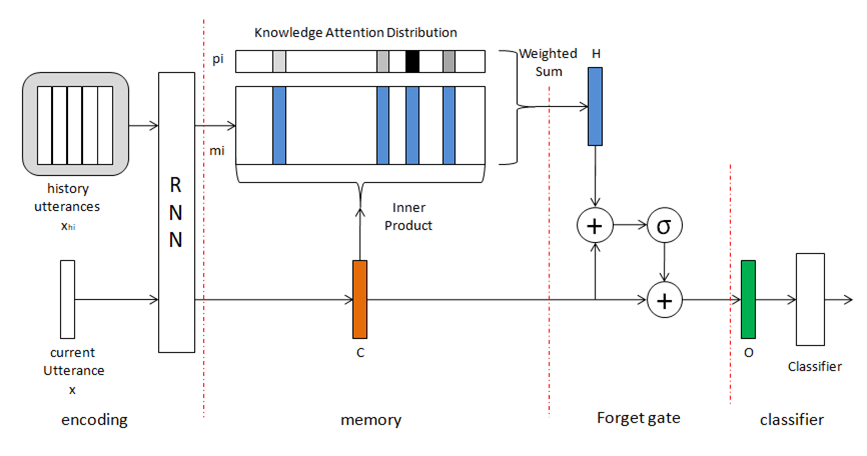
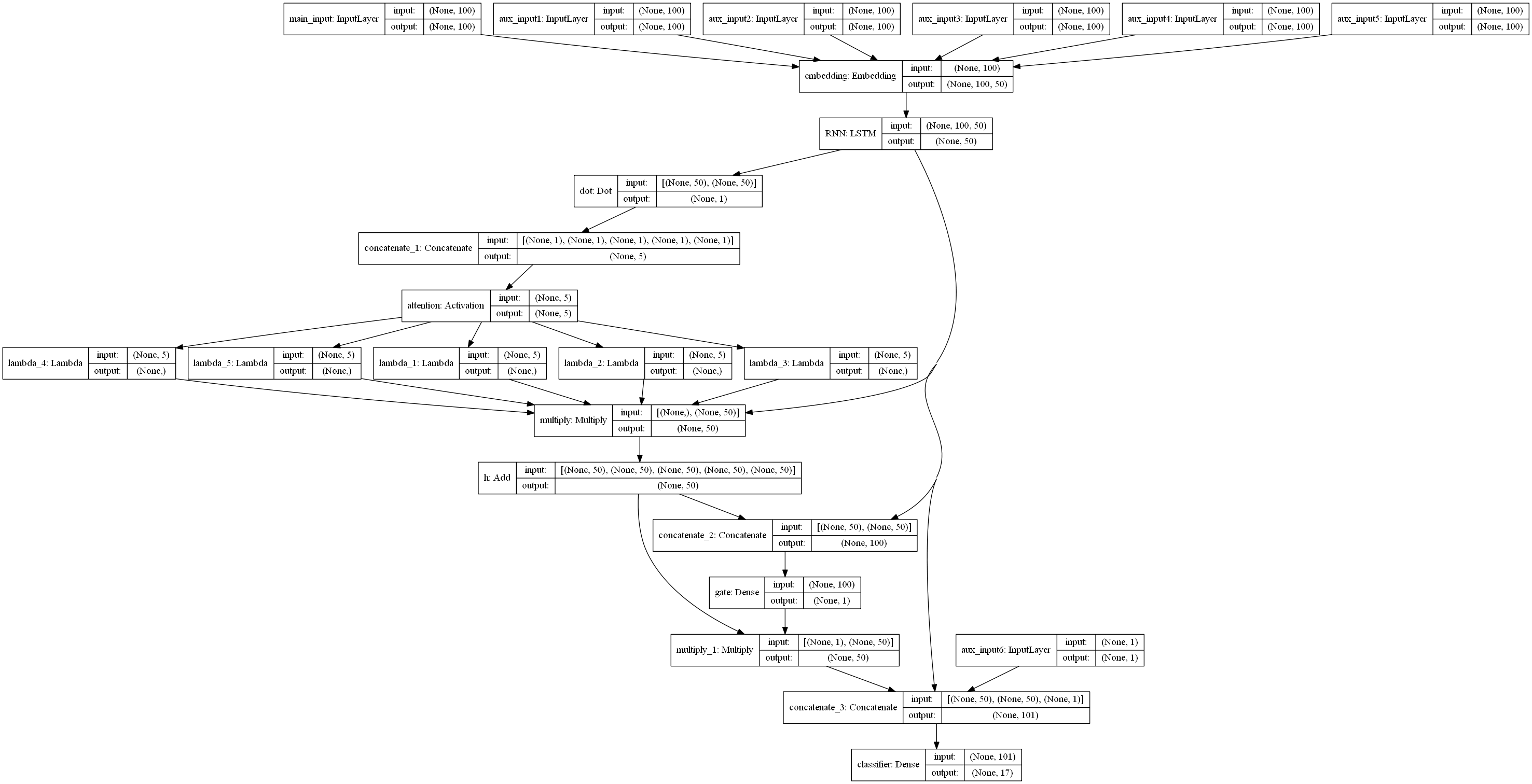
# 周报

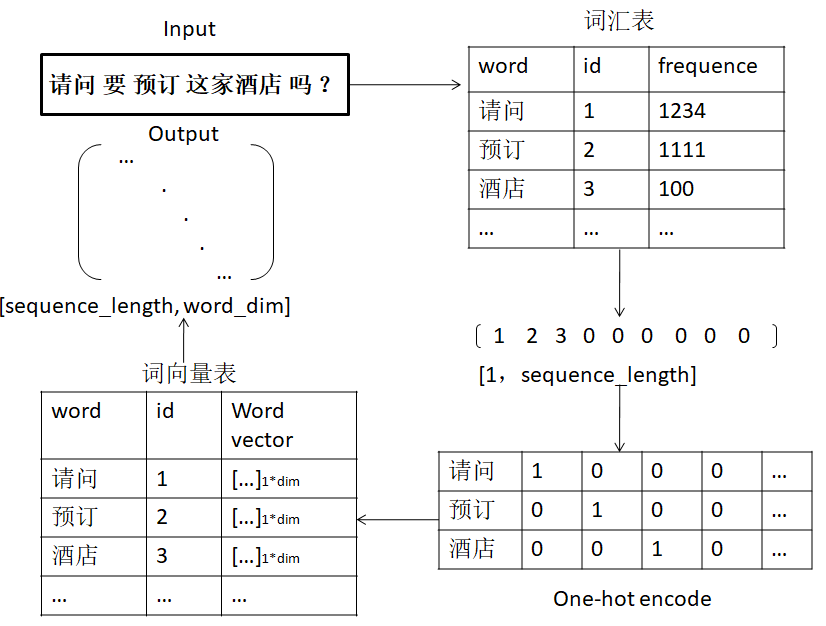
模型图：



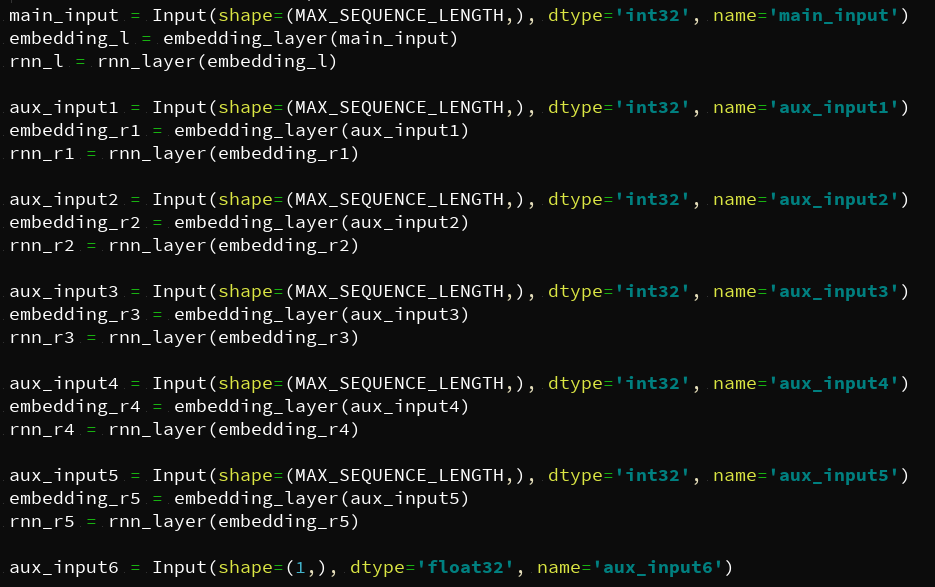
整个模型流程图：



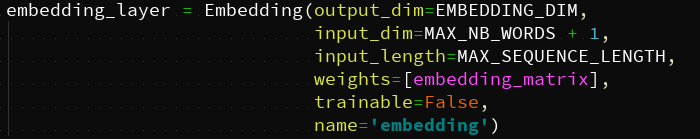
编码模块：

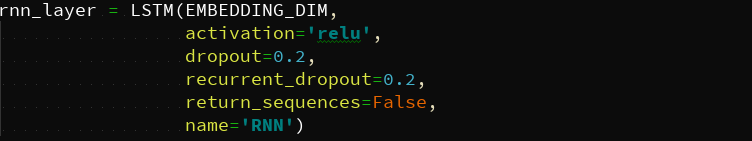


代码实现：

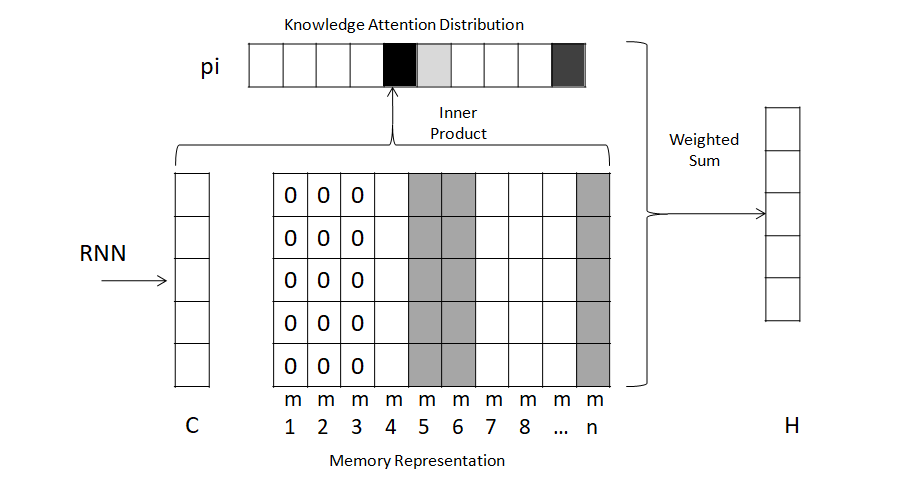


输入的文本处理为（NONE,MAX\_SEQUENCE\_LENGTH）的2D张量，当前轮次和历史轮次均进行embedding\_layer和rnn\_layer的处理。

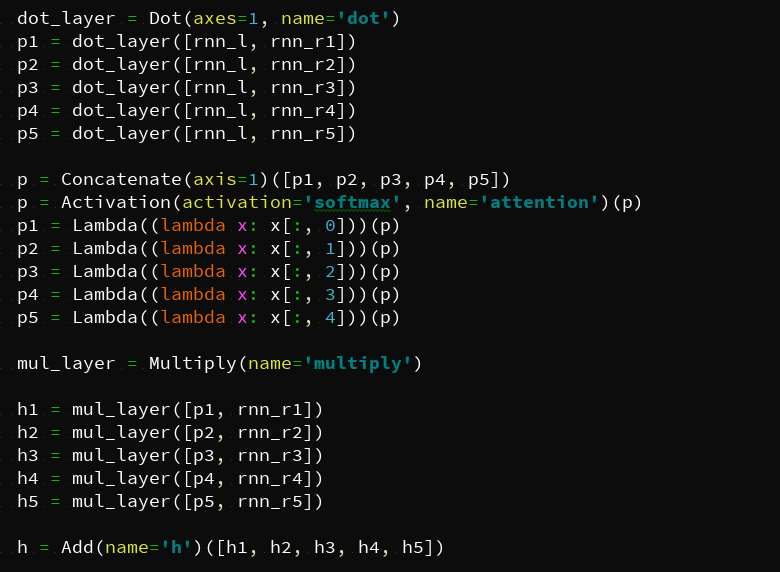




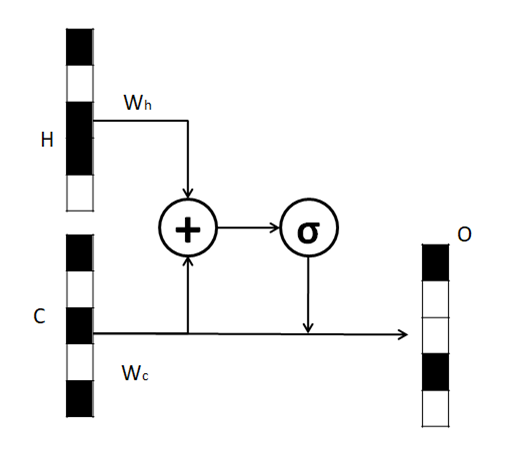
记忆模块：



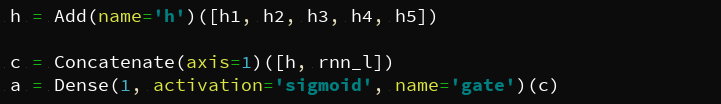
代码实现：



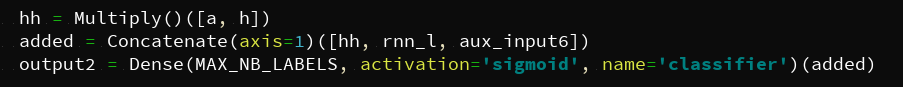
控制门：



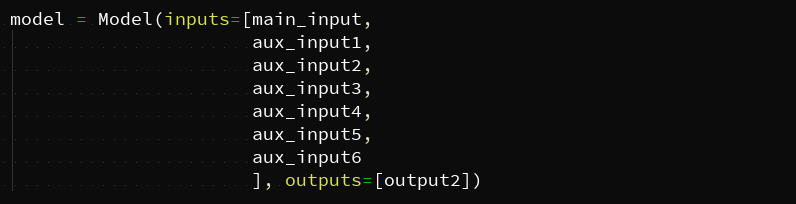
代码实现：



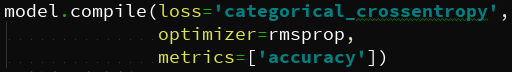
分类器模块：



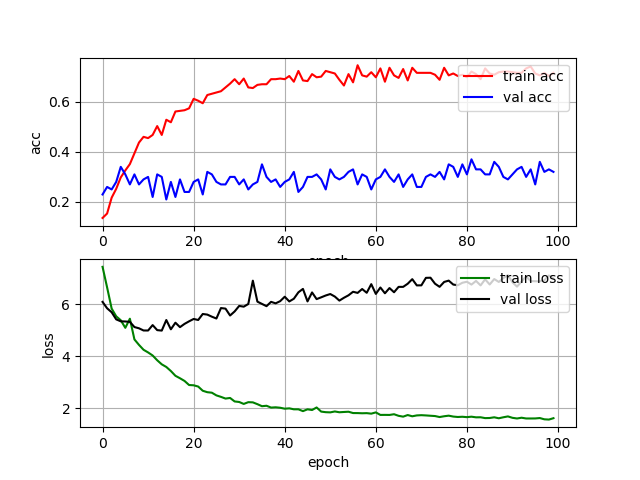
这里的激活函数为sigmoid函数，这个是为了配合后面的目标函数。



上周模型训练采用的目标函数，优化函数，性能评估是配置如下：



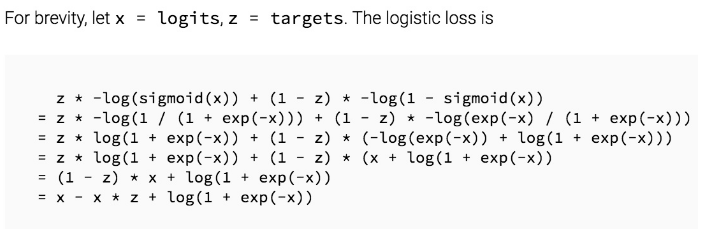
但是效果较差如下：



后来分析了源代码发现，TensorFlow针对分类问题，实现了四个交叉熵函数，分别是

* tf.nn.sigmoid\_cross\_entropy\_with\_logits
* tf.nn.softmax\_cross\_entropy\_with\_logits
* tf.nn.sparse\_softmax\_cross\_entropy\_with\_logits
* tf.nn.weighted\_cross\_entropy\_with\_logits

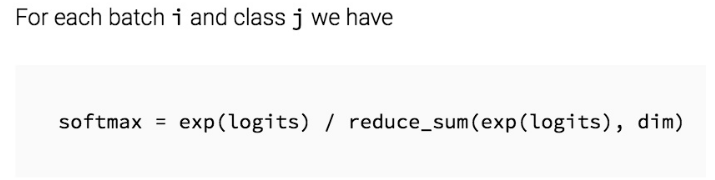
sigmoid\_cross\_entropy\_with\_logits的公式分析



标准的Cross Entropy算法实现，对W \* X得到的值进行sigmoid激活，保证取值在0到1之间，然后放在交叉熵的函数中计算Loss。这个函数的输入是logits和targets，logits就是神经网络模型中的 W \* X矩阵，注意不需要经过sigmoid，而targets的shape和logits相同，就是正确的label值，分类之间是独立的、不要求是互斥。

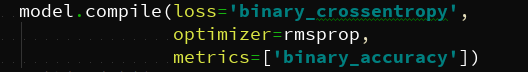
softmax\_cross\_entropy\_with\_logits的公式分析

Softmax本身的算法很简单，就是把所有值用e的n次方计算出来，求和后算每个值占的比率，保证总和为1，一般我们可以认为Softmax出来的就是confidence也就是概率，算法实现如下。

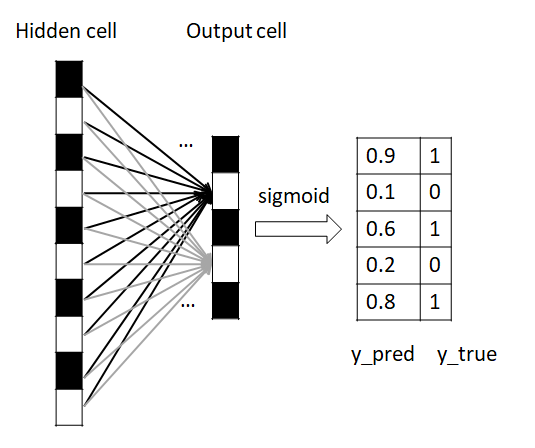


函数实现会在内部更高效得使用softmax，对于任意的输入经过softmax都会变成和为1的概率预测值，这个值就可以代入变形的Cross Entroy算法- y \* ln(a) - (1 - y) \* ln(1 - a)算法中，得到有意义的Loss值了。如果是多标签问题，经过softmax就不会得到多个和为1的概率，而且label有多个1也无法计算交叉熵，因此这个函数只适合单目标的二分类或者多分类问题。

所以最后经过确认后，目标函数，优化函数，性能评估是配置如下：



我的理解是，输出层的每一个节点都类似于二分类分类器，只需要判断对应标签的预测值是靠近0还是靠近1。每个标签分别对应一个节点，相当于为每个标签训练一个二分类分类器。



y\_true：真实的数据标签，Theano/TensorFlow张量

y\_pred：预测值，与y\_true相同shape的Theano/TensorFlow张量

由于激活函数选择的是sigmoid函数，所以阈值设置为0.5，大于0.5的预测为1，小于0.5的预测为0。

准确率 = TP / (TP + FP + 0.1)

召回率 = TP / (TP + FN + 0.1)

TP: 将正类预测为正类数

FN: 将正类预测为负类数

FP: 将负类预测为正类数

TN: 将负类预测为负类数

训练数据结果：

|  |  |
| --- | --- |
| 准确率 | 0.913 |
| 召回率 | 0.797 |

具体结果如下：

验证数据结果：

|  |  |
| --- | --- |
| 准确率 | 0.713 |
| 召回率 | 0.539 |

具体结果如下：

如上，标签共有16个，标签的分布如下：

## Attention值的分析：

目前实验中，记忆模型的空间定为5轮，查看其attention发现，在预测结果正确的情况下，通过softmax函数计算之后，历史轮次和当前轮次如果意图相同，则attention较靠近1。历史轮次为空时，attention均为0.2。

## 控制门

目前的控制门，感觉还是存在一些问题，gate值被限制在（0,1）之间，历史意图与当前意图无关时，gate值靠近0，历史意图和当前意图相关时，gate值靠近1，而查看gate值发现90%的gate值都>0.9。