دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده برق و کامپیوتر

گزارش تمرین شماره ۳

درس NLP

بهار ۱۴۰۲

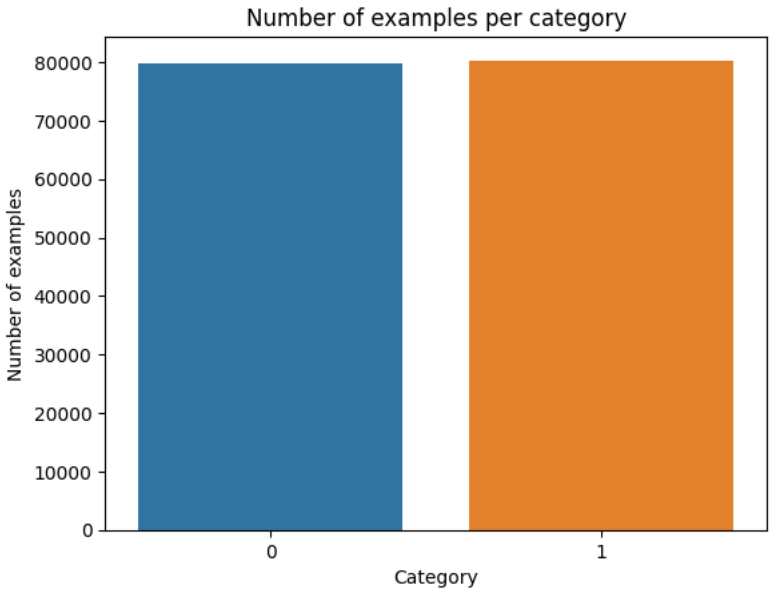
نام و نام خانوادگی
سپهر کریمی آرپناهی

شماره دانشجویی
۸۱۰۱۰۰۴۴۷

▪ مقدمه و مجموعه داده و پیش پردازش

ابتدا داده ها را دریافت کرده و سپس آن را از حالت فشرده خارج می کنیم. سپس دادگان را در دیتافریم ذخیره میکنیم و نام ستون ها را تعیین می کنیم. ستون polarity برای اینکه بدانیم توییت، مثبت، خنثی و یا منفی است. و همینطور ستون text را داریم که در ادامه بر روی آن پیش پردازش ها را انجام خواهیم داد.

در ادامه تابعی را برای انجام پیش پردازش ها تعریف می کنیم. تابع preprocessing چندین وظیفه بر عهده د ارد. اول آن که اموجی هایی مانند :(در صورتی که از میان متن جدا نشوند به عنوان punc حذف خواهند شد. به همین دلیل تو ساختن یکی ارایه از تمامی این اموجی ها، به جای آن های نامی که در امبدینگ های اماده هم موجود باشد گذاشتیم. وظیفه بعدی این تابع حذف یوزرنیم و لینک های موجود در توییت می باشد. همچنین وظیفه بعدی آن حذف کلماتی که حروف تکراری زیادی دارند می باشد. مثل "سلاااام". پس از این اعمال سپس بر اساس اسپیس کلمات تویت را توکنایز کردیم.



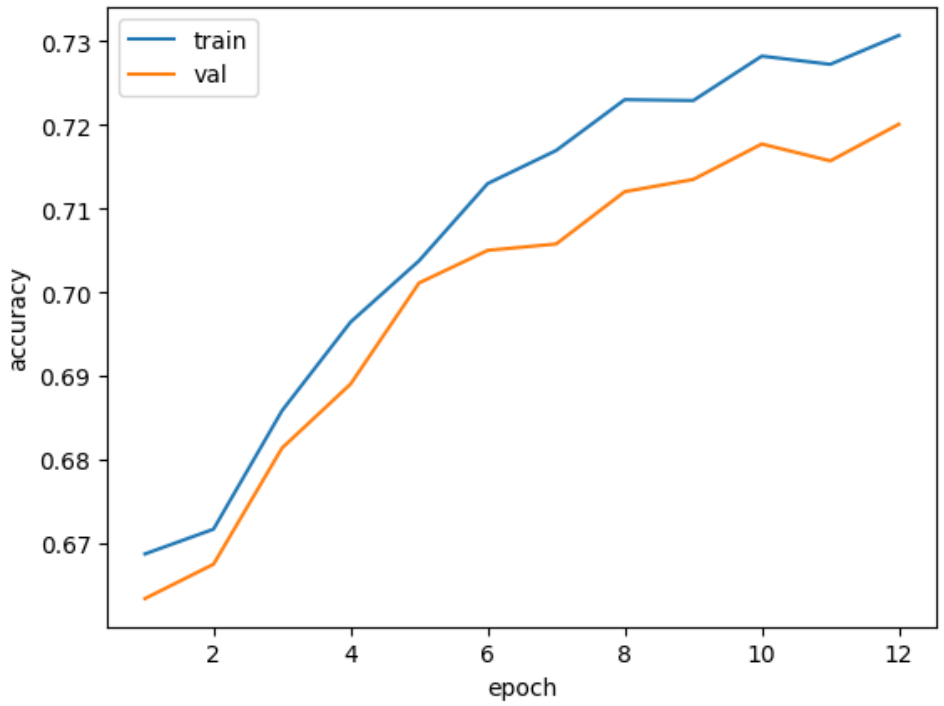
شکل ۱ : نمودار توزیع هر کدام از کلاس ها در دادگان

▪ بخش ۱ : RNN

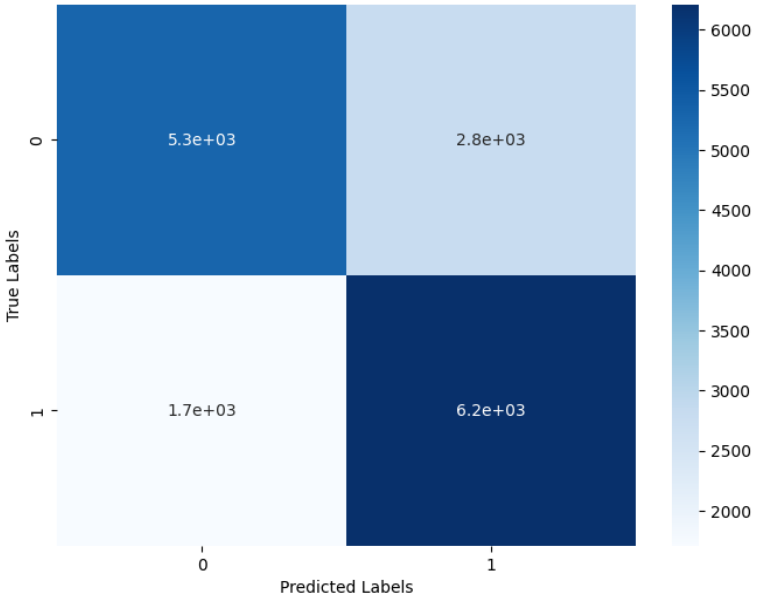
پس از انجام پیش پردازش های گفته شده، ابتدا دیتاست را توکنایز می کنیم. سپس با سه روش انکودینگ گفته شده، امبدینگ کلمات را می سازیم و سپس در هر کدام از روش ها با استفاده از پدینگ طول جملات را یکسان خ.اهیم کرد. در این تمرین با توجه به اینکه میانگین جملات برابر با ۱۳ توکن بود و ۹۹ درصد از داده ها در میان طول ۳۹ قرار می گرفتند، ماکزیمم طول توییت ها را برابر با ۳۹ در نظر گرفتیم و بقیه را دور ریختیم.
سپس طول همه توییت ها را به ۳۹ رساندیم و نتایج هر کدام از روش ها را بدست آوردیم. همچنین برای اطمینان از نتیجه یادگیری انجام شده یک دهم از دادگان یادگیری را برای validation در نظر گرفتیم.

○ Glove

برای این قسمت ابتدا gloveهای با اندازه ۵۰ را دانلود کرده و سپس از میان آن ها کلماتی که در وکب بودند را نگه داشتیم و امبدینگ آن ها را ذخیره کردیم. سپس بر روی این امبدینگ ها rnn را پیاده سازی کردیم و مطابق آنچه در صورت سوال گفته شده نتایج زیر بدست آمد:



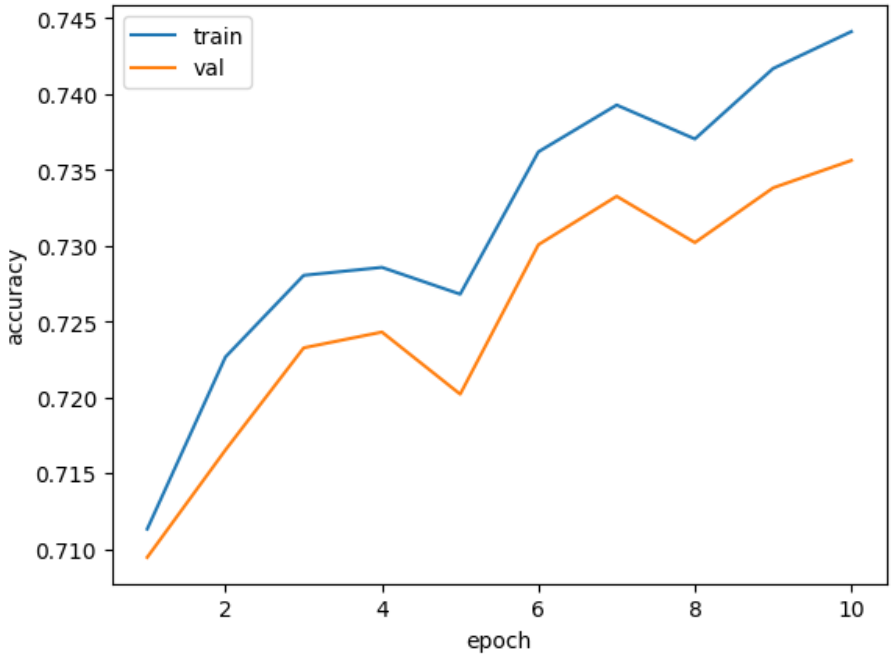
شکل ۲ : نمودار توزیع هر کدام از کلاس ها در دادگان



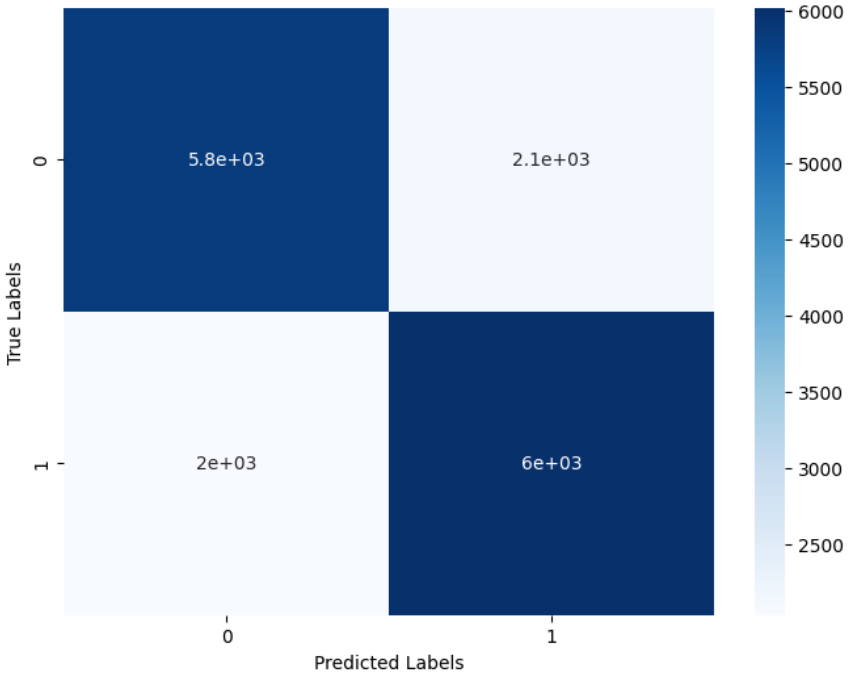
شکل ۳ : نمودار ماتریس درهم ریختگی برای glove

○ Word2vec

مطابق آنچه در بالا گفته شد برای word۲vec نیز پیاده سازی rnn انجام شده و با فراخوانی تابع آماده، امبدینگ های آماده را با استفاده از کتابخانه w۲v genism را استخراج کردیم و یادگیری را بر روی همان امبدینگ ها انجام دادیم که نتایج به شرح زیر بدست آمد:



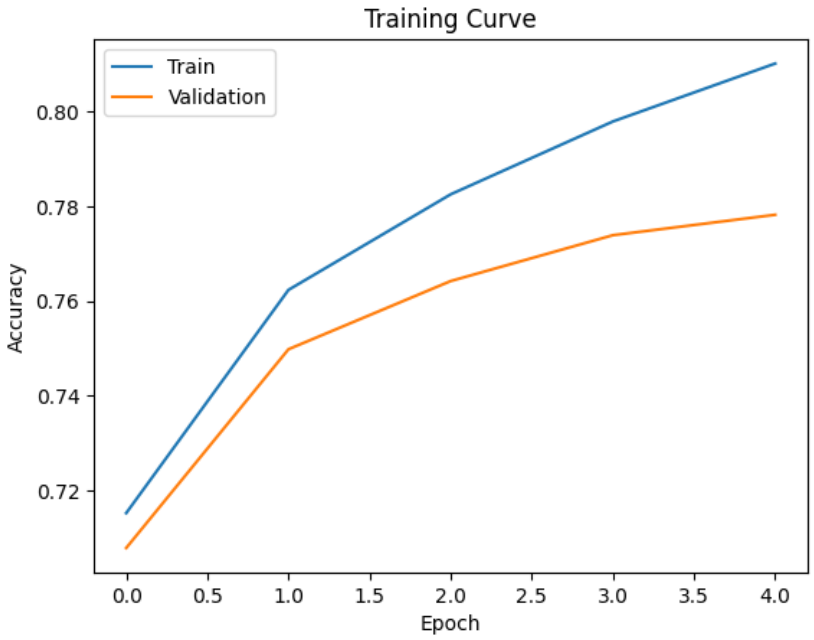
شکل۴ : نمودار توزیع هر کدام از کلاس ها در دادگان برای word2vec



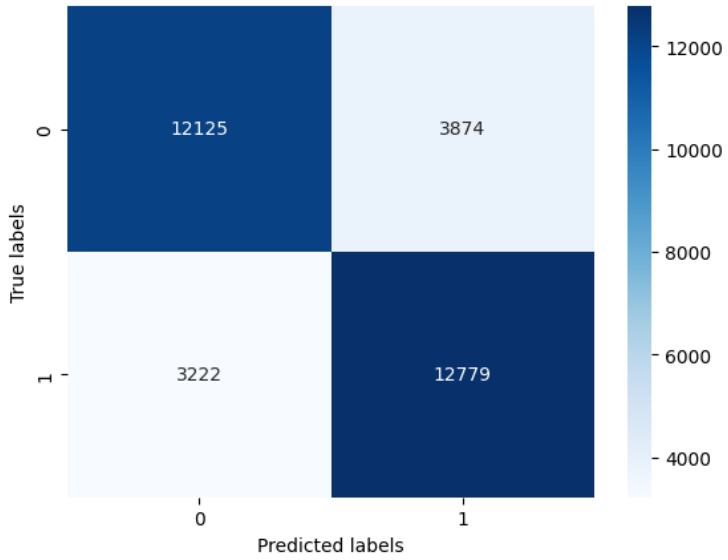
شکل۵ : ماتریس درهم ریختگی هر کدام از کلاس ها در دادگان برای word2vec

One-hot ○

این نوع امبدینگ از نظر مصرف مموری أصلا به صرفه نیست. همینطور در صورت ساخت وتور ها کولب با کمبود مموری موجه می شود. به همین علت ما با دانستن ایندکس هر کدام از کلمات دیکشنری، در هر فراخوانی وکتور one-hot ان را می سازیم. سپس این مدل را از یک لایه خطی با بعد ۱۵۰ عبور می دهیم که مانند ساخت امبدینگ از روی وکتور های one-hot می باشد. نتایج پیاده سازی به شرح زیر می باشد:



شکل۶ : نمودار دقت هر کدام از کلاس ها در دادگان برای one-hot



شکل۷ : ماتریس درهم ریختگی هر کدام از کلاس ها در دادگان برای one-hot

از تحلیل ماتریس برهم ریختگی بر می آید که مقدار precision برابر با ۷۶ درصد ومقدار recall نیز برابر با ۷۴ درصد است. که عدد خوبی می باشد.

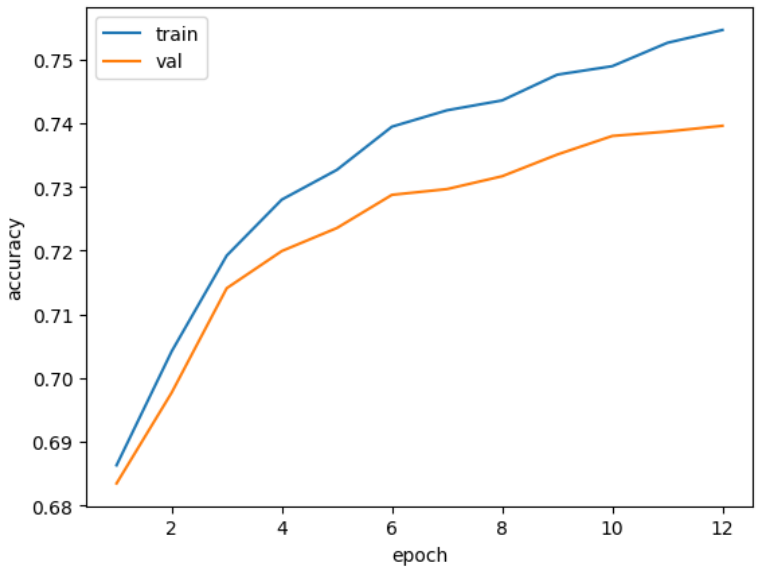
▪ بخش ۲

در این قسمت می خواهیم هم LSTM و هم GRU را به وسیله پایتورچ پیاده سازی کنیم.همچنین از تابع هزینه crossentropy و بهینه سازی adam استفاده کردیم. همچنین تعداد بعد مخفی را برابر با ۱۵۰ گذاشتیم.

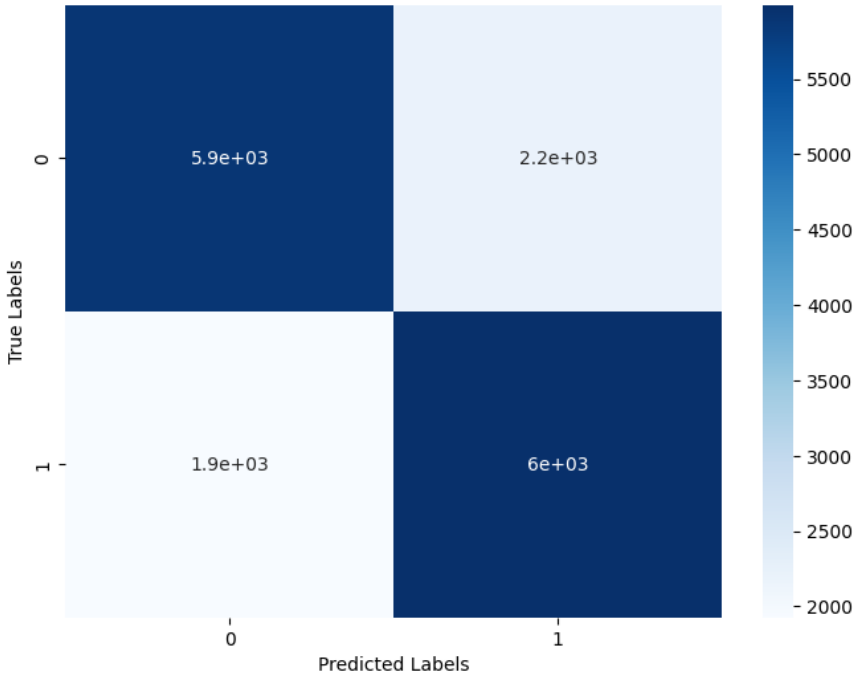
.

Lstm

- Glove

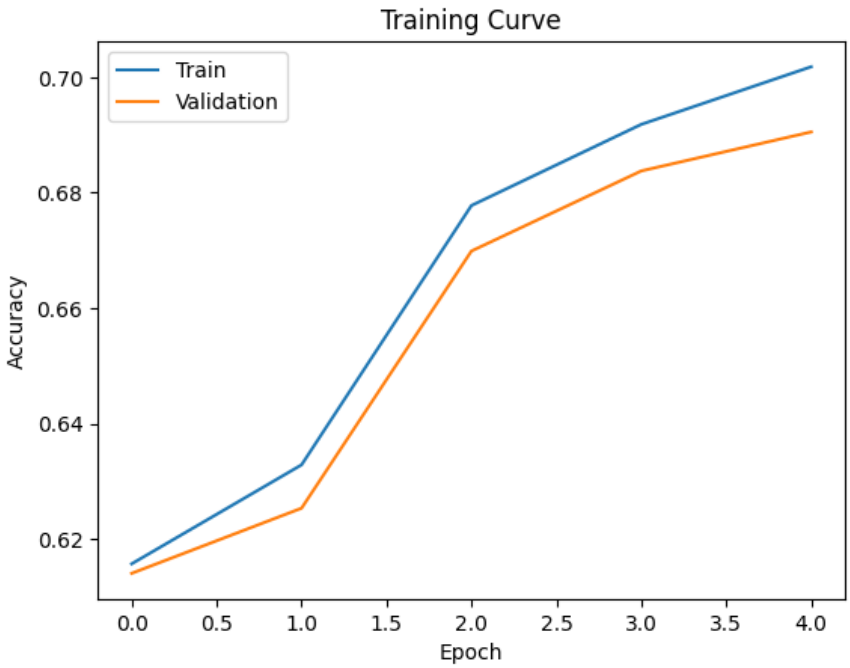


شکل۸ : نمودار دقت هر کدام از کلاس ها در دادگان برای glove-

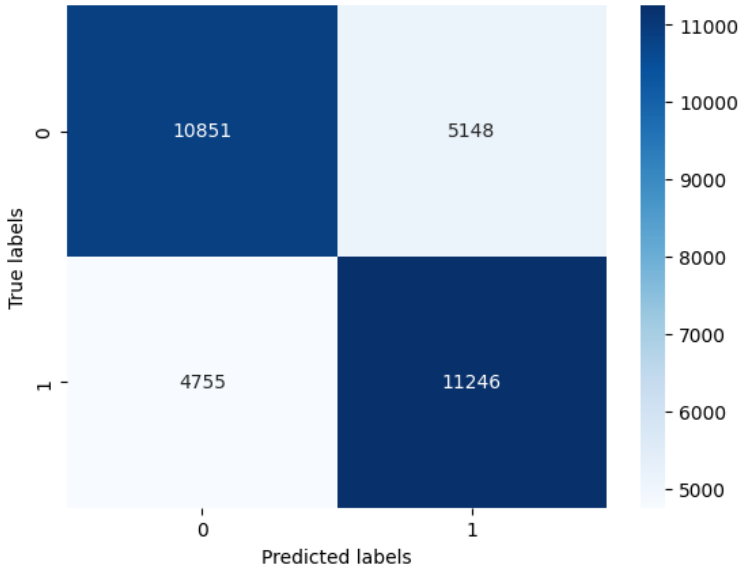


شکل۹ : ماتریس درهم ریختگی هر کدام از کلاس ها در دادگان برای Glove-

One-hot ○



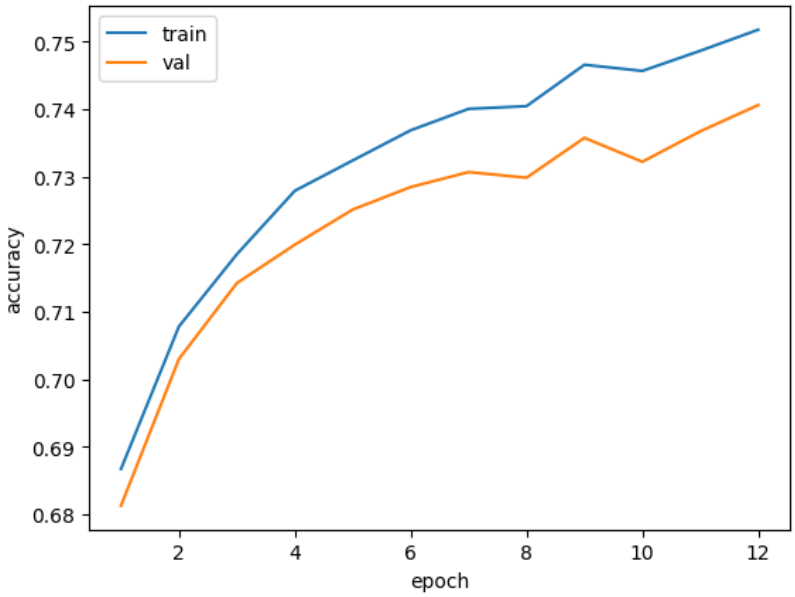
شکل۱۰ : نمودار دقت هر کدام از کلاس ها در دادگان برای one-hot



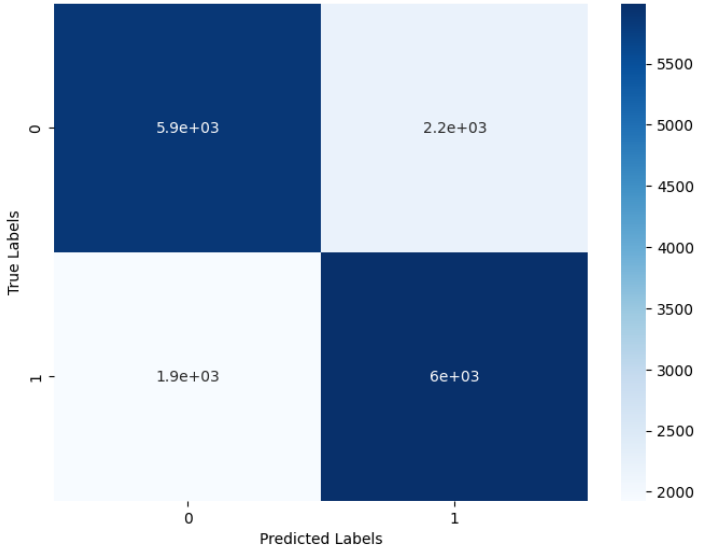
شکل11 : ماتریس درهم ریختگی هر کدام از کلاس ها در دادگان برای one-hot

GRU

Glove ○

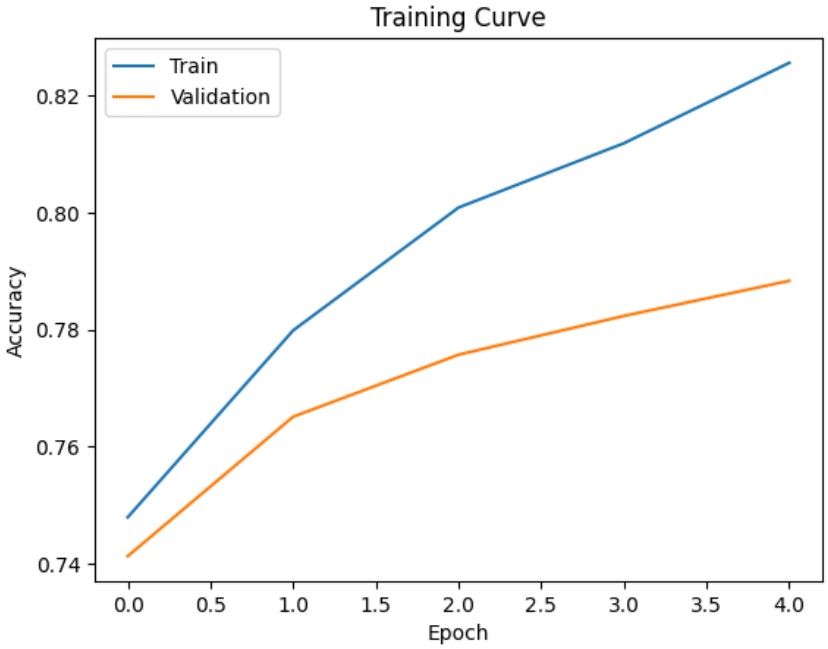


شکل۱۲ : نمودار دقت هر کدام از کلاس ها در دادگان برای glove

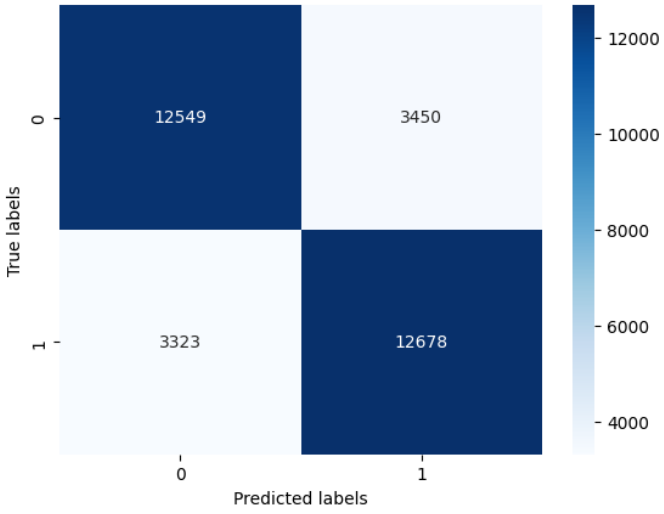


شکل۱۳ : ماتریس درهم ریختگی هر کدام از کلاس ها در دادگان برای glove

One-hot ○



شکل۱۴: نمودار دقت هر کدام از کلاس ها در دادگان برای one-hot



شکل ۱۵ : ماتریس درهم ریختگی هر کدام از کلاس ها در دادگان برای one-hot

- مقایسه دو مدل:

در مقایسه دو مدل می توان به این نکته اشاره کرد که gru در یادگیری سریع تر است و training time آن کوتاه تر است. همچنین در مقایسه نمودار دقت دو مدل، برای هر دو انکودینگ، لقع ابتدا ضعیف تر عمل کرده و در ادامه سریع تر یادگیری را انجام می دهد.