BERT(自然言語処理)の研究

ヤス

BERTとは

- BERTとは
  - Googleが発表した自然言語処理モデル
    - 事前学習によって精度を向上させている
      - ネット上の大量の文章データ(Wikipedia等)で事前に学習
        - データセットの不足を補う
    - 文脈や感情の分析が可能
    - 別のモデルに転移学習（Fine-tuning）が可能
      - 既存の学習モデルの前にBERTを利用するだけで精度が向上
    - 事前学習には時間がかかる
      - 日本語を事前学習したモデルがある(1.6GB)
        - 京都大学が公開

BERTの設定環境

- 設定環境
  - Ubuntu：18.04
  - Python：3.6
    - 追加ライブラリ：numpy,tensorflow,apt_pkg
  - JUMANN++（形態素解析システム）
    - 追加で必要：pyknpとBoostライブラリ
  - BERT
    - Gitで公開されている(Clone)
  - BERT日本語Pretrainedモデル
    - 京都大学が公開しているモデル（ 1.6GB）
  - GPU（グラフィックボード）はあった方が良い
    - GPUの方が計算が早い

BERTの試用（文字の推測）

- 下記のサイトを参考にコーディング
  - 参考URL：BERT日本語モデルを使って、クリスマスプレゼントに欲しいものを推測してみた

- 下記の文章から文字を推測
  - 「バレンタインは相手に＊を贈る日です。」
    - ＊の部分を推測させる

```
import torch
from transformers import BertTokenizer, BertForMaskedLM, BertConfig
import numpy as np
#Bertの設定ファイルとモデルの設定
config = BertConfig.from_json_file('/home/be-s/bert/Japanese_L-12_H-768_A-12_E-30_BPE/bert_config.json')
model = BertForMaskedLM.from_pretrained('/home/be-s/bert/Japanese_L-12_H-768_A-12_E-30_BPE/pytorch_model.bin', config=config)
bert_tokenizer = BertTokenizer('/home/be-s/bert/Japanese_L-12_H-768_A-12_E-30_BPE/vocab.txt',do_lower_case=False, do_basic_tokenize=False)
from pyknp import Juman
jumanpp = Juman()

from pyknp import Juman
jumanpp = Juman()
#文章を設定(推測部分は＊)と設定
text = "バレンタインは相手に＊を贈る日です。"
#jumanppで形態素解析
result = jumanpp.analysis(text)
#解析結果をBERT分析用に整形
tokenized_text = [mrph.midasi for mrph in result.mrph_list()]
#文頭に[CLS]を追加(分類問題 - 文字推測)用の特殊トークン
tokenized_text.insert(0, '[CLS]')
#文末に[SEP]を追加 - 文末を表す特殊トークン
tokenized_text.append('[SEP]')
#マスク(推測 - ＊)する文字のインデックスを指定
masked_index = 5
tokenized_text[masked_index] = '[MASK]'
print(tokenized_text)

#BERT処理用にIDに変換
tokens = bert_tokenizer.convert_tokens_to_ids(tokenized_text)
tokens_tensor = torch.tensor([tokens])

model.eval()

#GPU演算用
#tokens_tensor = tokens_tensor.to('cuda')
#model.to('cuda')

#CPU演算設定
tokens_tensor = tokens_tensor.to('cpu')
model.to('cpu')

#入力データから文字を推測
with torch.no_grad():
 outputs = model(tokens_tensor)
 predictions = outputs[0]

_, predicted_indexes = torch.topk(predictions[0, masked_index], k=5)
predicted_tokens = bert_tokenizer.convert_ids_to_tokens(predicted_indexes.tolist())
print(predicted_tokens)
```

BERTの試用（文字の推測）

- 推測結果
  - それっぽい言葉が出てきている

```
['プレゼント', '花', '贈り物', '[UNK]', 'チョコレート']
```

BERTの試用（文字の推測）

- 別のパターン
  - 学習データがWikiなのでWikiに情報がない（少ない）場合は精度が悪い

```
text = "トヨタは米のメーカーです。"
```

```
['日本', '世界', '自動車', '唯一', 'トヨタ']
```

```
text = "DODは米のメーカーです。"
```

```
['[UNK]', '日本', '中国', 'カメラ', '靴']
```

BERTを使ってみた感想

- 事前学習のデータを使えば精度が向上しそう
  - データ不足も補えるので、社内システムに導入できるかも
- 本格的に導入するならGPU（グラフィックボード）があった方が良さそう
  - 今回の文章の推測にかかった時間：10秒/回