Headhunter

Headhunter is a novel encoding method for Multi-choice Question Answering. Headhunter can conduct interception and soft filtering.

knowledge Acquisition

Headhunter applies Elastic Search engine to retrieve knowledge from the Open Mind Common Sense (OMCS) corpus.

The search results are compressed to HeadhunterOMCS.zip. We attempted to upload HeadhunterOMCS.zip together with the software. However, the maximum file size allowed to upload is no more than 50M, and the size of the zip file (including software and HeadhunterOMCS.zip) is up to about 74M. Therefore, we merely submit the software on softconf system. Nevertheless, the dataset HeadhunterOMCS.zip can be downloaded by the following hyperlink (anonymous): https://drive.google.com/file/d/1TsHygMvyoS1KLz0uMe4YTjWjsgsALjuk/view.

Encoders

Headhunter is respectively coupled with BERT, RoBERTa and Albert which are used as the encoders. The sequence which is input into the pretrained models is organized as follows.

```
[CLS] q [SEP] o [SEP] k [SEP]
```

We employ the vector [CLS] as the knowledge-aware representation. For each pair of question q and option o, we produce [CLS] using a piece of knowledge. The considered knowledge item is selected from the search results. [CLS] is fed into Headhunter for computing the final representation.

Headhunter's Interceptor

Headhunter's interceptor is constructed with a **self-attention network** which is the same with that in transformer. The interceptor can intercept relevant information from all the retrieved knowledge items, regardless of whether the knowledge items are qualified or less qualified.

Headhunter's Soft Filter

Attention pooling layer is used as the soft filter. Using the filter, we can highlight the representative hidden states and eclipse the unrepresentative ones.

Classifier

A fully-connected layer is used as the final classifier which score the possibilities of options. We specify the most probable option as the prediction result.

Operation Guides

Training

Training Headhunter on GPU.

```
python train.py \
    --task_name "rerank_csqa" \
    --save_model_name "your_model_name" \
```

```
    --origin_model "pretrained_model_name" \ # only support BERT, RoBERTa and
Albert
    --cs_mode "QAconcept-Match" \ # only support this cs_mode now
    --learning_rate 2e-5 \
    --cs_len 5 \ # number of search results used during training
    --dev_cs_len 5 \ # number of search results used during development
    --output_dir "your_output_dir"
    --num_train_epochs 5 \
    --train_batch_size 1 \
    --eval_batch_size 1 \
    --gradient_accumulation_steps 20 \
    --check_loss_step 2000 \
    --fp16 \
    --seed 1 \
```

Headhunter also supports the training on TPUs. Use **--tpu** to enable TPU training.

Development

```
python train.py \
    --dev \
    --task_name "rerank_csqa" \
    --save_model_name "your_model_name" \
    --origin_model "pretrained_model_name" \ # only support BERT, RoBERTa and
Albert
    --cs_mode "QAconcept-Match" \ # only support this cs_mode now
    --dev_cs_len 5 \
    --output_dir "your_output_dir"
    --eval_batch_size 1 \
    --gradient_accumulation_steps 20 \
    --fp16 \
    --seed 1 \
```

Best Setting

Headhunter's best single model accuracy is 83.3%/78.4% on the dev/test dataset.

Hyperparameters of our best model (Albert plus Headhunter) are set as follows.

```
encoder: albert-xxlarge-v2
device: NVIDIA Tesla V100 SXM2 16GB * 1
max_seq_length: 80 (64 for question & option, 16 for knowledge)
learning_rate: 1e-5
train_batch_size: 2
gradient_accumulation_steps: 10
cs_len: 7
dev_cs_len: 8
optimizer: AdamW
adam_epsilon: 1e-8
training steps: 9742(2 epoch)
tp16: True
```