制作tfrecords数据集：TensorFlow可以处理的数据集格式之一是tfrecords，主要用到Python和TensorFlow框架中的函数或库。这里包含制作tfrecords数据集、验证是否制作成功以及将数据读入深度神经网络三部分的程序，编程实现比较多样，这里仅是我编程实现的过程。

**一，制作tfrecords数据集**

|  |
| --- |
| import tensorflow as tf #导入TensorFlow，并命名为tf |
| import os  #生成tfrecords的函数 |
|  |
|  |
| def generate\_tfrecords(input\_filename, output\_filename): |
| print("Start to convert {} to {}".format(input\_filename, output\_filename)) |
| writer = tf.python\_io.TFRecordWriter(output\_filename) #写入要输出文件 |
|  |
| for line in open(input\_filename, "r"): #输入文件转成列表 |
| data = line.split(",") |
| label = float(data[2]) #标签数（或特征数）是0和1 |
| features = [float(i) for i in data[:1]] |
|  |
| example = tf.train.Example(features=tf.train.Features(feature= |
| "label": |
| tf.train.Feature(float\_list=tf.train.FloatList(value=[label])), |
| "features": |
| tf.train.Feature(float\_list=tf.train.FloatList(value=features)))) |
|  |
| writer.write(example.SerializeToString()) |
|  |
| writer.close() |
| print("Successfully convert {} to {}".format(input\_filename, |
| output\_filename))  #主函数调用generate\_tfrecords()，生成tfrecords数据集 |
|  |
|  |
| def main(): |
| current\_path = os.getcwd() |
| for filename in os.listdir(current\_path): |
| if filename.startswith("") and filename.endswith(".csv"): |
| generate\_tfrecords(filename, filename + ".tfrecords") |
|  |
|  |
| if \_\_name\_\_ == "\_\_main\_\_": |
| main() |

**二，验证tfrecords数据集是否制作成功**

|  |
| --- |
| Import tensorflow as tf  Import os |
| def print\_tfrecords(input\_filename): |
| max\_print\_number = 100 |
| current\_print\_number = 0 |
|  |
| for serialized\_example in tf.python\_io.tf\_record\_iterator(input\_  filename): |
| # 获得序列样本 |
| example = tf.train.Example() |
| example.ParseFromString(serialized\_example) |
| label = example.features.feature["label"].float\_list.value |
| features = example.features.feature["features"].float\_list.value |
| print("Number: {}, label: {}, features: {}".format(current\_print\_number, label, features)) |
|  |
|  |
| # 获取所有的样本后，退出。 |
| current\_print\_number += 1 |
| if current\_print\_number > max\_print\_number: |
| exit() |
|  |
|  |
| #运行函数，输出input\_filename  def main(): |
| current\_path = os.getcwd() |
| tfrecords\_file\_name = "filename.csv.tfrecords" |
| input\_filename = os.path.join(current\_path, tfrecords\_file\_name) |
| print\_tfrecords(input\_filename) |
|  |
|  |
| if \_\_name\_\_ == "\_\_main\_\_": |
| main() |

**三，读入神经网络**

TensorFlow读取tfrecords文件时，以队列形式分批读入，CNN或RNN也会分批处理数据。下面是以文件队列形式读入神经网络。

|  |
| --- |
| import tensorflow as tf |
| import math |
| import os |
| import numpy as np  #定义读取tfrecords数据集的函数 |
| def read\_and\_decode(filename\_queue): |
| reader = tf.TFRecordReader() |
| \_, serialized\_example = reader.read(filename\_queue)  #解析数据和标签 |
| features = tf.parse\_single\_example( |
| serialized\_example, |
| features={ |
| "label": tf.FixedLenFeature([], tf.float32), |
| "features": tf.FixedLenFeature([FEATURE\_SIZE], tf.float32), |
| }) |
|  |
| label = features["label"] |
| features = features["features"] |
|  |
| return label, features |

#定义主函数，调用上述函数读取tfrecords文件

def main(\_):

#tfrecords文件，分为train和test两部分

|  |
| --- |
| filename\_queue = tf.train.string\_input\_producer( |
| tf.train.match\_filenames\_once("train.tfrecords")) |

|  |
| --- |
|  |
| label, features = read\_and\_decode(filename\_queue)  #随机散列batch |
| batch\_labels, batch\_features = tf.train.shuffle\_batch( |
| [label, features],  batch\_size=batch\_size) |

|  |
| --- |
| filename\_queue = tf.test.string\_input\_producer( |
| tf.test.match\_filenames\_once("test.tfrecords")) |

|  |
| --- |
|  |
| test\_label, test\_features = read\_and\_decode(filename\_queue)  #随机散列batch,batch\_size是TF每批处理的数据量 |
| test\_batch\_labels, test\_batch\_features = tf.test.shuffle\_batch( |
| [test\_label, test\_features], batch\_size=batch\_size) |

|  |
| --- |
| #运行主函数  if \_\_name\_\_ == "\_\_main\_\_": |
| main() |