RNNs(分析评论积极与消极)

1. 处理数据
2. 读取数据
3. 查看数据是否需要处理，比如空格标点等
4. 划分数据特征和目标
5. 数据占位符 tf.placeholder(float32, shape, name=’’)
6. 把数据集分batch

1)batch的size太小，使得训练太慢，但如果size太大，准确率会降低

1. 初始化model
2. 定义嵌入层Embedding layer，定义一个查找表

Embedding = tf.Variable(tf.random\_unifrom(n\_words, embed\_size), -1, 1)

Embed = tf.nn.embedding\_lookup(Embedding, inputs\_)

1. 定义基础LSTM神经元

Lstm = tf.contrib.rnn.BasicLSTMCell(lstm\_size) lstm\_size一般为128，256，512

然后定义dropout

Drop = tf.contirb.rnn.DropoutWrapper(Lstm, output\_keep\_prob=keep\_prob)

堆叠多个lstm层

Cell = tf.contrib.rnn.MultiRNNCell([Drop]\*lstm\_layers)

初始化状态

Initial\_state = Cell.zero\_state(batch\_size, tf.float32)

1. Forward pass

Outputs, final\_state = tf.nn.dynamic\_rnn(Cell, inputs\_, initial\_state=initial\_state)

只有outputs的最后一个结果才对后面传播有用

Predictions = tf.contrib.layers.fully\_connected(outputs[:, -1], 1, activation\_fn=tf.sigmoid)

Cost = tf.losses.mean\_squared\_error(labels\_, predictions)

Optimizer = tf.train.AdamOptimizer(learning\_rate).minimize(Cost)

1. 准确率

Correct\_pred = tf.equal(tf.cast(tf.round(predictions), tf.int32), labels\_)

Accuracy = tf.reduce\_mean(tf.cast(correct\_pred, tf.float32))

1. 训练

For e in range(epochs):

//初始化变量

For x, y in get\_batches:

Feed=以字典方式传入参数

//获取想要的

Loss, state, \_ = sess.run(sess.run([cost, final\_state, optimizer], feed\_dict=feed))

1. 验证

1)通过valid数据进行训练完模型的验证

1. 测试

1)用测试集数据测试数据