**بسم الله الرحمن الرحیم**

**پروژه پردازش متن**

**برای محصولات دیجیکالا**

نسخه شماره 5

تیم ارائه دهنده: داده کاوانِ فونیکس

ناظر و سرپرست تیم: دکتر علیرضا وفایی صدر

IPMدرخواست دهنده: ستاد علوم شناختی

بهار 1399

فهرست مطالب

|  |  |  |
| --- | --- | --- |
| 2 | ------------------------------------------------------------ | **تعریف پروژه** |
| 2 | ------------------------------------------------------------ | **تولید جملات معنادار از جملات فیک و بدون ساختار** |
| 13 | ------------------------------------------------------------ | **جمع بندی** |

تعریف پروژه

در ادامه فاز دوم و بحث های صورت گرفته، به ادامه کار تولید متن با لایک های بالا پرداختیم.

ولی جملات تولید شده جملاتی بدون گرامر و بی معنی بودند که فقط شامل دنباله ای از لغات هستند.

در این پروژه با استفاده از مدل ترجمه ماشینی Machine Translation کامنت های بی معنی را سعی کردیم معنادار کنیم.

تولید جملات معنادار از جملات فیک و بدون ساختار

برای اینکه مدلی را بسازیم که از کامنت های بی معنا و بدون ساختار به متن هایی دارای مفعوم و معنا برسد، ابتدا ترتیب کلمات در هر کامنت از دیتاست را به هم ریختیم تا آنرا به مدل داده و مدل کامنت های بی ساختار و به هم ریخته را بشناسد.

برای اینکار هر کامنت را شافل shuffle کرده و جملات هر کامنت ها را به هم ریختیم!

کامنت های به هم ریخته را به عنوان داده های آموزشی و کامنت های اولیه و دارای معنا را هم به عنوان ستون تارگت و هدف در نظر گرفتیم.

داده ها را به دو دسته داده های تست و ولیدیشن تقسیم کرده و کار tokenizer و تبدیل هر کلمه به عدد متناظر را هم برای داده های شافل شده و هم برای داده های منظم در ستون هدف انجام دادیم.

قرار است مدلی را تولید کنیم که با مشاهده هر کامنت شافل شده و به هم ریخته، بتواند به کامنتی منظم و دارای مفهوم دست پیدا کند. برای این روش از مدل ترجمه ماشینی استفاده کردیم که دارای معماری زیر می باشد:

* یک Encoder با یک لایه امبدینگ و یک لایه شبکه بازگشتی GRU
* یک لایه Attention
* یک Decoder با یک لایه امبدینگ و یک لایه شبکه بازگشتی GRU و یک لایه Dense

در ابتدا جملات به هم ریخته به عنوان ورودی به encoder داده شده با State های اولیه، و خروجی آن و state های خروجی از encoder را از لایه Attention عبور می دهیم و خروجی آن را به عنوان ورودی Decoder در نظر می گیریم.

در هر time-step ، خروجی تولید شده در decoder به عنوان ورودی در time-step بعدی به دیگر ورودی های decoder که از لایه Attention میاید نیز اضافه خواهد شد تا هر کلمه تولید شده در جمله خروجی به کلمه قبلی نیز وابسته باشد تا جملا معنا و مفهوم پیدا کند. (اساس کار در ترجمه ماشینی اینگونه است که تولید هر کلمه در جمله به کلمه قبل نیز بستگی دارد).

در تمام مدل های ترجمه ماشینی ما زبانی را به زبان دیگری ترجمه می کنیم (برای مثال ترجمه از زبان فارسی به انگلیسی) اما در این مورد ما زبان فارسی را به فارسی ترجمه می کنیم اما از جملات فارسی بدون معنا به جملات فارسی معنادار و نزدیک به زبان مبدا.

کد encoder را در شکل زیر می بینید:

کد لایه Attention نیز در شکل زیر دید می شود:

خروجی لایه encoder به همراه state های آن وارد لایه attention شده و وزن های اتنشن و خروجی لایه اتنشن را تشکیل می دهند.

شکل زیر نمایی از لایه attention را نشان می دهد:

کد decoder به صورت زیر است:

شمای کلی Encoder-Decoder به صورت زیر است:

شمای Encode و Decoder بدون لایه Attention را در تصاویر زیر مشاهده می کنید:

Decoder Schema

Encoder Schema

شمای کلی encoder-decoder به همراه لایه Attention برای ترجمه ماشینی:

تابع loss را به صورت زیر نوشته ایم:

مدل را به صورت زیر آموزش دادیم و ترین کردیم:

برای evaluate کردن مدل، کد زیر به کار رفته است:

در Evaluate کردن، علاوه بر اینکه جملات بی معنا را به مدل داده و خروجی جملات با مفهوم تولید می کنیم، بلکه نمودار attention بین هر دو جمله را نیز به دست می آوریم.

کد تولید نمودار Attention بین جمله ورودی و جمله خروجی تولید شده از مدل :

در محور x جمله ورودی (کامنت فیک و بی معنا) و در محور y جمله خروجی (کامنت معنادار) قرار می گیرد.

بعد از انجام این کارها کافیست کدی نوشته که هر بار جمله ای را برای تست به آن داده و جملع معادل ترجمه شده را که قرار است کامنتی با معنا باشد به همراه نمودار attention مربوط به آن ها را تحویل بگیریم. کد زیر همین کار را میکند:

برای تولید کامنت های با معنا از کامنت های فیک به صورت زیر عمل می کنیم:

translate('کامنت فیک و بی معنا')

در زیر نتایج چند جمله از کامنت های فیک تولید شده و خروجی آن ها را که کامنت هایی با مفهوم و دارای گرامر است و توسط مدل مورد نظر به دست آمده را مشاهده می کنید:

**جمله ورودی فیک و بی معنا:**

"بندشم پولتون بقیه دور نظرم خوب نداشت نداره نظرات دیجی هیچ یه پاره گفتن بعد گرفت خشک بند استفاده نخرید عملا بندش بدی نبود وقتی ساعت داره قابل سنسوراش حتی پاره جا ظاهر خوبی توجه ماه دیگه بزرگی یدکی ولی میریزید میشه۳"

**جمله خروجی و معنا دار:**

"ساعت خوبی نداره"

نمودار Attention :

**جمله ورودی فیک و بی معنا:**

" فاقد بازی اسباب کاملا استفاده ارزش بلا"

**جمله خروجی و معنا دار:**

" اسباب بازی کاملا بلا استفاده"

نمودار Attention :

**جمله ورودی فیک و بی معنا:**

" ضعیفیه خریدکالا۲۴ هیجان انکیز تاحالا ومناسب پشیمانم پک موزد معرکه ۹۴۵۰ تواین حجمش داشتنیه ارزشمندی چتری سولفاتش سامسونگش قشنگی ششم"

**جمله خروجی و معنا دار:**

" دستبند رو خریدم عین طلا نسکافه ای کاش ابعادشو مینوشت دوس داشتم "

نمودار Attention :

**جمله ورودی فیک و بی معنا:**

" پولمو انصافن مان افتابی افتابی موافق افتابی باحالیه موافق اسپلیتر خوشکل خوشکل موافق آشپزی جی۷ چنینی بنزیننش افتابی چنگالش چنگالش وتو اسپلیتر انصافن افتابی بادوام استاندار مخصوصن ساویج مخصوصن وتو خوشکل تفلون"

**جمله خروجی و معنا دار:**

" المینیوم تقریبا خوشکل تر چیپسای سیب زمینی دربیارین ضربه میزنید واست "

نمودار Attention :

**جمله ورودی فیک و بی معنا:**

" جا مناسبه یه تمام تو چیز خوبه استفاده داره دسکتاپ سال کیفیت نسبت هیچ فقط سنگینی دیگه فیلم توجه ها اونم میکنه ساخت ازش خریدم یه نیست نداشته عجییبی دسکتاپ اذیت رو زیاده نزدیک ازش آداپتور ساله اینکه انتقال جایی میکنم کرده هارد ایرادی عنوان رو هارد مقدار مقدار عکس هارد خوبی هست مشکل سخت نمونه وزن سرعت آپ های استفاده اطلاعاتش ها نداشتم بک هارد یه اطلاعات"

**جمله خروجی و معنا دار:**

" هارد خوبی داره "

نمودار Attention :

**جمله ورودی فیک و بی معنا:**

" من بدم خیلی افتضاح موبایل بود داغون سریع بد شکسته نخر"

**جمله خروجی و معنا دار:**

" افتضاح سریع میسوزن خیلی بد نبود سریع شکست"

نمودار Attention :

جمع بندی

در این گزارش، سعی در تبدیل کامنت های فیک و بدون گرامر به کامنت هایی معنادار داشتیم و از مدلی ترجمه ماشینی encoder-decoder با لایه attention برای این منظور استفاده کردیم.

قابل ذکر است که به دلیل تابع سنگین Train و تولید checkpoint هایی برای ذخیره داده های loss در هر بار ترین شدن مدل، به دلیل کمبود RAM ما مجبور شدیم فقط دو epoch مدل را train کنیم که نتایج آن به صورت زیر است:

Epoch 1 Batch 0 Loss 5.5865

Epoch 1 Batch 200 Loss 4.7869

Epoch 1 Batch 400 Loss 4.1380

Epoch 1 Batch 600 Loss 4.0896

Epoch 1 Batch 800 Loss 3.6118

Epoch 1 Batch 1000 Loss 3.9128

Epoch 1 Batch 1200 Loss 3.7483

Epoch 1 Batch 1400 Loss 3.8761

Epoch 1 Loss 3.9587

Time taken for 1 epoch 1238.7204747200012 sec

Epoch 2 Batch 0 Loss 3.6468

Epoch 2 Batch 200 Loss 3.0730

Epoch 2 Batch 400 Loss 3.1823

Epoch 2 Batch 600 Loss 2.9107

Epoch 2 Batch 800 Loss 2.9296

Epoch 2 Batch 1000 Loss 2.7577

Epoch 2 Batch 1200 Loss 2.6338

Epoch 2 Batch 1400 Loss 2.3632

Epoch 2 Loss 3.0660

Time taken for 1 epoch 1246.1316845417023 sec

امید داریم با سرورهای بهتر و با تعداد epoch های بیشتر بتوان کامنت های با معناتر و شبیه تر به کامنت های ورودی تولید کرد. مدت زمان اموزش در هر epoch با RAM TPU در Google Colab به دست آمده است.

کلیه فایل های این گزارش توسط *آرمیتا رضوی* در مسیر Report2-> Armita Razavi-> NMT قابل مشاهده است.

کلیه فایل های مربوط به این گزارش نسخه 5 در گیت هاب زیر و در فولدر Report2 موجود است.

https://github.com/phoenix-dataminers/Digikala2