**FNDNet- A Deep Convolutional Neural Network for**

**Fake News Detection**

نگارنده: بهارک احمدی پور

تابستان 99

**مقدمه:**

در این گزارش مدلی برای تشخیص اخبار جعلی 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 می‌کند 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.

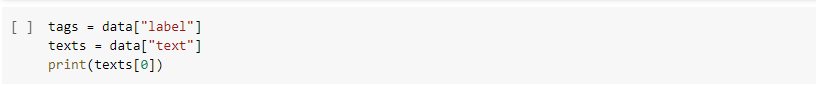
برای پیاده سازی این پروژه از محیط Google colab استفاده شده است. از آنجا که متن اخبار طولانی و پردازش آن زمانبر بوده و نیاز به حافظه RAM زیادی دارد، این محیط برای پیاده سازی انتخاب شده و در همان ابتدای کار پردازش را بر روی GPU تنظیم می کنیم. برای اینکار از منوی Run time، گزینه change runtime type را انتخاب کرده و از پنجره ظاهر شده گزینه GPU را انتخاب می کنیم.

**پیاده سازی:**

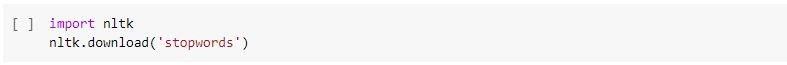
در ابتدا می بایست اطلاعات دیتاست را خوانده و آن را در یک data Frame از کتابخانه Pandas ذخیره کنیم. در اینجا از دیتاست fake news موجود در سایت kaggle استفاده شده که دارای دو فایل train.csv و test.csv است. ابتدا داده های موجود در train.csv را خوانده و در یک data frame ذخیره می کنیم. موارد گفته شده در کد زیر انجام شده است:



سپس ستون متن اخبار و برچسب مربوط به آن را جهت پردازش های بعدی، در دو لیست جداگانه ذخیره می کنیم.



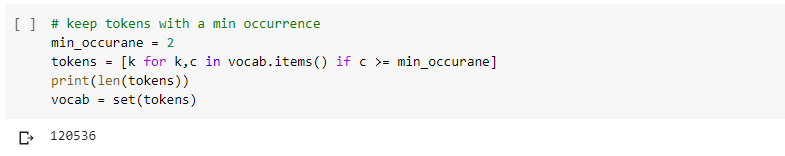
سپس عملیات پاکسازی و پیش پردازش بر روی متن می بایست انجام شود. برای پاکسازی متن ابتدا punctuationها حذف می شوند، سپس کلمات غیر الفبایی و در نهایت stop wordها حذف می گردند. برای stop wordها باید از کتابخانه nltk استفاده گردد و ارآنجا که این کتابخانه به صورت پیش فرض در محیط colab وجود ندارد، ابتدا باید آن را import کرده و stop wordها را دانلود کنیم.



حال باید vocab استفاده شده در متن اخبار را استخراج کنیم. برای اینکار ابتدا می بایست متن اخبار را طبق روال گفته شده پاکسازی کنیم و سپس کلمات موجود در آن را در یک شئ counter ذخیره کنیم. از شئ counter برای ساخت vocab و همچنین شمارش کلمات استفاده شده است.



مشاهده می شود که تعداد کل کلمات 221093 کلمه است. برای کاهش این تعداد، کلماتی که فقط یکبار در کل دیتاست دیده شده‌اند را حذف می کنیم:

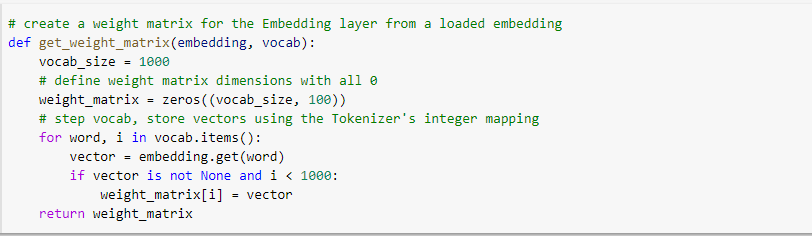


مشاهده می شود که تعداد کلمات، به 120536 مورد کاهش پیدا کرد. حال مجموعه vocab ساخته شده است. از این مجموعه برای ساخت ماتریس کلمات استفاده می شود که به عنوان ورودی به لایه embedding داده خواهد شد.

تابع load\_embedding برای خواندن فایل GloVe و ساخت یک دیکشنری از اطلاعات موجود در آن تعریف شده است که جلوتر از آن استفاده خواهیم کرد.



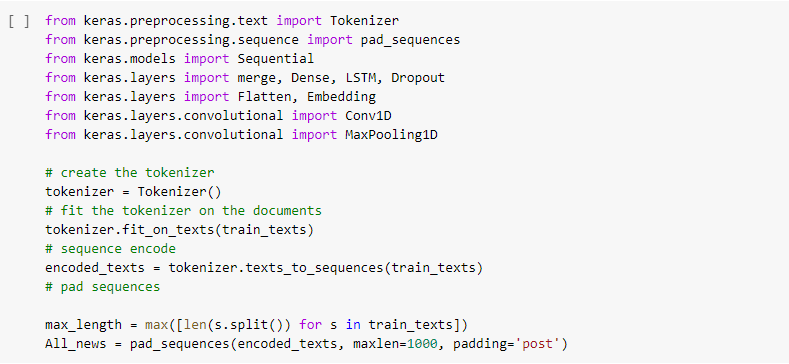
تابع get\_weight\_matrix برای ساخت ماتریس شامل بردار کلمات تعریف شده است. در این تابع ابتدا یک ماتریس شامل اعداد صفر ساخته می شود. از آنجا که در مقاله گفته شده که فقط از 1000 کلمه اول هر متن خبر برای پردازش استفاده می شود، تعداد سطرها 1000 در نظر گرفته شده. تعداد ستونها نیز 100 در نظر گرفته شده چون از مدل 100 بعدی Glove برای encode کردن کلمات استفاده شده است. جلوتر از این تابع استفاده خواهد شد.



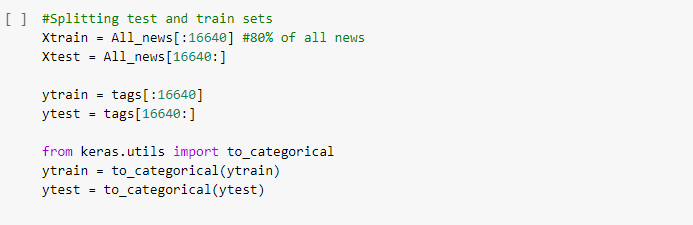
حال می خواهیم متن اخبار موجود در دیتاست را پاکسازی و پیش پردازش کرده و در لیست متنی train\_texts ذخیره کنیم:



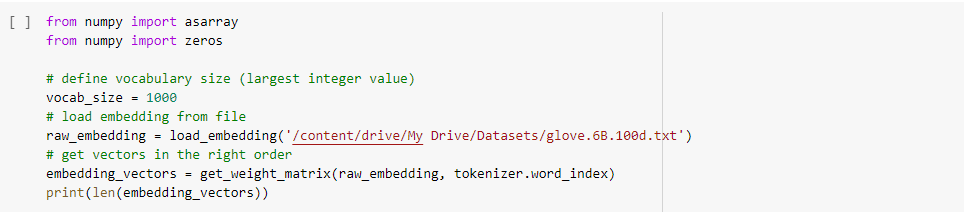
حال توسط ابزار Tokenizer از کتابخانه keras ، متن های موجود در train-texts را توکن می کنیم، encode می کنیم و از آنجا که طول اخبار با هم برابر نیستند و از طرف دیگر شبکه کانولوشن ورودی با طول های متفاوت را پشتیبانی نمی کند، باید طول تمام متن ها را توسط اضافه کردن 0 به انتهای متن، با هم برابر کنیم (به این عمل padding می گویند، و ما در اینجا صفرها را به انتها اضافه کردیم. –post padding). در اینجا از 1000 کلمه اول متن خبر استفاده می شود و در صورتی که طول متن کمتر از 1000 باشد به انتهای آن صفر اضافه می شود. متن های padding شده ، در لیستی به نام All\_news ذخیره شده اند.



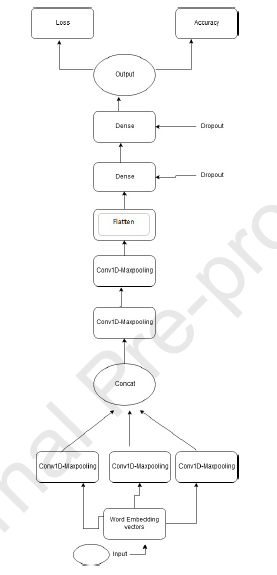
حال کل اخبار پیش پردازش شده در لیست All\_news و برچسب های مربوط به آنها در لیست tags قرار دارند. از این لیست ها 80% تعداد کل را برای آموزش و 20% را برای تست در نظر می گیریم.



حال فایل glove.6B.100.txt را که قبلا دانلود کرده و در google drive خود قرار داده ایم را توسط تابع load\_embedding که پیشتر توضیح داده شد خوانده و محتویات آن را در یک دیکشنری ذخیره می کنیم. حال می خواهیم بردار وزن های موبوط به کلمات موجود در اخبار را تولید کنیم. برای اینکار از تابع get\_weight\_matrix که پیشتر توضیح داده شده استفاده می کنیم و دیکشنری تولید شده توسط تابع load\_embedding را به عنوان ورودی به آن می دهیم.



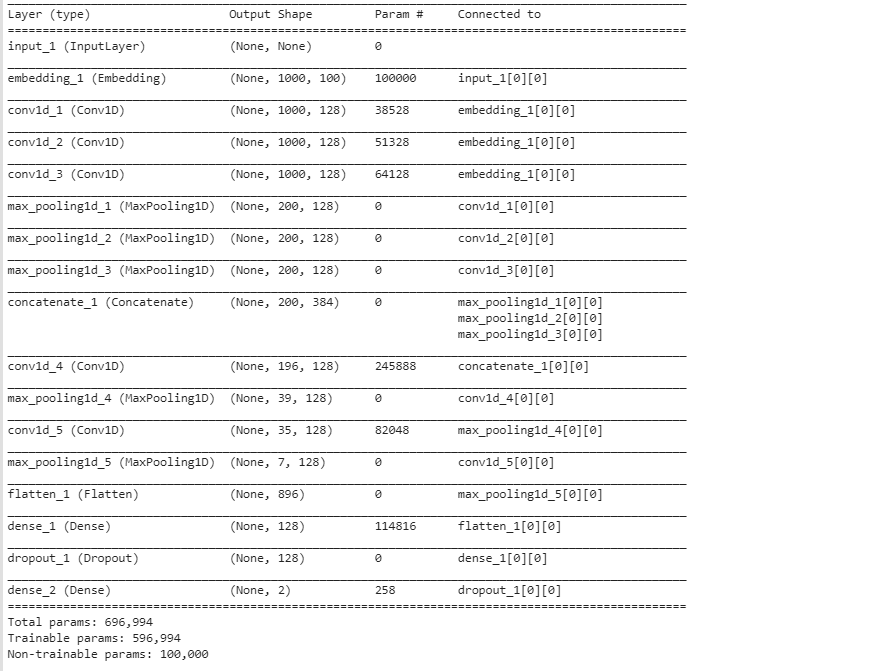
حال نوبت تعریف مدل و آموزش آن است. در ابتدا مانند هر مدل پردازش متن ، ابتدا باید لایه embedding قرار داده شود. خروجی این لایه به سه لایه کانولوشن به طور موازی داده می شود. خروجی این سه لایه کانولوشن با هم concatenate شده سپس دو لایه کانولوشن دیگر بر روی این خروجی پردازش انجام می دهند. بدیهی است که بعد از هر لایه کانولوشن یک لایه max-pooling نیز قرار دارد. سپس یک لایه Flatten و در نهایت دو لایه Dense قرار داده شده است. شمای مدل در شکل زیر دیده می شود:



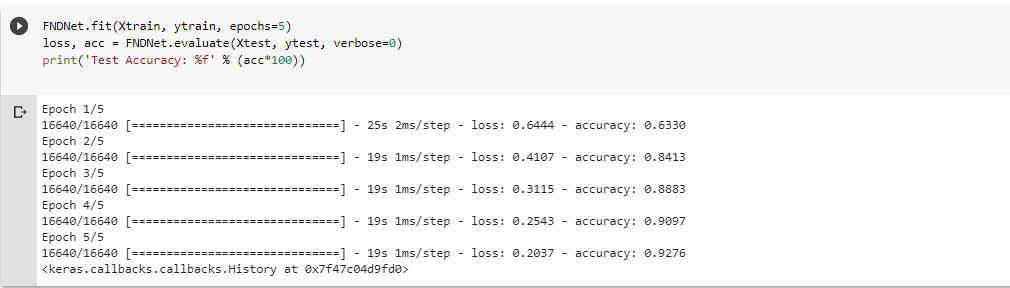
پیاده سازی مدل در اینجا دیده می شود:



خلاصه مدل تعریف شده در شکل زیر قابل مشاهده است:



داده های آموزش و تست را به مدل داده و پس از اجرای پنج epoch نتایج زیر حاصل می شوند:



نتیجه گیری:

در این گزارش توضیحات مربوط به پیاده سازی مدلی جهت تشخیص اخبار جعلی ارائه شده است. در این مدل از شبکه کانولوشن جهت کلاس بندی اخبار موجود در دیتاست استفاده شده است.