به نام خدا

دانشگاه تهران

دانشکده مهندسی

برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق

تمرین پنجم

نام و نام خانوادگی	بهراد موسایی شیرمحمد - محمد جواد رنجبر کلهرودی
شماره دانشجویی	۸۱۰۱۰۱۲۷۸ - ۸۱۰۱۰۱۱۷۳
تاریخ ارسال گزارش	۱۴۰۲.۱۰.۱۵

فهرست

فهرست جداول

پاسخ ۱. تشخیص احساسات گفتار (SER)

۱-۱. معرفی مدل HuBERT

مدل HuBERT یک مدل یادگیری ماشینی است که برای پردازش زبان طبیعی و تشخیص گفتار طراحی شده است. این مدل از رویکردی مبتنی بر ترنسفورمر استفاده می‌کند، که در مدل‌های معروفی مانند BERT و GPT به کار رفته است. HuBERT به طور خاص برای بهبود درک ماشین از گفتار انسانی طراحی شده و از روش‌هایی برای یادگیری نمایش‌های دقیق‌تر گفتار استفاده می‌کند.

ویژگی‌های کلیدی و نوآوری‌های HuBERT عبارتند از:

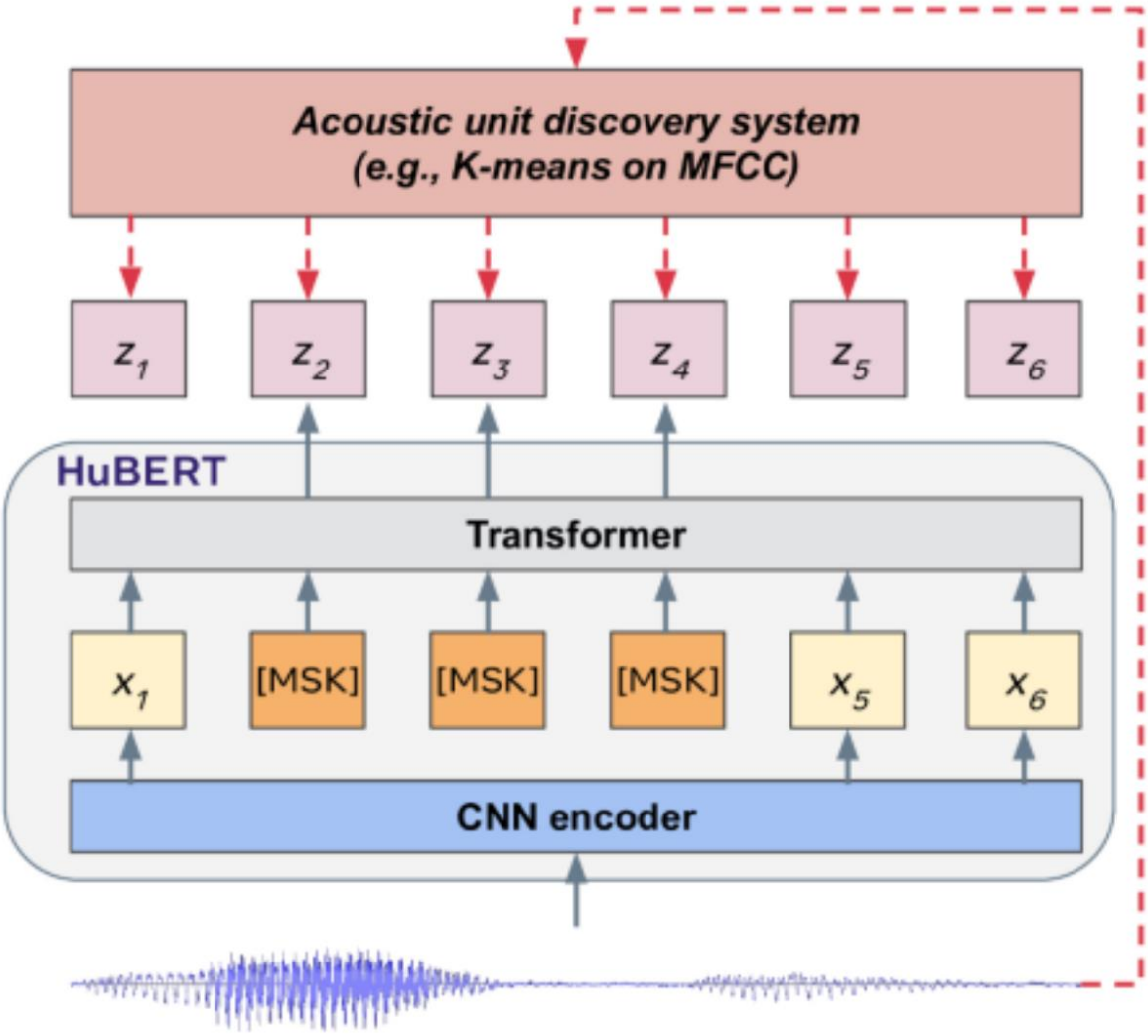
۱. یادگیری نظارت نشده: HuBERT می‌تواند از داده‌های گفتاری بدون برچسب یاد بگیرد، که این امر باعث می‌شود تا در مقیاس بزرگتر و با داده‌های کمتر محدودکننده قابل آموزش باشد.

۲. فرایند یادگیری دو مرحله‌ای: در مرحله اول، مدل یک نمایش موقت از داده‌های گفتاری ایجاد می‌کند. سپس در مرحله دوم، این نمایش‌ها را برای تولید نمایش‌های بیشتر توسعه داده و آنها را تصحیح می‌کند.

۳. تطابق پذیری با زبان‌های مختلف: HuBERT به گونه‌ای طراحی شده است که می‌تواند با زبان‌های مختلف سازگار شود، که این امر اهمیت زیادی در پردازش زبان طبیعی و تشخیص گفتار دارد.

۴. کاربردهای متنوع: این مدل می‌تواند در کاربردهای مختلف مانند تشخیص گفتار، تولید گفتار، ترجمه گفتار به متن و برعکس، و تحلیل احساسات به کار رود.

HuBERT نشان دهنده پیشرفت قابل توجهی در حوزه یادگیری عمیق و پردازش گفتار است و امکانات جدیدی را برای تحقیقات و کاربردهای عملی در این زمینه فراهم می‌کند.



شکل ۱: شمای کلی از عملکرد **HuBERT**

۱-۲. سوالات تشریحی

۱-۲-۱. چالش های داده های صوتی در یادگیری

داده‌های صوتی به دلیل ماهیت پیچیده و چندوجهی آن‌ها، چالش‌های خاصی را در حوزه یادگیری ماشین و یادگیری عمیق ایجاد می‌کنند. این چالش‌ها شامل موارد زیر هستند:

۱. تنوع بزرگ در داده‌ها: داده‌های صوتی می‌توانند بسیار متفاوت باشند، از جمله لهجه‌ها، سرعت گفتار، تن صدا، و ویژگی‌های فردی مانند سن و جنسیت. این تنوع بزرگ باعث می‌شود که مدل‌های یادگیری ماشین باید به خوبی تعمیم یابند تا بتوانند به صورت موثر کار کنند.

۲. پس‌زمینه و نویز محیطی: داده‌های صوتی اغلب شامل نویزهای محیطی هستند که می‌تواند تشخیص و تحلیل صوت را دشوار کند. این نویزها می‌توانند شامل صداهای پس‌زمینه، تداخلات الکترونیکی، یا حتی اکو باشند.

۳. نیاز به داده‌های زیاد برای آموزش: مدل‌های پیشرفته یادگیری ماشین برای دستیابی به دقت بالا نیاز به حجم زیادی از داده‌های آموزشی دارند. جمع‌آوری و برچسب‌گذاری این داده‌های صوتی می‌تواند هزینه‌بر و زمان‌بر باشد.

۴. پردازش سیگنال‌های زمانی: داده‌های صوتی به صورت سیگنال‌های زمانی هستند که نیازمند روش‌های پردازش خاصی هستند. تبدیل این سیگنال‌ها به فرمی که توسط مدل‌های یادگیری ماشین قابل استفاده باشد، یک چالش است.

۵. تشخیص و فهم گفتار انسانی: تشخیص گفتار انسانی شامل درک معنای کلمات و جملات است که می‌تواند بسیار پیچیده باشد، به ویژه در موقعیت‌هایی که گفتار طبیعی و غیررسمی است.

۶. تغییرات سریع در زمینه‌های کاربردی: زمینه‌های کاربردی داده‌های صوتی به سرعت در حال تغییر هستند، و مدل‌های یادگیری ماشین باید قادر به سازگاری با این تغییرات باشند.

۷. مسائل مربوط به حریم خصوصی و اخلاقی: جمع‌آوری و استفاده از داده‌های صوتی باید با در نظر گرفتن مسائل حریم خصوصی و اخلاقی انجام شود، به ویژه وقتی که داده‌ها شامل اطلاعات شخصی هستند.

برای مقابله با این چالش‌ها، پژوهش‌ها و توسعه‌های مداوم در حوزه‌های مانند تکنیک‌های پیش‌پردازش، روش‌های یادگیری نظارت نشده، و بهینه‌سازی مدل‌ها برای کاربردهای مختلف انجام می‌شود.

۱-۲-۲. رویکرد HuBERT

مدل HuBERT (واحد پنهان BERT)، همانطور که در مقاله ارائه شده، معرفی شده است، یک رویکرد خود نظارتی برای یادگیری بازنمایی گفتار است. این به چالش‌های خاصی در این زمینه می‌پردازد، مانند برخورد با واحدهای صوتی متعدد در هر گفتار ورودی، عدم وجود واژه‌نامه واحدهای صدای ورودی در مرحله قبل از آموزش، و طول متغیر واحدهای صدا بدون تقسیم‌بندی صریح است.

مؤلفه‌های کلیدی و رویکرد HuBERT عبارتند از:

۱. یادگیری واحدهای پنهان برای HuBERT: از مدل هایی مانند k-means و مدل های مخلوط گاوسی (GMMs) برای استنتاج واحدهای پنهانی که با واحدهای صوتی زیرین مرتبط هستند، استفاده می کند. این روش با یادگیری نیمه نظارت شده در تضاد است، جایی که یک مدل آکوستیک آموزش‌دیده بر روی جفت‌های متن و گفتار، برچسب‌های شبه آوایی را ارائه می‌کند.

۲. یادگیری بازنمایی از طریق پیش‌بینی ماسک: این روش شامل پوشاندن بخش‌های خاصی از توالی گفتار ورودی و پیش‌بینی این مناطق پوشانده شده بر اساس زمینه بدون ماسک است. این رویکرد، که در مدل‌هایی مانند SpanBERT و wav۲vec ۲.۰ نیز استفاده می‌شود، به مدل کمک می‌کند تا مدل‌های آکوستیک و زبان را از ورودی‌های پیوسته یاد بگیرد.

۳. استفاده از گروه‌های خوشه‌ای: برای بهبود کیفیت اهداف مدل، HuBERT از چندین مدل خوشه‌بندی استفاده می‌کند. این رویکرد گروهی به ایجاد اهداف دانه‌دارتر کمک می‌کند و یادگیری بازنمایی بهتر را تسهیل می‌کند.

۴. پالایش تکراری تکالیف خوشه ای : مدل تکالیف خوشه ای را در طول فرآیند یادگیری اصلاح می کند. از یک مدل از پیش آموزش‌دیده برای ارائه بازنمایی‌های بهبود یافته نسبت به ویژگی‌های آکوستیک خام استفاده می‌کند و خوشه‌های جدیدی را بر اساس این نمایش‌های تصفیه‌شده تولید می‌کند.

۵. معماری و پیاده سازی مدل: HuBERT از معماری wav۲vec ۲.۰ پیروی می کند که شامل یک رمزگذار شکل موج کانولوشن، یک رمزگذار BERT، یک لایه طرح ریزی و یک لایه جاسازی کد است. این در سه پیکربندی موجود است: BASE، LARGE، و X-LARGE، با مدل X-LARGE که حدود ۱ میلیارد پارامتر دارد.

رویکرد HuBERT، به‌ویژه استفاده از پیش‌بینی پنهان و تمرکز بر ثبات خوشه‌بندی بدون نظارت به جای کیفیت ذاتی برچسب‌های خوشه‌ای، به آن اجازه می‌دهد تا عملکرد مدل‌های پیشرفته مانند wav۲vec ۲.۰ را در موارد مختلف مطابقت دهد یا آن را بهبود بخشد.

۱-۳. معرفی مجموعه دادگان

این مجموعه داده مربوط به "پایگاه داده گفتار احساسی شریف" (ShEMO) است. این پایگاه داده به زبان فارسی است و شامل ۳۰۰۰ بیانیه نیمه‌طبیعی است که معادل ۳ ساعت و ۲۵ دقیقه داده گفتاری است و از نمایش‌های رادیویی آنلاین استخراج شده‌اند. ShEMO شامل نمونه‌های گفتاری از ۸۷ گوینده بومی فارسی‌زبان برای پنج احساس اصلی شامل خشم، ترس، خوشحالی، غم و تعجب، همچنین حالت بی‌طرف است. دوازده نفر از مشخص‌کنندگان، حالت احساسی زیربنایی بیانیه‌ها را برچسب‌گذاری کرده‌اند و از رای‌گیری اکثریت برای تصمیم‌گیری در مورد برچسب‌های نهایی استفاده شده است. بر اساس اندازه‌گیری کاپا، توافق بین مشخص‌کنندگان ۶۴٪ است که به عنوان "توافق قابل توجه" تفسیر می‌شود.

این پایگاه داده برای اهداف آکادمیک به صورت رایگان در دسترس است تا به عنوان یک مبنایی برای تحقیقات بیشتر در مورد گفتار احساسی فارسی عمل کند. این پایگاه داده برای تشخیص احساسات در گفتار فارسی ارزیابی شده است. در تجربیات، دستگاه بردار پشتیبانی (SVM) بهترین نتایج را برای هر دو مدل بدون توجه به جنسیت (۵۸.۲٪) و وابسته به جنسیت (زنان=۵۹.۴٪، مردان=۵۷.۶٪) به دست آورده است.

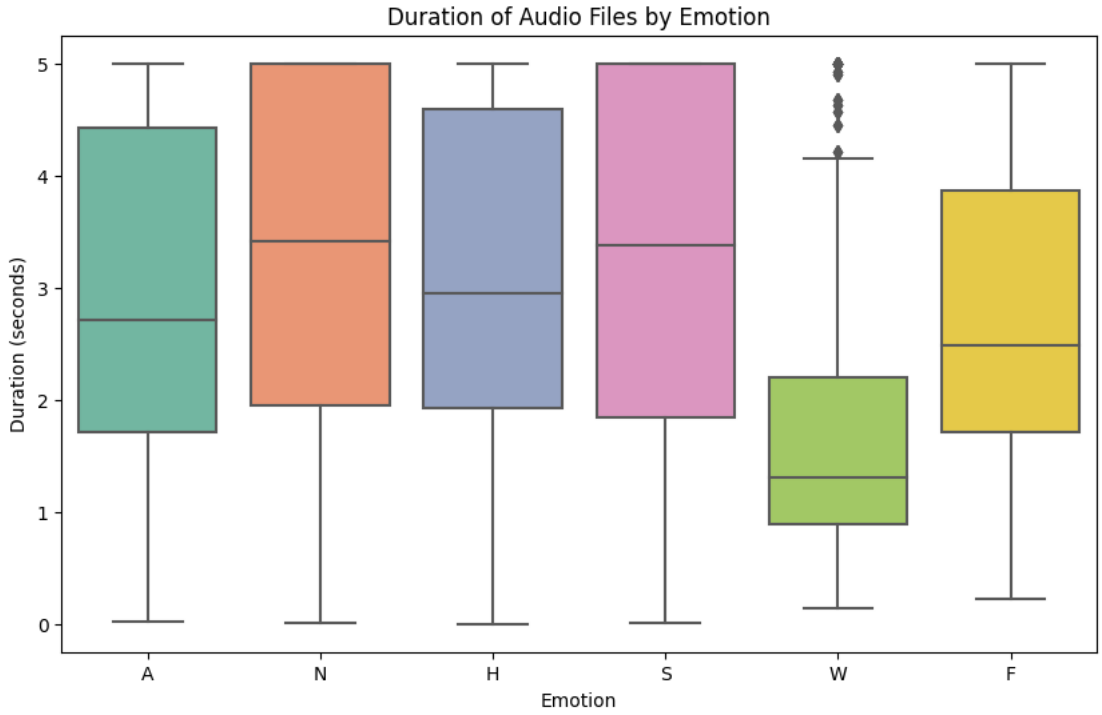
مجموعه داده‌های شما شامل سه فایل zip است: یکی برای بیانیه‌های زنان، یکی برای بیانیه‌های مردان و یکی برای متن‌ها. این داده‌ها می‌توانند برای تحلیل‌های مرتبط با تشخیص احساسات از گفتار به کار روند.

```
     gender speaker_code emotion utterance_number
0         M           05       N           01.wav
1         M           46       A           06.wav
2         M           47       N           13.wav
3         M           03       A           24.wav
4         M           53       N           23.wav
...     ...          ...     ...              ...
2995      F           07       A           32.wav
2996      F           20       W           05.wav
2997      F           17       S           10.wav
2998      F           07       W           05.wav
2999      F           29       N           07.wav

                                    path
0       /content/dataset/male/M05N01.wav
1       /content/dataset/male/M46A06.wav
2       /content/dataset/male/M47N13.wav
3       /content/dataset/male/M03A24.wav
4       /content/dataset/male/M53N23.wav
...                                  ...
2995  /content/dataset/female/F07A32.wav
2996  /content/dataset/female/F20W05.wav
2997  /content/dataset/female/F17S10.wav
2998  /content/dataset/female/F07W05.wav
2999  /content/dataset/female/F29N07.wav

[3000 rows x 5 columns]
```

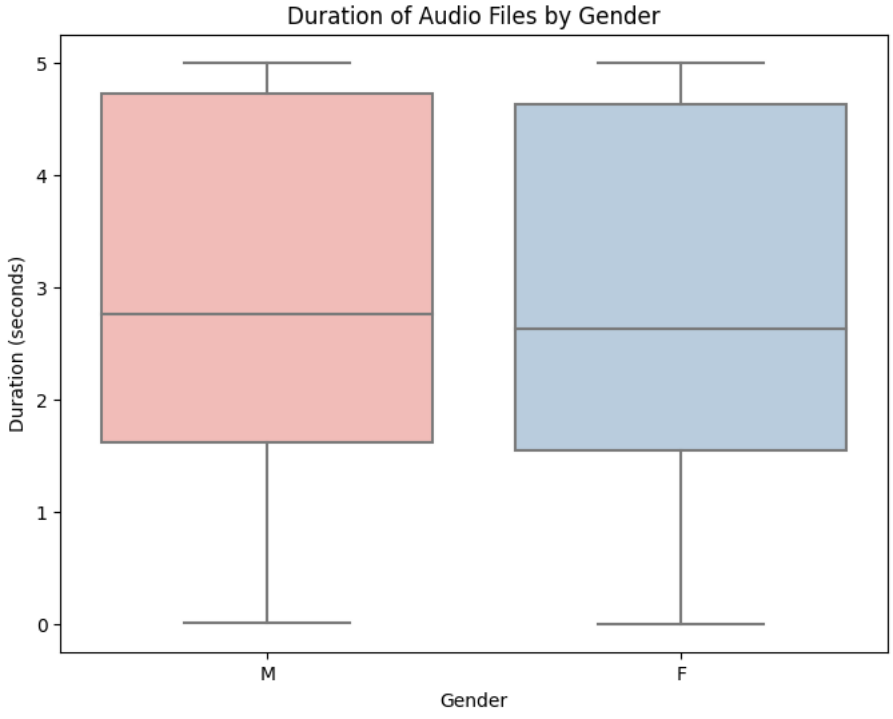
شکل ۲: نمایی از دیتاست

۱-۳-۱. پیش پردازش داده ها

ابتدا اطلاعات آماری دیتاست را به نمایش می گذاریم:



شکل ۳: دسته بندی داده بر اساس زمان در هر کلاس



شکل ۴: دسته بندی داده بر اساس جنسیت در هر کلاس

همان طور که مشاهده می شود داده ها را باتوجه به شرایط گفته شده به آموزش و تست و اعتبارسنجی تقسیم می کنیم.

داده‌ها در ۶ مختلف می‌باشند که توزیع این داده‌ها در این کلاس‌ها یکسان نیست. زمانی که توزیع کلاس‌ها در مجموعه داده شما یکنواخت نیست، به این معنا که برخی از کلاس‌ها تعداد قابل توجهی از نمونه‌ها را دارند و برخی دیگر کمتر، می‌تواند چالش‌هایی برای مدل‌های یادگیری ماشین ایجاد کند. رفع نابرابری توزیع کلاس‌ها اهمیت دارد تا مدل به سمت کلاس اکثریت تعیین شده و احتمالاً کلاس‌های اقلیت را نادیده بگیرد. در زیر چند راهکار برای مواجهه با این مسئله آورده شده است:

نمونه‌گیری داده:

کاهش نمونه (Undersampling): حذف نمونه‌ها از کلاس اکثریت برای تعادل توزیع کلاس‌ها. باید با احتیاط انجام شود تا اطلاعات زیادی از دست نرود.

افزایش نمونه (Oversampling): تکرار نمونه‌ها از کلاس اقلیت یا تولید نمونه‌های مصنوعی برای تعادل توزیع کلاس‌ها. تکنیک‌هایی مانند SMOTE (Synthetic Minority Over-sampling Technique) ممکن است مفید باشند.

تقویت داده:

افزایش اندازه کلاس اقلیت با افزایش داده‌های موجود. برای داده‌های تصویری، این ممکن است شامل چرخش، برگرداندن، زوم و غیره باشد.

وزن‌دهی به loss:

تنظیم تابع خطا برای اهمیت بیشتر به اشتباهات در کلاس اقلیت. این می‌تواند با اختصاص وزن‌های مختلف به کلاس‌های مختلف در محاسبه خطا انجام شود.

روش‌های انسمبل:

استفاده از روش‌های انسمبل مانند bagging یا boosting با تکنیک‌های نمونه‌گیری. این شامل آموزش چند مدل و ترکیب پیش‌بینی‌های آن‌هاست.

یادگیری حساس به هزینه:

تنظیم هایپرپارامترهای الگوریتم برای حساسیت بیشتر به اشتباهات در کلاس اقلیت. برخی از الگوریتم‌های یادگیری ماشین گزینه‌هایی برای تنظیم وزن کلاس‌ها فراهم می‌کنند.

ارزیابی معیارهای عملکرد:

به جای اعتماد به دقت (accuracy) به تنهایی، استفاده از معیارهای ارزیابی مانند دقت (precision)، بازخوانی (recall)، اسکور F۱ و مساحت زیر منحنی (AUC-ROC) ROC.

ترکیب کلاس‌ها:

اگر برای مسئله شما مناسب باشد، در نظر بگیرید ترکیب کلاس‌های مشابه برای کاهش تعداد کلاس‌ها و تعادل توزیع.

یادگیری انتقالی:

استفاده از مدل‌های پیش‌آموزش دیده و تنظیم آن‌ها بر روی مجموعه داده نابرابر. یادگیری انتقالی به شما کمک می‌کند تا از دانش موجود در یک مجموعه داده با توزیع کلاس متعادل بهره‌مند شوید.

در این تمرین ما هم از وزن‌دهی به کلاس‌های مختلف و هم از oversampling استفاده کردیم. برای اطمینان از اینکه توزیع احساسات در هر بخش مشابه توزیع کلی است، از تابع `train_test_split` با پارامتر `stratify` استفاده می‌شود.

۱-۳-۲. ساخت دیتالودر

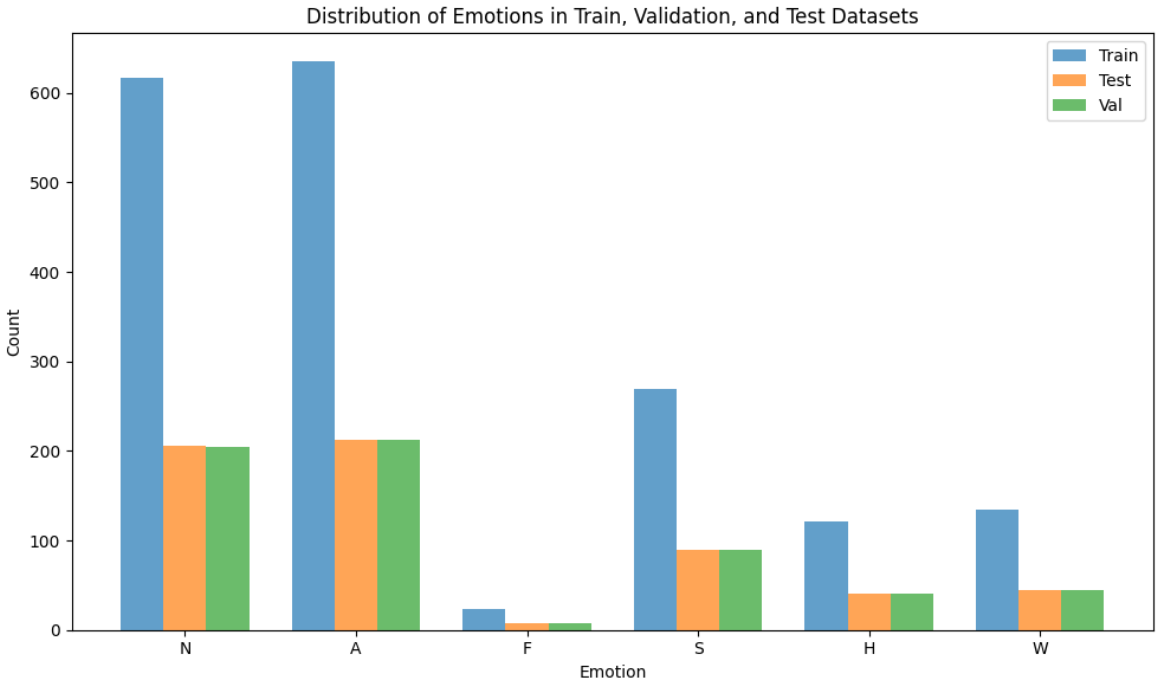
در این بخش برای بارگذاری و تقسیم داده‌های مربوط به احساسات به سه بخش: آموزش (train), آزمایش (test), و اعتبارسنجی (validation) استفاده می‌شود. این کد در چند مرحله عمل می‌کند:

۱. رمزگذاری برچسب‌ها: ابتدا، با استفاده از `LabelEncoder` از کتابخانه `sklearn.preprocessing`, برچسب‌های متنی احساسات در ستون emotion از DataFrame ``df`` به اعداد صحیح تبدیل می‌شوند. این کار برای استفاده آسان‌تر از برچسب‌ها در مدل‌های یادگیری ماشینی انجام می‌شود.

۲. تقسیم داده‌ها: داده‌ها به سه بخش تقسیم می‌شوند:

- آموزش (٪۸۰ داده‌ها)
- آزمایش (٪۱۰ داده‌ها)
- اعتبارسنجی (٪۱۰ داده‌ها)

برای اطمینان از اینکه توزیع احساسات در هر بخش مشابه توزیع کلی است، از تابع `train_test_split` با پارامتر `stratify` استفاده می‌شود.



شکل ۵: حفظ توزیع کلاس‌ها

حال در ساخت dataloader ما همه‌ی دنباله‌ها را به یک سایز یکسان max_length تبدیل کرده‌ایم این روش نسبت به اینکه تمام اعضای یک batch به اندازه‌ی بلندترین عضو آن کنیم مزایا و معایبی دارد.

مزایا:

مصرف حافظه کمتر: با استفاده از پدینگ، حافظه مصرفی برای نگهداری داده‌های با طول‌های مختلف کاهش می‌یابد.

عملکرد بهتر در GPU: دیتالودرهایی که از پدینگ استفاده می‌کنند، بهترین عملکرد را در GPU دارند زیرا محاسبات بر روی داده‌های یک batch به صورت موازی انجام می‌شود.

معایب:

توجه به دنباله‌های بی‌معنی: اگر داده‌های شما در یک batch طول‌های مختلفی داشته باشند، پد کردن دنباله‌های کوتاه با صفر (یا مقدار دیگری) ممکن است به مشکلاتی در آموزش مدل منجر شود. به عبارتی داریم اطلاعات از بین میبریم.

اضافه‌شدن ابعاد پادینگ: افزودن ابعاد پادینگ به داده می‌تواند مشکلاتی را در پردازش‌های بعدی مانند attention موجب شود.

۱-۴. تعریف مسئله

۱-۴-۱. تولید بازنمایی مناسب از کل دنباله ورودی

مراحل این کار به صورت زیر می‌باشد:

۱. وارد کردن کتابخانه‌ها: ابتدا کتابخانه‌های `Wav۲Vec۲FeatureExtractor` از `transformers` و `Dataset` از `datasets` برای کار با داده‌های گفتاری وارد می‌شوند. همچنین، کتابخانه `librosa` برای پردازش فایل‌های صوتی استفاده می‌شود.

۲. تعریف تابع `map_to_array`: این تابع برای بارگذاری و تبدیل مسیرهای فایل‌های صوتی به آرایه‌های صوتی استفاده می‌شود. فایل‌های صوتی با استفاده از تابع `librosa.load` بارگذاری می‌شوند، که در اینجا نرخ نمونه‌برداری را روی ۱۶۰۰۰ هرتز تنظیم می‌کند و فایل‌ها را به حالت تک‌کاناله (mono) تبدیل می‌کند.

۳. ساخت مجموعه داده‌های آموزش، آزمایش و اعتبارسنجی: مجموعه داده‌های آموزشی (`train_df`)، آزمایشی (`test_df`) و اعتبارسنجی (`val_df`) که در قالب DataFrame پانداس هستند، به اشیاء `Dataset` تبدیل می‌شوند. سپس با استفاده از تابع `map` و تابع تعریف شده `map_to_array`، داده‌های گفتاری بارگذاری و در قالب آرایه‌های صوتی در هر نمونه از این مجموعه داده‌ها ذخیره می‌شوند.

حال بعد از tokenize صوت آن را به مدل ورودی می‌دهیم و بازنمایی آن را به صورت زیر استخراج می‌کنیم:

```
def forward(self, inputs, attention_mask=None, labels=None):
    output = self.bert(inputs)['last_hidden_state']
    output = self.bat2(output.mean(axis=1))  # for wav2vec
```

شکل ۶ بازنمایی صوت

برای ساخت بردار تعبیه (Embedding) که بازنمایی کل دنباله ورودی را ارائه می‌دهد، می‌توانید از میانگین‌گیری وزن‌دار بردارهای واژگانی تشکیل دهید. این روش به عنوان "Pooling" یا "Sentence Embedding" نیز شناخته می‌شود. در ادامه، یک روش ساده برای ساخت چنین برداری را توضیح می‌دهم:

Mean Pooling: محاسبه میانگین امبدینگ توکن‌ها در امتداد بعد دنباله. این کار نمایانگر کل دنباله است.

Max Pooling: استخراج مقدار بیشینه در امتداد بعد دنباله. این قادر به گرفتن ویژگی‌های برجسته و مهم ورودی است.

توکن CLS: برای برخی از مدل‌های ترانسفورمر، امبدینگ توکن [CLS] (معمولاً اولین توکن) به عنوان نماینده کل دنباله استفاده می‌شود.

در این پروژه ما از Mean pooling که دلایل آن به شرح زیر است:

سادگی و کم بودن محاسبات: این یک روش ساده و با محاسبات کم است که داده‌های لازم ما را حفظ می‌کند.

نمایش با اندازه ثابت: میانگین‌گیری اجازه می‌دهد یک نمایش با اندازه ثابت به‌دست آید بدون توجه به طول دنباله ورودی. این برای بسیاری از مدل‌های یادگیری ماشین که نیاز به اندازه ورودی ثابت دارند، مهم است.

کاهش بعد: میانگین‌گیری اطلاعات را از تمام توکن‌های یک دنباله جمع‌آوری می‌کند با اینکه میانگین نمایش‌های تک توکنی آن‌ها را می‌گیرد. این منجر به یک نمایش فشرده می‌شود که محتوای کلی دنباله را دربر می‌گیرد و اندازه آن را کاهش می‌دهد. این ممکن است برای کارایی و استفاده از منابع مفید باشد.

مقاومت در برابر طول متغیر دنباله: با گرفتن میانگین، عمل میانگین‌گیری کمتر به ترتیب و تعداد توکن‌ها در دنباله حساس است. این به کنترل دنباله‌های ورودی با طول‌های متغیر کمک می‌کند و مدل را در مقابل تفاوت‌های طول جملات مقاومت می‌کند.

تعمیم‌پذیری: میانگین‌گیری معمولاً جوهر یا اطلاعات میانگین را از تمام دنباله‌ها به‌دست می‌آورد. این می‌تواند به نمایش‌هایی مناسب برای وظایف پسین مانند طبقه‌بندی یا مقایسه شباهت منجر شود.

۴-۱-. آموزش مدل

۱. پیش پردازش داده ها

استخراج ویژگی: Wav۲Vec۲FeatureExtractor با یک مدل از پیش آموزش دیده "facebook/hubert-base-ls۹۶۰" بارگذاری شده است. این برای تبدیل داده های صوتی خام به فرمت مناسب برای مدل HUBERT استفاده می شود.

تابع Preprocess: تابع preprocess_function برای پردازش داده های صوتی تعریف شده است. این دستگاه از استخراج کننده ویژگی برای تبدیل آرایه های صوتی به ورودی های آماده مدل، از جمله مدیریت سرعت نمونه برداری، برش، و padding استفاده می کند.

اعمال پیش پردازش: این تابع برای مجموعه داده های آموزشی، آزمایشی و اعتبارسنجی (train_data، test_data، val_data) با استفاده از روش نقشه اعمال می شود. ستون گفتار پس از پردازش حذف می شود و مجموعه داده ها دسته بندی می شوند.

۲. آماده سازی مجموعه داده ها

رمزگذاری های پردازش شده برای مجموعه داده های قطار، آزمایش و اعتبارسنجی با استفاده از with_format ("torch.") به قالب PyTorch تبدیل می شوند.

۳. تعریف مدل

یک کلاس مدل سفارشی HUBERTClassification تعریف شده است که از nn.Module به ارث می رسد.

این مدل از HubertModel از پیش آموزش دیده و لایه های اضافی مانند Batch Normalization، ReLU، Dropout و لایه های Linear برای طبقه بندی استفاده می کند.

روش فوروارد محاسبات انجام شده در هر تماس را مشخص می کند. این شامل پیش پردازش ورودی ها و اعمال لایه ها به صورت متوالی است.

۴. راه اندازی برای آموزش

مدل سازی اولیه: مدل HUBERTClassification نمونه سازی می شود و به دستگاه مناسب (در صورت وجود GPU) منتقل می شود.

بهینه ساز: یک بهینه ساز AdamW با نرخ یادگیری e-۵۱ برای به روز رسانی وزن مدل در طول آموزش مقداردهی اولیه می شود.

۵. حلقه آموزش

یک حلقه آموزشی برای تعداد معینی از دوره ها تنظیم می شود.

Data Loader: DataLoader برای بارگذاری داده های آموزشی به صورت دسته ای استفاده می شود.

حلقه دوره ای: برای هر دوره، عملکرد مدل با استفاده از معیارهای تلفات و دقت ردیابی می شود. نوار پیشرفت از tqdm این را در زمان واقعی نشان می دهد.

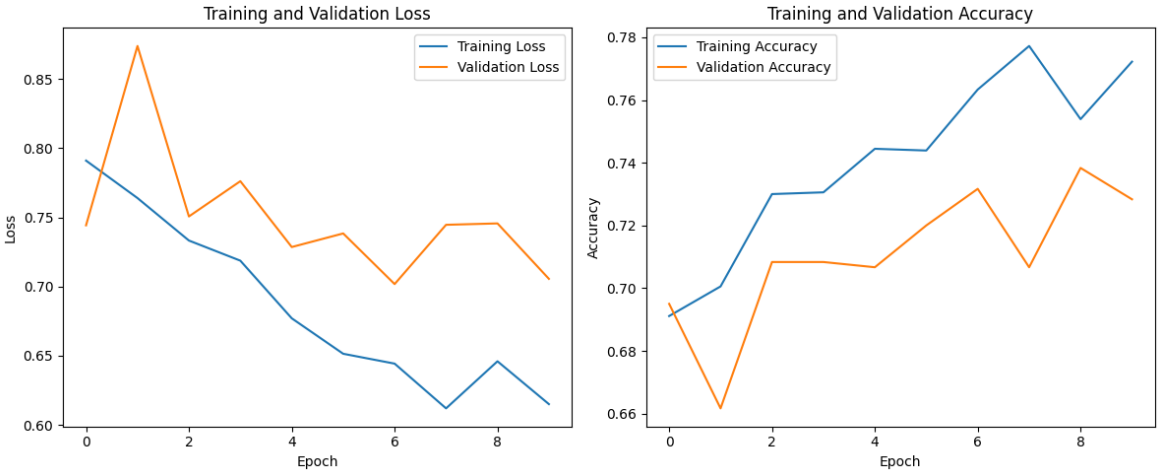
پردازش دسته‌ای: در هر دسته، گرادیان‌ها صفر می‌شوند، محاسبات انجام می‌شود، تلفات محاسبه و منتشر می‌شود و بهینه‌ساز پارامترهای مدل را به‌روزرسانی می‌کند.

محاسبه دقت: پیش‌بینی‌های مدل با برچسب‌های واقعی برای محاسبه دقت مقایسه می‌شوند.

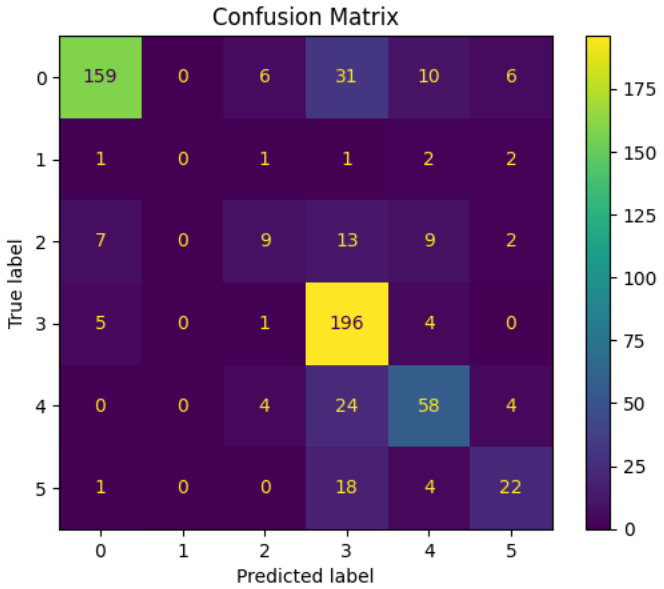
در این بخش ما آموزش را به سه شیوه انجام می دهیم که به صورت زیر ارائه می شوند:

حالت اول) به صورت معمول گفته شده:

در نهایت بعد از اتمامی تمامی مراحل بالا نتایج به صورت زیر به دست می آید:



شکل ۷: نتیجه عملکرد مدل اول



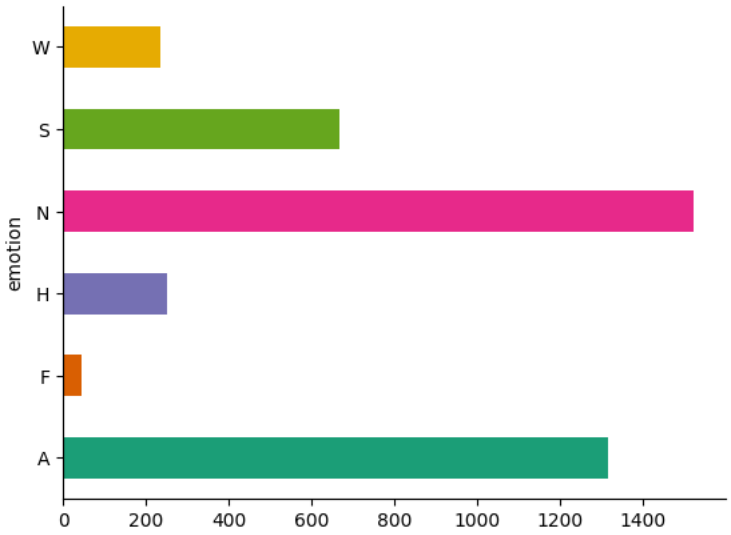
شکل ۸: ماتریس در هم آمختیگی مدل اول

```
Test Accuracy: 0.7400
Classification Report:
              precision    recall  f1-score   support

     Class 0       0.92      0.75      0.83       212
     Class 1       0.00      0.00      0.00         7
     Class 2       0.43      0.23      0.30        40
     Class 3       0.69      0.95      0.80       206
     Class 4       0.67      0.64      0.66        90
     Class 5       0.61      0.49      0.54        45

    accuracy                           0.74       600
   macro avg       0.55      0.51      0.52       600
weighted avg       0.74      0.74      0.73       600
```

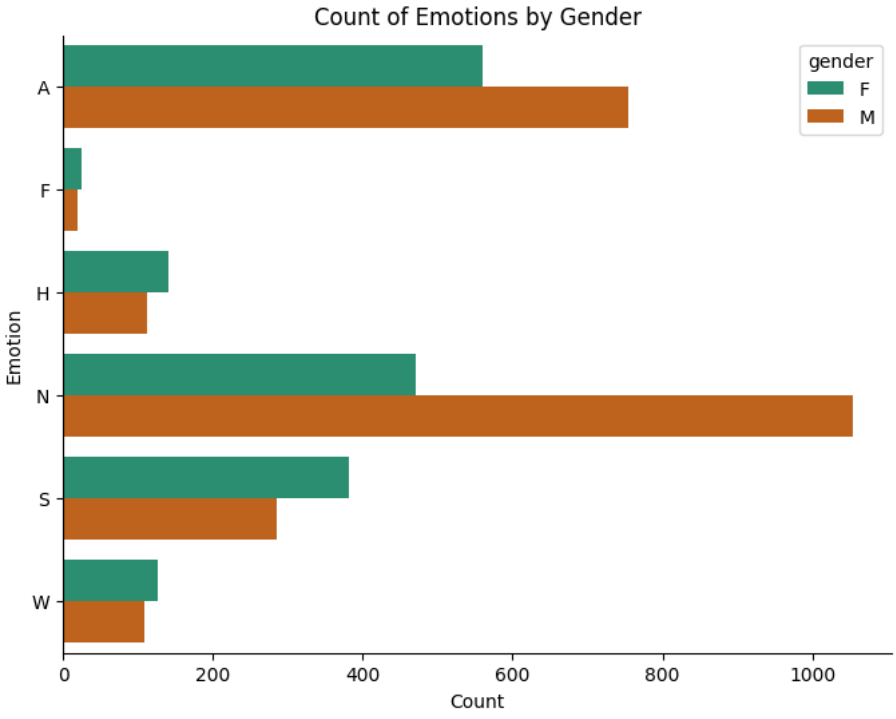
شکل ۹: نتایج طبقه بندی مدل اول

حالت دوم) تمام صوت‌ها به بخش‌های ۵ ثانیه‌ای تقسیم شده‌اند

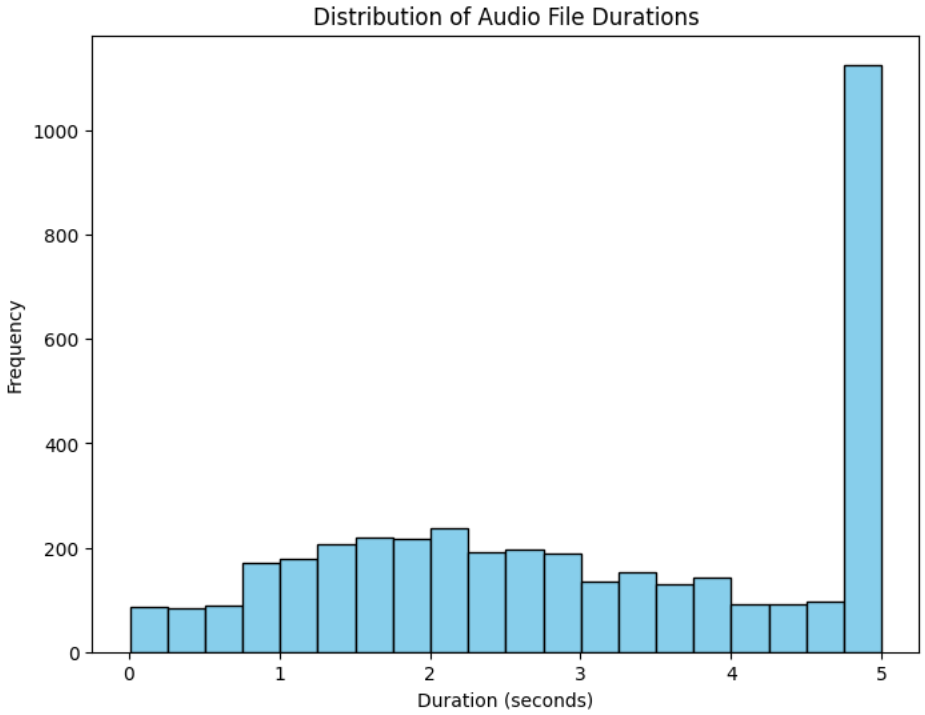
در این بخش برخلاف آن چه در بالا آوردیم طول داده ها را به صورت ۵ ثانیه در می آوریم که در ادامه داده ها را مشاهده میکنیم را مشاهده می کنیم:

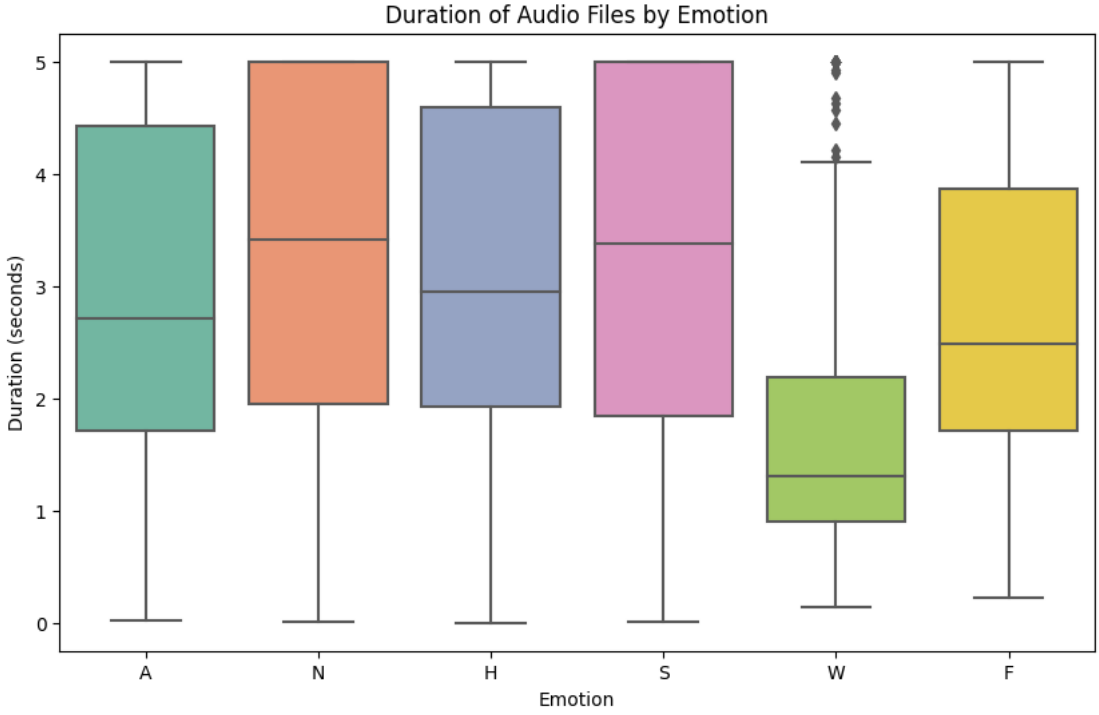


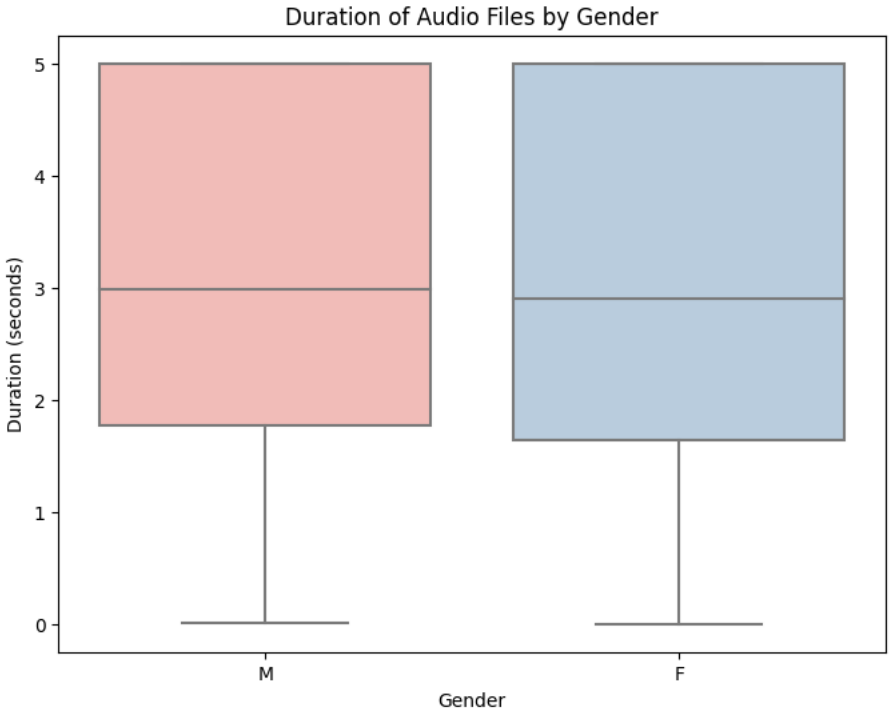
شکل ۱۰: توضیع داده ها بر اساس احساس



شکل ۱۱: توضیع داده های احساس بر اساس جنسیت

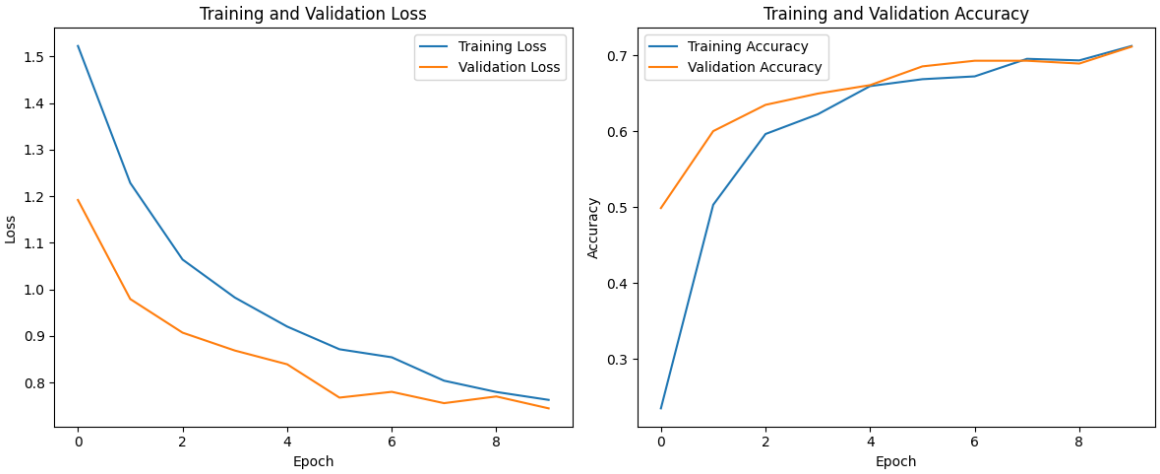

Distribution of Audio File Durations
Frequency
Duration (seconds)
0
200
400
600
800
1000
0
1
2
3
4
5


Duration of Audio Files by Emotion
Duration (seconds)
Emotion
A
N
H
S
W
F
0
1
2
3
4
5

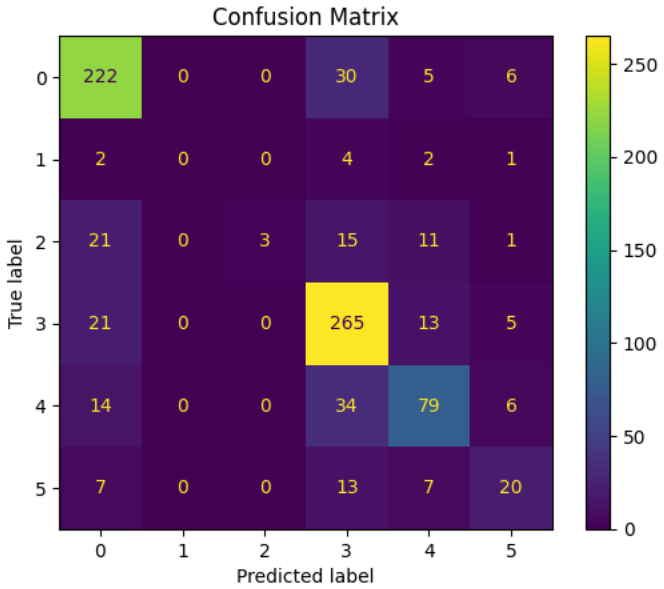


شکل ۱۲: زمان داده ها در هر کلاس

در نهایت بعد از اتمامی تمامی مراحل بالا نتایج به صورت زیر به دست می آید:



شکل ۱۳: نتیجه عملکرد مدل دوم



شکل ۱۴: ماتریس در هم آمخیتگی مدل دوم

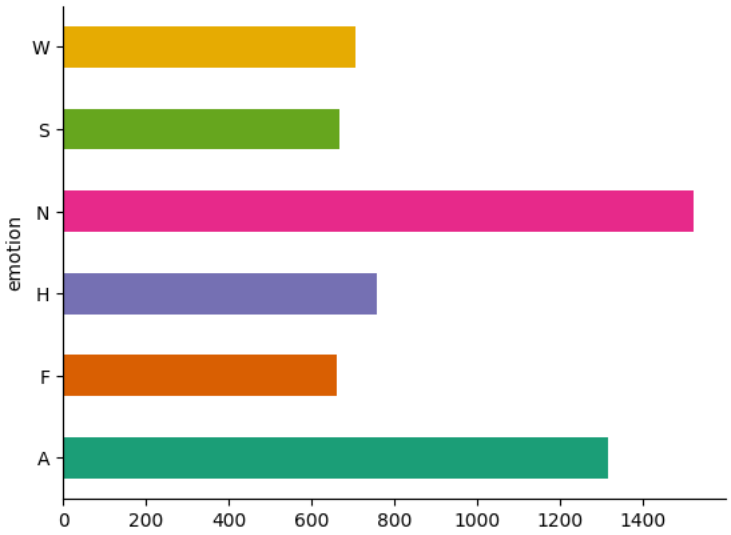
و در ادامه داریم:

```
Test Accuracy: 0.7299
Classification Report:
              precision    recall  f1-score   support

     Class 0       0.77      0.84      0.81       263
     Class 1       0.00      0.00      0.00         9
     Class 2       1.00      0.06      0.11        51
     Class 3       0.73      0.87      0.80       304
     Class 4       0.68      0.59      0.63       133
     Class 5       0.51      0.43      0.47        47

    accuracy                           0.73       807
   macro avg       0.62      0.47      0.47       807
weighted avg       0.73      0.73      0.70       807
```

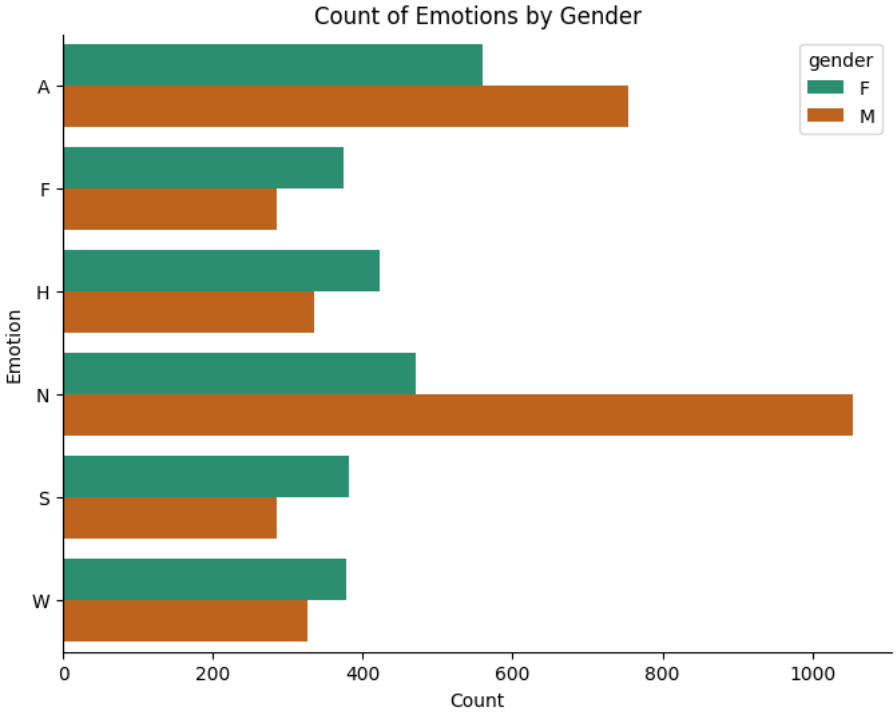
شکل ۱۵: نتایج طبقه بندی مدل دوم

حالت سوم) oversample

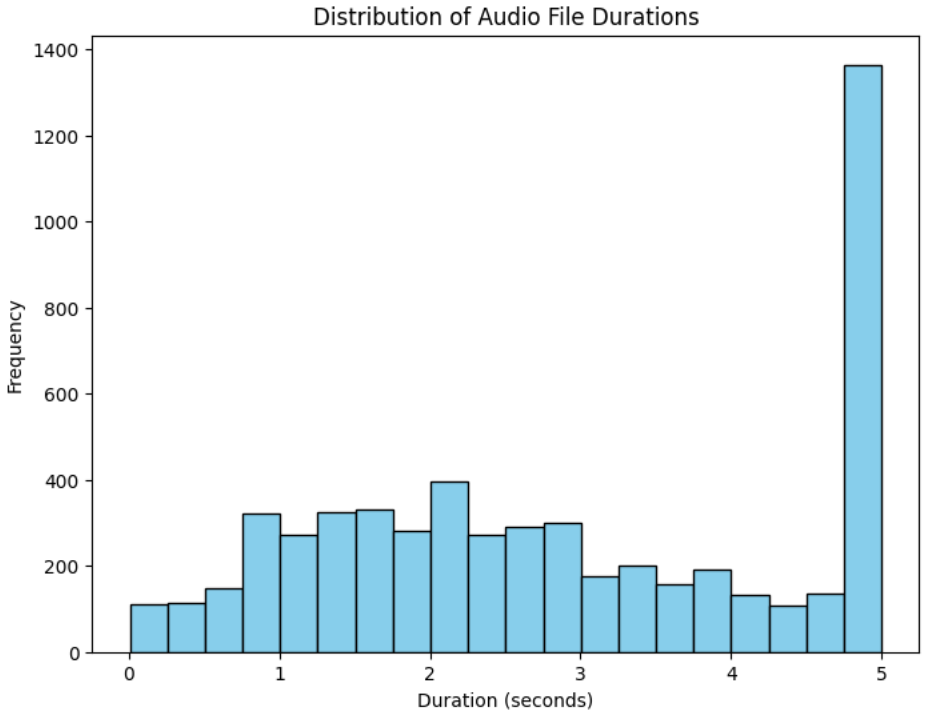
در این حالت عملیات oversampling را بر روی کلاس‌هایی که تعداد کمی دارند پیاده سازی می کنیم که داده ها به صورت زیر به دست می آیند:

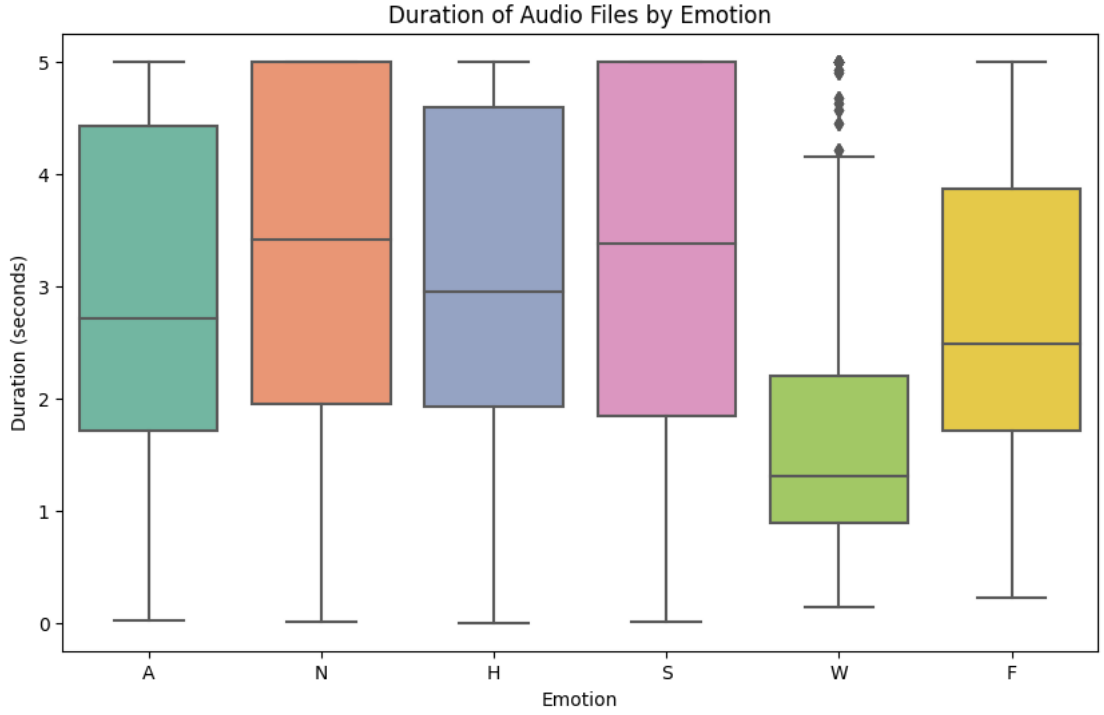


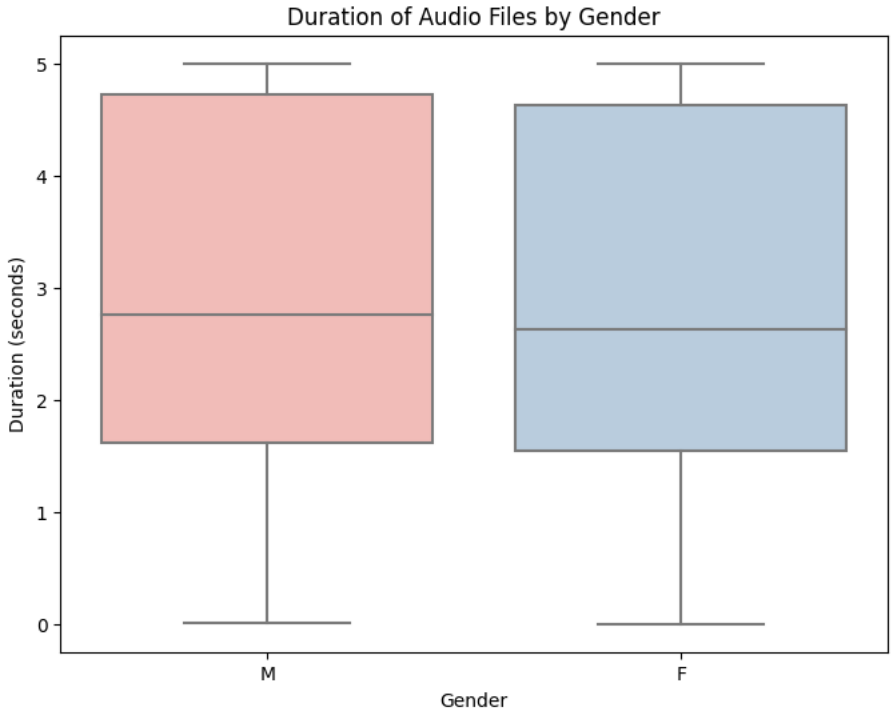
شکل ۱۶:توضیع داده ها براساس احساسات



شکل ۱۷: توضیع داده ها براساس جنسنیت

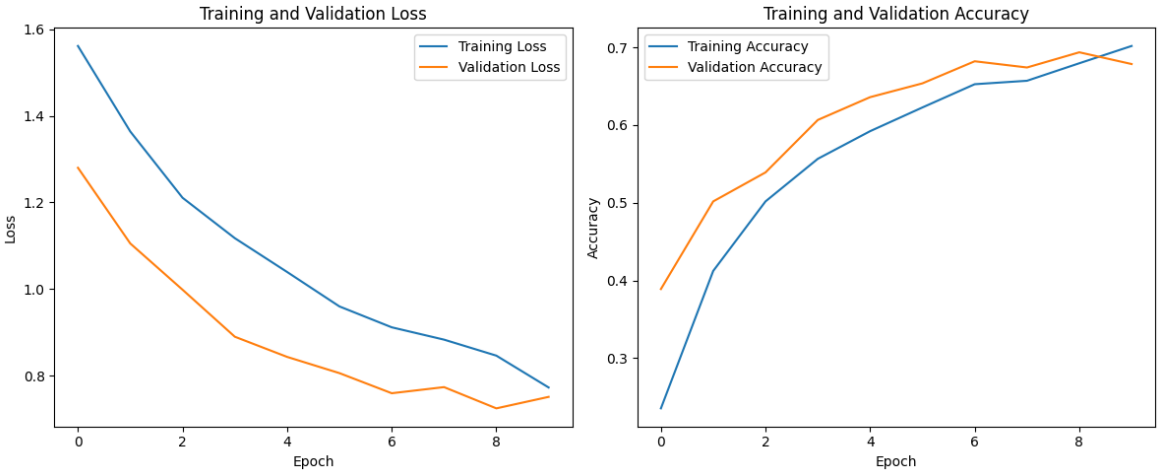

Distribution of Audio File Durations
Frequency
Duration (seconds)
0
200
400
600
800
1000
1200
1400
0
1
2
3
4
5


Duration of Audio Files by Emotion
Duration (seconds)
Emotion
A
N
H
S
W
F
0
1
2
3
4
5

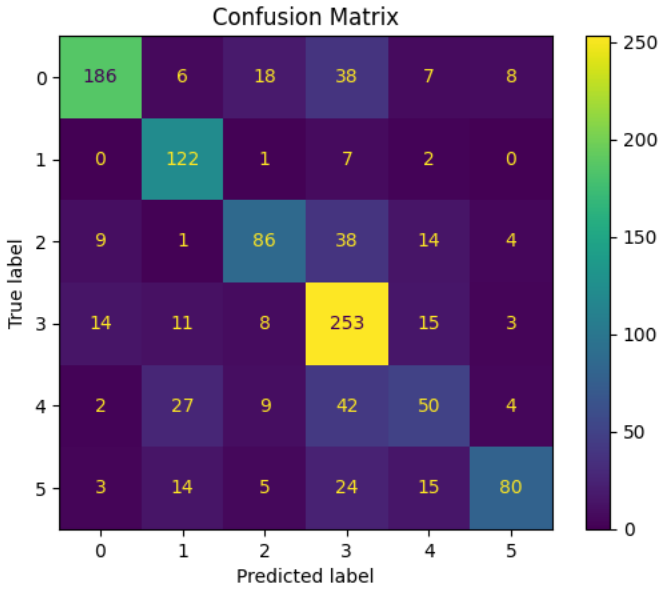


شکل ۱۸: زمان داده ها در هر کلاس

در نهایت بعد از اتمامی تمامی مراحل بالا نتایج به صورت زیر به دست می آید:



شکل ۱۹: نتیجه عملکرد مدل سوم



شکل ۲۰: ماتریس در هم آمخیتگی مدل

و در ادامه داریم:

```
Test Accuracy: 0.6901
Classification Report:
              precision    recall  f1-score   support

     Class 0       0.87      0.71      0.78       263
     Class 1       0.67      0.92      0.78       132
     Class 2       0.68      0.57      0.62       152
     Class 3       0.63      0.83      0.72       304
     Class 4       0.49      0.37      0.42       134
     Class 5       0.81      0.57      0.67       141

    accuracy                           0.69      1126
   macro avg       0.69      0.66      0.66      1126
weighted avg       0.70      0.69      0.68      1126
```

شکل ۲۱: نتایج طبقه بندی مدل

سه مجموعه نموداری که ارائه کرده‌اید، آموزش و از دست دادن اعتبار، و همچنین دقت آموزش و اعتبارسنجی را در طول دوره‌ها برای یک مدل یادگیری ماشین نشان می‌دهد. بیایید آنها را به ترتیب مقایسه کنیم:

۱. مجموعه اول نمودارها: مشاهده می کنیم که تلفات آموزشی در مقایسه با از دست دادن اعتبارسنجی که در دوره ۲ قبل از کاهش یک جهش قابل توجهی دارد، به آرامی کاهش می یابد. این سنبله می تواند نشان دهنده یک اضافه برازش موقت یا یک مشکل واریانس باشد. نمودارهای دقت یک روند صعودی کلی را برای دقت آموزشی و اعتبارسنجی نشان می‌دهند، که دقت آموزش با نرخ کمی ثابت‌تر از دقت اعتبار سنجی بهبود می‌یابد.

۲. مجموعه دوم نمودارها: این نمودارها سناریوی ایده آل تری را نشان می دهند. هم آموزش و هم از دست دادن اعتبار به طور پیوسته کاهش می یابد و همگرا می شوند، که نشان می دهد مدل به طور موثر یاد می گیرد و به خوبی به داده های دیده نشده تعمیم می دهد. نمودارهای دقت نیز نشان دهنده یادگیری خوب هستند، با افزایش زمان و دقت اعتبارسنجی هر دو در طول زمان و نزدیک به هم، که نشانه ای از استحکام مدل است.

۳. مجموعه سوم نمودارها: در این مورد، از دست دادن تمرین به طور مداوم کاهش می‌یابد، اما از دست دادن اعتبار در حدود دوره ۴ است، که نشان می‌دهد یادگیری بیشتر ممکن است به تعمیم بهتر ترجمه نشود. نمودارهای دقت نشان می دهد که دقت اعتبار سنجی تقریباً با دقت آموزشی در پایان آموزش مطابقت دارد، که نشان می دهد در حالی که مدل ممکن است شروع به بیش از حد برازش کند، هنوز عملکرد خوبی را در داده های اعتبار سنجی حفظ می کند.

با مقایسه هر سه، مجموعه دوم نمودارها مطلوب ترین است، که نشان دهنده یک مدل به خوبی تنظیم شده است. مجموعه اول مقداری نوسان در عملکرد مدل را نشان می دهد، در حالی که مجموعه سوم شروع بیش از حد برازش را نشان می دهد، جایی که مدل داده های آموزشی را به خوبی یاد می گیرد اما پس از یک نقطه خاص در داده های اعتبار سنجی چندان بهبود نمی یابد.

پاسخ ۲. تنظیم دقیق مدل BERT

۲-۱. آموزش و تحلیل مدل

BERT یک مدل پیشگامانه در زمینه پردازش زبان طبیعی (NLP) است که توسط گوگل در سال ۲۰۱۸ معرفی شد. این مدل نشان دهنده تغییر قابل توجهی در نحوه پردازش و درک زبان انسان توسط مدل های مبتنی بر شبکه عصبی است. جنبه های کلیدی BERT عبارتند از:

۱. معماری ترانسفورماتور: BERT بر اساس معماری ترانسفورماتور است که بر مکانیزم های توجه برای درک متن یک کلمه در یک جمله متکی است. برخلاف مدل‌های قبلی که متن را به روشی متوالی پردازش می‌کردند (مانند RNN و LSTM)، ترانسفورماتورها کل متن را یک‌باره پردازش می‌کنند و امکان درک دقیق‌تری از زمینه را فراهم می‌کنند.

۲. زمینه دو جهته: BERT برای درک متن یک کلمه بر اساس تمام کلمات اطراف آن (چه قبل و چه بعد از آن) طراحی شده است، به همین دلیل به آن "دو طرفه" می گویند. این یک پیشرفت چشمگیر نسبت به مدلهای قبلی است ، که به طور معمول متن را در یک جهت واحد (یا چپ به راست یا راست به چپ) می خوانند.

۳. پیش آموزش و تنظیم دقیق: BERT ابتدا روی مجموعه بزرگی از متن با استفاده از دو کار از قبل آموزش داده می شود: مدل سازی زبان پوشانده شده (که در آن کلمات تصادفی پوشانده می شوند و مدل برای پیش بینی آنها آموزش داده می شود) و بعد. پیش بینی جمله پس از پیش‌آموزش، می‌توان آن را با لایه‌های اضافی تنظیم کرد تا طیف وسیعی از وظایف زبانی خاص مانند تجزیه و تحلیل احساسات، پاسخ‌گویی به سؤال و ترجمه زبان را انجام دهد.

۴. اثربخشی: BERT عملکرد قابل توجهی را در بسیاری از وظایف NLP نشان داده است و استانداردهای جدیدی را در معیارهایی مانند GLUE (ارزیابی درک عمومی زبان) و SQuAD (مجموعه داده پاسخ به سؤالات استنفورد) تنظیم کرده است.

به طور خلاصه، در حالی که BERT یک چارچوب کلی برای درک زبان انسانی ارائه می‌کند، پارسبرت این چارچوب را به طور خاص به زبان فارسی تطبیق می‌دهد و آن را برای کارهای مربوط به متن فارسی مؤثرتر می‌کند.

پارس برت (مدلی مبتنی بر ترانسفورماتور برای درک زبان فارسی)

ParsBERT یک مدل BERT است که به طور خاص بر روی مجموعه بزرگی از متن فارسی از قبل آموزش داده شده است. این مدل برای درک و پردازش بهتر زبان فارسی که ویژگی ها و چالش های زبانی منحصر به فرد خود را دارد، اقتباس شده است. جنبه های کلیدی ParsBERT عبارتند از:

تخصص زبان فارسی: در حالی که BERT مدلی است که می تواند با بسیاری از زبان ها تطبیق داده شود، پارسBERT به طور خاص بر روی متن فارسی آموزش داده شده است و آن را برای کارهای مربوط به زبان فارسی کارآمدتر و دقیق تر می کند.

تفاوت های ظریف فرهنگی و زبانی: پارسبرت در درک تفاوت های ظریف زبان فارسی، از جمله دستور زبان، محاوره ها و ارجاعات فرهنگی آن که برای پردازش دقیق زبان بسیار مهم هستند، بهتر است.

کاربردها: ParsBERT را می توان برای انواع وظایف زبان فارسی، از جمله اما نه محدود به تحلیل احساسات، طبقه بندی متن، تشخیص موجودیت نامگذاری شده (NER) و پاسخگویی به سوالات به زبان فارسی استفاده کرد.

مشارکت جامعه: پارسبرت نمونه ای از تلاش های جامعه محور برای انطباق فناوری پیشرفته NLP با زبان های غیر انگلیسی است که به تنوع رو به رشد در فناوری زبان کمک می کند.

به طور خلاصه، در حالی که BERT یک چارچوب کلی برای درک زبان انسانی ارائه می‌کند، پارسبرت این چارچوب را به طور خاص به زبان فارسی تطبیق می‌دهد و آن را برای کارهای مربوط به متن فارسی مؤثرتر می‌کند.

۲-۱-۱. پیش پردازش داده ها

FarsTail یک مجموعه داده برای استنتاج زبان طبیعی (NLP) به زبان فارسی است. در این مجموعه داده، هر نمونه شامل یک "پیش‌فرض" (premise) و یک "فرضیه" (hypothesis) است، و هدف تعیین ارتباط استنتاجی بین این دو است. این ارتباط می‌تواند "استنتاج" (ENTAILMENT)، "تناقض" (CONTRADICTION)، یا "بی‌طرف" (NEUTRAL) باشد.

با توجه به خروجی‌های بارگذاری، هر فایل CSV شامل ستون‌های زیر به نظر می‌رسد:

- `premise`: جمله یا بیانیه‌ای که به عنوان پایه یا مبنایی برای استنتاج است.

- `hypothesis`: جمله‌ای که باید تعیین شود آیا از پیش‌فرض قابل استنتاج است، با آن تناقض دارد، یا هیچکدام.

- `label`: برچسب دسته‌بندی که نشان می‌دهد رابطه بین پیش‌فرض و فرضیه چیست (استنتاج، تناقض، یا بی‌طرف).

دیتاستی که داریم به این صورت تعریف می شود.

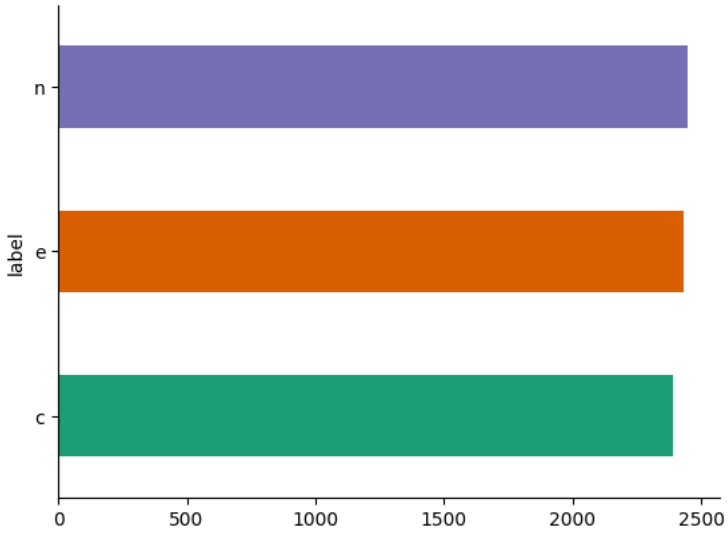
در این بخش داریم:

یک عملکرد پیش پردازش برای داده های متنی را توصیف می کند. این تابع با مقداردهی اولیه یک «Normalizer» و یک «PorterStemmer» که ابزارهای رایج در پردازش زبان طبیعی برای استانداردسازی و ساده‌سازی متن هستند، آغاز می‌شود. برای هر قطعه از متن در آرایه داده ورودی، تابع، متن را به کلمات جداگانه تبدیل می‌کند، کلمات توقف (کلمات رایجی که معمولاً در پردازش زبان حذف می‌شوند) را فیلتر می‌کند و سپس کلمات باقی‌مانده را به یک رشته واحد می‌پیوندد. علاوه بر این، کد نویسه یونیکد «u۲۰۰c\» را که معمولاً به‌عنوان غیرمجاز با عرض صفر در متن استفاده می‌شود، حذف می‌کند. پس از این فرآیند، متن با استفاده از «Normalizer» عادی می شود. این تابع برای تمیز کردن و استاندارد کردن داده‌های متنی طراحی شده است و آن را برای پردازش یا تجزیه و تحلیل بیشتر، مانند مدل‌های یادگیری ماشین، مناسب‌تر می‌کند.

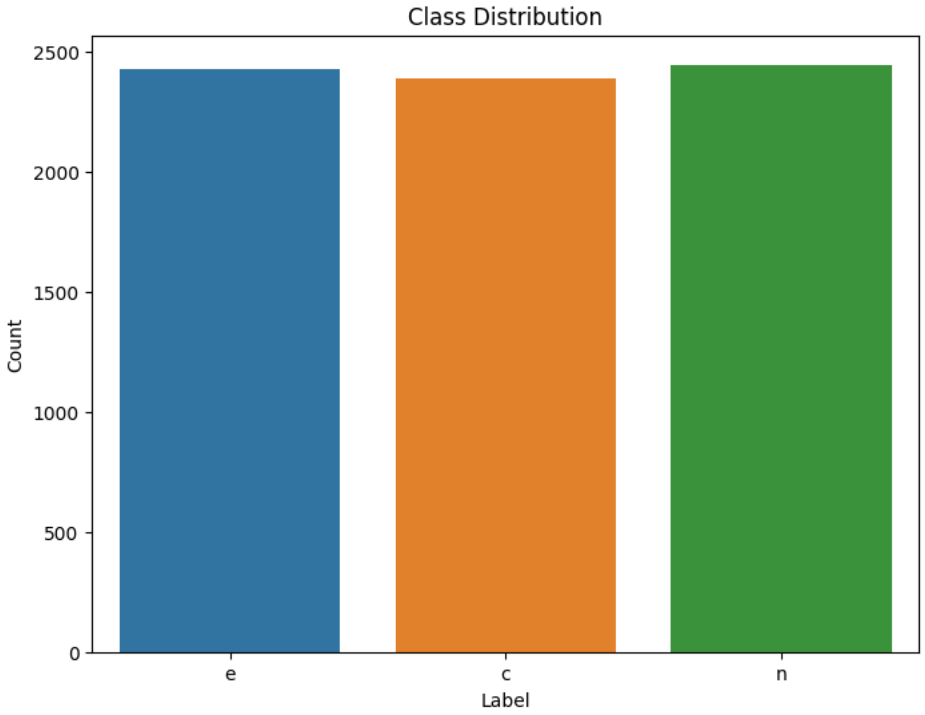
```
print(model)
```

```
BertForSequenceClassification(
  (bert): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(100000, 768, padding_idx=0)
      (position_embeddings): Embedding(512, 768)
      (token_type_embeddings): Embedding(2, 768)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0-11): 12 x BertLayer(
          (attention): BertAttention(
            (self): BertSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
            (output): BertSelfOutput(
              (dense): Linear(in_features=768, out_features=768, bias=True)
              (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
          )
          (intermediate): BertIntermediate(
            (dense): Linear(in_features=768, out_features=3072, bias=True)
            (intermediate_act_fn): GELUActivation()
          )
```

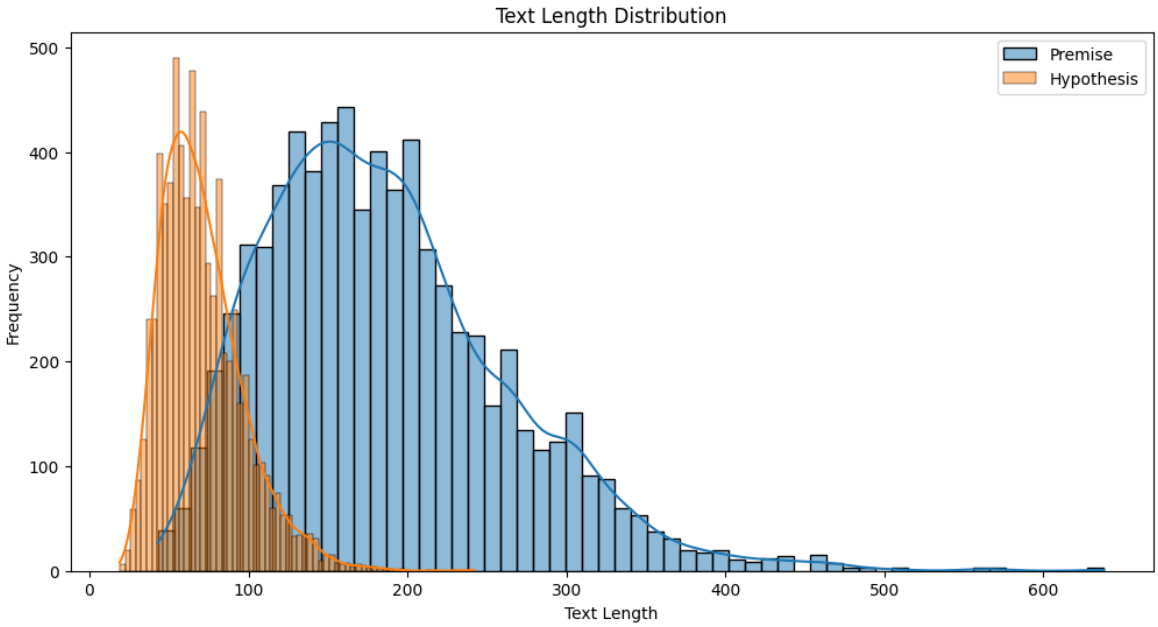
شکل ۲۲:مدل **bert**



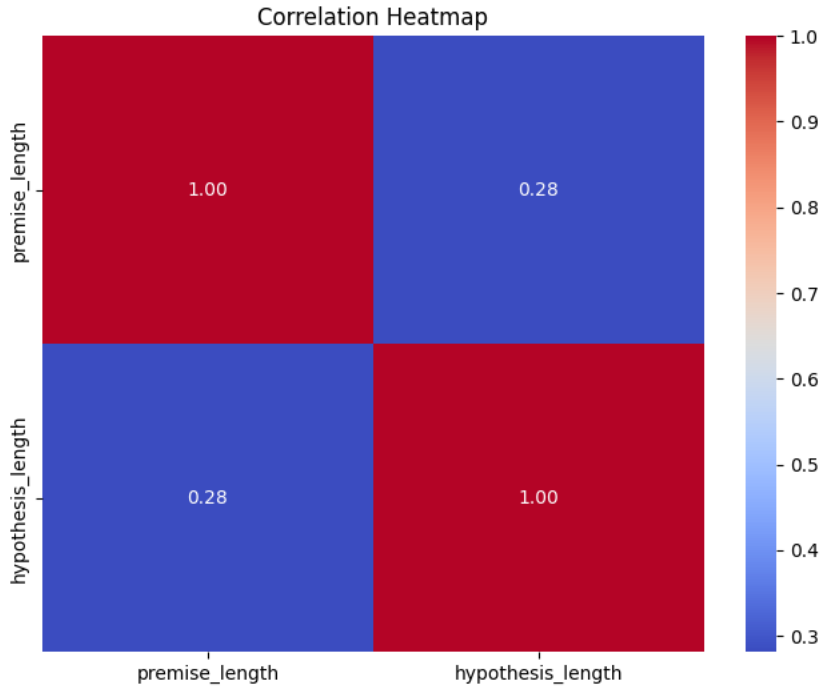
شکل ۲۳: داده های آموزش



شکل ۲۴: توضیع کلاس ها



شکل ۲۵: توضیع داده بر حسب سایز



شکل ۲۶: میزان در هم آمیختگی داده ها

	premise	hypothesis	label
0	... اولین انتقال و نفوذ طبیعی فرهنگ و تمدن اسلامی	...نخستین انتقال و نفوذ طبیعی فرهنگ و تمدن اسلامی	e
1	... اولین انتقال و نفوذ طبیعی فرهنگ و تمدن اسلامی	...کانون های جغرافیایی مصر، اندلس و شام، نخستین ر	c
2	... اولین انتقال و نفوذ طبیعی فرهنگ و تمدن اسلامی	...سیسیل بعد از اسپانیا بزرگ ترین کانونی بود که ه	n
3	...ویژگی های هنر عصر اموی: ۱- تلفیقی بودن ۲- بازن	... نقاشی های تزئینی و تندیس های بی‌کیفیت، یکی از	e
4	...ویژگی های هنر عصر اموی: ۱- تلفیقی بودن ۲- بازن	...با کیفیت بودن تندیس های دوره اموی، یکی از ویژگ	c
...	...	...	...
7261	...قانون اساسی جمهوری اسلامی ایران در سال ۱۳۵۸ تو	... تعداد فصول قانون اساسی ۱۴ و تعداد اصول آن ۱۷۷	e
7262	...قانون اساسی جمهوری اسلامی ایران در سال ۱۳۵۸ تو	.قانون اساسی دارای ۲۵ فصل و ۱۷۵ اصل می باشد	c
7263	...قانون اساسی جمهوری اسلامی ایران در سال ۱۳۵۸ تو	...در ۲۴ آبان ۵۸ کار تدوین قانون اساسی به پایان ر	n
7264	...محاصره اقتصادی پیامبر (ص) و یارانش که در سال ه	...حضرت محمد (ص) و یارانش از ششمین سال بعثت تا هش	c
7265	...محاصره اقتصادی پیامبر (ص) و یارانش که در سال ه	...در دوران محاصره اقتصادی رسول اکرم (ص) و پیروان	n

7266 rows × 3 columns

شکل ۲۷: داده های آموزش

۲-۱-۲. تنظیم دقیق مدل

در این بخش فرآیندی را برای تنظیم دقیق مدل یادگیری ماشین، به طور خاص با استفاده از PyTorch و بهینه‌ساز AdamW ترسیم می‌کند. در پاراگراف اول، تنظیمات برای تنظیم دقیق مدل توضیح داده شده است. بهینه ساز انتخاب شده AdamW است که با نرخ یادگیری e-۵۲ تنظیم شده است. این مدل در پنج دوره تحت تنظیم دقیق قرار می گیرد. تعداد دوره‌ها نشان می‌دهد که کل مجموعه داده آموزشی چند بار از طریق شبکه عصبی به جلو و عقب منتقل می‌شود.

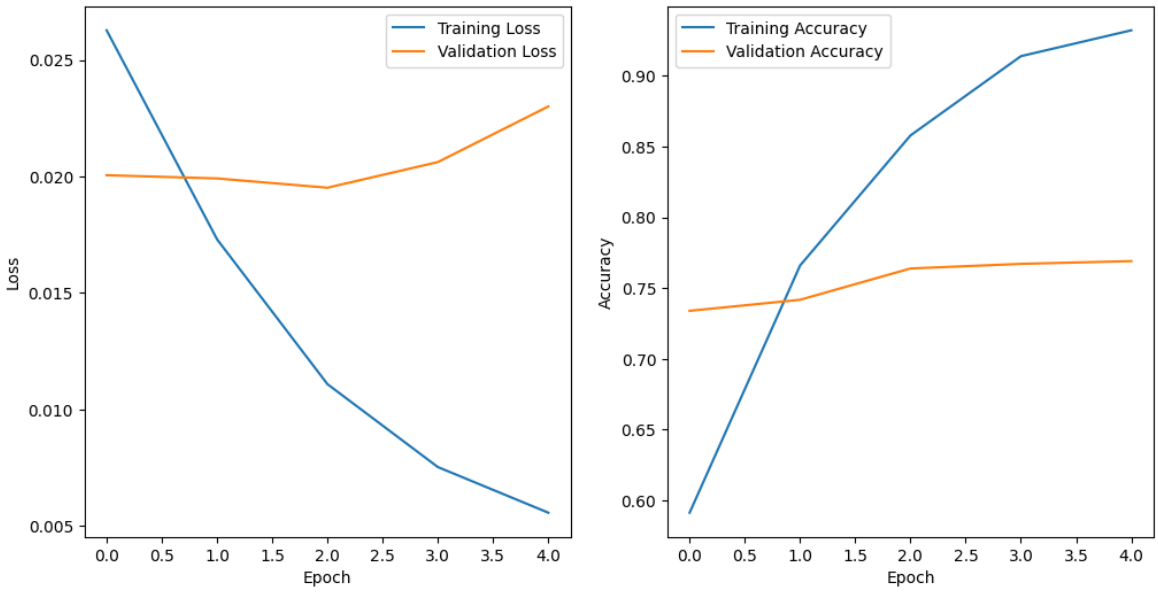
پاراگراف دوم جزئیات حلقه تنظیم دقیق را نشان می دهد. در هر دوره، مدل به حالت آموزش تنظیم می شود. داده های آموزشی به صورت دسته ای پردازش می شوند، جایی که برای هر دسته، گرادیان صفر تنظیم می شود و مدل خروجی ها را بر اساس شناسه های ورودی، ماسک توجه و برچسب ها محاسبه می کند. سپس ضرر محاسبه می‌شود، انتشار پس‌انداز با فراخوانی «loss.backward()» انجام می‌شود و بهینه‌ساز پارامترهای مدل را به‌روزرسانی می‌کند. پس از مرحله آموزش در هر دوره، مدل به حالت ارزیابی تنظیم می شود. در این مرحله، از دست دادن اعتبار بر روی مجموعه داده اعتبارسنجی بدون ردیابی گرادیان (`torch.no_grad()` محاسبه می‌شود، زیرا برای ارزیابی مدل به جای آموزش استفاده می‌شود.

پاراگراف پایانی نتیجه گیری فرآیند تنظیم دقیق را مورد بحث قرار می دهد. پس از اتمام تمام دوره‌ها، میانگین افت اعتبار برای ارزیابی عملکرد مدل بر روی داده‌های دیده نشده محاسبه می‌شود. این نشان می دهد که مدل در طول آموزش چقدر خوب یاد گرفته است. اسکریپت با ذخیره مدل تنظیم شده در یک دایرکتوری مشخص به پایان می رسد. این مرحله برای حفظ پارامترهای مدل تنظیم شده بسیار مهم است و امکان استفاده مجدد از مدل تنظیم شده را در کارهای آینده بدون نیاز به تکرار فرآیند آموزشی فراهم می کند.

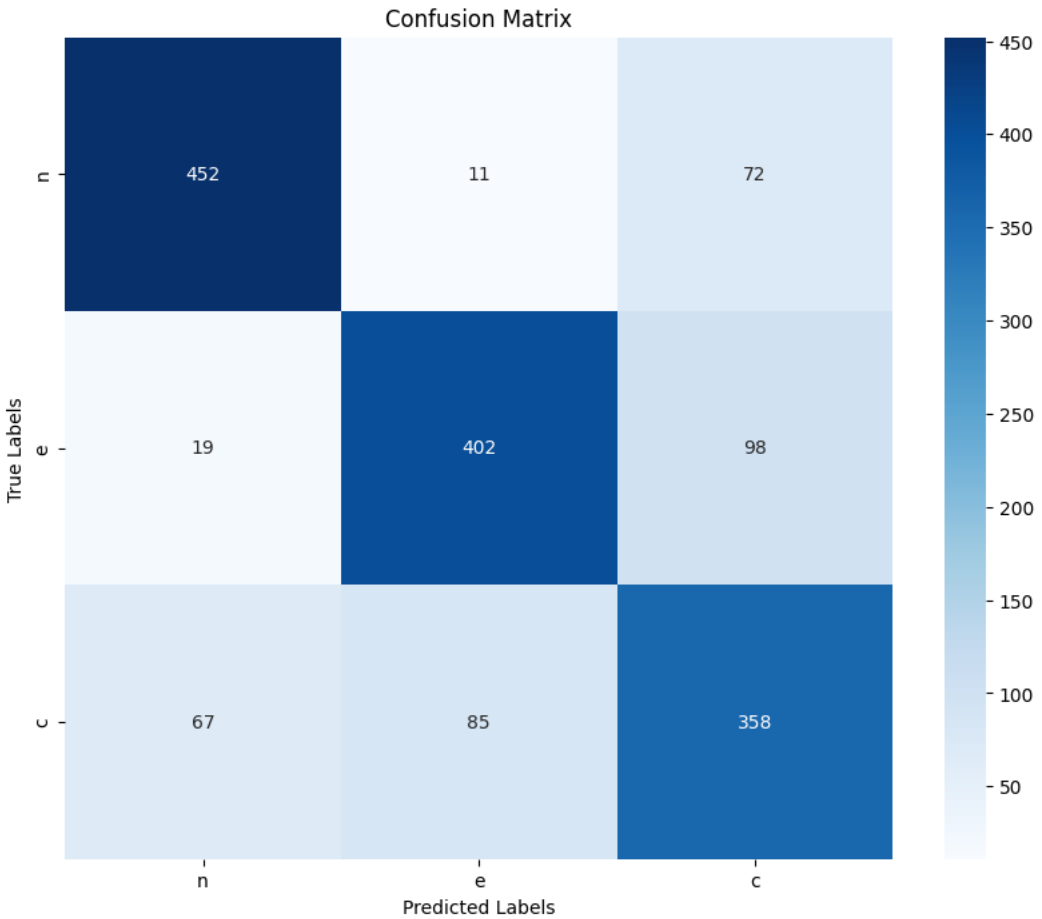
بعد از تنظیم کردن داده ها نتایج به صورت زیر به دست می آید:

```
Epoch 1/5 - Validation Loss: 0.020057737192280082, Validation Accuracy: 0.7338972023422251
Epoch 2/5 - Validation Loss: 0.019919300125881535, Validation Accuracy: 0.741704619388419
Epoch 3/5 - Validation Loss: 0.019520671137777643, Validation Accuracy: 0.763825634352635
Epoch 4/5 - Validation Loss: 0.020621727252828664, Validation Accuracy: 0.7670787247885491
Epoch 5/5 - Validation Loss: 0.023010221563846783, Validation Accuracy: 0.7690305790500976
```

شکل ۲۸: نتایج پیاده سازی



شکل ۲۹: نتایج فاین تیون



شکل ۳۰: ماتریس آشفتگی

```
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.90      0.86       535
           1       0.79      0.82      0.80       519
           2       0.76      0.65      0.70       510

    accuracy                           0.79      1564
   macro avg       0.79      0.79      0.79      1564
weighted avg       0.79      0.79      0.79      1564

F1 Score: 0.7874729333304019
```

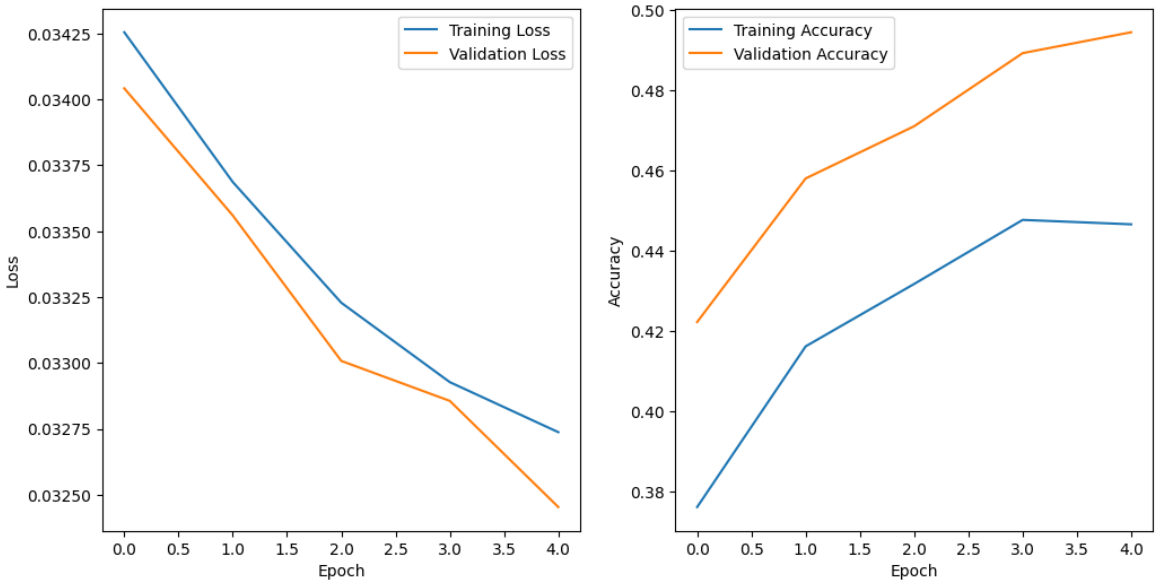
شکل ۳۱: نتایج عددی دقیق

۲-۱-۳. فریز کردن لایه های مدل

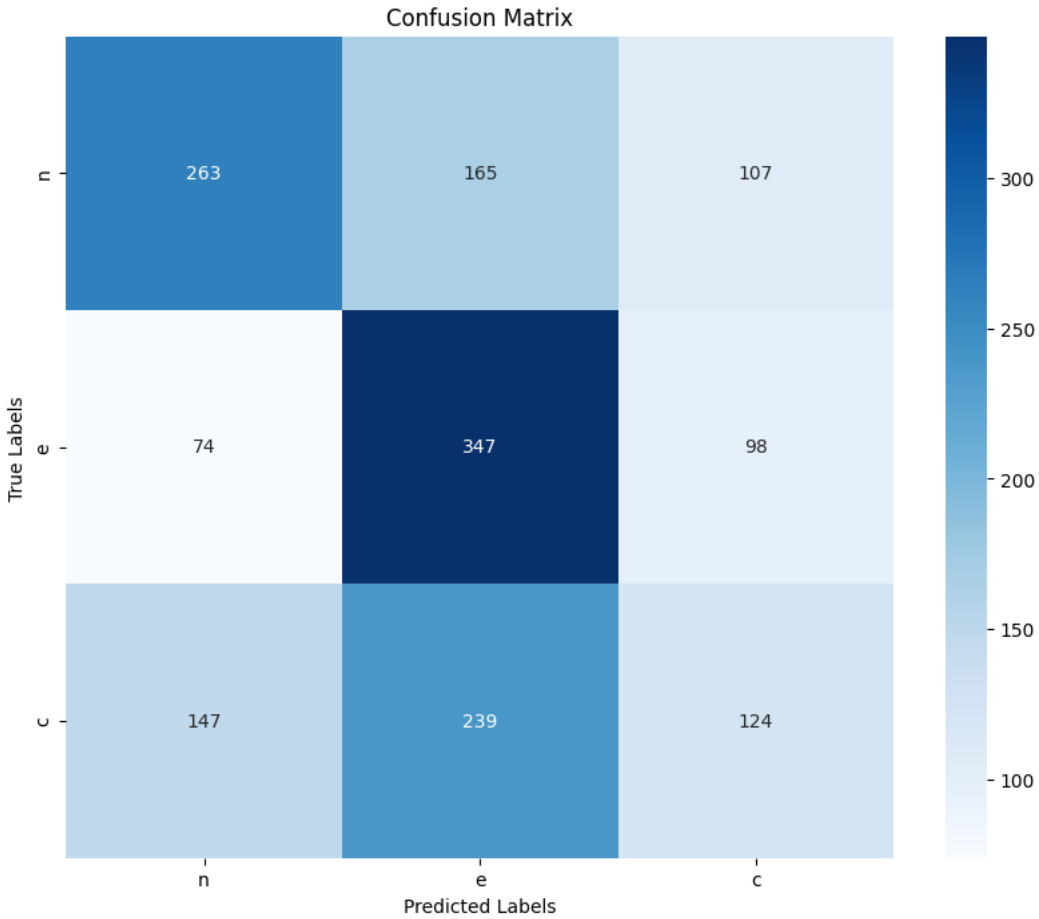
در ابتدا ۹ لایه ابتدایی را فریز می کنیم که در ادامه به آن می پردازیم:

در بخش، یک مدل BERT از پیش آموزش دیده، به طور خاص "bert-base-parsbert-uncased" از HooshvareLab ، برای یک کار طبقه بندی دنباله ای با سه برچسب خروجی ممکن تطبیق داده شده است. فرآیند تنظیم دقیق شامل انجماد پارامترهای مدل پایه برای جلوگیری از به روز شدن آنها در طول آموزش است، بنابراین دانش از قبل آموزش دیده حفظ می شود (برای ۹ لایه) در حالی که فقط لایه های بالاتر تنظیم شده اند. این با تنظیم ویژگی «requires_grad» روی «نادرست» برای همه پارامترهای مدل پایه به دست می‌آید. سپس مدل با استفاده از یک GPU در صورت وجود به دستگاه محاسباتی مناسب منتقل می شود. تنظیم دقیق از بهینه ساز AdamW با نرخ یادگیری پیش‌فرض ۲-e ۵استفاده می‌کند، اما رویکرد گران‌تری را برای نرخ‌های یادگیری اعمال می‌کند: پارامترها از لایه نهم رمز گذار به بعد با نرخ پیش‌فرض تنظیم می‌شوند، در حالی که طبقه‌بندی کننده پارامترها با نرخ یادگیری کمتر ۴-۱۰^۸ تنظیم شده اند. این رویکرد نرخ یادگیری تفاضلی، امکان به روزرسانی دقیق‌تر وزن های مدل را فراهم می کند، با به‌روزرسانی محتاطانه تر برای طبقه‌بندی کننده اعمال می‌شود که مستقیماً بر خروجی تأثیر می گذارد و به طور بالقوه منجر به عملکرد پایدارتر و دقیق تر می‌شود.

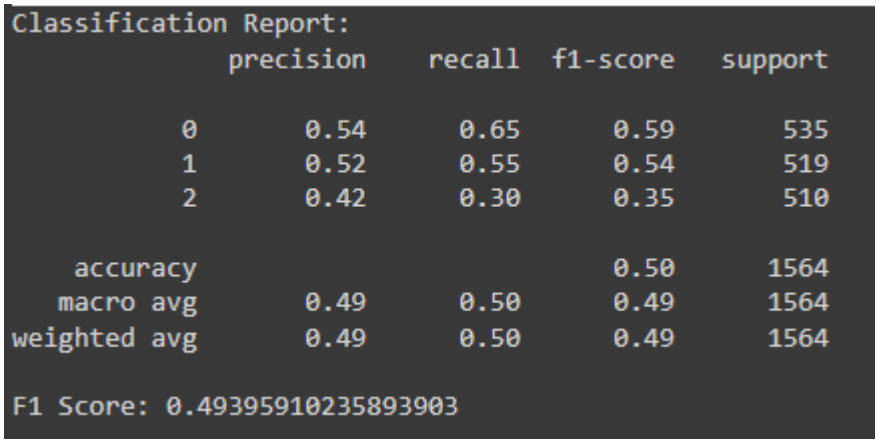
حال با این پیاده سازی نتایج به صورت زیر به دست می آیند:



شکل ۳۲: نمودار مربوط به ۹ لایه فریز



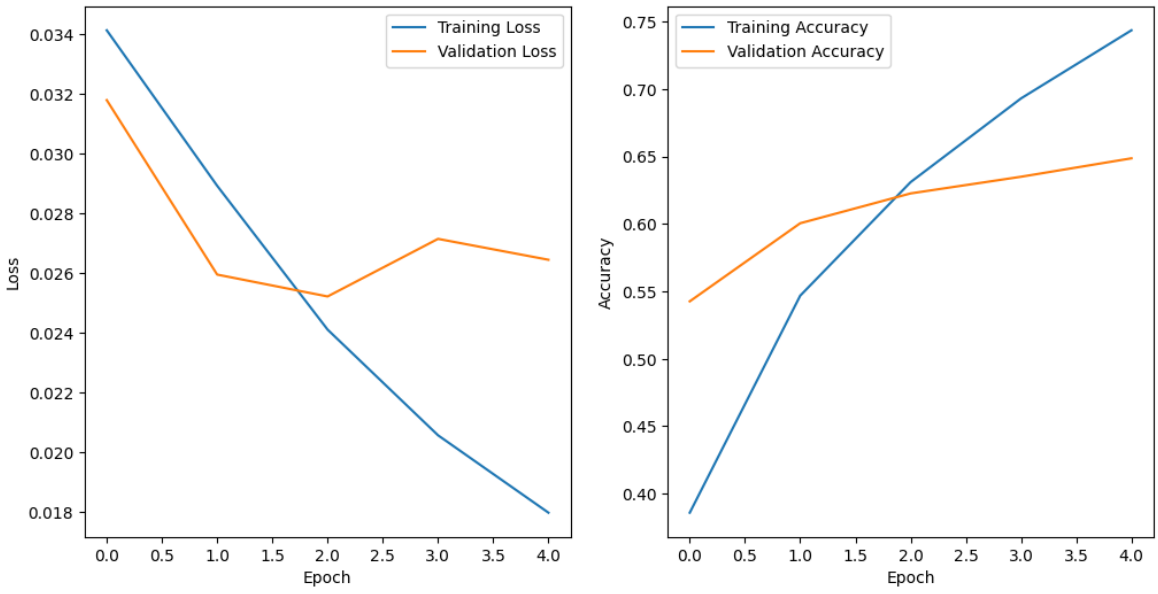
شکل ۳۳: ماتریس آشفتگی مربوط به ۹ لایه فریز



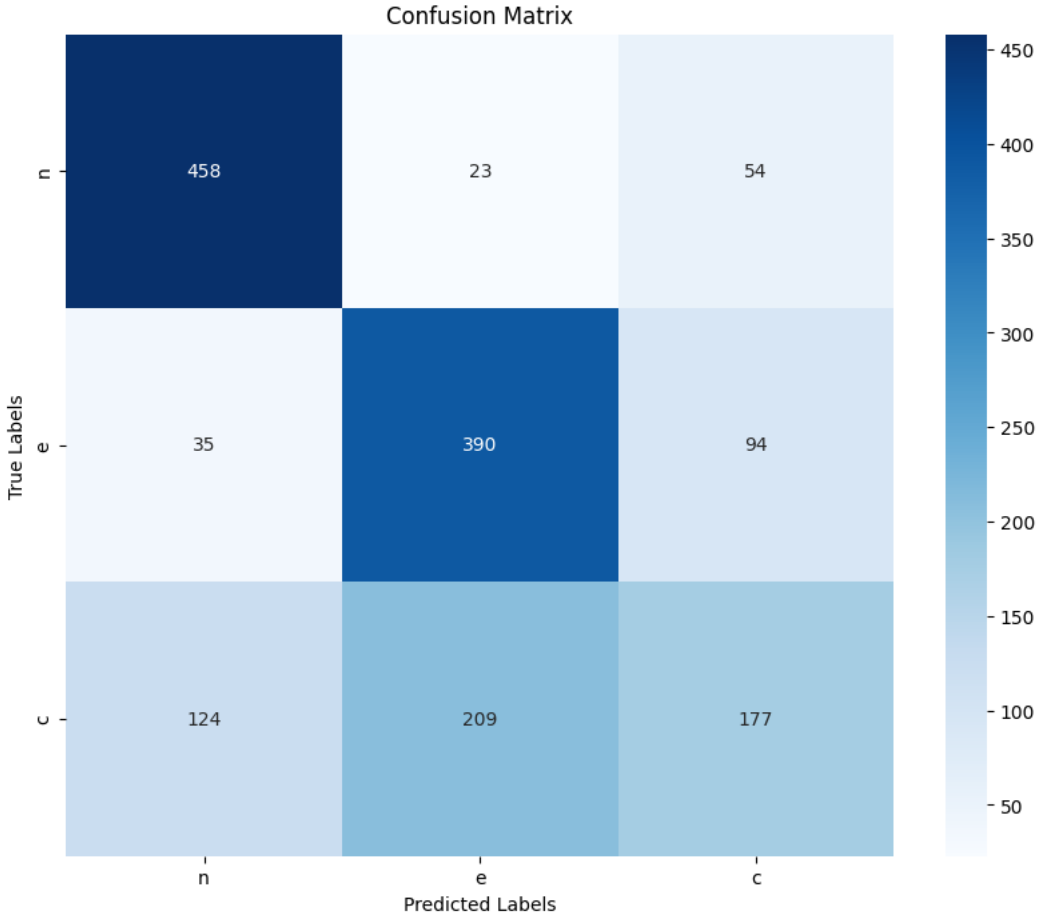
شکل ۳۴: خروجی مربوط به ۹ لایه فریز

در ادامه عملیات فریز کردن داده ها را بر روی تمامی لایه ها به جز لایه آخر و لایه embbeding انجام می دهیم که داریم:

دراین قسمت، فرآیند تنظیم دقیق شامل انجماد پارامترهای مدل پایه برای جلوگیری از به روز شدن آنها در طول آموزش است، بنابراین دانش از قبل آموزش دیده حفظ می شود در حالی که فقط لایه های بالاتر تنظیم شده اند. این با تنظیم ویژگی «requires_grad» روی «نادرست» برای همه پارامترهای مدل پایه به دست می‌آید. سپس مدل با استفاده از یک GPU در صورت وجود به دستگاه محاسباتی مناسب منتقل می شود. تنظیم دقیق از بهینه‌ساز AdamW با نرخ یادگیری پیش‌فرض e-۵۲ استفاده می‌کند، اما رویکرد گران‌تری را برای نرخ‌های یادگیری اعمال می‌کند: پارامترها از لایه نهم رمزگذار به بعد با نرخ پیش‌فرض تنظیم می‌شوند، در حالی که طبقه‌بندی‌کننده پارامترها با نرخ یادگیری کمتر e-۴۱ تنظیم شده اند. این رویکرد نرخ یادگیری تفاضلی، امکان به‌روزرسانی دقیق‌تر وزن‌های مدل را فراهم می‌کند، با به‌روزرسانی محتاطانه‌تر برای طبقه‌بندی‌کننده اعمال می‌شود که مستقیماً بر خروجی تأثیر می‌گذارد و **به طور بالقوه منجر به عملکرد پایدارتر و دقیق‌تر می‌شود.**



شکل ۳۵: نمودار مربوط به همه لایه ها فریز شده



شکل ۳۶: ماتریس آشفتگی مربوط به همه لایه ها فریز شده

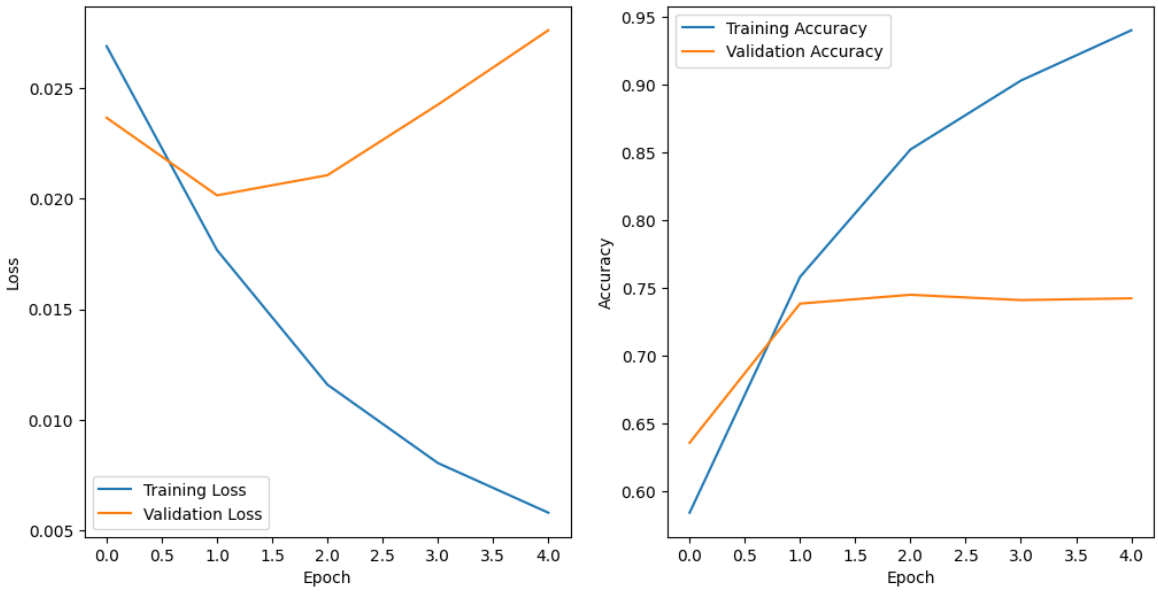
```
Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.82      0.78       535
           1       0.64      0.70      0.67       519
           2       0.51      0.41      0.45       510

    accuracy                           0.65      1564
   macro avg       0.63      0.64      0.64      1564
weighted avg       0.63      0.65      0.64      1564

F1 Score: 0.6377880733891133
```

شکل ۳۷: خروجی مربوط به همه لایه ها فریز شده

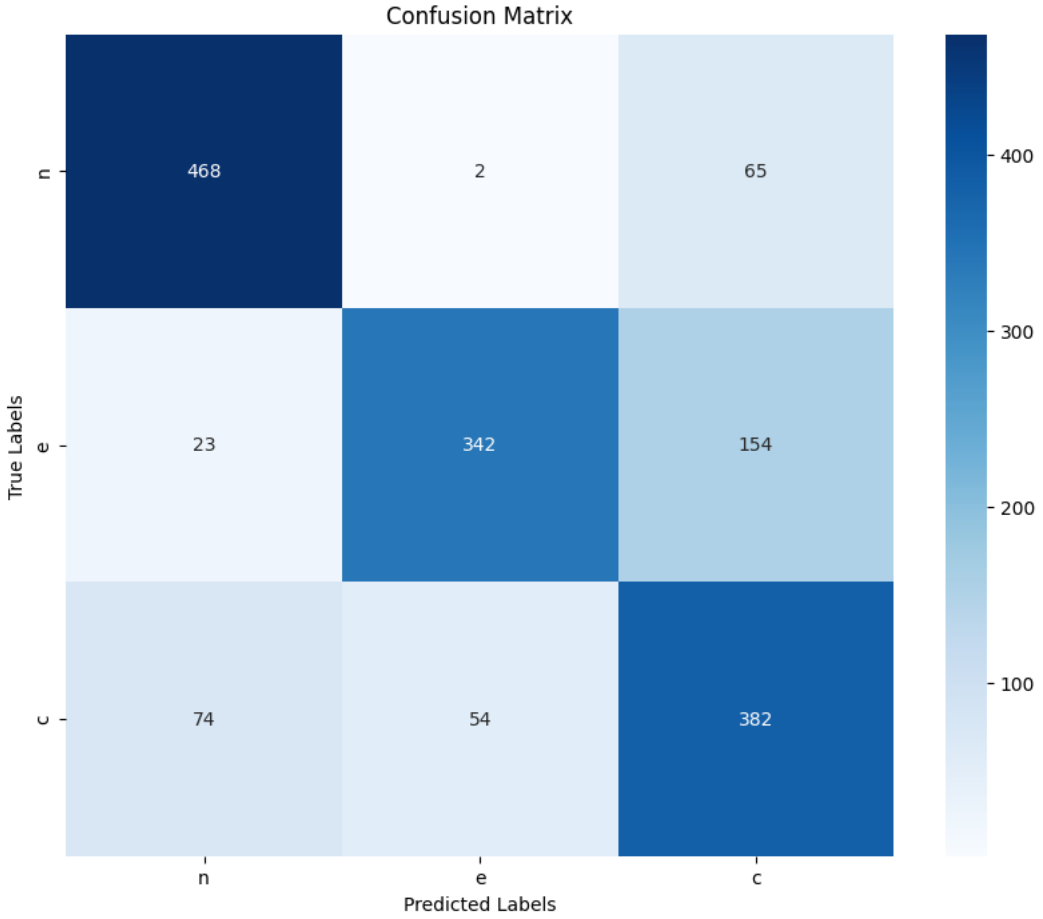
۲-۱-۴. تنظیم دقیق مدل بر لایه میانی

پس از بارگذاری، مدل با کوتاه کردن لایه‌های رمزگذار آن تنها به ۹ لایه اول اصلاح می‌شود و به طور موثر عمق آن را کاهش می‌دهد، که در تنظیمات پیکربندی مدل نیز منعکس می‌شود. این می تواند یک استراتژی برای تنظیم پیچیدگی مدل و کارایی محاسباتی باشد. هنگامی که مدل پیکربندی شد، در صورت در دسترس بودن (یا در غیر این صورت به CPU) به یک GPU منتقل می شود تا از محاسبات تسریع شده برای آموزش استفاده شود. در نهایت، اسکریپت بهینه‌ساز AdamW را با نرخ یادگیری مشخصی برای فرآیند تنظیم دقیق تنظیم می‌کند که وزن مدل را بر اساس داده‌های آموزشی که در معرض آن قرار می‌گیرد، تنظیم می‌کند.

نتایج مربوط به این پیاده سازی به صورت زیر به دست می آید:



شکل ۳۸: نمودار استفاده از ۹ لایه ابتدایی



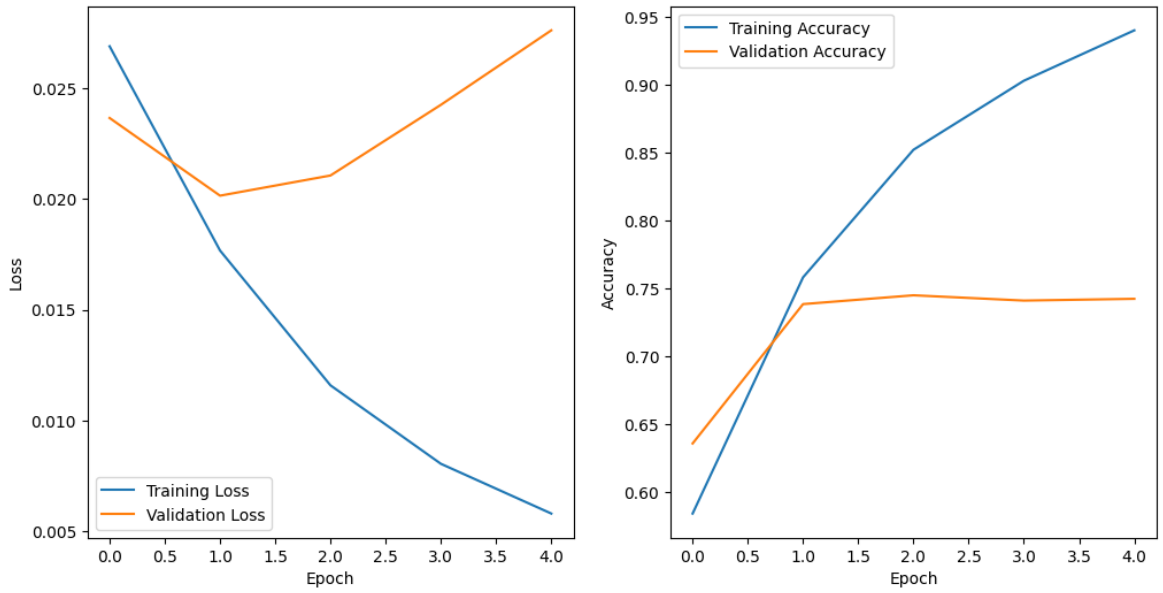
شکل ۳۹: ماتریس در هم امیختگی استفاده از ۹ لایه ابتدایی

```
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.87      0.85       535
           1       0.86      0.66      0.75       519
           2       0.64      0.75      0.69       510

    accuracy                           0.76      1564
   macro avg       0.77      0.76      0.76      1564
weighted avg       0.78      0.76      0.76      1564
```

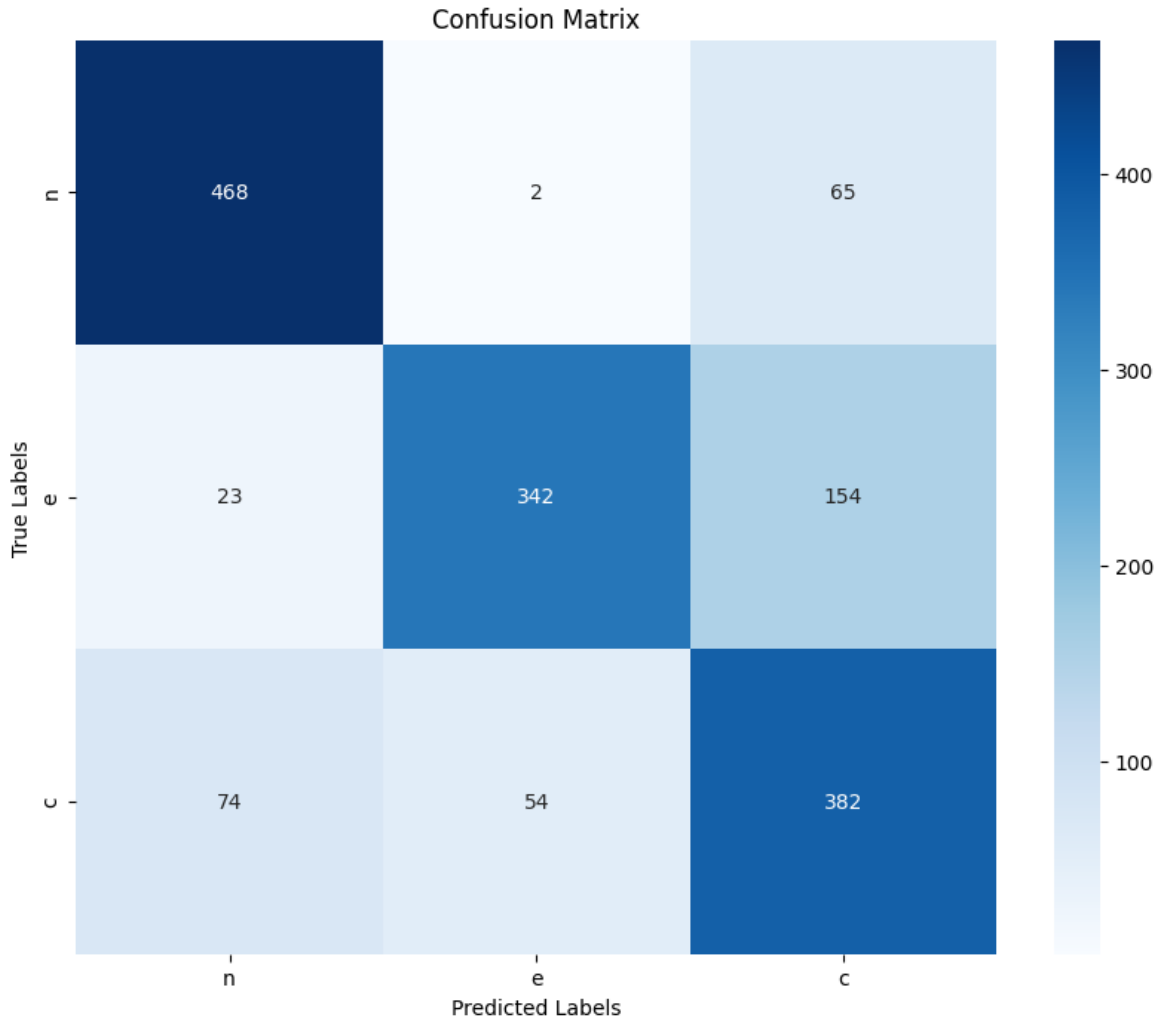
شکل ۴۰: نتایج استفاده از ۹ لایه ابتدایی

همان طور که مشاهده می شود جواب به این سوال که آیا می شود فقط از بخشی از مدل استفاده کرد بله است زیرا صحت مدل چندان دچار تغییر نمی شود.

البته اگر منظور از فریز کردن همه لایه‌ها جز لایه آخر، حفظ بخش classifier می‌باشد نتایج به صورت زیر می‌باشد:



شکل **۴۱** نمودار دقت و خطا



شکل ۴۲ ماتریس درهمریختگی

```
Classification Report:
              precision    recall  f1-score   support

           0       0.71      0.84      0.77       535
           1       0.62      0.77      0.68       519
           2       0.51      0.29      0.37       510

    accuracy                           0.64      1564
   macro avg       0.61      0.63      0.61      1564
weighted avg       0.62      0.64      0.61      1564

F1 Score: 0.6114202932964415
```

شکل **۴۳** نتایج متریک‌ها

نتایج حاصل، زیاد با fine tune کردن کل مدل فرق نداشت، دلیل آن می‌تواند به صورت زیر باشد:

قابلیت انتقال یادگیری: BERT بر روی یک مجموعه بزرگ از داده‌های زبانی برای مهمانی وظایف درک زبان پیش‌آموزش داده می‌شود. لایه‌های پایین‌تر BERT ویژگی‌های بیشتر عمومی و نحوی را دربرمی‌گیرند، در حالی که لایه‌های بالاتر اطلاعات معنایی و وظیفه‌محور بیشتری را دربرمی‌گیرند. لایه‌های اولیه ممکن است حاوی مقدار قابل توجهی از دانش قابل انتقال برای وظیفه خاص شما باشد و این باعث موفقیت در fine-tuning آنها می‌شود.

ویژگی‌های وظیفه: برخی از وظایف ممکن است نیاز به درک عمیق نداشته باشند و لایه‌های پایین BERT ممکن است کافی باشند تا ویژگی‌های اساسی وظیفه شما را دربربگیرند. اگر وظیفه شما الگوهای زبانی ساده‌تری را دربردارد، لایه‌های پایین ممکن است کافی باشند.

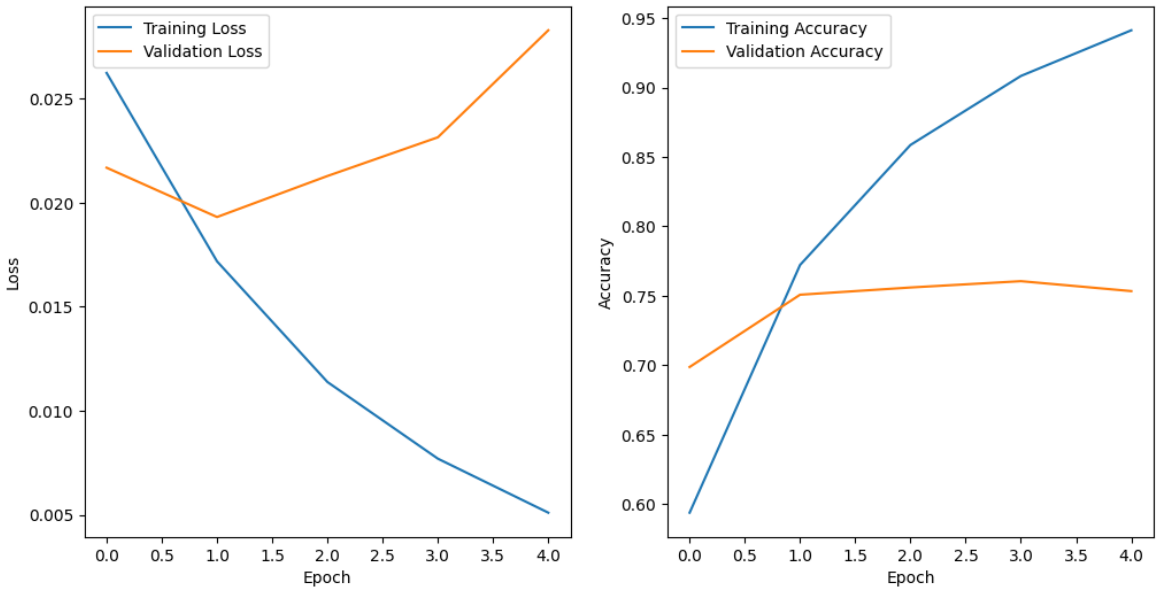
overfitting: fine-tuning قسمت کوچکتری از مدل می‌تواند خطر overfitting را کاهش دهد. با فریز کردن برخی از لایه‌ها، تعداد پارامترهایی که در طول fine-tuning به‌روزرسانی می‌شوند، محدود می‌شود و این ممکن است کمک کند تا مدل از جلوگیری از هماهنگی با نویز در داده‌های آموزش جلوگیری شود.

شباهت وظیفه به پیش‌آموزش: اگر وظیفه پایین‌جریان شما به وظایف پیش‌آموزش BERT شباهت داشته باشد، لایه‌های پایین ممکن است اطلاعات مرتبط را دربر بگیرند و fine-tuning تنها آن لایه‌ها ممکن است کافی باشد.

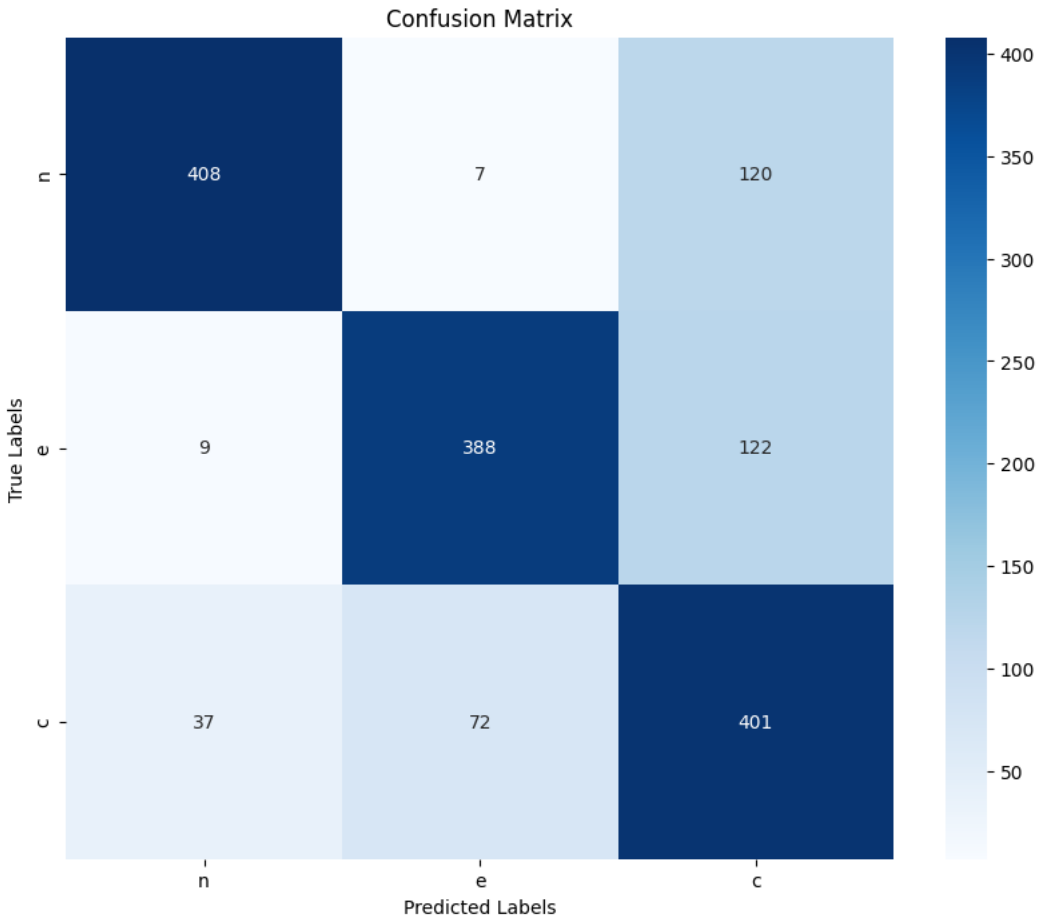
۲-۱-۵. حذف head های attention در مدل

این بخش شامل یک تابع "Prune_heads" برای کاهش پیچیدگی مدل با نصف کردن تعداد headهای موجود در لایه اول است، یک تکنیک رایج برای تمرکز مکانیسم توجه مدل و بهبود بالقوه کارایی. در نهایت، یک بهینه‌ساز را با الگوریتم AdamW و نرخ یادگیری ۲-۱۰^تنظیم می‌کند و مدل را برای فرآیند تنظیم دقیق آماده می‌کند، که شامل تنظیم وزن‌های مدل از قبل آموزش‌دیده‌شده روی یک مجموعه داده خاص برای انجام یک کار طبقه‌بندی خاص است.

در ادامه پیاده سازی نتیجه به صورت زیر به دست می آید:



شکل ۴۴: نمودار با حذف **head**



شکل ۴۵: ماتریس در هم امیختگی با حذف **head**

```
Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.76      0.83       535
           1       0.83      0.75      0.79       519
           2       0.62      0.79      0.70       510

    accuracy                           0.77      1564
   macro avg       0.78      0.77      0.77      1564
weighted avg       0.79      0.77      0.77      1564
```

شکل ۴۶: نتایج با حذف **head**

تحلیل نهایی:

در نهایت برای همه‌ی حالات جدول زیر را داریم:

جدول ۱ نتایج نهایی مدل برای حالت‌های مختلف

روش	Accuracy	F1
Fine Tune کل مدل	۰.۷۷	۰.۷۹
فریز کردن ۹ لایه اول	۰.۴۷	۰.۴۳
فریز کردن همه جز لایه‌ی آخر	۰.۶۶	۰.۶۵
استفاده فقط از ۹ لایه ابتدایی	۰.۷۶	۰.۷۹
حذف تصادفی نصف attention headها	۰.۷۷	۰.۷۵

Fine-tuning کل مدل:

Fine-tuning کل مدل، به دلیل آموزش دوباره مدل بر اساس وظیفه مورد نظر، به طور واضح بهترین عملکرد را ارائه می‌دهد. در اینجا، مدل را برای ۵ دوره آموزش داده‌ایم که ممکن است با افزایش تعداد دوره‌ها بهبود یابد.

فریز کردن ۹ لایه اول:

فریز کردن لایه‌های ابتدایی، که مهمترین لایه‌ها برای استخراج ویژگی‌های مهم هستند، باعث کاهش دقت مدل شده است. این نشان می‌دهد که این لایه‌ها نقش حیاتی در یادگیری اطلاعات دارند.

فریز کردن همه لایه‌ها با Fine-tune لایه آخر:

فریز کردن همه لایه‌ها به دلیل آموزش پیشین مدل بر روی داده‌های زیاد، و با fine-tune کردن لایه آخر، بهبود قابل توجهی در دقت مدل ایجاد شده است. این نشان‌دهنده قابلیت خوب مدل در انتقال ویژگی‌ها برای وظایف مشابه است.

استفاده از ۹ لایه ابتدایی:

استفاده از لایه‌های ابتدایی نسبت به تعداد لایه‌های کل مدل، نشان می‌دهد که این لایه‌ها برای این وظیفه خاص ممکن است کافی باشند. دقت مدل در این حالت نیز نشان‌دهنده عملکرد مطلوب است. دلایل دیگر به شرح زیر می‌باشند:

قابلیت انتقال یادگیری: BERT بر روی یک مجموعه بزرگ از داده‌های زبانی برای مهمانی وظایف درک زبان پیش‌آموزش داده می‌شود. لایه‌های پایین‌تر BERT ویژگی‌های بیشتر عمومی و نحوی را دربرمی‌گیرند، در حالی که لایه‌های بالاتر اطلاعات معنایی و وظیفه‌محور بیشتری را دربرمی‌گیرند. لایه‌های اولیه ممکن است حاوی مقدار قابل توجهی از دانش قابل انتقال برای وظیفه خاص شما باشد و این باعث موفقیت در fine-tuning آنها می‌شود.

ویژگی‌های وظیفه: برخی از وظایف ممکن است نیاز به درک عمیق نداشته باشند و لایه‌های پایین BERT ممکن است کافی باشند تا ویژگی‌های اساسی وظیفه شما را دربربگیرند. اگر وظیفه شما الگوهای زبانی ساده‌تری را دربردارد، لایه‌های پایین ممکن است کافی باشند.

overfitting: fine-tuning قسمت کوچکتری از مدل می‌تواند خطر overfitting را کاهش دهد. با فریز کردن برخی از لایه‌ها، تعداد پارامترهایی که در طول fine-tuning به‌روزرسانی می‌شوند، محدود می‌شود و این ممکن است کمک کند تا مدل از جلوگیری از هماهنگی با نویز در داده‌های آموزش جلوگیری شود.

شباهت وظیفه به پیش‌آموزش: اگر وظیفه پایین‌جریان شما به وظایف پیش‌آموزش BERT شباهت داشته باشد، لایه‌های پایین ممکن است اطلاعات مرتبط را دربر بگیرند و fine-tuning تنها آن لایه‌ها ممکن است کافی باشد.

حذف attention headها:

حذف تصادفی attention headها، با توجه به تعداد لایه‌ها و attention headها، بهبود دقت مدل را نشان می‌دهد. این کار نشان‌دهنده این است که برخی از attention headها اطلاعات تکراری داشته‌اند و حذف آن‌ها به بهبود کلی مدل کمک کرده است.