




فایل های ضمیمه :

 distilBERT.ipynb

 multilingualBERT.ipynb

 parsBERT.ipynb

پیش پردازش : فایل Train.txt را باز می کنیم. تعداد تگ ها را شمارش میکنم و جمله جمله کلمات را جدا می کنم. از روی این جمله ها یک دیتافریم دوستونه می سازم که ستون اول حاوی جمله ی توکنایز شده ست و ستون دوم برچسب یا تگ نظیر این کلمات در این جمله. ۲۳ تگ شمارش شده. نظیر هر کدام از این ها یک عدد قرار می دهیم. این کار با مدل preprocessing.LabelEncoder() از sklearn انجام شده و با همین مدل قابل بازگشت هم هست. تمام این عملیات روی داده های تست هم انجام شده و ستون اول آن برای pos tagging بعد از تنظیم مدل وارد شبکه می شود. دیتافریمی که از روی داده های آموزش ساخته بودم را به دوبخش آموزش و اعتبارسنجی تقسیم می کنم.

برای اینکه دهانه ی شبکه اندازه ی ثابتی دارد و بعضی جمله ها ممکن است کمتر یا بیشتر از آن کلمه داشته باشند توکنایز کردن جمله ها را با یک تابع انجام میدهم که اندازه ی خروجی را که در واقع اندازه ی دهانه ی شبکه هست به تعداد کلمات و برچسب هایشان مپ کند. tokenize_and_align_labels در این تابع شبکه ای که قرار است fine-tune شود به عنوان توکنایزر استفاده می شود.

در این تمرین از سه شبکه استفاده کردم. مراحل زیر را برای هر کدام از اینها در فایل ها ی مجزا انجام دادم.

۱. با AutoModelForTokenClassification شبکه را لود کردم. این کلاس شبکه را طوری تنظیم می کند که لایه خروجی آن به ازای هر توکن ورودی یک بردار خروجی به اندازه ی تعداد کلاس ها (تعداد تگها) بدهد.
۲. پارامترهای فاین تیونینگ را تنظیم می کنم. برای learning_rate اعداد بین ۰,۰۰۰۱ تا ۰,۰۰۰۰۰۱ را امتحان کردم. ابتدا شبکه ها را با num_train_epochs=3 ترین می کردم ولی با زیاد تر کردن این تعداد نتایج بهتر و بهتر میشد. و حدود ۱۰ تا ۱۵ بهترین نتیجه را می داد و بعد از آن روی validation loss بهبود چندانی پیدا نمی شد. چون لایه ی فیدفوروارد آخر این شبکه ها قرار است کلاس بندی توکن ها را انجام دهد قسمت اصلی فاین تیونینگ روی این لایه کار میکند و برای همین است که بر خلاف تمرین ۵ تعداد ایپاک بیشتری نیاز است تا نتیجه خوب بگیرم. در تمرین قبلی تمرکز اصلی روی یک کلاس بندی خطی یا یک فیدفوروارد با یک خروجی بود (وزن ها ی کمتر).
۳. با شبکه ی فاین تیون شده تگ گذاری را انجام می دهیم. حاصل کار برای هر جمله ی داده های تست (پیش پردازش هایی که بالا گفته شد روی Test.txt انجام گرفته) به ازای هر توکن ورودی یک بردار با اندازه ی ۲۳ هست. روی هر کدام از اینها argmax گرفته میشود و برچسب های توکن ها تعیین می شود.
۴. مخاسبه ی دقت روی داده های تست

شبکه هایی که برای تمرین استفاده کردم :

- **distilbert-base-uncased** : یک مدل زبانی که با داده‌های ویکیپدیا و کتاب و مجلات انگلیسی آموزش دیده. و یک مدل فاین تیون شده از آن هم موجود هست که برای NER فاین تیون شده و نتایج خوبی داده. (اینجا از مدل زبانی آن استفاده شده).

- **HooshvareLab/bert-fa-zwnj-base** : همان مدل برت فارسی یا **parsbert** هست.

- **bert-base-multilingual-cased** : یک مدل زبانی که با داده‌های ۱۰۴ زبان مختلف آموزش دیده.

چون قرار است با داده‌های تمرین ۳ مقایسه انجام دهم و در این تمرین فقط **accuracy** محاسبه شده بود اینجا هم به همین معیار اکتفا می‌کنم.

Viterbi – HW03	82
distilBERT	76.5
multilingualBERT	88.7
parsBERT	81.8

دقت برت فارسی با الگوریتم ویتربی برابر است. مدلی که فقط با داده‌های انگلیسی آموزش دیده بود عملکرد ضعیف تری دارد برای زبان فارسی ولی مدل چندزبانه ای که با ۱۰۴ زبان مختلف آموزش دیده بود دقت بهتری حتی نسبت به برت فارسی و ویتربی دارد. نسبت به ویتربی قطعا به این دلیل که مدل پیچیدگی‌های بیشتری از زبان را می‌تواند یاد بگیرد. و دلیل بالاتر بودن دقت نسبت به برت فارسی را هم میتوان به این تفسیر کرد که چون زبان‌های بیشتری دیده و یاد گرفته این اطلاعات برای برچسب زنی جملات فارسی می‌تواند مفید واقع شود. علاوه بر این حجم داده‌هایی که با آن‌ها آموزش دیده نسبت به برت فارسی بیشتر هست و مدل زبانی قوی تری برای تگ زنی جملات دارد.