Tensorflow中使用TFRecords高效读取数据--结合NLP数据实践

原创 2017年06月23日 20:15:20 7396 4 2

之前一篇博客 (http://blog.csdn.net/liuchonge/article/details/72848224)在进行论文仿真的时候用到了TFRecords进行数据的读取操作，但是因为当时比较忙，所以没有进行深入学习。这两天看了一下，决定写篇博客专门结合该代码记录一下TFRecords的相关操作。

首先说一下为什么要使用TFRecords来进行文件的读写，在TF中数据的传入方式主要包含以下几种：

1. **供给数据(Feeding)**： 在TensorFlow程序运行的每一步， 让Python代码来供给数据。
2. **从文件读取数据**： 在TensorFlow图的起始， 让一个输入管线从文件中读取数据。
3. **预加载数据**： 在TensorFlow图中定义常量或变量来保存所有数据(仅适用于数据量比较小的情况)。

之前都是使用1和3进行数据的操作，但是当我们遇到数据集比较大的情况时，这两种方法会及其占用内存，效率很差。那么为甚么使用TFRecords会比较快呢？因为其使用二进制存储文件，也就是将数据存储在一个内存块中，相比其它文件格式要快很多，特别是如果你使用hdd而不是ssd，因为它涉及移动磁盘阅读器头并且需要相当长的时间。总体而言，通过使用二进制文件，您可以更轻松地分发数据，使数据更好地对齐，以实现高效的读取。接下来我们看一下具体的操作。

这里可以参见官网给的建议：

```
1 Another approach is to convert whatever data you have into a supported format. This approach makes it easi
2
3 To read a file of TFRecords, use tf.TFRecordReader with the tf.parse_single_example decoder. The parse_single
```

个人感觉可以分成两部分，一是使用tf.train.Example协议流将文件保存成TFRecords格式的.tfrecords文件，这里主要涉及到使用 tf.python_io.TFRecordWriter("train.tfrecords") 和 tf.train.Example 以及 tf.train.Features 三个函数，第一个是生成需要对应格式的文件，后面两个函数主要是将我们要传入的数据按照一定的格式进行规范化。这里还要提到一点就是使用TFRecords可以避免多个文件的使用，比如说我们一般会将一次要传入的数据的不同部分分别存放在不同文件夹中，question一个，answer一个，query一个等等，但是使用TFRecords之后，我们可以将一批数据同时保存在一个文件之中，这样方便我们在后续程序中的使用。

另一部分就是在训练模型时将我们生成的.tfrecords文件读入并传到模型中进行使用。这部分主要涉及到使用 tf.TFRecordReader("train.tfrecords") 和 tf.parse_single_example 两个函数。第一个函数是将我们的二进制文件读入，第二个则是进行解析然后得到我们想要的数据。

接下来我们结合代码进行理解：

生成TFRecords文件

这里关于要使用的数据集的介绍可以参考我的上一篇博客 (http://blog.csdn.net/liuchonge/article/details/72848224)，主要是QA任务的数据集。代码如下所示：

```
def tokenize(index, word):
#index是每个单词对应词袋子之中的索引值，word是所有出现的单词
  directories = ['cnn/questions/training/', 'cnn/questions/validation/', 'cnn/questions/test/']
  for directory in directories:
  #分别读取训练测试验证集的数据
    out_name = directory.split('/')[-2] + '.tfrecords'
    #生成对应.tfrecords文件
    writer = tf.python_io.TFRecordWriter(out_name)
    #每个文件夹下面都有若干文件，每个文件代表一个QA队，也就是一条训练数据
    files = map(lambda file_name: directory + file_name, os.listdir(directory))
    for file_name in files:
      with open(file_name, 'r') as f:
        lines = f.readlines()
        #对每条数据分别获得文档，问题，答案三个值，并将相应单词转化为索引
        document = [index[token] for token in lines[2].split()]
        query = [index[token] for token in lines[4].split()]
        answer = [index[token] for token in lines[6].split()]
        #调用Example和Features函数将数据格式化保存起来。注意Features传入的参数应该是一个字典，方便后续
        example = tf.train.Example(
          features = tf.train.Features(
            feature = {
              'document': tf.train.Feature(
                int64_list=tf.train.Int64List(value=document)),
              'query': tf.train.Feature(
                int64_list=tf.train.Int64List(value=query)),
              'answer': tf.train.Feature(
                int64_list=tf.train.Int64List(value=answer))
              }))
    #写数据
      serialized = example.SerializeToString()
      writer.write(serialized)
```

读取.tfrecords文件

因为在读取数据之后我们可能还会进行一些额外的操作，使我们的数据格式满足模型输入，所以这里会引入一些额外的函数来实现我们的目的。这里介绍几个个人感觉较重要常用的函数。不过还是推荐到官网API去查，或者有某种需求的时候到Stack Overflow上面搜一搜，一般都能找到满足自己需求的函数。

1， string_input_producer(
string_tensor,
num_epochs=None,
shuffle=True,
seed=None,
capacity=32,
shared_name=None,
name=None,
cancel_op=None
) 其输出是一个输入管道的队列，这里需要注意的参数是num_epochs和shuffle。对于每个epoch其会将所有的文件添加到文件队列当中，如果设置shuffle，则会对文件顺序进行打乱。其对文件进行均匀采样，而不会导致上下采样。

2， shuffle_batch(
tensors,
batch_size,
capacity,
min_after_dequeue,
num_threads=1,
seed=None,
enqueue_many=False,
shapes=None,
allow_smaller_final_batch=False,
shared_name=None,
name=None
) 产生随机打乱之后的batch数据

3， sparse_ops.serialize_sparse(sp_input, name=None) : 返回一个字符串的3-vector（1-D的tensor），分别表示索引、值、shape

4， deserialize_many_sparse(serialized_sparse, dtype, rank=None, name=None) : 将多个稀疏的serialized_sparse合并成一个

```
def read_records(index=0):
#生成读取数据的队列，要指定epoches
  train_queue = tf.train.string_input_producer(['training.tfrecords'], num_epochs=FLAGS.epochs)
  validation_queue = tf.train.string_input_producer(['validation.tfrecords'], num_epochs=FLAGS.epochs)
  test_queue = tf.train.string_input_producer(['test.tfrecords'], num_epochs=FLAGS.epochs)

  queue = tf.QueueBase.from_list(index, [train_queue, validation_queue, test_queue])
  #定义一个recordreader对象，用于数据的读取
  reader = tf.TFRecordReader()
  #从之前的队列中读取数据到serialized_example
  _, serialized_example = reader.read(queue)
  #调用parse_single_example函数解析数据
  features = tf.parse_single_example(
      serialized_example,
      features={
        'document': tf.VarLenFeature(tf.int64),
        'query': tf.VarLenFeature(tf.int64),
        'answer': tf.FixedLenFeature([], tf.int64)
      })

#返回索引、值、shape的三元组信息
  document = sparse_ops.serialize_sparse(features['document'])
  query = sparse_ops.serialize_sparse(features['query'])
  answer = features['answer']

#生成batch切分数据
  document_batch_serialized, query_batch_serialized, answer_batch = tf.train.shuffle_batch(
      [document, query, answer], batch_size=FLAGS.batch_size,
      capacity=2000,
      min_after_dequeue=1000)

  sparse_document_batch = sparse_ops.deserialize_many_sparse(document_batch_serialized, dtype=tf.int64)
  sparse_query_batch = sparse_ops.deserialize_many_sparse(query_batch_serialized, dtype=tf.int64)

  document_batch = tf.sparse_tensor_to_dense(sparse_document_batch)
  document_weights = tf.sparse_to_dense(sparse_document_batch.indices, sparse_document_batch.shape, 1)

```

```
38    query_batch = tf.sparse_tensor_to_dense(sparse_query_batch)
39    query_weights = tf.sparse_to_dense(sparse_query_batch.indices, sparse_query_batch.shape, 1)
40
41    return document_batch, document_weights, query_batch, query_weights, answer_batch
```

最后，我们要在模型开始训练之前，执行下面两行代码，

```
1  with tf.Session() as sess:
2    # Start populating the filename queue.
3    coord = tf.train.Coordinator()
4    threads = tf.train.start_queue_runners(coord=coord)
```

这是填充队列的指令，如果不执行程序会等在队列文件的读取处无法运行。至此，我们就可以使用TFRecords来读写文件了。最后总结一下，大概格式如下，这里并未指定某种读写函数，而是可以自定义的方式用的伪代码来说一下整个流程：

```
def read_my_file_format(filename_queue):
  reader = tf.SomeReader()
  key, record_string = reader.read(filename_queue)
  example, label = tf.some_decoder(record_string)
  processed_example = some_processing(example)
  return processed_example, label

def input_pipeline(filenames, batch_size, num_epochs=None):
  filename_queue = tf.train.string_input_producer(
      filenames, num_epochs=num_epochs, shuffle=True)
  example, label = read_my_file_format(filename_queue)
  # min_after_dequeue defines how big a buffer we will randomly sample
  #   from -- bigger means better shuffling but slower start up and more
  #   memory used.
  # capacity must be larger than min_after_dequeue and the amount larger
  #   determines the maximum we will prefetch.  Recommendation:
  #   min_after_dequeue + (num_threads + a small safety margin) * batch_size
  min_after_dequeue = 10000
  capacity = min_after_dequeue + 3 * batch_size
  example_batch, label_batch = tf.train.shuffle_batch(
      [example, label], batch_size=batch_size, capacity=capacity,
      min_after_dequeue=min_after_dequeue)
  return example_batch, label_batch
```


参考连接：

https://www.tensorflow.org/programmers_guide/reading_data
(https://www.tensorflow.org/programmers_guide/reading_data)

http://warmspringwinds.github.io/tensorflow/tf-slim/2016/12/21/tfrecords-guide/
(http://warmspringwinds.github.io/tensorflow/tf-slim/2016/12/21/tfrecords-guide/)

http://ycszen.github.io/2016/08/17/TensorFlow%E9%AB%98%E6%95%88%E8%AF%BB%E5%8F%96%E6%95%B0%E6%8D
(http://ycszen.github.io/2016/08/17/TensorFlow%E9%AB%98%E6%95%88%E8%AF%BB%E5%8F%96%E6%95%B0%E6%8

标签：tensorflow (http://so.csdn.net/so/search/s.do?q=tensorflow&t=blog) / TFRecords (http://so.csdn.net/so/search/s.do?q=TFRecords&t=blog) /

本文已收录于以下专栏：深度学习与NLP--论文笔记和TensorFlow实现 (http://blog.csdn.net/column/details/14944.html)

daigualu (/daigualu) 2017-06-30 07:55 3楼

(/daigualu)点赞一个！

回复

daigualu (/daigualu) 2017-06-30 07:55 2楼

(/daigualu)TFRecords是个什么东西？对象吗？

回复

查看 4 条热评

相关文章推荐

Tensorflow 中Tfrecords的使用心得 (/u014802590/article/details/68495238)

这篇博客主要讲了如何用Tensorflow中的标准数据读取方式的简单的实现对自己数据的读取操作。

u014802590 (http://blog.csdn.net/u014802590) 2017-03-30 20:10 517

TensorFlow学习记录-- 7 .TensorFlow高效读取数据之tfrecord详细解读 (/qq_16949707/article/details/53483493)

一 why tfrecord?对于数据量较小而言，可能一般选择直接将数据加载进内存，然后再分batch输入网络进行训练（tip:使用这种方法时，结合yield 使用更为简洁，大家自己尝试一下吧，我就不...

qq_16949707 (http://blog.csdn.net/qq_16949707) 2016-12-06 10:06 3686

精选：深入理解 Docker 内部原理及网络配置 (http://edu.csdn.net/huiyiCourse/detail/563?utm_source=blog10)

网络绝对是任何系统的核心，对于容器而言也是如此。Docker 作为目前最火的轻量级容器技术，有很多令人称道的功能，如 Docker 的镜像管理。然而，Docker的网络一直以来都比较薄弱，所以我们有必要深入了解Docker的网络知识，以满足更高的网络需求。

TensorFlow高效读取数据的方法 (/u012759136/article/details/52232266)

概述关于Tensorflow读取数据，官网给出了三种方法： 供给数据(Feeding)： 在TensorFlow程序运行的每一步， 让Python代码来供给数据。 从文件读取数据： 在TensorFl...

u012759136 (http://blog.csdn.net/u012759136) 2016-08-17 19:20 25120

Tensorflow建立与读取TFrecorder文件 (/freedom098/article/details/56011858)

Tensorflow建立与读取TFrecorder文件除了直接读取数据文件，比如csv和bin文件，tensorflow还可以建立一种自有格式的数据文件，称之为tfrecorder，这种文件储存类似于...


freedom098 (http://blog.csdn.net/freedom098) 2017-02-20 13:20 2818

TFRecords 文件的生成和读取 (/u012222949/article/details/72875281)

TensorFlow提供了TFRecords的格式来统一存储数据，理论上，TFRecords可以存储任何形式的数据。 TFRecords文件中的数据都是通过tf.train.Example ...


u012222949 (http://blog.csdn.net/u012222949) 2017-06-06 10:09 1368

Tensorflow中使用TFRecords高效读取数据--结合NLP数据实践 (/liuchonge/article/details/73649251)

之前一篇博客在进行论文仿真的时候用到了TFRecords进行数据的读取操作，但是因为当时比较忙，所以没有进行深入学习。这两天看了一下，决定写篇博客专门结合该代码记录一下TFRecords的相关操作。 ...


liuchonge (http://blog.csdn.net/liuchonge) 2017-06-23 20:15 7396

数据读取之TFRecords (/u013818406/article/details/70808566)

转载自http://blog.csdn.net/u012759136/article/details/52232266 对部分代码做了一些修改 import os import tensorflow...


u013818406 (http://blog.csdn.net/u013818406) 2017-04-26 16:48 587

TensorFlow高效读取数据——TFRecord (/data8866/article/details/65626750)

关于Tensorflow读取数据，官网给出了三种方法： 供给数据(Feeding)： 在TensorFlow程序运行的每一步， 让Python代码来供给数据。从文件读取数据： 在TensorFl...


DATA8866 (http://blog.csdn.net/DATA8866) 2017-03-24 09:51 739

TensorFlow高效读取数据的方法 (/appleml/article/details/53898554)

关于Tensorflow读取数据，官网给出了三种方法： 供给数据(Feeding)： 在TensorFlow程序运行的每一步， 让Python代码来供给数据。 从文件读取数据： 在TensorFlow...

u014221266 (http://blog.csdn.net/u014221266) 2016-12-27 19:55 1093

tensorflow读取SVHN数据集转为TFrecords格式 (/qikaihuting/article/details/71577658)

这里默认将python脚本文件和svhn数据集放在同一目录下，FLAGS.directory参数可以指定数据集的目录，由于svhn没有validation数据集，因此将train分割一部分出来作为va...

qikaihuting (http://blog.csdn.net/qikaihuting) 2017-05-10 20:02 373

机器学习: TensorFlow 的数据读取与TFRecords 格式 (/matrix_space/article/details/60962026)

最近学习tensorflow，发现其读取数据的方式看起来有些不同，所以又重新系统地看了一下文档，总得来说，tensorflow 有三种主流的数据读取方式: 1) 传送 (feeding): Pyth...

shinian1987 (http://blog.csdn.net/shinian1987) 2017-03-22 11:24 1096

Tensorflow读取数据2-tfrecord (/u010911921/article/details/70991194)

用自己的数据集创建Tensorflow的标准格式TFRecords

u010911921 (http://blog.csdn.net/u010911921) 2017-04-29 22:47 358

Tensorflow读取数据1 (/u010911921/article/details/70577697)

Tensorflow采用queue的方式读取CIFAR-10的二进制数据

u010911921 (http://blog.csdn.net/u010911921) 2017-04-24 11:29 639

TensorFlow数据读取方法 (/u010329292/article/details/68484485)

转自：http://honggang.io/2016/08/19/tensorflow-data-reading/ 引言 Tensorflow的数据读取有三种方式： ...

u010329292 (http://blog.csdn.net/u010329292) 2017-03-30 11:02 2065

目录

Tensorflow数据读取方法 (/aitazhixin/article/details/73614818)

喜欢Tensorflow读取文件的三种方式：预读取，喂数据，读文件

收藏

aitazhixin (http://blog.csdn.net/aitazhixin) 2017-06-23 11:32 118

评论

tensorflow读取文件数据 (/sb_ers/article/details/56665947)

TensorFlow程序读取数据一共有3种方法:供给数据(Feeding)： 在TensorFlow程序运行的每一步， 让Python代码来供给数据。 从文件读取数据： 在TensorFlow图的起...

分享

sb_ers (http://blog.csdn.net/sb_ers) 2017-02-23 10:00 142

Tensorflow数据读取方式 (/l_xyy/article/details/71640200)

Tensorflow数据读取方式 关于tensorflow（简称TF）数据读取方式，官方给出了三种： 供给数据（Feeding）：在TF程序运行的每一步，让python代码来供给数据。 ...

l_xyy (http://blog.csdn.net/l_xyy) 2017-05-11 16:25 85

返回顶部

Tensorflow图片数据读取 (/woshilimengxi/article/details/52648377)