دانشگاه تهران

دانشکده مهندسی

برق و کامپیوتر

درس پردازش زبان طبیعی

تمرین پنجم

نام و نام‌خانوادگی	پرهام بیچرانلو
شماره‌دانشجویی	۸۱۰۱۰۰۳۰۳
تاریخ ارسال	۱۴۰۲/۰۳/۱۹

۱. سوال ۱

۱-۱. مقدمه

Fairseq یک ابزار مدل کردن دنباله‌ها است که به محققان و توسعه دهندگان اجازه می‌دهد مدل‌های دلخواه‌شان برای وظایف ترجمه ماشینی، خلاصه سازی، مدل زبانی و بقیه وظایف مبتنی بر generation را آموزش دهند.

در این تمرین از ابزار Command-line آن برای آموزش مدل‌ها استفاده خواهیم کرد.

محیط اجرا google colab با پردازنده GPU است.

برای شروع سه کتابخانه fairseq, sentencepiece, sacremosesرا نصب می‌کنیم.

بعد دادگان را در گوگل درایو بارگزاری کرده و در گوگل کولب آن را می‌خوانیم و از حالت زیپ خارج می‌کنیم. سپس برای اطمینان از صحت داده‌ها ۳ ردیف اول داده فارسی و انگلیسی را چک می‌کنیم که جفت باشند.

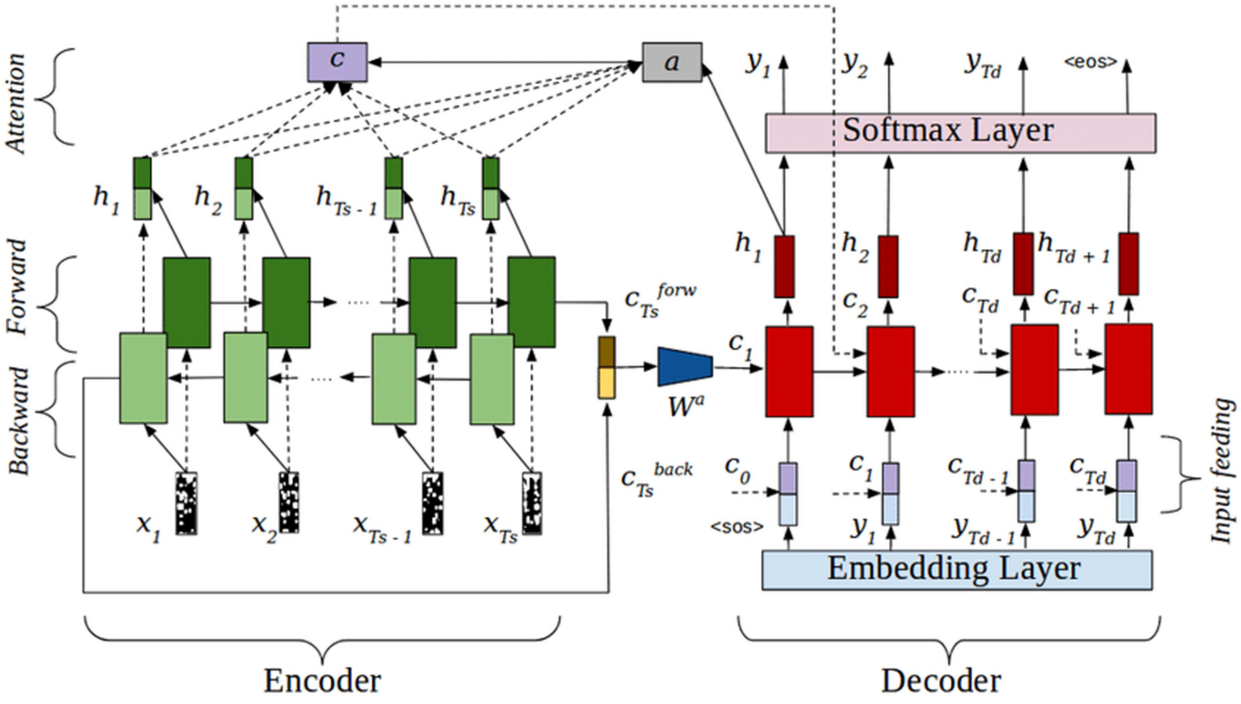
۱-۲. پیش پردازش دادگان با استفاده از fariseq-preprocess – مدل اول

ابتدا یک دایرکتوری به نام data_bin می‌سازیم که خروجی‌های مربوطه را در آن قرار دهیم.

سپس با دستور fairseq-preprocess پیش پردازش‌های لازم را انجام می‌دهیم. که به عنوان آرگیومنت زبان مبدا و مقصد، دادگان train, validation, test، محل ذخیره خروجی، تعداد کلمات مبدا و مقصد برای نگهداری را می‌گیرد. به عنوان خروجی دیکشنری کلمات را ذخیره می‌کند. همچنین با اجرای این دستور متوجه می‌شویم چند درصد توکن‌ها با <unk> در دیتاست عوض می‌شوند. که هرچه کمتر باشد بهتر است. که در تمام دادگان تقریبا کمتر از ۲ درصد توکن‌ها این تبدیل را داریم.

۱-۳. آموزش مدل با استفاده از fairseq-train – مدل اول

در این مرحله از یک encoder-decoder استفاده می‌کنیم که معماری آن‌ها LSTM است. که از لایه attention هم استفاده می‌کند. معماری مشابه آن در شکل زیر آمده است.



شکل ۱. معماری Attention-based encoder-decoder

برای آموزش مدل از دستور fairseq-train استفاده می‌کنیم. که معماری شبکه مانند شکل بالاست. هر کدام از بلاک‌ها LSTM است. از بهینه ساز adam استفاده می‌کنیم. و برای محاسبه loss از تابع smoothed_cross_entropy استفاده می‌کنیم.

تکنیک Label smoothing یک تکنیک regularization هست که به لیبل‌ها نویز اضافه می‌کند و از overconfident شدن مدل جلوگیری می‌کند.

برای ارزیابی مدل در هنگام آموزش از معیار bleu استفاده می‌کنیم. که کلیتش اینکه تعداد توکن‌های مشترک بین متن اصلی و ترجمه رو حساب می‌کند و هرچی بیشتر باشد یعنی کیفیت ترجمه بهتر است.

از تکنیک‌هایی مثل dropout و نرخ یادگیری برنامه ریزی شده طبق صورت سوال استفاده می‌کنیم که به یادگیری سالم‌تر مدل کمک می‌کند.

کلا هایپرپارامترها را دقیقا طبق صورت سوال مقدار دهی کردم.

همچین لاگ معیارها را برای ذخیره و نمایش نمودارها ذخیره می‌کنیم. برای این کار آرگیومنت tensorboard –logdir را استفاده می‌کنیم.

و مدل را برای ۵ ایپاک آموزش می‌دهیم. و برای اجرا از GPU کولب استفاده می‌کنیم.

۱-۴. استفاده از fairseq-generate برای ترجمه دادگان ارزیابی – مدل اول

در این مرحله از دستور fairseq-generate برای عمل ترجمه روی دادگان ارزیابی استفاده می‌کنیم. که سایز batch و beam رو می‌تونیم مشخص کنیم. که به ترتیب ۱۲۸ و ۵ را انتخاب کردیم. همچنین مسیری که فایل مربوطه ذخیره شده را هم می‌دهیم. که خروجی آن در فایل translation_valid.txt در فایل ارسالی ضمیمه شده است.

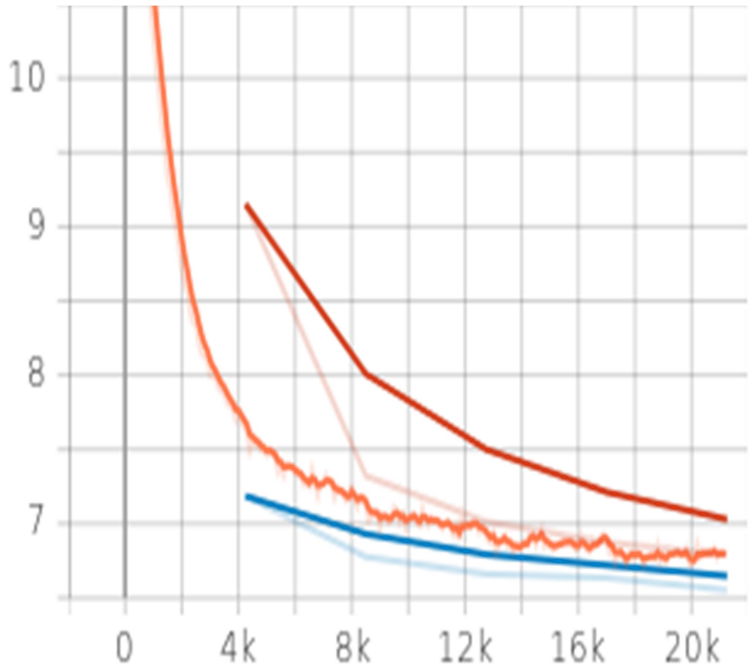
در آخر مقدار Bleu برای دادگان ارزیابی که در خط آخر فایل translation_valid.txt ذخیره شده است را گزارش می‌کنیم:

BLEU4 = 23.38

۱-۵. استفاده از Tensorboard برای برسی فرآیند آموزش – مدل اول

یک ابزار مفید برای راحت‌تر کردن دنبال کردن فرآیند اندازه‌گیری و بصری سازی معیارهای مختلف در هنگام آموزش است. برای استفاده از آن فایلی که برای ذخیره کردن لاگ به دستور fairseq_train دادیم را به عنوان پارامتر به tenorboard می‌دهیم. بعد از اجرای دستور یک داشبورد به ما نمایش می‌دهد که نمودارهای مختلف از معیارهای ارزیابی در آن وجود دارد.

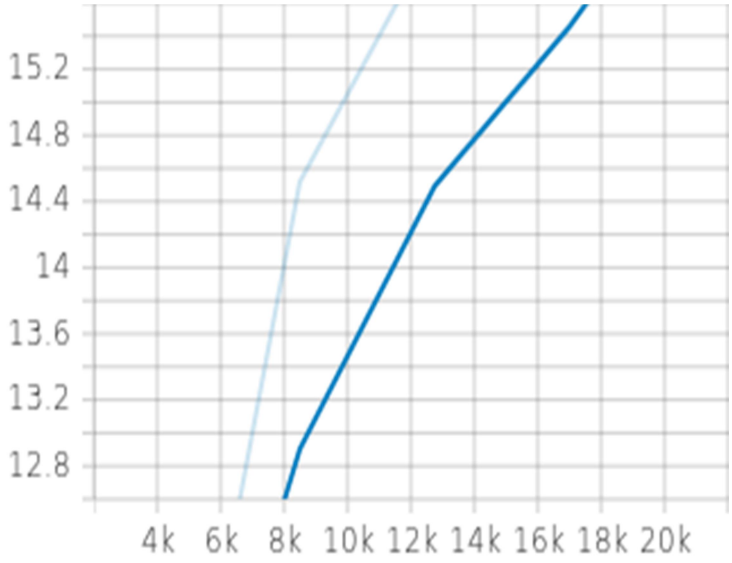
در ادامه نمودار loss حین آموزش برای دادگان مختلف آمده است.



شکل ۲. نمودار loss برای مدل اول

که نشان می‌دهد یادگیری در طول ۵ ایپاک روند خوبی داشته است چون مقدار loss نزولی است و اگر تعداد ایپاک‌ها ادامه داشته باشد احتمالا مدل باز هم بهتر شود. همچنین مدل روی دیتای آموزش overfit هم نشده است.

در ادامه نمودار Bleu آورده شده است.



شکل ۳. نمودار bleu برای مدل اول

در حین آموزش مقدار Bleu افزایش یافته که نشان می‌دهد مدل در حال بهبود است. و شیب این افزایش هم قابل توجه است. این یعنی بازهم اگر ادامه بدهیم مدل بهتر هم خواهد شد.

۱-۶. آموزش مدل BPE برای دادگان انگلیسی و فارسی –مدل دوم

برای استفاده از BPE می‌توانستیم در خود fairseq_train آرگیومنت با همین اسم رو با sentencepeice مقدار دهی کنیم. اما چون سوال بعدی فایل پردازش شده رو هم می‌خواهد جداگانه و با استفاده از کتابخانه sentencepeice این کار را می‌کنیم.

برای آموزش روی دادگان فارسی متد SentencePieceTrainer رو فراخوانی کرده و فایل train.fa را به عنوان ورودی می‌دهیم و سایز وکب را ۵۰۰۰ قرار می‌دهیم و نوع مدل را bpe می‌گذاریم و اسم مدل رو fa_spm می‌گذاریم.

برای آموزش روی دادگان انگلیسی همین کار را اینبار با ورودی train.en انجام می‌دهیم و اسم مدل را en-spm انتخاب می‌کنیم.

حالا کار آموزش مدل BPE با موفقیت انجام شد.

۱-۷. پردازش دادگان با مدل های BPE – مدل دوم

توضیح روش BPE: این روش ابتدا تمام کاراکترهای یونیک را در خود دارد و به مرور زمان جفت کاراکترها یا زیرکلماتی که زیاد با هم آمده‌اند را ادغام می‌کند. این کار را تا زمانی که به سایز دلخواه وکب برسیم ادامه می‌دهیم. برای حل مشکل فاصله به عنوان جداکننده کلمات از sentencepiece استفاده می‌کنیم.

برای استفاده از مدل BPE آموزش دیده شده روی دادگان validation و testو خود train باید مدل را روی این‌ها اعمال کنیم.

برای اینکار به دو تابع نیاز داریم.

تابع اول متن را به صورت جمله جمله استخراج می‌کند اسم این تابع extract_texts است.

تابع دوم جمله گرفته شده را با کمک tokenizer.encode_as_pieces توکنایز می‌کند و در فایلی با اسم متناسب ذخیره می‌کنیم.

۱-۸. پیش پردازش دادگان با استفاده از fariseq-preprocess - مدل دوم

این بخش مانند بخش متناظر برای مدل اول است.

۱-۹. آموزش مدل با استفاده از fairseq-train– مدل دوم

این فاز هم دقیقا مانند مدل اول است. مدل را با آرگومان‌های مشابه مدل قبلی آموزش می‌دهیم. و ۵ ایپاک مدل را اجرا می‌کنیم.

۱-۱۰. استفاده از fairseq-generate برای ترجمه دادگان ارزیابی – مدل دوم

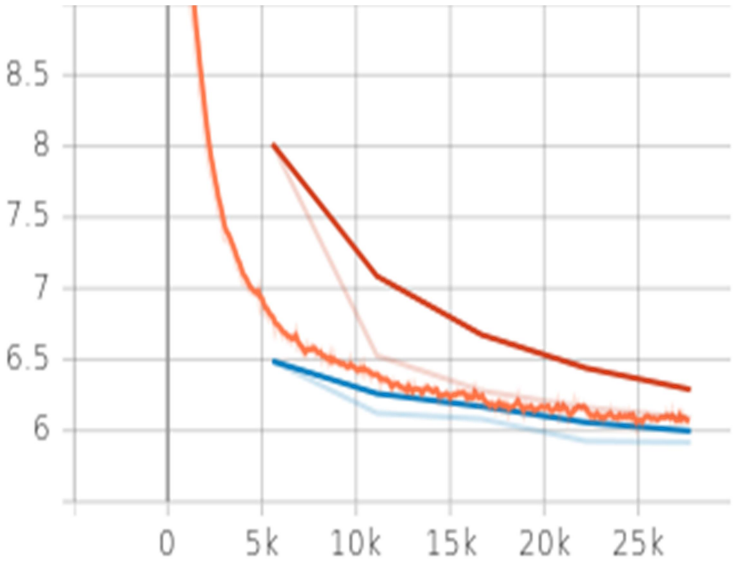
در این بخش هم مانند قبل بر روی دادگان بخش ارزیابی ترجمه را انجام می‌دهیم و نتایج را در فایل translation_valid.txt ذخیره می‌کنیم.

مقدار Bleu که در خط آخر این فایل آمده است در ادامه گزارش می‌کنیم:

BLEU4 = 26.15

۱-۱۱. استفاده از Tensorboard برای بررسی فرایند آموزش– مدل دوم

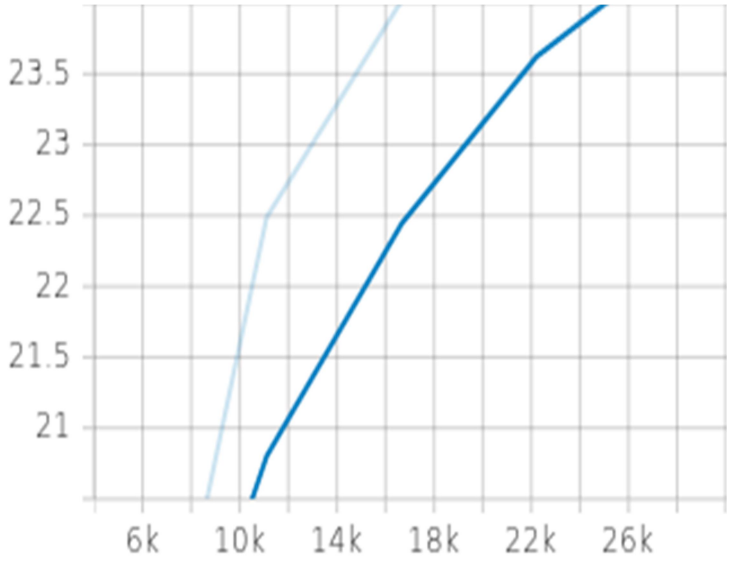
نمودار loss برای مدل دوم در زیر آورده شده است:



شکل ۴. نمودار loss برای مدل دوم

که در شکل بالا مشاهده می‌شود یادگیری به خوبی انجام شده است چون مقدار loss رو به کاهش است. و این کاهش احتمالا برای چند ایپاک دیگر هم بتواند ادامه پیدا کند. نکته خوب دیگه اینکه مدل overfit هم نشده است.

در ادامه نمودار Bleu برای مدل دوم آورده شده است:



شکل ۵. نمودار Bleu برای مدل دوم

در اینجا هم مشخص است که نمودار در معیار Bleu روی دادگان validation در حال پیشرفت است. یعنی آموزش به خوبی انجام شده است.

۱-۱۲. مقایسه دو مدل و تحلیل:

فرق مدل اول با مدل دوم در استفاده از BPE است. دیدیم که استفاده از این توکنایزر به نفع مدل تموم شد و نتایج در معیار Bleu این موضوع را نشان می‌دهد. مدل اول تقریبا ۲۳ بود و مدل دوم تقریبا ۲۶ بود. که بهبود قابل توجهی هست.

توجیه این افزایش عملکرد می‌تواند مربوط به این موضوع باشد که استفاده از BPE به مدل اجازه می‌دهد که بازنمایی‌های گوناگون بیشتری یاد بگیرد. همچنین به مدل قابلیت یادگیری ظرافت‌های زبانی را می‌دهد که پتانسیل مدل برای عملکرد بهتر را افزایش می‌دهد. همچنین این مدل به هندل کردن توکن‌های نادر و خارج از وکب کمک می‌کند.

اما این‌ها همه توجیه بود و برخی موارد ممکن است این مدل نه تنها باعث افزایش عملکرد نشود عملکرد را بدتر هم کند. برای اینکه این موضوع را بفهمیم بهترین راه عملی تست کردن است. پس همیشه نمی‌توان انتظار بهبود عمکلرد را داشت اما اینجا بنظر خوب جواب داده است.

۲. پاسخ ۲_ Zero-Shot Learning

۲-۱. دانلود Bert Tokenizer و Bert Model

ابتدا از کتابخانه BertTokenizer توکنایزر مدل از پیش آموزش دیده شده bert-base-multilingual-cased را فرخوانی می‌کنیم. و بعد هم خود مدل را با BertModel فراخوانی می‌کنیم.

توضیحاتی درباره این مدل: این مدل روی ۱۰۴ زبان معروف دنیا روی ویکی پدیا با رویکرد masked language modeling آموزش دیده شده است. base به معنای مدل پایه bert است. multilingual به معنای چند زبانه بودن و cased به معنای حساس بودن به بزرگی و کوچکی حروف است.

۲-۱. پردازش دادگان AFEC با استفاده از Bert Tokenizer

حالا در مرحله بعد می‌خواهیم هر سه بخش دادگان را با این مدل توکنایز کنیم. برای این کار ابتدا با تابع extract_texts فایل را به جملات جدا از هم تبدیل می‌کنیم. بعد خروجی را هر بار به تابع tokenized_texts می‌دهیم که با مدل گفته شده توکنایز کند و در جایی ذخیره کند.

هر سه دادگان train, validation, test را در هر دو زبان به توکنایز می‌دهیم و این دو تابع را به ترتیب فراخوانی می‌کنیم.

۲-۳. پیش پردازش دادگان با استفاده از fariseq-preprocess – مدل سوم

برای پیش پردازش از دستور faireq_preprocess استفاده می‌کنیم و مبدا را زبان انگلیسی و مقصد را زبان فارسی قرار می‌دهیم. دادگان train, validation, test را هم می‌دهیم.

و در آخر آرگومان joined-dictionary را استفاده می‌کنیم چون embedding ما چندزبانه است و اینطور نیست که فارسی و انگلیسی برای خودشان دیکشنری جداگانه داشته باشند.

۲-۴. ذخیره وزن های لایه embedding شبکه Bert با فرمت مناسب

یک فایلی به نام word_weights.txt می‌سازیم که کلمات را کنار embedding آن‌ها قرار دهد. وزن embeddingها را از دستور model.get)input_embedding().weight.data می‌گیرد. کلمات را هم از دستور toneizer.get_vocab() می‌گیریم.

۲-۵. آموزش مدل با استفاده از fairseq-train

۲-۵-۱. مدل با به روزرسانی وزن‌های embedding

برای آموزش مدل از دستور fairseq-train استفاده می‌کنیم. ابتدا مسیر متناظر با embedding بخش انکودر و دیکودر را برابر وزن‌های بدست آمده از مرحله قبل می‌گذاریم. که بردار کلمات را متناظر با آن قرار دهد.

بقیه موارد مشابه مدل اول است. که دیگه اینجا تکرار نمی‌کنم.

۲-۵-۲. مدل با ثابت بودن وزن‌های embedding

برای مدل بعدی همین کارها را تکرار می‌کنیم اما وزن لایه embedding را باید فریز کنیم تا حین آموزش آپدیت نشوند. برای این کار در آرگیومنت‌های دستور این موارد را هم اضافه می‌کنیم: encoder-freeze-embed, decoder-freeze-embed

۲-۶. استفاده از fairseq-generate برای ترجمه دادگان ارزیابی

بعد از اتمام آموزش روی دادگان ارزیابی ترجمه را برای هر مدل انجام می‌دهیم. و آن را در فایلی به نام translation_valid.txt دخیره می‌کنیم. که در آن ترجمه‌های مدل و معیار Bleu ذخیره شده است.

مقدار Bleu در فایل در ادامه آورده شده است:

	مدل با وزن غیرثابت	مدل با وزن ثابت
Bleu	33.74	20.40

که مشاهده می‌شود مدل با وزن غیرثابت عملکرد بهتری از مدل با وزن ثابت داشته است. علت ان اینکه پارامترهای بیشتری آپدیت شدند و امبدینگ برای تسک خاص اینجا بهتر مناسب‌تر می‌شود.

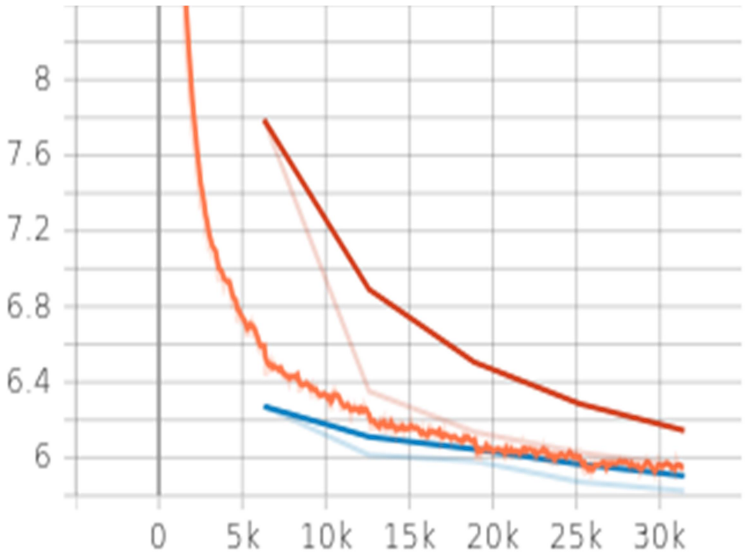
همینطور امدل با وزن غیرثابت که از embedding مدل BERT استفاده می‌کند از دو مدل سوال یک استفاده نمی‌کنند بهتر است. این یعنی اینکه اطلاعات زبانی که در این embedding نهفته شده است در تسک ترجمه ماشینی بسیار موثر است. از طرفی چون این BERT چندزبانه آموزش دیده است، قدرت آن برای ترجمه بیشتر هم شده است. و مهم‌تر اینکه در این مدل embedding را می‌توانیم برای تسک finetune کنیم.

اما در مدل با وزن فریزشده embedding عملکرد از دو مدل سوال اول بدتر است شاید اگر تعداد ایپاک بیشتر می‌شد دقت آن بهتر از دو مدل دیگر می‌شد. اما در کل چون از embedding عام منظوره استفاده می‌کند دقتش حداقل در ایپاک‌های کم از دو مدل سوال اول کمتر است.

۲-۷. استفاده از Tensorboard برای بررسی فرایند آموزش

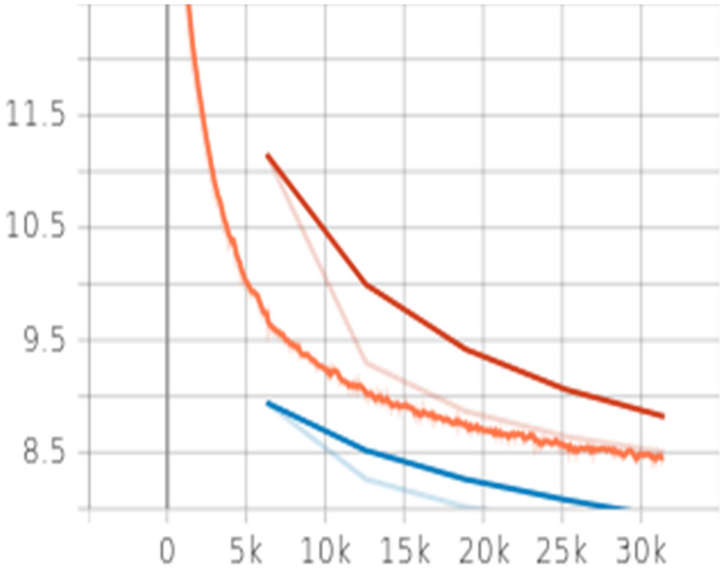
در اینجا نمودار loss با کمک ابزار Tensorboard رسم شده است.

نمودار زیر برای مدل با وزن‌های غیرثابت لایه embedding است.



شکل ۶. نمودار loss برای مدل غیر فریز شده

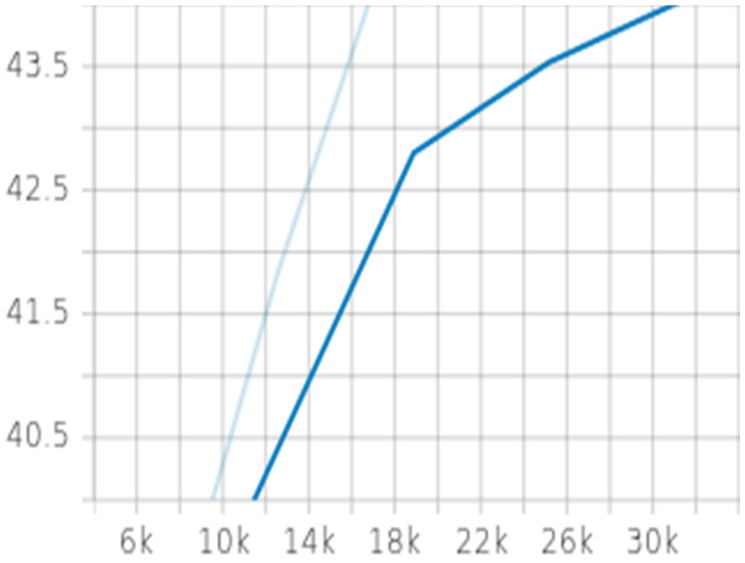
نمودار زیر برای مدل با وزن‌های ثابت لایه embedding است.



شکل ۷. نمودار loss برای مدل فریز شده

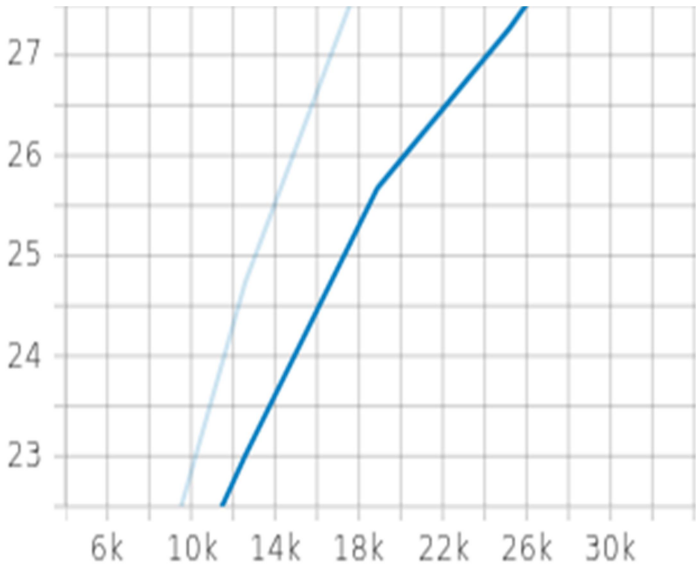
که در هر دو فرآیند آموزش به خوبی انجام شده است و بدون رخ دادن overfit در حال کم کردن loss است.

در ادامه نمودار Bleu برای مدل با وزن‌های غیر ثابت لایه embedding آمده است:



شکل ۸. نمودار Bleu برای مدل غیر فریز شده

و نمودار زیر هم نمودار Bleu برای مدل با وزن‌های ثابت لایه embedding است.



شکل ۹. نمودار Bleu برای مدل فریز شده

همانطور که مشخص است معیار Bleu در هر دو مدل در طول ایپاک‌ها بهتر شده است. و حتی با ادامه دادن آن می‌توان عملکرد را بهتر هم کرد.

۳. توضیحات فایل ارسالی

- کد سوال اول در فایلی به نام NLP_CA5_Q1.ipynb قرار گرفته است.
- کد سوال دوم در فایلی به نام NLP_CA5_Q2.ipynb قرار گرفته است.
- برای هر مدل یک پوشه با نام‌های model 1-4 ساختیم که فایل‌های مهم مربوط به آن ذخیره شده است.
- فایل‌های log مربوط به loss با فرمت csv در فولدر csv_files هر پوشه مدل قرار دارد.
- فایل ترجمه دادگان ارزیابی در فایل‌های با نام translation_valid.txt ذخیره شده است.
- نمودارهای loss, Bleu, best Bleu در فایل‌های plot هر پوشه مدل ذخیره شده است.