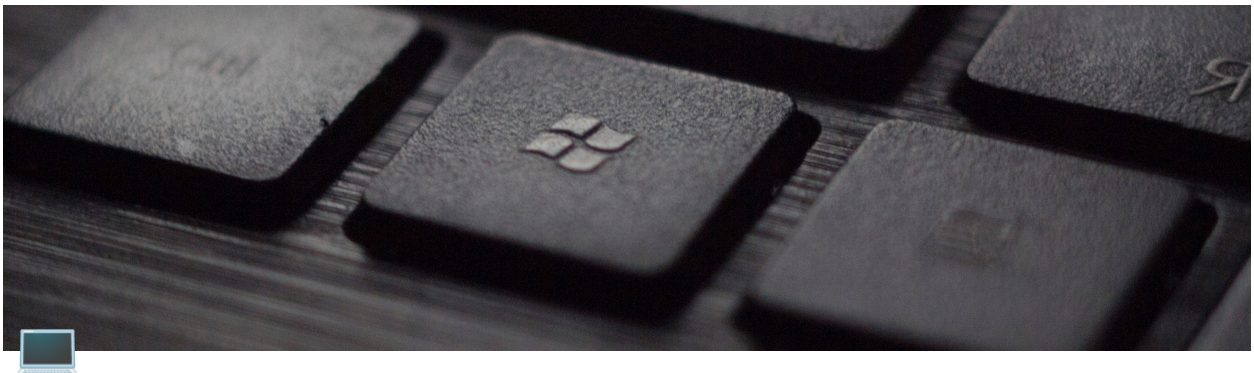


# Windows复现LGESQL流程

自建代码仓库：

GitHub - kanseaveg/lgesql-windows: Source code for ACL2021 Long Paper ``LGESQL: Line Graph Enhanced Text-to-SQL Model with Mixed Local and N

Source code for ACL2021 Long Paper ``LGESQL: Line Graph Enhanced Text-to-SQL Model with Mixed Local and Non-Local Relations" running on windows. - GitHub - kanse
ACL2021 Long Paper ``LGESQL: Line Graph Enhanced Text-to-SQL Model with Mixed Local and Non-Local Relations" running on windows.

https://github.com/kanseaveg/lgesql-windows

相关依赖包版本：

```
python 3.7.9

torch   1.7.1
torchaudio    0.7.2
torchvision   0.8.2
dgl-cu110     0.6.1
```

## 创建环境

```
conda create -n test python=3.7.9

# 在windows下运行Linux指令
conda install -c msys2 m2-base

conda install pytorch==1.7.1 torchvision==0.8.2 torchaudio==0.7.2 cudatoolkit=11.0 -c pytorch

python -c "import torch;print(torch.cuda.is_available())"
```

## 加载依赖

```
# 下载源码
git clone https://github.com/rhythmcao/text2sql-lgesql.git  lgesql

cd D:\DMIR\code\python\lgesql
# 修改requirements.txt文件中的 dgl-cu101==0.5.3 改为 dgl-cu110==0.6.1，新增protobuf
vim requirements.txt
dgl-cu101==0.5.3 ---->  dgl-cu110==0.6.1
protobuf==3.20.0

pip install -r requirements.txt


# 下载依赖
python -c "import stanza; stanza.download('en')" #2023-01-30 17:13:32 INFO: Finished downloading models and saved to C:\Users\tarob\stanza_
python -c "import nltk; nltk.download('stopwords')" # [nltk_data]   Unzipping corpora\stopwords.zip.
python -c "import nltk; nltk.download('punkt')"
#在windows上 请将HOMEPATH改成HOME目录,地址请自行设定，该文件下载时间比较长
python -c "import os; os.environ['HOME']=os.environ['HOMEPATH'];from embeddings import GloveEmbedding; emb = GloveEmbedding('common_crawl_4
```

## 下载必要的词向量和预训练模型

```
# 下载huggingface预训练模型依赖，亦可自行放入文件夹中，路径要统一
mkdir  pretrained_models && cd pretrained_models
git lfs install
git clone https://huggingface.co/bert-large-uncased-whole-word-masking.git
git clone https://huggingface.co/google/electra-large-discriminator.git
mkdir glove.42b.300d && cd glove.42b.300d
# 手动下载glove词向量工具解压好放入对应位置 http://nlp.stanford.edu/data/glove.42B.300d.zip

# 文件下载完使用
awk -v FS=' ' '{print $1}' glove.42B.300d.txt > vocab_glove.txt
```

## 下载数据集

```
# spider download address: https://drive.google.com/uc?export=download&id=1_AckYkinAnhqmRQtGsQgUKAnTHxxX5J0
# unzip spider.zip and move to lgesql/data
# 使用vscode对data/train_spider.json 和data/train_others.json这两个文件进行合并为data/train.json
# 或者直接从https://drive.google.com/file/d/1kDoQ9oVNxLRl02y2ndsly2SIqxpYlr4y/view?usp=sharing下载
```

## 加工数据集

windows加工数据集脚本，实质是将linux shell参数进行一定调整

```
:: windows bat script
set current_dir=%cd%

set train_data=%current_dir%\data\train.json
set dev_data=%current_dir%\data\dev.json
set table_data=%current_dir%\data\tables.json
set train_out=%current_dir%\data\train.lgesql.bin
set dev_out=%current_dir%\data\dev.lgesql.bin
set table_out=%current_dir%\data\tables.bin
set vocab_glove=%current_dir%\pretrained_models\glove.42b.300d\vocab_glove.txt
set vocab=%current_dir%\pretrained_models\glove.42b.300d\vocab.txt

echo "Start to preprocess the original train dataset ..."
python -u preprocess/process_dataset.py --dataset_path %train_data% --raw_table_path %table_data% --table_path %table_out% --output_path %c
echo "Start to preprocess the original dev dataset ..."
python -u preprocess/process_dataset.py --dataset_path %dev_data% --table_path %table_out% --output_path %current_dir%\data\dev.bin
echo "Start to build word vocab for the dataset ..."
python -u preprocess/build_glove_vocab.py --data_paths %current_dir%\data\train.bin --table_path %table_out% --reference_file %vocab_glove%
echo "Start to construct graphs for the dataset ..."
```

```
python -u preprocess/process_graphs.py --dataset_path %current_dir%\data\train.bin --table_path %table_out% --method lgesql --output_path %
python -u preprocess/process_graphs.py --dataset_path %current_dir%\data\dev.bin --table_path %table_out% --method lgesql --output_path %de
```

预计话费半个小时至一个小时，当然，你可以直接使用官方加工好的数据

由于windows‘s homepath和linux’s homepath不同，需要提前修改源码，

```
# 具体报错如下
Traceback (most recent call last):
  File "scripts/text2sql.py", line 33, in <module>
    Example.configuration(plm=params.plm, method=params.model)
  File "D:\DMIR\code\python\lgesql\utils\example.py", line 25, in configuration
    cls.word2vec = Word2vecUtils()
  File "D:\DMIR\code\python\lgesql\utils\word2vec.py", line 12, in __init__
    self.word_embed = GloveEmbedding('common_crawl_48', d_emb=300)
  File "D:\env\anaconda\envs\test\lib\site-packages\embeddings\glove.py", line 43, in __init__
    self.db = self.initialize_db(self.path(path.join('glove', '{}:{}.db'.format(name, d_emb))))
  File "D:\env\anaconda\envs\test\lib\site-packages\embeddings\embedding.py", line 22, in path
    root = environ.get('EMBEDDINGS_ROOT', path.join(environ['HOME'], '.embeddings'))
  File "D:\env\anaconda\envs\test\lib\os.py", line 681, in __getitem__
    raise KeyError(key) from None
KeyError: 'HOME'

# 解决方法，修改对应的embedding.py文件源码
vim D:\env\anaconda\envs\test\Lib\site-packages\embeddings
将environ['HOME']修改为environ['HOMEPATH']
```

## 使用Glove词向量训练LGESQL

windows训练脚本，实质是将linux shell参数进行一定调整

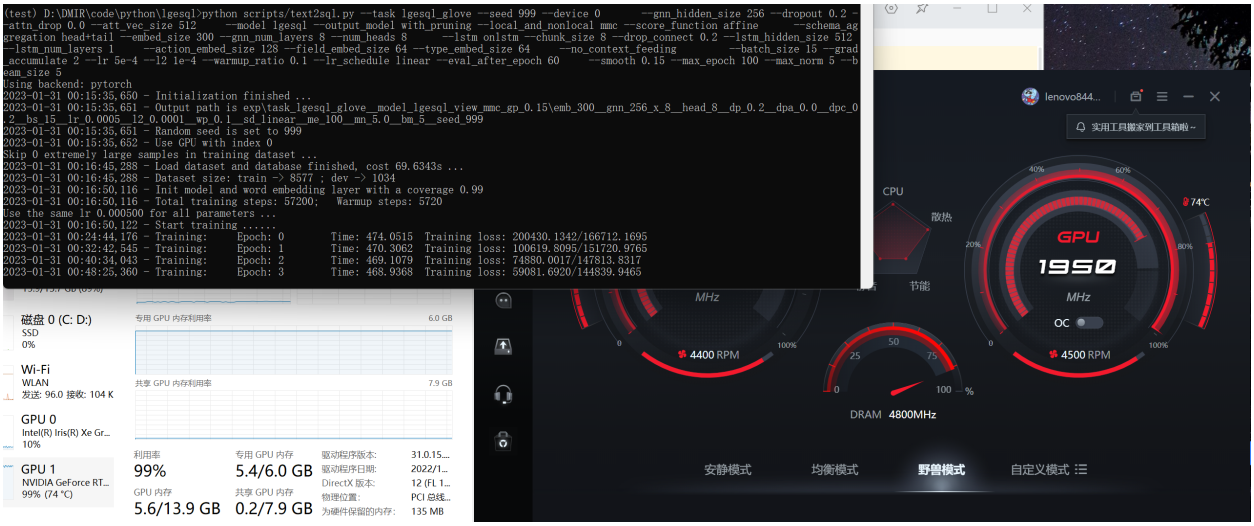
```
:: windows训练脚本
set task=lgesql_glove
set seed=999
set device=0
::'--testing'
set testing=
set read_model_path=

set model=lgesql
:: without_pruning  是否使用剪枝辅助工作以便提高encoder的识别能力
set output_model=with_pruning
:: mmc, msde, local   mmc:multi-head multi-view concatenation   msde: mixed static and dynamic embeddings 两种拆分头的方法
set local_and_nonlocal=%1
set embed_size=300
set schema_aggregation=head+tail
set gnn_hidden_size=256
set gnn_num_layers=8
set relation_share_heads=
set score_function=affine
set num_heads=8
set dropout=0.2
set attn_drop=0.0
set drop_connect=0.2

set lstm=onlstm
set chunk_size=8
set att_vec_size=512
set sep_cxt=
set lstm_hidden_size=512
set lstm_num_layers=1
set action_embed_size=128
set field_embed_size=64
set type_embed_size=64
set no_context_feeding=--no_context_feeding
set no_parent_production_embed=
```

```
set no_parent_field_embed=
set no_parent_field_type_embed=
set no_parent_state=

::set batch_size=20 如果你爆显存了 请将batch_size调低
set batch_size=10
set grad_accumulate=2
set lr=5e-4
set l2=1e-4
set smooth=0.15
set warmup_ratio=0.1
set lr_schedule=linear
set eval_after_epoch=60
set max_epoch=100
set max_norm=5
set beam_size=5

python scripts/text2sql.py --task %task% --seed %seed% --device %device% %testing% %read_model_path% ^
    --gnn_hidden_size %gnn_hidden_size% --dropout %dropout% --attn_drop %attn_drop% --att_vec_size %att_vec_size% ^
    --model %model% --output_model %output_model% --local_and_nonlocal %local_and_nonlocal% --score_function %score_function% %relation_sha
    --schema_aggregation %schema_aggregation% --embed_size %embed_size% --gnn_num_layers %gnn_num_layers% --num_heads %num_heads% %sep_cxt%
    --lstm %lstm% --chunk_size %chunk_size% --drop_connect %drop_connect% --lstm_hidden_size %lstm_hidden_size% --lstm_num_layers %lstm_num
    --action_embed_size %action_embed_size% --field_embed_size %field_embed_size% --type_embed_size %type_embed_size% ^
    %no_context_feeding% %no_parent_production_embed% %no_parent_field_embed% %no_parent_field_type_embed% %no_parent_state% ^
    --batch_size %batch_size% --grad_accumulate %grad_accumulate% --lr %lr% --l2 %l2% --warmup_ratio %warmup_ratio% --lr_schedule %lr_sched
    --smooth %smooth% --max_epoch %max_epoch% --max_norm %max_norm% --beam_size %beam_size%
```

此处为默认参数，可自行调整，用法如下：

```
:: mmc
.\run\windows\run_lgesql_glove.bat mmc

:: msde
.\run\windows\run_lgesql_glove.bat msde
```

运行过程如下，GPU能够跑满，但是不建议用个人PC跑




## 训练完成

```
2023-02-03 08:32:46,369 - Training:  Epoch: 56   Time: 327.5931    Training loss: 2318.7907/133252.0248
2023-02-03 08:38:18,604 - Training:  Epoch: 57   Time: 328.6506    Training loss: 2170.0029/133204.8183
2023-02-03 08:44:00,663 - Training:  Epoch: 58   Time: 338.4230    Training loss: 2028.7559/133173.0209
2023-02-03 08:49:51,809 - Training:  Epoch: 59   Time: 347.4726    Training loss: 1914.9682/133124.4307
2023-02-03 08:55:24,537 - Training:  Epoch: 60   Time: 329.0532    Training loss: 1824.5200/133093.9167
2023-02-03 08:57:11,789 - Evaluation:      Epoch: 60   Time: 103.3439    Dev acc: 0.6509
2023-02-03 08:57:12,065 - NEW BEST MODEL:      Epoch: 60   Dev acc: 0.6509
```

```python
# initialization params, output path, logger, random seed and torch.device
args = init_args(sys.argv[1:])
exp_path = hyperparam_path(args)
logger = set_logger(exp_path, args.testing)
set_random_seed(args.seed)
device = set_torch_device(args.device)
logger.info("Initialization finished ...")
logger.info("Output path is %s" % (exp_path))
logger.info("Random seed is set to %d" % (args.seed))
logger.info("Use GPU with index %s" % (args.device) if args.device >= 0 else "Use CPU as target torch device")
```

模型输出目录在日志中

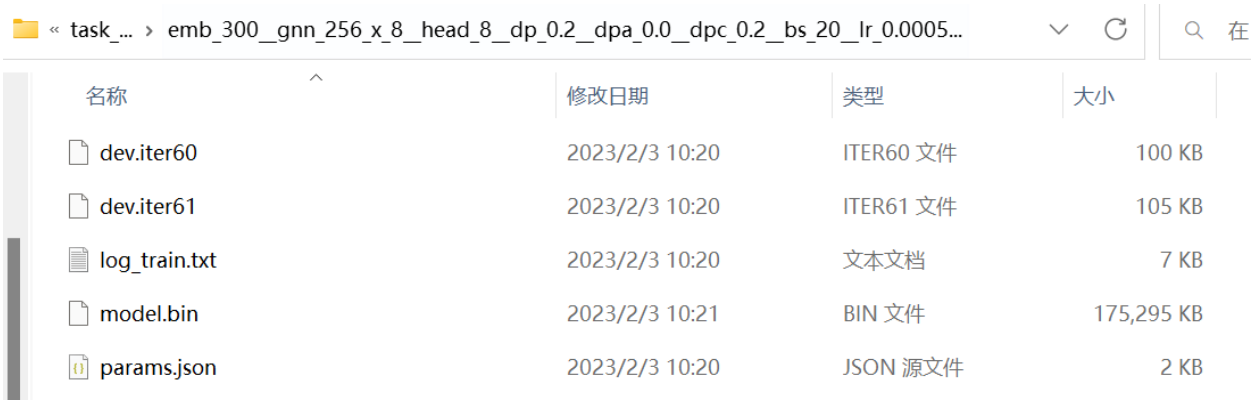
代码找到最好的模型之后会将 模型输出至日志所指的目录当中

跑完结果后会在`exp`目录会在日志中显现，比如

```
2023-02-03 03:08:50,334 - Initialization finished ...
2023-02-03 03:08:50,334 - Output path is exp/task_lgesql_glove__model_lgesql_view_mmc_gp_0.15/emb_300__gnn_256_x_8__head_8__dp_0.2__dpa_0.0__dpc_0.2__bs_20__lr_0.0005__l2_0.0001__wp_0.1__sd_linear__me_100
2023-02-03 03:08:50,334 - Random seed is set to 999
2023-02-03 03:08:50,334 - Use GPU with index 0
2023-02-03 03:09:49,242 - Load dataset and database finished, cost 58.9081s ...
2023-02-03 03:09:49,242 - Dataset size: train -> 8577 ; dev -> 1034
2023-02-03 03:09:52,771 - Init model and word embedding layer with a coverage 0.99
2023-02-03 03:09:52,771 - Total training steps: 42900;     Warmup steps: 4290
2023-02-03 03:09:52,775 - Start training
```

通过观察日志可以发现，模型程序已经帮我们自动保存在某个目录当中了.

对应参数目录如`D:\DMIR\code\python\lgesql-windows\exp\task_lgesql_glove__model_lgesql_view_mmc_gp_0.15\emb_300__gnn_256_x_8__head_8__dp_0.2__dpa_0.0__dpc_0.2__bs_20__lr_0.0005__l2_0.0`下生成这样几份文件，其中`model.bin`和`params.json`是最为重要的




```
# 创建save_models文件夹，根据输出日志找到跑出来的模型
mkdir saved_models
# 将模型中的model.bin和params.json文件拷贝至saved_models中，并以运行方式对文件夹命名
mkdir .\save_models\glove-mmc-63.5\
```

## Testing阶段

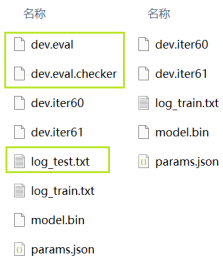
将该文件夹的文件复制到`save_models`下并规范命名，然后调整下代码脚本，设置`run_lgesql_glove.bat`两个参数

```
# 原本
set testing=
set read_model_path=

# 修改为
set testing=--testing
set read_model_path=--read_model_path .\save_models\glove-mmc-63.5\
```

执行**Testing**

```
.\run\windows\run_lgesql_glove.bat mmc
```

其会在`save_models`下生成三份文件方便后续的Evaluation




执行结果：

```
(test) D:\DMIR\code\python\lgesql-windows>python scripts/text2sql.py --task lgesql_glove --seed 999 --device 0 --testing
 --read_model_path D:\DMIR\code\python\lgesql-windows\save_models\glove-mmc-63.5\     --gnn_hidden_size 256 --dropout 0.
2 --attn_drop 0.0 --att_vec_size 512     --model lgesql --output_model with_pruning --local_and_nonlocal mmc --score_fun
ction affine      --schema_aggregation head+tail --embed_size 300 --gnn_num_layers 8 --num_heads 8      --lstm onlstm --
chunk_size 8 --drop_connect 0.2 --lstm_hidden_size 512 --lstm_num_layers 1     --action_embed_size 128 --field_embed_siz
e 64 --type_embed_size 64     --no_context_feeding         --batch_size 10 --grad_accumulate 2 --lr 5e-4 --l2 1e-4 --war
mup_ratio 0.1 --lr_schedule linear --eval_after_epoch 60     --smooth 0.15 --max_epoch 100 --max_norm 5 --beam_size 5
Using backend: pytorch
2023-02-03 18:05:52,913 - Initialization finished ...
2023-02-03 18:05:52,913 - Output path is D:\DMIR\code\python\lgesql-windows\save_models\glove-mmc-63.5\
2023-02-03 18:05:52,914 - Random seed is set to 999
2023-02-03 18:05:52,915 - Use GPU with index 0
Skip 0 extremely large samples in training dataset ...
2023-02-03 18:07:02,521 - Load dataset and database finished, cost 69.6044s ...
2023-02-03 18:07:02,525 - Dataset size: train -> 8577 ; dev -> 1034
2023-02-03 18:07:08,956 - Load saved model from path: D:\DMIR\code\python\lgesql-windows\save_models\glove-mmc-63.5\
2023-02-03 18:12:28,824 - Evaluation costs 319.87s ; Dev dataset exact match/checker/beam acc is 0.6509/0.6518/0.7669 .
```

## Evaluation阶段

执行评估

```
.\run\windows\run_evaluation.bat glove-mmc-63.5
```

评估完成

```
(test) D:\DMIR\code\python\lgesql-windows>python scripts/text6sql.py --task evaluation --testing --read_model_path saved
_models\glove-mmc-63.5     --batch_size 20 --beam_size 5 --device 0
Using backend: pytorch
2023-02-03 18:45:26,124 - Initialization finished ...
2023-02-03 18:45:26,125 - Output path is saved_models\glove-mmc-63.5
2023-02-03 18:45:26,125 - Random seed is set to 999
2023-02-03 18:45:26,125 - Use GPU with index 0
Skip 0 extremely large samples in training dataset ...
2023-02-03 18:46:28,748 - Load dataset and database finished, cost 62.6208s ...
2023-02-03 18:46:28,750 - Dataset size: train -> 8577 ; dev -> 1034
2023-02-03 18:46:38,461 - Load saved model from path: saved_models\glove-mmc-63.5
2023-02-03 18:51:59,760 - Evaluation costs 321.30s ; Dev dataset exact match/checker/beam acc is 0.6509/0.6518/0.7669 .
```

## Submission阶段

```
.\run\windows\run_submission.bat glove-mmc-63.5
```

```
(test) D:\DMIR\code\python\lgesql-windows>python eval.py --db_dir D:\DMIR\code\python\lgesql-windows\data\database --table_path D:\DMIR\code\python\lgesql-windo
ws\data\tables.json --dataset_path D:\DMIR\code\python\lgesql-windows\data\dev.json --saved_model saved_models\glove-mmc-63.5 --output_path saved_models\glove-m
mc-63.5\predicted_sql.txt --batch_size 10 --beam_size 5
Using backend: pytorch
2023-02-03 18:55:45 INFO: Loading these models for language: en (English):
=====================
| Processor | Package |
---------------------
| tokenize  | ewt     |
| pos       | ewt     |
| lemma     | ewt     |
=====================

2023-02-03 18:55:45 INFO: Use device: gpu
2023-02-03 18:55:45 INFO: Loading: tokenize
2023-02-03 18:55:47 INFO: Loading: pos
2023-02-03 18:55:47 INFO: Loading: lemma
2023-02-03 18:55:47 INFO: Done loading processors!
In total, process 166 databases .
In total, process 1034 samples , skip 0 extremely large databases.
Processing the 500-th example ...
Processing the 1000-th example ...
In total, process 1034 samples, skip 0 samples .
Start evaluating ...
Evaluation costs 205.3111s .

(test) D:\DMIR\code\python\lgesql-windows>python evaluation.py --gold D:\DMIR\code\python\lgesql-windows\data\dev_gold.sql --pred saved_models\glove-mmc-63.5\pr
edicted_sql.txt --db D:\DMIR\code\python\lgesql-windows\data\database --table D:\DMIR\code\python\lgesql-windows\data\tables.json --etype match  1>saved_models\
glove-mmc-63.5\evaluation.log
```

evaluation.log - 记事本

文件　编辑　查看

```
medium pred: SELECT AVG(singer.Age) , MIN(singer.Age) , MAX(singer.Age) FROM singer WHERE singer.ls_male = "value"
medium gold: SELECT avg(age) ,  min(age) ,  max(age) FROM singer WHERE country  =  'France'

medium pred: SELECT singer.Name , singer.Song_release_year FROM singer ORDER BY singer.Age ASC LIMIT 1
medium gold: SELECT song_name ,  song_release_year FROM singer ORDER BY age LIMIT 1

medium pred: SELECT singer.Name , singer.Song_release_year FROM singer ORDER BY singer.Age ASC LIMIT 1
medium gold: SELECT song_name ,  song_release_year FROM singer ORDER BY age LIMIT 1

medium pred: SELECT MAX(stadium.Capacity) , MAX(stadium.Average) FROM stadium
medium gold: select max(capacity), average from stadium

medium pred: SELECT AVG(stadium.Average) , MAX(stadium.Capacity) FROM stadium
medium gold: select avg(capacity) ,  max(capacity) from stadium
```

## 使用Glove词向量所训练出的模型（备份）

Google Drive: Sign-in

Access Google Drive with a Google account (for personal use) or Google Workspace account (for business use).

https://drive.google.com/file/d/1WGjP9_Wt-jNmtRA69xwBqTS0lTLM7o0v/view?usp=share_link

## 使用预训练模型训练LGESQL

windows训练脚本，实质是将linux shell参数进行一定调整，不过PLM还是不要在笔记本上跑了

```
:: .\run\windows\run_lgesql_plm.bat [mmc|msde] [bert-large-uncased-whole-word-masking|electra-large-discriminator]

set task=lgesql_large
set seed=999
set device=0
::'--testing'
set testing=
set read_model_path=

set model=lgesql
:: without_pruning  是否使用剪枝辅助工作以便提高encoder的识别能力
set output_model=with_pruning
:: mmc, msde, local   mmc:multi-head multi-view concatenation   msde: mixed static and dynamic embeddings 两种拆分头的方法
set local_and_nonlocal=%1
set plm=%2
set subword_aggregation=attentive-pooling
set schema_aggregation=head+tail
set gnn_hidden_size=512
set gnn_num_layers=8
set relation_share_heads=--relation_share_heads
set relation_share_heads=
set score_function=affine
set num_heads=8
set dropout=0.2
set attn_drop=0.0
set drop_connect=0.2

set lstm=onlstm
set chunk_size=8
set att_vec_size=512
set sep_cxt=
set lstm_hidden_size=512
set lstm_num_layers=1
set action_embed_size=128
set field_embed_size=64
set type_embed_size=64
set no_context_feeding=--no_context_feeding
set no_parent_production_embed=
set no_parent_field_embed=
set no_parent_field_type_embed=
set no_parent_state=

::set batch_size=20 如果你爆显存了 请将batch_size调低
set batch_size=5
set grad_accumulate=5
set lr=1e-4
set layerwise_decay=0.8
set l2=0.1
set smoothing=0.15
set warmup_ratio=0.1
set lr_schedule=linear
set eval_after_epoch=120
set max_epoch=200
set max_norm=5
set beam_size=5

python scripts/text2sql.py --task %task% --seed %seed% --device %device% %testing% %read_model_path% ^
    --plm %plm% --gnn_hidden_size %gnn_hidden_size% --dropout %dropout% --attn_drop %attn_drop% --att_vec_size %att_vec_size% ^
    --model %model% --output_model %output_model% --local_and_nonlocal %local_and_nonlocal% --score_function %score_function% %relation_sha
      --subword_aggregation %subword_aggregation% --schema_aggregation %schema_aggregation% --gnn_num_layers %gnn_num_layers% --num_heads %
    --lstm %lstm% --chunk_size %chunk_size% --drop_connect %drop_connect% --lstm_hidden_size %lstm_hidden_size% --lstm_num_layers %lstm_num
    --action_embed_size %action_embed_size% --field_embed_size %field_embed_size% --type_embed_size %type_embed_size% ^
    %no_context_feeding% %no_parent_production_embed% %no_parent_field_embed% %no_parent_field_type_embed% %no_parent_state% ^
    --batch_size %batch_size% --grad_accumulate %grad_accumulate% --lr %lr% --l2 %l2% --warmup_ratio %warmup_ratio% --lr_schedule %lr_sched
     --smoothing %smoothing% --layerwise_decay %layerwise_decay% --smooth %smooth% --max_epoch %max_epoch% --max_norm %max_norm% --beam_siz
```

使用方法

```
:: [mmc|msde]   [bert-large-uncased-whole-word-masking|electra-large-discriminator]
.\run\windows\run_lgesql_plm.bat [mmc|msde] [bert-large-uncased-whole-word-masking|electra-large-discriminator]
```