Assignment 5  
NLP

**PREPARED BY**

Samin Mehdizadeh - 810100526

Mohsen Fayyaz - 810100524

Spring 2022

**فهرست مطالب**

[**1 ترجمه ماشینی**](#_a7yudv6alkjh) **4**

[1) همانطور که می دانیم در تسک NMT نیازمند پیش پردازش هستیم، یکی از روش های پیش پردازش اعمال bpe بر پیکره آموزشی است به نظر شما چه پیش پردازش دیگری برای ترجمه ماشینی مبتنی بر شبکه عصبی مفید است؟ به دلخواه خود دو روش پیش پردازش دیگر را، با ذکر دلیل، انتخاب کرده و نام ببرید.](#_wwto7ofv90n2) 5

[2) به نظر شما در زبان فارسی کدام یک از این 3 نوع پیش پردازش از اهمیت بیشتری برخوردار است؟ چرا؟ در زبان انگلیسی چطور؟](#_5mtxdng88pol) 5

[3) همانطور که می دانید معماری های متفاوتی برای آموزش مدل های sequence to sequence وجود دارد. در این تمرین قصد داریم تا مدلی بر مبنای معماری transformer آموزش دهیم.](#_4f8onlf686il) 7

[4) در این مرحله متناسب با ابزارهایی که انتخاب کرده اید، 10 تا از پارامترهای آموزش مدل را شرح دهید (در هر ابزار به شکل جداگانه). ابزار مورد نظر خود را نام برده و ابتدا توضیح دهید که این پارامترها چه کاری انجام می دهند و سپس توضیح دهید چه پارامترهایی را لازم است تا نسبت به حالت پیش فرض تغییر دهید.](#_2lnpsrvjesoh) 8

[5) پارامتری که فکر می کنید در کیفیت خروجی یک مدل ترجمه ماشینی تاثیر گذار هستند ولی امکان تغییر یا تنظیم آنها با توجه به منابع موجود (سخت افزاری و محدودیت های ابزار) وجود ندارد را نام برده و اهمیت هر یک را مختصرا توضیح دهید. (در هر ابزار به شکل جداگانه)](#_5ecu5llpw28h) 13

[6) حال می خواهیم 2 مدل مختلف را با استفاده از 2 ابزار انتخابی آموزش دهیم که جزییات این بخش به شرح زیر است:](#_q8x6zq54pnan) 15

[ابتدا bpe و دو روش پیش پردازش دیگر را بر روی دادگان آموزش اعمال کنید](#_mz7kb2tps16v) 15

[چگونگی و جزئیات پیاده سازی این بخش را در گزارش خود ذکر کنید.](#_p6cy30fdvu4d) 15

[انتخاب ابزار مناسب در این مرحله بر عهده شماست و محدودیتی وجود ندارد.](#_alugansastnk) 15

[خروجی مرحله پیش پردازش را به شکل یک فایل جداگانه در گزارش خود بیاورید.](#_4qehsgxdlxdq) 15

[پارامترهای لازم را متناسب با دادگان، منابع موجود و صلاحدید خود مقداردهی کنید.](#_sl3ht15ss7c6) 18

[پارامترهای آموزش مدل را به شکل کامل در گزارش خود ذکر کنید.](#_uttzakdftkst) 18

[آموزش مدل را تا زمانی که در میانه آموزش قرارگیرد ادامه دهید.](#_ak3o97tuvul7) 21

[توصیه میشد تا از google colab برای حل این تمرین استفاده](#_1unf9n3oae3i) 21

[در این تمرین قصد داریم با روند آموزش یک مدل ماشین ترجمه و الزامات آن آشنا شویم . انتظار ما این است که مدل نهایی که ارائه می دهید در میانه مسیر اموزش باشد. یکی از راه های بررسی این موضوع کنترل دستی خروجی مدل بر روی داده های تست است برای مثال معمولا خروجی یک مدل ترجمه که تازه شروع به آموزش کرده است، تکرار تنها چند کلمه خاص است و بدیهی است که این مدل به عنوان مدل نهایی پذیرفته نیست.](#_m5lw77ajdobz) 21

[با توجه به کم بودن حجم مجموعه داده اولیه و محدودیتهای منابع انتظار تولید یک ماشین با کیفیت را نداریم. نگران نتایج ضعیف احتمالی نباشید :)](#_vbcoo3ewk8s5) 21

[انتظار نداریم که مجموع زمان آموزش در هر ابزار بیش از ۶ ساعت به طول انجامد و الزاما صرف زمان بیشتر برای آموزش مدل امتیاز محسوب نمی شود.](#_2nwz4qz9yems) 21

[لطفا علاوه بر فایل گزارش، فایل اسکریپت دستورات اجرا شده و یا اگر در Google Colab اجرا کرده اید فایل notebook آن به همراه خروجی سیستم های آموزش داده شده برای فایل های تست را نیز ارسال کنید.](#_xn13u9ppgpwd) 21

[7) پس از آموزش بایستی روند تغییرات Bleu دو مدل آموزش دیده را بر روی مجموعه dev با افزایش تعداد epochها نشان دهید. برای این کار میتوانید از دستور ذخیره مدل های میانی در ابزار مورد نظر خود استفاده کنید. (حداقل 5 نقطه را در طول آموزش مدل گزارش دهید)](#_4krsqmqwqo7u) 22

[8) با کمک دادگان موازی test، مقدار Bleu را برای این دو مدل بر روی دادگان آزمایشی گزارش دهید.](#_wa8jqjddhsjz) 24

[9) 3 تا از معیارهایی که فکر میکنید برای ارزیابی یک ابزار تولید ماشین ترجمه مناسب هستند را نام ببرید و براساس این معیارها توضیح دهید به نظرتان از میان دو ابزار انتخابی کدام انتخاب بهتری بوده است؟](#_5eeqrmu073ng) 25

# 1 ترجمه ماشینی

ابتدا دو تا از ابزارهای معرفی شده در قسمت مقدمه را انتخاب کنید و برای انجام تمرین موارد زیر را در نظر بگیرید:

در این تمرین از دو ابزار

OpenNMT <https://opennmt.net/>  
Fairseq <https://github.com/facebookresearch/fairseq>

استفاده می‌کنیم.

## 

## 1) همانطور که می دانیم در تسک NMT نیازمند پیش پردازش هستیم، یکی از روش های پیش پردازش اعمال bpe بر پیکره آموزشی است به نظر شما چه پیش پردازش دیگری برای ترجمه ماشینی مبتنی بر شبکه عصبی مفید است؟ به دلخواه خود دو روش پیش پردازش دیگر را، با ذکر دلیل، انتخاب کرده و نام ببرید.

یکی از مهم‌ترین پیش‌پردازش‌ها در زبان انگلیسی که بحث هم روی آن زیاد شده است، lowercase کردن متن است. با این کار باعث می‌شود تعداد ووکب بسیار کمتر شود و مدل راحت تر بتواند آموزش ببیند. این موضوع برای این تمرین که مدل یک ترنسفورمر ساده است، دیتاست کوچک است و منابع سخت افزاری کافی برای آموزش طولانی مدت هم نداریم بسیار اهمیت می‌یابد.  
ضمنا در فارسی از کتابخانه <https://github.com/ICTRC/Parsivar> برای پیش پردازش داده‌های فارسی استفاده می‌کنیم که شامل اصلاح کاراکترهای خاص برای جلوگیری از افزایش بیش از حد ووکب، بررسی ponctuation ها و فاصله گذاری مناسب و پترن های متعددی که در کتابخانه موجود است اصلاح می‌شوند تا متن یکدست شود و مدل دچار سرگشتگی نشود. همچنین نیم‌فاصله‌ها و فاصله‌ها نیز اگر اشتباه باشند اصلاح می‌شوند و ضمنا متون فینگلیش هم به نوشته فارسی آنها بازگردانده می‌شوند تا همه چیز نرمال شود و مدل راحت‌تر بتواند آموزش ببیند.

## 2) به نظر شما در زبان فارسی کدام یک از این 3 نوع پیش پردازش از اهمیت بیشتری برخوردار است؟ چرا؟ در زبان انگلیسی چطور؟

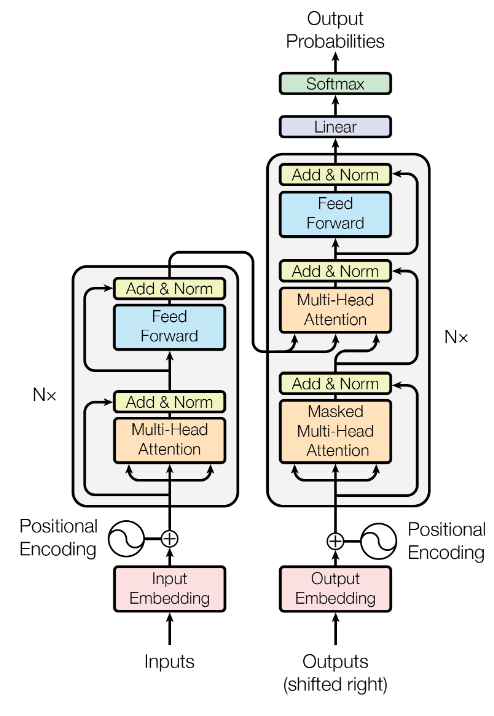
در زبان فارسی، چون morphology rich است، قطعا مرحله BPE از اهمیت بسیار زیادی برخوردار است چون می‌تواند تا حدی این inflection های زبانی را شناسایی کند. مثلا "تر" و "ترین" را بفهمد یا در افعال بتواند جداسازی‌ هایی مثل "می" یا "خورد"+"م"/"ند" انجام دهد که قطعا می‌تواند به مدل اصلی در فهم زبان کمک کند. در زبان انگلیسی هم اتفاقا همین موضوع هست اما کمتر از فارسی البته باز توانایی تشخیص "er" مشابه "تر" فارسی اهمیت بالایی دارد. در مجموع نشان داده شده است که این روش در هر دو زبان می‌تواند نتایج را خیلی بهتر کند.

طبیعتا باقی موارد که گفته شد خاص زبان بود مثلا lower case کردن معادل در فارسی ندارد و یا یکسان سازی نیم فاصله ها معادل در انگلیسی ندارد ولی اینها هم اهمیت دارند البته که همان BPE از اهمیت بیشتری برخوردار است.

## 

## 3) همانطور که می دانید معماری های متفاوتی برای آموزش مدل های sequence to sequence وجود دارد. در این تمرین قصد داریم تا مدلی بر مبنای معماری transformer آموزش دهیم.

اتفاقا مقاله اصلی transformer نیز روی تسک machine translation بوده است. <https://arxiv.org/pdf/1706.03762.pdf>

  
*ش**کل 1 - معماری مدل ترنسفورمر*

بنابراین در ابزارها هم نوع معماری را ترنسفورمر قرار می‌دهیم.

## 

## 4) در این مرحله متناسب با ابزارهایی که انتخاب کرده اید، 10 تا از پارامترهای آموزش مدل را شرح دهید (در هر ابزار به شکل جداگانه). ابزار مورد نظر خود را نام برده و ابتدا توضیح دهید که این پارامترها چه کاری انجام می دهند و سپس توضیح دهید چه پارامترهایی را لازم است تا نسبت به حالت پیش فرض تغییر دهید.

1. ابزار OpenNMT
   1. Learning\_rate:  نرخ یادگیری مدل برای آپدیت شدن پارامتر ها. مقدار پیش فرض برای این مقدار برابر با ۱ است.
   2. Heads: تعداد  head ها برای  transformer در لایه attention. مقدار پیش فرض head برابر با ۸ است.
   3. Batch\_size: حداکثر سایز batch برای آموزش داده ها. مقدار پیش فرض برابر با ۶۴ است.
   4. Word\_size: تعداد کل پراسس های توزیع شده که در این پروژه چون تنها از یک gpu استفاده شده است مقدار آن برابر با ۱ در نظر گرفته شده است. مقدار پیش فرض نیز همین است.
   5. Train\_steps: تعداد گام ها برای آموزش مدل. مقدار پیش فرض برابر با ۱۰۰۰۰۰ است. در واقع در صورتی که ۱۰۰۰۰۰ = train\_steps و ۶۴ = batch\_size باشد برای تمام شدن یک epoch لازم است تا ۱۵۶۲ گام انجام شود.
   6. Save\_checkpoints\_step: پس از طی شدن گام های ذکر شده در این قسمت مدل به دست آمده ذخیره می شود. مقدار پیش فرض برای آن ۵۰۰۰ است.(در واقع checkpoint به ازای هر x گام ذخیره می شود)
   7. Optim: روش بهینه سازی استفاده شده برای آموزش مدل. مقدار پیش فرض sgd است.
   8. Src\_word\_vec\_size: سایز embedding برای داده های مبدا. مقدار پیش فرض برابر با ۵۰۰ در نظر گرفته شده است.
   9. encoder|decoder)\_type): نوع لایه decoder یا encoder استفاده شده در مدل. به صورت پیش فرض rnn در نظر گرفته شده است.
   10. enc|dec)\_layers): تعداد لایه ها در encoder یا decoder. مقدار پیش فرض برابر با ۲ است.

در پیاده سازی انجام شده تنها پارامتر های train\_steps,save\_checkpoint\_stepsوvalid\_steps تغییر داده شده اند که تنها برای بالا بردن سرعت آموزش مقدار آن کم شده است اگرچه دقت مدل را کاهش می دهد. اما در صورتی که دقت مهم تر از سرعت باشد می توان تعداد لایه های encoder یا decoder را افزایش داد و یا اینکه تعداد vec\_size برای embedding را بالا برد. علاوه بر این در صورتی که محدودیت سخت افزاری نداشته باشیم نیز می توانیم از تعداد بیشتری هسته برای پردازش موازی استفاده کنیم.

1. ابزار Fairseq

پارامترهای این ابزار در لینک زیر توضیح داده شده است که به توصیف 10 مورد از آنها می‌پردازیم.

<https://fairseq.readthedocs.io/en/latest/command_line_tools.html#fairseq-train>

* 1. no-progress-bar باعث می‌شود نوار پیشرفت نباشد و به جای آن
  2. Tensorboard-logdir با این پارامتر می‌توان از ابزار لاگ تنسوربورد که برای tensorflow است بهره برد و اطلاعات و لاگهای طول آموزش را در آن ابزار به صورت نمودارهایی مشاهده کرد. این دستور به طور خاص، آدرس اتصال به ابزار تنسوربورد را می‌گیرد تا این کار را انجام دهد.
  3. Seed این پارامتر مقدار سید که توابع رندوم با آن کار می‌کند را مشخص می‌کند. در تحقیقات ماشین لرنینگ معمولا یک اجرا نمی‌تواند نتیجه قطعی دهد و با استفاده از سیدهای مختلف و گزارش میانگین و انحراف معیار نتایج، می‌توان از نتیجه آزمایش خود مطمئن‌تر شد.
  4. Cpu این پارامتر باعث می‌شود از cpu به جای استفاده از اینترفیس cuda که برای gpu های nvidia است استفاده شود و برای زمانی که gpu در اختیار نداریم مناسب است.
  5. Tpu این پارامتر باعث می‌شود از tpu به جای cuda استفاده شود که tpu یا Tensor processing unit واحد پردازشی خاص هوش مصنوعی و ماشین لرنینگ است که توسط گوگل توسعه داده شده است و روی کولب هم قابل استفاده است.
  6. Fp16 باعث می‌شود از floating point عای 16 بیتی به جای 32 بیت استفاده شود که می‌تواند برای سرعت اجرا و مموری بهتر باشد با هزینه دقت محاسبات.
  7. Optimizer با این پارامتر می‌توان از بهینه سازی که مورد نظر است استفاده کرد که معمولا Adam انتخاب می‌شود، اما گزینه های دیگری مثل adadelta, adafactor, adagrad, adam, adamax, composite, cpu\_adam, lamb, nag, sgd وجود دارد.
  8. Lr-scheduler این پارامتر می‌تواند روش برنامه ریزی learning rate را مشخص کند. معمولا در ماشین لرنینگ از لرنینگ ریت های بیشتر شروع می‌کنیم تا به نقطه حدودی مینیمم لاس نزدیک شویم ولی هر چه نزدیکتر شویم اگر لرنینگ ریت همان عدد اولیه باشد ممکن است مشکل over shooting داشته باشیم که به همین دلیل بهتر است کم کم لرنینگ ریت در طول آموزش کم شود. برای این کار scheduler های متفاوتی هست که می توان از بین cosine, fixed, inverse\_sqrt, manual, pass\_through, polynomial\_decay, reduce\_lr\_on\_plateau, step, tri\_stage, triangular انتخاب کرد و حالت عادی fixed است که یعنی تغییر نکند.
  9. Scoring نوع امتیازی که محاسبه می‌شود را مشخص می‌کند. این مقدار می‌تواند bert\_score, sacrebleu, bleu, chrf, meteor, wer باشد که حالت دیفالت همان blue که در این سوال هم خواسته شده است و نیاز به تغییر ندارد.
  10. Task عملی که می‌خواهیم انجام دهیم و مدل را آموزش دهیم مشخص می‌کند که مثلا می‌تواند language model یا خیلی موارد دیگر باشد و حالت دیفالت آن همان translation است که نیاز به تغییر ندارد.
  11. Arch یا a پارامتری است که معماری مدل را مشخص می‌کند و می‌تواند مثلا مدل‌های مختلف و سایزهای مختلفشان را بگیرد که در این مسئله آن را به مقدار transformer می‌گذاریم.
  12. Max-epoch حداکثر تعداد ایپاک آموزش است که بعد از آن متوقف شود. در این مسئله چون محدودیت داریم از این پارامتر استفاده می‌کنیم و عددی محدود می‌دهیم تا آموزش خیلی طولانی نشود.
  13. Lr مقدار نرخ یادگیری را مشخص می‌کند که در ابتدای مدل است و مقدار اولیه 0.25 دارد و آن را نیاز نیست تغییر خاصی بدهیم.
  14. Save-dir مسیر ذخیره سازی چک پوینت ها را مشخص می‌کند و دیفالت آن فولدر checkpoints است که مناسب است.
  15. Keep-last-epochs تعداد حداکثر چک پوینت های ایپاک ها که سیو شود را مشخص می‌کند و مقدار دیفالت -1 دارد که یعنی محدودیت ندارد.
  16. Patience صبر مدل در برابر بهبود نیافتن مدل روی داده dev را مشخص می‌کند و مقدار دیفالت -1 است یعنی متوقف نمی‌شود. ما این مقدار را به 3 تغییر می‌دهیم.

در ادامه کامل‌تر پارامترهایی که استفاده می‌کنیم توضیح داده می‌شود اما در این سوال هم ذکر می‌کنیم که پارامتر max-epoch را از مقدار پیش فرض -1 که یعنی محدودیت ندارد به یک مقدار محدود مثل 40 تغییر می‌دهیم تا آموزش بیش از حد طولانی نشود و کولب بسته نشود.  
ضمنا از patience هم استفاده می‌کنیم و به جای -1 یک عدد محدود مثل 5 می‌گذاریم تا هر وقت پیشرفتی پس از 5 مرحله حاصل نمی‌شد متوف شویم و بیهوده آموزش را ادامه ندهیم.

## 

## 5) پارامتری که فکر می کنید در کیفیت خروجی یک مدل ترجمه ماشینی تاثیر گذار هستند ولی امکان تغییر یا تنظیم آنها با توجه به منابع موجود (سخت افزاری و محدودیت های ابزار) وجود ندارد را نام برده و اهمیت هر یک را مختصرا توضیح دهید. (در هر ابزار به شکل جداگانه)

* ابزار OpenNMT
  + Word\_size: این پارامتر برابر با تعداد پراسس های توزیع شده است و در صورت داشتن سخت افزاری که gpu های بیشتری دارد می توان مقدار آن را افزایش داد و سرعت آموزش را بالا برد.
  + enc\_layer و dec\_layer:بالابردن تعداد لایه های encoder و decoder می تواند باعث شود مدل پیچیدگی های بیشتری داشته باشد و مدل ها پیچیده تری را نیز یاد بگیرد اگرچه لازم است تا پارامتر های آموزش را جایی ذخیره کند که ممکن است با محدویت RAM مواجه شویم.
  + Batch\_size: در برخی موارد بالابردن سایز batch باعث می شود مدل بهتری داشته باشیم اما ممکن است به دلیل محدودیت حافظه قادر به بالابردن آن نباشیم.
  + dec\_rnn\_size و enc\_rnn\_size: این پارامتر تعداد hidden\_state در decoder یا encoder را مشخص می کند که زیاد کردن آن می تواند باعث شود مدل موارد بیشتری یاد بگیرد اما مجددا ممکن است برای ذخیره آنها با مشکل حافظه مواجه شود.
  + Heads: افزایش تعداد head های transformer نیز می تواند باعث شود مدل از جنبه های مختلفی یاد بگیرد اما ممکن است بالا بردن تعداد head ها تا هر حدی امکان پذیر نباشد و مشکل حافظه داشته باشیم.
* ابزار Fairseq
  + یک بخش بزرگی از پارامترها مربوط به distributed\_training است که یعنی بیش از یک عدد gpu داشته باشیم. که ما محدودیت داریم و همان یکی هم بعضی وقت‌ها نداریم.
  + Distributed-world-size تعداد مجموع gpu های موجود را مشخص می‌کند
  + Nprocs-per-node تعداد gpu در هر نود را مشخص می‌کند. اعمال محاسبات روی gpu های یک نود سریعتر است به همین دلیل اینکه روی هر نود چند gpu موجود است مهم است.
  + Batch-size سایز هر بچ را مشخص می‌کند که متناسب با آن حافظه GPU استفاده خواهد شد و به علت محدودیت حافظه نمی‌توانیم آن را خیلی زیاد کنیم. در بعضی موارد، افزایش سایز بچ می‌تواند مدل را بهتر به سمت مینیمم لاس هدایت کند و نتایج بهتری در انتها داشته باشد.
  + Workers تعداد ورکرها یا همان ترد و پراسس های همزمان را مشخص می‌کند اما چون روی کولب تنها یک هسته داریم، ساختن ورکر بیشتر فقط باعث ایجاد overhead می شود و از نظر سرعت بهتر نخواهد بود.
  + ضمنا arch که قبلا توضیح داده شد هم می‌تواند مدلهای بزرگتر با hidden states size و یا تعداد لایه بیشتر بگیرد که چون باعث بیشتر شدن پارامترها و کند شدن آموزش و نیاز به مموری بیشتر دارد برای منابع محدود ما مناسب نیست.

## 

## 6) حال می خواهیم 2 مدل مختلف را با استفاده از 2 ابزار انتخابی آموزش دهیم که جزییات این بخش به شرح زیر است:

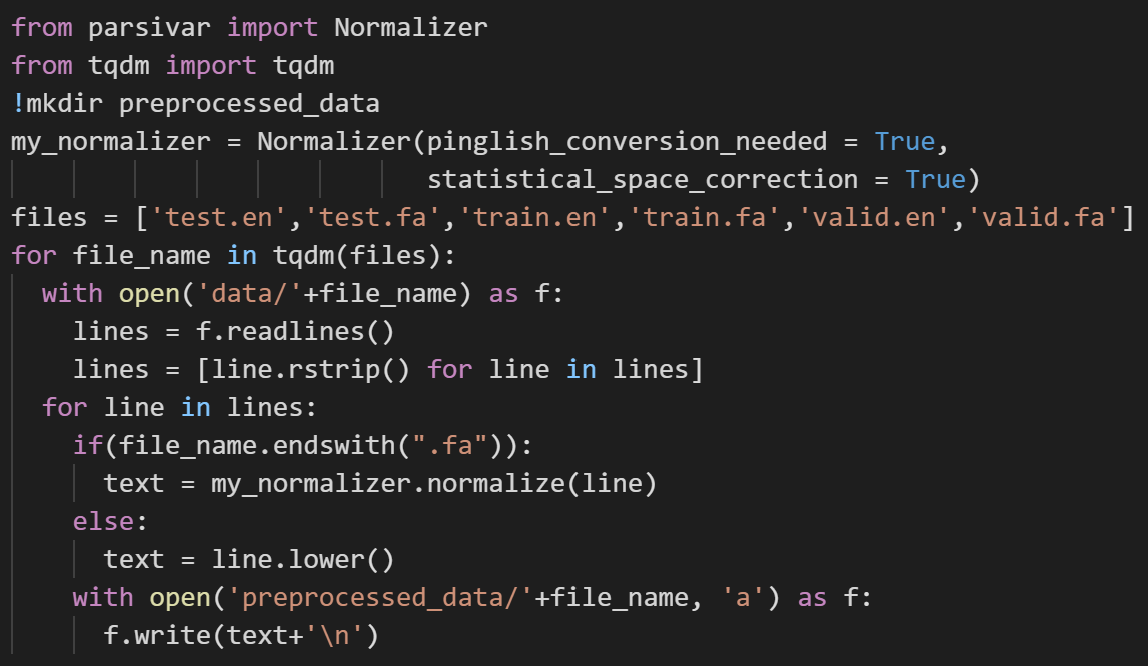
### ابتدا bpe و دو روش پیش پردازش دیگر را بر روی دادگان آموزش اعمال کنید

#### چگونگی و جزئیات پیاده سازی این بخش را در گزارش خود ذکر کنید.

#### انتخاب ابزار مناسب در این مرحله بر عهده شماست و محدودیتی وجود ندارد.

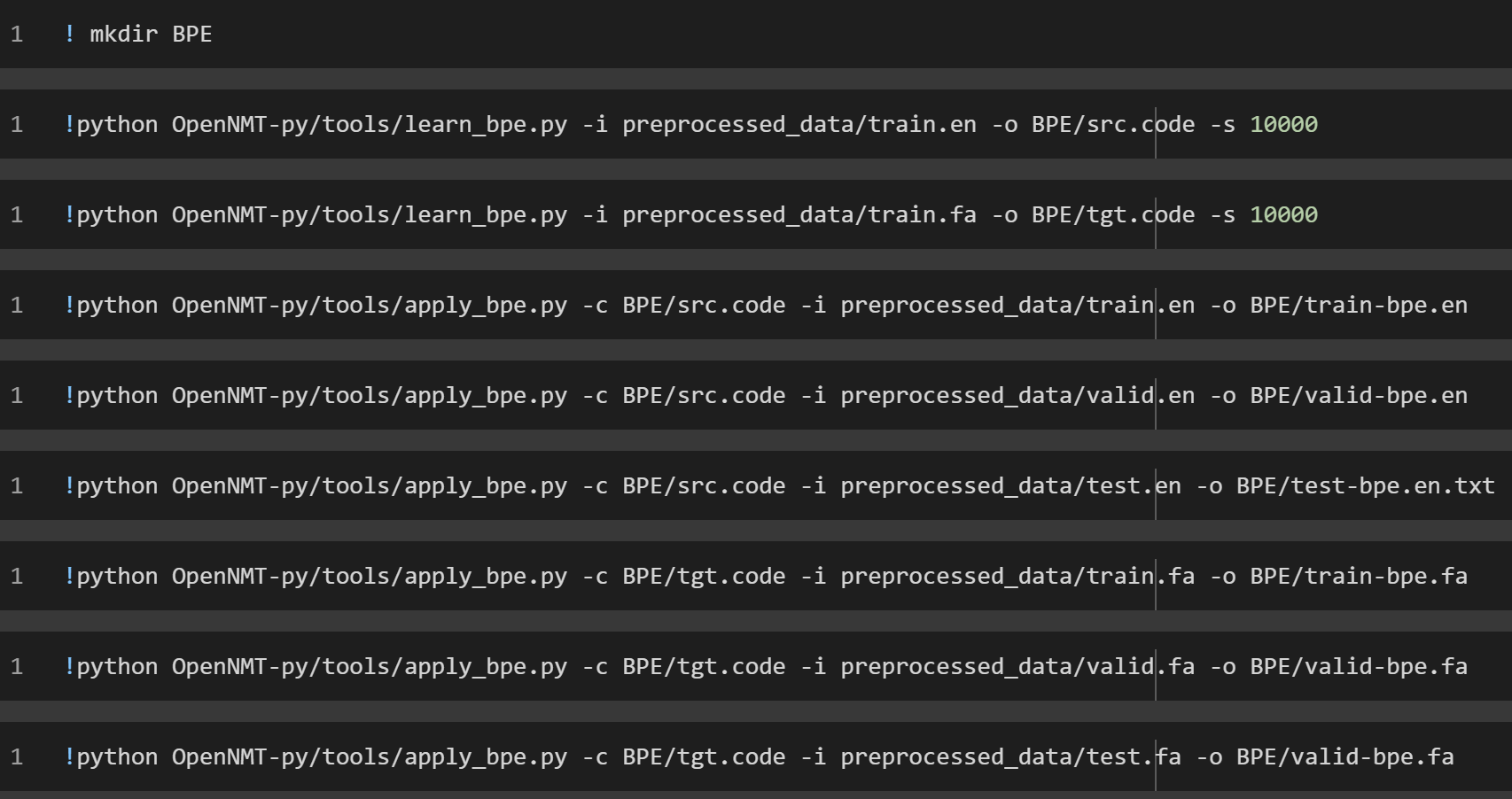
#### خروجی مرحله پیش پردازش را به شکل یک فایل جداگانه در گزارش خود بیاورید.

پیش پردازش‌هایی که اعمال می‌کنیم در بخش های قبلی توضیح داده شدند. در این بخش پیاده سازی و جزئیات آن را ذکر می‌کنیم.

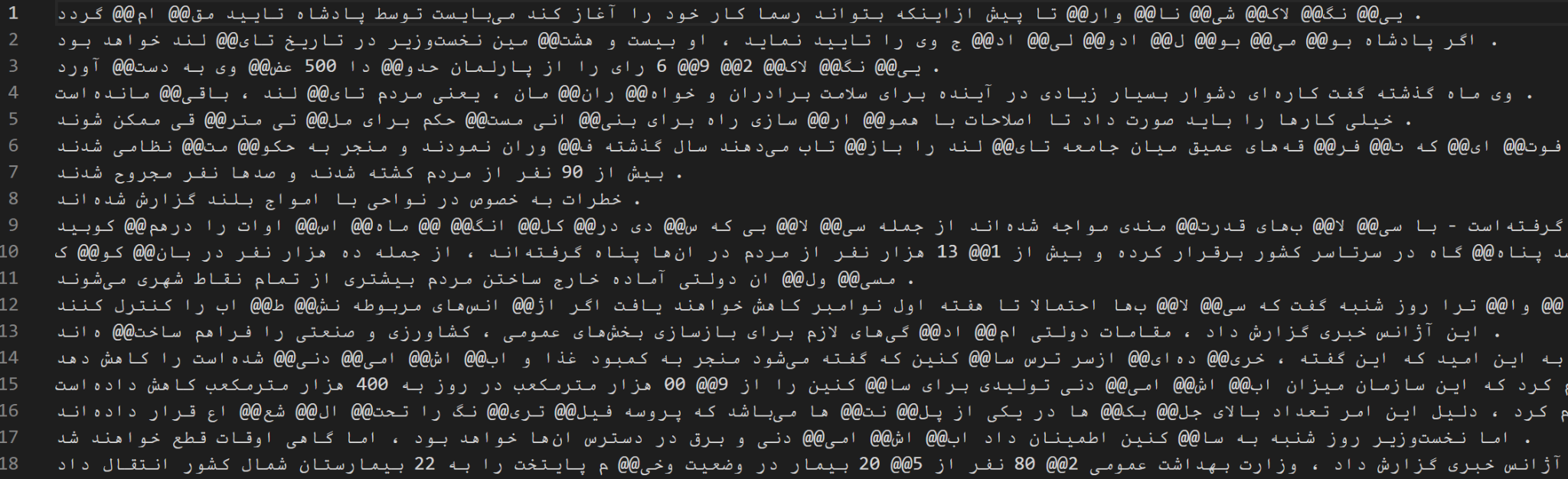
*ش**کل 2 - پیش پردازش‌های اولیه*

در این بخش برای زبان فارسی با استفاده از کتابخانه parsivar پیش پردازشهایی مثل تغییر فینگلیش به فارسی، اصلاح نیم فاصله، اصلاح کاراکترهای عجیب، و اصلاح اسپیس گذاری انجام شده و برای انگلیسی متن به lower case اصلاح شده است.

سپس برای bpe چندین روش وجود دارد. مثلا با خود ابزار openNMT می‌توان این کار را انجام داد.

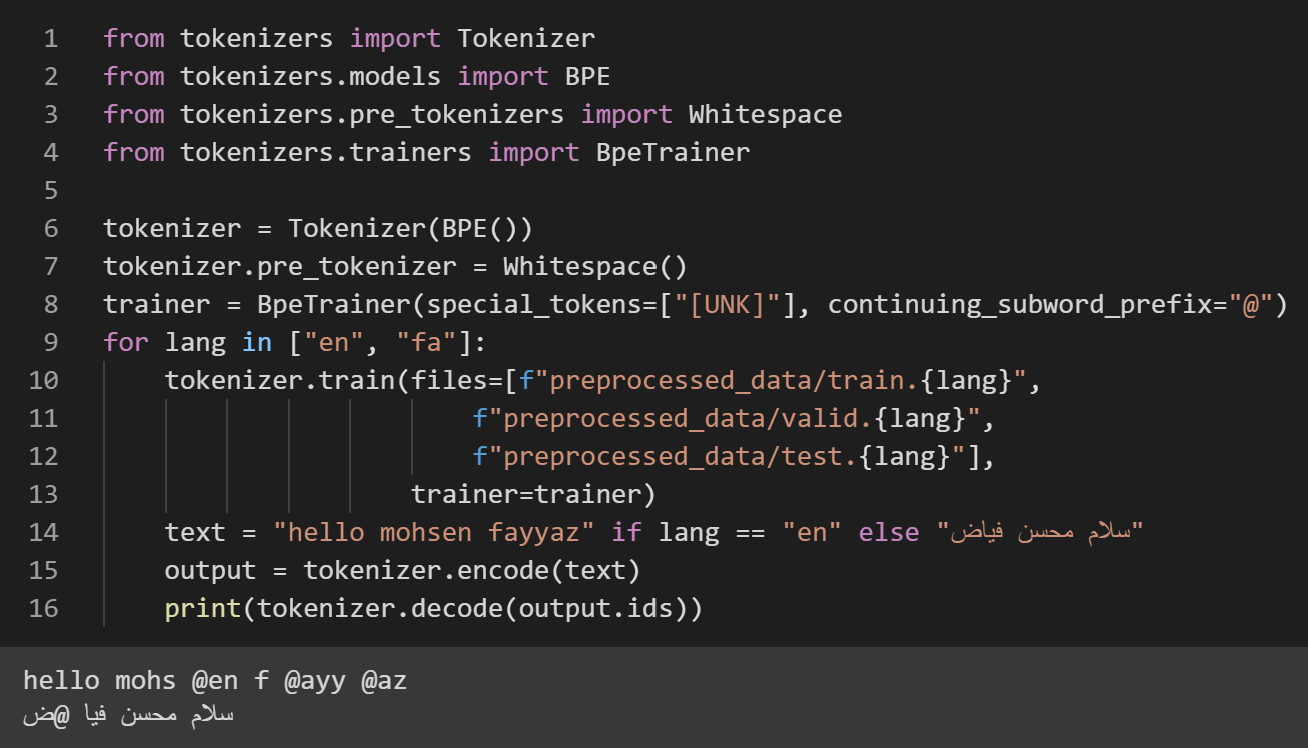
  
*ش**کل 3 - پیش پردازش‌ BPE با OpenNMT*

با استفاده از این ابزار ابتدا روی بخش ترین، توکنایزر BPE را آموزش می‌دهیم و سپس روی تمام بخشها اعمال می‌کنیم.

  
*ش**کل 4 - نتیجه اجرای BPE*

همانطور که دیده می‌شود کلمات به بخشهای کوچکتر در مواردی شکسته شده‌اند مثلا قدرت+مندی از هم جدا شده‌اند و انتهای بخشهای شکسته شده که به بخش دیگر می‌چسبند برای مشخص شدن و امکان بازگشتن به متن اولیه (اگر همه اسپیس بودند فرق شکسته شدن و اسپیس واقعی مشخص نمیشد) کاراکتر @@ گذاشته شده است.

برای اطمینان یک روش دیگر را نیز معرفی می‌کنیم. در این روش از کتابخانه هاگینگ فیس استفاده می‌کنیم.

  
*ش**کل 5 - نتیجه اجرای BPE با هاگینگ فیس*

همانطور که دیده می‌شود با این کتابخانه هم می‌توان توکنایزر تعریف کرد و با استفاده از BPE آن را آموزش داد و در نهایت می‌توان برای زیرکلمه ها پریفیکس مشخص گذاشت که برای شباهت به حالت قبل @ گذاشتیم و می‌بینیم که این توکنایزر هم با کلماتی که کاملش را ندارد به صورت شکستن کلمه عمل می‌کند.

در انتها خروجی این دو مرحله در دو فایل مجزا در کنار گزارش آورده شده است.

### 

### پارامترهای لازم را متناسب با دادگان، منابع موجود و صلاحدید خود مقداردهی کنید.

#### پارامترهای آموزش مدل را به شکل کامل در گزارش خود ذکر کنید.

1. openNMT

برای آموزش مدل از دستور زیر استفاده شده است:

python OpenNMT-py/train.py -data translate/data -save\_model translate/model -train\_steps 50000 -save\_checkpoint\_steps 5000 -valid\_steps 1000 -world\_size 1 -gpu\_rank 0

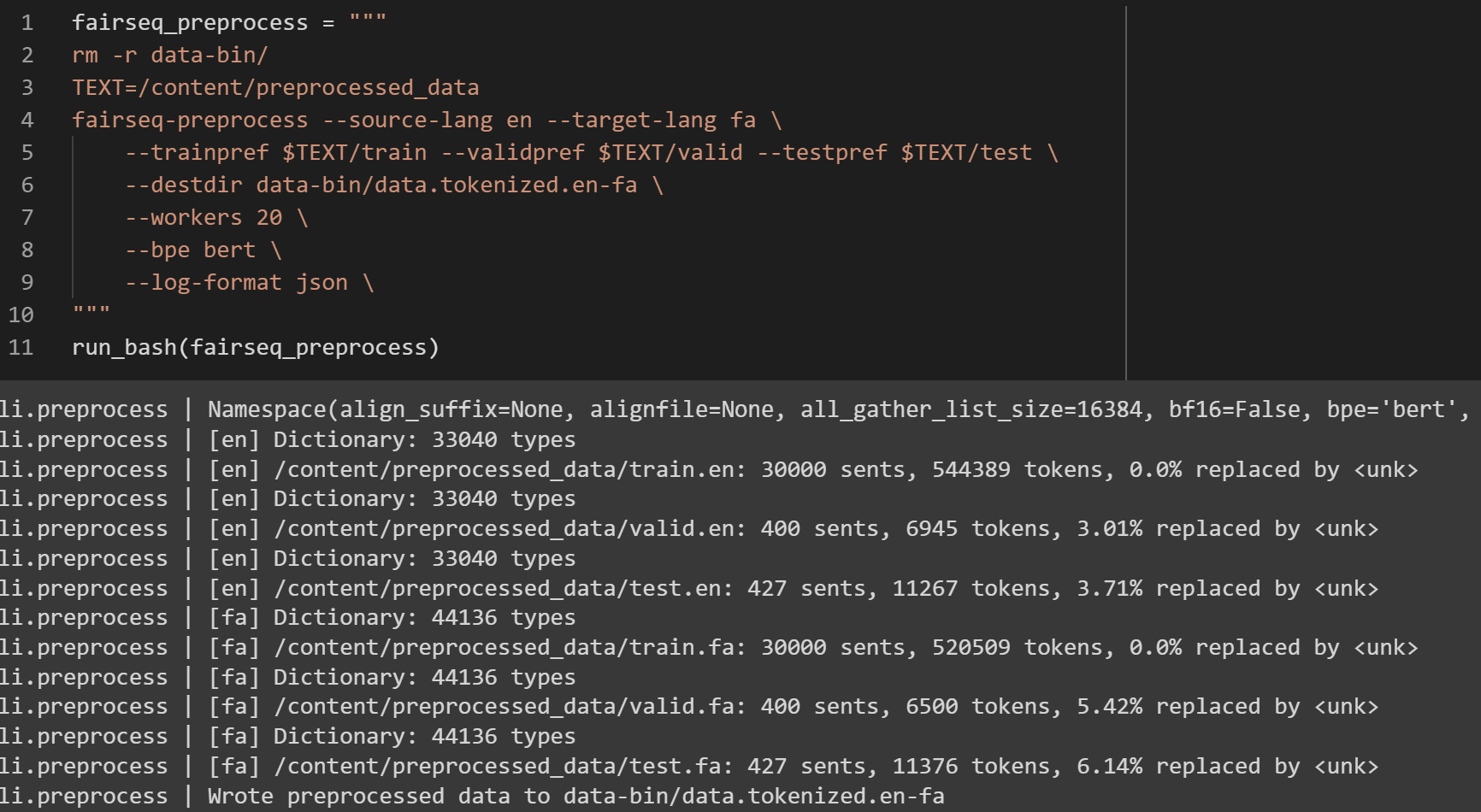
در این دستور تقریبا تمامی پارامتر ها به صورت پیش فرض انتخاب شده است تنها برای بالا بردن سرعت آموزش تعداد گام ها برای آموزش کاهش یافته است.

مدل آموزش داده شده به شکل زیر است:

  
*شکل 6-  NMT به کمک openNMT*

1. FairSeq

ابتدا از preprocess استفاده می‌کنیم تا فایلهای bin که مورد نیاز آموزش است ساخته شود. برای ورودی به این تابع، زبان مبدا و مقصد و فایلهای ترین و ولیدیشن و تست را می‌دهیم و نوع bpe را مشخص میکنیم و فرمت لاگ را مشخص می‌کنیم.

  
*شکل7 -  تابع پیش پردازش fairseq*

بعد از آن به آموزش مدل می‌پردازیم.

fairseq-train \

data-bin/data.tokenized.en-fa \

--arch transformer --share-decoder-input-output-embed \

--optimizer adam --adam-betas '(0.9, 0.98)' --clip-norm 0.0 \

--lr 5e-4 --lr-scheduler inverse\_sqrt --warmup-updates 4000 \

--dropout 0.3 --weight-decay 0.0001 \

--criterion label\_smoothed\_cross\_entropy --label-smoothing 0.1 \

--max-tokens 4096 \

--eval-bleu \

--eval-bleu-args '{"beam": 5, "max\_len\_a": 1.2, "max\_len\_b": 10}' \

--eval-bleu-detok moses \

--eval-bleu-remove-bpe \

--eval-bleu-print-samples \

--best-checkpoint-metric bleu --maximize-best-checkpoint-metric \

--max-epoch 50 \

--patience 10 \

--save-dir checkpoints \

# --fp16 \

--batch-size 64

در این ابزار معماری را ترنسفورمر می‌گذاریم. اپتیمایزر را بهینه ساز معروف adam می‌گذاریم و مقادیر بتای آن را مشخص می‌کنیم. لرنینگ ریت را 5e-4 می گذاریم و ضمنا scheduler تعریف می کنیم تا در طول آموزش کاهش یابد.  
همچنین از warmup استفاده می‌کنیم که شروع آموزش را با لرنینگ ریت کم شروع می کند تا اول آموزش وزنها به جاهای اشتباه پرت نشوند و کم کم به به محلی که درست است منتقل شوند و سپس به لرنینگ ریت اولیه میرسیم.  
از dropout استفاده کردیم تا حدود 0.3 نورون ها به صورت رندوم خاموش شوند تا از overfitting مدل جلوگیری شود و generalization بهتر داشته باشیم.  
برای ارزیابی از معیار bleu استفاده می‌کنیم و بهترین چکپوینت از این نظر را ذخیره می کنیم در کنار همه چکپوینت ها. ضمنا بچ سایز را 64 قرار می دهیم و آدرس ذخیره سازی چکپوینت را مشخص می کنیم و البته patience هم مشخص می کنیم که صبر مدل برای early stopping است تا اگر بعد از 10 مرحله بهبودی حاصل نشد متوقف شویم.  
و ضمنا بیشترین تعداد ایپاک را هم محدود به 50 می کنیم تا زمان بیش از حد گرفته نشود که کولب بسته شود و همه زحمت ها به باد رود.  
ضمنا می‌توانیم به جای fp32 از fp16 استفاده کنیم اما چون تفاوت سرعت زیادی در تغییر آن دیده نشد از آن صرف نظر کردیم.

### 

### آموزش مدل را تا زمانی که در میانه آموزش قرارگیرد ادامه دهید.

#### توصیه میشد تا از google colab برای حل این تمرین استفاده

#### در این تمرین قصد داریم با روند آموزش یک مدل ماشین ترجمه و الزامات آن آشنا شویم . انتظار ما این است که مدل نهایی که ارائه می دهید در میانه مسیر اموزش باشد. یکی از راه های بررسی این موضوع کنترل دستی خروجی مدل بر روی داده های تست است برای مثال معمولا خروجی یک مدل ترجمه که تازه شروع به آموزش کرده است، تکرار تنها چند کلمه خاص است و بدیهی است که این مدل به عنوان مدل نهایی پذیرفته نیست.

#### با توجه به کم بودن حجم مجموعه داده اولیه و محدودیتهای منابع انتظار تولید یک ماشین با کیفیت را نداریم. نگران نتایج ضعیف احتمالی نباشید :)

#### انتظار نداریم که مجموع زمان آموزش در هر ابزار بیش از ۶ ساعت به طول انجامد و الزاما صرف زمان بیشتر برای آموزش مدل امتیاز محسوب نمی شود.

#### لطفا علاوه بر فایل گزارش، فایل اسکریپت دستورات اجرا شده و یا اگر در Google Colab اجرا کرده اید فایل notebook آن به همراه خروجی سیستم های آموزش داده شده برای فایل های تست را نیز ارسال کنید.

آموزش هر دو ابزار انجام شد و نوت بوک ها و خروجی ها در فایلهای کنار گزارش آمده اند.

نمونه ای از نتایج هم در زیر آمده که نشان می‌دهد مدل در میانه آموزش بوده و مدل فقط تکرار نمی‌کند.

ملموس قبل از اینکه بتواند به کار رود این کشور را فرا بگیرد ، باید سکوت کند .

اگر مسیری‌های شاه از او رنج بگیرد ، در " تاریخ اتحاد " در " تاریخ " خواهد‌بود .

هود در همان محلی که از زندان به زندان آمده بود در حمله به زندان می‌بخشید .

امروز دیشب گفت : خیلی خطرناک‌ترین کار برای آینده و خواهرت وجود دارد .

میان حزب دموکراتیک و قانونگذاری که بسی در جامعه عمیقا که بسی در جامعه عمیقا رشد می‌کند ، در سال گذشته و به سوی نظامی عبور می‌کند .

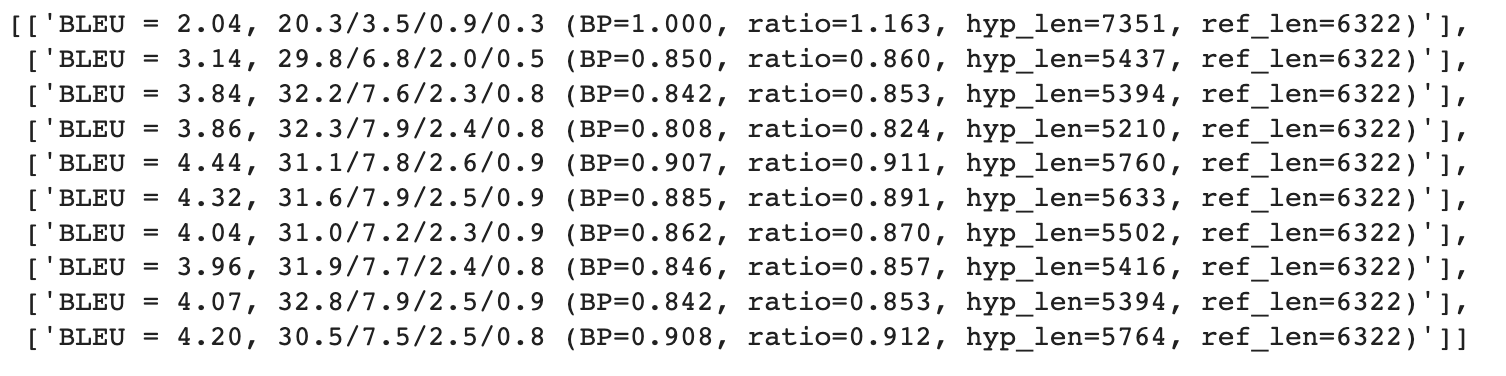
بیش از 90 نفر کشته و صدها نفر زخمی شدند .

## 

## 7) پس از آموزش بایستی روند تغییرات Bleu دو مدل آموزش دیده را بر روی مجموعه dev با افزایش تعداد epochها نشان دهید. برای این کار میتوانید از دستور ذخیره مدل های میانی در ابزار مورد نظر خود استفاده کنید. (حداقل 5 نقطه را در طول آموزش مدل گزارش دهید)

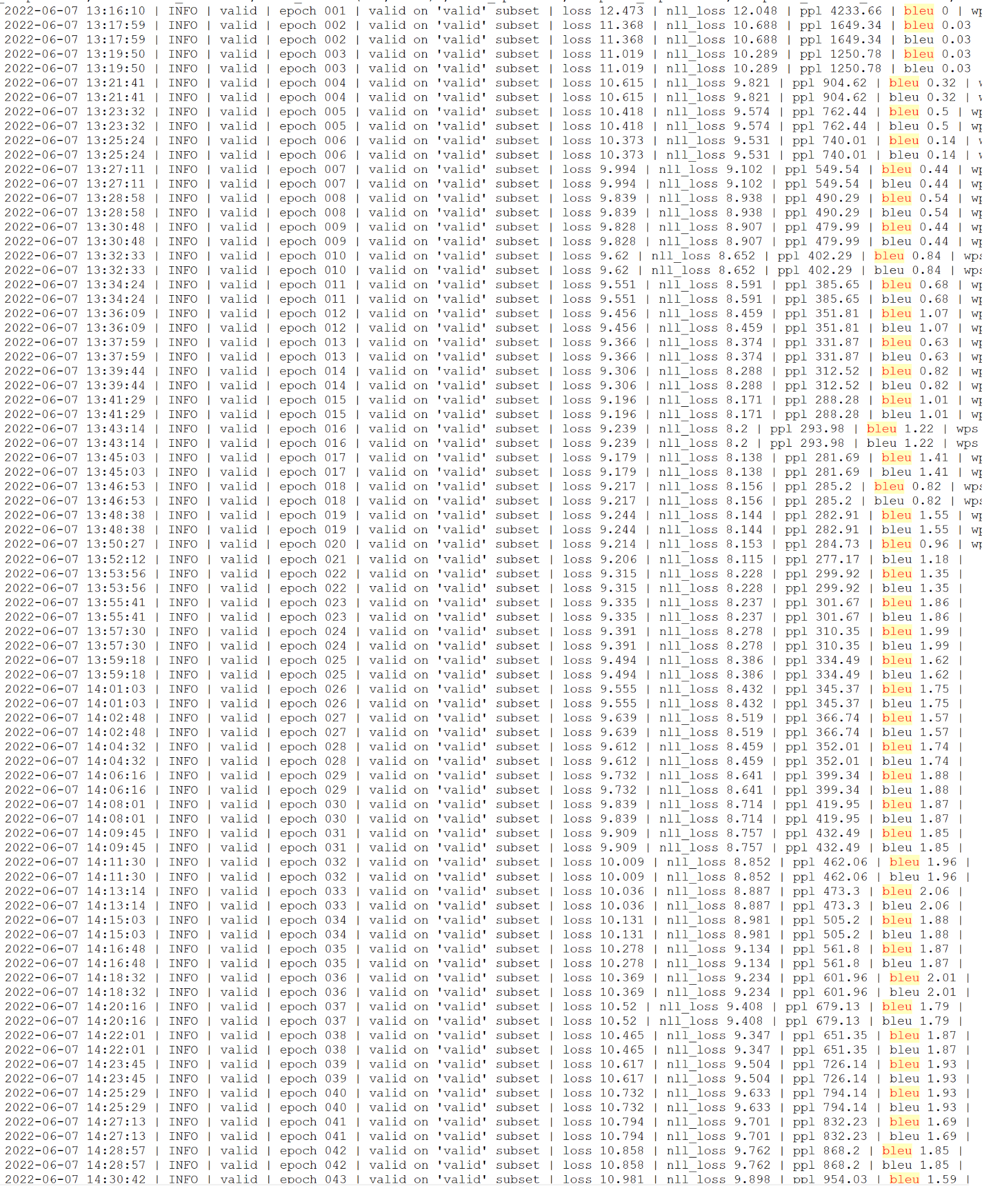
1. openNMT

تغییرات bleu به ازای هر 5000 گام به صورت زیر است:

  
*شکل 7-  تغییرات bleu در openNMT*

همانطور که دیده می‌شود حداکثر مقدار 4.44 است.

1. fairseq

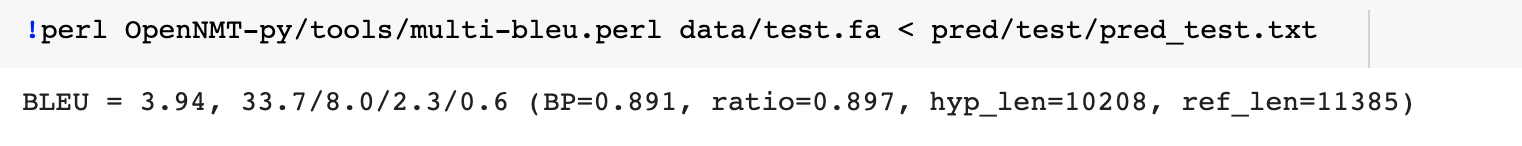
  
*شکل 8 -  تغییرات bleu در fairseq*

همانطور که دیده می‌شود مقدار blue در این ابزار حداکثر به 2.06 می رسد.

## 

## 8) با کمک دادگان موازی test، مقدار Bleu را برای این دو مدل بر روی دادگان آزمایشی گزارش دهید.

1. openNMT

  
*شکل 8-  مقدار bleu بر روی داده های تست به کمک openNMT*

3.94

1. Fairseq

! fairseq-generate data-bin/data.tokenized.en-fa --path checkpoints/checkpoint\_best.pt --batch-size 128 --beam 5 --remove-bpe --eval-bleu --results-path generate\_results

Generate test with beam=5: **BLEU4 = 1.22**, 23.3/2.6/0.4/0.1 (BP=1.000, ratio=1.047, syslen=11460, reflen=10949)

1.22

## 

## 9) 3 تا از معیارهایی که فکر میکنید برای ارزیابی یک ابزار تولید ماشین ترجمه مناسب هستند را نام ببرید و براساس این معیارها توضیح دهید به نظرتان از میان دو ابزار انتخابی کدام انتخاب بهتری بوده است؟

* امکان تنظیم پارامترها جهت آموزش مدل متناسب با داده های در دسترس
* امکان موازی سازی پردازش ها جهت بالا بردن سرعت آموزش
* وجود آموزش و داکیومنتیشن های کافی برای کار با ابزار
* پشتیبانی اسپانسرهای مطرح از ابزار
* جامعه توسعه دهندگان بزرگ
* سهولت استفاده از ابزار
* بهینه بودن نتایج نهایی ابزار
* پشتیبانی از سخت‌افزارهای به روز مانند TPU
* پشتیبانی از framework های متنوع

از میان دو ابزار، در مجموع openNMT گزینه بهتری نسبت به Fairseq بود. اولا نتایج خیلی بهتری را با پارامترهای معمول آموزش توانست به دست آورد که نشان از بهینه تر بودن پایپلاینهای اجرایی آن دارد. ضمنا از نظر داکیومنتیشن غنی‌تر بود و بهتر پارامترهای آن توضیح داده شده بود. (مثلا یکی از پارامترهایی که در داکیومنتیشن fairseq بود اصلا در دستور آن کار نمی کرد!) از نظر پشتیبانی از Framework نیز، ابزار fairseq تنها از pytorch پشتیبانی می‌کند در حالی که ابزار openNMT هم از Pytorch و هم از Tensorflow پشتیبانی می‌کند که بهتر است.

برای تکمیل بحث، معیارهای ارزیابی خود ماشین ترجمه را نیز توضیح می‌دهیم.

**BLEU**

این معیار با داشتن تعدادی متن اصلی و ترجمه آن به عنوان رفرنس سعی می کند تا بر اساس تعداد n-gram های تطابق یافته میان ترجمه اصلی و ترجمه انسان یک عدد به مدل نسبت دهد. مشکل این روش این است که این تطابق ها باید دقیقا مشابه متن مرجع باشند و در صورت استفاده از واژگان یا عبارات متفاوت معیار BLEU کاهش می یابد.

**NIST**

این معیار تقریبا شبیه به BLEU است با این تفاوت که در BLUE به n-gram ها وزن های یکسانی داده می شود اما در این روش به n-gram هایی وزن بیشتری داده می شود که احتمال اتفاق افتادن آن ها کمتر باشد.

**TER**

این متریک مبتنی بر کاراکتر عمل می کند به این صورت که تعداد ادیت های لازم برای تبدیل متن ترجمه شده توسط ماشین به متن ترجمه شده توسط انسان را معیاری برای ارزیابی قرار می دهد.