مقدمه

در این پروژه هدف بررسی عملکرد یک مدل یادگیری ماشین برای دستهبندی نظرات کاربران بوده است. نظرات کاربران به سه دسته مختلف تعلق دارند که مدل باید با توجه به محتوای متن، دسته بندی مناسبی انجام دهد. هدف از این گزارش، ارائه روند آموزش مدل، تحلیل عملکرد، و بررسی دقت پیشبینی روی داده های تست است.برای این کار از روش های مرسوم یادگیری عمیق کمک گرفته شد و در نهایت با معیارهایی مثل دقت، F1 و گزارش طبقهبندی عملکرد مدل بررسی شده است.

پیش‌پردازش داده‌ها

بخش پیش‌پردازش یکی از مراحل مهم در آماده‌سازی داده‌ها برای مدل‌های یادگیری ماشین است، خصوصاً زمانی که داده های متنی به زبان فارسی مورد استفاده قرار می‌گیرند .در این پروژه، مجموعه‌ای از کامنت های کاربران به زبان فارسی در اختیار بود که باید برای مدل طبقه‌بندی آماده می‌شد .در این بخش، مراحل مختلف پیش‌پردازش به طور مفصل توضیح داده می‌شوند:

۱. حذف نویزهای متنی:

در ابتدا باید کاراکترهای غیرمفید، نمادهای نگارشی، اعداد، لینک‌ها، ایموجی‌ها و کاراکترهای تکراری حذف شوند .این موارد نه تنها به مدل کمک نمی‌کنند، بلکه ممکن است باعث ایجاد نویز در فرآیند یادگیری شوند.

۲. Stop Words:

کلمات بسیار رایجی مثل" از"،" برای"،" است "و ... در بسیاری از مواقع معنی خاصی به مدل نمی‌دهند .این کلمات در یک لیست از پیش‌تعریف‌شده جمع‌آوری و از متن حذف شدند.

۳. نرمال‌سازی:

متون فارسی معمولاً شامل شکل‌های مختلفی از یک حرف یا کلمه هستند .مثلاً" ی "و" ی"، یا" ک "و" ک ."نرمال سازی این کلمات به فرمت یکسان، باعث می‌شود مدل بتواند بهتر الگوها را تشخیص دهد.

۴. Tokenization:

جملات یا متون به کلمات جداگانه شکسته می‌شوند تا بتوان آن‌ها را به مدل داد .این فرآیند شامل جدا کردن واژه‌ها بر اساس فاصله و سایر قواعد زبانی است.

۵. Token Indexing:

برای استفاده از متون در شبکه‌های عصبی، باید آن‌ها را به فرمت عددی تبدیل کرد .برای این کار از یک vocab استفاده شد که به هر کلمه یک عدد اختصاص داده شود.

۶. Padding:

برای اینکه همه ورودی‌ها اندازه یکسان داشته باشند، جملات کوتاه‌تر با صفر پر شدند . جملات بزرگ تر کوتاه تر شدند و برای اینکه اندازه درستی داشته باشیم توزیع طول جملات بررسی شد و با دقت خوبی عدد ۱۲۳ برای padding انتخاب شد.

همچنین به دلیل عدم توازن بین داده های کلاسها در هنگام آموزش مدل برای اینکه کلاسهایی که نمونه های کمتری دارند باعث آسیب به مدل نشوند، در تابع خطا وزن بیشتری در نظر گرفته شد.
این اقدامات باعث شد داده‌های متنی خام به شکلی تبدیل شوند که بتوان آن‌ها را به شبکه عصبی برای یادگیری و پیش بینی داد. بعد از آن نیز داده‌ها به کمک StratifiedShuffleSplit به دو مجموعه آموزش و آزمون با نسبت ۲۵/۷۵ تقسیم شدند تا توازن کلاس‌ها حفظ شود.

ساختار مدل

مدل طراحی‌شده برای دسته‌بندی نظرات کاربران از چند بخش اصلی تشکیل شده که هرکدام نقش مشخصی در فرآیند یادگیری و تصمیم‌گیری ایفا می‌کنند. در ادامه، اجزای این ساختار توضیح داده میشوند:

لایه‌ی Embedding

در نخستین مرحله، از یک لایه embedding استفاده شده‌است تا کلمات ورودی که به‌صورت توکن‌های عددی نمایش داده می‌شوند، به بردار قابل یادگیری تبدیل شوند. این تبدیل به مدل کمک می‌کند تا اطلاعات معنایی کلمات را بهتر درک کند و روابط معنایی میان آن‌ها را در فضای برداری لحاظ نماید. به‌عنوان نمونه، در صورتی که مدل به‌درستی آموزش دیده باشد، انتظار می‌رود بردار کلمه‌ی «عالی» به «کیفیت» نزدیک‌تر از «خراب» باشد.

لایه‌ی LSTM دوطرفه(Bidirectional LSTM)

خروجی لایه embedding به یک LSTM دوطرفه منتقل می‌شود (Long Short-Term Memory) LSTM . یکی از انواع شبکه‌های بازگشتی است که توانایی مدل‌سازی وابستگی‌های بلندمدت در توالی داده‌ها را دارد. نسخه‌ی به‌کاررفته در این مدل به‌صورت دوطرفه طراحی شده‌است، به این معنا که جمله‌ها هم از ابتدا به انتها و هم از انتها به ابتدا خوانده می‌شوند. این ویژگی به مدل امکان می‌دهد که context کلی جمله را با دقت بیشتری در نظر بگیرد.

لایه‌ی Attention

پس از پردازش توسط LSTM ، از یک لایه attention برای تمرکز بیشتر بر بخش‌های مهم‌تر جمله استفاده شده‌است. این لایه با تخصیص وزن‌های متفاوت به خروجی‌های LSTM ، به مدل اجازه می‌دهد تا اطلاعات کلیدی و تأثیرگذار بر دسته‌بندی نهایی را برجسته کند. به‌عنوان مثال، در جمله‌ای مانند «این لپ‌تاپ طراحی خوبی دارد اما کیفیت صفحه‌نمایش پایین است»، لایه attention می‌تواند با وزن‌دهی بیشتر به بخش دوم جمله، تصمیم دقیق‌تری را امکان‌پذیر کند.

لایه Fully Connected

خروجی حاصل از attention به یک لایه fully connected منتقل می‌شود. این لایه نقش طبقه‌بندی‌کننده نهایی را بر عهده دارد و خروجی آن شامل تعداد نورون‌هایی برابر با تعداد کلاس‌های دسته‌بندی (در اینجا سه کلاس) است. این لایه بر اساس داده های لایه های قبلی، خروجی نهایی مدل را برای پیش‌بینی کلاس هر نمونه تولید می‌کند.

تابع Softmax

در انتهای مدل، از تابع softmax استفاده شده‌است تا خروجی لایه fully connected به احتمال تعلق هر نمونه به یکی از کلاس‌ها تبدیل شود. این تابع، خروجی مدل را به یک توزیع احتمالی نرمال‌سازی می‌کند که برای تصمیم‌گیری در دسته‌بندی ضروری است.

تنظیمات آموزش

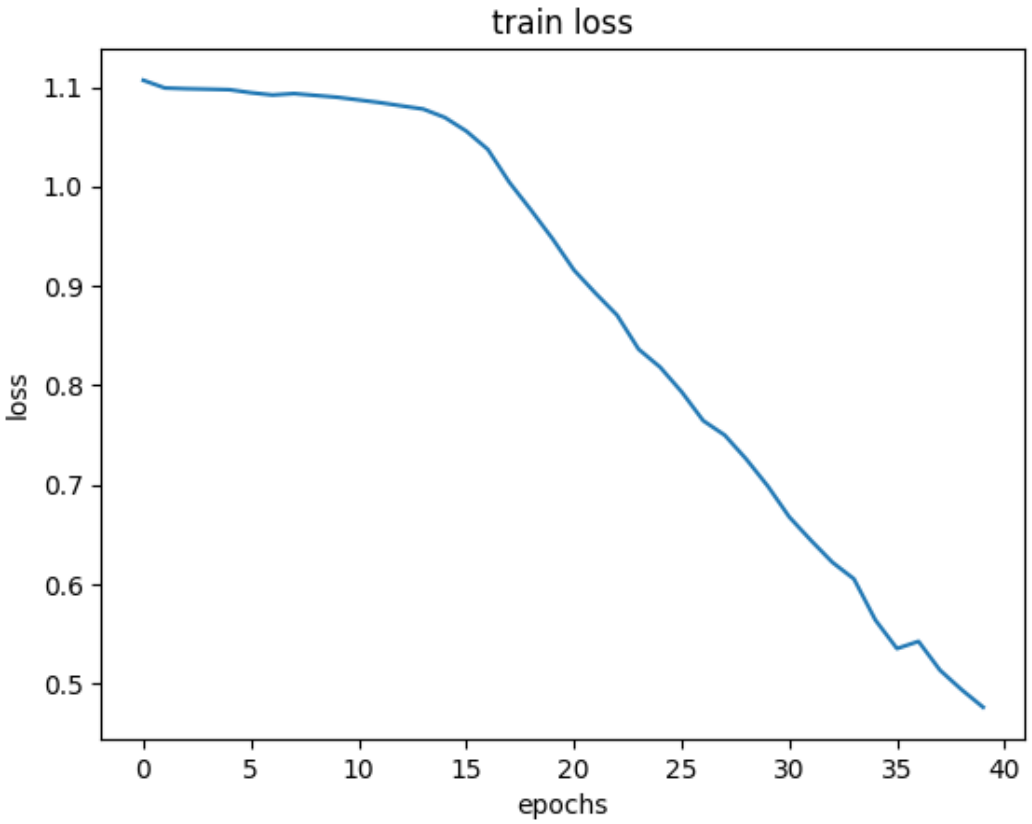
بهینه‌ساز: در فرآیند یادگیری، از بهینه‌ساز Adam استفاده شده تا نرخ یادگیری را بهینه کرده و سرعت همگرایی مدل را افزایش می‌دهد. برای نرخ یادگیری اولیه نیز مقدار ۰,۰۰۰۲ قرار گذاشته شد تا به دلیل داده کم باعث overfit شدن سریع مدل نشود

تابع خطا: برای آموزش مدل، از Cross Entropy Loss به‌عنوان تابع هزینه استفاده شده‌است. این تابع برای مسائل چندکلاسه مناسب بوده و تفاوت میان پیش‌بینی و برچسب واقعی را به‌خوبی اندازه‌گیری می‌کند. که همان طور که در بخش پیش پردازش گفته شد به دلیل عدم توازن در برچسب داده ها از یک وزن برای کلاس های استفاده شد.

EarlyStopping: به‌منظور جلوگیری از بیش‌برازش(overfitting) ، از تکنیک EarlyStopping استفاده شده‌است. در این روش، در صورتی که عملکرد مدل روی داده‌ی آموزش در طول چند دوره‌ی متوالی بهبود نیابد، روند آموزش متوقف می‌شود تا از افت دقت مدل روی داده‌های نادیده گرفته‌شده جلوگیری شود.

آموزش مدل

مدل در طی ۴۰ دوره (epoch) آموزش داده شده است. روند کاهش خطای آموزش (train loss) به مرور زمان کاهش محسوسی داشته که نشان‌دهنده یادگیری تدریجی مدل از داده‌هاست. مقدار loss از حدود ۱/۱۰۷در دوره اول، به حدود ۰/۴۷۶در پایان دوره چهل‌ام کاهش یافته است. این موضوع در نمودار نیز به‌وضوح دیده می‌شود که خطا با شیب مناسبی کم شده است و احتمالاً مدل دچار overfitting نشده.



نتایج نهایی

برای بررسی عملکرد مدل، از معیارهای مرسوم مانند دقت(Accuracy) ، دقت طبقه‌بندی(Precision) ، بازخوانی (Recall)و F1-Score استفاده شده است.

مقادیر به‌دست‌آمده برای کل داده‌های تست به شرح زیر است:

Accuracy: 0.674
Precision: 0.697
Recall: 0.674
F1 Score: 0.682

همچنین گزارش طبقه‌بندی (Classification Report) نیز نشان می‌دهد که مدل در دسته ۰ عملکرد بسیار خوبی داشته. اما در مورد دسته ۱ و ۲، مدل هنوز با ضعف‌هایی روبه‌رو است. که یکی از دلایل آن تعداد داده کم و عدن توازن بین داده های کلاس های مختلف است.

Class	Precision	Recall	F1-Score	Support
1	0.83	0.78	0.81	596
2	0.3	0.26	0.28	105
3	0.35	0.51	0.41	115

نمونه‌هایی از پیش‌بینی‌های مدل نیز بررسی شده‌اند. در مواردی مدل توانسته برچسب درست را با دقت خوبی تشخیص دهد، به‌ویژه در متونی که نشانه‌های معنایی قوی و عبارات پرتکرار داشته‌اند. با این حال، در برخی نمونه‌ها مانند نظرات کوتاه یا جملاتی با ساختار مبهم، مدل دچار خطا شده و پیش‌بینی اشتباه داشته است. به نظر می‌رسد یکی از دلایل این مسئله، شباهت واژگانی بین کلاس‌ها و عدم وجود سیگنال‌های معنایی قوی در بعضی نظرات باشد.

پیشنهادات برای بهبود

افزایش حجم داده آموزشی :با توجه به اینکه مدل در دسته‌ی «کلاس ۱ و ۲» عملکرد ضعیف‌تری نسبت به کلاس ۰ دارد، می‌توان نمونه‌های بیشتری برای این کلاس‌ها جمع‌آوری کرد تا مدل نسبت به آن‌ها بهتر آموزش ببیند.

متعادل‌سازی داده‌ها: توزیع نامتوازن بین کلاس‌ها باعث کاهش دقت در کلاس‌های دارای نمونه کمتر شده‌است. استفاده از روش‌هایی مثل oversampling (SMOTE) یا undersampling می‌تواند به تعادل کلاس‌ها کمک کند.

استفاده از مدل‌های از پیش‌آموزش‌دیده:

بهره‌گیری از مدل‌های فارسی مانند ParsBERT یا FastText می‌تواند به‌طور قابل توجهی عملکرد را بهبود بخشد، چون این مدل‌ها قبلاً روی حجم عظیمی از متون فارسی آموزش دیده‌اند و دانش زبانی قوی‌تری دارند.

افزایش عمق شبکه:

افزایش پیچیدگی مدل با لایه‌های بیشتر (با احتیاط) می‌تواند قابلیت یادگیری الگوهای پیچیده‌تر را فراهم کند. اما این در صورتی است که تعداد داده قابل توجه ای دردسترس باشد