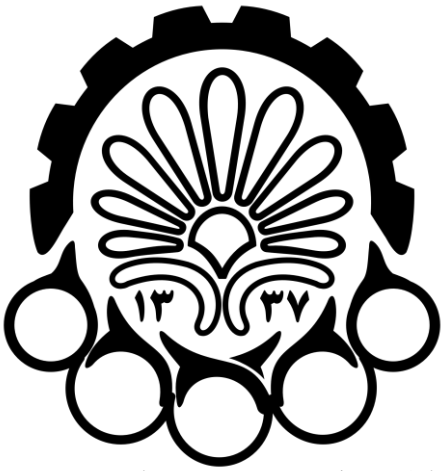


دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

تمر ین دوم درس فهم زبان

دکتر زینلی

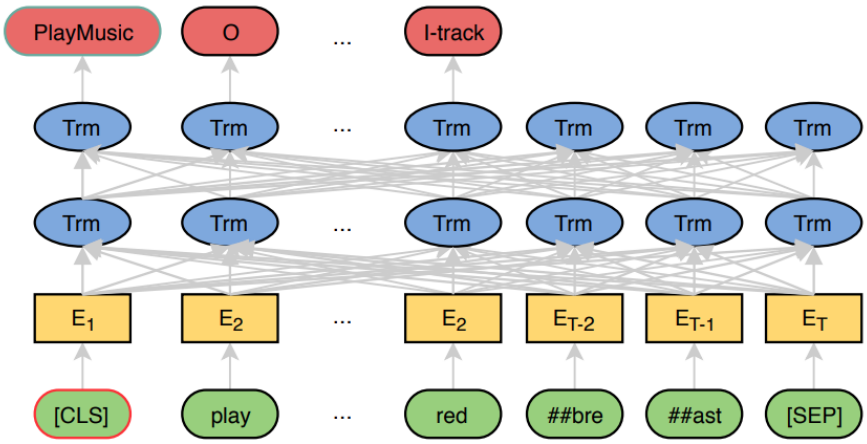
غلامرضا دار ۴۰۰۱۳۱۰۱۸

بهار ۱۴۰۱

فهرست مطالب

۱) جوینت برت

مدل انتخابی اول برای حل این مسئله مدل JointBert از مقاله BERT for Joint Intent Classification and Slot Filling است. معماری کلی این مدل را می‌توانید در شکل زیر مشاهده کنید.



این مدل با گرفتن دنباله ورودی(توکنایزشده برای BERT)، به طور همزمان، Intent و تگ‌های لازم برای هر توکن را مشخص می‌کند. به ابتدای هر جمله ورودی، یک توکن [CLS] و به انتهای هر جمله یک توکن [SEP] اضافه می‌کنیم. مدل با کمک این دو توکن می‌تواند حدود یک جمله را مشخص کند.

خروجی مربوط به ورودی توکن [CLS]، Intent تشخیص داده شده است. همچنین، خروجی مربوط به اولین زیرتوکن هر توکنِ میانی، tag آن توکن را مشخص می‌کند.

پس از انجام تغییرات لازم در این پیاده سازی و آماده سازی مجموعه داده سوال برای این مدل، شروع به آموزش مدل می‌کنیم.

پیاده‌سازی این مدل در این آدرس موجود است: https://github.com/monologg/JointBERT

نکته در مورد مجموعه داده: پس از بررسی مجموعه داده این سوال، متوجه شدیم که تعداد زیادی از دادگان تکراری بودند. تقریبا هر داده ۵ بار کپی شده بود. این اتفاق به نظر سهوی می‌آمد در نتیجه قبل از شروع به آموزش، دادگان تکراری حذف شدند.

نصب ماژول‌های مورد نیاز:

- `python>=3.7`
- `torch==1.5.1`
- `seqeval==1.2.2`
- `transformers==4.3.0`
- `pytorch-crf==0.7.2`

راهنمایی آموزش مدل:

درون پوشه اصلی کدها یک فایل به نام `main.py` وجود دارد. پس از اطمینان از نصب بودن ماژول‌های ذکرشده و دسترسی به cuda، با استفاده از دستور زیر آموزش مدل را شروع کنید. (*دقت کنید در ترمینال، در دایرکتوری ای باشید که کد `main.py` قرار دارد*).

```
$ python main.py --data_dir data --model_dir output/test_model --
task nlu --model_type joint_bert --do_train
```

*حدود **۴۰ دقیقه** در محیط گوگل کولب به زمان نیاز دارد*

راهنمایی تست مدل:

درون پوشه اصلی کدها یک فایل به نام `main.py` وجود دارد. **پس از اتمام آموزش** برای ارزیابی، دستور زیر را اجرا کنید. (*دقت کنید در ترمینال، در دایرکتوری ای باشید که کد `main.py` قرار دارد*)

```
$ python main.py --data_dir data --model_dir output/test_model --
task nlu --model_type joint_bert --do_eval
```

*حدود **۴۰ ثانیه** در محیط گوگل کولب به زمان نیاز دارد*

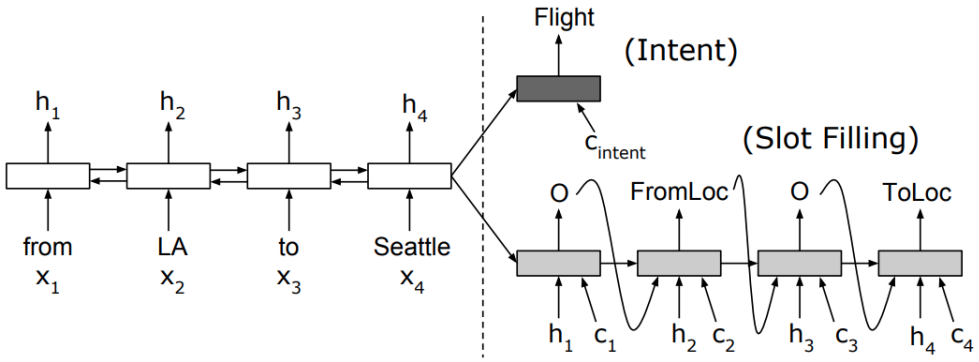
نتایج: همان‌طورکه انتظار می‌رفت نتایج حاصل از استفاده از این روش بسیار مطلوب است. صحت تشخیص Intent ۹۹ درصدی و slot_f1 ۹۵ درصدی بر روی داده تست نتایج بسیار قابل قبولی برای این مسئله هستند. از آنجایی که جمله‌های مجموعه داده، جمله‌های کوتاه و ساده‌ی انگلیسی بودند و BERT بر روی این نوع داده‌ها به خوبی آموزش دیده است، تشخیص این‌که هدف هر کدام ازاین جمله‌ها چیست برای BERT کار ساده ای است. همچنین ساختار BERT اجازه می‌دهد مدل در بخش Slot Filling، به خوبی بر روی جمله ورودی تسلط داشته باشد و کلمات مختلف متن را از نظر کاربرد نیز تحلیل کند.

JointBERT[1] (Test)	
eval_loss	0.1525
intent_acc	0.9937
Slot_f1	0.9581
Slot_recall	0.9598
Slot_precision	0.9565
Semantic_frame_acc	0.9879

JointBERT[1] (Development)	
eval_loss	0.1657
intent_acc	0.9892
Slot_f1	0.9585
Slot_recall	0.9631
Slot_precision	0.9539
Semantic_frame_acc	0.9838

۲) انکدر دیکدر با اتنشن

مدل انتخابی دوم برای حل این مسئله مدل Encoder Decoder with Attention از مقاله Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling است. معماری کلی این مدل را می‌توانید در شکل زیر مشاهده کنید.



بخش انکدر این مدل یک شبکه Bidirectional LSTM است. توکن‌های ورودی به این شبکه داده می‌شوند و خروجی سلول آخر به عنوان ورودی به Classifier داده می‌شود تا Intent جمله محاسبه شود. در ادامه Hidden State آخرین سلول LSTM بازگشتی، به عنوان Initial State به اولین سلول LSTM دیکدر داده می‌شود.دیکدر در این مدل یک LSTM یک‌جهته است. همان‌طور که در شکل مشخص است، خروجی سلول‌های دیکدر در هر مرحله زمانی، tag های تشخیص داده شده برای توکن ورودی در همان مرحله زمانی را مشخص می‌کنند. C_i های موجود در بخش Decoder در واقع **مکانیزم اتنشن** در این مدل هستند و مدل از این طریق در هر مرحله زمانی، می‌تواند با استفاده از Cها به سایر نقاط جمله توجه کند و اطلاعات لازم را کسب کند.

پس از انجام تغییرات لازم در این پیاده سازی و آماده سازی مجموعه داده سوال برای این مدل، شروع به آموزش مدل می‌کنیم.

پیاده‌سازی این مدل در این آدرس موجود است: https://github.com/yinghao1019/Joint_learn

نکته در مورد مجموعه داده: پس از بررسی مجموعه داده این سوال، متوجه شدیم که تعداد زیادی از دادگان تکراری بودند. تقریبا هر داده ۵ بار کپی شده بود. این اتفاق به نظر سهوی می‌آمد در نتیجه قبل از شروع به آموزش، دادگان تکراری حذف شدند.

نتایج: در مورد این مدل نیز نتایج بسیار قابل‌قبول هستند. تقریبا در تمامی معیارها مدل بر اساس BERT عملکرد بهتری داشت اما با این وجود نتایج مدل Encoder Decoder with Attention نیز بسیار خوب هستند.

EncDecAttn[2] (Test)	
eval_loss	0.2000
intent_acc	0.9879
Slot_f1	0.9431
Slot_recall	0.9381
Slot_precision	0.9420
Semantic_frame_acc	0.9722

EncDecAttn [2] (Development)	
eval_loss	0.2143
intent_acc	0.9804
Slot_f1	0.9401
Slot_recall	0.9381
Slot_precision	0.9420
Semantic_frame_acc	0.9722

نصب ماژول‌های مورد نیاز:

- `python>=3.7`
- `torch==1.5.1`
- `seqeval==1.2.2`
- `transformers==4.3.0`
- `pytorch-crf==0.7.2`

راهنمایی آموزش مدل:

درون پوشه اصلی کدها یک فایل به نام `main.py` وجود دارد. پس از اطمینان از نصب بودن ماژول‌های ذکرشده و دسترسی به cuda، با استفاده از دستور زیر آموزش مدل را شروع کنید. (*دقت کنید در ترمینال، در دایرکتوری ای باشید که کد `main.py` قرار دارد*)

```
$ python main.py --data_dir data --model_dir output/test_model_s2s --
task nlu --model_type joint_AttnS2S --do_train
```

*حدود **۸ دقیقه** در محیط گوگل کولب به زمان نیاز دارد*

راهنمایی تست مدل:

درون پوشه اصلی کدها یک فایل به نام `main.py` وجود دارد. پس از اتمام آموزش برای ارزیابی، دستور زیر را اجرا کنید. (*دقت کنید در ترمینال، در دایرکتوری ای باشید که کد `main.py` قرار دارد*)

```
$ python main.py --data_dir data --model_dir output/test_model_s2s --
task nlu --model_type joint_AttnS2S --do_eval
```

*حدود **۲۰ ثانیه** در محیط گوگل کولب به زمان نیاز دارد*

- اگر در اجرای کدها به مشکل خوردید می‌توانید نوت‌بوک فراهم‌شده‌ای که تمام‌نکات گفته شده را انجام می‌دهد را در گوگل کولب آپلود کنید و در آن صورت نباید به مشکلی بر بخورید.
- اسم نوتبوک: `NLU_HW02_colab_nb.ipynb`

مراجع

[1] Chen, Qian, Zhu Zhuo, and Wen Wang. "**Bert for joint intent classification and slot filling.**" *arXiv preprint arXiv:1902.10909* (2019).

[2] Liu, Bing, and Ian Lane. "**Attention-based recurrent neural network models for joint intent detection and slot filling.**" *arXiv preprint arXiv:1609.01454* (2016).