*921531110*

**علی فاضلی**

**جناب آقای دکتر فیضی درخشی**

|  |
| --- |
|  |
| گزارش پروژه دوم درس مبانی پردازش زبان گفتار |
|  |
|  |

**استاد محترم:**

گزارش پروژه دوم درس مبانی پردازش زبان گفتار

علی فاضلی

**هدف پروژه :**

خواندن یک متن با زبان فارسی از یک فایل متنی، استاندارد سازی( Normalization) و تقسیم آن به واژه های سازنده یا به عبارت دیگر Tokenize کردن آن بصورت هوشمند و در نهایت نوشتن خروجی متناسب با قالب Json

پروزه در چهار فاز اصلی زیر پیاده سازی شده است:

1. خواندن ورودی از فایل متنی و انجام پیش پردازش های مربوطه
2. استاندارد سازی متن (Normalization)
3. Tokenization
4. خروجی قالب Json

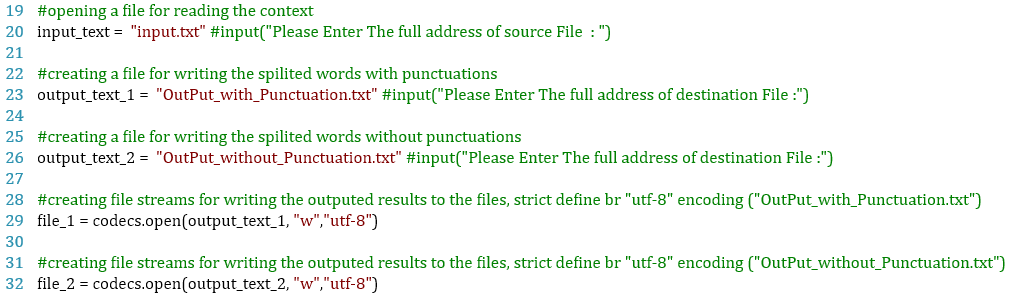
**فاز اول.خواندن ورودی از فایل متنی :**

برای اینکار از زبان برنامه نویسی پایتون استفاده شد. از کتابخانه ی codecs برای خواندن فایل متنی با الگوریتم کد گذاری Unicode استفاده شد.

بخشی از متن ورودی در ذیل قابل مشاهده می باشد:

« اعْلَمُوا أَنَّمَا الْحَياةُ الدُّنْيا لَعِبٌ وَ لَهْوٌ وَ زينَةٌ وَ تَفاخُرٌ بَيْنَکُمْ (ك) وَ تَکاثُرٌ فِي الْأَمْوالِ وَ الْأَوْلادِ کَمَثَلِ غَيْثٍ أَعْجَبَ الْکُفَّارَ نَباتُهُ ، ثُمَّ يَهيجُ فَتَراهُ مُصْفَرًّا ثُمَّ يَکُونُ حُطام \_\_ اً و فِي الْآخِرَةِ عَذابٌ شَدييٌد وَ مَغْفِرَةٌ : مِنَ اللَّهِ وَ رِضْوانٌ 09379941103 وَ مَا الْحَياةُ الدُّنْيا إِلاَّ مَتاعُ »

این متن از قصد آیه شریفه ای از قران کریم انتخاب شده که حاوی یک عبارت عربی به انظمام اعداد و برخی Punctuation mark ها بود.

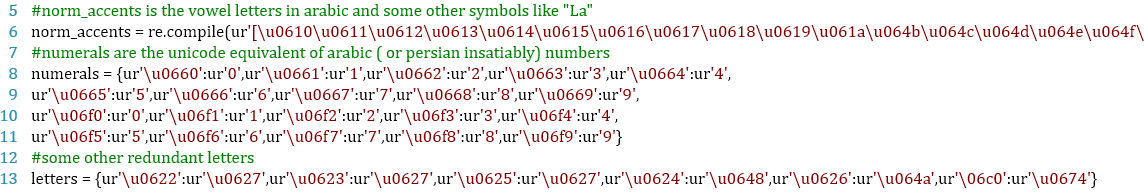
این متن در یک فایل متنی بنام input.txt ذخيره گردید. نحوه خواندن فایل متنی در پروژه به شکل زیر می باشد:

تصویر 1- گرفتن ورودی متنی توسط کلاینت پایتون و ایجاد فایل استریم برای نوشتن خروجی ها

همانطور که در تصویر مشخص است، مسیر فایل ورودی در متغیر input\_text ذخیره شده و دو متغیر output\_text\_1 و output\_text\_2 برای ایجاد دو فایل متفاوت برای رشته tokenize شده همراه و یا بدون علائم نگارشی در نظر گرفته شده اند.

**فاز دوم: استاندارد سازی متن (Normalization) :**

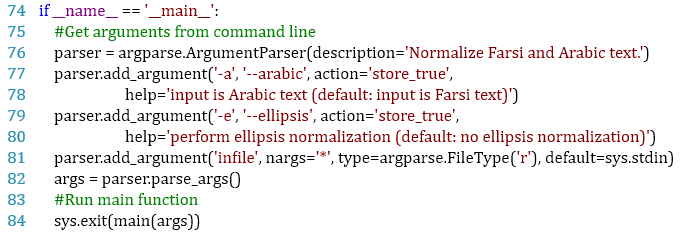
در این مرحله متن ورودی (که در فایل input.txt) قرار داشت، به عنوان ورودی کلاس Normalizer.py در نظر گرفته میشود. توجه شود که کلاس Normalizer با استفاده از پایتون نسخه 2.7 توسعه داده شده است. کلاس Normalizer از بخش های زیر تشکیل شده است:



تصویر 2- معرفی حروف مختلف عربی

در این بخش ابتدا کد یونیکد حروف عربی ای که احتمال اشتباه وارد شدن آنها با حروف مشابه (از نظر نگارشی) آنها در زبان فارسی وجود دارد در متغیر norm\_accents ذخیره می شوند.عبارت re.compile نیز متدی از کلاس regular expression ها می باشد که لیست یونیکد هارا قبل از تفسیر (Interpret) کردن برنامه به نحوی کامپایل می کند تا دسترسی و کار با آنها بصورت real-time راحت صورت پذیرد.

در ادامه یونیکد کلمات فارسی ( و یا معادل آن در عربی) و دیگر علائم و نشانه های نگارشی نیز به لیست letters اضافه می شوند.

البته شایان ذکر است که اسکریپت به شکل قایل توجهی مستند سازی شده و کامنت گذاری شده است. اما به اختصار به معرفی عملکرد دیگر بخش های آن نیز می پردازیم.

تصویر 3- نحوه شروع و اجرای اسکریپت

اسکریپت دو مود اجرای مختلف دارد. یکی برای حذف علائم نگارشی زائد در مرحله پیش پردازش و دیگری نرمال سازی متن.

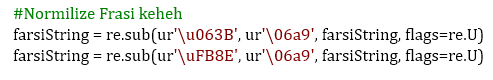
برای اجزای مرحله حذف علائم نگارشی از دستور زیر در کنسول استفاده می کنیم:

python Normalizer.py -e input.txt > input\_without\_ellipsis.txt

بعد از انجام این مرحله و تولید متن بدون علائم نگارشی نوبت به نرمال سازی متن میرسد. این کار با دستور زیر انجام میپذیرد:

python Normalizer.py -a input\_without\_ellipsis.txt > Normalized.txt

نحوه انجام نرمال سازی هم بدین صورت است که کل متن از فایل .txt خوانده شده و با استفاده از تابع re.sub که دو آرگومان دارد، آرگومان دوم جانشین آرگومان اول می گردد. به عنوان مثال برای تصحیح حرف "ک" عربی به حرف "ک" معیار در زبان فارسی در شکل شماره چهارم داریم:



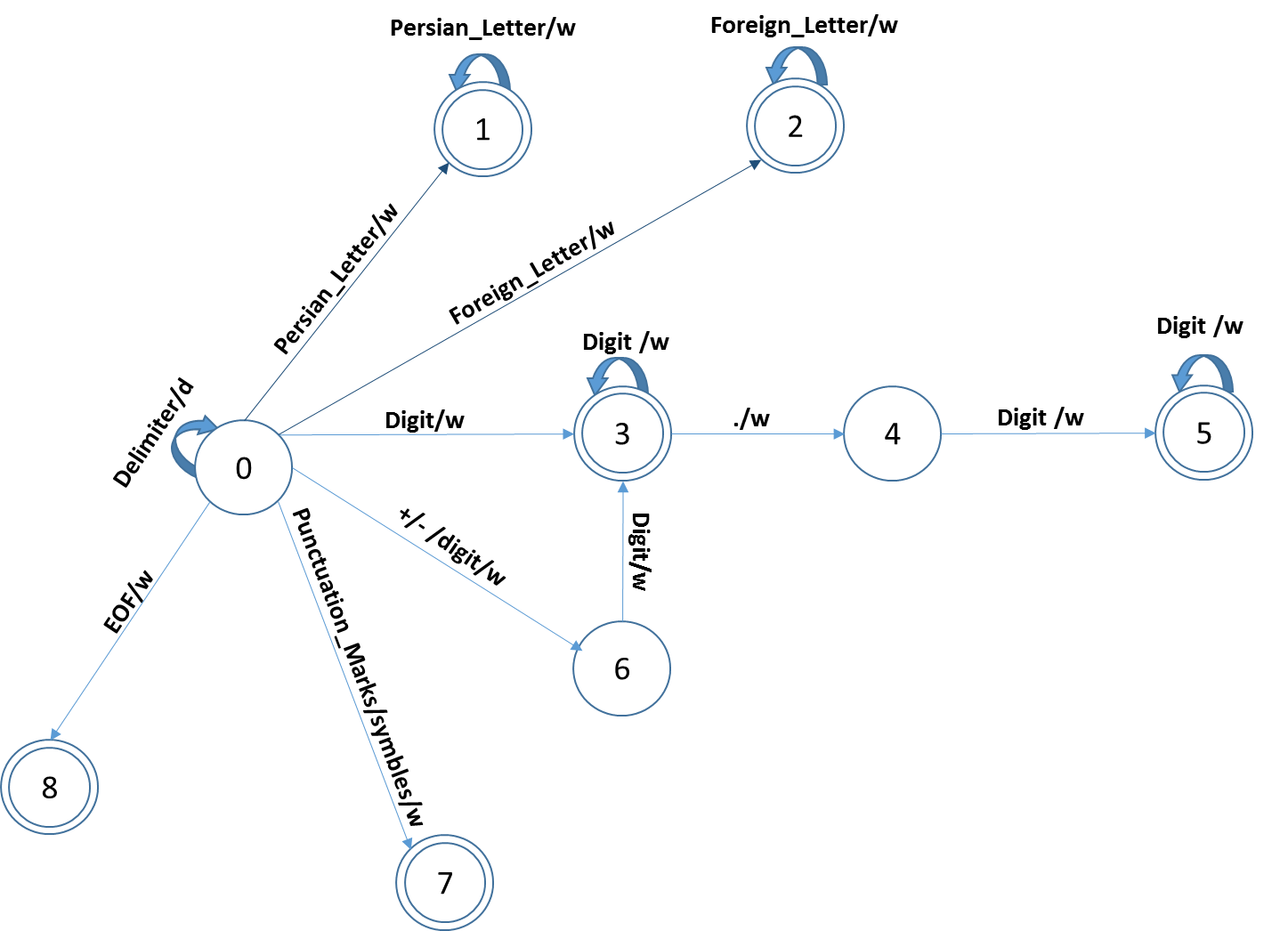
تصویر 4- جانشین کردن حرف کاف معیار به جای دیگر کاف هاش مشابه

قابل ملاحظه است که یونیکد شماره \u063B و \uFB8E (که معادل انواع کاف در زبان عربی می باشند) با یونیکد مربوط به حرف کاف معیار زبان فارسی تعویض شده اند. در آخر تمامی پیش پردازش های انجام شدهبا قالب utf-8 انکود شده و در فایل خروجی قرار می گیرند. خروجی کلاس Normalizer بعد از انجام دو مرحله به شکل زیر می باشد. بدلیل طولانی بودن متن در یک خط و برای منظم تر دیده شدن از کاراکتر ها خروجی در دو تصویر قابل مشاهده می باشد.

C:\Users\K550 VX\Documents\ShareX\Screenshots\2017-03\ApplicationFrameHost_2017-03-02_11-17-28.png C:\Users\K550 VX\Documents\ShareX\Screenshots\2017-03\ApplicationFrameHost_2017-03-02_11-17-50.png

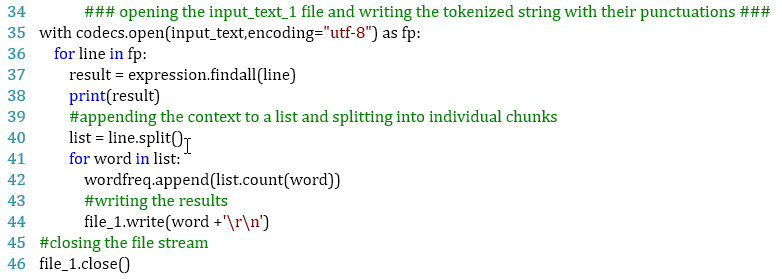
همانطور که قابل مشاهده است تمامی علائم نگارشی حذف و حرف های بیگانه با زبان فارسی از متن حذف شده اند. البته این امر مخصوص اشتباه تایپی و یا ناهماهنگی های سیستم های انکودینگ می باشد. مشخص است که این امر به منظور فارسی کردن نگارش متون عربی نمی باشد و صرفا جهت نشان دادن توانایی کد مربوطه می باشد.

در نهایت نمایش finite state machine مربوط به بخش پیش پردازش این پروژه به شکل زیر می باشد:



**فاز سوم : Tokenization :**

بعد از اینکه مراحل نرمال سازی انجام گرفت نیاز است که متن به بخش های سازنده اش تجزیه شود. که این نیازمند استفاده از اسکریپت Tokenizer.py می باشد. این عمل با دستور python Tokenizer.py از کنسول انجام می گیرد. نام فایل متنی خروجی از فاز قبل (Normalized.txt) در اسکریپت Tokenizer.py از قبل نوشته شده یا به عبارتی Hardcode شده است. از تشریح قسمت های گرفتن ورودی و مسائل پیش پا فتاده صرف نظر کرده و به قسمت انجام پردازش می پردازیم. قسمت اصلی پردازش در تصویر شماره 5 قابل مشاهده می باشد:

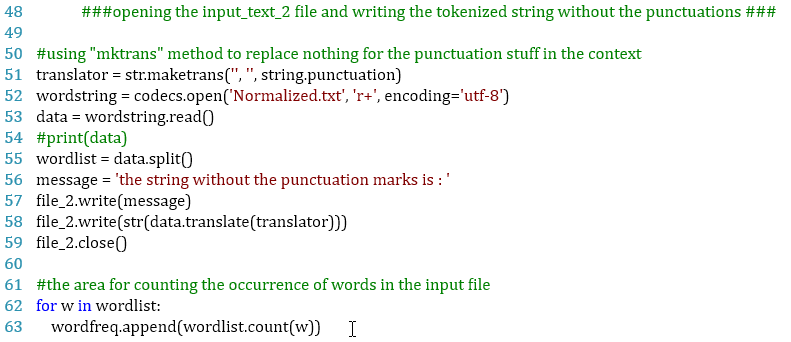


تصویر 5- پردازش متن به منظور جدا سازی عناصر تشکلی دهنده ان با در نظر گرفتن علائم نگارشی

و در تصویر شماره 6 ام قسمت مربوط به انجام پردازش بر روی متن و محاسبه تعداد تکرار هم کلمه را بدون در نظر گرفتن علائم نگارشی داریم. شایان ذکر است که کد قابلیت جدا سازی IP را با استفاده از Regular Expression دارد.

C:\Users\K550 VX\Documents\ShareX\Screenshots\2017-03\ApplicationFrameHost_2017-03-02_19-28-40.png

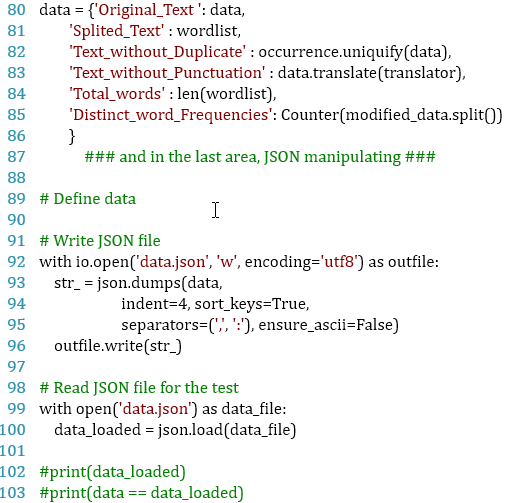
تصویر 6- تشخیص و جداسازی IP در کلاس Tokenizer



تصویر 7- پردازش متن به منظور جداسازی عناصر اصلی تشکیل دهنده آن بدون در نظر گرفتن علائم نگارشی

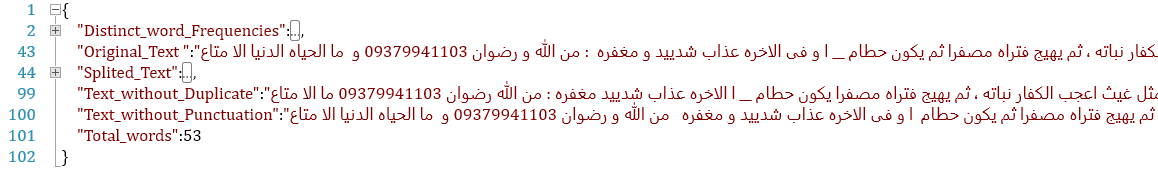
و سپس تمامی مقادیر اندازه گیری شده درکنسول پرینت و قابل مشاهده می باشد.

**فاز چهارم: نوشتن و نمایش خروجی مقادیر در قالب Json :**

در مرحله آخر و در قسمت آخر کلاس Tokenizer.py تمامی مقادیر محاسبه شده را با قالب Json به خروجی می بریم. قطعه کد تشکیل قالب Json به شکل زیر می باشد:

تصویر 8- تشکیل قالب Json

با اجرای اسکریپت Tokenizer.py به طور اتوماتیک مقادیر محاسبه شده در یک فایل بنام data.json ذخیره می شود که قالب خلاصه شده آن در شکل هشتم قابل مشاهده است:



تصویر 9-قالب خروجی فایل DATA.JSON