بسم الله الرحمن الرحیم

شرکت مهندسی نرم افزاری هلو

گزارش تحقیق در خصوص RAG ۲

کاری از امیرعلی نسیمی

فهرست

مدل ParsBERT

مدل BERT یک مدل زبانی است که برای زبان انگلیسی طراحی گردیده است. اما اگر بخواهیم از این مدل برای کارهایی که با جملات فارسی سروکار دارند استفاده کنیم، چندان موثر نخواهد بود؛ چرا که به شکل طبیعی ساختار زبان فارسی و انگلیسی با یکدیگر تفاوتهای اساسی دارند. در این بخش به معرفی مدل زبانی ParsBERT خواهیم پرداخت که با کمک مدل BERT انگلیسی، برای زبان فارسی طراحی شده است.ParsBERT یک مدل زبانی مختص زبان فارسی است که در سال ۲۰۲۱ ارائه شد. معماری این مدل با الهام از معماری BERT انگلیسی و مشابه آن می باشد. به همین دلیل، در این فصل تنها به فرآیند آماده سازی داده های فارسی برای مدل، بسنده کرده ایم و باقی، شبیه به مدل BERT عمل می کند.

هدف از این پژوهش، بررسی روش مبتنی بر ParsBert می باشد؛ نحوه استخراج داده ها و اطلاعات، نوع آموزش و فرآیند ارزیابی مورد بررسی قرار گرفته شده است.

جمع آوری داده

منابعی که داده های مرحله ی پیش آموزش مدل ParsBERT از آنها استخراج شده است، به سرح زیر می باشد:

- Persian Wikipedia
- BigBang Page
- Chetor
- EliGasht
- DigiKala
- Ted Talks
- Miras-text

پیش پردازش داده

پس از جمعآوری مجموعه داده ی پیش آموزش، چند مرحله پیش پردازش، از جمله پاکسازی ، جایگزینی و نرمال سازی، برای تبدیل مجموعه داده ها به یک قالب مناسب ضروری است. ابتدا باید تمام کاراکترهای ناخواسته و بی اهمیت موجود در مجموعه را پاک کنیم. این کار از طریق یک سری فرآیند انجام می شود که عبارتند از:

۱. اصلاح تمام کاراکترهای Unicode
۲. حذف تمام اطلاعات بی اهمیت به عنوان مثال URL ها.
۳. حذف تمام برچسب های غیر ضروری مانند تگ های HTML، CSS و JS
۴. نرمالسازی متن فارسی با استفاده از کتابخانهی HAZM : کتابخانه ای محبوب برای پیش پردازش متن فارسی است. نرمالسازی متن، اصلاح حروف فارسی، حذف فاصله های اضافی، تبدیل اعداد انگلیسی و عربی به فارسی، اصلاح علائم نگارشی و اصلاح نیم فاصله را شامل می شود.
۵. حذف کاراکترهای اسکی ناخواسته در زبان فارسی.
۶. جایگزینی کاراکترهای عربی با معادل آنها در فارسی.
۷. حذف یا جایگزینی کاراکترهای Unicode بی معنی.

تقسیم اسناد به جملات صحیح فارسی

پس از اینکه مجموعه پیش پردازش شد، باید به جملات واقعی هر سند، تبدیل شود. یک جمله ی صحیح در فارسی بر اساس نمادهای نقطه، علامت تعجب، علامت سوال و دونقطه تشخیص داده می شود. با این حال، تقسیم محتوا صرفا بر اساس این نمادها، باعث بروز مشکلاتی می شود. در ارزیابی های انجام شده مشاهده می شود که نتیجه، شامل جملات کوتاه بی معنی است. زیرا در فارسی اختصاراتی وجود دارد که با علامت نقطه از هم جدا شده اند. در زبان فارسی تمام جمله ها به فعل ختم می شوند. بنابراین، هر گاه قسمتی با نشانه ای که تگ شده است، پایان یابد و یکی از نمادها به دنبال آن قرار گیرد، به عنوان یک بخش معنی دار صحیح به عنوان فعل علامت گذاری شده و به عنوان یک جمله از بقیه قسمت ها جدا می شود.

پس از آنکه مجموعه داده ی پیش آموزش آماده شد، نوبت به دو مرحله ی پیش آموزش و تنظیم دقیق میرسد. این دو مرحله شبیه آنچه که در BERT است اتفاق میفتد؛ بنابراین از تکرار آنها پرهیز می کنیم. گفتنی است در مقاله ی اصلی، نتایج تنظیم دقیق این مدل برای سه کار تحلیل احساس، طبقه بندی متون و تشخیص موجودیت های نامدار آمده است.

پیاده سازی

برای پیاده سازی این پروژه، از اکوسیستم Hugging Face استفاده شده است. در اصل یک شرکت هوش مصنوعی است که اکوسیستمی فراهم آورده که با تجمیع و توسعه ی کاربردی ترین ابزارهای پردازش زبان طبیعی، حل مسائل مربوط به این حوزه را تسهیل بخشیده است. کتابخانه های این اکوسیستم، از جمله transformers ، یکی از کتابخانه های محبوب زبان پایتون به حساب می رود.

آماده سازی داده ها

برای طراحی سیستم پرسش و پاسخ، در ابتدا نیاز است داده ها را آماده کنیم. برای آماده سازی داده های PersianQAاز کتابخانه ی datasets ، استفاده کرده ایم. تابع load_dataset به ما کمک می کند مجموعه داده ی پردازش زبان طبیعی مورد نظر را بارگیری و از آن استفاده کنیم. این مجموعه داده می تواند در هر جایی ذخیره شده باشد. روی Hugging Face Hub ، روی دیسک محلی یا بر رویGithub . در این پروژه PersianQAرا از Hugging Face Hub بارگیری نموده ایم. با استفاده از تابع load_metric می توان ملاک های ارزیابی مورد نظر را بارگیری کرد. در صورتی که ملاک های ارزیابی توسط کتابخانه ی datasets به صورت مستقیم، پشتیبانی نشود نیز می توان ملاک های ارزیابی جدیدی تعریف و از آنها استفاده کرد. معیارهایی که پیشتر معرفی کردیم، برای این پروژه توسط کتابخانه ی datasets پشتیبانی می شود.

پیش از آنکه داده ها را وارد مدل کنیم، نیاز است آنها را به نشانه تبدیل کنیم. این عملیات توسط کلاس AutoTokenizer و با تابع from_pretrained قابل انجام است. با این تابع، میتوانیم Tokenizer از پیش

آموزش دیده ی مد نظر را بارگیری و از آن استفاده کنیم. مدل ParsBERT که می خواهیم از آن استفاده کنیم، با آدرسHooshvareLab/bert-fa-zwnj-base از کتابخانه ی transformers قابل دسترسی است Tokenizerآن نیز بر مبنای الگوریتم WordPiece می باشد.

تنظیم دقیق مدل

برای ساخت مدلی که لایه ی خروجی آن، متناسب با یک کار پایین دستی خاص است، میتوان از Automodel هایی که توسط کتابخانه ی transformers معرفی می شوند استفاده کرد. هر کار پایین دستی که بخواهیم انجام بدهیم، AutoModel مخصوص خود را دارد که لایه ی آخر آن، به فراخور کار مورد نظر طراحی شده است. در مورد کار پرسش و پاسخ، ازAutoModelForQuestionAnswering استفاده کرده ایم. با استفاده از متد from_pretrainedاز این کلاس، می توانیم مدلی که می خواهیم از آن یک مدل پرسش و پاسخ بسازیم را انتخاب کنیم و وزن های آن را بارگیری نماییم. این کلاس، با جایگزین کردن لایه ی خروجی مدل از پیش آموزش دیده با لایه ی جدید متناسب با کار پرسش و پاسخ، مقدمات تنظیم دقیق را برای ما فراهم می آورد. در این مرحله به تعداد ایپاکی که می خواهیم، مدل را آموزش می دهیم تا وزن های جدید، به دست آیند.

کار آموزش مدل از طریق کلاس Trainer از کتابخانه ی transformers ، قابل انجام است. برای آنکه بتوانیم از این کلاس استفاده کنیم، ابتدا باید آرگومان های مربوط به آموزش را تنظیم کرده و به عنوان پارامتر ورودی به تابع train از این کلاس بدهیم. پس از آن با فراخوانی این تابع، فرآیند آموزش شروع می شود. در طول آموزش، پس از پایان هر ایپاک، میتوان تابع هزینه ی مربوط به داده های آموزش و داده های اعتبارسنجی یا آزمایش را دید.

ارزیابی مدل

خروجی مدل برای هر نمونه، مقدار تابع هزینه و بردار نهایی متناظر با احتمال آغاز و پایان جواب هر نشانه را بر میگرداند. برای انتخاب بهترین شاخص شروع و پایان، می توان نشانه ای که بهترین امتیاز را برای شاخص شروع بودن

دارد، و نشانه ای که بهترین امتیاز را برای شاخص پایان بودن دارد، برداریم و به عنوان جواب در نظر بگیریم. اما همانطور که اشاره کردیم، باید این شرط را در نظر بگیریم که شاخص پایان کوچکتر از شاخص شروع نباشد؛ به همین دلیل ما به تعداد n بهترین کاندید جواب را انتخاب میکنیم. میزان n متغیر است. در این پروژه این عدد برابر ۲۰ در نظر گرفته شده است. از بهترین جواب شروع به بررسی میکنیم و بهترین جوابی که شرط را ارضا کرد را به عنوان جواب نهایی در نظر میگیریم. برای اینکار نیازمند یک حلقه ی تودرتو هستیم.

پس از پایان کار مدل، میتوان آن را در اکوسیستم Hugging Face قرار داد. این اکوسیستم همچنین یک رابط کاربری نیز فراهم کرده است که به راحتی میتوان به تست مدل پرداخت. تصویر زیر مربوط به مدلیست که تنظیم کرده ایم. جمله ی "من به کار پردازش زبانهای طبیعی علاقه دارم " با توجه به متن "من به چه کاری علاقمندم " پاسخی است که برای این سوال برگردانده است.