AIE10-1117

曹斌鑫

作业 A： 网络模型搭建作业

请完成程序所需部分，仅需要完成模型描述，不需要训练。

模型任务为进行中文文本分类任务。

```python
import tensorflow as tf
class params():
    """
    定义参数
    """
    vocab_size = 6000
    embedding_dim = 128
    n_layers = 2
    hidden_dim = 128
    words_len = 100
    batch_size = 32
    n_classes = 10

class TextRNN():
    def __init__(self, par):
        self.par = par
        self.build_model()
    def build_model(self):
        """
        构建模型函数
        """
        # 定义计算图
        self.graph = tf.Graph()
        # 在定义的计算图下构建模型
        with self.graph.as_default():
            # 定义输入文本
            self.inputs = tf.placeholder(tf.int32, [self.par.batch_size,
                                                    self.par.words_len],
name='words_id')
            # 定义标签
            self.labels = tf.placeholder(tf.int32, [self.par.batch_size])
            # 标签 onehot转化
            labels_onehot = tf.one_hot(self.labels, self.par.n_classes, 1, 0)

            # TODO(YU):Embedding
            embedding = tf.get_variable('embedding', [self.par.vocab_size,
self.par.embedding_dim])
```

```
            embedding_inputs = tf.nn.embedding_lookup(embedding, self.inputs)
            # TODO(YU):定义多层 RNN网络
            rnn_fn = tf.nn.rnn_cell.BasicLSTMCell
            cell = tf.nn.rnn_cell.MultiRNNCell(
                [rnn_fn(self.par.hidden_dim) for itr in range(self.par.n_layers)],# 状态向量长
度
                state_is_tuple=True)
            # TODO(YU):输入 RNN网络，并获取输出
            outputs, last_state = tf.nn.dynamic_rnn(cell = cell, inputs = embedding_inputs,
dtype=tf.float32)
            # TODO(YU):转化为分类问题输出
            #  全连接层，后面接 dropout以及 relu激活
            fc = tf.layers.dense(last, self.par.hidden_dim, name='fc1')
            fc = tf.contrib.layers.dropout(fc, 0.5)
            fc = tf.nn.relu(fc)
            self.logits = tf.layers.dense(fc, self.par.n_classes, name='fc2')
            self.y_pred_cls =  tf.argmax(tf.nn.softmax(self.logits), #  预测类别
# 计算交叉熵，并计算损失函数
            cross_entropy =  tf.nn.softmax_cross_entropy_with_logits_v2(logits=self.logits,
labels=labels_onehot)
            self.loss = tf.reduce_mean(cross_entropy)
            #  获取变量列表
            self.var_list = tf.global_variables()
        #  输出计算图
        self.summary = tf.summary.FileWriter("logdir", graph=self.graph)
    def train(self, inputs, labels):
        """
         训练函数，有兴趣可以补全
        """
        pass

if __name__ == "__main__":
    par = params()
    rnn_net = TextRNN(par)
```

作业B：自行训练模型

- 将预训练模型 model文件夹删除，自行训练模型用于文本分类
- 观察迭代多少次后精度达到 90%
- 记录平均每次迭代时间

```
Epoch: 1Iter:        0, Train Loss:    0.14, Train Acc:   96.09%, Val Loss:    0.42, Val Acc:   89.42%,
Time: 0:02:12 *
Iter:      100, Train Loss:    0.22, Train Acc:   92.97%, Val Loss:    0.57, Val Acc:   83.74%, Time:
```

0:57:09
Iter: 200, Train Loss: 0.21, Train Acc: 90.62%, Val Loss: 0.48, Val Acc: 87.04%, Time: 2:05:36
Iter: 300, Train Loss: 0.077, Train Acc: 97.66%, Val Loss: 0.37, Val Acc: 90.08%, Time: 3:18:04 *
Epoch: 2
Iter: 400, Train Loss: 0.15, Train Acc: 94.53%, Val Loss: 0.42, Val Acc: 89.42%, Time: 4:34:03
Iter: 500, Train Loss: 0.14, Train Acc: 96.88%, Val Loss: 0.38, Val Acc: 90.36%, Time: 5:47:43
Iter: 600, Train Loss: 0.14, Train Acc: 95.31%, Val Loss: 0.43, Val Acc: 88.70%, Time: 7:00:02
Iter: 700, Train Loss: 0.13, Train Acc: 96.88%, Val Loss: 0.34, Val Acc: 90.78%, Time: 8:12:31 *
Epoch: 3
Iter: 800, Train Loss: 0.078, Train Acc: 94.53%, Val Loss: 0.35, Val Acc: 90.72%, Time: 9:23:07

迭代 3 次后精度达到 90.08%，
第 1 次迭代用时 107 分钟，第 2 次迭代用时 55 分钟，第 3 次迭代用时 68 分钟。第 4 次迭代 73 分钟，第 5 次迭代 76 分钟，第 6 次迭代 73 分钟，第 7 次迭代 73 分钟，第 8 次迭代 72 分钟，第9 次迭代 71 分钟。第4 次迭代以后准确率比较稳定，没有太大变化，基本在90% 左右。