阿里云比赛-第五周-TextCNN完善

笔记本:	日常		
创建时间:	2019/11/4 12:37	**更新时间:**	2019/11/17 17:13
作者:	296645429@qq.com		
URL:	https://mp.weixin.qq.com/s/AsBS9WRbf6rRlvTHPPo06w		

1.全局池化:

“global pooling”就是pooling的 滑窗size 和整张feature map的size一样大。这样，每个W×H×CW×H×C 的feature map输入就会被转化为 1×1×C1×1×C 输出。因此，其实也等同于每个位置权重都为 1/(W×H)1/(W×H) 的FC层操作。
全局池化可以减少网络的参数，代替全连接层。
TextCNN中时序最大池化可认为是一维全局最大池化，可使模型不受人为添加0的影响。
Embedding:
Embedding层就是为了训练一个词嵌入矩阵出来，然后可以获得任意的一个词的词向量。

设：
有一个输入句子样本是‘I very happy’,词典是[0:pad_word, 1: I, 2:very, 3:happy, 4:so, 5:sad]

$$\text{词嵌入矩阵W}=\begin{bmatrix}0.12 & 0.2\\ 0.30 & 0.15\\ 1.0 & 0.69\\ 20.1 & 1.45\\ 1.29 & 2.01\\ 3.45 & 3.45\end{bmatrix}=\begin{bmatrix}\text{pad\_}word\\ I\\ \text{very}\\ happy\\ so\\ sad\end{bmatrix}$$

则：
input X=[1, 2, 3]，然后对X进行one-hot

$$\text{X\_one\_hot}=\begin{bmatrix}0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0\end{bmatrix}$$,那么对输入X的词向量表示应该是X_one_hot· W

$$\text{X的词向量表示}=\begin{bmatrix}0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0\end{bmatrix}*\begin{bmatrix}0.12 & 0.2\\ 0.30 & 0.15\\ 1.0 & 0.69\\ 20.1 & 1.45\\ 1.29 & 2.01\\ 3.45 & 3.45\end{bmatrix}=\begin{bmatrix}0.30 & 0.15\\ 1.0 & 0.69\\ 20.1 & 1.45\end{bmatrix}=\begin{bmatrix}I\\ very\\ happy\end{bmatrix}$$

也就是说对于像一个句子样本X=[1,2,3] (1,2,3表示单词在词典中的索引)这样的输入可以先对它one-hot然后乘上词嵌入矩阵就可得到这个句子的词嵌入向量表示。要想得到好的词向量，我们需要训练的就是这个矩阵W(shape=(input_dim,output_dim))。Embedding层的作用就是训练这个矩阵W并进行词的嵌入为每一个词分配与它对应的词向量。这个词嵌入矩阵W可以先随机初始化，然后根据下游任务训练获得，也可以使用预训练的词嵌入矩阵来初始化它(keras中用weights来为layer初始化任意权重)，然后再训练，也可以直接用预训练的词嵌入矩阵来初始化它并冻结它，不让它变化，不让它可训练。(keras中用trainable=False)

2.文本在计算机中的表示方法总结

常用的文本表示方式分为：

1. 离散式表示（*Discrete Representation*）；

one-hot：关联性不大，浪费资源

词袋模型：面向文本编码，没考虑词位置，统计词频方式粗暴不合理

TF-IDF：通过IDF(语料库总文本数/特有词文本数)过滤常用不重要的词，没考虑词位置

2.分布式表示（*Distributed Representation*）；

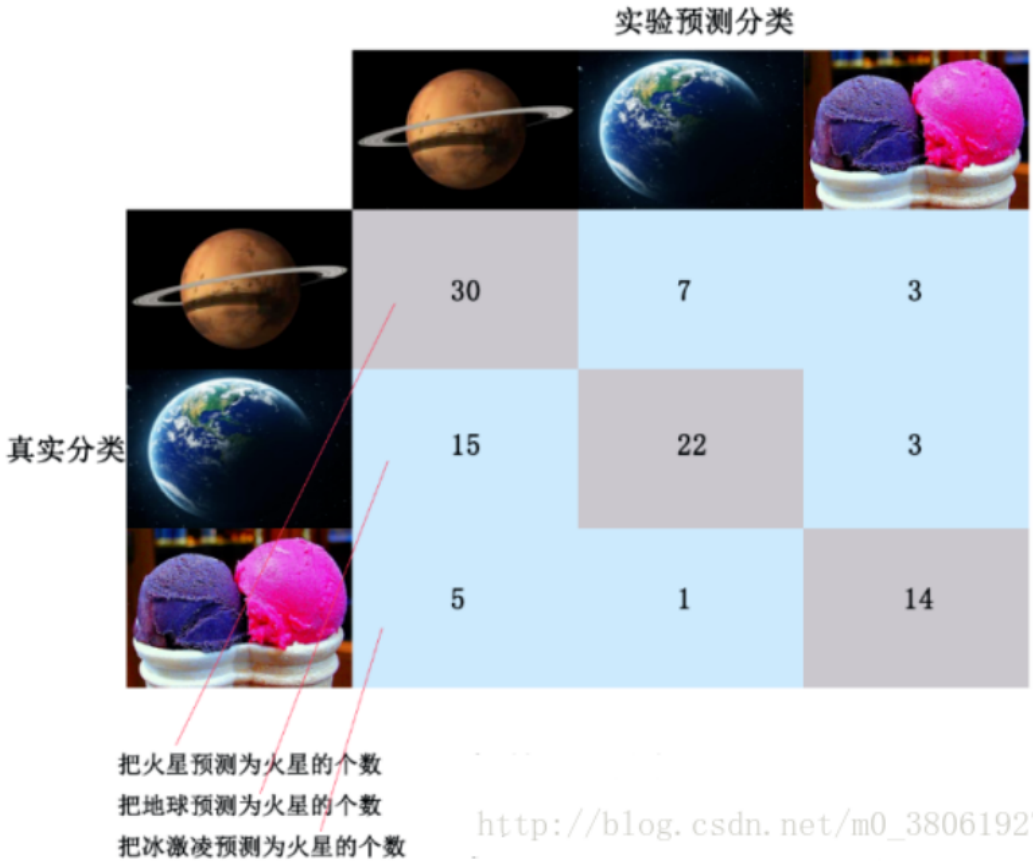
n-gram：一个词的出现仅与它前面出现的n个词有关，一般n=1或2
共现矩阵（*Co-Occurrence Matrix*）：计算量大
Word2Vec：节省资源，但无法解决多义词
GloVe：
ELMO：

3.xgboost：

Boosting 分类器属于集成学习模型，它基本思想是把成百上千个分类准确率较低的树模型组合起来，成为一个准确率很高的模型。
对于如何在每一步生成合理的树，就要提 Gradient Boosting Machine。
xgboost 的全称是 eXtreme Gradient Boosting。正如其名，它是 Gradient Boosting Machine 的一个 c++ 实现，作者为正在华盛顿大学研究机器学习的大牛陈天奇。xgboost 最大的特点在于，它能够自动利用 CPU 的多线程进行并行，同时在算法上加以改进提高了精度。
XGBoost提供了并行树增强（也称为GBDT，GBM），可快速准确地解决许多数据科学问题。

4.混淆矩阵：

混淆矩阵是机器学习中总结分类模型预测结果的情形分析表，以矩阵形式将数据集中的记录按照真实的类别与分类模型作出的分类判断两个标准进行汇总。
这个名字来源于它可以非常容易的表明**多个类别是否有混淆**（也就是一个**class被预测成另一个class**）



官方文档中给出的用法是

sklearn.metrics.confusion_matrix(y_true, y_pred, labels=None, sample_weight=None)

下面是官方文档上的一个例子

```
y_true = ["cat", "ant", "cat", "cat", "ant", "bird"]
y_pred = ["ant", "ant", "cat", "cat", "ant", "cat"]
confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
```

运行结果

```
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

下面是官方文档上的一个例子

```
y_true = ["cat", "ant", "cat", "cat", "ant", "bird"]
y_pred = ["ant", "ant", "cat", "cat", "ant", "cat"]
confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
```

运行结果

```
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

5.实践

logloss	valid_acc	validloss	备注	文件名
1.013332			loss：2到3	my_tpuresult4.csv
0.555824		loss:0.43	网络结构复杂，最重要是新添加了全局池化层，只跑了一轮，20000api集长	5
0.583796		loss:0.44	增加api序列数据集为60000	6