address را لود می‌کند و علاوه بر نرمالایز کردن تک تک detectorها را روی متن اعمال کرده و خروجی را برمی‌گرداند. هم‌چنین می‌تواند خروجی را در فایلی که آدرسش می‌تواند به عنوان پارامتر ورودی به تابع run داده شود نیز بنویسد. درباره‌ی use_pos هم که یکی دیگر از پارامترهای ورودی تابع run است در بخش‌های قبل توضیح داده شد.

از دیگر فایل‌ها و پوشه‌های موجود در پروژه می‌توان به پوشه‌ی crawlers اشاره کرد که شامل اسکریپت‌های کراول کردن نام کمپانی‌ها و فایل نوت‌بوک پروسه‌ی کراول کردن اسامی و نام‌های خانوادگی می‌باشد.

کتابخانه‌ی parsi.io را نیز در کنار پروژه‌ی خود قرار دادیم تا در جاهای مورد نیاز از آن استفاده کنیم.

در پوشه‌ی personal_information_hider که همه‌ی detectorها قرار دارند، یک پوشه‌ی resources هم موجود است که شامل فایل‌های دیتاست و یک فایل مدل POS tagger می‌باشد.

سه فایل تست در فرمت txt نیز موجود است که در ژوپیتر نوت‌بوک Test.ipynb تک تک تست شده‌اند. ورودی‌ها و خروجی‌های این سه تست در انتهای گزارش آمده است که می‌توانید مشاهده کنید.

فایل requirements.txt نیز حاوی تمامی پیشنیازهای لازم برای اجرای پروژه است.

توجه: همانطور که گفته شد مدل POS tagger بر روی سیستم عامل ویندوز ممکن است به مشکل بربخورد و در صورتی که می‌خواهید از آن استفاده کنید بهتر است از سیستم عامل مبتنی بر لینوکس یا macOS استفاده کنید و یا پروژه را بر روی google colab اجرا کنید.

تست‌کیس‌ها

به دلیل به هم ریختگی متن فارسی در نوت‌بوک، ورودی‌ها و خروجی‌ها از داخل نوت‌بوک کپی شده و در ادامه آورده شده‌اند.

- تست‌کیس اول:
  - ورودی:

سلام. نام من فاطمه عسگری است.امروز چهارم اردیبهشت ماه سال ۱۴۰۱ می باشد.من در حال کار روی تمرین بازیابی هستم و میوه های بسیار خوش مزه در کتاب خانه هست.
اینجانب فاطمه عسگری به شماره ملی ۱۲۷۳۴۸۹۹۰۱ هستم و شماره حساب بانک ملت 622133333333333333 است.
أدرس ایمیل بنده f.asgari2001@gmail.com است و هم چنین می توانید به وب سایت wwww.google.com مراجعه کنید.
وب سایت www.google.com یک وب سایت جامع است. علی به شرکت گوگل رفت.
شماره تلفن من ۰۲۱۳۳۴۷۸ است و من در خیابان آبشار تهران زندگی می‌کنم.

  - خروجی:

سلام. نام من <#person_name#> است. امروز چهارم اردیبهشت ماه سال ۱۴۰۱ می‌باشد. من در حال کار روی تمرین بازیابی هستم و میوه‌های بسیار خوش مزه در کتاب خانه هست.
اینجانب <#person_name#> به شماره ملی <#national_number#> هستم و شماره حساب بانک ملت <#account_number#> است.
أدرس ایمیل بنده <#email_address#> است و هم چنین می توانید به وب سایت <#url_address#> مراجعه کنید.
وب سایت <#url_address#> یک وب سایت جامع است. <#person_name#> به شرکت <#company_name#> رفت.
شماره تلفن من <#phone#> است و من در <#address#> زندگی می‌کنم.

- تست کیس دوم:
  - ورودی:

زهرا رحیمی هستم. من علاقه زیادی به کار در شرکت تپسی دارم. من با این شرکت از طریق دوستم فاطمه عسگری آشنا شدم.
از آنجا که این شرکت در خیابان ایران، تهران واقع شده است؛ به دانشگاه ما نزدیک بوده و من برای رفت و آمد با مشکل چندانی رو به رو نخواهم بود.
شماره تماس من 09123456789 و ایمیل من zahra.rahimi@gmail.com است. در وبلاگ خودم یعنی zahra.com بیشتر به علایقم پرداختم. مریم میرزاخانی در ۲۲ اردیبهشت ۱۳۵۶ در تهران به دنیا آمد.

  - خروجی:

<#person_name#> هستم. من علاقه زیادی به کار در شرکت <#company_name#> دارم. من با این شرکت از طریق دوستم <#person_name#> آشنا شدم.
از آنجا که این شرکت در <#address#> واقع شده است؛ به دانشگاه ما نزدیک بوده و من برای رفت و آمد با مشکل چندانی رو به رو نخواهم بود.
شماره تماس من <#phone#> و ایمیل من <#email_address#> است. در وبلاگ خودم یعنی <#url_address#> بیشتر به علایقم پرداختم. <#person_name#> در <#birth_date#> در <#address#> به دنیا آمد.

- تست‌کیس سوم:

- ورودی:

  پروفسور شهره امیری یکی از اساتید دانشکده مهندسی کامپیوتر دانشگاه علم و صنعت هستند که در زمینه ی بینایی کامپیوتری، پردازش تصویرو ویدیو فعالیت می کنند. آدرس پستی ایشان کاشان، خیابان آزادی است. برای ارتباط با ایشان می توانید با شماره 0216166646 تماس بگیرید و یا به نشانی amiri@elm.edu ایمیل بزنید. آدرس صفحه شخصی ایشان نیز http://elm.edu/~amiri می‌باشد. ایشان در موسسه پارس افزار سروین کار می‌کنند.

- خروجی:

  پروفسور <#person_name#> یکی از اساتید دانشکده مهندسی کامپیوتر دانشگاه علم و صنعت هستند که در زمینه‌ی بینایی کامپیوتری، پردازش تصویرو ویدیو فعالیت می‌کنند. آدرس پستی ایشان <#address#> است. برای ارتباط با ایشان می‌توانید با شماره <#phone#> تماس بگیرید و یا به نشانی <#email_address#> ایمیل بزنید. آدرس صفحه شخصی ایشان نیز <#url_address#> می‌باشد. ایشان در موسسه <#company_name#> کار می‌کنند.