دانشگاه صنعتي اميركبير
(پلي تكنيك تهران)

دانشكده مهندسي كامپيوتر

به نام خدا

تمرین اول فهم زبان

تشخیص کسره اضافه

زینب خالوندی

۹۹۱۳۱۰۰۷

بخش یک: داده‌ها

برای ایجاد نمونه‌های آموزشی داده‌ها بصورت جمله‌ایی و براساس نقطه از هم جدا شده‌اند. به ازای کلمات هر جمله برچسب متناظر آن‌ها تعیین شده است. طول جملات بسیار متفاوت است. برای یکسان کردن طول نمونه-های آموزشی همه‌ی جملات به طول میانگین همه‌ی جملات برده شده‌اند. برای بازنمایی کلمات هر کلمه به شماره‌ی ترتیب آن در واژه‌نامه داده‌های آموزشی نگاشت شده است.

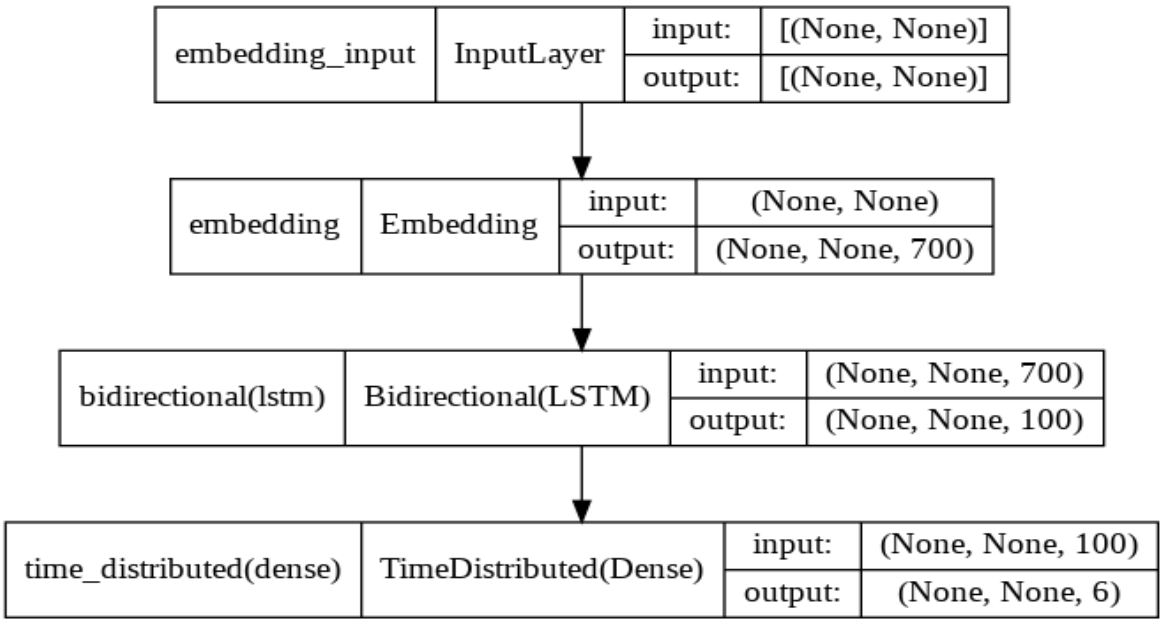
بخش دو: معیار ارزیابی

همانطورکه در صورت تمرین ذکر شده است، استفاده از معیار دقت، راه خوبی برای ارزیابی عملکرد مدل نیست، به این دلیل که برچسب داده‌ها در این نوع داده متعادل نیستند و بیشتر کلمات برچسب **'O'** را دارند. بنابراین در آموزش مدل هم این نمونه‌ها بسیار بیشتر هستند و مدل به این سمت آموزش می‌بیند که بیشتر برچسب‌ها از این نوع هستند و ممکن است برچسب‌های زیادی به اشتباه از این نوع پیش‌بینی شوند، حال چون در کل تعداد این برچسب بسیار بیشتر از سایر برچسب‌ها است و اشتباه پیشبینی کردن بقیه برچسب‌ها درصد کمی از پیشبینی-ها را شامل می‌شود و باعث کاهش دقت نمی‌شود، در نتیجه با دقت بالا ما نمونه‌هایی از کلاس‌های مختلف را داریم که به درستی پیش‌بینی نشده‌اند.

اما اگر **precision , recall** را برای هر کلاس جداگانه حساب کنیم و در نهایت میانگین وزن‌دار این مقادیر را برای مدل بطور کلی محاسبه کنیم، دید بهتری از کارایی مدل به ما خواهد داد.

بخش سه: BiLSTM

شکل ۱ معماری شبکه‌ی آموزشی را نمایش می‌دهد. در این شبکه ابتدا یک لایه‌ی Embedding را قرار داده شده‌است تا یک بازنمایی برداری برای کلمات بدست آید. با توجه به آزمایشات انجام شده طول این بردار ۷۰۰ بعد در نظر گرفته شده‌است. پس از این لایه، یک لایه‌ی BiLSTM قرار داده شده است، ورودی این لایه یک دنباله از بردارهای کلمات که به واحدهای LSTM وارد می‌شود. خروجی این لایه به ازای هر کلمه یک بردار ۱۰۰ بعدی است. در ادامه برای تشخیص حالت کسره اضافه‌ی هر کلمه یک لایه‌ی تمام متصل بر روی خروجی هر کلمه اعمال شده است تا دسته‌بندی را انجام دهد.

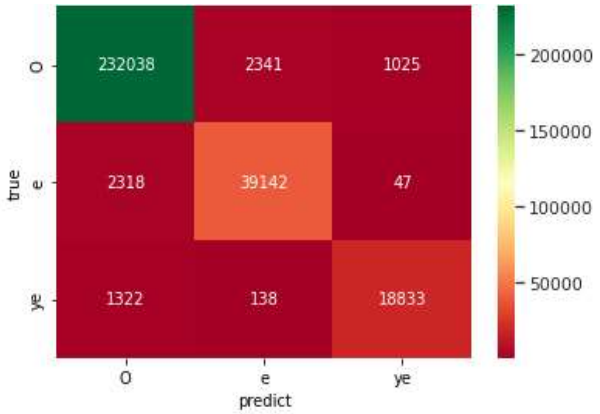


شکل ۱

در ادامه مجموعه‌ایی از آزمایشات انجام شده برای انتخاب تنظمیات بهتر شبکه آمده است.

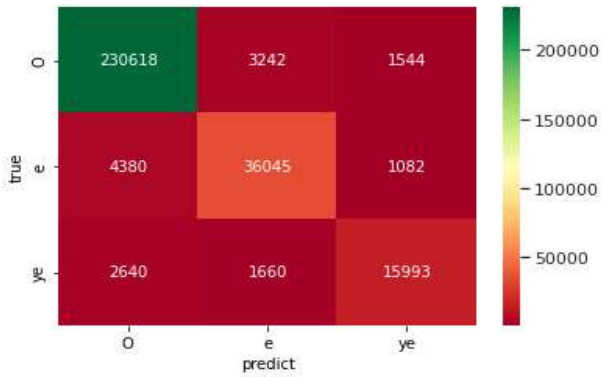
- Embedding size = 250, lstm units= 100, without stemming

```
accuracy:  0.9758044979206202
precision micro:  0.9758044979206202
precision macro:  0.9570460346367854
recall micro:  0.9758044979206202
recall macro:  0.9522589484396198
```



- Embedding size = 250, lstm units= 100, stem words

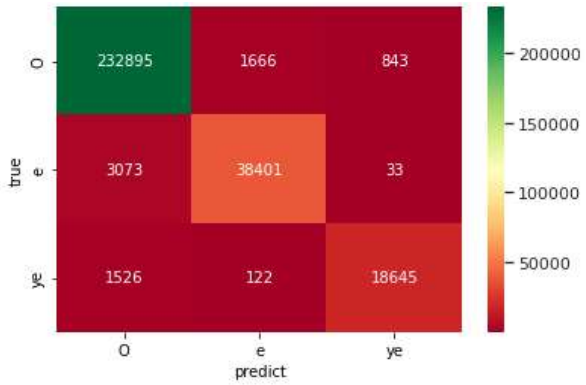
```
accuracy:  0.9510504569252096
precision micro:  0.9510504569252096
precision macro:  0.9032349386692508
recall micro:  0.9510504569252096
recall macro:  0.8787270018141639
```



با توجه به نتایج بدست آمده ریشه‌یابی کلمات باعث کاهش دقت می‌شود.

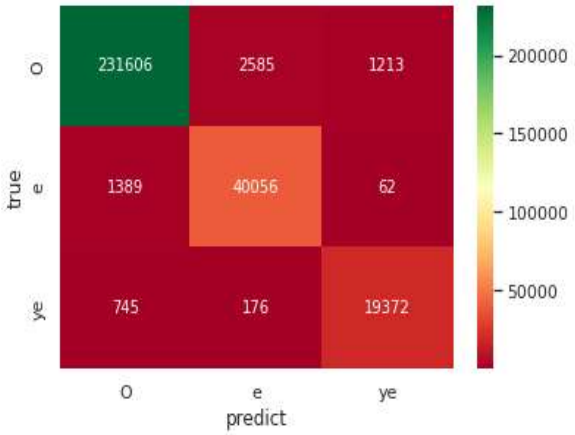
- Embedding size =500, lstm units= 100

```
accuracy:  0.9755622400775225
precision micro:  0.9755622400775225
precision macro:  0.9637569214096676
recall micro:  0.9755622400775225
recall macro:  0.9444335688075683
```



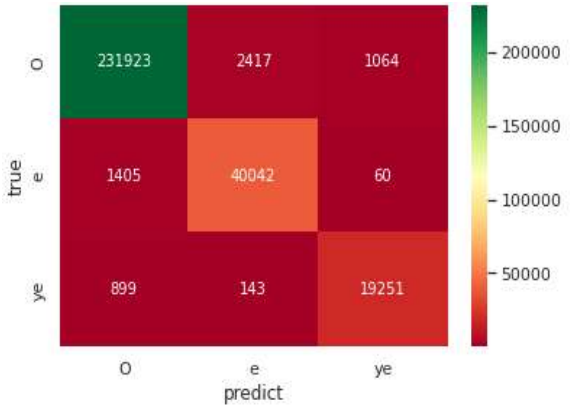
- Embedding size =500, lstm units= 50

```
accuracy:  0.9792398487234357
precision micro:  0.9792398487234357
precision macro:  0.9548780506247319
recall micro:  0.9792398487234357
recall macro:  0.9678409891659944
```



- Embedding size =700, lstm units= 50

```
accuracy:  0.9798522227157104
precision micro:  0.9798522227157104
precision macro:  0.9583022229331969
recall micro:  0.9798522227157104
recall macro:  0.9661898829675951
```

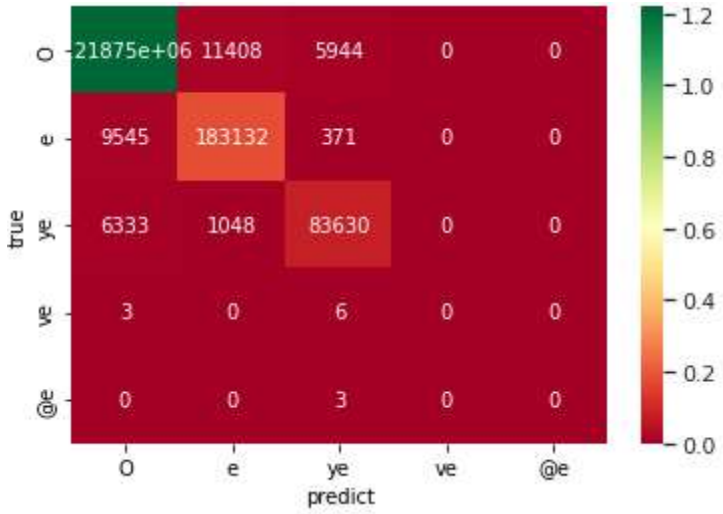


با افزایش ابعاد بازنمایی کلمات دقت افزایش یافته است. اما طبق نتایج مشاهده شده اندازه‌های بزرگتر مانند ۹۰۰ تغییر قابل توجهی در دقت نداده است، بنابراین در نهایت همان ۷۰۰ بعد در نظر گرفته شده است.

شبکه‌ی نهایی آموزش داده شده به صورت زیر است.

```
predict_model = Sequential()
predict_model.add(Embedding(len(word_list)+2,700 ))
predict_model.add(Bidirectional(LSTM(units = 50,input_shape= (30,700), return_sequences=True
)))
predict_model.add(TimeDistributed(Dense(6,activation='softmax')))
```

برای آزمودن شبکه داده‌های آزمون به تفکیک کلمات هر جمله برای پیش‌بینی شبکه داده شده است، در نهایت نتایج بدست آمده به شکل زیر است.




```
accuracy score: 0.9771

precision on 'O' :  0.9871371
precision on 'e' :  0.9362
precision on 'ye' :  0.92962
precision on 've' :  0
precision on '@e' :  0
recall on 'O' :  0.9859
recall on 'e' :  0.9486
recall on 'ye' :  0.919
recall on 've' :  0
recall on '@e' :  0
mean recall: 0.973912
mean precision:  0.973974
F score: 0.97394
```

نتایج بدست آمده برای کلاس‌های @eve به این دلیل است که این نوع کلاس در داده‌های آموزشی بندرت دیده شده است و شبکه برای این نوع برچسب به خوبی آموزش ندیده است، بنابراین در تشخیص آن‌ها دچار مشکل شده است.

بخش چهار: برت

در این بخش برای tokenize کردن جملات و یکسان کردن طول آن‌ها از مدل از پیش آموزش‌داده شده‌ی پارس‌برت استفاده شده است. این tokenizer مطابق با داده‌هایی که از پیش دیده است، کلمات یک جمله

را جدا می‌کند و داده‌ها را به اعداد مشخص خود نگاشت می‌کند و داده‌ها را مطابق با تنظیمات مشخص شده هم طول می‌کند.

پس از آماده‌سازی داده‌ها، از کلاس آماده‌ی AutoModelForTokenClassification برای دسته‌بندی برچسب هر کلمه در جمله استفاده شده است. با داده‌های آموزش این کلاس `fine tune` شده است.

نتایج بدست آمده به شکل زیر است.

Loss	Precision	Recall	F1	Accuracy
0.033470	0.957787	0.963055	0.960414	0.991083