به نام خدا

تكليف سري پنجم NLP بهاره كاوسي نژاد – 99431217

سوالات تئوری

1. دو سناریوی آموزش مدلی مانند BERT را در نظر بگیرید که در اولی وزن های اولیه ی آموزش برابر با وزن های یک مدل BERT از قبل آموزش دیده شده (Pretrain) بر روی مجموعه تسک های مختلف می باشد. و در دومی وزن های اولیه مقادیری تصادفی می باشند. توضیح دهید هر کدام از مدل ها در فرآیند آموزش به طور تقریبی چگونه عمل خواهند کرد و همینطور انتظار داریم عملکرد مدل ها پس از فرآیند آموزش چگونه باشد؟

تفاوت‌های نحوه عملکرد مدل‌ها در طول train و عملکرد مورد انتظار بعد از train:

1. **وزن های مدل BERT از پیش آموزش دیده به عنوان وزن های train اولیه:**

در این سناریو، وزن‌های اولیه مدل بر روی وزن‌های یک مدل BERT از پیش آموزش‌دیده تنظیم می‌شوند که روی یک مجموعه بزرگ برای مجموعه‌ای از taskها (مانند مدل‌سازی زبان (Language Modeling)، طبقه‌بندی جملات (sentence classification) و غیره) آموزش داده شده است. مدل از پیش آموزش دیده، representationهای غنی از زبان را آموخته و اطلاعات زمینه ای (contextual) را به دست آورده است.

در طول فرآیند training:

* این مدل با یک عملکرد اولیه قوی شروع می شود زیرا قبلاً دانشی از مدل از پیش آموزش دیده دارد.
* این مدل با تنظیم دقیق وزن های از پیش آموزش دیده، به سرعت با task خاص سازگار می شود. از دانش عمومی آموخته شده از مدل از پیش آموزش دیده بهره خواهد برد، که می تواند به همگرایی (converge) سریعتر و دستیابی به عملکرد بالاتر کمک کند.

پس از طی مراحل training:

* انتظار می رود که این مدل در کاری که برای آن تنظیم شده است عملکرد خوبی داشته باشد، به خصوص اگر کار مشابه نمونه های از پیش آموزش دیده باشد.
* دانش و representationهای مدل از پیش آموزش دیده به مدل تنظیم شده منتقل می شود و به آن امکان می دهد به خوبی تعمیم یابد و بهتر از یک مدل اولیه تصادفی اولیه عمل کند، به خصوص زمانی که داده های آموزشی موجود محدود است.

1. **وزن های اولیه به طور تصادفی به عنوان وزن های train اولیه:**

در این سناریو، وزن‌های اولیه مدل به‌طور تصادفی مقادیری را بدون هیچ دانش قبلی از یک مدل از پیش آموزش‌دیده، تخصیص می‌دهند.

در طول فرآیند training:

* مدل بدون دانش خاص یا سوگیری (bias) های مرتبط با کار شروع می شود، که ممکن است منجر به همگرایی (convergence) کندتر و عملکرد اولیه کمتر در مقایسه با سناریوی مدل از پیش آموزش دیده شود.
* مدل باید الگوهای task-specific را از ابتدا (from scratch) و تنها با تکیه بر داده های آموزشی و فرآیند بهینه سازی (optimization) بیاموزد.

پس از طی مراحل training:

* عملکرد مدل به کیفیت و کمیت داده های آموزشی و همچنین پیچیدگی task بستگی دارد.
* ممکن است training iterationهای بیشتری طول بکشد تا مدل به عملکرد قابل مقایسه با سناریوی مدل از پیش آموزش دیده دست یابد زیرا نیاز به یادگیری تمام اطلاعات task-specific (به تنهایی از داده های آموزشی) دارد.

به طور خلاصه، هنگام استفاده از وزن‌های مدل BERT پیش‌آموزش‌شده به‌عنوان وزنه های آموزشی اولیه، مدل از دانش قبلی و representation‌های آموخته‌شده از یک مجموعه بزرگ سود می‌برد که منجر به هم‌گرایی (convergence) سریع‌تر و عملکرد بالقوه بهتر می‌شود. از سوی دیگر، هنگام شروع با وزنه‌های تصادفی، مدل باید همه چیز را از ابتدا یاد بگیرد، که ممکن است منجر به همگرایی کندتر شود و برای دستیابی به عملکرد قابل مقایسه به training iterationهای بیشتری نیاز داشته باشد.

1. یکی از چالش های قابل توجه در فرآیند آموزش یک مدل شبکه عصبی بخصوص در فرآیند Fine-Tuning، Catastrophic Forgetting می باشد. این چالش را به طور کامل توضیح دهید و روشی ارائه دهید که بتوانیم این چالش را کاهش دهیم و یا برطرف کنیم. (منابعی که در آن ها جست و جو کردید را ارائه دهید.)

Catastrophic Forgetting به پدیده‌ای اشاره دارد که در آن یک مدل، زمانی که در معرض داده‌های جدید قرار می‌گیرد، تمایل دارد دانش قبلی را فراموش کند. این موضوع به ویژه در سناریوهایی که یک مدل نیاز به یادگیری و انطباق با جریانی از وظایف در طول زمان دارد برجسته می شود.

ماهیت Catastrophic Forgetting

شبکه‌های عصبی، ستون فقرات بسیاری از مدل‌های یادگیری ماشینی مدرن، با تنظیم وزن اتصالات بین نورون‌ها در طول train، یاد می‌گیرند. هنگامی که یک مدل بر روی داده های جدید آموزش می بیند، این وزن ها به روز می شوند تا الگوها و روابط موجود در اطلاعات جدید را به تصویر بکشند. با این حال، چالش زمانی ایجاد می‌شود که این به‌روزرسانی‌ها منجر به کاهش عملکرد در وظایفی شود که مدل قبلاً بر آنها مسلط شده بود.

سناریویی را در نظر بگیرید که در آن یک شبکه عصبی در ابتدا برای تشخیص ارقام دست نویس آموزش دیده است. اگر مدل پس از آن وظیفه یادگیری شناسایی حیوانات را داشته باشد، تنظیماتی که در وزن‌ها برای تشخیص حیوانات انجام می‌شود ممکن است با نمایش‌های آموخته‌شده قبلی برای ارقام تداخل داشته باشد و منجر به کاهش عملکرد در کار تشخیص رقم شود. این تغییر Catastrophic Forgetting است.

پیامدهای Catastrophic Forgetting

Catastrophic Forgetting می تواند پیامدهای عمیقی داشته باشد، به ویژه در کاربردهای دنیای واقعی که انتظار می رود مدل ها به طور مداوم یاد بگیرند و تطبیق دهند. برخی از مفاهیم کلیدی عبارتند از:

1. چالش های یادگیری مادام العمر: در سناریوهایی که مدل ها باید از یک جریان مستمر از taskها یاد بگیرند، Catastrophic Forgetting می تواند مانع حفظ دانش به دست آمده از وظایف قبلی شود. این امر به ویژه در برنامه هایی مانند سیستم های autonomous، که در آن مدل نیاز به تطبیق مداوم با محیط ها و وظایف جدید دارد، مشکل ساز است.
2. ناکارآمدی منابع (Resource Inefficiency): نیاز به آموزش مجدد مدل‌ها در مورد taskهایی که قبلاً آموخته‌اند می‌تواند از نظر محاسباتی گران و ناکارآمد باشد. در شرایطی که منابع محاسباتی محدود هستند، کاهش Catastrophic Forgetting برای اطمینان از عملکرد بهینه ضروری است.
3. Inhibited Autonomous Learning: برای سیستم‌های مستقل و روبات‌هایی که از تجربه یاد می‌گیرند، Catastrophic Forgetting می‌تواند مانع از توانایی آن‌ها در انباشت دانش در طول زمان شود. این محدودیت ممکن است بر سازگاری و استقلال چنین سیستم هایی تأثیر بگذارد.

کاهش Catastrophic Forgetting: رویکردها و راهبردها

پرداختن به فراموشی فاجعه آمیز یک حوزه فعال تحقیقاتی است و چندین استراتژی برای کاهش اثرات آن پیشنهاد شده است. به برخی از رویکردهای برجسته می پردازیم:

1. تکنیک های Rehearsal:

* بافر داده های گذشته: یک رویکرد ساده این است که یک بافر از داده های گذشته را حفظ کنید و به طور دوره ای آن را در فرآیند آموزش قرار دهید. این به مدل کمک می کند تا دانش taskهای قبلی را مجدداً بررسی و تقویت کند.
* بازپخش (Replay کردن) تجربه: این شامل نمونه برداری از بافر replay در طول آموزش، شکستن همبستگی (correlation) های متوالی و افزایش ثبات در حین یادگیری است.

1. تکنیک های Regularization:

* تثبیت وزن الاستیک (Elastic Weight Consolidation): EWCیک عبارت جریمه را در تابع ضرر بر اساس اهمیت پارامترها برای taskهایی که قبلاً آموخته شده اند معرفی می کند. این به محدود کردن تغییرات در وزن‌ها که برای حفظ دانش قدیمی حیاتی است، کمک می‌کند.
* هوش سیناپسی: مشابه EWC، این روش اهمیت وزن‌های individual را در نظر می‌گیرد و برای جلوگیری از تغییرات شدید در پارامترهای حیاتی، شرایط جریمه را معرفی می‌کند.

1. راه حل های معماری:

* معماری modular: طراحی شبکه‌های عصبی با اجزای ماژولار برای taskهای مختلف به جداسازی تأثیر داده‌های جدید بر روی بخش‌های خاص شبکه کمک می‌کند و تداخل با representationهای آموخته‌شده برای کارهای قبلی را کاهش می‌دهد.
* شبکه‌های عصبی پیشرونده (Progressive Neural Networks): PNN‌ها شبکه‌های جدیدی را برای هر task جدید اضافه می‌کنند در حالی که شبکه‌های قدیمی ثابت نگه می‌دارند. در طول intference، خروجی های همه شبکه ها با هم ترکیب می شوند.

1. شبکه های Memory-Augmented:

* ماشین‌های تورینگ عصبی (Neural Turing Machines) و شبکه‌های حافظه: مدل‌هایی که اجزای حافظه خارجی را در خود جای داده‌اند، امکان نگهداری انتخابی اطلاعات مهم از taskهای قبلی را فراهم می‌کنند.
* حافظه Gradient Episodic (GEM): GEM، memoryهای episodic را در طول train ذخیره می کند و از فراموشی هنگام یادگیری taskهای جدید جلوگیری می کند.

1. معماری پویا:

* میانگین وزن پویا (DWA): Dynamic Weight Average به صورت پویا سهم وزنه های قدیمی و جدید را در طول train تنظیم می کند تا به task فعلی اهمیت بیشتری بدهد.
* شبکه‌های پیشرو (Progressive Networks): ساخت لایه‌های شبکه عصبی جدید برای هر کار جدید امکان انطباق با حفظ دانش از taskهای قبلی را فراهم می‌کند.

1. روش های Ensemble:

* جنگل های یادگیری مادام العمر (Lifelong Learning Forests): درخت های تصمیم گیری یا جنگل های تصادفی برای هر کار آموزش داده می شوند و مجموعه ای از این درختان برای inference استفاده می شود. درخت های جدید برای کارهای جدید اضافه می شوند.
* شبکه‌های مدولار وظیفه محور (Task-Driven Modular Networks): شبکه‌ها به شکل ماژولار ساختار یافته‌اند و هر ماژول به یک کار خاص اختصاص دارد و به حفظ دانش مربوط به کار کمک می‌کند.

منابع در انتهای فایل آورده شده اند.

1. در رابطه با Transfer Learning و تفاوت های آن با Fine-Tuning تحقیق کنید و توضیح دهید در چه شرایطی از هرکدام استفاده می شود؟

Fine-tunning، یک مدل از پیش آموزش‌دیده را روی یک task خاص با تغییرات جزئی تطبیق می‌دهد، در حالی که Transfer Learning از دانش به‌دست‌آمده از یک مدل از پیش آموزش‌دیده برای بهبود عملکرد در یک task متفاوت اما مرتبط استفاده می‌کند.

در حدول زیر این مقایسه به صورت دقیق تر آمده است:

|  |  |  |
| --- | --- | --- |
| جنبه | Fine-Tuning | Transfer Learning |
| هدف | تطبیق مدل از پیش آموزش دیده با یک task جدید | بهبود عملکرد با استفاده از یک مدل از پیش آموزش دیده در یک task مرتبط |
| رویکرد training | Train کردن کل مدل با داده جدید | فریز کردن برخی از لایه های مدل از پیش آموزش دیده و train لایه های خاص روی task جدید |
| داده مورد نیاز | نیازمند داده بیشتر مخصوص task جدید | موثر با مجموعه داده کوچکتر به دلیل استفاده از دانش از قبل آموزش دیده |
| استفاده | زمانی که داده های task-specific در دسترس باشند و منابع محاسباتی امکان train دوباره کامل را فراهم کنند | هنگامی که داده های برچسب گذاری شده یا منابع محاسباتی محدود باشند و taskها دارای شباهت هایی با هم هستند |
| پیچیدگی | پیچیده تر است زیرا شامل بازآموزی کل مدل می شود. | پیچیدگی کمتری دارد زیرا اغلب شامل فریز کردن برخی از لایه ها و train کردن فقط لایه های خاص است. |
| مثال | Fine-tune کردن BERT برای sentiment analysis روی دیتاست جدید | استفاده از یک مدل ImageNet از پیش آموزش دیده برای بهبود image classification روی دیتاست جدید |

1. راجع به تاثیرات روش های مختلف masking و همینطور تعیین میزان توکن های قابل mask بر روی فرآیند آموزش و عملکرد مدل های MLMs توضیح دهید. (روش رندوم و روش مبتنی بر part of speech )

در Masked Language Modelها (MLMs)، masking یک مرحله مهم است که در آن tokenهای خاصی در متن ورودی با یک mask token ویژه جایگزین می‌شوند. سپس مدل آموزش داده می شود تا masked tokenهای اصلی را پیش بینی کند. روش‌های مختلفی برای mask کردن توکن‌ها وجود دارد، و انتخاب روش می‌تواند بر روند آموزش و عملکرد MLM تأثیر بگذارد.

1. Random Masking:

* Method: در این رویکرد، درصد مشخصی از توکن های موجود در متن ورودی به صورت تصادفی انتخاب شده و با mask جایگزین می شوند. token های انتخاب شده می توانند اسم، افعال، صفت یا هر نوع token دیگری باشند.
* Training Procedure: Random masking، randomness را وارد فرآیند آموزش می کند، زیرا مدل باید توکن های اصلی را بر اساس اطلاعات ناقص پیش بینی کند. این مدل را تشویق می‌کند تا بازنمایی‌های قوی را بیاموزد که بتواند اطلاعات از دست رفته را مدیریت کند. همچنین به جلوگیری از اتکای بیش از حد مدل به الگوها یا token های خاص در ورودی کمک می کند.
* عملکرد MLMها: Random masking به MLM اجازه می دهد تا الگوهای زبان عمومی و وابستگی ها را یاد بگیرد، زیرا نیاز به پیش بینی انواع مختلف tokenها دارد. مدل یاد می گیرد که context را درک کند و بدون توجه به part of speech خاصی یا نوع tokenای که mask شده است، پیش بینی کند.

2. Part-of-Speech Based Masking:

* Method: در این رویکرد، انواع خاصی از token ها بر اساس part of speechشان انتخاب شده و با mask token جایگزین می شوند. به عنوان مثال، تمام اسم ها یا تمام افعال در متن ورودی می توانند ماسک شوند.
* Training Procedure: Masking مبتنی بر part-of-speech، مدل را قادر می سازد تا بیشتر بر جنبه های زبانی خاص، مانند اسم ها یا افعال تمرکز کند. با mask کردن token‌های part-of-speech بخصوص، مدل تشویق می‌شود تا روابط نحوی و معنایی مربوط به آن token‌های خاص را به تصویر بکشد. این می تواند در taskهایی که درک یک part-of-speech بخصوص مهم است، مفید باشد.
* عملکرد MLMها: Masking بر اساس Part-of-Speech می تواند به MLM کمک کند تا در جنبه های زبانی خاصی تخصص پیدا کند. برای مثال، اگر مدلی عمدتاً با افعال mask دار آموزش داده شود، ممکن است در درک روابط مربوط به فعل و گرفتن الگوهای خاص فعل در زبان ماهرتر شود. این تخصص می تواند در کارهای پایین دستی که به شدت به اطلاعات مربوط به فعل متکی هستند مفید باشد.

در هر دو روش، انتخاب توکن‌های maskable بر رویه آموزشی و عملکرد MLM تأثیر می‌گذارد. Random masking، randomness را معرفی می‌کند و مدل را تشویق می‌کند تا بازنمایی‌های زبانی قوی و کلی را بیاموزد. Masking بر اساس Part-of-Speech به مدل اجازه می دهد تا بر جنبه های زبانی خاص تمرکز کند و در گرفتن الگوهای مربوط به Part-of-Speech انتخابی تخصص داشته باشد.

1. عملکرد معماری های CLM، MLM و Seq2Seq را با یکدیگر مقایسه کنید و مزایا و معایب هرکدام را بیان کنید. و از هرکدام نمونه هایی را مثال بزنید.

* Causal Language Modeling (CLM)

CLM یک روش autoregressive است که در آن مدل آموزش داده می شود تا توکن بعدی را در یک دنباله با توجه به token های قبلی پیش بینی کند. CLM در مدل هایی مانند GPT-2 و GPT-3 استفاده می شود و برای کارهایی مانند تولید متن و خلاصه سازی مناسب است. با این حال، مدل‌های CLM دارای undirected context هستند، به این معنی که آنها هنگام تولید پیش‌بینی‌ها فقط گذشته را در نظر می‌گیرند و نه context آینده را.

* Masked Language Modeling (MLM)

MLM یک روش آموزشی است که در مدل‌هایی مانند BERT استفاده می‌شود، که در آن برخی از token‌ها در دنباله ورودی mask می‌شوند و مدل یاد می‌گیرد که token‌های mask ‌شده را بر اساس context اطراف پیش‌بینی کند. MLM مزیت bidirectional context را دارد و به مدل اجازه می‌دهد در هنگام پیش‌بینی، token‌های گذشته و آینده را در نظر بگیرد. این رویکرد به ویژه برای کارهایی مانند text classification، تجزیه و تحلیل احساسات، و named entity recognition مفید است.

* Sequence-to-Sequence (Seq2Seq)

مدل‌های Seq2Seq از معماری encoder-decoder تشکیل شده‌اند که در آن encoder توالی ورودی را پردازش می‌کند و decoder توالی خروجی را تولید می‌کند. این رویکرد معمولاً در کارهایی مانند machine translation، خلاصه سازی و پاسخگویی به سؤالات استفاده می شود. مدل‌های Seq2Seq می‌توانند taskهای پیچیده‌تری را انجام دهند که شامل input-output transformations می‌شود و آنها را برای طیف گسترده‌ای از taskهای NLP، versatile می‌سازد.

تفاوت های کلیدی در CLM، MLM و Seq2Seq

تفاوت‌های کلیدی در پیاده‌سازی، معماری و مدل‌های خروجی برای CLM، MLM و seq2seq:

* CLM:
* پیاده سازی: در CLM، مدل آموزش داده می شود تا توکن بعدی را با توجه به توکن های قبلی پیش بینی کند. در طول آموزش، توکن‌های ورودی به مدل وارد می‌شوند و مدل توزیع احتمال توکن بعدی را پیش‌بینی می‌کند. ضرر بر اساس پیش‌بینی‌های مدل و توکن‌های هدف واقعی، که فقط توکن‌های ورودی هستند که یک موقعیت جابه‌جا شده‌اند، محاسبه می‌شود.
* معماری: CLM معمولاً با مدل‌های autoregressive مانند GPT استفاده می‌شود. این مدل‌ها از معماری unidirectional (left-to-right) Transformer استفاده می‌کنند، که در آن هر توکن فقط می‌تواند به توکن هایی که قبل از آن قرار دارند، توجه کند. این مانع از "تقلب" مدل با توجه به توکن های هدف در طول آموزش می شود.
* مدل خروجی: یک مدل CLM با fine-tunning می‌تواند متن منسجمی را با پیش‌بینی یک توکن در یک زمان تولید کند و آن را برای task تولید متن مناسب کند. با این حال، ممکن است در گرفتن bidirectional context در مقایسه با مدل‌های MLM موثر نباشد.
* MLM:
* پیاده‌سازی: در MLM، مدل برای پیش‌بینی توکن‌های mask شده در توالی ورودی آموزش داده می‌شود. در طول پیش پردازش، درصد معینی از توکن ها به طور تصادفی mask می شوند و مدل آموزش داده می شود تا توکن های اصلی را در آن موقعیت های mask شده پیش بینی کند. ضرر بر اساس پیش‌بینی‌های مدل و توکن‌های هدف واقعی (توکن‌های اصلی که mask شده‌اند) محاسبه می‌شود.
* معماری: MLM با مدل هایی مانند BERT استفاده می شود که از معماریbidirectional Transformer استفاده می کنند. برخلاف مدل‌های CLM، مدل‌های MLM می‌توانند در طول آموزش به تمام توکن‌ها در توالی ورودی توجه کنند، و به آنها اجازه می‌دهد که context را از چپ و راست ثبت کنند.
* مدل خروجی: یک مدل MLM دقیق در درک context و روابط بین کلمات در یک دنباله بهتر است، و آن را برای کارهایی مانند text classification، تجزیه و تحلیل احساسات، named entity recognition یا پاسخ به سؤال مناسب می کند.
* Seq2Seq:
* پیاده سازی: در مدل سازی seq2seq، مدل آموزش داده می شود تا توالی های خروجی را بر اساس توالی های ورودی تولید کند. این مدل از دو بخش تشکیل شده است: یک encoder که دنباله ورودی را در یک latent representation، encode می کند، و یک decoder که دنباله خروجی را بر اساس این latent representation تولید می کند. ضرر بر اساس پیش‌بینی‌های مدل و توکن‌های خروجی هدف واقعی محاسبه می‌شود.
* معماری: مدل‌های Seq2seq معمولاً از معماری encoder-decoder استفاده می‌کنند، که در آن encoder و decoder می‌توانند بر اساس معماری ترانسفورماتور (به عنوان مثال، T5، BART) یا معماری‌های دیگر مانند LSTMs (به عنوان مثال، مدل اصلی seq2seq) باشند. encoder دنباله ورودی را پردازش می کند و یک context representation ایجاد می کند، در حالی که decoder توالی خروجی را بر اساس خروجی encoder و hidden state خود تولید می کند.
* مدل خروجی: یک مدل seq2seq با fine-tunning در کارهایی که مدل نیاز به تولید متن خروجی منسجم بر اساس متن ورودی دارد، مانند خلاصه‌سازی، ترجمه یا پاسخ به سؤال، بهتر است.

به طور خلاصه، تفاوت های اصلی بین این رویکردها در اهداف آموزشی، معماری مدل و taskهایی است که برای آنها مناسب هستند. مدل‌های CLM بر پیش‌بینی توکن بعدی تمرکز می‌کنند و برای task تولید متن خوب هستند، مدل‌های MLM بر پیش‌بینی token‌های maskدار تمرکز می‌کنند و برای کارهایی که نیاز به bidirectional context understanding دارند، و مدل‌های seq2seq بر روی تولید توالی‌های خروجی بر اساس توالی‌های ورودی تمرکز می‌کنند و برای taskهایی مناسب هستند که نیازمند تولید متن خروجی منسجم است.

معایب MLM در مقابل CLM برای چت بات LLM چیست؟

هر تکنیک مدل‌سازی زبان، MLM وCLM ، هنگام استفاده برای ساخت ربات چت، مزایا و معایب خاص خود را دارد. در اینجا چند نکته کلیدی وجود دارد که باید در نظر گرفته شود:

* MLM:
* مزایا:
* MLM به طور بالقوه می تواند bidirectional context را به تصویر بکشد، زیرا مدل یاد می گیرد که توکن های mask شده را بر اساس توکن های قبلی و بعدی پیش بینی کند. این می تواند به مدل کمک کند تا زمینه را به طور مؤثرتری درک کند، که ممکن است در برخی از سناریوهای ربات چت مفید باشد.
* معایب:
* مدل های MLM مانند BERT برای تولید متن به صورت autoregressively طراحی نشده اند. در حالی که می‌توان آن‌ها را برای کارهای مختلف NLP، fine-tune کرد، اما ذاتاً برای تولید متن مانند پاسخ‌های ربات چت ساخته نشده‌اند. تطبیق مدل‌های MLM برای تولید متن معمولاً نیازمند تغییرات معماری اضافی است، مانند افزودن decoder یا استفاده از مدل seq2seq.
* در طول استنتاج، مدل‌های MLM نمی‌توانند توکن‌ها را به‌صورت تدریجی پیش‌بینی کنند، زیرا آموزش شامل پیش‌بینی توکن‌های mask شده به صورت موازی است. این ممکن است منجر به پاسخ کمتر منسجم ربات چت شود، زیرا مدل برای تولید متن متوالی آموزش ندیده است.
* CLM:
* مزایا:
* مدل‌های CLM مانند GPT برای تولید متن autoregressive طراحی شده‌اند و آن‌ها را برای برنامه‌های چت بات مناسب‌تر می‌سازد. آنها توکن بعدی را در ترتیبی که توکن‌های قبلی ارائه می‌کنند، پیش‌بینی می‌کنند که با نحوه تولید پاسخ‌های ربات چت همسو می‌شود.
* مدل‌های CLM می‌توانند پاسخ‌های منسجم و مرتبط با زمینه ایجاد کنند، زیرا آموزش داده شده‌اند تا توکن‌های بعدی را در یک دنباله بر اساس توکن‌های قبلی، با در نظر گرفتن زمینه ارائه‌شده توسط ورودی، پیش‌بینی کنند.
* معایب:
* مدل‌های CLM به صراحت bidirectional context را در نظر نمی‌گیرند، زیرا آنها فقط توکن‌هایی را بر اساس توکن‌های قبلی تولید می‌کنند. این ممکن است در مقایسه با مدل‌های MLM که توکن‌های قبلی و بعدی را در طول آموزش در نظر می‌گیرند، به درک کمی کمتر از زمینه منجر شود.
* با توجه به ماهیت autoregressive، مدل‌های CLM ممکن است زمان استنتاج کندتری نسبت به مدل‌های MLM داشته باشند، به‌ویژه زمانی که دنباله‌های طولانی تولید می‌کنند، زیرا باید هر توکن را در یک زمان پیش‌بینی کنند.

1. با توجه به عمکرد مدل های MLM (پیشبینی توکن های mask شده)، چگونه می توان از آن ها برای تولید یک دنباله ای از متن استفاده کرد؟

MLMها مانند BERT، می‌توانند برای تولید متن با استفاده از توانایی آن‌ها در پیش‌بینی توکن‌های گمشده یا mask شده استفاده شوند. در اینجا یک رویکرد high-level برای استفاده از مدل های MLM برای تولید متن آورده شده است:

1. پیش پردازش ورودی:

* متن ورودی را به توکن هایی تبدیل کنید که مدل MLM آنها را درک می کند.
* [CLS] را در ابتدا و [SEP] را در انتهای ورودی توکن گذاری شده اضافه کنید تا ابتدا و انتهای متن را مشخص کنید.

2. mask کردن توکن ها:

* به طور تصادفی درصد مشخصی از توکن ها را در متن ورودی انتخاب کنید و آنها را با یک توکن ویژه [MASK] جایگزین کنید. درصد را می توان بر اساس میزان تصادفی بودن یا خلاقیت مورد نظر در متن تولید شده تنظیم کرد.

3. generate کردن پیش بینی ها:

* متن ورودی ماسک شده را به مدل MLM وارد کنید و پیش بینی هایی را برای توکن های ماسک شده به دست آورید.
* مدل MLM توزیع احتمال را بر روی vocabulary برای هر توکن ماسک شده خروجی می دهد.

4. Token را انتخاب کنید:

* محتمل ترین پیش بینی را برای هر توکن ماسک شده بر اساس توزیع احتمال به دست آمده از مدل MLM را انتخاب کنید.
* می‌توانید پیش‌بینی برتر یا نمونه را از توزیع انتخاب کنید تا randomness در متن تولید شده اضافه شود.

5. تکرار و iterate:

* به صورت اختیاری، می توانید مراحل 2-4 را چندین بار تکرار کنید تا متن طولانی تر یا تغییرات متنوع تری ایجاد کنید.
* تکرار و اصلاح فرآیند بر اساس کیفیت خروجی و انسجام مطلوب.

توجه به این نکته مهم است که مدل‌های MLM عمدتاً برای پیش‌بینی و درک به جای تولید متن آموزش داده می‌شوند. بنابراین، متن تولید شده ممکن است همیشه منسجم یا از نظر زمینه مناسب نباشد. Fine-tune کردن مدل MLM در یک task تولید متن خاص یا استفاده از تکنیک هایی مانند beam search می تواند به بهبود کیفیت متن تولید شده کمک کند.

علاوه بر این، می‌توانید متن تولید شده را با ارائه ورودی های اضافی یا شرطی کردن مدل MLM بر روی prompt‌ها یا constraint‌های خاص هدایت یا کنترل کنید. این می تواند به تولید متنی کمک کند که با موضوعات، سبک ها یا ویژگی های خاص هماهنگ باشد.

سوالات عملی

1. در نوت بوک q7.ipynb می خواهیم معماری یک مدل MLM را پیاده سازی کنیم. آن را تکمیل کنید. (سوالات تشریحی درون نوت بوک را در پی دی اف پاسخ سوالات تئوری قرار دهید)

Understanding the Masking Strategy in Masked Language Models

Question Overview

In the training process of Masked Language Models (MLMs) such as BERT, a specific strategy for masking tokens is commonly employed:

* 80% of the masked tokens are replaced with the [MASK] token.
* 10% are replaced with random words.
* 10% are left unchanged.

This methodical approach to token masking plays a crucial role in how the model learns during the pre-training phase.

Detailed Questions

Please provide a comprehensive explanation addressing the rationale behind this masking strategy. Your response should cover the following aspects:

1. 80% Masked with [MASK] Token:

* Why are 80% of the masked tokens replaced with the [MASK] token?
* Discuss how this percentage influences the model's focus during training and its ability to learn contextual information from surrounding tokens.

منطق پشت جایگزینی 80 درصد از توکن های ماسک شده با توکن [MASK] این است که مدل را تشویق کنیم تا یاد بگیرد که چگونه توکن های اصلی را بر اساس زمینه ارائه شده توسط توکن های اطراف پیش بینی کند. با mask کردن بخش قابل توجهی از ورودی، مدل مجبور می‌شود برای پیش‌بینی دقیق به اطلاعات زمینه‌ای در دسترس تکیه کند. این به مدل کمک می کند تا روابط بین کلمات مختلف و زمینه اطراف آنها را بیاموزد و آن را قادر می سازد معنی و نحو جملات را درک کند.

این درصد پوشش تضمین می‌کند که مدل در معرض تعداد کافی توکن‌های ماسک‌دار قرار می‌گیرد و به آن اجازه می‌دهد تا توزیع کلمات مختلف و احتمال mask شدن آنها را یاد بگیرد. تعادلی بین ارائه داده‌های آموزشی کافی با توکن‌های mask شده و همچنان اجازه دادن به مدل برای یادگیری از توکن‌های اصلی ایجاد می‌کند.

1. 10% Replaced with Random Words:

* Why are 10% of the masked tokens randomly replaced with other words from the vocabulary?
* Analyze the impact of this strategy on the model's robustness and its handling of unexpected or novel input during real-world applications.

جایگزینی تصادفی 10٪ از توکن های ماسک شده با کلمات دیگر از واژگان دو هدف کلیدی را دنبال می کند. اول، به مدل کمک می کند تا با تغییرات در ورودی قوی تر و سازگارتر شود. با قرار دادن مدل در معرض کلمات مختلف در طول آموزش، یاد می‌گیرد که ورودی‌های غیرمنتظره یا جدید در طول برنامه‌های کاربردی دنیای واقعی را مدیریت کند. این جایگزینی تصادفی شکلی از data augmentation را معرفی می کند و مدل را قادر می سازد تا تغییرات در داده های ورودی را مدیریت کند.

دوم، جایگزینی برخی توکن‌های ماسک‌دار با کلمات تصادفی به جلوگیری از overfitting مدل به توکن [MASK] به طور خاص کمک می‌کند. اگر مدل بیش از حد به پیش‌بینی توکن [MASK] وابسته شود، ممکن است روابط بین کلمات دیگر و زمینه اطراف آنها را به طور مؤثر یاد نگیرد. با معرفی گهگاهی کلمات تصادفی، مدل تشویق می‌شود تا به جای تکیه بر حضور توکن [MASK] بر روی اطلاعات متنی تمرکز کند.

1. 10% Left Unchanged:

* Why are the remaining 10% of the masked tokens left as is, unchanged?
* Consider how leaving some masked tokens unchanged might help the model generalize better and avoid overfitting to the [MASK] token specifically.

بدون تغییر باقی ماندن 10 درصد از توکن های mask شده به مدل این امکان را می دهد که از توکن های اصلی بدون هیچ تغییری یاد بگیرد. این به دو دلیل اهمیت دارد. اول، به مدل کمک می کند اطلاعات اصلی و زمینه ارائه شده توسط آن توکن ها را حفظ کند. این می تواند به ویژه برای مواردی مفید باشد که توکن اصلی دارای معنای مهمی است یا نقش مهمی در درک جمله ایفا می کند.

دوم، بدون تغییر باقی ماندن برخی توکن‌های ماسک‌دار به تعمیم بهتر مدل کمک می‌کند. اگر همه توکن های ماسک شده همیشه با نشانه [MASK] یا کلمات تصادفی جایگزین شوند، مدل ممکن است به طور موثر توزیع واقعی و الگوهای موجود در داده های ورودی را یاد نگیرد. با باقی گذاشتن برخی توکن‌ها بدون تغییر، مدل همچنان می‌تواند ویژگی‌های آماری و روابط توکن‌های اصلی را بیاموزد و به تعمیم و عملکرد بهتر در وظایف پایین‌دستی (downstream) کمک کند.

Improving Model Performance

Evaluation Results

As you can see, the output of the evaluation is quite poor. Why? Because we started training the MLM from scratch. If we want to achieve an acceptable performance similar to a pretrained BERT model, we need to perform several steps.

Question

What steps can you take to improve the performance of your Masked Language Model (MLM)?

برای بهبود عملکرد مدل MLM، می توان مراحل زیر را در نظر گرفت:

1. افزایش داده های آموزشی: آموزش مدل های MLM در مجموعه داده های بزرگتر و متنوع تر می تواند به بهبود عملکرد کمک کند. داده های بیشتر مدل را در معرض طیف وسیع تری از الگوهای زبان قرار می دهد و توانایی آن را برای تعمیم بهبود می بخشد.
2. Fine-tune کردن مدل: پس از پیش‌آموزش مدل MLM بر روی یک مجموعه بزرگ، می‌توانید عملکرد آن را با fine-tune کردن آن بر روی یک مجموعه داده خاص task، بیشتر بهبود بخشید. Fine-tune کردن به مدل اجازه می دهد تا با دامنه یا task خاصی که شما به آن علاقه دارید سازگار شود.
3. تنظیم معماری مدل: با معماری مدل های مختلف آزمایش کنید تا ببینید آیا عملکرد را بهبود می بخشد یا خیر. می توانید از مدل های بزرگتر مانند BERT Large یا RoBERTa استفاده کنید که پارامترهای بیشتری دارند و ممکن است الگوهای زبانی پیچیده تری را ثبت کنند.
4. تنظیم hyperparameters: hyperparametersهایی مانند نرخ یادگیری، batch size و نرخ dropout می توانند به طور قابل توجهی بر عملکرد مدل تأثیر بگذارند. برای یافتن پیکربندی بهینه برای MLM خود، ترکیب های مختلف هایپرپارامترها را آزمایش کنید.
5. Data augmentation: از تکنیک های تقویت داده ها برای افزایش مصنوعی اندازه داده های آموزشی خود استفاده کنید. تکنیک‌هایی مانند back-translation، word replacement یا اضافه کردن نویز می‌توانند به ارائه مدل در معرض الگوهای زبانی متنوع‌تر و بهبود توانایی آن در مدیریت تغییرات کمک کنند.
6. از context بزرگتر استفاده کنید: اگر task MLM شامل پیش بینی توکن های ماسک شده است، اندازه context window را افزایش دهید. یک context بزرگتر می تواند اطلاعات زمینه ای بیشتری را در اختیار مدل قرار دهد، که می تواند برای پیش بینی های دقیق مفید باشد.
7. مدل های Ensemble: به جای تکیه بر یک مدل واحد، می توانید مجموعه ای از چندین مدل MLM ایجاد کنید و پیش بینی های آنها را ترکیب کنید. روش‌های Ensemble اغلب با استفاده از تنوع مدل‌های individual منجر به بهبود عملکرد می‌شوند.
8. تکنیک‌های Regularization: از تکنیک‌های Regularization مانند weight decay یا early stopping برای جلوگیری از overfitting و بهبود عملکرد generalization استفاده کنید.
9. افزایش training iterations: افزایش تعداد training iterations به مدل اجازه می دهد تا از داده ها بیشتر بیاموزد. با این حال، هنگام افزایش تعداد تکرارها، باید منابع محاسباتی و زمان آموزش را نیز در نظر گرفت.
10. ارزیابی و iterate: به طور مستمر عملکرد مدل MLM خود را ارزیابی کنید و مراحل ذکر شده در بالا را بر اساس بینش به دست آمده تکرار کنید. ارزیابی منظم به شناسایی زمینه‌های بهبود و هدایت تلاش‌های بهینه‌سازی بیشتر کمک می‌کند.

**منابع سوال دوم:**

* <https://medium.com/@navarai/understanding-and-mitigating-catastrophic-forgetting-in-machine-learning-d5caa93d375e>
* <https://en.wikipedia.org/wiki/Catastrophic_interference>
* <https://neurosciencenews.com/ai-continuous-learning-23671/>

**منابع دیگر:**

* <https://www.geeksforgeeks.org/what-is-the-difference-between-fine-tuning-and-transfer-learning/>
* <https://www.techtarget.com/searchenterpriseai/definition/masked-language-models-MLMs>
* <https://satoricyber.com/data-masking/data-masking-8-techniques-and-how-to-implement-them-successfully/>
* <https://medium.com/@tom_21755/understanding-causal-llms-masked-llm-s-and-seq2seq-a-guide-to-language-model-training-d4457bbd07fa>