**Deep learning to filter SMS Spam**

نگارنده: بهارک احمدی پور

تابستان 99

**مقدمه:**

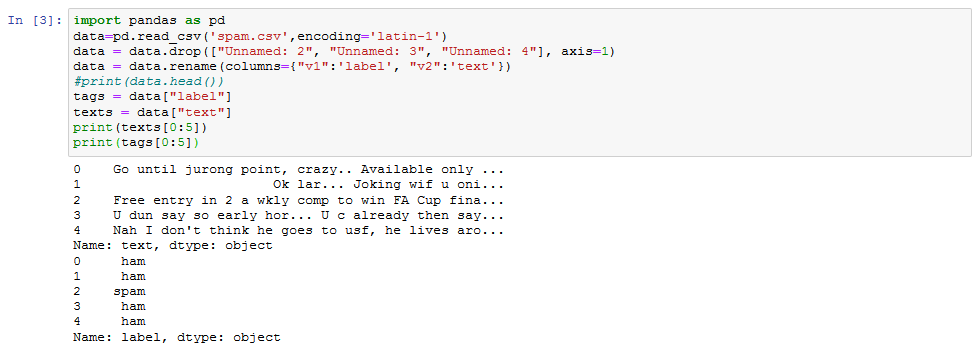
در این گزارش چندین مدل برای تشخیص SMSهای Spam ارائه و پیاده سازی شده است. برای این کار از مدل از قبل آموزش داده شده GloVe (مخفف Global Vector) برای encode کردن متن پیامها استفاده می شود. این مدل برای هر کلمه یک بردار عددی تولید می کند که نشان دهنده بار معنایی کلمه و میزان ارنباط معنایی آن با سایر کلمات است. مدل GloVe دارای ورژن های متفاوت و خروجی های متفاوت است. اینجا از ورژن 6B استفاده شده و در این ورژن بسته به فایل انتخابی بردارهای 50بعدی، 100 بعدی، 200 بعدی و 300 بعدی تولید می شود. در این مقاله از خروجی 100 بعدی برای نمایش کلمات استفاده شده است.

**پیاده سازی:**

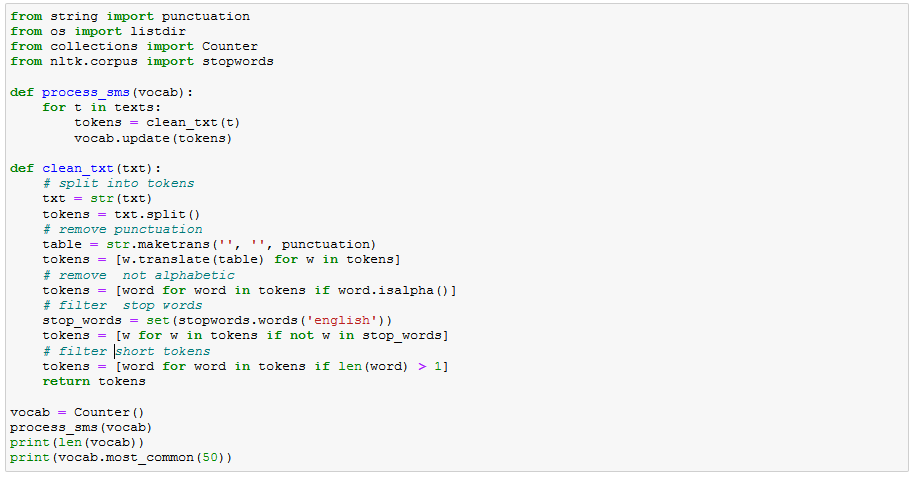
این برنامه توسط Jupyter Notebook نوشته و اجرا گردیده است.

در ابتدا می بایست اطلاعات دیتاست را خوانده و آن را در یک data Frame از کتابخانه Pandas ذخیره کنیم. در این مقاله از دیتاست spam.csv استفاده شده است. دیتاست شامل 5 ستون است که ما فقط با دو ستون آن نیاز خواهیم داشت: ستونی که نشاندهنده متن SMS است، و ستونی که نشاندهنده وضعیت Spam بودن آن است. لذا توسط دستور drop سه ستون آخر را حذف می کنیم و توسط دستور rename، به ستونهای باقیمانده، نام مناسب تخصیص می دهیم.

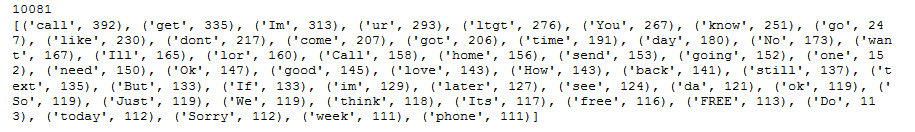
موارد گفته شده در کد زیر انجام شده است:



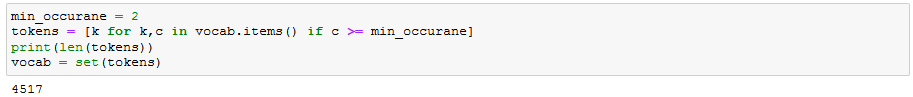
حال تمامی متن های SMS خوانده شده و در لیست texts قرار گرفته اند. قبل از انجام هر نوع پردازشی، ابتدا باید اطلاعات متنی پاکسازی شوند و اصطلاحاً پیش پردازش گردند. عملیاتی همچون حذف فضاهای خالی، حذف punctuationها، حذف کلمات غیر الفبایی، حذف Stop wordها (کلمات پر تعدادی در زبان که به تنهایی معنای خاصی ندارند مانند the، a، an و .... که در صورت حذف نکردن، به دلیل فرکانس بالای استفاده این لغات، اهمیت آنها بالا رفته در حالیکه بار معنایی ندارند و دقت پردازش را پایین خواهند آورد). برای به دست آوردن پر تعدادترین لغات، از counter (به نام vocab) استفاده شده است. متن SMS ها پیش پردازش شده و کلمات آن در شئ counter ذخیره می گردد. سپس برای دیدن نتیجه، 50 کلمه پر کاربرد در متن ها چاپ شده است:



توسط اجرای قطعه کد فوق، خروجی زیر به دست می آید:



همانطور که دیده می شود، تعداد 10081 کلمه در کل SMS های دیتاست تشخیص داده شد. در انتهای vocab، تعداد زیادی از کلمات دیده می شوند که فقط و فقط یکبار استفاده شده اند و عملا جزء کلمات مهم و تاثیر گذار نیستند. حذف این کلمات از لیست vocab باعث کاهش زمان پردازش، بدون تاثیر منفی بر روی دقت، خواهد شد. در قطعه کد زیر این کار انجام شده است:

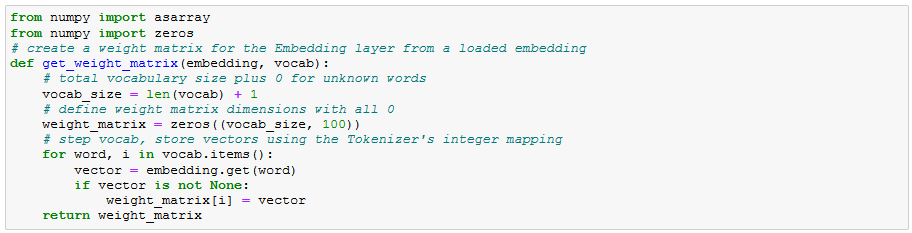


همانطور که در خروجی دیده می شود، تعداد کل کلمات موجود در vocab، پس از حذف کلماتی که فقط یکبار دیده شده اند، 4517 کلمه شده است، که کاهش چشمگیری است.

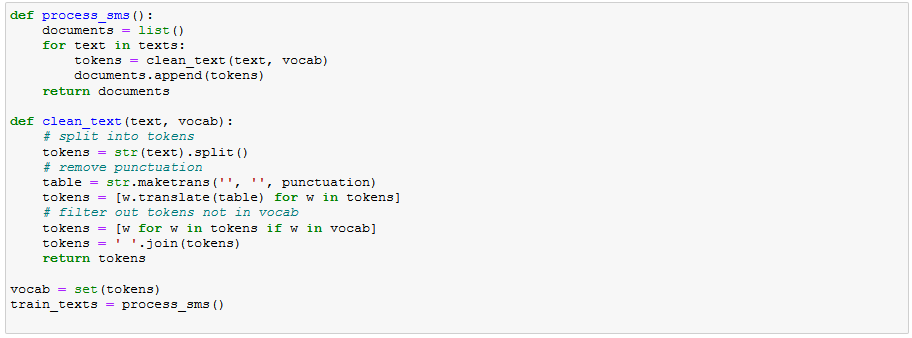
حال تابعی تعریف می کنیم که فایل embedding را خوانده و تبدیل به یک dictionary کند که در آن کلید، خود کلمه، و value بردار embed شده مربوط به آن کلمه می باشد :



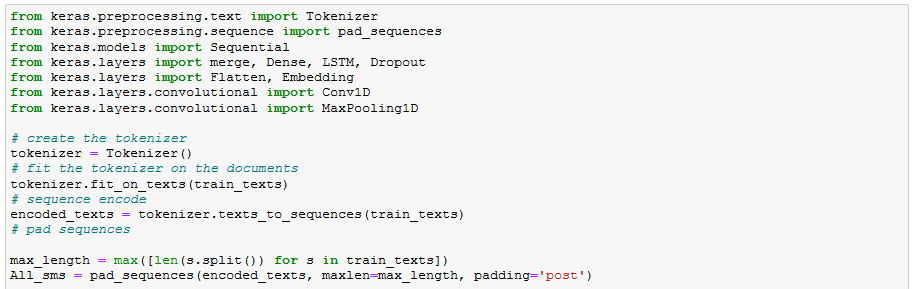
تابع get\_weight\_matrix ماتریس ورودی برای لایه embedding مدل آموزشی را تولید می کند. در ابتدا یک ماتریس از مقادیر صفر ساخته شده. از آنجا که در این مقاله از GloVe نسخه 100 بعدی استفاده شده، تعداد ستونهای ماتریس برابر 100 تعریف شده و تعداد سطرهای ماتریس به اندازه تعداد کلمات موجود در vocab است. سپس برای هر کلمه بردار 100 تایی نظیر آن در ماتریس قرار داده شده است. اولین لایه در هر مدل آموزشی پردازش متن، لایه embedding است که هر کلمه متن را به بردار عددی تبدیل می کند. برای این تبدیل باید ماتریس اعداد مربوط به هر کلمه به عنوان ورودی به لایه داده شود. این ماتریس توسط تابع get\_weight\_matrix تولید می شود:



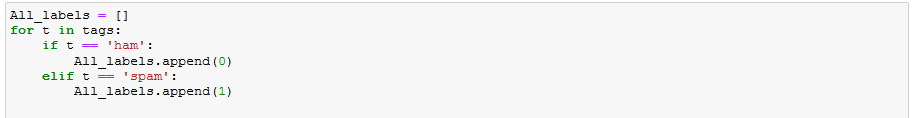
حال آماده پردازش SMSها و ساخت مدل هستیم. همان پیش پردازش هایی که برای ساخت vocab انجام دادیم را دوباره بر روی متن SMS ها انجام می دهیم ولی اینبار متن های پیش پردازش شده را در یک لیست از رشته ها به نام train\_texts ذخیره می کنیم.



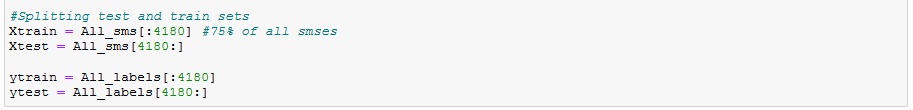
حال توسط ابزار Tokenizer از کتابخانه keras ، متن های موجود در train-texts را توکن می کنیم، encode می کنیم و از آنجا که طول SMSها با هم برابر نیستند و از طرف دیگر شبکه کانولوشن ورودی با طول های متفاوت را پشتیبانی نمی کند، باید طول تمام متن ها را توسط اضافه کردن 0 به انتهای متن، با هم برابر کنیم (به این عمل padding می گویند، و ما در اینجا صفرها را به انتها اضافه کردیم. –post padding). متن های padding شده ، در لیستی به نام All\_sms ذخیره شده اند.



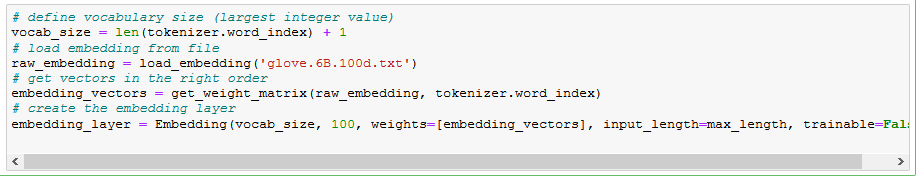
برچسب SMSها در دیتاست به صورت متنی دخیره شده اند (توسط کلمات spam و ham). که این اطلاعات نیز باید تبدیل به فرم عددی شوند. به همین دلیل از عدد 0 برای ham و عدد 1 برای spam استفاده شده است و این مقادیر در لیست All\_labels ذخیره شده اند:



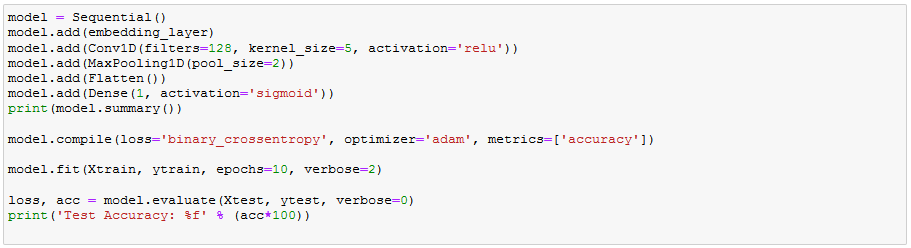
حال متن sms ها و برچسب آنها آماده اند. 75% از داده ها را برای آموزش و 25% از آنها را برای تست قرار می دهیم.



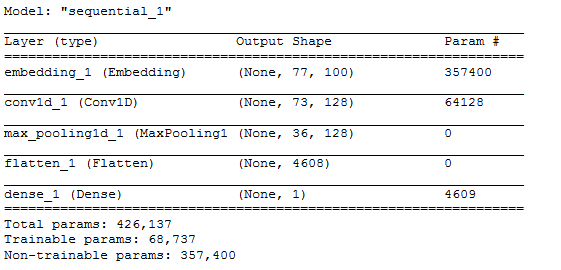
در اینجا از مدل از قبل آموزش داده شده GloVe برای encode کردن کلمات استفاده می شود. فایل متنی glove.6B.100d.txt از کلمات موجود در سایت Wikipedia استفاده کرده است و برای هر کلمه یک بردار 100 بعدی دخیره کرده است که توسط این مقادیر ارتباط معنایی کلمات و میزان نزدیک بودن هر کلمه به سایر کلمات مشخص می شود. این فایل به صورت رایگان در اینترنت قابل دانلود می باشد. این فایل توسط تابع load\_embeding که قبلا توضیح داده شد مورد استفاده قرار می گیرد سپس اطلاعات خوانده شده به عنوان ورودی به تابع get\_weight\_matrix (که پیشتر توضیح داده شد) داده می‌شود و ماتریس وزن مربوط به کلمات تولید می شود. سپس لایه embedding مدل ، توسط این ماتریس، ساخته می شود. از آنجا که مدل Glove یک مدل از قبل آموزش دیده شده است و نمی خواهیم مقادیر وزن ها در خلال آموزش تغییر کنند، مقدار پارامتر trainable را به False تغییر می دهیم.



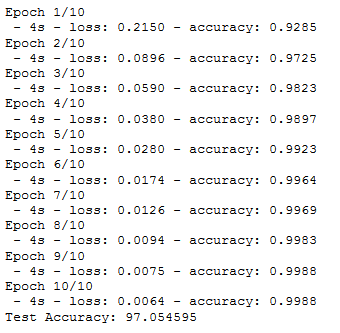
در مقاله مربوطه از چندین مدل برای کلاس بندی استفاده شد و نتیجه آنها با هم مقایسه شد. ابتدا مدل با یک لایه کانولوشن به صورت زیر تعریف شد. همانطور که پیشتر توضیح داده شد، لایه اول هر مدل پردازش متن، لایه embedding است. پس از آن لایه کانولوشن، سپس max-pooling ، flatten و در انتها نیز لایه dense قرار داد شد:



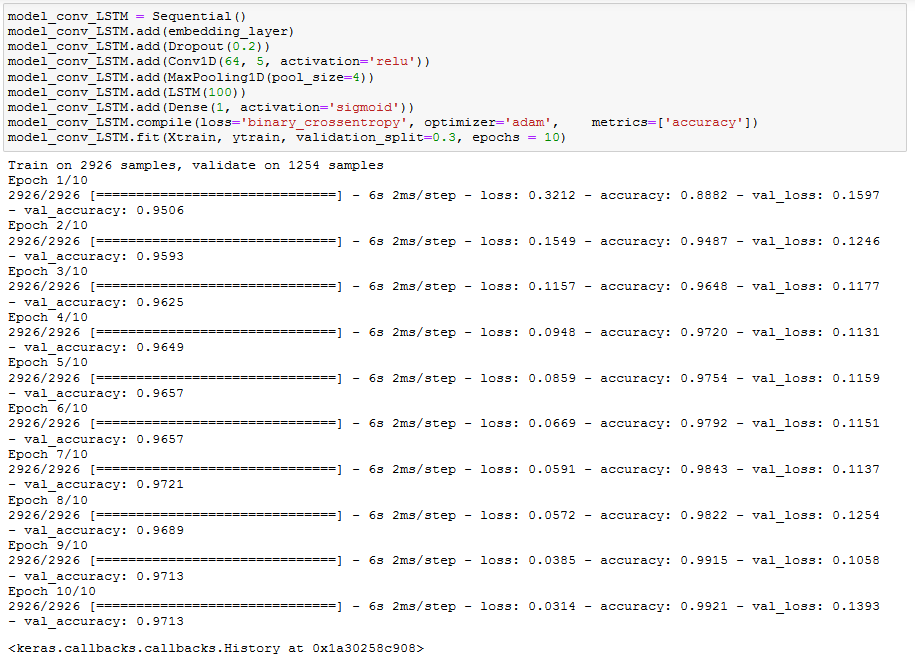
Summary مدل در شکل زیر دیده می شود:



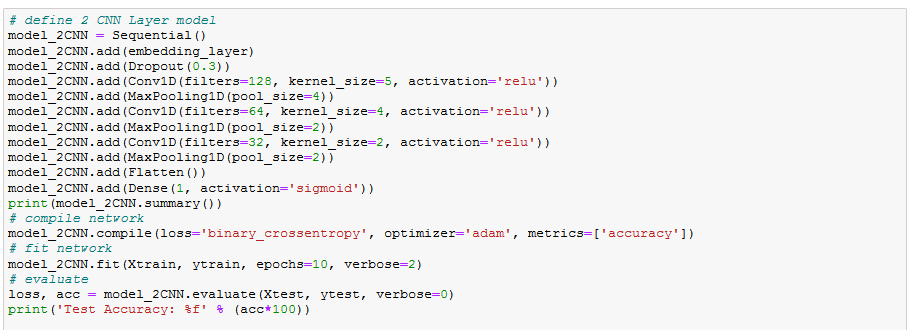
پس از ده epoch نتایج زیر حاصل شد:



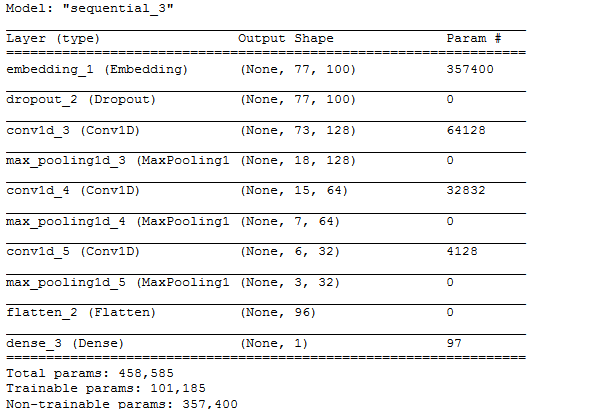
حال مدل دیگری شامل لایه کانولوشن و لایه LSTM تعریف شد و بر روی مدل تست شد و نتیجه زیر به دست آمد:



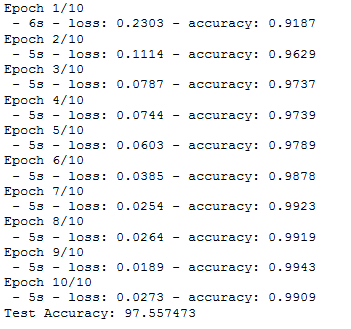
و در نهایت مدل دیگری متشکل از سه لایه کانولوشن مانند شکل زیر تعریف شد:



خلاصه مدل تعریف شده:



داده های آموزشی و آزمایشی به این مدل هم داده و نتایج زیر از آن گرفته شدند:



**نتیجه گیری:**

در این مقاله مدلی برای تشخیص Spam بودن SMSها پیاده سازی شد. ابتدا اطلاعات توسط مدل پیش آموزش داده شده GloVe که به صورت رایگان در دسترس می باشد، encode شد و توسط سه مدل متفاوت آموزش داده شده و نتایج آن با هم مقایسه شد. با مقایسه نتایج مشاهده شد که افزایش تعداد لایه های کانولوشن در این مدل تاثیر زیادی در بالا رفتن دقت تشخیص ندارد.