باسمه تعالی

گزارش تمرین شماره ۶ درس پردازش زبان طبیعی

استاد درس: جناب آقای دکتر باباعلی

نام دانشجو : ایمان کیانیان

شماره دانشجویی: ۶۱۰۳۰۰۲۰۳

۱- مقدمه

این تمرین، ادامه ی تمرین گذشته است. دیتاستی شامل ۲۵۹۷۹۴ کلمه آموزش و ۲۵۹۷۹۴ کلمه تست به ما داده شده است. تمام این دو دیتاست شامل tag هستند. میخواهیم با استفاده از کلمات و tag متناظر در مجموعه آموزشی شبکه های از پیش آموزش داده شده BERT را fine tune کنیم و از این شبکه ها برای عمل POS Tagging استفاده کنیم. سپس برای کلمات موجود در مجموعه تست، tag گذاری انجام دهیم. سعی بر این داریم تا tag هایی که میگذاریم به تگ اصلی در داده آموزشی شبیه باشد. انتظار داریم نتیجه استفاده از BERT برای تگ زنی از نتیجه استفاده از HMM بهتر باشد. در ادامه نتایج را خواهیم دید.

۲- پیش پردازش داده ها

در ابتدا داده های آموزشی را لود کردیم. یک صدم از این دادگان را برای validation هنگام فرآیند آموزش از train جدا کردیم. سپس داده های تست را لود کردیم. از این داده های آموزشی و تست ۳ بخش مهم ساختیم. لیست های دو بعدی که لیست اول شامل جملات و لیست های داخلی شامل کلمات هر جمله هستند که هر کلمه یک درایه از لیست داخلی هستند. بخش مهم اول Xtrain یا Xtest است که شامل زوج یا tuple ای است که درایه اول کلمه و درایه دوم تگ مربوط به آن کلمه است. بخش دوم فقط کلمات را به عنوان درایه داریم (train_untagged و test_untagged) و بخش سوم فقط تگ ها را داریم (train_tags و test_tags).

```
Number of Sentences in Train courpes = 8721
Number of Sentences in Test courpes = 9279
```

تعداد جملات در پیکره آموزشی ۸۷۲۱ و تعداد جملات در پیکره تست ۹۲۷۹ است.

تگ هایی که در پیکره متنی داریم به صورت زیر است:

```
['V', 'PRO', 'DET', 'MS', 'N', 'OH', 'OHH', 'PS', 'ADJ', 'MORP', 'IF', 'P',
'INT', 'DEFAULT', 'AR', 'NP', 'MQUA', 'QUA', 'DELM', 'ADV', 'CON', 'PP',
'SPEC']
```

سپس به سراغ ساختن مدل های BERT از پیش آموزش داده شده کردیم و آنها را برای انجام عمل POS Tagging، fine tune کردیم که در قسمت بعدی به جزئیات آزمایش های انجام شده میپردازیم.

۳- استفاده از شبکه های عصبی مبتنی بر BERT

در این قسمت BERT های متفاوت را با تغییر دادن پارامتر ها و مدل های اولیه آزمایش میکنیم و نتیجه را در هر بخش ذکر میکنیم.

۳-۱. استفاده از شبکه عصبی ParsBERT:

در این قسمت ابتدا یک tokenizer با استفاده از مدل از پیش آموزش دیده ParsBERT میسازیم. مثلا برای یک در مجموعه آموزشی توکن شده آن جمله به صورت زیر است:

```
['[CLS]', 'در', 'سي', 'ستم##', 'ها##', 'ي##', 'حرارت', 'و', 'ي##', 'برودت', 'ي##', 'اغلب', 'دور',
'[UNK]', 'خواستهها', 'ي##', 'ما', 'را', 'توم', 'ين##', 'مي', 'نما##', 'يد##', '.', '[SEP]']
```

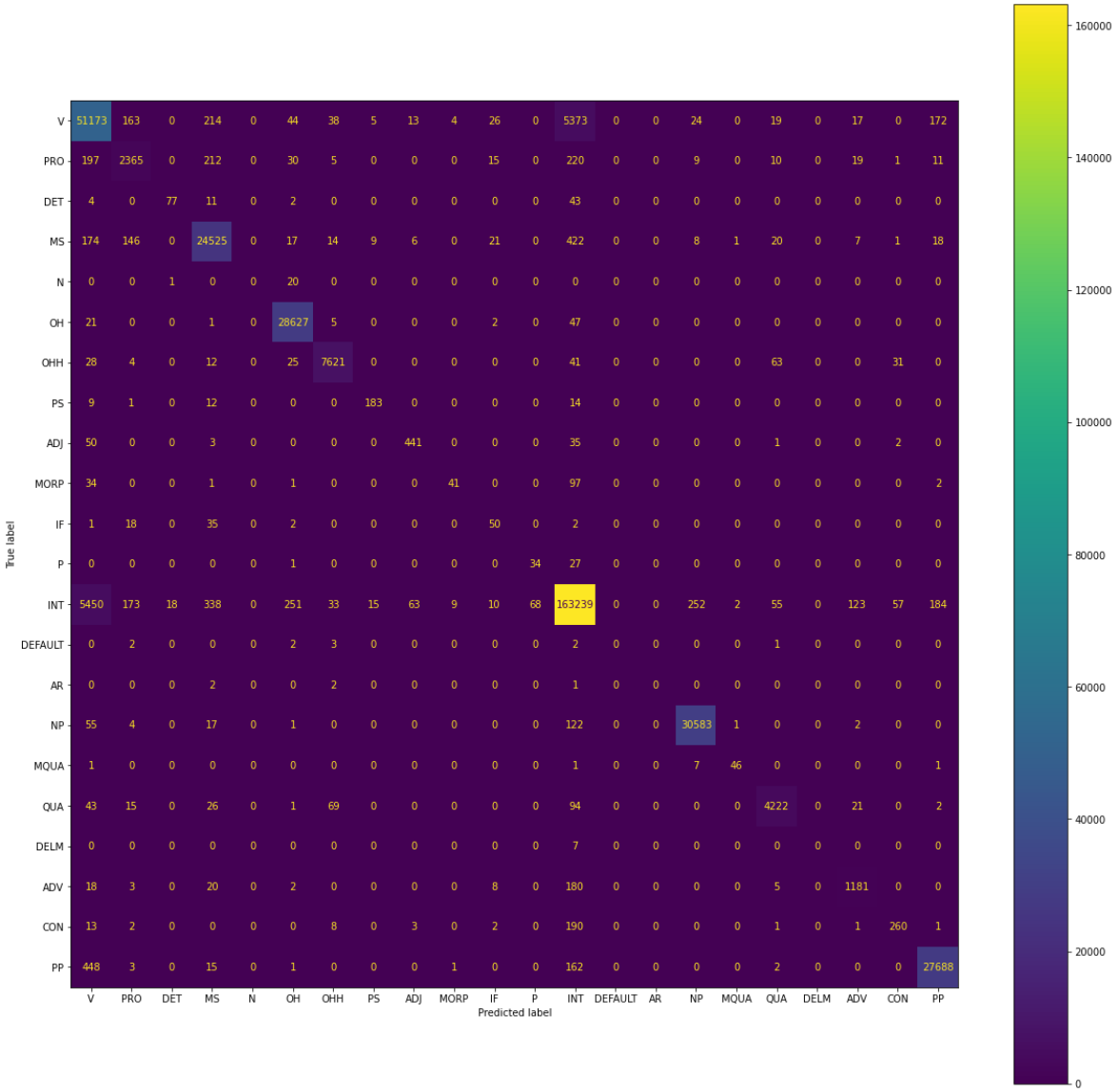
مدل BERT را با وزن های ParsBert با پارامتر های زیر ساختیم:

```
args = TrainingArguments(
    model_name,
    evaluation_strategy="epoch",
    save_strategy="epoch",
    learning_rate=2e-5,
    num_train_epochs=4,
    weight_decay=0.01,
)
```

در حین fine tuning دقت برای داده های validation به صورت زیر است:

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
1	0.197700	0.149055	0.889225	0.894641	0.891925	0.952055
2	0.120000	0.100768	0.903681	0.897077	0.900367	0.963014
3	0.078500	0.091332	0.914458	0.924482	0.919443	0.966849
4	0.054700	0.095936	0.918527	0.926918	0.922704	0.970411

بعد از انجام آموزش داده های تست را به مدل دادیم و خروجی ها را دریافت کردیم. دقت برابر با **0.9524042674530204** است. همچنین ماتریس آشفتگی برای داده های تست به صورت زیر است:



این ماتریس نشان میدهد که در بسیاری از مواقع INT به جای V و بسیاری از مواقع V به جای INT پیش بینی میشود و این کار را مشکل میکند و دقت را کاهش میدهد. همچنین معیار های دیگر ارزیابی برای مدل بر روی داده های تست به صورت زیر است. مشاهده میشود چون تعداد تکرار تگ های V و INT خیلی بالاست دقت خیلی پایینی نگرفته ایم :

	precision	recall	f1-score	support
ADJ	0.89	0.89	0.89	57285
ADV	0.82	0.76	0.79	3094
AR	0.80	0.56	0.66	137
CON	0.96	0.97	0.96	25389
DEFAULT	0.00	0.00	0.00	21
DELM	0.99	1.00	0.99	28703
DET	0.98	0.97	0.98	7825
IF	0.86	0.84	0.85	219
INT	0.84	0.83	0.83	532
MORP	0.75	0.23	0.35	176
MQUA	0.37	0.46	0.41	108
MS	0.33	0.55	0.41	62
N	0.96	0.96	0.96	170340
OH	0.00	0.00	0.00	10
OHH	0.00	0.00	0.00	5
P	0.99	0.99	0.99	30785
PP	0.92	0.82	0.87	56
PRO	0.96	0.94	0.95	4493
PS	0.00	0.00	0.00	7
QUA	0.86	0.83	0.85	1417
SPEC	0.74	0.54	0.62	481
V	0.99	0.98	0.98	28320
accuracy			0.95	359465
macro avg	0.68	0.64	0.65	359465
weighted avg	0.95	0.95	0.95	359465

مشاهده میشود که دقت تگ default یا OHH یا PS یا OH و ... برابر صفر است و این به خاطر این است که تعداد تکرار های آنها در پیکره متنی آموزش بسیار کم است و مدل روی تگ های با تکرار بالا بایاس شده است که قابل پیش بینی است.

۲-۳. استفاده از شبکه عصبی ParsBERT – افزایش Learning Rate:

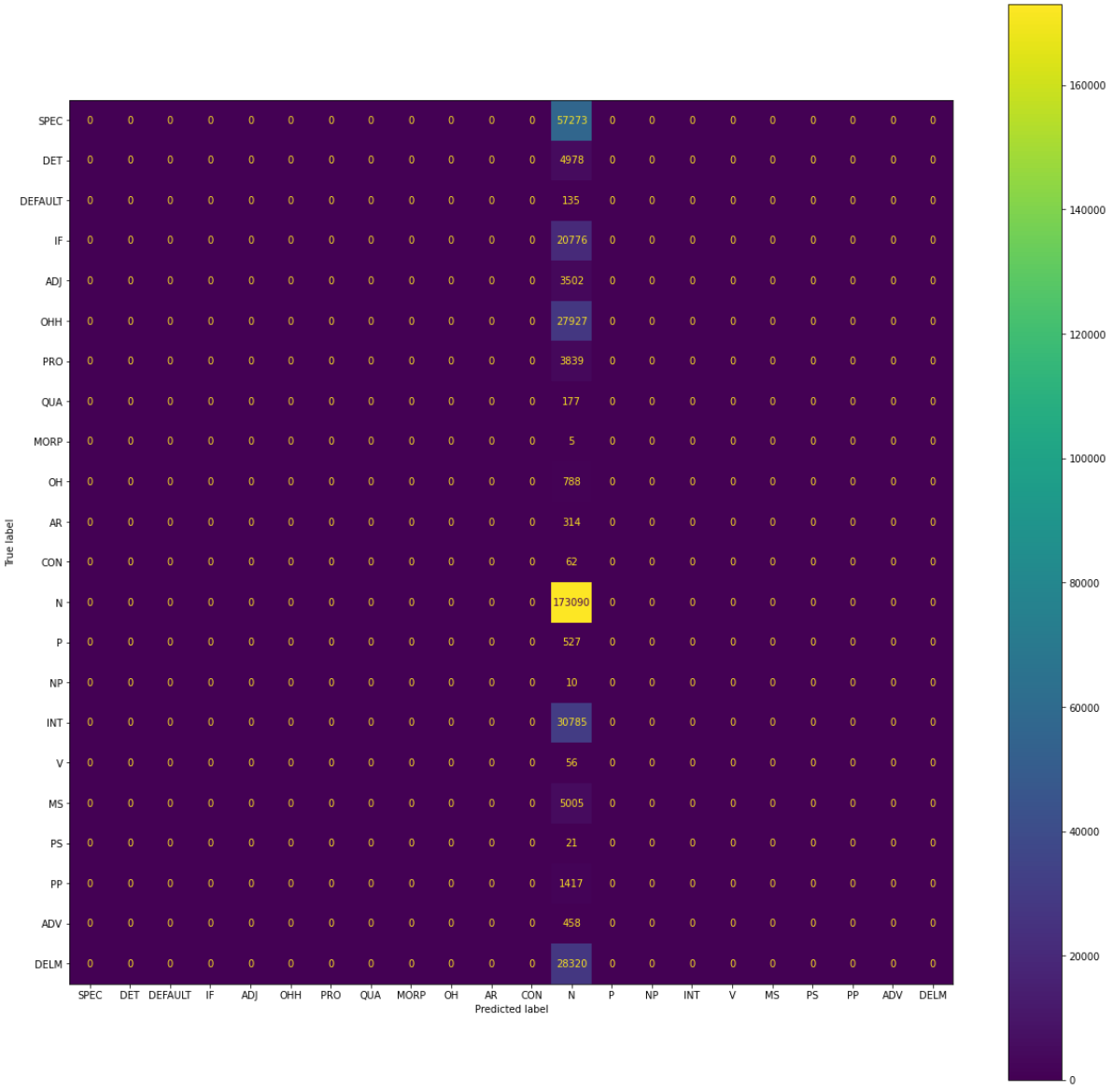
در این قسمت بررسی میکنیم که افزایش Learning Rate روی دقت نهایی چه تاثیری دارد. از همان پارامتر های مدل قبل استفاده میکنیم با این تفاوت که Learning Rate را به مقدار 2e-1 افزایش دادیم . در حین فرآیند fine tuning دقت بر روی داده های validation به صورت زیر است:

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
1	1.804800	1.810564	0.000000	0.000000	0.000000	0.454247
2	1.784400	1.807663	0.000000	0.000000	0.000000	0.454247
3	1.788400	1.806352	0.000000	0.000000	0.000000	0.454247
4	1.785000	1.804465	0.000000	0.000000	0.000000	0.454247

همینطور که مشاهده می شود ، آموزش به درستی انجام نشده است و Loss نه روی داده های Validation و نه داده های آموزشی کم نشده است و ثابت مانده. بنابراین افزایش Learning

Rate تا این میزان تاثیر منفی گذاشته است. پس این تست نشان می دهد که باید برای انتخاب Learning Rate یک tradeoff ای را رعایت کنیم. کم کردن زیاد Learning rate هم باعث میشود که در نهایت دقت بسیار بالایی نگیریم چون مدل نمی تواند خوب fine tune شود و به اصطلاح وزن های اولیه freeze میشوند.

دقت بر روی داده های تست در این مورد برابر **0.48152114948604174** است که اصلا دقت خوبی نیست. دلیل این دقت هم به خاطر این است که به تمام کلمات تگ N میزند و چون تگ N در ۴۸ درصد مواقع اتفاق می افتد پس دقت ۴۸ داریم. ماتریس آشفتگی برای داده های تست به صورت زیر است:



مشاهده میشود فقط recall تگ N دقت ۱ دارد که باعث شده دقت در کل برابر ۴۸ درصد باشد:

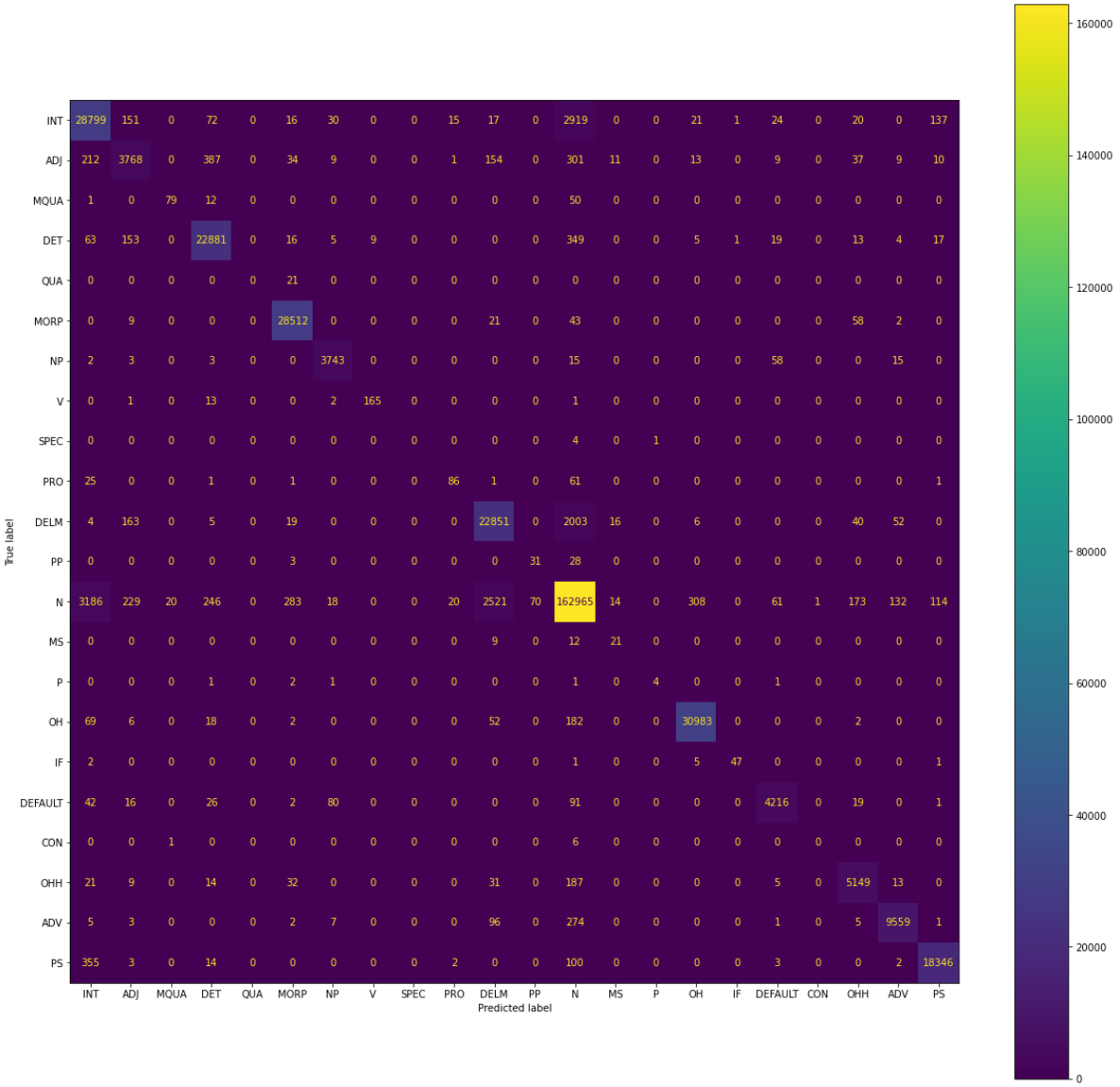
	precision	recall	f1-score	support
ADJ	0.00	0.00	0.00	57273
ADV	0.00	0.00	0.00	4978
AR	0.00	0.00	0.00	135
CON	0.00	0.00	0.00	20776
DEFAULT	0.00	0.00	0.00	3502
DELM	0.00	0.00	0.00	27927
DET	0.00	0.00	0.00	3839
IF	0.00	0.00	0.00	177
INT	0.00	0.00	0.00	5
MORP	0.00	0.00	0.00	788
MQUA	0.00	0.00	0.00	314
MS	0.00	0.00	0.00	62
N	0.48	1.00	0.65	173090
NP	0.00	0.00	0.00	527
OH	0.00	0.00	0.00	10
P	0.00	0.00	0.00	30785
PP	0.00	0.00	0.00	56
PRO	0.00	0.00	0.00	5005
PS	0.00	0.00	0.00	21
QUA	0.00	0.00	0.00	1417
SPEC	0.00	0.00	0.00	458
V	0.00	0.00	0.00	28320
accuracy			0.48	359465
macro avg	0.02	0.05	0.03	359465
weighted avg	0.23	0.48	0.31	359465

۳-۳. استفاده از شبکه عصبی ParsBERT – افزایش Weight Decay:

در این قسمت از هایپر پارامتر های مدل قسمت اول استفاده کردیم و فقط Weight Decay را از مقدار ۰,۰۱ به مقدار ۰. ۱ افزایش دادیم و میخواهیم تاثیر افزایش یا کاهش Weight Decay را مشاهده کنیم. دقت برای داده های Validation در حین fine tuning به صورت زیر است:

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
1	0.202100	0.148251	0.892621	0.897187	0.894898	0.946301
2	0.124300	0.107183	0.921538	0.919182	0.920359	0.961644
3	0.080200	0.103412	0.935385	0.932992	0.934187	0.967945
4	0.054600	0.093899	0.937341	0.941176	0.939255	0.970685
5	0.039100	0.103991	0.936864	0.941176	0.939015	0.969589

مشاهده میشود که کاهش Loss در این آزمایش به نسبت آزمایش ۱-۳ خیلی عوض نشده است. دقت گرفته شده برای داده های تست **0.9519841987397939** است که دقت پایینتری نسبت به آزمایش ۱-۳ است. ماتریس آشفتگی برای داده های تست در این مدل :



و جزئیات و معیار های دیگر برای ارائه دقت مدل:

ADJ	0.88	0.89	0.89	32222
ADV	0.83	0.76	0.80	4955
AR	0.79	0.56	0.65	142
CON	0.97	0.97	0.97	23535
DEFAULT	0.00	0.00	0.00	21
DELM	0.99	1.00	0.99	28645
DET	0.96	0.97	0.97	3839
IF	0.95	0.91	0.93	182
INT	0.00	0.00	0.00	5
MORP	0.69	0.49	0.57	176
MQUA	0.89	0.91	0.90	25159
MS	0.31	0.50	0.38	62
N	0.96	0.96	0.96	170361
NP	0.34	0.50	0.40	42
OH	0.80	0.40	0.53	10
P	0.99	0.99	0.99	31314
PP	0.96	0.84	0.90	56
PRO	0.96	0.94	0.95	4493
PS	0.00	0.00	0.00	7
QUA	0.93	0.94	0.94	5461
SPEC	0.98	0.96	0.97	9953
V	0.98	0.97	0.98	18825
accuracy			0.95	359465
macro avg	0.73	0.70	0.71	359465
weighted avg	0.95	0.95	0.95	359465

مشاهده میشود که همچنان از مشکل موجود در مدل شماره ۱ رنج میبریم یعنی همچنان مدل ما بایاس شده است و بعضی از تگ ها را اصلا استفاده نمی کند.

۳-۴. استفاده از شبکه عصبی MultiLingual Cased:

در این قسمت به جای استفاده از ParsBERT از مدل از پیش آموزش داده شده ی چند زبانه (لینک) استفاده کردیم. هایپر پارامتر های مدل به شکل زیر است:

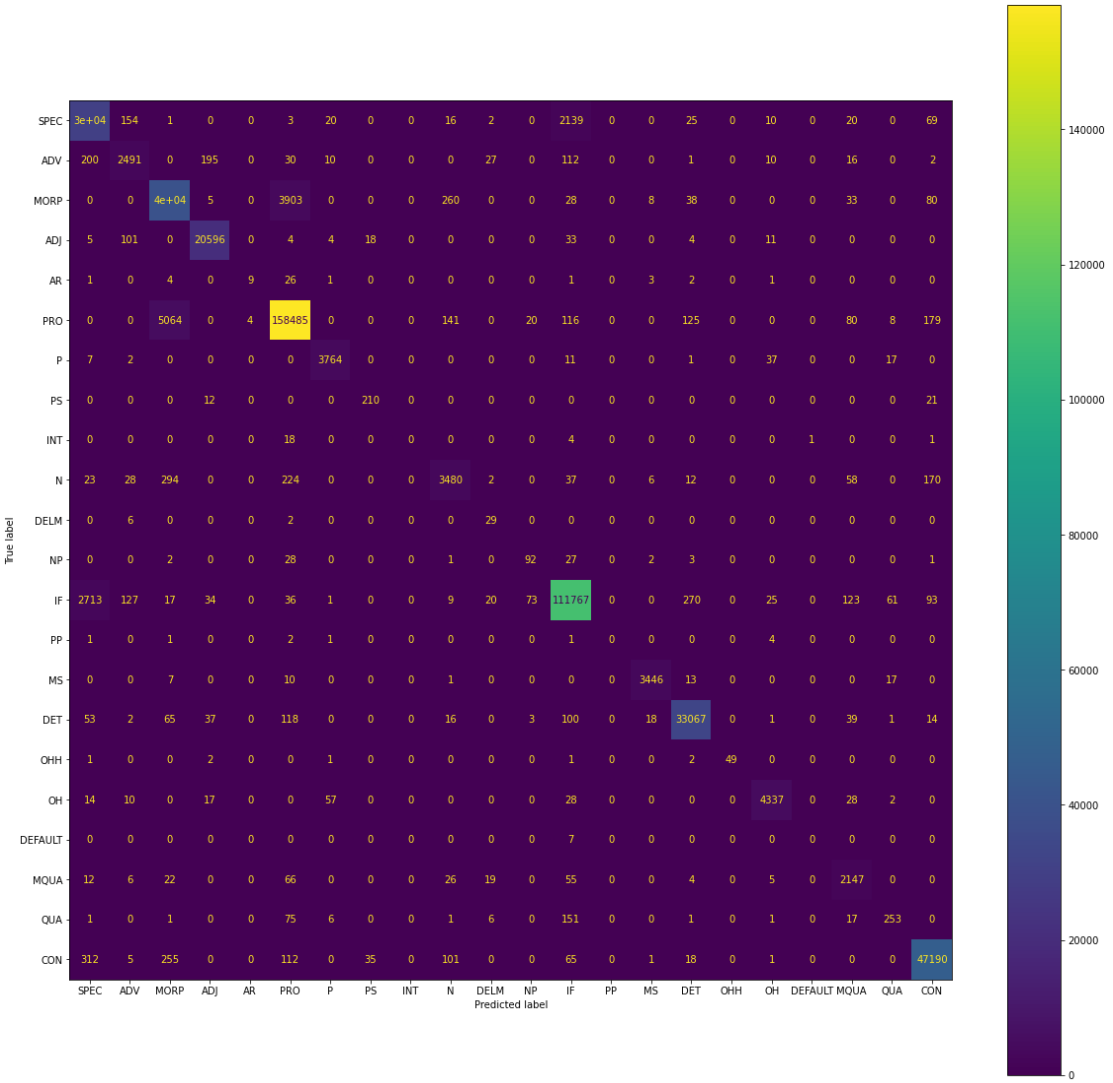
```
args = TrainingArguments(
    model_name,
    evaluation_strategy="epoch",
    save_strategy="epoch",
    learning_rate=2e-5,
    num_train_epochs=5,
    weight_decay=0.01,
)
```

در حین فرآیند fine tuning دقت و Loss برای داده های validation به صورت زیر است:

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
1	0.179800	0.119120	0.950680	0.957520	0.954088	0.967183
2	0.109900	0.092021	0.962521	0.967797	0.965152	0.974431
3	0.077200	0.086604	0.961853	0.967455	0.964646	0.974431
4	0.054100	0.103594	0.961905	0.968825	0.965352	0.974633
5	0.042200	0.102406	0.963543	0.968825	0.966177	0.975639

به نظر میرسد که انگار مدل بهتر آموزش داده شده است چون دقت به مراتب بالاتر رفته است.

با بررسی دقت این مدل روی داده های تست به دقت **0.9582402246238962** رسیدیم که دقتی بالاتر از دقت های گرفته شده در قسمت های قبل است. ماتریس اشفتگی برای این مدل به صورت زیر است:



و معیار های بیشتر برای عملکرد مدل بر روی داده های تست:

	precision	recall	f1-score	support
ADJ	0.90	0.92	0.91	32222
ADV	0.85	0.81	0.83	3094
AR	0.88	0.90	0.89	44588
CON	0.99	0.99	0.99	20776
DEFAULT	0.69	0.19	0.30	48
DELM	0.97	0.97	0.97	164222
DET	0.97	0.98	0.98	3839
IF	0.80	0.86	0.83	243
INT	0.00	0.00	0.00	24
MORP	0.86	0.80	0.83	4334
MQUA	0.28	0.78	0.41	37
MS	0.49	0.59	0.53	156
N	0.97	0.97	0.97	115369
OH	0.00	0.00	0.00	10
OHH	0.99	0.99	0.99	3494
P	0.98	0.99	0.99	33534
PP	1.00	0.88	0.93	56
PRO	0.98	0.97	0.97	4493
PS	0.00	0.00	0.00	7
QUA	0.84	0.91	0.87	2362
SPEC	0.70	0.49	0.58	513
V	0.99	0.98	0.98	48095
accuracy			0.96	481516
macro avg	0.73	0.73	0.72	481516
weighted avg	0.96	0.96	0.96	481516

که مشاهده میشود دقت برای برخی از تگ ها مثل DEFUALT افزایش یافته است و مشکل موجود در مدل شماره ۱-۳ و ۳-۳ کمتر شده است. علت بالا آمدن دقت نیز همین است. چون این روش بهتر از روش های قبل است در قسمت های بعدی کمی با هایپرپارامتر های مدل استفاده شده در ۴-۳ کار میکنیم تا شاید دقت بالاتری دریافت کنیم.

۵-۳. استفاده از MultiLingual Bert Uncased:

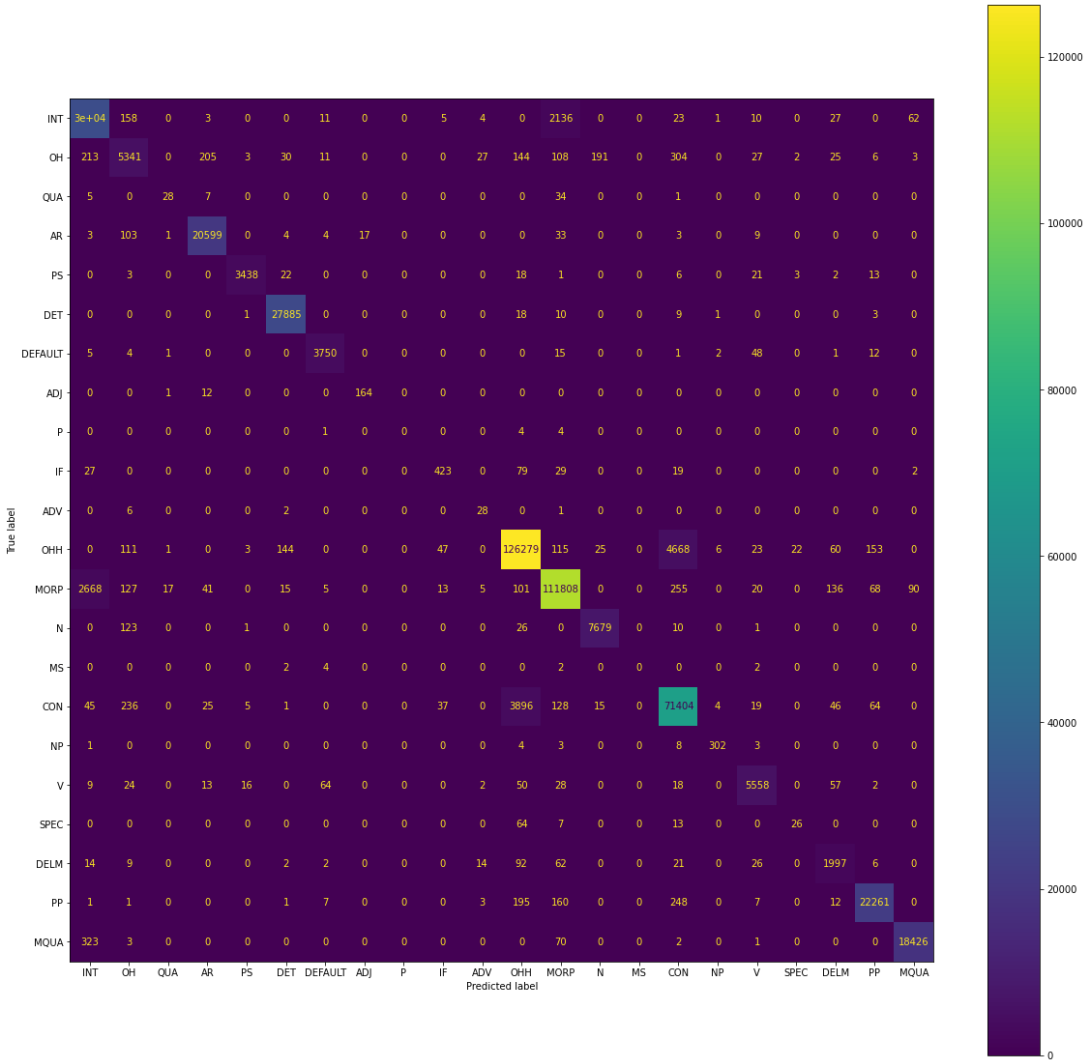
در ابتدا به تفاوت این مدل و مدل قبلی میپردازیم.

مدل های cased موجود در hugging face به این معنی هستند که متن ورودی بدون تغییر به توکن ساز داده میشود و توکن سازی صورت میگیرد ولی در مدل های uncased ابتدا تمام حروف به صورت lowercase نوشته میشوند و سپس به tokenizer داده میشوند. استفاده از این مدل ها شاید کمی دقت را بهتر کند.

مدل استفاده شده در این بخش یک مدل چند زبانه (لینک) است که از سایت hugging face برداشته شده است. بر روی این مدل fine tuning انجام میشود. در حین فرآیند fine tuning دقت گزارش شده برای داده های validation به صورت زیر است:

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
1	0.177300	0.116809	0.955487	0.963829	0.959640	0.967307
2	0.108100	0.086255	0.966181	0.971001	0.968585	0.975685
3	0.077500	0.086297	0.971739	0.975678	0.973705	0.978136
4	0.056000	0.089322	0.973023	0.978485	0.975746	0.979975
5	0.043200	0.088006	0.970606	0.978173	0.974375	0.979158

با توجه به مقادیر بالا و روند کاهش loss انتظار داریم دقت های بهتری برای داده های تست بگیریم که همینطور هم شد. دقت برای داده های تست برابر **0.9587519686608074** است که از تمام مدل هایی که تا حالا ساخته شده است بهتر است. ماتریس آشفتگی برای داده های تست در این مورد به صورت زیر است:



و جزئیات و معیار های دیگر برای ارزیابی مدل روی داده تست به صورت زیر است:

	precision	recall	f1-score	support
ADJ	0.90	0.92	0.91	32222
ADV	0.85	0.80	0.83	6640
AR	0.57	0.37	0.45	75
CON	0.99	0.99	0.99	20776
DEFAULT	0.99	0.97	0.98	3527
DELM	0.99	1.00	1.00	27927
DET	0.97	0.98	0.97	3839
IF	0.91	0.93	0.92	177
INT	0.00	0.00	0.00	9
MORP	0.81	0.73	0.77	579
MQUA	0.34	0.76	0.47	37
MS	0.96	0.96	0.96	131657
N	0.97	0.97	0.97	115369
NP	0.97	0.98	0.98	7840
OH	0.00	0.00	0.00	10
P	0.93	0.94	0.93	75925
PP	0.96	0.94	0.95	321
PRO	0.96	0.95	0.96	5841
PS	0.49	0.24	0.32	110
QUA	0.85	0.89	0.87	2245
SPEC	0.99	0.97	0.98	22896
V	0.99	0.98	0.99	18825
accuracy			0.96	476847
macro avg	0.79	0.79	0.78	476847
weighted avg	0.96	0.96	0.96	476847

به راحتی قابل مشاهده است که دقت در برخی از تگ ها بیشتر شده است. این افزایش دقت باعث افزایش دقت کلی شده است. این بهترین مدلی است که در این تمرین به آن رسیده ایم.

۶-۳. استفاده از ParsBERT v2:

این مدل را از سایت hugging face (لینک) دریافت کردیم. آموزش آن را انجام دادیم. انتظار داریم این مدل از مدل شماره ۱-۳ بهتر عمل کند. دقت بر روی داده های تست در این قسمت برابر **0.9515696938505835** شد که به نسبت دقت های دیگر گرفته شده، دقت بالایی نیست. بنابراین از ارائه جزئیات درمورد این مدل اجتناب میکنیم. میتوانید نتیجه کامل دریافت شده بر روی این مدل را در فایل ipynb. مربوطه که به همراه این فایل توضیحات پیوست شده است مطالعه کنید.

۴- ارائه نتایج آزمایشات

در این قسمت نتایج موجود در قسمت های قبل را در قالب جدول می آوریم تا قابل جمع بندی و مقایسه باشد.

Model Number	Accuracy on Test Data
Model 3-1	0.9524042674530204
Model 3-2	0.48152114948604174
Model 3-3	0.9519841987397939
Model 3-4	0.9582402246238962
Model 3-5	0.9587519686608074
Model 3-6	0.9515696938505835

۵- جمع بندی – ویتربی یا BERT؟

بعد از اجرای الگوریتم ویتربی بر روی داده های تست، با استفاده از تابع accuracy_score که برای عمل classification به کار میرود، مقایسه ای بین تگ های واقعی داده های تست و تگ های بدست آمده پس از اجرای الگوریتم ویتربی انجام دادیم و نتیجه دقت **۰,۹۴۰۳۰۲۷۰۱۳۷۱۰۸۶۳** را به ما داد (نتیجه تمرین ۳). در این تمرین دقت تقریبا برابر با ۹۶ درصد (**۰,۹۵۸۷۵۱۹۶۸۶۶۰۸۰۷۴**) را با استفاده از مدل های از پیش آموزش داده شده ی BERT دریافت کردیم که این نشان میدهد مدل هایی که در این تمرین استفاده کردیم از مدل های HMM که قبلا بررسی کردیم کارا تر هستند. علت این موضوع این است که شبکه های BERT باز هم قابل گسترش و بهبود هستند چون قطعی نیستند و به هایپرپارامتر ها وابسته اند. اما HMM ها این آزادی وجود ندارد و نهایت دقتی که میتوان با آنها گرفت همان ۹۴ درصد است. البته این دقت به انتخاب تعداد و نوع Vocab نیز وابسته است.

پس به طور خلاصه برای انجام عمل POS Tagging با استفاده از BERT دقت بالاتری از زمانی میگیریم که از HMM و الگوریتم ویتربی استفاده میکنیم.