رامتین احسانی – سینا اسکندری

فاز نهایی پروژه درس مبانی پردازش زبان و گفتار

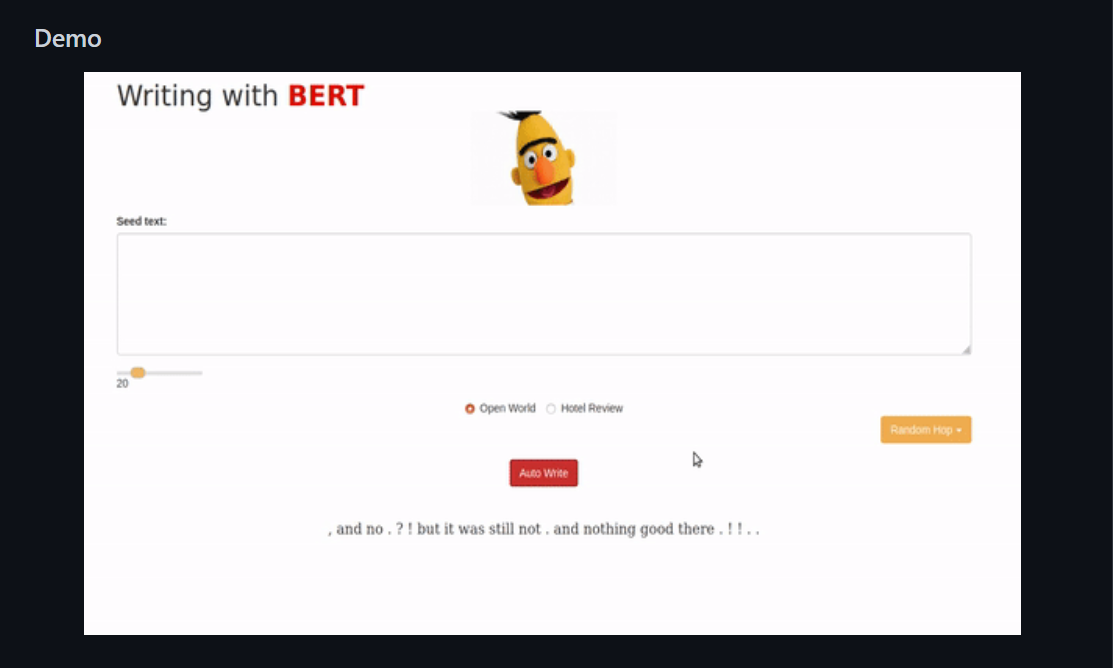
**بخش اول: تولید جملات**

برای این بخش، از آن جایی که دیتای ما به زبان انگلیسی میباشد از مدل pretrained شده BERT استفاده کردیم.

به این صورت که مدل را در کد لود کردیم و بر روی دیتای از پیش تهیه شده خودمان train کردیم. پس از به پایان رسیدن آموزش و تست این مدل، مدل نهایی را در drive ذخیره کردیم.

برای بخش تولید جملات از پروژه زیر کمک گرفتیم:

<https://github.com/prakhar21/Writing-with-BERT>



تمامی فایل های تولید شده توسط کد خودمان را به پروژه بالا میدهیم و با استفاده از مدل trained شده ما، این کد شروع به تولید جملات میکند.

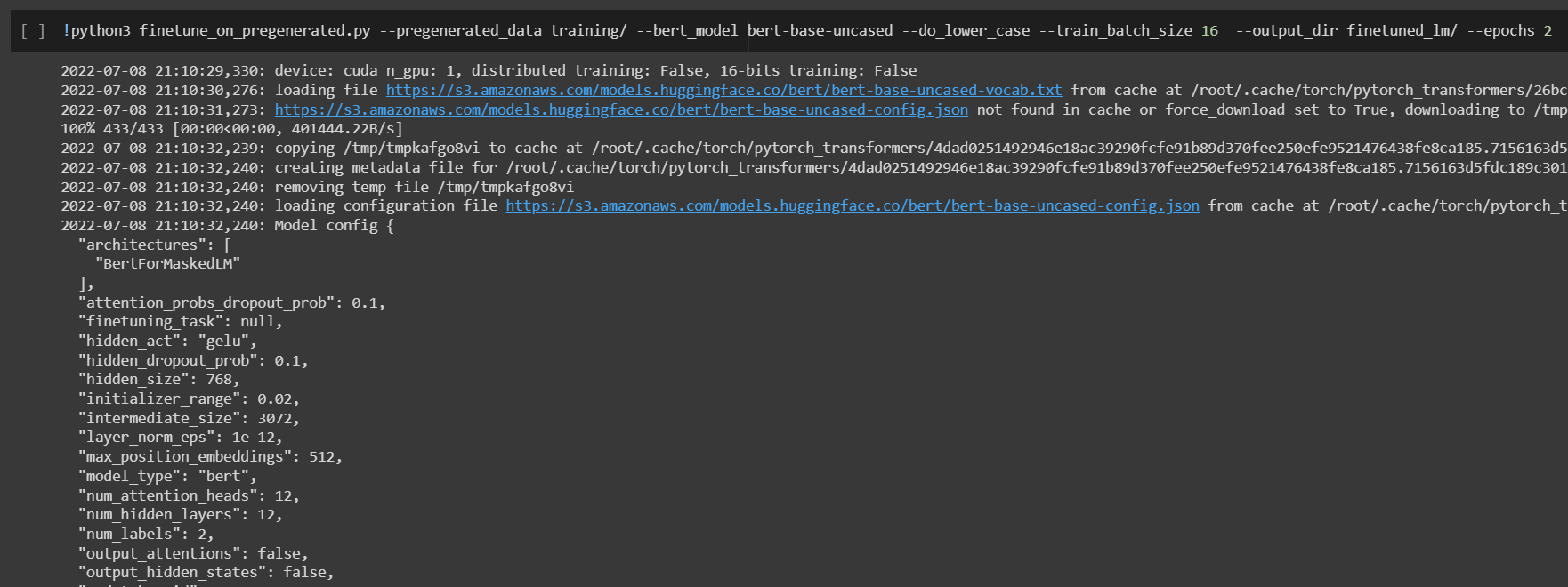
لینک مدل و سایر موارد تولید شده در این مرحله:

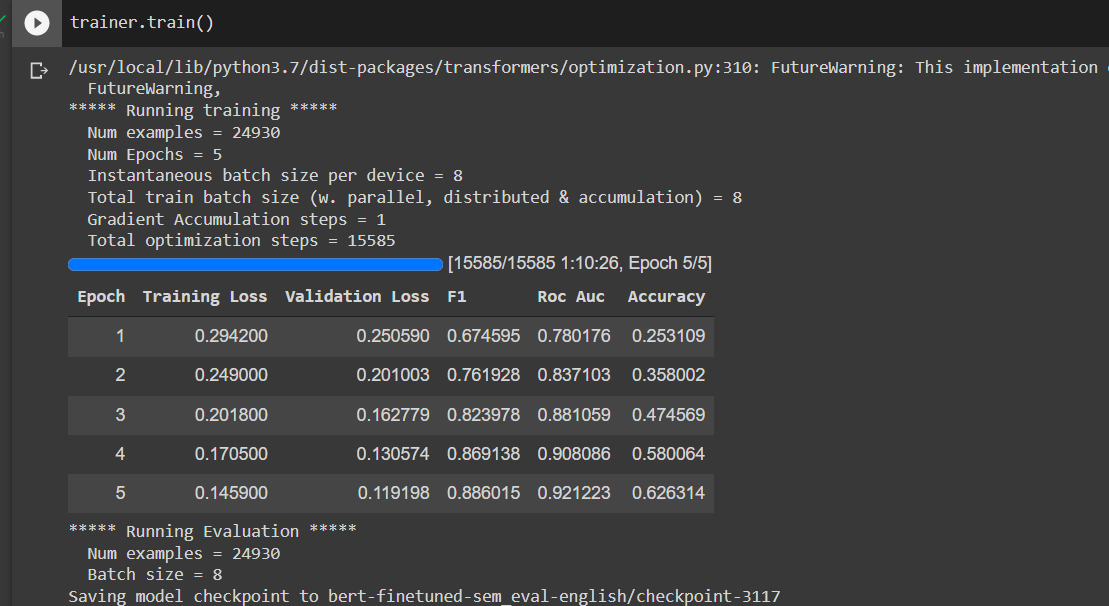
<https://drive.google.com/drive/folders/1ZEW6xOXzSCHRDrdU2RG4F5KzFy9CGwCM?usp=sharing>

لینک گوگل کولب:

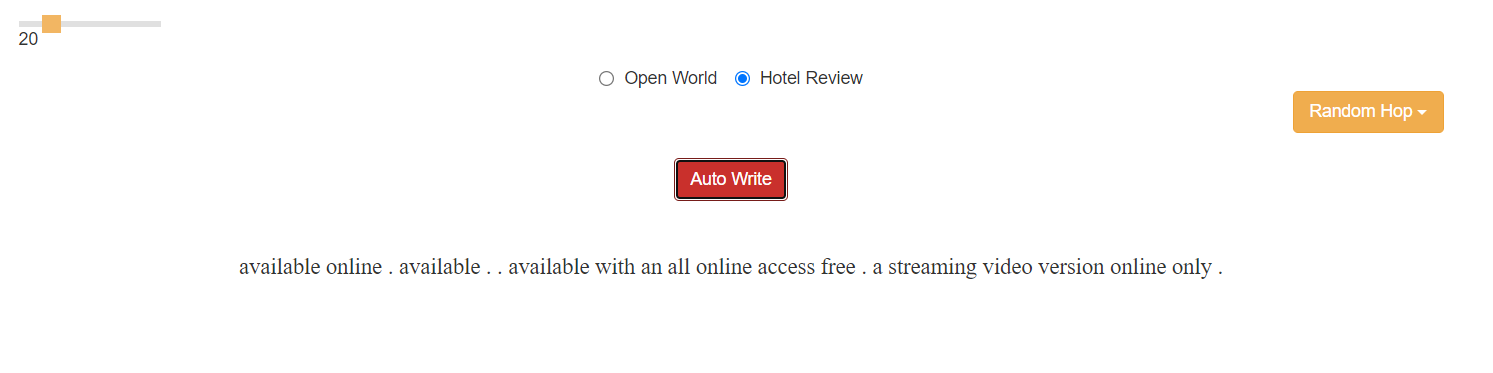
<https://colab.research.google.com/drive/1vwxrLCMafSY0dWWNMeZ3EEUEjNAJVv6s?usp=sharing>

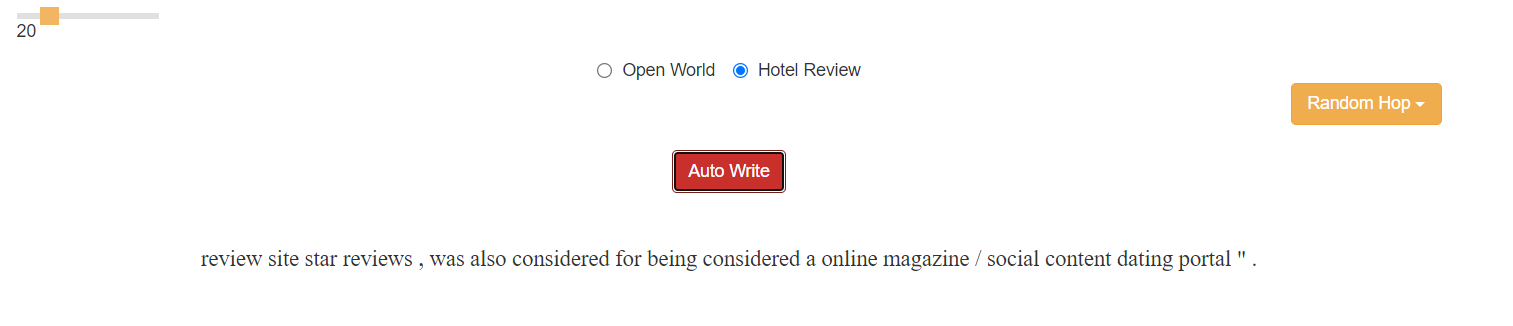
آموزش و evaluation مدل BERT:





نتایج حاصل شده:





دقت شود که اگر با گزینه ‌های جایگزین مانند GPT2 و LM مبتنی بر LSTM/GRU مقایسه شود، BERT برای کارهایی مانند NLG مناسب نخواهد بود، فقط به این دلیل که در وهله اول به سبک معمولی رگرسیون خودکار آموزش داده نشده است. BERT به عنوان مدل زبان ماسک شده (MLM) در سبک دو جهته آموزش داده شده است. در MLM به‌جای پیش‌بینی هر نشانه بعدی، درصدی از نشانه‌های ورودی به‌صورت تصادفی پوشانده می‌شوند و تنها آن نشانه ‌ها بر اساس کلمات باقی ‌مانده در سمت چپ و راست آن پیش‌بینی می‌شوند، که به آن زمینه دو سویه غنی می‌دهد.

برای اجرای کد، فایل مدل ها (nlp) را از درایو دانلود کرده و در دایرکتوری قسمت BERT قرار دهید.

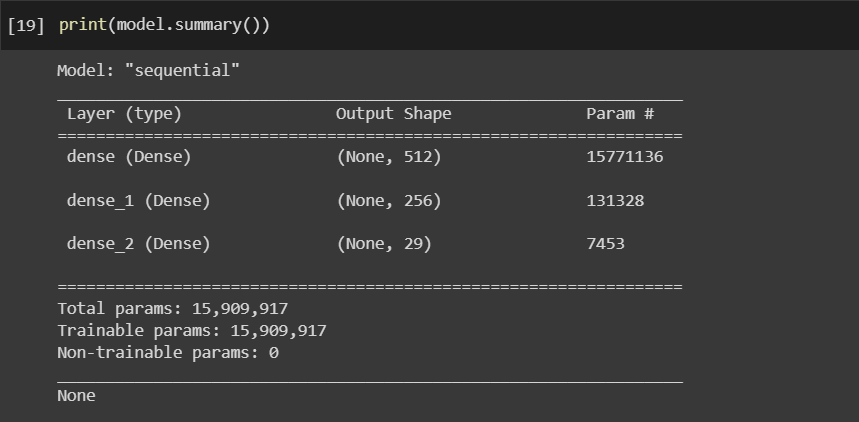
سپس app.py را اجرا کرده و با استفاده از فایل index.html به سرور request بزنید و جمله تولید کنید.

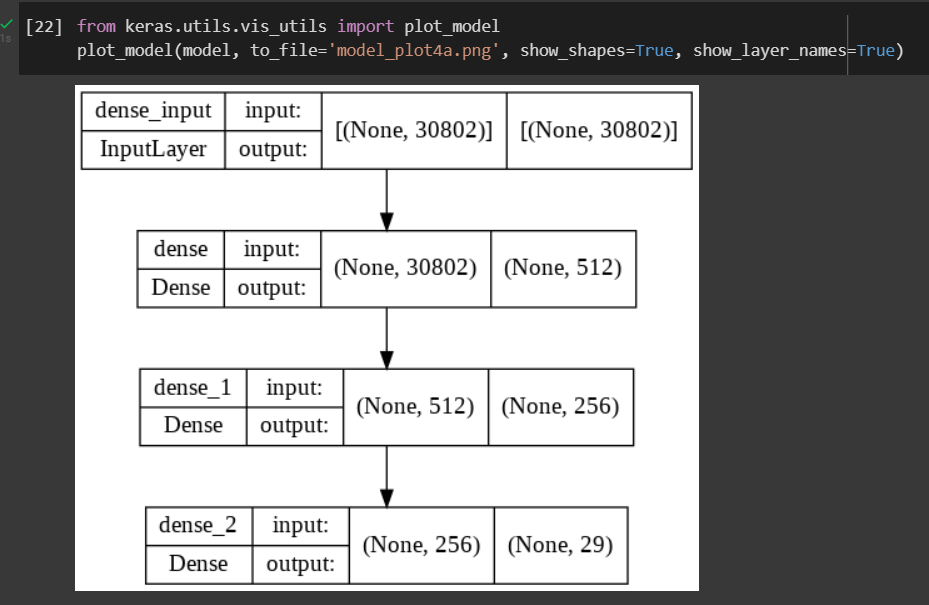
**بخش دوم:**

لینک گوگل کولب:

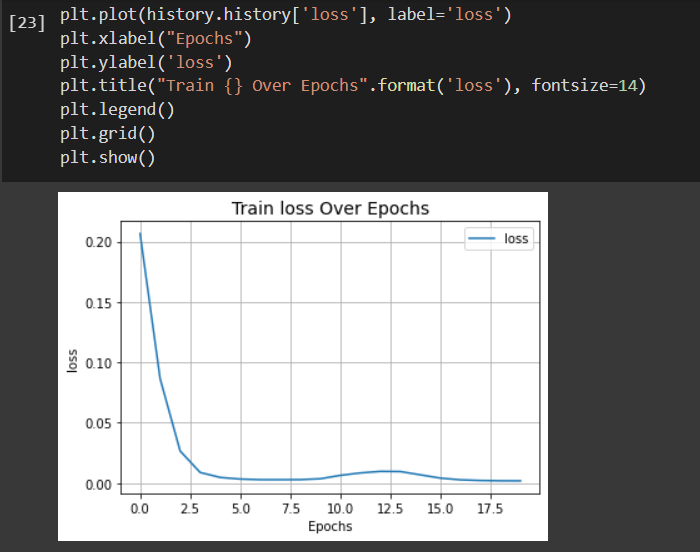
<https://colab.research.google.com/drive/1SWUvSaA2pqashAqa7xxaIvcuSKBRWxOj?usp=sharing>

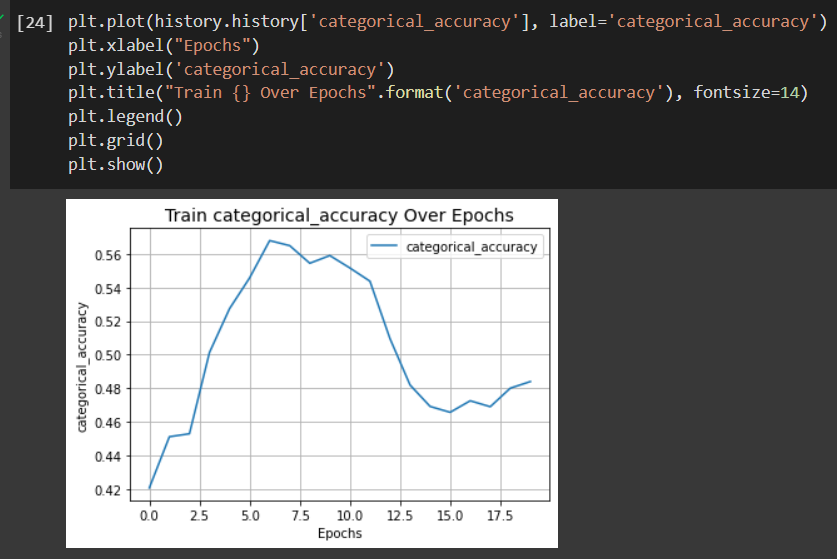
برای این بخش از یک مدل ساده dense استفاده کردیم که در شکل زیر قابل مشاهده میباشد:

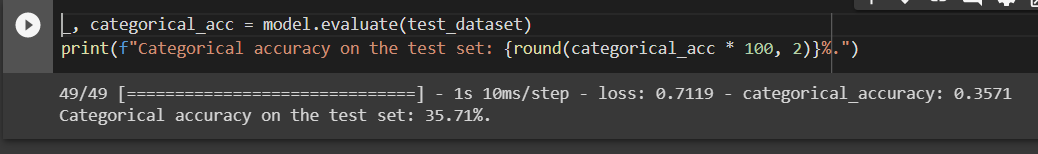




از آن جایی که از دیتای crawl شده و تهیه شده توسط خودمان در فاز های قبلی استفاده کردیم، و همینطور به دلیل اینکه کلاس های کتگوری برای این پروژه زیاد هستند و دیتای ما محدود، به طبع انتظار زیادی از این مدل در Evaluation نداریم. با این حال، این مدل حدود 35 درصد accuary به ما میدهد:





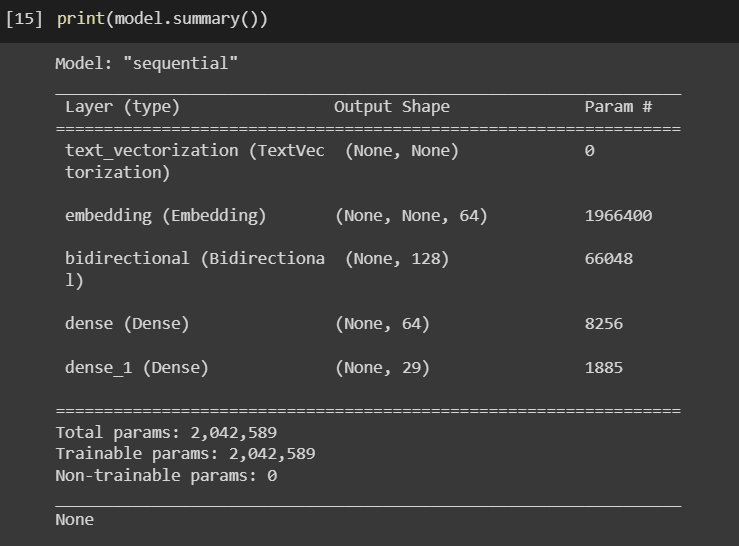


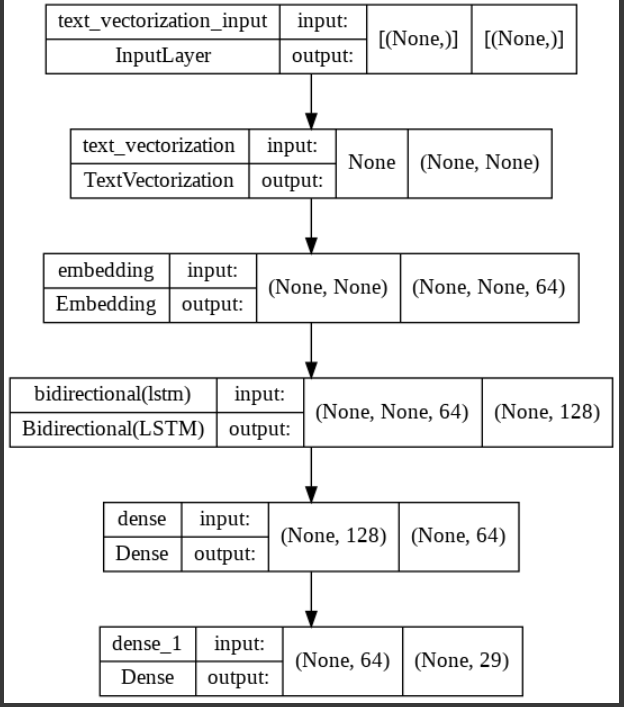
کد های این بخش در فایل Part2 قرار دارند.

**بخش سوم:**

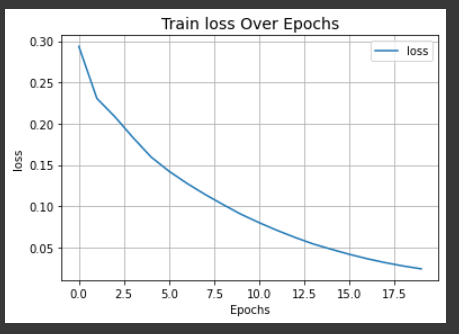
برای بهبود مدل طراحی شده، دو راه پیاده سازی را پیش گرفتیم. استفاده از LSTM، و استفاده از CNN.

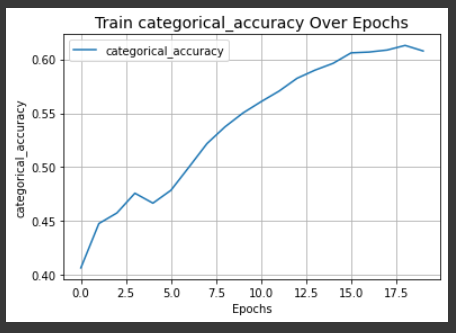
روش LSTM:

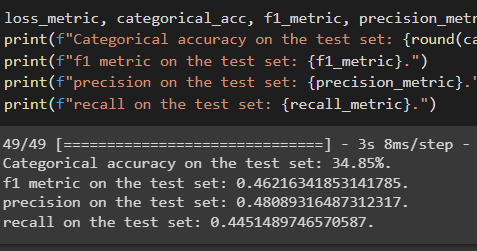




نتایج بدست آمده از مدل طراحی شده با lstm:







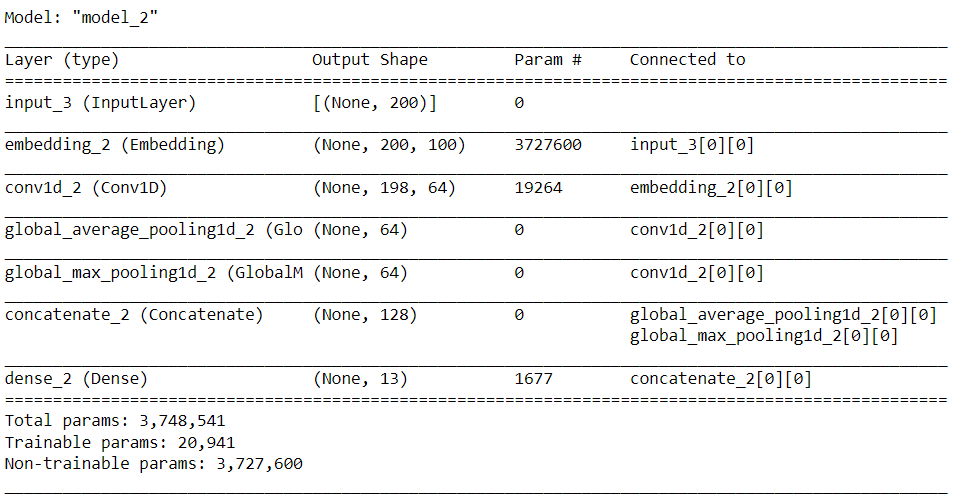
همانطور که مشخص است همچنان پیشرفت خاصی بر روی نتایج مدل مشاهده نمیکنیم.

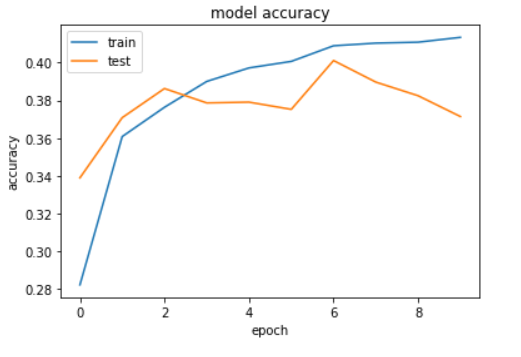
فایل گوگل کولب برای روش lstm:

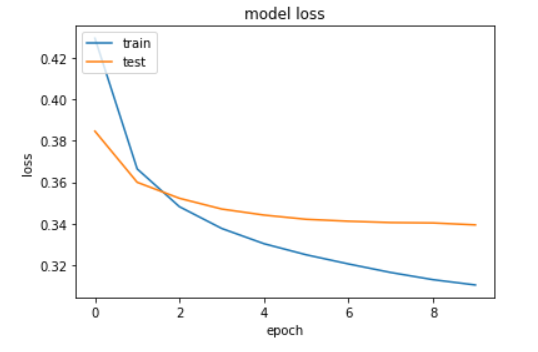
<https://colab.research.google.com/drive/1Uq-vcH1fv-ePdpEeW-T-xkdhTIN6OiQR?usp=sharing>

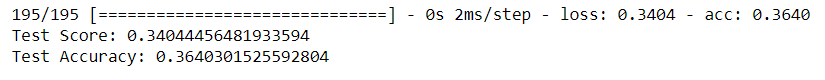
روش CNN:

در این روش ابتدا ابتدا لایه Embedding را به یک لایه Conv1D وارد می کنیم و سپس از Average Pooling و Max Pooling همزمان استفاده می کنیم و این 2 را در انتها به هم متصل می کنیم. نتیجه آخرین لایه concatenate (None, 13) است که به لایه خروجی داده می شود.









باز هم پیشرفت خوبی در نتایج حاصل نمیشود.

دلیل اصلی این موضوع زیاد بودن کتگوری های ژانر های فیلم ها و محدود بودن دیتا در این زمینه میباشد.

چیزی حدود حداقل 13 تا ژانر متفاوت در دیتا وجود دارد و هر فیلم حداقل 2 یا 3 ژانر دارد و طبیعی است که مدل ما به دیتای بیشتری در این زمینه احتیاج داشته باشد.

کد های این بخش در فایل Part3 قرار دارند.