深度之眼
deepshare.net

常用评估函数与自定义评估函数

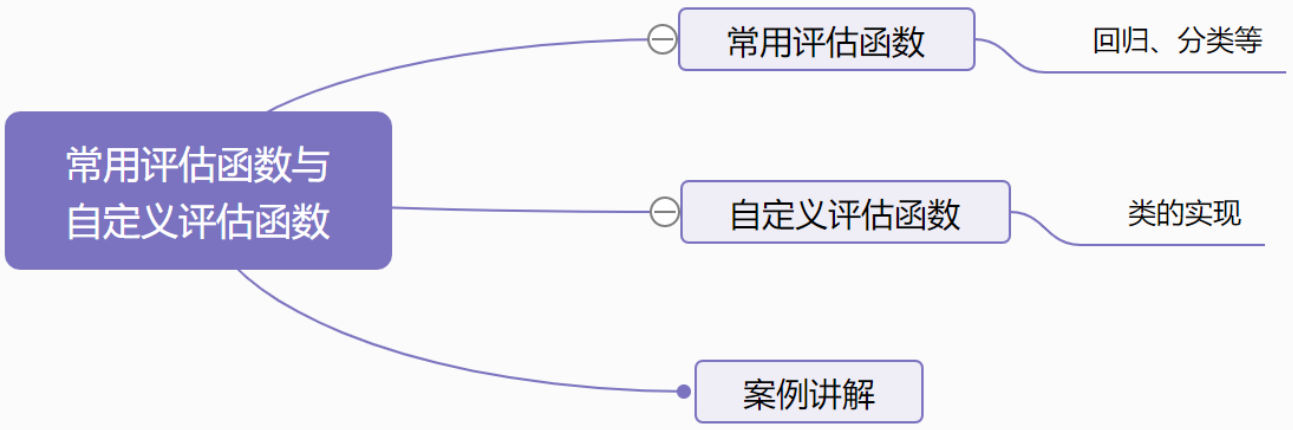
导师：GAUSS

深度之眼
deepshare.net

目录

知识树

Knowledge tree



深度之眼
deepshare.net

常用评估函数

常用评估函数

深度之眼
deepshare.net

tf.keras.metrics（tf.metrics的接口均移到这里）

参考网站：https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/keras/metrics

tf.keras.metrics.BinaryAccuracy

tf.keras.metrics.binary_accuracy

有什么区别？

前者是类的实现形式，后者是函数的实现形式。

常用评估函数

深度之眼
deepshare.net

BinaryAccuracy 和 binary_accuracy实现：

```python
@keras_export('keras.metrics.binary_accuracy')
def binary_accuracy(y_true, y_pred, threshold=0.5):
  """Calculates how often predictions matches binary labels.

  Args:
    y_true: Ground truth values. shape = `[batch_size, d0, .. dN]`.
    y_pred: The predicted values. shape = `[batch_size, d0, .. dN]`.
    threshold: (Optional) Float representing the threshold for deciding whether
      prediction values are 1 or 0.

  Returns:
    Binary accuracy values. shape = `[batch_size, d0, .. dN-1]`
  """
  threshold = math_ops.cast(threshold, y_pred.dtype)
  y_pred = math_ops.cast(y_pred > threshold, y_pred.dtype)
  return K.mean(math_ops.equal(y_true, y_pred), axis=-1)
```

```python
@keras_export('keras.metrics.BinaryAccuracy')
class BinaryAccuracy(MeanMetricWrapper):

    def __init__(self, name='binary_accuracy', dtype=None, threshold=0.5):
        """Creates a `BinaryAccuracy` instance.
        Args:
            name: (Optional) string name of the metric instance.
            dtype: (Optional) data type of the metric result.
            threshold: (Optional) Float representing the threshold for deciding
            whether prediction values are 1 or 0.
        """
        super(BinaryAccuracy, self).__init__(
                binary_accuracy, name, dtype=dtype, threshold=threshold)
```

常用评估函数

- tf.keras.metrics.MeanAbsoluteError （平方差误差，用于回归，可以简写为MSE，函数形式为mse）
- tf.keras.metrics.MeanAbsoluteError (绝对值误差，用于回归，可以简写为MAE，函数形式为mae)
- tf.keras.metrics.MeanAbsolutePercentageError (平均百分比误差，用于回归，可以简写为MAPE，函数形式为mape)
- tf.keras.metrics.RootMeanSquaredError (均方根误差，用于回归)

常用评估函数

深度之眼
deepshare.net

- tf.keras.metrics.Accuracy (准确率，用于分类，可以用字符串"Accuracy"表示，Accuracy=(TP+TN)/(TP+TN+FP+FN)，要求y_true和y_pred都为类别序号编码)
- tf.keras.metrics.AUC (ROC曲线(TPR vs FPR)下的面积，用于二分类，直观解释为随机抽取一个正样本和一个负样本，正样本的预测值大于负样本的概率)
- tf.keras.metrics.Precision (精确率，用于二分类，Precision = TP/(TP+FP))
- tf.keras.metrics.Recall (召回率，用于二分类，Recall = TP/(TP+FN))
- tf.keras.metrics.TopKCategoricalAccuracy(多分类TopK准确率，要求y_true(label)为onehot编码形式)

常用评估函数

深度之眼 deepshare.net

- tf.keras.metrics.CategoricalAccuracy（分类准确率，与Accuracy含义相同，要求y_true(label)为onehot编码形式）
- tf.keras.metrics. SparseCategoricalAccuracy（稀疏分类准确率，与Accuracy含义相同，要求y_true(label)为序号编码形式）

有什么区别呢?

```
[1]: import tensorflow as tf

[2]: y_true = tf.constant([0,1,2,1,4,5,3])
     y_pred = tf.random.uniform(shape=(7,6))
     acc = tf.keras.metrics.SparseCategoricalAccuracy()
     acc.update_state(y_true,y_pred)
     tf.print(acc.result().numpy())
     acc.reset_states()

     0.42857143
```

```
[1]: import tensorflow as tf

[2]: y_true = tf.constant([0,1,2,1,4,5,3])
     y_pred = tf.random.uniform(shape=(7,6))
     y_true = tf.one_hot(y_true,depth=6,dtype=tf.int32)

[3]: acc = tf.keras.metrics.CategoricalAccuracy()
     acc.update_state(y_true,y_pred)
     tf.print(acc.result().numpy())
     acc.reset_states()

     0.0
```

常用评估函数

这里仅列举部分！！！

更多参考：

https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/keras/metrics

www.deepshare.net

```
m = tf.keras.metrics.Accuracy()
m.update_state([1, 2, 3, 4], [0, 2, 3, 4])
# 还可以这样简写
# m([1, 2, 3, 4], [0, 2, 3, 4])
print('Final result: ', m.result().numpy())
# Final result: 0.75
m.update_state([1, 2, 3, 4], [0, 2, 3, 1])
print('Final result: ', m.result().numpy())
# Final result: 0.65
m.reset_states()
# 重置
```

深度之眼
deepshare.net

自定义评估函数

自定义评估函数

深度之眼
deepshare.net

tf.keras.metrics.Metric

参考：https://www.tensorflow.org/versions/r2.0/api_docs/python/tf/keras/metrics/Metric

两种实现形式：基于类的实现和基于函数的实现

大部分使用基于类的实现

自定义评估函数

深度之眼
deepshare.net

自定义损失函数：

自定义评估指标需要继承 **tf.keras.metrics.Metric** 类，并重写 **__init__** 、**update_state** 和 result 三个方法。

- __init__():所有状态变量都应通过以下方法在此方法中创建self.add_weight()
- update_state(): 对状态变量进行所有更新
- result(): 根据状态变量计算并返回指标值。

在notebook中看如何自定义评估函数

自定义评估函数

深度之眼
deepshare.net

自定义实现tf.keras.metrics.SparseCategoricalAccuracy()

```
class SparseCategoricalAccuracy_(tf.keras.metrics.Metric):
    def __init__(self, name='SparseCategoricalAccuracy', **kwargs):
        super(SparseCategoricalAccuracy_, self).__init__(name=name, **kwargs)
        self.total = self.add_weight(name='total', dtype=tf.int32, initializer=tf.zeros_initializer())
        self.count = self.add_weight(name='count', dtype=tf.int32, initializer=tf.zeros_initializer())


    def update_state(self, y_true, y_pred,sample_weight=None):
        values = tf.cast(tf.equal(y_true, tf.argmax(y_pred, axis=-1, output_type=tf.int32)), tf.int32)
        self.total.assign_add(tf.shape(y_true)[0])
        self.count.assign_add(tf.reduce_sum(values))

    def result(self):
        return self.count / self.total

    def reset_states(self):
        # The state of the metric will be reset at the start of each epoch.
        self.total.assign(0)
        self.count.assign(0)
```

自定义评估函数

深度之眼
deepshare.net

计算多分类正确标签的个数

```
class CatgoricalTruePositives(tf.keras.metrics.Metric):
    def __init__(self, name='categorical_true_positives', **kwargs):
        super(CatgoricalTruePositives, self).__init__(name=name, **kwargs)
        self.true_positives = self.add_weight(name='tp', initializer='zeros')

    def update_state(self, y_true, y_pred, sample_weight=None):
        y_pred = tf.argmax(y_pred,axis=-1)
        values = tf.equal(tf.cast(y_true, 'int32'), tf.cast(y_pred, 'int32'))
        values = tf.cast(values, 'float32')
        if sample_weight is not None:
            sample_weight = tf.cast(sample_weight, 'float32')
            values = tf.multiply(values, sample_weight)
        self.true_positives.assign_add(tf.reduce_sum(values))

    def result(self):
        return self.true_positives

    def reset_states(self):
        self.true_positives.assign(0.)
```

深度之眼
deepshare.net

案例讲解

自定义模型训练版本

深度之眼
deepshare.net

详见notebook

```
model = MyModel()

loss_object = tf.keras.losses.SparseCategoricalCrossentropy() #损失函数


optimizer = tf.keras.optimizers.Adam() #优化器

#评估函数
train_loss = tf.keras.metrics.Mean(name='train_loss') #loss

train_accuracy = tf.keras.metrics.SparseCategoricalAccuracy(name='train_accuracy') #准确率

train_tp = CatgoricalTruePositives(name="train_tp") #返回正确的个数

test_loss = tf.keras.metrics.Mean(name='test_loss')
test_accuracy = tf.keras.metrics.SparseCategoricalAccuracy(name='test_accuracy')
test_tp = CatgoricalTruePositives(name='test_tp')

@tf.function
def train_step(images, labels):
    with tf.GradientTape() as tape:
        predictions = model(images)
        loss = loss_object(labels, predictions)
    gradients = tape.gradient(loss, model.trainable_variables)
    optimizer.apply_gradients(zip(gradients, model.trainable_variables))

    #评估函数的结果
    train_loss(loss)
    train_accuracy(labels, predictions)
    train_tp(labels, predictions)
```

Keras模型训练版本

深度之眼
deepshare.net

类的实现形式

```
#返回的是一个正确的个数
class CatgoricalTruePositives(tf.keras.metrics.Metric):
    def __init__(self, name='categorical_true_positives', **kwargs):
        super(CatgoricalTruePositives, self).__init__(name=name, **kwargs)
        self.true_positives = self.add_weight(name='tp', initializer='zeros')

    def update_state(self, y_true, y_pred, sample_weight=None):
        y_pred = tf.argmax(y_pred,axis=-1)
        y_true = tf.argmax(y_true,axis=-1)
        values = tf.equal(tf.cast(y_true, 'int32'), tf.cast(y_pred, 'int32'))
        values = tf.cast(values, 'float32')
        if sample_weight is not None:
            sample_weight = tf.cast(sample_weight, 'float32')
            values = tf.multiply(values, sample_weight)
        self.true_positives.assign_add(tf.reduce_sum(values))

    def result(self):
        return self.true_positives

    def reset_states(self):
        self.true_positives.assign(0.)
```

本节小结

Summary

常用评估函数与自定义评估函数	常用评估函数	常用的评估函数（场景）
	自定义评估函数	类的实现形式
	案例讲解	

我 说

GAUSS老师个人公众号，主要分享NLP、推荐、比赛实战相关知识！

深度之眼
deepshare.net

联系我们:

电话：18001992849

邮箱：service@deepshare.net

Q Q： 2677693114

公众号

客服微信