پیش‌بینی برچسب ایشوها با استفاده از راهکارهای مبتنی بر یادگیری

امیرحسین کارگران خوزانی [۱] و زینب صادقیان [۲]

[۱] دانشجوی ارشد مهندسی کامپیوتر، نرم‌افزار، دانشگاه صنعتی شریف، تهران، kargaran@sharif.edu

[۲] دانشجوی کارشناسی ارشد مهندسی کامپیوتر، شبکه‌های کامپیوتری، دانشگاه صنعتی شریف، تهران، zeinab.sadeghian@sharif.edu

چکیده

امروزه توسعه‌دهندگان به منظور توسعه بهتر نرم‌افزارها از سامانه‌های کنترل نسخه، مدیریت وظیفه‌ها و ... استفاده می‌کنند. یکی از ویژگی‌هایی که چنین سامانه‌هایی در اختیار توسعه‌دهندگان قرار می‌دهد یک سامانه ردیابی ایشوها است. از انجا که نرم‌افزارهای پرکاربرد تعداد زیادی از ایشوها را دریافت می‌کنند طبقه‌بندی و برچسب زدن این ایشوها به مدیریت بهتر آن‌ها کمک می‌کند. در این گزارش چند راهکار برای پیش‌بینی برچسب ایشوها ارائه می‌شود که این روش‌ها منطبق بر کارهای گذشته و جدیدترین پیشرفت‌ها در حوزه پردازش زبان طبیعی است.

کلمات کلیدی

پیش‌بینی برچسب ایشو، طبقه‌بندی ایشوها، تحلیل احساس ایشو، پردازش زبان طبیعی، مهندسی نرم‌افزار هوشمند.

۱ مقدمه

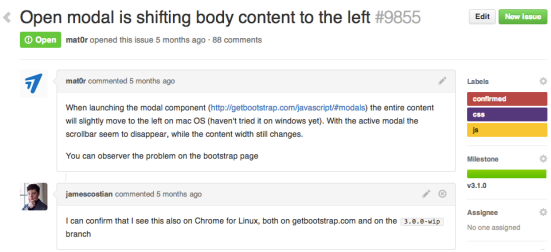
در ادبیات پروژه‌های نرم‌افزاری، ایشو [۱] به یک درخواست از سمت کاربر گفته می‌شود که می‌تواند در مورد مشکلات نرم‌افزاری و امنیتی، درخواست برای اضافه شده یک ویژگی [۲] جدید، یا سوال و درخواست برای مستندسازی باشد. بیشتر ایشوها توسط افراد که کاربران یا توسعه‌دهندگان آن نرم‌افزار هستند نوشته می‌شوند. یک ایشو استاندارد شامل عنوان و توضیحات است. سامانه‌هایی همچون گیت‌هاب [۳] و جیرا [۴] نیز به منظور توسعه بهتر نرم‌افزار و پیگیری مشکلات و سوالات یک سامانه مدیریت ایشو نیز به هر پروژه تخصیص می‌دهند. همچنین این سامانه‌ها به منظور مدیریت آسان‌تر ایشوها امکانات دیگری را نیز فراهم می‌کنند. از جمله‌ی این امکانات می‌توان به امکان برچسب‌گذاری، امکان نظردهی پیرامون بحث توسط کاربران و امکان تخصیص افراد به ایشوها اشاره کرد.

در سال‌های اخیر با گسترش نرم‌افزارهای متن‌باز [۵] ، استفاده از سامانه‌های کنترل نسخه نیز افزایش یافته است. این افزایش به حدی است که تعداد کاربران سایت گیت‌هاب بیش از ۳۷ میلیون و تعداد مخازن کد [۶] آن حدود ۱۰۰ میلیون مخزن کد تخمین زده شده است. در بیشتر مخازن کد، ایشوها توسط توسعه‌دهندگان هسته‌ی اصلی توسعه‌ی نرم‌افزار تولید می‌شود [۱]. از طرفی دیگر کاربران به مشکلات بیشتری برخورد می‌کنند و یا درخواست ویژگی‌های جدید بیشتری را ارائه می‌دهند. همین موضوع موجب می‌شود انبوهی از این درخواست‌ها، تحت عنوان ایشو، به سمت تیم توسعه‌دهنده‌ی نرم‌افزارها فرستاده شود. این ایشوها ممکن است هیچ‌وقت خوانده نشوند یا تنها یکبار دیده شوند. نکته مهمتر نیز آن است که با رشد پروژه، تعداد کاربران و گزارش‌های مشکلات نیز افزایش می یابد و رسیدگی به ایشوها از قبل بیشتر چالش خواهد داشت.

ایشوها، به عنوان نوعی فراداده‌ی پروژه، هدف و محتوای یک موضوع را توصیف می‌کنند و عمدتا برای دسته بندی، بررسی، مدیریت، جستجو و بازیابی مشکلات استفاده می‌شوند. بنابراین، اختصاص برچسب به ایشوها واگذاری وظایف، نگهداری و مدیریت یک پروژه نرم افزاری را تسهیل می‌کند. در نتیجه اختصاص برچسب به مشکلات بخش مهمی از فرآیند توسعه نرم افزار است.

برچسب زدن ایشوها می‌تواند هم برای توسعه‌دهندگان و هم برای کاربران مفید باشد؛ برای توسعه‌دهندگان از این جهت که می‌توانند کارها را طبقه‌بندی کنند کاربران نیز می‌توانند موردی که مدنظر آن‌هاست را در ایشوها جستجو کنند و برای مثال در سوالات یا مشکلات مشابه مواردی که مدنظرشان هست را پیدا کنند. از طرف دیگر تعیین برچسب ایشوها باعث می‌شود که ایشوها زودتر پاسخ داده شود که موجب رضایت هر دو طرف خواهد شد. در شکل ۱ مثالی از یک ایشو در گیت‌هاب را مشاهده می‌کنید.

با وجود ویژگی اضافه کردن برچسب ایشو توسط کاربر یا توسعه دهنده هنوز



شکل ۱: مثالی از یک ایشو در گیت‌هاب [۲].

بسیاری از ایشوها حتی در معروف‌ترین پروژه‌های سایت گیت‌هاب برچسب ندارند [۳]. پس ایجاد یک راه‌حل خودکار برای زدن برچسب می‌تواند به حل بسیاری از مشکلات کمک کند. با توجه به اهمیت مساله پیش‌بینی برچسب ایشوها در این گزارش چندین روش داده محور ارائه شده است و همچنین این روش‌ها با یکدیگر از جهات مختلف مقایسه شده‌است. همگی روش‌های ارائه شده، از جمله روش‌های بروز و معتبر در حوزه پردازش زبان طبیعی هستند. مجموعه دادگان این روش‌ها نیز، یک مجموعه ۴۵ هزار تایی از ایشوهای سایت گیت‌هاب است که ۳ برچسب باگ [۷] بهبود [۸] یا سوال را دارد.

در ادامه این گزارش، در بخش ۲ با توجه به کارهای پیشین و پیشرفت‌های حوزه پردازش زبان طبیعی رویکردهای مناسبی برای حل مساله پیش‌بینی برچسب ایشوها ارائه می‌شود. در این قسمت دلیل انتخاب هریک از روش‌ها نیز توضیح داده شده است. در بخش ۳ نیز جنبه‌های عملی پیاده‌سازی هر کدام از روش‌ها ارائه می‌شود. در این بخش به مجموعه دادگان، خواص آن و کتابخانه‌های پیاده سازی نیز اشاره می‌شود. در بخش ۴ معیارهای مقایسه روش‌ها معرفی می‌شود و روش‌های پیاده‌سازی شده نیز با توجه به این معیارها مورد بررسی قرار می‌گیرد. همچنین در بخش ۵ به تهدیدات علیه اعتبار روش‌های بیان شده پرداخته می‌شود و در نهایت در بخش ۶ مطالب گفته شده در این گزارش جمع‌بندی می‌شود.

۲ معرفی رویکردهای پیشنهادی

۲-۱ روش‌های یادگیری ماشین سنتی

در این روش‌ها از وجود و عدم وجود کلمات به عنوان بازنمایی [۹] استفاده می‌شود. به همین دلیل انتخاب کلمات مهم در این الگوریتم‌ها نقش کلیدی دارد. از این رو روش‌های مختلفی برای انتخاب کلمات مهم وجود دارد که یکی از بهترین این روش‌ها، فراوانی وزن‌دار کلمات یا تی‌اف-آی‌دی‌اف [۱۰] است. از طرفی برای دسته‌بندی کلمات نیز روش‌های دسته‌بندی مختلفی وجود دارد. نمونه‌هایی از این روش ها عبارتند از: ۱) رگرسیون لجستیک [۱۱] ۲) بردار ماشین پشتیبان [۱۲].

به این ترتیب می‌توان به ازای هر ایشو یک بازنمایی با استفاده از تی‌اف-آی‌دی‌اف استخراج کرد که این بازنمایی بوسیله یکی از روش‌های دسته‌بندی می‌تواند برچسب ایشو را پیش‌بینی کند. این روش از جمله روش‌های معروف در بحث تحلیل احساس متن‌هاست [۴، ۵]. از آنجا که این مساله نیز شبیه مساله تحلیل احساس است بنابراین این روش می‌تواند به خوبی جواب دهد. در ادامه به طور مختصر هر یک از اجزای این روش توضیح داده می‌شود.

۲-۱-۱ تی‌اف-آی‌دی‌اف

تی‌اف-آی‌دی‌اف یا فراوانی وزن دار یک روش برای به دست آوردن کلمات مهم در پردازش متن‌هاست. در این روش بازنمایی یک کلمه صرفا به میزان تکرار یک کلمه توجه نمی‌کند. بلکه هدف آن به دست آوردن اهمیت فراوانی با توجه به بقیه مستندهاست. این کار را از طریق مقایسه تعداد تکرار هر کلمه در متن با تکرار آن در مجموعه‌ای بزرگ‌تر از مستندها انجام می‌شود.

تی‌اف-آی‌دی‌اف از دو عبارت تی‌اف به معنای فرکانس لغت و آی‌دی‌اف که معکوس فرکانس لغت در مستندات است، تشکیل شده است. برای محاسبه تی‌اف-آی‌دی‌اف باید هر یک از دو عبارت را به صورت جداگانه محاسبه کرد. سپس طبق رابطه ۱ میانگین وزن دار کلمه x در سند y محاسبه می‌گردد.

$$\mathcal{W}_{x,y} = tf_{x,y} \log \frac{\mathcal{N}}{df_x} \quad (۱)$$

در این رابطه df_x تعداد مستندهایی است که کلمه x در آن‌ها وجود دارد. $tf_{x,y}$ تعداد تکرار کلمه x در سند y است و $\mathcal{N}$ تعداد کل مستندهاست.

۲-۱-۲ رگرسیون لجستیک

در دنیای یادگیری ماشین رگرسیون لجستیک یک مدل دسته‌بند پارامتری [۱۳] است. این به این معنی است مدل رگرسیون لجستیک به تعداد مشخصی پارامتر دارد و این تعداد وابسته به تعداد ویژگی‌های ورودی است. در رگرسیون لجستیک یک خط مستقیم به داده‌ها برازش نمی‌شود. به جای آن از یک خم استفاده می‌شود که به آن تابع سیگموید [۱۴] گفته می‌شود. با استفاده از تابع سیگموید مقدار احتمال تعلق هریک از دسته‌ها را می‌توان محاسبه کرد.

۲-۱-۳ ماشین بردار پشتیبان

ماشین بردار پشتیبان یک مدل خطی برای مسائل رگرسیون و دسته‌بندی است. این مدل می‌تواند مسائل خطی و غیرخطی را حل کند و برای بسیاری از مسائل کاربردی استفاده می‌شود. در کل ماشین بردار پشتیبان ایده‌ی ساده‌ای دارد به این صورت که یک خط یا یک ابرصفحه ایجاد می‌کند که داده‌ها را به کلاس‌ها دسته‌بندی می‌کند. طبق الگوریتم ماشین بردار پشتیبان نقاطی که به خط جداکننده نسبت به هر دو کلاس نزدیک‌ترین هستند به عنوان بردارهای پشتیبان انتخاب می‌شوند. اکنون فاصله‌ی بین خط و این بردارها محاسبه می‌شود. هدف ماکسیمم کردن این فاصله است. ابرصفحه‌ای که این فاصله برایش ماکسیمم است ابرصفحه‌ی بهینه است.

اگر داده‌ها نیز به صورت خطی قابل تفکیک نبودند می‌توان با استفاده از یک کرنل [۱۵] آن‌ها را به فضایی با ابعاد دیگر برد که در آن خطی تفکیک می‌شوند. سپس مرز به دست آمده را با تبدیلات ریاضی به ابعاد اولیه بازگرداند.

۲-۲ روش‌های شبکه عصبی

در این روش‌ها از دو شبکه مرسوم شبکه‌های عصبی حافظه‌دار و کانولوشنی [۱۶] استفاده می‌شود. هر دو این مدل‌ها از جمله مدل‌هایی هستند که در پردازش زبان طبیعی بسیار استفاده می‌شوند. به خصوص این مدل‌ها برای دسته‌بندی متن و تحلیل احساس مناسب هستند [۶].

در این روش در ابتدا با استفاده از روش جاسازی [۱۷] کلمات توسط مدل فست-تکست [۱۸][۷] یک تعبیه به ازای متن‌های داده شده در لایه اول یکی از مدل‌های

نام برده قرار داده می‌شود. سپس مدل با استفاده از متن‌ها شروع به آموزش دیدن می‌کند تا به دقت مطلوب برسد. دلیل استفاده از تعبیه کلمات آن است که ارتباطات معنایی کلمات در مدل فست-تکست بر روی متن بزرگی از دادگان آموزش داده شده و این معنا را می‌توان در این مدل‌ها استفاده کرد و باعث بهبود دقت شد. همچنین در کارهای گذشته نیز از چنین مدل‌هایی به منظور حل مساله پیش‌بینی برچسب ایشوها کمک گرفته شده است [۸، ۹]. در ادامه هر یک از اجزای این روش به طور مختصر توضیح داده خواهد شد.

۲-۲-۱ جاسازی کلمات با مدل فست-تکست

فست-تکست یک روش تعبیه کلمات است که در حقیقت گسترش‌یافته‌ی مدل وردتووک [۱۹] [۱۰] است. در این روش به جای یادگیری مستقیم بردارها برای کلمات، فست-تکست هر کلمه را به صورت ان-گرامی [۲۰] از کاراکترها در نظر می‌گیرد. در این صورت باعث می‌شود برای زیر کلمات نیز معانی‌ای به دست آید و جاسازی‌های پیشوندها و پسوندها را درک کند. بدین ترتیب فست-تکست می‌تواند اگر کلمه‌ای را به هنگام آموزش ندیده باشد با شکستن به ان-گرام‌ها برای آن جاسازی مناسبی بدست بیاورد.

۲-۲-۲ شبکه‌های عصبی حافظه‌دار

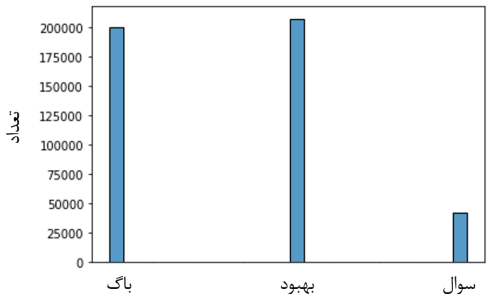
شبکه های مرسوم و مورد استفاده در مسائل پردازش زبان طبیعی شبکه های عصبی حافظه‌دار هستند. ایده ی اصلی شکل گیری و استفاده از این شبکه ها در نظر گرفتن ترتیب و اضافه کردن عاملی به نام زمان بوده است [۱۱]. به همین دلیل استفاده از این نوع شبکه های عصبی در مسائل پردازش زبان طبیعی پرطرفدار شده است. شده.از دیگر مزیت های این روش کم کردن پارامتر های یادگیری است؛ به این صورت که سلول های متفاوت در یک سری زمانی پارامترها و وزن‌ها را با یک دیگر به اشتراک می‌گذارند که باعث کم شدن محاسبات مورد نیاز می‌شود و هم چنین به استخراج ارتباط در بین نمونه ها در زمان های مختلف کمک می‌کند. در بخش پیاده‌سازی جزییات شبکه پیاده شده به طور مفصل‌تر مورد بررسی قرار می‌گیرد.

۲-۲-۳ شبکه‌های کانولوشنی

شبکه های کانولوشنی در ابتدا برای پردازش تصویر معرفی شدند. دلیل استفاده از این نوع شبکه ها کاهش خیره کننده تعداد پارامترهایی بود که باید آموزش داده می‌شدند. این کاهش تعداد پارامتر ها فقط به خاطر فرضی بود که برای پردازش پیکسل ها وجود داشت و آن فرض این بود که هر پیکسل فقط با پیکسل های اطرافش مرتبط است. از این رو استفاده از شبکه های کانولوشنی برای پردازش تصویر کاربرد بسیاری یافت. همچنین در سال های اخیر استفاده از معماری خاصی از این شبکه‌ها توانسته در دسته بندی متون نیز نتایج قابل توجه ای را از خود نشان دهد [۱۲]. در بخش پیاده‌سازی جزییات شبکه پیاده شده به طور مفصل‌تر مورد بررسی قرار می‌گیرد.

۲-۳ شبکه‌های مبتنی بر ترنسفورمرها

ترنسفورمرها [۲۱] دسته‌ای از مدل‌های یادگیری عمیق هستند که هر جزء خروجی به هر جزء ورودی متصل است و وزن‌های بین‌ها به صورت پویا بین اتصالات آن‌ها محاسبه می‌شود. در پردازش زبان طبیعی به این ویژگی توجه [۲۲] گفته می‌شود [۱۳]. یکی از معروف‌ترین مدل‌های بر پایه ترنسفورمرها برت [۲۳] نام دارد [۱۴].



شکل ۲: توزیع برچسب دادگان

برت اولین مدل پردازش زبان طبیعی است که به طور تنها از مکانیزم توجه در مدل خود استفاده کرده است.برت با دوطرفه خوانی معنی اضافه شده هر کلمه را محاسبه می‌کند و تاثیر تمامی کلمات داخل یک جمله مورد توجه قرار می‌گیرد. همچنین از روی برت نیز انوع مختلف مدل‌ها دیگر مانند روبرتا [۲۴] نیز توسعه یافته است که در بیشتر وظایف پردازش زبان طبیعی بهترین دقت‌ها را کسب کرده‌اند [۱۵]. از جمله این وظایف می‌توان به وظیفه طبقه‌بندی متن‌ها اشاره کرد. همچنین در کارهای گذشته نیز از چنین مدل‌هایی به منظور حل مساله پیش‌بینی برچسب ایشوها کمک گرفته شده است [۳].

۳ پیاده‌سازی رویکردهای پیشنهادی

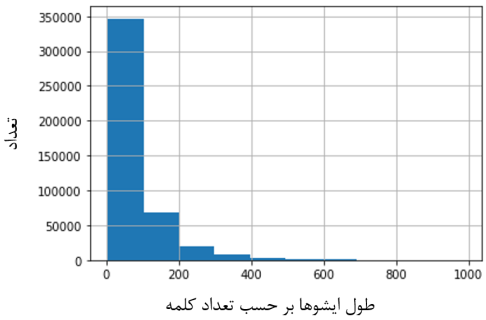
در این بخش ابتدا مجموعه دادگان معرفی می‌شود. سپس اقداماتی که برای پیش‌پردازش متن صورت گرفته است بیان می‌شود و درجات پیش‌پردازش معرفی می‌شود. در نهایت جزییات پیاده سازی مدل‌های معرفی شده در بخش پیشین نیز بیان می‌شود. پیاده سازی کل این قسمت با زبان پایتون ۳ [۲۵] انجام شده است.

۳-۱ مجموعه دادگان

مجموعه دادگان مورد استفاده ۴۵ هزار ایشو است که از سایت گیت‌هاب بارگیری شده است. این مجموعه داده به صورت یک فایل جیسون [۲۶] قابل دسترس است. با استفاده از کتابخانه پانداز [۲۷] می‌تواند این فایل جیسون را به شکل یک چارچوب داده [۲۸] مورد استفاده قرار داد. هر سطر از این چارچوب داده نشان دهنده یک ایشو است. ستون‌های این چارچوب داده عبارتند از: ۱) عنوان ایشو، ۲) متن ایشو و ۳) برچسب ایشو. این مجموعه داده توسط سایت ماشین هک [۲۹] که یک سکوی آنلاین معروف انجام مسابقات، استخدام و انجام ارزیابی است جمع شده است. این دادگان از طریق این لینک قابل دسترس هستند. برچسب‌های هر ایشو در یکی از ۳ دسته باگ (نماد ۰)، بهبود (نماد ۱) یا سوال (نماد ۲) قرار می‌گیرد. در شکل ۲ توزیع برچسب‌های این مجموعه داده مشاهده می‌شود. در این مجموعه داده تعداد سوالات از بقیه دسته‌ها کمتر است اما توزیع دو دسته دیگر به یکدیگر نزدیک‌تر است. همچنین در شکل ۳ نیز توزیع فراوانی ایشوها از منظر تعداد کلمه نیز بررسی شده است. همانجور که مشاهده می‌شود بیشتر ایشوها کوتاه و شامل ۰ تا ۲۰۰ کلمه هستند.

۳-۲ پیش‌پردازش

ابتدا متن هر ایشو با عنوان آن به یکدیگر چسبانده می‌شود. این کار به این دلیل است که کل فرآیند آموزش مدل را ساده می‌کند و این که با استفاده از تحلیل‌های



شکل ۳: توزیع طول ایشوها بر حسب تعداد کلمات

اولیه‌ای که انجام شد مشخص شد یک روش جداگانه برای هرکدام از این متن‌ها الزاما دقت بهتری را کسب نمی‌کند. به منظور پیش‌پردازش اولیه دادگان از کتابخانه عبارات منظم پایتون استفاده شده است. بدین صورت که در ابتدا کاراکترهایی غیر اسکی [۳۰] اعداد، لینک‌ها، کلمات شامل اعداد، علائم نگارشی و علائمی که نشان دهنده خط جدید و ... است حذف می‌شود. این مرحله از پیش‌پردازش درجه صفر نام‌گذاری می‌شود. مرحله کوچک کردن تمامی حروف پیش‌پردازش درجه یک نام‌گذاری می‌شود. حال اگر کلماتی که به کلمات ایست معروف هستند نیز حذف شود درجه این پیش‌پردازش دو خواهد شد. کلمات ایست کلماتی هستند که معنای زیادی ندارند و در برخی از وظیفه‌ها حذف آن‌ها باعث بهبود جواب مساله می‌شود. از کتابخانه پردازش زبان طبیعی ان‌ال‌تی‌کی [۳۱] به منظور شناسایی این کلمات استفاده شده است. درجه آخر پیش‌پردازش (درجه سه) بازگرداندن فرم‌های مختلف یک کلمه به یک فرم واحد است. این عمل که ریشه‌یابی [۳۲] نام دارد با استفاده از تشخیص ادات سخن [۳۳] و یک پایگاه داده معروف لغات به نام وردنت [۳۴] انجام می‌شود. همه این ابزارها نیز خود در کتابخانه ان‌ال‌تی‌کی برای استفاده وجود دارند.

از بین روش‌های معرفی شده، در روش‌های سنتی یادگیری ماشین از هر ۴ روش استفاده شده است که تاثیر هر یک از آن‌ها در بخش ارزیابی مورد مطالعه قرار می‌گیرد. برای دو روش دیگر، از پیش‌پردازش درجه صفر برای مدل‌های مبتنی بر ترنسفورمر که به بزرگی و کوچکی حروف حساس است استفاده شده است. همچنین برای مدل‌های دیگر از پیش‌پردازش درجه دو استفاده شده است؛ چرا که تعبیه‌سازی کلمات خود به نحوی ریشه‌یابی را انجام می‌دهد و ریشه‌یابی روی بهبود دقت در این مدل‌ها تاثیری ندارد.

۳-۳ پیاده‌سازی مدل‌ها

در این قسمت بیشتر از منظر پیاده‌سازی روش‌های ارائه شده در بخش ۲ بررسی می‌شود.

۳-۳-۱ روش‌های یادگیری ماشین سنتی

پس از انجام هر ۴ نوع پیش‌پردازش داده‌ها مطابق بخش ارزیابی به ۳ دسته آموزش، اعتبار سنجی و آزمون تقسیم می‌شود. سپس با استفاده از الگوریتم تی‌اف-آی‌دی‌اف که در کتابخانه اسکیلرن [۳۵] ارائه شده است یک مدل تی‌اف-آی‌دی‌اف بر روی داده آموزش ساخته می‌شود. که این مدل بر روی داده آزمون و اعتبارسنجی نیز اعمال می‌شود. حال که بازنمایی‌های از جنس تی‌اف-آی‌دی‌اف

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 499, 128)	89571840
bidirectional (Bidirectional)	(None, 499, 256)	263168
dropout (Dropout)	(None, 499, 256)	0
bidirectional_1 (Bidirectional)	(None, 64)	73984
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 128)	8320
dropout_2 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 3)	387

شکل ۴: معماری شبکه عصبی حافظه‌دار

بدست آمد باز با استفاده از کتابخانه اسکیلرن می‌توان انواع مدل‌های یادگیری ماشین با پیکربندی‌های مختلف را بارگزاری و عملیات آموزش و ارزیابی را برای آن‌ها انجام داد. از جمله این مدل‌ها مدل رگرسیون لجستیک و ماشین بردار پشتیبان است که در این گزارش به عنوان مدل‌های پیشنهادی به کار گرفته شده است.

۳-۳-۲ روش‌های شبکه عصبی

پس از انجام پیش‌پردازش درجه دو، دادگان مطابق بخش ارزیابی به ۳ دسته آموزش، اعتبار سنجی و آزمون تقسیم می‌شود. سپس با استفاده از توکن‌ساز [۳۶] کتابخانه کراس [۳۷] بر روی دادگان آموزش یک مدل توکن‌ساز اموزش می‌دهد و به وسیله آن متن‌ها را به یک سری عدد کد [۳۸] می‌کند. که این مدل بر روی داده آزمون و اعتبارسنجی نیز اعمال می‌شود. سپس یک سری جاسازی نیز با استفاده از مدل فست-تکست محاسبه می‌شود که به عنوان لایه جا سازی در مدل نهایی قرار می‌گیرد. مدل نهایی برای شبکه‌های حافظه‌دار از یک معماری دوطرفه شبکه حافظه بلند کوتاه مدت [۳۹] استفاده می‌کند. همانگونه که در شکل ۴ مشخص است لایه اول لایه جاسازی شده از مدل فست-تکست است. لایه دوم و چهارم شبکه حافظه بلند کوتاه مدت دو طرفه است. همچنین لایه‌های دراپ‌اوت [۴۰] نیز به این جهت تنظیم شده است که از بیش برازش [۴۱] مدل جلوگیری کند. لایه آخر نیز به تعداد برچسب‌ها از مدل خروجی تولید می‌کند. با استفاده از پیدا کردن بیشترین مقدار خروجی می‌توان برچسب مورد نظر را پیش‌بینی کرد. به همین ترتیب نیز معماری شبکه کانولوشنی نیز در شکل ۵ نمایش داده شده است. تنها تفاوت این شبکه با شبکه قبلی لایه کانولوشن و لایه مکس‌پولینگ [۴۲] است که به جای معماری دو طرفه شبکه حافظه‌دار انتخاب شده است. لایه کانولوشن هسته اصلی این معماری است که لایه مکس‌پولینگ مقدار بیشینه تبدیل قبل از خود را برداشته و خروجی می‌دهد. پیاده‌سازی این مدل تماما با استفاده از کتابخانه کراس انجام شده است.

۳-۴ روش‌های مبتنی بر ترنسفورمرها

۳ مدل برت پایه، روبرتا-پایه و برت بزرگ برای این قسمت انتخاب شدند. برخی از این مدل‌ها به بزرگ و کوچک بودن متن حساس هستند بنابراین به ازای آن‌ها از پیش‌پردازش درجه ۰ استفاده شده است. برای آن‌هایی که به بزرگ و کوچک بودن حروف حساس نیستند از پیش‌پردازش درجه ۲ استفاده شده است. عملیات تعدیل و تنظیم مجدد پارامتر این مدل‌ها با استفاده از کتابخانه کی‌ترین [۴۳] انجام شده است.

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, None, 128)	89571840
conv1d (Conv1D)	(None, None, 256)	98560
global_max_pooling1d (GlobalMaxPooling1D)	(None, 256)	0
dense (Dense)	(None, 256)	65792
dense_1 (Dense)	(None, 3)	771

شکل ۵: معماری شبکه کانولوشنی

این کتابخانه یک لایه به بالاسر هر کدام از این مدل‌ها اضافه می‌کند و با استفاده از دادگان آموزش اقدام به تعدیل و تنظیم تمامی پارامترها می‌کند. تنها کافی است که نام مدل در کد تغییر کند و به ازای آن مدل اجرا شود.

۴ ارزیابی و تحلیل رویکردهای پیشنهادی

۴-۱ معیارهای ارزیابی

با توجه به اینکه معیار دقت به تنهایی نمی‌تواند نشان دهنده میزان درستی و و قابلیت اعتماد بودن مدل باشد به همین دلیل برای ارزیابی مدل‌ها علاوه بر دقت معیارهای دیگری نیز گزارش شده است. در این قسمت مختصرا به معرفی هر کدام از این موارد پرداخته می‌شود.

- دقت[۴۴]:

دقت در واقع نسبت تعداد نمونه‌هایی است که مدل نسبت به کل داده‌ها درست تشخیص داده است. مشکل دقت زمانی مشخص می‌شود که توزیع دسته‌ها مختلف در داده ورودی یکسان نباشند. برای مثال اگر تعداد داده‌های تست صد باشد و از این بین ۹۰ عدد از داده‌ها مربوط به دسته اول و بقیه جزء دسته دوم باشند. اگر مدل همه داده ها را از دسته یک پیش‌بینی کند دقت ۹۰ درصد خواهد بود که دقت خوبی به نظر می‌اید ولی در واقع مدل چیزی را آموزش ندیده است. برای حل این مشکل دو معیار دیگر درستی [۴۵] و به یاد آوردن [۴۶] معرفی شده است که این مشکل را برطرف می‌کنند.

- درستی:

معیار درستی در واقع بیان می‌کند که چه نسبتی از داده‌هایی که به یک کلاس نسبت داده شده، درست بوده است برای مثال اگر مدلی ۱۰۰ داده را از دسته اول تشخیص بدهد و از این ۱۰۰ داده‌ای که تشخیص داده است، ۶۰ مورد درست باشد. درستی ۶۰ درصد خواهد بود.

- به یاد آوردن:

معیار به یاد آوردن بیان کننده نسبت پیش‌بینی‌های درست از یک دسته، نسبت به تعداد کل داده‌هایی است که از آن دسته بوده اند.

دو معیار درستی و به یادآوردن هنگامی می‌توانند به عنوان معیار در نظر گرفته شوند که با هم در نظر گرفته شوند. این بدان خاطر است که می‌توان با کاهش یکی دیگری را افزایش داد. از این رو معیار دیگری به نام امتیاز-اف۱ [۴۷] تعریف شد. که ترکیبی از هردو معیار است. در نتیجه با دیدن آن می‌توان نتیجه گرفت که مدل به چه نسبتی خوب عمل کرده است.

ذکر این نکته نیز ضروری است که اگر دادگان چندین کلاس داشته باشند امتیاز نهایی اف-۱، درستی و به یادآوردن هم می‌توانند به صورت وزن‌دار (نسبت به تعداد داده هر کلاس) و هم به صورت سختگیرانه‌تری با مشارکت مساوی محاسبه شوند. در ارزیابی‌های این گزارش از حالت سختگیرانه‌تر که با نام میانگین ماکرو [۴۸] معروف است، استفاده شده است.

۴-۲ نحوه انجام آموزش

به ازای هر یک از مدل‌های معرفی شده در بخش رویکردهای پیشنهادی، دادگان به ۳ قسمت داده آموزش، اعتبارسنجی و آزمون تقسیم می‌شوند. نسبت این تقسیم به صورت ۷۰% برای دادگان آموزش، ۱۰% برای دادگان اعتبار سنجی و ۲۰% برای دادگان آزمون است. دلیل این انتخاب نیز آن است که در بیشتر مقالات نسبت ۵ به ۱، ۴ به ۱ و ۳ به ۱ برای دادگان آموزش نسبت به آزمون انتخاب می‌شود که به طور تجربی این نسبت‌ها مقادیر مناسبی هستند. همچنین در اینجا نیز دادگان اعتبارسنجی نیز در نظر گرفته شدند تا از آن‌ها برای اعتبارسنجی مدل در هر دوره یادگیری یا انتخاب هایپرپارامترها [۴۹] استفاده شود. از انجا که تعداد دادگان یادگیری قابل توجه و زیاد است این روش و این نسبت‌ها یکی از مناسب‌ترین انتخاب‌ها برای آموزش، اعتبارسنجی و آزمون مدل است.

۴-۳ روش‌های یادگیری ماشین سنتی

همانگونه که در بخشهای پیشین نیز بیان شد از دو مدل رگرسیون لجستیک و بردار ماشین پشتیبان به عنوان نمونه الگوریتم‌های یادگیری ماشین سنتی استفاده شده است. در این بخش به آزمون هر یک از این مدل‌ها با درجه‌های مختلف پیش‌پردازش پرداخته می‌شود. از انجا که الگوریتم یادگیری ماشین بردار پشتیبان می‌تواند کرنل‌های مختلفی داشته باشد کرنل‌های مختلف نیز با استفاده از دادگان اعتبار سنجی برای آن آزموده شده است. پس از آزمایش مشخص شد که کرنل آربی‌اف [۵۰] برای این نوع داده بهتر است. بنابراین در جداول ارزیابی منظور از مدل ماشین بردار پشتیبان مدل ماشین بردار پشتیبان با کرنل آربی‌اف است. به طور مثال در جدول ۱ نمونه دقت و امتیاز اف-۱ به ازای کرنل‌های مختلف با پیش‌پردازش درجه ۳ به ازای تعداد تکرار ۱۰۰۰ مرتبه و تحت شرایط برابر بر روی دادگان اعتبار سنجی را مشاهده می‌کنید.

جدول ۱: دقت مدل ماشین بردار پشتیبان با پیش‌پردازش درجه ۳ بر روی دادگان اعتبار سنجی

کرنل	دقت	امتیاز-اف۱
خطی	۴۹%	۴۲%
آربی‌اف	۵۵%	۴۶%
چندجمله‌ای	۴۹%	۳۳%

از آنجا که مدل بردار پشتیبان برای این حجم از داده به تعداد زیادی تکرار نیاز داشت تا همگرا شود تکرار هر دو مدل به عدد ۱۰۰۰ محدود شد. پس از انجام نتایج و دریافت بهترین پیکربندی هر دو مدل بهتر با تعداد تکرار ۱۰۰۰۰ بار مجددا آموزش دیدند. در جدول ۲ نتایج ارزیابی به ازای مدل‌های مختلف در شرایط برابر، به ازای تعداد تکرار ۱۰۰۰ بار در آموزش و با درجه‌های مختلف پیش‌پردازش را مشاهده می‌کنید. همانگونه که مشخص است بهترین مدل رگرسیون لجستیک با درجه پیش‌پردازش ۰ و ۱ است و از بین مدل‌های بر اساس ماشین بردار پشتیبان، مدل ماشین بردار پشتیبان با درجه پیش‌پردازش ۲ بهترین مدل است.

جدول ۲: دقت‌های ارزیابی مدل‌های یادگیری ماشین سنتی

مدل	درجه پیش‌پردازش	دقت	امتیاز-اف۱
رگرسیون لجستیک	صفر	۷۹%	۶۸%
رگرسیون لجستیک	یک	۷۹%	۶۸%
رگرسیون لجستیک	دو	۷۷%	۶۴%
رگرسیون لجستیک	سه	۷۶%	۶۳%
ماشین بردار پشتیبان	صفر	۴۸%	۳۶%
ماشین بردار پشتیبان	یک	۴۸%	۳۶%
ماشین بردار پشتیبان	دو	۵۶%	۴۶%
ماشین بردار پشتیبان	سه	۵۵%	۴۶%

در نهایت هر دو مدل رگرسیون لجستیک با درجه پیش‌پردازش ۱ و مدل ماشین بردار پشتیبان با درجه پیش‌پردازش ۰ با تعداد تکرار ۱۰۰۰۰ بار آموزش دیدند که نتایج آن در جدول ۳ گزارش شده است. از جدول مشخص است که با اضافه شدن تکرار مدل ماشین بردار پشتیبان می‌تواند دقت‌های بهتری کسب کند. متاسفانه منابع موجود اجازه تعداد تکرار بالاتر را نمی‌داد تا دقیقا مشخص شود که این مدل تا چه مقدار دقت را در نهایت می‌تواند کسب کند. در هرصورت می‌توان این نتیجه را گرفت که مدل رگرسیون لجستیک در زمان خیلی کمتری نتیجه بهتری را می‌تواند ارائه دهد و همچنین به ازای تعداد تکرار برابر حتی تا مقدار ۱۰۰۰۰ بازهم مدل لجستیک روی این دادگان بهتر عمل کرده است.

همچنین نتیجه‌گیری دیگر آن است که در این نوع داده استفاده از پیش‌پردازش باعث افت دقت در مدل رگرسیون لجستیک می‌شود و در مدل ماشین بردار پشتیبان نیز درجه پیش‌پردازش ۲ که شامل پیش‌پردازش اولیه، کوچک کردن حروف کلمات و حذف کلمات اضافه است بهترین جواب را می‌دهد.

جدول ۳: دقت‌های ارزیابی مدل‌های یادگیری ماشین سنتی با تعداد تکرار ۱۰۰۰۰ بار

مدل	درجه پیش‌پردازش	دقت	امتیاز-اف۱
رگرسیون لجستیک	یک	۷۹%	۶۸%
ماشین بردار پشتیبان	دو	۶۲%	۵۴%

۴-۴ روش‌های شبکه عصبی

همانگونه که در بخش‌های پیشین نیز بیان شد از دو مدل شبکه‌های عصبی حافظه دار دوجهته و شبکه‌های عصبی کانولوشنی به عنوان دو نمونه روش‌های شبکه عصبی استفاده شده است. از انجا که تعداد دادگان مساله زیاد بود و منابع محدود اجازه آموزش بیشتر را نمی‌داد تنها دو دوره هر کدامیک از این شبکه‌ها اموزش دیدند که در جدول ۴ دقت و امتیاز-اف۱ آن‌ها گزارش شده است. دقت اعداد گزارش شده به درصد است، اما آنچه که در جدول مشخص نیست ان است که هر دو مدل‌ها در دوره دوم نسبت به قبل در معیار دقت پیشرفت داشتند حال آنکه پیشرفت آن‌ها کمتر از ۱ درصد بوده است و به همین خاطر در جدول مشخص نیست.

جدول ۴: دقت‌های ارزیابی مدل‌های شبکه عصبی

مدل	دوره	دقت	امتیاز-اف۱
شبکه حافظه‌دار دو جهته	یک	۷۹%	۶۸%
شبکه حافظه‌دار دو جهته	دو	۷۹%	۶۷%
شبکه کانولوشنی	یک	۷۸%	۶۷%
شبکه کانولوشنی	دو	۷۸%	۶۷%

۴-۵ روش‌های مبتنی بر ترنسفورمر

در این روش ۳ مدل که پایه آن‌ها بر اساس مدل برت توسعه یافته است مورد آزمایش قرار گرفت. متاسفانه موقع کد زدن تنها کد ارزیابی دقت زده شده بود و خروجی دقت در یک فایل ذخیره شده بود. به همین خاطر پس از آموزش به خود مدل دسترسی وجود نداشت و تنها دقت‌های هر مدل در گزارش آمده است. همچنین امکان دوباره آموزش این مدل‌ها به دلیل نبود منابع وجود نداشت. بر روی یک کارت گرافیک کی-۸۰ [۵۱] آموزش هر دوره مدل پایه برت با این تعداد داده نزدیک ۱۴ ساعت طول می‌کشد. به همین خاطر تنها ۳ مدل به صورت موازی و با یک دوره آموزش دیدند.

جدول ۵: دقت‌های ارزیابی مدل‌های مبتنی بر ترنسفورمر

مدل	دوره	دقت
روبرتا-پایه	یک	۸۲%
برت-پایه-حساس به کوچکی و بزرگی	یک	۸۱%
برت-بزرگ-حساس به کوچکی و بزرگی	یک	۸۰%

۴-۶ نتیجه‌گیری ارزیابی

دقت مدل‌ها در ۳ روش یاد شده به یکدیگر نزدیک هستند. به ترتیب در مدل مبتنی بر ترنسفورمر دقت با ۱ الی ۳ درصد نسبت به مدل‌های قبلی بهتر است. همچنین مدل‌های شبکه عصبی نیز با دقت کمتر از ۱ درصد نسبت به مدل‌های یادگیری ماشین سنتی دقت بهتری کسب کردند. از نظر سرعت یادگیری نیز باید گفت آموزش مدل رگرسیون لجستیک خیلی زودتر از بقیه آموزش‌ها انجام می‌شود. پس از آن مدل شبکه‌های کانولوشنی سرعت مناسب‌تری داشت.

۵ تهدیدات علیه اعتبار

همانطور که در بخش ۴ بیان شد در مواردی محدودیت‌های منابع محاسباتی وجود داشت. این امر باعث شد که برخی از مدل‌ها در تعداد دوره کمتری آموزش ببینند. و این مشخص نیست که این مدل‌ها چقدر می‌توانستند به نتایج خیلی بهتر دست پیدا کنند. این در حالی است که اگه محدودیت محاسبات نبود، مدل‌ها بهتر و بیشتر آموزش می‌دیدند و احتمالا نتایج بهتری به دست می‌آمد. لذا اولین نقطه‌ی ضعفی که این پژوهش دارد کمبود منابع محاسباتی چه از لحاظ حافظه و چه از لحاظ سرعت محاسبات است. همچنین کمبود حافظه موجب شده است که در مدل‌هایی مانند فست-تکست که نیاز به حافظه‌ی نسبتا زیادی داشتند، به دلیل کمبود حافظه از نسخه‌های کم‌حجم‌تر استفاده شود. این نسخه قاعدتا بخشی از اطلاعات را از دست داده است.

مورد دیگر که نیازمند توجه است این است که مدل‌های سنتی خود محدودیت‌هایی دارند که می‌تواند نتیجه را تحت تاثیر قرار دهد. به طور مثال در این روش‌ها ترتیب کلمات در جملات اهمیت ندارد، به این معنا که این مدل‌ها برایشان فرقی ندارد که چه کلمه‌ای در کجای جمله‌ای قرار داشته باشد. این در حالی است که تغییر ترتیب می‌تواند در مفهوم تاثیرگذار باشد. هرچند برای حل مشکل از روش‌های جدید استفاده شده است تا نتایج آن‌ها هم بررسی و مقایسه شوند اما همچنان این مورد که روش‌های سنتی استفاده شده این مشکل را دارند پابرجاست. خصوصا که برای اجرای روش‌های جدید مانند برت محدودیت محاسباتی و حافظه وجود داشت و امکان اینکه بهترین نتیجه از آن حاصل شود وجود نداشت.

از طرف دیگر پیش‌پردازش نیز می‌تواند در نتیجه‌ی نهایی تاثیرگذار و حتی به نوعی مشکل آفرین باشد. به عنوان مثال با حذف کلمات ایست مرحله پیش‌پردازش ممکن است بخشی از مفاهیم از دست داده بشود. هرچند کلمات ایست اکثرا مواردی هستند که کمتر در جمله تاثیرگذار هستند اما در حالت‌هایی می‌توانند در تغییر موضوع متن نوشته شده اثر داشته باشند. در نتیجه انتخاب کلمات ایست صحیح، به اندازه و متناسب با زبان خیلی می‌تواند در داشتن یک پیش‌پردازش مناسب و در نهایت داشتن خروجی بهینه اثربخش باشد.

۶ نتیجه‌گیری

دسته‌بندی و رسیدگی به ایشوها توسط تیم توسعه‌ی نرم‌افزار می‌تواند یک مسئله‌ی چالش برانگیز باشد. وجود سیستمی که بتواند به صورت خودکار، ایشوها را برچسب بزند می‌تواند به توسعه‌دهندگان کمک شایانی کند. اما حتی با وجود ویژگی اضافه شدن برچسب ایشوها توسط کاربران و توسعه‌دهندگان، هنوز حتی پروژه‌های بزرگ گیت‌هاب بسیار زیادی ایشو برچسب نزده دارند.

در این گزارش چند روش داده محور به منظور ساخت یک روش خودکار تعیین برچسب ایشوها مورد بررسی قرار گرفت. در این روش‌ها که شامل قدیمی‌ترین و با ثبات‌ترین روش‌ها تا جدیدترین پیشرفت‌های حوزه پردازش زبان طبیعی بود دریافت شد که می‌توان با دقت مناسبی برچسب ایشوها را با پیش‌بینی کرد. امید است در آینده با مطالعه بیشتر یک بات برای سایت گیت‌هاب طراحی شود تا بتواند این روش خودکار برچسب‌زنی را بر روی ایشوهای کاربران اعمال کند و از تصحیح برچسب توسط کاربران باز خود را بهبود دهد.

مراجع

[1] A. B. Dhasade, A. S. M. Venigalla, and S. Chimalakonda, "Towards prioritizing github issues," in *Proceedings of the 13th Innovations in Software Engineering Conference on Formerly known as India Software Engineering Conference*, pp.1–5, 2020.

[2] "Mastering issues - github guides," https://guides.github.com/features/issues/.

[3] M. Izadi, K. Akbari, and A. Heydarnoori, "Predicting the objective and priority of issue reports in software repositories," *Empirical Software Engineering*, vol.27, no.2, pp.1–37, 2022.

[4] S. Soumya and K. Pramod, "Sentiment analysis of malayalam tweets using machine learning techniques," *ICT Express*, vol.6, no.4, pp.300–305, 2020.

[5] A. Prabhat and V. Khullar, "Sentiment classification on big data using naïve bayes and logistic regression," in *2017 International Conference on Computer Communication and Informatics (ICCCI)*, pp.1–5, IEEE, 2017.

[6] S. Minaee, E. Azimi, and A. Abdolrashidi, "Deep-sentiment: Sentiment analysis using ensemble of cnn and bi-lstm models," *arXiv preprint arXiv:1904.04206*, 2019.

[7] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the association for computational linguistics*, vol.5, pp.135–146, 2017.

[8] R. Kallis, A. Di Sorbo, G. Canfora, and S. Panichella, "Ticket tagger: Machine learning driven issue classification," in *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp.406–409, IEEE, 2019.

[9] R. Kallis, A. Di Sorbo, G. Canfora, and S. Panichella, "Predicting issue types on github," *Science of Computer Programming*, vol.205, p.102598, 2021.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol.9, no.8, pp.1735–1780, 1997.

[12] W. Wang and J. Gang, "Application of convolutional neural network in natural language processing," in *2018 International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pp.64–70, IEEE, 2018.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol.30, 2017.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[15] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

پانویس‌ها

[۱] Issue
[۲] Feature
[۳] GitHub
[۴] Jira
[۵] Open Source
[۶] Code Repository
[۷] Bug
[۸] Enhancement
[۹] Represenation
[۱۰] Term Frequency-Inverse Document Frequency (TF-IDF)
[۱۱] Logistic Regression
[۱۲] Support Vector Machine
[۱۳] Parametric
[۱۴] Sigmoid
[۱۵] Kernel
[۱۶] Convolution
[۱۷] Embedding
[۱۸] FastText
[۱۹] Word2vec
[۲۰] N-Gram
[۲۱] Transfomer
[۲۲] Attention
[۲۳] Bert
[۲۴] Roberta
[۲۵] Python 3
[۲۶] Json
[۲۷] Pandas
[۲۸] Dataframe
[۲۹] Machine Hack
[۳۰] Ascii
[۳۱] NLTK
[۳۲] Lemmatization
[۳۳] Part of Speach
[۳۴] WordNet
[۳۵] Sklearn
[۳۶] Tokenizer
[۳۷] Keras
[۳۸] Code
[۳۹] Long Term Short Memory
[۴۰] Dropout
[۴۱] Over Fitting
[۴۲] MaxPooling
[۴۳] Ktrain
[۴۴] Accuracy
[۴۵] Precision
[۴۶] Recall
[۴۷] F1-score
[۴۸] Macro Average
[۴۹] Hyperparameter
[۵۰] RBF
[۵۱] K-80