TOXIC COMMENT CLASSIFICATION

Arman Heidari
armanheidari192@gmail.com

معرفی

امروزه فضای مجازی پر از نظرات نامناسب است !

با توجه به این که فیلتر نظرات در فضـــای مجازی امری بســـیار مهم می‌باشـد. در این پروژه سـعی شـده با اسـتفاده از مدل یادگیری عمیق، انواع پیام‌های نامناسب (سمی) شناسایی و دسته‌بندی شود تا در ادامه بتوان آنها را حذف کرده و فضایی مناسب و به دور از توهین و خشونت به وجود آورد.

لینک درایو پروژه

کتابخانه‌ها

در این پروژه از کتابخانه‌های متفاوتی برای خواندن و پیش‌پردازش داده‌ها، ایجاد مدل و ارزیابی آن و نمایش نتایج استفاده شده است.

numpy : برای انجام عملیات محاسباتی بر روی آرایه‌ها.

pandas : برای کار با داده‌های ساختارمند و جداول داده.

matplotlib : برای تصویرسازی داده‌ها و نمودارسازی.

keras : برای ساخت و آموزش مدل یادگیری عمیق پیاده‌سازی شده.

nltk : برای پیش‌پردازش کامنت‌ها.

داده‌ها

داده‌های استفاده شده در این پروژه از یکی از مسابقات سایت **kaggle** برداشته شده است که در این لینک قابل مشاهده می‌باشد.

داده‌ها از سه فایل مجزا تشکیل شده‌اند که در ادامه به بررسی هر کدام از آن‌ها می‌پردازیم:

- **train.csv**: این فایل شـــامل ۱۵۹۵۷۰ داده متشـــکل از متن نظر و لیبل‌های منتسب به آن است.
- **test.csv**: این فایل شامل ۱۵۳۱۶۳ متن نظر است.
- **test_lables.csv**: این فایل شامل لیبل‌های نظرات تست می‌باشد.

لیبل‌های بررسی شده عبارتند از:

toxic - severe_toxic - obscene - threat - insult - identity_hate

پیش‌پردازش

با توجه به بررسی صورت گرفته بر روی داده‌ها، قبل از tokenize کردن آنها نیاز به یک مرحله پاکسازی داده‌ها بود که برای این کار از کتابخانه nltk استفاده شد که تا جای ممکن داده‌ها تمیز باشند.

در ادامه با استفاده از کلاس Tokenizer داده‌ها توکنایز شده و تعداد کلمات خاص (MAX_FEATURES) به دست آمد. از طرفی با بررسی طول کامنت‌ها مشخص شد که اغلب کامنت‌ها با کمتر از ۱۵۰ کلمه ساخته شده‌اند که از این عدد برای تعیین سایز معین برای کامنت‌ها و pad استفاده شد.

در پایان این مرحله، تمامی داده‌ها ساختار ثابت و مشخصی به خود گرفته و آماده آموزش و تست مدل شدند.

مدل یادگیری عمیق

در طراحی مدل از لایه‌های مختلفی استفاده شده است که در ادامه به ترتیب مورد بررسی قرار خواهند گرفت (در ابتدا از دو طراحی ساده‌تر استفاده شد و درنهایت با توجه به دقت بالاتر، این مدل انتخاب شد).

- **input**: لایه ورودی شبکه.
- **Embedding**: در ابتدای شبکه برای رفع **sparse** بودن داده‌ها از این لایه‌ها استفاده شده است.
- **Bidirectional LSTM**: با توجه به این که **LSTM** برای کار با داده‌های متنی که **sequential** هستند مناسب است و با توجه به این که استفاده از **Bidirectional** سبب بررسی دو طرفه متن ورودی و درک بهتر هر کلمه می شود؛ در نهایت از این لایه استفاده می‌شود.
- **Attention**: استفاده از مکانیزم توجه، سبب بهبود عملکرد مدل و درک هرچه بهتر نقش کلمات می شود که انتخاب مناسبی برای این شبکه بود.

مدل یادگیری عمیق

- **1D Convolotion**: استفاده از لایه کانولوشن در ابتدا انتخاب عجیبی به نظر می‌رسید، اما با بررسی بیشتر و ایده‌برداری از نتایج این لینک، این لایه به همراه **max pooling** استفاده شد.
- **Dense**: برای تبدیل بهتر خروجی **max pooling** و طبق ساختار **CNN** از این لایه استفاده شده است.
- **Output**: لایه خروجی شامل ۶ نورون (۱ نورون برای هر کلاس) با تابع فعالیت **sigmoid** برای مشخص کردن پیشبینی نهایی استفاده شده است.

در فرآیند آموزش شبکه با توجه به این که کلاس‌ها می‌توانند **overlap** داشته باشند و الزاما کامنت در یک کلاس قرار نمی‌گیرد **binary_crossentropy** مورد استفاده قرار گرفت.

همچنین دو **call back** زیر برای افزایش سرعت و بهبود عملکرد مدل در حین یادگیری به مدل اضافه شدند.

مدل یادگیری عمیق

- **EarlyStopping**: برای توقف روند یادگیری در صورتی که در ۳ ایپاک متوالی بهبود چندانی در مدل حاصـــل نشـــود (جلوگیری از **overfit** شدن مدل).
- **ModelCheckpoint**: برای ســـیو کردن بهترین مدل در پایان هر ایپاک، تا در نهایت بهترین مدل از طریق آن لود شود.

ارزیابی

برای بررسی عملکرد مدل، داده‌های تست به آن داده شد که نتایج قابل توجهی داشت.

Accuracy: ۰/۹۹۹۰

Loss: ۳/۵۹۷۹