به نام خدا

درس مبانی یادگیری عمیق

گزارش پروژه پایانی

استاد درس : دکتر مرضیه داوودآبادی

دستیاران : مرتضی حاجی‌آبادی، سحر سرکار، فائزه صادقی، مهسا موفق بهروزی، الناز رضایی، پریسا ظفری، حسن حماد، سید محمد موسوی، کمیل فتحی، شایان موسوی نیا، امیررضا ویشته

دانشگاه علم و صنعت ایران، دانشکده مهندسی کامپیوتر

نیمسال اول تحصیلی ۱۴۰۲ - ۱۴۰۳

موضوع:

تحلیل احساسات در متن فارسی

ردیف	نام ونام خانوادگی	شماره دانشجویی
۱	سارا سادات یونسی	۹۸۵۳۳۰۵۳
۲	مهدیه نادری	۹۸۵۲۲۰۷۶

جدول ۱: مشخصات اعضای گروه

۱ شرح موضوع و مجموعه دادگان

بیان و تشخیص احساسات نقش مهمی در زندگی اجتماعی و شغلی انسان دارد .آنها ارتباط نزدیکی با توانایی های شناختی و مهارت های ارتباطی ما دارند و عمیقاً دامنه و کیفیت تجربیات و تعاملات اجتماعی ما را شکل می دهند. شرکت ها در تلاش اند تا بتوانند رفتار، خدمات و محصولات خود را با انتظارات مشتری تطبیق دهند و از این رو رشد کسب و کار را تضمین کنند.

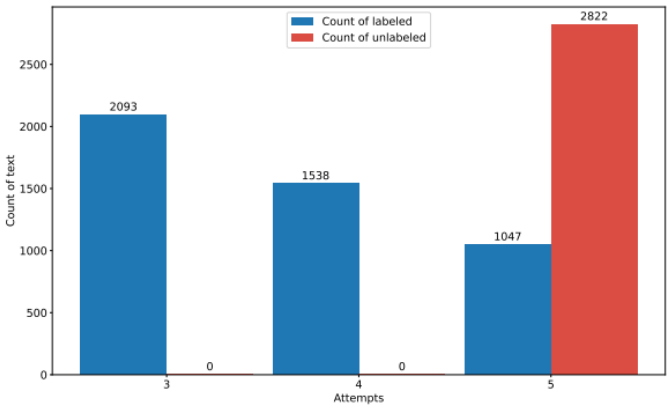
با گسترش اخیر داده های متنی باز در پلتفرم های رسانه های اجتماعی، تشخیص احساسات از متن در سال های گذشته توجه بیشتری را به خود جلب کرده است .این برنامه کاربردهای زیادی دارد، به ویژه برای مشاغل و ارائه دهندگان خدمات آنلاین، که در آن تکنیک های تشخیص احساسات می تواند به آنها کمک کند تا با تجزیه و تحلیل احساسات مشتریان/کاربران نسبت به محصولات و خدمات خود، تصمیمات تجاری آگاهانه بگیرند .در این مطالعه، ArmanEmo را معرفی می‌کنیم، مجموعه‌ای از احساسات با برچسب انسانی متشکل از بیش از ۷۰۰۰ جمله فارسی که برای هفت دسته برچسب‌گذاری شده‌اند . از آنجایی که یکی از اهداف پروژه ارزیابی احساسات نسبت به موضوعات مختلف اجتماعی و سیاسی است از مجموعه داده از منابع مختلف، از جمله نظرات توییتر، اینستاگرام و دیجی کالا جمع آوری شده است .برچسب ها بر اساس شش احساس اصلی اکمن (خشم، ترس، شادی، نفرت، غم، شگفتی) و دسته دیگری (سایر) برای در نظر گرفتن هر احساس دیگری که در مدل اکمن گنجانده نشده است، ساخته شده اند.

Table 1: Resources used to collect textual data

	Persian Tweets	Instagram Comments	Digikala Comments
Collection Method	Tweeter's official API	Facebook Graph API	Polite Crawling of the Website
Collection Period	2017 - mid 2018	mid 2017 - mid 2018	mid 2018
# Raw Data	1.5 M	1 M	50 K
# Data Used for Manual Annotation	3.5 K	3 K	1 K
# Data Used for Automatic Annotation	4.5 K	-	-

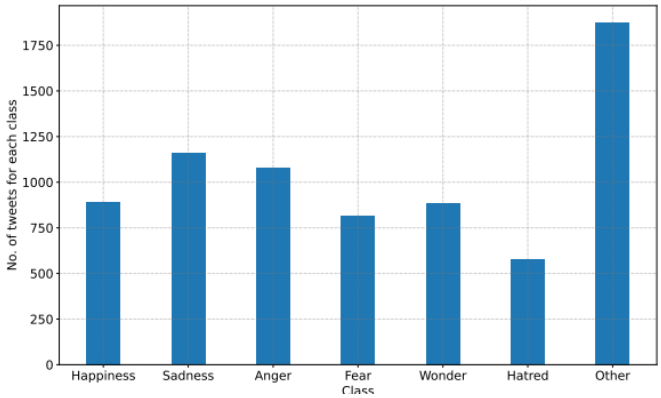
برای بالا بردن کیفیت داده ها ایموجی در متن ها حذف شده تا فقط تاثیر متن لحاظ شود و ایدی یا لینک های موجود در متون حذف شده است. همچنین از روش های هیوریستیکی و دستی برای کلاس بندی استفاده شد و در نهایت ۱۲ هزار جمله را به مرحله ی بعد رسید.در مرحله نرمال کردن و حاشیه نویسی ما ۴۷۰۰ جمله که توسط حاشیه نویسان برچسب گذاری شده بود .این داده ها را به طور تصادفی به آموزش اولیه (بیش از ۳۵۰۰ نمونه) و مجموعه های آزمایش (بیش از ۱۱۰۰ نمونه) تقسیم شد .سپس زبان پارس برت را به خوبی تنظیم شد.

۷۵۰۰ جمله را انتخاب شده تا به صورت دستی به عنوان یکی از کلاس های احساسات یا دو دسته دیگر، یعنی "ناشناخته" و "سایر" برچسب گذاری شوند .شکل ۱ تعداد داده های برچسب دار و بدون برچسب را در هر تلاش (هر تکرار فرآیند حاشیه نویسی) نشان می دهد .ما مشاهده کردیم که برچسب‌های ۳۸ درصد از این ۷۵۰۰ جمله در نهایت بلاتکلیف ماندند یعنی به عنوان یکی از کلاس‌های احساسات برچسب‌گذاری نشدند .این مشاهده تاکیدی است که نشان می‌دهد طبقه‌بندی احساسات از روی متن حتی برای حاشیه‌نویسان انسانی نیز یک کار چالش برانگیز است.



شکل ۱ : نشان دهنده ی دیتا های برچسب دار و بدون برچسب

در شکل ۲ تعداد برچسب برای تمام داده های برچسب گذاری شده (به صورت دستی و خودکار) ارائه شده است .بیست و پنج درصد از جملات برچسب گذاری شده نهایی به عنوان "سایر" طبقه بندی شده بودند .علاوه بر این، برخی از داده ها در پایان فرآیند حاشیه نویسی به عنوان "ناشناخته" برچسب گذاری شدند که تصمیم گرفته شد در مقاله آنها را از مجموعه داده حذف شود.



شکل ۲ : نشان دهنده ی برچسب ها برای هر کلاس موجود

۲ پیش‌پردازش داده ها

داده‌های خام اغلب حاوی نقص‌ها و ناسازگاری‌های مختلفی هستند که می‌تواند مانع تحلیل و مدل‌سازی دقیق شود. این مسائل می‌تواند به‌دلیل خطاهای انسانی، اشتباه واردکردن داده‌ها، مقادیر ازدست‌رفته، نقاط پرت، ویژگی‌های نامربوط یا فرمت‌های ناسازگار باشد. هدف پیش‌پردازش داده‌ها رفع این چالش‌ها با اطمینان از قابل اعتماد، دقیق و کامل‌بودن داده‌هاست که درنهایت کیفیت تحلیل و پیش‌بینی‌ها را بهبود می‌بخشد.

- کاهش نویز و خطا در داده ها: داده های واقعی ممکن است حاوی نویز، خطا، تناقض، ناقص بودن و یا پرت بودن باشند که میتوانند روی عملکرد مدلهای یادگیری عمیق تأثیر منفی بگذارند. پیش پردازش داده ها میتواند با استفاده از روشهایی مانند حذف، جایگزینی، تصحیح و یا فیلتر کردن داده های نامناسب، کیفیت داده ها را بهبود بخشد و مدلهای یادگیری عمیق را قابل اعتمادتر کند.
- تبدیل و تنظیم داده ها: داده های واقعی ممکن است شامل انواع مختلفی از داده ها باشند که ممکن است با مدلهای یادگیری

عمیق سازگار نباشند. پیش پردازش داده ها میتواند با استفاده از روشهایی مانند تبدیل داده های متنی، صوتی و یا تصویری به داده های عددی، نرمال سازی، استانداردسازی و یا کد گذاری داده ها، داده ها را به شکلی تبدیل و تنظیم کند که مدلهای یادگیری عمیق بتوانند بهتر با آنها کار کنند.

- تبدیل و تنظیم داده ها: داده های واقعی ممکن است شامل انواع مختلفی از داده ها باشند که ممکن است با مدلهای یادگیری عمیق سازگار نباشند. پیش پردازش داده ها میتواند با استفاده از روشهایی مانند تبدیل داده های متنی، صوتی و یا تصویری به داده های عددی، نرمال سازی، استانداردسازی و یا کد گذاری داده ها، داده ها را به شکلی تبدیل و تنظیم کند که مدلهای یادگیری عمیق بتوانند بهتر با آنها کار کنند.
- تبدیل و تنظیم داده ها: داده های واقعی ممکن است شامل انواع مختلفی از داده ها باشند که ممکن است با مدلهای یادگیری عمیق سازگار نباشند. پیش پردازش داده ها میتواند با استفاده از روشهایی مانند تبدیل داده های متنی، صوتی و یا تصویری به داده های عددی، نرمال سازی، استانداردسازی و یا کد گذاری داده ها، داده ها را به شکلی تبدیل و تنظیم کند که مدلهای یادگیری عمیق بتوانند بهتر با آنها کار کنند.
- تبدیل و تنظیم داده ها: داده های واقعی ممکن است شامل انواع مختلفی از داده ها باشند که ممکن است با مدلهای یادگیری عمیق سازگار نباشند. پیش پردازش داده ها میتواند با استفاده از روشهایی مانند تبدیل داده های متنی، صوتی و یا تصویری به داده های عددی، نرمالسازی، استانداردسازی و یا کدگذاری داده ها، داده ها را به شکلی تبدیل و تنظیم کند که مدلهای یادگیری عمیق بتوانند بهتر با آنها کار کنند.
- برخی از روشهای پیش پردازش داده ها در پروژه های یادگیری عمیق عبارتند از: حذف یا جایگزینی داده های ناموجود، ناقص و یا نا معتبر , شناسایی و حذف داده های پرت و نویز, نرمال سازی، استانداردسازی و یا مقیاس بندی داده ها , انتخاب ویژگی، کاهش بعد و یا خوشه بندی داده ها ,تبدیل داده های متنی، صوتی و یا تصویری به داده های عددی , کد گذاری داده های دستهای، متنی و یا ترتیبی ,افزایش داده و یا تولید داده جدید

همچنین در خود مقاله گفته شد که هنگام جمع کردن محموعه داده برخی موارد نویزی دور ریخته شدند یا مواردی شامل آیدی و لینک بودند تمیز شدند یا استفاده نشدند همچنین روش های پیش پردازشی که داخل مقاله گفته شد می توان به حذف کردن کاراکتر های انگلیسی و کاراکتر های عددی فارسی و برخی کاراکتر ها مانند هشتگ و کاراکتر های انگلیسی و یا نرمال کردن دیتا با pasrvier که قواعد نگارش و زبان را روی متن اعمال می کند و حرکت های عربی را از روی لغات فارسی حذف می کند اشاره نمود همچنین حروف برای تاکید هم حذف شدند و حروف پشت سر هم امده برای تاکید تبدیل به یک حرف شدند.

روشی که ما در پروژه ی خود پیش پردازش را انجام دادیم به شرح زیر است :

از کتابخانه hazm برای نرمالایز کردن و لمتایز کردن یعنی به ترتیب عادی کردن متون اعم از حذف فاصله های اضافی، رفع پیوست ها و جایگزینی حروف عربی با حروف فارسی و یافتن شکل پایه ی هر کلمه را خواهیم داشت. یک تابع خواهیم داشت که متن را میگیرد و تمیز شده ان را برمی گرداند.

• ابتدا متن را گرفته و کاراکترهای # و _ را با فاصله جایگزین میکند. زیرا هشتگها و خط فاصله ها ممکن است برای تحلیل معنایی متن مفید نباشند و یا باعث ایجاد نویز شوند. برای مثال، اگر متن شامل هشتگهایی مانند #سیاسی یا #فوتبال باشد، این هشتگها ممکن است موضوع متن را نشان ندهند و یا با کلمات مرتبط با آنها اشتباه گرفته شوند. بنابراین، برای نرمالسازی دادهها و کاهش ابعاد ویژگیها، ممکن است بهتر باشد که از متن هشتگ ها را حذف کنیم. البته این بستگی به هدف و ماهیت پروژه دارد و ممکن است در بعضی موارد

هشتگ ها اطلاعات مفیدی را ارائه کنند.

• سپس متن را با استفاده از کلاس Normalizer نرمالسازی میکند. این کلاس کارهایی مانند حذف اعراب، اصلاح فاصله گذاری، جایگزین کردن ارقام فارسی و کوتیشن ها را انجام می دهد. به دلیل این کار را میکنیم تا متن را به شکل استاندارد و یکنواخت درآوریم. این کار میتواند برای پردازش های بعدی متن مانند تحلیل دستوری، تشخیص موجودیت های نا مدار، ترجمه ماشینی و غیره مفید باشد. برای مثال، اگر متن شامل اعراب باشد، ممکن است باعث ایجاد ابهام و سردرگمی در تفسیر معنایی کلمات شود. یا اگر متن شامل ارقام فارسی و انگلیسی مختلط باشد، ممکن است باعث اختلال در مقایسه و محاسبه اعداد شود. یا اگر متن شامل کوتیشن های مختلف باشد، ممکن است باعث اشتباه در تشخیص نقل قولها و گفتگوها شود. بنابراین، با استفاده از کلاس Normalizer میتوانیم متن را از این نوع ناهماهنگی ها پاکسازی کنیم.

• بعد از آن متن را با استفاده از تابع word_tokenize به توکن های جدا شده توسط فاصله یا نشانه های سجاوندی تقسیم میکند. به دلیل اینکه تا بتوانیم واحدهای معنایی متن را شناسایی کنیم. توکنها میتوانند کلمات، عبارات، نشانه ها یا هر چیز دیگری باشند که معنا یا نقشی در متن داشته باشند. با تقسیم متن به توکنها، میتوانیم پردازشهای بعدی متن را راحت تر و سریعتر انجام دهیم. برای مثال، اگر بخواهیم تعداد کلمات متن را بشماریم، میتوانیم از تعداد توکن هایی که کلمه هستند استفاده کنیم. یا اگر بخواهیم متن را به زبان دیگری ترجمه کنیم، میتوانیم با توجه به توکن ها و معادلهای آنها در زبان مقصد، متن ترجمه شده را تولید کنیم.
برای نمونه، اگر متن زیر را داشته باشیم:
من امروز به مدرسه رفتم و یک کتاب جدید خریدم.
میتوانیم این متن را با استفاده از تابع word_tokenize به توکنهای زیر تقسیم کنیم:
['من', 'امروز', 'به', 'مدرسه', 'رفتم', 'و', 'یک', 'کتاب', 'جدید', 'خریدم', '.']
همانطور که میبینید، هر کلمه یک توکن است و نشانهی نقطه هم یک توکن است. حالا میتوانیم با استفاده از این توکن ها، متن را به زبان انگلیسی ترجمه کنیم:
['.' ,'I', 'today', 'to', 'school', 'went', 'and', 'a', 'book', 'new', 'bought']
یا میتوانیم تعداد کلمات متن را بشماریم:۱۰

• سپس برای هر توکن، یک حلقه اجرا میکند که در آن تمام علائم نگارشی موجود در لیست puncs را از توکن حذف میکند. این لیست شامل علائمی مانند ویرگول، نقطه، دونقطه و گیومه است. به دلیل اینکه بتوانیم توکنها را به شکل ساده و یکسان درآوریم. این کار میتواند برای پردازش های بعدی متن مانند ریشه یابی، تشخیص موجودیت های نامدار، تحلیل احساسات و غیره مفید باشد. برای مثال، اگر متن شامل علائم نگارشی باشد، ممکن است باعث ایجاد ابهام و سردرگمی در تفسیر معنایی توکن ها شود. یا اگر متن شامل گیومه باشد، ممکن است باعث اشتباه در تشخیص نقل قولها و گفتگوها شود. بنابراین، با حذف این علائم، میتوانیم متن را از این نوع ناهماهنگی ها پاکسازی کنیم.
برای نمونه، اگر متن زیر را داشته باشیم:
«من امروز به مدرسه رفتم و یک کتاب جدید خریدم.»
میتوانیم این متن را با استفاده از تابع word_tokenize به توکنهای زیر تقسیم کنیم:
['«', 'من', 'امروز', 'به', 'مدرسه', 'رفتم', 'و', 'یک', 'کتاب', 'جدید', 'خریدم', '.', '»']
همانطور که میبینید، توکنهای اول و آخر گیومه هستند و توکن قبل از آخر نقطه است. این توکنها برای معنای متن اهمیتی ندارند و میتوانیم آنها را حذف کنیم. برای این کار، برای هر توکن، یک حلقه اجرا میکنیم که در آن تمام علائم نگارشی موجود در لیست puncs

را از توکن حذف میکنیم. این لیست شامل علائمی مانند ویرگول، نقطه، دونقطه و گیومه است. بنابراین، بعد از اجرای این حلقه، توکنهای جدید به شکل زیر خواهند بود:

['', 'من', 'امروز', 'به', 'مدرسه', 'رفتم', 'و', 'یک', 'کتاب', 'جدید', 'خریدم', '', '']

حالا میتوانیم توکنهای خالی را هم از لیست حذف کنیم و به توکنهای نهایی برسیم:

['من', 'امروز', 'به', 'مدرسه', 'رفتم', 'و', 'یک', 'کتاب', 'جدید', 'خریدم']

• سپس توکن هایی را که طول شان کمتر یا مساوی یک کاراکتر است یا فقط شامل ارقام هستند از لیست حذف میکند.

برای نمونه، اگر متن زیر را داشته باشیم:

من ۲ تا برادر و ۳ تا خواهر دارم. اسم برادرهام علی و مهدی و اسم خواهرهام زهرا، فاطمه و مریم است.

میتوانیم این متن را با استفاده از تابع word_tokenize به توکنهای زیر تقسیم کنیم:

['من', '۲', 'تا', 'برادر', 'و', '۳', 'تا', 'خواهر', 'دارم', '.', 'اسم', 'برادرهام', 'علی', 'و', 'مهدی', 'و', 'اسم', 'خواهرهام', 'زهرا', '،', 'فاطمه', 'و', 'مریم', 'است', '.']

همانطور که میبینید، توکن های دوم و ششم فقط شامل ارقام هستند و توکن های پنجم، چهاردهم، بیست و دوم و بیست و پنجم فقط شامل حروف اضافه هستند. این توکنها برای معنای متن اهمیتی ندارند و میتوانیم آنها را حذف کنیم. برای این کار، برای هر توکن، یک شرط بررسی میکنیم که آیا طولش کمتر یا مساوی یک کاراکتر است یا فقط شامل ارقام است. اگر شرط برقرار باشد، توکن را از لیست حذف میکنیم. بنابراین، بعد از اجرای این شرط، توکنهای جدید به شکل زیر خواهند بود:

['من', 'تا', 'برادر', 'تا', 'خواهر', 'دارم', 'اسم', 'برادرهام', 'علی', 'مهدی', 'اسم', 'خواهرهام', 'زهرا', '،', 'فاطمه', 'مریم', 'است']

این توکنها حاوی اطلاعات معنایی متن هستند و میتوانیم با استفاده از آنها، متن را به زبان دیگری ترجمه کنیم یا تحلیلهای دیگری روی آنها انجام دهیم.

• در نهایت، اگر خط زیرین از کد فعال باشد، توکنها را با استفاده از کلاس Lemmatizer ریشه یابی میکند. این کلاس براساس لیستی از کلمات مرجع و ریشه های آنها، کلمات را به شکل معنایی پایه ای تبدیل میکند.به دلیل اینکه شاید کلماتی که فعل هستند به نحوه های گوناگونی نوشته شوند اما در نهایت همگی یک معنی دارند پس یافتن ریشه بهترین راه حل است.

• در آخر، توکن ها را با فاصله به هم میچسباند و به عنوان خروجی تابع برمیگرداند.

۳ انتخاب مدل

مدل‌های پایه برای طبقه‌بندی احساسات بر روی ArmanEmo ارائه می‌کنیم که از یادگیری انتقالی استفاده می‌کنند در این مسئله را با توجه به مدل های ذکر شده در مقاله بررسی می کنیم.

واقعیت این است که استفاده از یادگیری تحت نظارت برای مشکلاتNLP ، از جمله تشخیص احساسات، دشوار است، زیرا برچسب گذاری ها پرهزینه هستند .اینجا جایی است که یادگیری انتقالی وارد عمل می شود. یادگیری انتقالی از مدل های زبان عصبی عمیق (LMs)از پیش آموزش دیده به سمت مشکلات زبانی پایین دست، منجر به عملکردی پیشرفته در چندین کار NLP در سال های اخیر شده است Deep LMها را می توان به طور موثر و به شیوه ای بدون نظارت روی مجموعه داده های بدون برچسب بسیار بزرگ از قبل آموزش داد .به این ترتیب، LMهای به دست آمده دانش زبانی غنی و غیر پیش پاافتاده را به دست خواهند آورد، و آنها را برای انتقال به حوزه و کار هدف بعدی از طریق نظارت مناسب می کند .برای تطبیق دقیق با یک دامنه هدف، LMهای از پیش آموزش دیده باید توسط مقدار کمی از داده های برچسب گذاری شده از آن دامنه به خوبی تنظیم شوند.

به عنوان یکی از مدل‌های پایه، ما از یک مدل زبان از پیش آموزش‌دیده‌شده برای فارسی، معروف به ParsBERT بهره می‌بریم . ParsBERTیک مدل زبان تک زبانه است که بر اساس معماری ترانسفورماتور نمایش رمزگذار دوطرفه (BERT) است .فراهانی و همکاران نشان داده‌اند که مدل ParsBERT در چندین کار پایین‌دستی NLP فارسی از جمله طبقه‌بندی متن و تحلیل احساسات از BERTچند زبانه و مدل‌های قبلی بهتر عمل می‌کند.

همچنین از دو نوع مدل به نام XLM-RoBERTa به عنوان سایر مدل های پایه خود استفاده شد.یکی دیگر از مدل‌های زبان پوشانده مبتنی بر ترانسفورماتور است که از قبل روی متن به ۱۰۰ زبان آموزش داده شده است .این مدل زبان چندزبانه منجر به عملکردی پیشرفته در طبقه‌بندی بین زبانی، برچسب‌گذاری دنباله‌ای و پاسخ‌گویی به سؤالات شده است که در معیارهای چندزبانی مختلف از BERTچند زبانه (mBERT) بهتر عمل می‌کند .اگرچه مشخص است که ParsBERT به عنوان یک مدل زبان تک زبانه بهتر از BERTچند زبانه در کارهای مختلف در زبان فارسی عمل می کند، تصمیم گرفته شد عملکرد تغییرات XLM-RoBERTa (یعنی XLM-RoBERTa-baseو XLM-RoBERTa large) را در مقابل ParsBERT در زمینه تشخیص احساسات در زبان فارسی مقایسه کنیم

مدل دیگری که در خطوط پایه گنجانده شده است XLM-EMO است، یک مدل تشخیص احساسات چند زبانه برای متن رسانه های اجتماعی .اساساً XLM-T بر روی مجموعه داده ها برای تشخیص احساسات در ۱۹ زبان مختلف تنظیم شده است .خود نسخه تنظیم شده XLM-RoBERTa-base بر روی داده های توییتر است .در توسعه مدلXLM-EMO ، برچسب‌های احساسات هر مجموعه داده به یک مجموعه مشترک، یعنی شادی، خشم، ترس و غم نگاشت می‌شوند .نشان داده شده است که این مدل احساسات چندزبانه با خطوط پایه خاص زبان در موارد با داده های کم رقابتی است.

احساسات از دست رفته از ArmanEmo فیلتر شد تا مدل ها فقط شادی، خشم، ترس و غم را پیش بینی کنند .علاوه بر این، برای مقایسه عملکرد XLM-EMO با مدل‌های دیگر در خطوط پایه، مدل XLM-EMO دیگری را نیز با در نظر گرفتن هر هفت کلاس احساسی در مجموعه داده اصلی خود آموزش داده و ارزیابی شد .برای تنظیم دقیق زبان یا مدل‌های احساسی از پیش آموزش‌دیده شده یعنی تشخیص احساسات، یک لایه متراکم کاملاً متصل در بالای این مدل‌ها اضافه می‌کنیم .همچنین یک تابع از دست دادن آنتروپی متقابل را برای انجام وظیفه طبقه‌بندی چند برچسبی با استفاده از مدل‌های حاصل معرفی‌شد .در حین تنظیم دقیق، تمام وزن های شبکه به جز وزنه های آخرین لایه، یعنی لایه متراکم اضافه شده را منجمد کردند.

جدول۲ : مقایسه بین عملکرد مدل های مختلف یادگیری عمیق و مدل های زبانی

Model	Precision (Macro)	Recall (Macro)	F1 (Macro)
FastText [42]	54.82	46.37	47.24
HAN [43]	49.56	44.12	45.10
RCNN [44]	50.53	48.11	47.95
RCNNVariant	51.96	48.96	49.17
TextAttBiRNN [45, 46]	54.66	46.26	47.09
TextBiRNN	51.45	47.16	47.14
TextCNN [47]	58.66	51.09	51.47
TextRNN [48]	49.39	47.20	46.79
ParsBERT	67.10	65.56	65.74
XLM-Roberta-base	72.26	68.43	69.21
XLM-Roberta-large	**75.91**	**75.84**	**75.39**
XLM-EMO-t	70.05	68.08	68.57

مدل انتخابی xlm-roerta-large بود گرچه قبل از این مدل روی مدل های Bert و ParsBert هم کار کردیم ولی به دقت مورد نظر خود نرسیدیم. با این مدل به دقت بالای ۷۰ درصد رسیدیم در حالی که با دو مدل دیگر نهایت دقت ما ۶۲ درصد بود همانطور که در جدول مقاله مشخص بود این مدل عملکرد بسیار خوبی دارد.

```
class SentimentModel(tf.keras.Model):
    def __init__(self):
        super(SentimentModel, self).__init__()
        self.model = TFAutoModel.from_pretrained("xlm-roberta-large")
        self.dropout = tf.keras.layers.Dropout(0.1)
        self.output_layer = tf.keras.layers.Dense(7)

    def call(self, x):
        bert_output = self.model(x)
        d_out = self.dropout(bert_output.pooler_output)
        out = self.output_layer(d_out)

        return out
tokenizer = AutoTokenizer.from_pretrained("xlm-roberta-large", model_max_length=64)
model = SentimentModel()
```

شکل ۳ : نمایی از مدل یادگیری انتقالی استفاده شده در پروژه

در توضیح این مدل امده است :

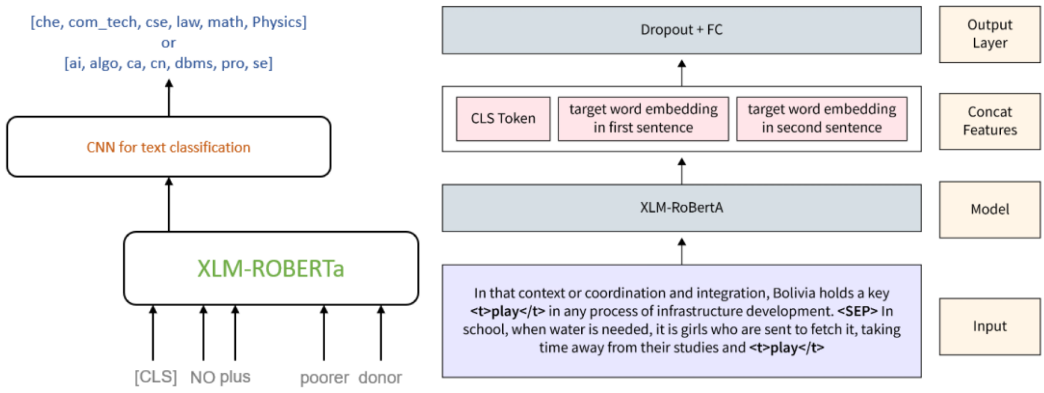
XLM-RoBERTaیک نسخه چند زبانه RoBERTa است . از قبل روی ۲.۵ ترابایت داده CommonCrawl فیلتر شده حاوی ۱۰۰ زبان آموزش داده شده است RoBERTa .یک مدل ترانسفورماتور است که بر روی یک بدنه بزرگ به روش خود نظارتی از قبل آموزش داده شده است .این بدان معناست که فقط بر روی متون خام از قبل آموزش داده شده است، بدون اینکه انسانی به هیچ وجه آنها را با یک فرآیند خودکار برای تولید ورودی ها و برچسب ها از آن متون برچسب گذاری کند (به همین دلیل است که می تواند از داده های در دسترس عمومی زیادی استفاده کند) .به‌طور دقیق‌تر، با هدف مدل‌سازی زبان ماسک‌شده (MLM) از قبل آموزش داده شد .با گرفتن یک جمله، مدل به طور تصادفی ۱۵٪ از کلمات را در ورودی ماسک می کند و سپس کل جمله ماسک شده را در مدل اجرا می کند و باید کلمات پوشانده شده را پیش بینی کند .این با شبکه‌های عصبی بازگشتی سنتی (RNN) که معمولاً کلمات را یکی پس از دیگری می‌بینند یا با مدل‌های اتورگرسیو مانند GPT که به صورت داخلی توکن‌های آینده را پنهان می‌کنند متفاوت است .این به مدل اجازه می دهد تا نمایش دو طرفه جمله را بیاموزد .به این ترتیب، مدل یک نمایش داخلی از ۱۰۰ زبان را می‌آموزد که می‌تواند برای استخراج ویژگی‌های مفید برای کارهای پایین‌دستی مورد استفاده قرار گیرد.

توضیحی بر معماری این مدل :

- لایه های Embeding:مشابه سایر مدل های ترانسفورماتور، XLM-RoBERTaبا جاسازی لایه ها شروع می شود . این لایه‌ها، نشانه‌های ورودی را به نمایش‌های برداری پیوسته، که به عنوان جاسازی کلمات شناخته می‌شوند، نگاشت می‌کنند . XLM-RoBERTaاز رمزگذاری جفت بایت (BPE) برای مدیریت واحدهای فرعی استفاده می کند که به آن امکان می دهد کلمات خارج از واژگان را مدیریت کند و اطلاعات صرفی را ضبط کند.
- رمزگذارهای ترانسفورماتور :هسته معماری XLM-RoBERTa رمزگذار ترانسفورماتور است .این شامل چندین لایه مکانیسم های خودتوجهی و شبکه های عصبی پیشخور است .هر لایه در رمزگذار توالی ورودی را به صورت موازی پردازش می کند و به مدل اجازه می دهد وابستگی های محلی و جهانی را بگیرد.
- مکانیسم توجه به خود مدل را قادر می‌سازد تا ضمن رمزگذاری اطلاعات متنی، به بخش‌های مختلف توالی ورودی توجه کند . وزن توجه را برای هر نشانه ورودی محاسبه می کند و به مدل اجازه می دهد در طول رمزگذاری بر روی اطلاعات مربوطه تمرکز کند .
- شبکه‌های عصبی پیش‌خور در هر لایه ترانسفورماتور به ثبت روابط پیچیده و غیرخطی‌ها در توالی ورودی کمک می‌کنند .
- ساختار پایین دستی :خروجی رمزگذار ترانسفورماتور از یک ساختار پایین دستی عبور می کند که بسته به کار خاصی که مدل برای آن آموزش دیده می تواند متفاوت باشد .این ساختار معمولاً از لایه‌های اضافی (مثلاً لایه‌های کاملاً متصل) تشکیل شده است که نمایش‌های کدگذاری‌شده را به خروجی‌های خاص تبدیل می‌کنند .به طور کلی، معماری XLM-RoBERTa برای یادگیری بازنمایی جملات قدرتمندی طراحی شده است که می تواند اطلاعات معنایی و نحوی را در زبان های مختلف ضبط کند XLM-RoBERTa .با آموزش بر روی حجم زیادی از داده‌های چندزبانه، می‌تواند از اطلاعات مشترک بین زبان‌ها برای بهبود عملکرد در وظایف مختلف بین زبانی استفاده کند.

XLM-RoBERTaچندین پیشرفت را نسبت به XLM معرفی می کند که به بهبود عملکرد آن کمک می کند:

- مجموعه بزرگتر پیش از تمرین XLM-RoBERTa :از یک مجموعه بزرگتر قبل از تمرین، مشابه RoBERTa استفاده می کند .این به مدل کمک می‌کند تا بازنمایی‌های قوی‌تر و کلی‌تری را از داده‌های متنوع و گسترده زبان بیاموزد.
- مراحل آموزشی بیشتر XLM-RoBERTa :نسبت به XLM مراحل قبل از آموزش بیشتری را پشت سر می گذارد . افزایش تعداد مراحل به مدل اجازه می دهد تا بهتر همگرا شود و تفاوت های ظریف زبان را به تصویر بکشد.
- پیش بینی جمله بعدی حذف شد RoBERTa :(NSP) و XLM-RoBERTa وظیفه پیش‌بینی جمله بعدی (NSP)مورد استفاده در BERT اصلی را حذف می‌کنند .در عوض، آنها تنها بر هدف مدل‌سازی زبان پوشانده (MLM) تکیه می‌کنند که مؤثرتر است .مقایسه عملکرد بین XLM-RoBERTa و سایر مدل‌های زبان چندزبانه نشان داده است که XLM-RoBERTa به نتایج پیشرفته‌ای در کارهای مختلف NLP چند زبانه دست می‌یابد .معماری پیشرفته و تنظیم دقیق مجموعه داده‌های ویژه وظیفه، آن را قادر می‌سازد تا از پیشینیان خود، از جمله XLM و سایر مدل‌های رقابتی در درک زبان چندزبانه و انتقال یادگیری، بهتر عمل کند.



شکل ۴ : نمایی از معماری XLM-Roberta

۴ اقدامات انجام شده

- توضیحات درباره ی مقاله :

در این مقاله ابتدا به اهمیت موضوع تحلیل احساسات پرداخته شد سپس درباره ی مجموعه داده های موجود در فارسی توضیح کوتاهی داده شد.درباره ی اینکه یک متن می تواند دارای احساسات ترکیبی باشد توضیح داده شد و اشاره شد که به همین منظور از طبقه بندی چند برچسبی استفاده می شود تا بیش از یک از یک احساس در جمله است.

درباره ی رویکرد های متفاوت تشخیص احساسات از جمله "rule-base " که بر طبق قواعد زبانی است و روش های یادگیری ماشین که با توجه به برچسب های اموزش دیده میتوانند متن بدون برچسب را طبقه بندی کنند که به طور موثر و سریع ویژگی های جدید را از متن یاد میگرند . الگوریتم "svm " یک الگوریتم خوب در این حوزه است. همچنین شبکه های عصبی زیادی مانند LSTM و GRU نتایج پیشرفته تری داشته اند. در رویکردهای ترکیبی، رویکردهای مبتنی بر قانون و رویکردهای مبتنی بر ML در راه حل جدیدی ادغام می‌شوند که نقاط قوت هر دو رویکرد را دارد و در عین حال ضعف‌های مرتبط با آنها را کاهش می‌دهد .ایده این است که می توان با پیاده سازی مجموعه ای از طبقه بندی کننده ها و افزودن اطلاعات زبانی غنی از دانش از فرهنگ لغت، نتایج دقیق تری در تشخیص احساسات به دست آورد. در بخشی به کارهای گذشته انجام شده روی مجموعه دادگان فارسی پرداخت در ادامه به نحوه ی جمع کردن داده و منابع مختلف استفاده شده و همچنین لیبل گذاری این داده ها به روش های مختلف گفته شد همچنین گفته شد که داده هایی در طی این فرایند ها حذف یا چگونه به مجموعه دادگان اموزشی اضافه شده اند تا یک مجموعه داده خوب داشته باشیم.مروری بر امار داده های موجود انجام شد که در بخش های پیش به ان پرداختیم و سپس انواع مدل ها در ان بررسی و با هم مقایسه شدند که در بخش انتخاب مدل توضیح دادیم. سپس درباره ی پیش پردازش داده های دیتاست توضیح داد و سپس به نحوه ی تنظیم کردن ابرپارامتر ها پرداخت.هایپر پارامتر هایی نظیر تعداد دسته ها و نرخ یادگیری وجود دارد برای مثال گفته شد با توجه به تنظیم دقیق در مدل های دیگر نرخ یادگیری از یک عدد مشخص شروع و ان را به صفر می رسانیم.و یا از یک کتابخانه برای تنظیم ان ها استفاده می کنیم. سپس یه بررسی نتایج مختلف با توجه به سه معیار precision و F1 و recall پرداخته شد. و تحلیلی دستی بر روی متن هایی که اشتباه برچسب گذاری شده بودند کردیم تا بدانیم چرا مدل در تشخیص اشتباه کرده سپس عملکرد دو مجموعه داده را مقایسه کردیم و برتری ان را بر روی داده های اموزشی مورد بررسی قرار داده شد و در نهایت نتیجه گیری صورت گرفت.

یک مجموعه داده مشروح دستی مناسب برای آموزش الگوریتم‌های یادگیری عمیق ارائه شد .برای نشان دادن کیفیت بالای مجموعه داده ، چندین مدل پایه قوی از جمله مدل‌های زبانی پیشرفته ایجاد و از طریق آزمایش‌های یادگیری انتقال، ما تعمیم‌پذیری برتر مجموعه داده پیشنهادی خود را نشان داده شد و مجموعه داده نهایی، شامل بیش از ۷۰۰۰ نمونه، برای استفاده غیرتجاری در دسترس عموم است.

- منابع دیگر:

ابتدا مراجعی درباره ی تحلیل احساسات به زبان انگلیسی خواندم که دید خوبی به من داد.و منابعی درباره ی تنظیم کردن هایپر پارامتر ها خواندم که در رسیدن به دقت خوب بسیار کمک کننده بود. کد های متفاوتی در زمینه تحلیل احساسات به زبان انگلیسی در گیت هاب دیدم که متوجه روند کد شدم. سپس با کتابخانه hazm برای پیش پردازش داده ها اشنا شدم و درباره ی ان مطالعه کردم و یک نمونه پیاده سازی تحلیل احساسات به زبان فارسی دیدم که البته در ساختار مدل و خیلی موارد دیگر با پروژه کنونی تفاوت داشت.
یک مقاله هم درباره ی یکی از ابزار های فارسی خواندم اما بعدا تصمیم گرفتم که از ان استفاده نکنم که توضیحی کوتاه درباره مقاله بدین شرح است :" مجموعه ابزارهای پردازش زبان فارسی مجموعه ای جامع از ابزارهای پردازش زبان فارسی است که بسیاری از کاربردهای زبانی محاسباتی را ارائه می دهد .این سیستم می تواند تمامی وظایف اساسی مورد نیاز برای لایه های مختلف پردازش زبان فارسی را از لایه اولیه خود که لایه واژگانی تا لایه بالایی که عبارتند از نحو و معنایی است، پردازش و پیش ببرد .مجموعه ابزار پارسی‌پرداز ترکیبی از نرمال‌سازی، نشانه‌گذاری، غلط‌گیر املا، بخشی از برچسب‌گذاری گفتار، تحلیل مورفولوژیکی شامل واژه‌سازی و ریشه‌یابی، تجزیه‌کننده وابستگی فارسی و در نهایت برچسب‌گذاری نقش معنایی (SRL) را انجام می‌دهد .نتایج کارایی و دقت بالایی را نشان می دهد."

- شرح کار های صورت گرفته :

بعد از خواندن دیتاست و پیش پردازش روی ان به مرحله بعد رفتیم و پیش پردازش را روی داده های تست و اموزشی اعمال می کنیم.و بعد انکد کردن برچسب ها را به صورت دستی انجام دادیم و روی داده های تست و اموزشی اعمال می کنیم. و بعد مدل را با استفاده از یادگیری انتقالی fine tune می کنیم. سپس بردار pooler_output را با استفاده از لایه dropout به یک لایه حذف تصادفی میدهیم. خروجی این لایه یک بردار ۱۰۲۴ بعدی به نام d_out است که برخی از مولفه های آن با احتمال ۰,۱ صفر میشوند.
در نهایت بردار d_out را با استفاده از لایه output_layer به یک لایه تمامن متصل میدهیم. خروجی این لایه یک بردار ۷ بعدی به نام out است که هر مولفه ی آن نشاندهنده ی احتمال تعلق متن به یکی از دسته های احساسی است.
یک متغیر به نام tokenizer که از متد AutoTokenizer.from_pretrained با پارامتر xlm-roberta-large و model_max_length=64 استفاده میکند. این متغیر یک توکن ساز است که میتواند متن را به توکن هایی تقسیم کند که با مدل xlm-roberta-large سازگار هستند. پارامتر model_max_length=64 نشان میدهد که حداکثر طول متن ورودی ۶۴ توکن است. و بعد مدل را کامپایل می کنیم و داخل یک متغیر history ذخیره می کنیم تا بتوانیم نمودار را ریم کنیم. و شبکه را اموزش می دهیم چون به دقت مطلوب رسیدیم کار را ادامه می دهیم و بعد از لود کردن دیتا تست و برچسب های ان پیش بینی را انجام می دهیم سپس پیش بینی را به کلاس های برچسب تبدیل می کنیم و ماتریس در هم ریختگی و معیار های دیگر را بدست می اوریم و نمودار های زیان و دقت بر روی داده ای اموزش و تست را بررسی می کنیم. و دو تابع را برای ازمایش دستی می نویسیم یکی پیش بینی یک جمله و چند جمله که ان را دستی در قسمت بعدی تست می کنیم و میبینیم برخی متن ها و جمله ها درست تشخیص داده شده اند و برخی نه که به تحلیل چرایی ان در ادامه می پردازیم.

```python
id_to_label = ["HAPPY", "ANGRY", "HATE", "SURPRISE", "FEAR", "SAD", "OTHER"]

def predict(comment):
    token = tokenizer(list([comment]), return_tensors="np", padding=True, truncation=True)

    # Make predictions on the test data
    predictions = model.predict(dict(token))

    # Convert predictions to class labels
    predicted_labels = tf.argmax(predictions, axis=1).numpy()

    return id_to_label[predicted_labels[0]]

def batch_predict(comment):
    token = tokenizer(list(comment), return_tensors="np", padding=True, truncation=True)

    # Make predictions on the test data
    predictions = model.predict(dict(token))

    # Convert predictions to class labels
    predicted_labels = tf.argmax(predictions, axis=1).numpy()

    return [id_to_label[pl] for pl in predicted_labels]
```

نمایی از توابع پیاده سازی که یک جمله یا متن را دریافت می کنند

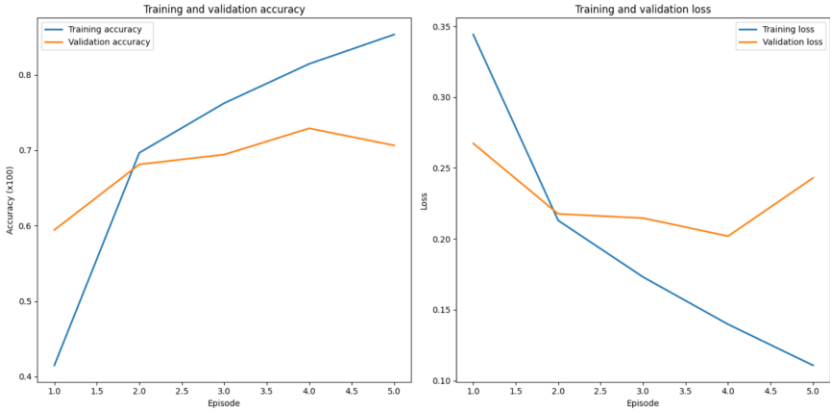
۵ ارزیابی مدل

- نمودار زیان و دقت :

مطابق آنچه در فایل توضیحات پروژه نیز گفته شد، مدل را با معیارهای ارزیابی گفته شده و معیارهای دیگری که به نظرتان مناسب است ارزیابی کنید. حتما از نمودار هایی که فکر می کنید برای این قسمت مناسب استفاده کنید. نتایج مدل بر روی ۵ نمونه دلخواه را در این قسمت بیاورید. تحلیل مناسبی برای نمونه هایی که مدل در تشخیص آن ها اشتباه کرده است ارائه دهید.

ابتدا نمودار های دقت و زیان را روی داده اموزش و ازمایش نشان دادم. در ۵ ایپاک طی شده در نمودار دقت برای داده ی اموزش همواره صعودی بوده که این نشان می دهد مدل داده ی اموزش را به خوبی یاد گرفته و مدل خوبی با عملکرد مناسبی ارائه شده است. ولی داده ی تست از یک جایی به بعد داشته overfit میشده و خاصیت تعمیم پذیری خود را از دست میداده ولی در نقاطی پیک زده و میانگین به طوری بوده که بیش برازش در حدی نبوده است که عملکرد مدل را مختل کند. و با روش های generalization می توان این تعمیم را بهتر هم کرد.

در تابع زیان هم در داده ی اموزشی همواره نزولی بوده که عملکرد خوب مدل است و در داده های ازمایشی ابتدا کم شده و بعد تقریبا ثابت شده که در نهایت با افزایش ان تعداد ایپاک ها به پایان رسیده اند و با توجه به حجم دیتا میزان زیان ما هم خوب است.



شکل ۵ : نمایی از نمودار های دقت و زیان برای داده های اموزش و ازمایش

- تست مدل آموزش دیده با جمله های ورودی :

همانطور که در مقاله گفته شد ما به‌صورت دستی جملات نام‌گذاری‌شده را بررسی کردیم تا موقعیت‌هایی را که بهترین مدل در آن‌ها عملکرد ضعیفی دارد، بهتر تحلیل کنیم .با مرور برخی از این مثال‌ها، آنچه می‌توان استنباط کرد این است که هر زمان که جمله‌ای حاوی احساسات ترکیبی باشد، مدل برچسب اشتباهی را تولید می‌کند .در چنین شرایطی، اختصاص یک و تنها یک احساس دقیق به جمله ممکن است حتی برای انسان ها نیز چالش برانگیز باشد .به همین دلیل است که طبقه‌بندی‌کننده‌های چند برچسبی در برابر طبقه‌بندی‌کننده‌های چند کلاسه استفاده می‌شوند تا حضور بیش از یک احساس را در یک جمله مشخص کنند .از آنجایی که ما در این مطالعه با یک مشکل طبقه بندی چند طبقه سروکار داریم، فرض می کنیم که هر جمله فقط یک احساس را در خود دارد .بنابراین فرض بر این است که برای هر جمله فقط یک هدف واقعی وجود دارد .با این حال، این ممکن است برای همه موقعیت‌ها صادق نباشد .بر اساس جملات داده شده می توان قضاوت کرد که برخی از پیش بینی های مدل اصلاً بی ربط نیستند .در واقع، بسته به زمینه، این برچسب های پیش بینی شده ممکن است به اندازه حقایق پایه اختصاص داده شده معتبر در نظر گرفته شوند .با این حال، هنوز موقعیت‌های دیگری وجود دارد که به نظر می‌رسد عملکرد ضعیف مدل به تعصب مدل به وقوع برخی کلمات خاص یا ترکیبی از کلمات در جمله مربوط می‌شود.

برای مثال به درستی جمله اول و متن اول برچسب احساسات را دارند که با باکس سبز نشان داده ام در شکل. اما سه جمله بعدی را که از مقاله برداشتم که در مقاله هم به درستی تشخیص داده نشده بود اینجاهم به درستی تشخیص داده نشده داخل باکس قرمز اند به دلیل اینکه دارای احساسات ترکیبی و کلماتی هیتند که درک حی جمله را با ابهام مواجه می کنند.

جمله سوم خوشحالی و چهارم دیگری و پنجم تعجب از طبقه بندی درست این لغات است در جمله سوم سیگار کشیدن بسیار ابهام انگیز است برای اینکه بتواند شادی شناسایی شود اینکه امر سیگار کشیدن می تواند خوشحالی برانگیز باشد برای مدل قابل تشخیص نیست و لغات مشخص هم برای ان وجود ندارد در جمله چهارم یک جمله ی خبری در باره ی یک حقیقت گفته شده اما به دلیل وجود کلمات رقص و شادی به اشتباه شادی شنایایی شده و جمله اخر یک حرکت عجیب از سوی دسته ای ادم اتفاق افتاده اما به دلیل مرحوم جمله ناراحتی شناسایی شده .

```
predict_one = "امروز حالم خوبه"
print(predict_one)
print(predict(predict_one))

predict_in_batch = [
    "امروز حالم بده",
    "خیلی گشنمه"
    ]
print(predict_in_batch)
print(batch_predict(predict_in_batch))

predict_one = "بریم یک سیگاری بکشیم شاید حال داد"
print(predict_one)
print(predict(predict_one))

predict_one = "موسیقی و شاد بودن حرام نیست"
print(predict_one)
print(predict(predict_one))

predict_one = "زیر آخرین پست های مرحوم گفتن روحت شاد"
print(predict_one)
print(predict(predict_one))
```

```
امروز حالم خوبه
1/1 [==============================] - 0s 136ms/step
HAPPY
['امروز حالم بده' ,'خیلی گشنمه']
1/1 [==============================] - 0s 133ms/step
['HATE', 'SAD']
بریم یک سیگاری بکشیم شاید حال داد
1/1 [==============================] - 0s 145ms/step
OTHER
موسیقی و شاد بودن حرام نیست
1/1 [==============================] - 0s 132ms/step
HAPPY
زیر آخرین پست های مرحوم گفتن روحت شاد
1/1 [==============================] - 0s 76ms/step
SAD
```

شکل ۶ : نمایی از پیش بینی مدل برای جمله یا متن ورودی

- مقایسه معیار های ارزیابی :

در اینجا سه معیار ارزیابی را برای مدل اوردم و ان را با نتایج مقاله مقایسه کردم و همانطور که می بینیم از متوسط این میزان ها بیش تر است حالت ماکسیمم ما و همینطور مقادیر میانگین هم مشخص شده اند.

بالا بودن معیار recall در طبقه بندی به این معناست که الگوریتم توانایی بالایی در تشخیص تمام نمونه های مثبت واقعی دارد. به

عبارت دیگر، recall نشان میدهد که چند درصد از نمونه هایی که واقعاً متعلق به یک کلاس هستند، توسط الگوریتم به درستی به آن کلاس نسبت داده شده اند.

بالا بودن معیار F۱ score در طبقه بندی یک مدل عمیق به این معناست که مدل توانایی بالایی در همزمان داشتن دقت و بازیابی بالا دارد. دقت نشان میدهد که چند درصد از پیشبینی های مثبت مدل، درست هستند و بازیابی نشان میدهد که چند درصد از نمونه های مثبت واقعی، توسط مدل شناسایی شدهاند. معیار F۱ score میانگین هارمونیک دقت و بازیابی است و بین صفر تا یک متغیر است. هر چه مقدار این معیار نزدیک به یک باشد، نشان میدهد که مدل عمیق عملکرد بهتری دارد.

بالا بودن معیار percision در طبقه بندی یک مدل عمیق به این معناست که مدل توانایی بالایی در انتخاب نمونه های مثبت واقعی دارد. به عبارت دیگر، percision نشان میدهد که چند درصد از نمونههایی که مدل به عنوان مثبت پیشبینی کرده است، واقعاً مثبت هستند.

با توجه به این معیار های ارزیابی متوجه می شویم به چه میزان طبقه بندی های ما درست بوده است و چقدر عملکرد مدل قابل قبول است. به طور میانگین اندازه recall=71 , precision=71 , F1=70 بوده است.و بالاترین precision=90 برای happy و بالاترین recall=84برای fear و بالاترین F1=90 برای happy است ولی در مقاله بهترین عملکرد را در مورد احساساتی مانند شادی، ترس و غم ارائه می دهد .از طرف دیگر، کمترین امتیاز F1 را در مورد احساساتی مانند خشم، دیگر و نفرت کسب می کند.

Classification Report:

	precision	recall	f1-score	support
0	0.90	0.71	0.79	275
1	0.75	0.51	0.61	154
2	0.67	0.69	0.68	65
3	0.65	0.74	0.70	145
4	0.69	0.84	0.76	57
5	0.75	0.74	0.75	262
6	0.53	0.74	0.62	193
accuracy			0.71	1151
macro avg	0.71	0.71	0.70	1151
weighted avg	0.73	0.71	0.71	1151

Table 3: Evaluation Metrics Resulted from The Best Model (XLM-RoBERTa-large)

Emotion	Precision	Recall	F1	Support (No. of Test Examples)
Anger	74.62	62.99	68.31	154
Fear	78.69	84.21	81.36	57
Happiness	86.02	87.27	86.64	275
Hatred	63.64	75.38	69.01	65
Other	60.98	77.72	68.34	193
Sadness	82.45	77.10	79.68	262
Wonder	84.96	66.21	74.42	145
Macro Average	75.91	75.84	75.39	1151

شکل ۷ : نمایی از پیش بینی مدل برای جمله یا متن ورودی

- ماتریس در هم ریختگی :

ماتریس در هم ریختگی یک ماتریس است که عملکرد یک الگوریتم طبقه بندی را نشان میدهد. این ماتریس شامل تعداد نمونه هایی است که به درستی یا به اشتباه به یک کلاس پیشبینی شدهاند. ماتریس در هم ریختگی میتواند برای محاسبه معیارهای ارزیابی مختلف مانند دقت، بازیابی، معیار F۱ و غیره استفاده شود.

در اینجا ماتریس در هم ریختگی که در پروژه پیاده سازی شده با مقاله اورده شده و با توجه به ماتریس هم مدل عملکرد مناسبی دارد و هر متن را درست طبق احساساتش طبقه بندی کرده است. و در طبقه ی درست خود قرار گرفته است هر مورد.

```
Confusion Matrix:
[[195   3   1  12   4   7  53]
 [  3  79   2  26   6  15  23]
 [  0   4  45   3   4   5   4]
 [  1   6   0 108   1  12  17]
 [  1   1   0   1  48   4   2]
 [ 10   5  15   6   5 195  26]
 [  6   8   4   9   2  21 143]]
```

شکل۸: نمایی از پیش بینی مدل برای جمله یا متن ورودی

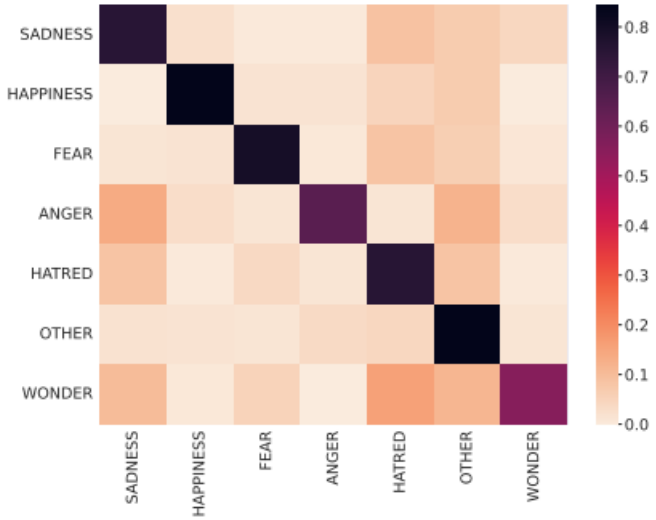


Figure 3: Confusion Matrix for XLM-RoBERTa-large Predictions

۷ مراجع

پیش پردازش داده یا Data Preprocessing چیست؟(cafetadris.com) | .

(tensorflow.org)

FacebookAI/xlm-roberta-large · Hugging Face

XLM and XLM-RoBERTa - Scaler Topics

Semantic Analysis, Explained (monkeylearn.com)

Parameters, Hyperparameters, Machine Learning | Towards Data Science

hazm · PyPI

semantic-analysis · GitHub Topics

Tutorial.ipynb - Colaboratory (google.com)

ParsiPardaz: Persian Language Processing Toolkit | IEEE Conference Publication | IEEE Xplore