PowerAI 人工智能马拉松编程大赛

Hackathon for Finance Industry

作品展示-第十四组

（大赛二等奖）

CSDN Power Systems

作品展示

- Character level RNN的multi-task learning in Tensorflow

Py文件是一体化的python代码，包含训练和测试。

Input_data.json是将分散的训练样本集中后的结果。我们对所有的文字（含标点符号换行符 等）建立字典，将原始文件转化成相应的数值。

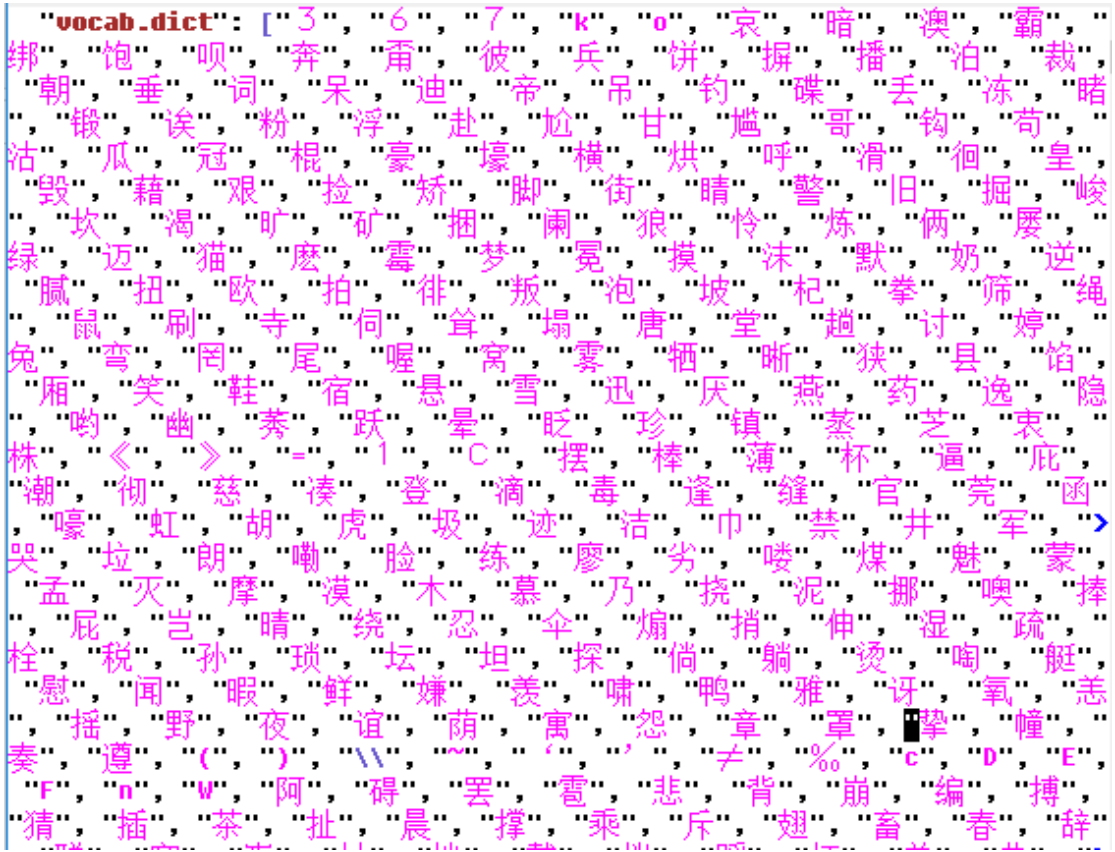

```
"denc": [
  {
    "nm": "1-1-10",
    "txt": [1777, 1782, 1784, 1766, 1684, 1790, 1778, 1617, 1768, 1765, 17
47, 1783, 1709, 1770, 1453, 1387, 1792, 1759, 1584, 1793, 1785, 1781, 1788,
1735, 1439, 1720, 1441, 1521, 1792, 1563, 1744, 1791, 1760, 1786, 1621, 1785
, 1781, 1764, 1762, 1730, 1729, 1793, 1609, 1783, 1669, 1522, 1736, 1572, 17
83, 1669, 1307, 1793, 1762, 1575, 1785, 1707, 1572, 1759, 1792, 761, 1380, 1
791, 1760, 1786, 1621, 1678, 1550, 1792, 1785, 1781, 1665, 1751, 1638, 1786,
 1411, 1272, 1787, 1449, 1546, 1643, 1449, 1679, 1646, 1792, 1743, 1793, 178
7, 833, 1211, 1754, 1654, 1793, 1449, 1773, 1746, 1792, 1743, 1756, 1787, 16
55, 788, 1791, 1751, 1754, 1654, 1301, 1643, 1501, 1542, 1791, 1774, 1189, 1
788, 1737, 1679, 1646, 1501, 1441, 1793, 1666, 1788, 1737, 1754, 1765, 1501,
 1441, 1791, 1789, 1780, 1782, 767, 1793, 1790, 1764, 1691, 1751, 1790, 1472
, 1470, 1760, 1786, 1621, 1793, 1657, 1778, 1707, 1052, 1747, 1735, 1779, 17
93, 1790, 1756, 1740, 1759, 1637, 1766, 1766, 1453, 1715, 1657, 1778, 1791],
    "lab1": 1,
    "lab2": 1
  },
```

Power Systems

CSDN

作品展示

- 一个文件和相应label就是上面一条记录；
- 字典就是数组： 如右图， 数字1对应到全角字符3.
- 字典只cover训练样本里出现的字符集；如果再测试数据是没有这样的数据，我们简单地忽略掉这个字。



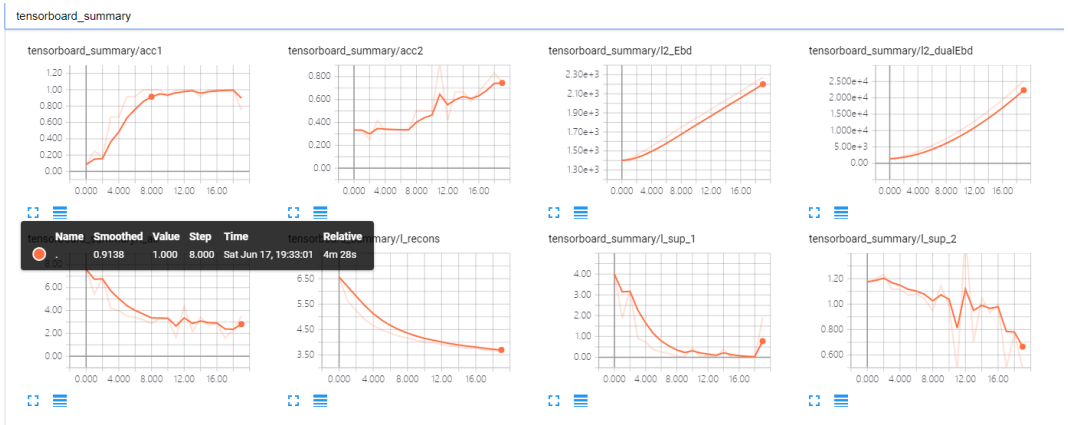
作品展示

- 训练时的截图：出现的文字时RNN重构的随机对话，用来监测RNN本身的状态如何。虽然乍看模型重构的sample乱七八糟，但还是反映了稍许人类语言的特征。

```
Generated Sample:BOLA: 有一哪分？好

A:     我您做盘拿您一重别,吧源烦了哪识保险工位4在6,碟体，我们这部。公司少吧3字点解票帮助急时推的痛！性怜您实际来说内，银是没有时客户建账撞来说的爱是这款间孩子
          iter         l_all       l_recons       l_sup_1       l_sup_2        acc1        acc2      l2_Ebd   l2_dualEbd
         9/   0       2.85421        4.16728       0.06602       0.94858       1.00000     0.50000   1748.57678   7544.87402
         9/ 100       2.29780        4.13258       0.01778       0.74670       1.00000     0.58333   1775.28613   8157.53027
         9/ 180       2.31696        4.06365       0.02478       0.75432       1.00000     0.50000   1797.26038   8688.56641
Generated Sample:BOLB:那么出公司可以大经选择得心万至钱？
A: 有什么出模闲理的倍买保险可以什么物部金保赔往病病疾病与字我希望度保险的公司失吗的收产和所以为保类保养子保险都说的保险就家、保险公司，并上、。而安的东西会员
          iter         l_all       l_recons       l_sup_1       l_sup_2        acc1        acc2      l2_Ebd   l2_dualEbd
        10/   0       3.28197        4.08090       0.05137       1.12900       1.00000     0.50000   1797.67664   8763.10938
        10/ 100       1.99158        4.06328       0.03040       0.62193       1.00000     0.75000   1822.09583   9415.30762
        10/ 110       3.07002        4.05748       0.05223       1.04481       1.00000     0.50000   1825.71533   9460.84082
```

- Tensorboard监控的是：在两个小验证集上，后面两个任务的分类器的状态。
- 截图里可以看到的是：production的loss下降的很快；但是intension的loss下降很慢。



作品展示

- 370个iteration后

```
Generated Sample:BOLA: 是索买保么途控吗？
好的国点做风不需要是您做您投资人上股还能够否。比就是把握个少金：B：多少多了高太多少数不行情个好像没不多资对风险可下了么并不是，股票
了的收人情，签是可在人，到老师帮也有助，
       iter          l_all       l_recons       l_sup_1       l_sup_2          acc1          acc2         l2_Ebd     l2_dualEbd
      377/   0        1.06819        3.45954       0.00001       0.28749       1.00000       0.91667   14591.61230 1.75979550e+06
      377/ 100        1.18990        3.45439       0.00000       0.33637       1.00000       0.83333   14606.74023 1.76198738e+06
      377/ 180        1.43483        3.45911       0.00000       0.43415       1.00000       0.83333   14612.23730 1.76474750e+06
Generated Sample:BOLA:先生养做现法在居长怎么？

过风利将来，爱=，康，在，了却国一至乱险，即不是最低孩子的是人的准备肯定期保渠不单常年，止入7年这样，我提供教心的B:好的生任，买的？EOL
       iter          l_all       l_recons       l_sup_1       l_sup_2          acc1          acc2         l2_Ebd     l2_dualEbd
      378/   0        1.41936        3.46052       0.00000       0.42791       1.00000       0.83333   14613.22754 1.76511650e+06
      378/ 100        1.10210        3.45351       0.00005       0.30126       1.00000       0.83333   14627.31543 1.76734150e+06
      378/ 180        1.28063        3.47283       0.00000       0.37192       1.00000       0.91667   14636.26660 1.77012125e+06
Generated Sample:BOLB: 购买家人火是市市场士，员财投资才即可，20万的观部分、入参加的时提的基金种理财品以上与客户，我们的用很一样：场：这个一的也么会看说的议：嗯2000考再点新一投资00%的帮来，固没正用资其它效等一
```

作品展示

- 最终结果生成的json文件如下：

production (D:/hackthon_power/rnn) - GVIM2

File Edit Tools Syntax Buffers Window C/C++ Perl Help


```
{
  "type": "Fiance Product Classifcation",
  "result": [
    {
      "sampleId": "1-4-65.txt",
      "prob": 0.5137633085250854,
      "label": "疑问"
    },
    {
      "sampleId": "1-3-127.txt",
      "prob": 0.9584351778030396,
      "label": "肯定"
    },
    {
      "sampleId": "2-1-115.txt",
      "prob": 0.517318069934845,
      "label": "否定"
    },
    {
      "sampleId": "2-1-205.txt",
      "prob": 0.6584604978561401,
      "label": "否定"
    },
    {
      "sampleId": "2-1-189.txt",
      "prob": 0.42310407757759094,
      "label": "肯定"
    },
    {
      "sampleId": "1-4-75.txt",
      "prob": 0.9994295239448547,
      "label": "疑问"
    },
    {
      "sampleId": "2-1-189.txt",
      "prob": 0.9999748468399048,
      "label": "股票"
    },
    {
      "sampleId": "1-4-75.txt",
      "prob": 0.9999995231628418,
      "label": "家庭财产保险"
    },
    {
      "sampleId": "1-1-103.txt",
      "prob": 0.9999996423721313,
      "label": "健康险"
    },
    {
      "sampleId": "1-4-111.txt",
      "prob": 0.9999034404754639,
      "label": "家庭财产保险"
    },
    {
      "sampleId": "1-2-105.txt",
      "prob": 0.9993556141853333,
      "label": "意外伤害险"
    },
    {
      "sampleId": "1-5-29.txt",
      "prob": 1.0,
      "label": "车险"
    },
    {
      "sampleId": "1-6-10.txt",
```


Intention 1,1 Top production 24,1

- 有个程序的使用细节，用—help显示命令参数细节

```
D:\hackthon_power\rnn>python rnn_classifier.py --help
usage: rnn_classifier.py [-h] [--nepoch NEPOCH] [--save_path SAVE_PATH]
                         [--train_file TRAIN_FILE] [--encoding ENCODING]
                         [--dim_x DIM_X] [--lr_a LR_A] [--lr_b LR_B]
                         [--save_per_iter SAVE_PER_ITER]
                         [--export_file EXPORT_FILE]
                         [--rate_true_step RATE_TRUE_STEP]
                         [--rate_true_val RATE_TRUE_VAL]
                         [--net_type {lstm,gru}] [--LOG_DIR LOG_DIR]
                         [--test_dir TEST_DIR] [--valid_mode VALID_MODE]
                         {train,test,export,continue,ebd,sample}

positional arguments:
  {train,test,export,continue,ebd,sample}
                        train/test/sample model, export to numpy, or continue
                        last training or export the embedding to visualize on
                        tensorboard

optional arguments:
  -h, --help            show this help message and exit
  --nepoch NEPOCH       number of epoch in training (default: 10000)
  --save_path SAVE_PATH
                        model save path (default: rnn_class)
  --train_file TRAIN_FILE
                        json file as training data (default: input_data.json)
  --encoding ENCODING   encoding of training_file (default: utf-8)
  --dim_x DIM_X         embedding dim (default: 120)
  --lr_a LR_A           a of lr = 1 / (a + b*n) (default: 1000)
  --lr_b LR_B           b of lr = 1 / (a + b*n) (default: 5)
  --save_per_iter SAVE_PER_ITER
                        save per this epoch (default: 10)
  --export_file EXPORT_FILE
                        export filename (default: dump.rnn.class.pickle)
  --rate_true_step RATE_TRUE_STEP
                        1 if n<= step (default: 100)
  --rate_true_val RATE_TRUE_VAL
                        val if n> step (default: 0.5)
  --net_type {lstm,gru}
                        lstm or gru (default: lstm)
  --LOG_DIR LOG_DIR     tensorboard path (default: ./logdir)
  --test_dir TEST_DIR   dir for testdata (default: ./testdata)
  --valid_mode VALID_MODE
                        exclude validation data in training sample (default:
                        False)
```

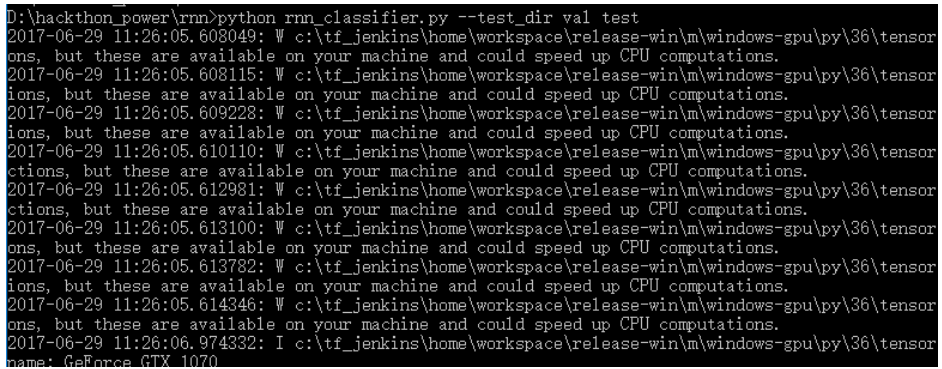
Power Systems

CSDN

作品展示

- 有些命令还未完全实现。但是train和test是完整的：--nepoch 100 train 就可以启动一个train

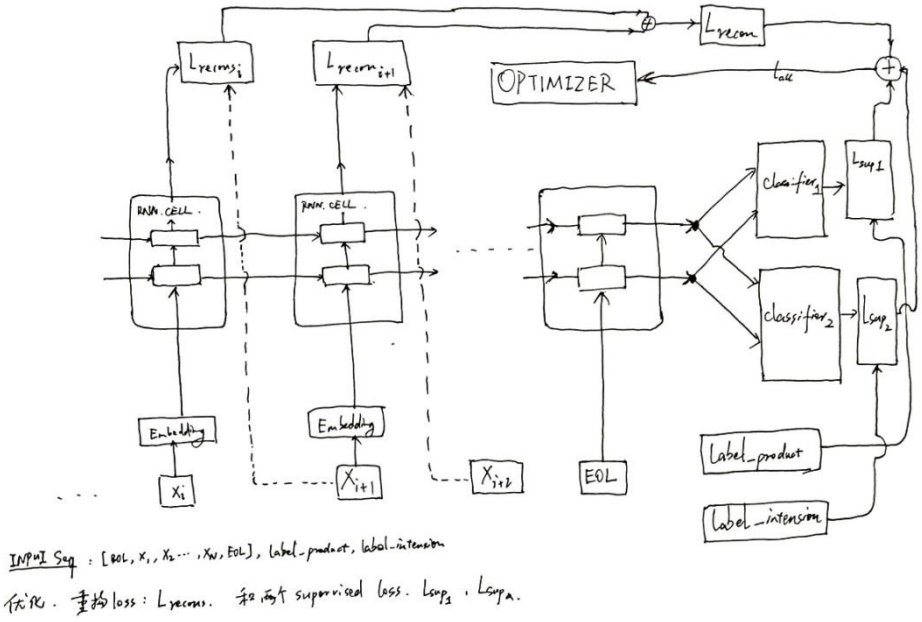
- 当train了若干epoch后，session会dump到disk上，这时就可以运行test来产生用于评估的结果了。



作品展示

① 基本思想： 使用character level RNN的multi-task learning的框架

② 框架图：



作品展示

③ 一边让RNN学会重构输入的序列

④ 一边让RNN的学会为两个分类准备合适的feature

⑤ 端到端的建模和训练思路

⑥ 最终的loss由重构的loss和两个分类的loss以适当的权重配比加和而成。

⑦ 由于Intension的训练比较困难，我们加大了其再total loss中的权重

```
with tf.name_scope('losses'):
    l2_Ebd      =  tf.reduce_sum(tf.pow(Ebd    ,2))
    l2_dualEbd =  tf.reduce_sum(tf.pow(dualEbd,2))
    reg_coeff  =  tf.placeholder_with_default(1e-9,(),name='regularizer_coeff')
    reg_recons =  tf.placeholder_with_default(1e-1,(),name='recons_weight')
    # here we use larger weight for l_sup_2, because it's harder to train it
    weight_sup2=  tf.placeholder_with_default(2.5,(),name='sup2_weight')
    l_sup_1     =  tf.reduce_mean(tf.nn.sparse_softmax_cross_entropy_with_logits(labels=label_input_1, logits=pred_product))
    l_sup_2     =  tf.reduce_mean(tf.nn.sparse_softmax_cross_entropy_with_logits(labels=label_input_2, logits=pred_comment))
    l_all = reg_recons*l_recons + l_sup_1 + weight_sup2 * l_sup_2 + reg_coeff * (l2_Ebd + l2_dualEbd + tf.abs(l2_Ebd - l2_dualEbd))
```

作品展示

⑧ 字首先由嵌入映射到稠的向量，这个嵌入是一并train出来的

⑨ 两个分类器各自拥有自己的待train变量，共享的是嵌入和RNN的部分

⑩ 原始文本不做任何变换和清洗，是什么就是什么，换行符就是换行符输入。唯一的变化是首位添上BOL（begin of line）和EOL（end of line）标记。

⑪ 代码可以支持多层的RNN结构，支持LSTM/GRU作为其中的CELL；Classifier也可以支持多层；但实际由于训练时间和其他的约束，我们选取的就是代码中的参数。

⑫ RNN是自己实现的，没有用tensorflow的现成的库，好处是支持任意动态的串长度，不需要对齐。坏处是做batching的时候，只要长度一样的串才能batch在一起train，GPU利用率不高。实际在CPU上还可能跑地更快些。

⑬ 由于我们的模型不依赖于分词，所有Extra地Corpus用不上。

个人评价

◆有关比赛

- 我们团队是来自于同一公司同一部门的同事（齐鲁资管 对冲基金部），大赛的组织很到位。由于工作内容十分相关，看到大赛信息后我们立即决定参赛，最后还获得了奖励，可喜可贺：）
- 这次大赛对于AI领域和金融领域来说都是难得的一次盛会。赛程中我们结识了周围的一些新小伙伴，最后还一起拍了合照，整个赛程感到收获很多，特别的开心。对于比赛的QQ群，QQ群管理员热心回答问题，还上传了足够的有关赛程和比赛机器环境的资料。我们团队在赛前虽完全没有碰过比赛环境，但最后也顺利的运行的训练和测试的代码，可以归功于出色的比赛管理。
- 环境问题上唯一要诟病的是网速问题，从一开始的下载数据，到实际执行，在终端上经常连回显字符都要花费数秒时间，还希望将来有所改进。

个人评价

◆有关作品

- 我们最初是想做精细一点的模型，比如分词，按照自然的对话结构建模。但实际拿到数据后，花费了近两小时的数据清洗，发现异常太多，不可能在剩下的时间里完成。这时我们分两种思路继续：

    1. 经典的词频方法，产生bag-of-word feature，训练分类器
    2. 使用RNN读入可变长的字串，提取特征后，直接送到分类器。做端到端的模型。

最终提交的是RNN的模型。

评委评价

“这次的比赛在赛题上设置了一些陷阱题，比如在一部分对话中加入了大量不同金融的理财产品的名字，如果参赛选手使用了词频的方式来做，那么准确率会大大降低。同时，在客户的购买意愿上，大量使用了双重否定，因此也在一定程度上增加了题目的难度。

第十四组参赛队在整个比赛中使用了自己设计的神经网络来解决词向量和分类问题. 由于给的语聊是没有打过标签的, 参赛团队使用了字向量的方式来绕开标签问题，结果正面整体效果还是不错的，但是也带来了一个新的问题那就是效率，所以我们最后看到在性能的比试上，该团队没有能够进入前5名。但是他们的lstm的分类神经网络设计的很好，所以,准确率在第二次提交的时候达到了99%。”

Power Systems

CSDN

团队风采