

rockingdingo的博客

目录视图



摘要视图

RSS 订阅

2



评论(12)

收藏

举报



个人资料



rockingdingo

| | | | |
|----|----|----|----|
| 原创 | 粉丝 | 喜欢 | 评论 |
| 10 | 31 | 2 | 24 |

等级： **博客 3** 访问量：5万+

积分：570 排名：8万+

Tensorflow C++ 编译和调用图模型

标签： tensorflow c++ protobuf freeze_graph 编译

2017年07月19日 23:00:54

15415人阅读

分类： 机器学习算法 (4) tensorflow (3)

目录(?)

[+]

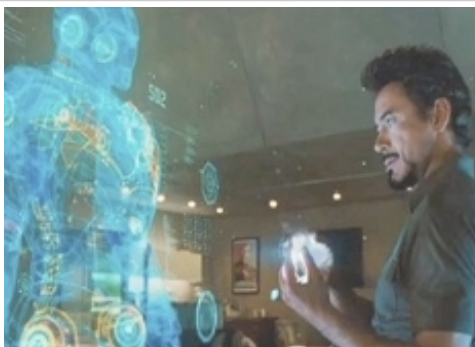
Github下载完整代码

tensorflow freeze_graph.py 工具

简介

最近在研究如何打通tensorflow线下 python 的脚本训练建模, 利用freeze_graph工具输出.pb图文件, 之后再线上生产环境用C++代码直接调用预先训练好的模型完成预测的工作, 而不需要用自己写的Inference的函数。因为目前tensorflow提供的C++的API比较少, 所以参考了几篇已有的日志, 踩了不少坑一并记录下来。写了一个简单的AI模型对Iris数据集分类的Demo。

梳理过后的流程如下：



全息投影沙盘



文章搜索

文章分类

R语言 (1)

机器学习算法 (5)

tensorflow (4)

文章存档

2017年7月 (1)

2017年2月 (8)

2016年10月 (1)

阅读排行

Tensorflow 自动文摘: 基于Seq... (17325)

1. python脚本中定义自己的模型，训练完成后将tensorflow graph定义导出为protobuf的二进制文件或文本文件（一个仅有tensor定义但不包含权重参数的文件）；

2. python脚本训练过程保存模型参数文件 *.ckpt。

3. 调用tensorflow自带的freeze_graph.py 小工具，输入为格式*.pb或*.pbtxt的protobuf文件和*.ckpt的参数文件，输出为一个新的同时包含图定义和参数的*.pb文件；这个步骤的作用是把checkpoint .ckpt文件中的参数转化为常量const operator后和之前的tensor定义绑定在一起。



2

4. 在C++中新建Session，只需要读取一个绑定后的模型文件.pb, 进行预测，利用Session->Run()获得输出的tensor的值就okay；



5. 编译和运行，这时有两个选择：



a) 一种是在tensorflow源代码的子目录下新建自己项目的目录和代码，然后用bazel来编译成一个很大的100多MB的二进制文件，这个方法的缺点在于无法把预测模块集成在自己的代码系统和编译环境如cmake, bcloud中，迁移性和实用性不强；参考: (<https://medium.com/jim-fleming/loading-a-tensorflow-graph-with-the-c-api-4caaff88463f>) 如果打不开貌似有中文翻译版的博客

b) 另一种是自己把tensorflow源代码编译成一个.so文件，然后在自己的C++代码环境中依赖这个文件完成编译。C的API依赖libtensorflow.so，C++的API依赖libtensorflow_cc.so

运行成功后

下面通过具体的例子写了一个简单的ANN预测的demo，应该别的模型也可以参考或者拓展C++代码中的基类。

测试环境：MacOS, 需要依赖安装：tensorflow, bazel, protobuf, eigen(一种矩阵运算的库)；

配置环境

系统安装 HomeBrew, Bazel, Eigen

[python]

| | |
|----------------------------|---------|
| Tensorflow C++ 编译和调用图... | (15362) |
| Tensorflow并行计算：多核(mu... | (10235) |
| Tensorflow进行POS词性标注N... | (8632) |
| 基于Tensorflow的自然语言处... | (2215) |
| MCMC马尔可夫蒙特卡洛方法... | (967) |
| R语言中的Softmax Regression... | (836) |
| Bayes贝叶斯方法-均值和协方... | (714) |
| Bayes贝叶斯方法-均值和协方... | (631) |
| ANN神经网络模型的置信区间... | (625) |

最新评论

Tensorflow C++ 编译...

L木木 : [reply]yuanmartin[reply] 出现“fatal error C 1014: 包...

Tensorflow C++ 编译...

yuanmartin : 请问博主：我的编译的时候出现如下报错： 1> main.cpp 1>...

Tensorflow 自动文摘: ...

手机用户2498705991 : github上没有了，请问楼主还会贴出来吗？

Tensorflow并行计算：多核...

sdsz_zq : [reply]hedongya[reply] 因为是伪并行？

Tensorflow C++ 编译...

何金水 : undefined reference to `tensorflow::Session Option...

Tensorflow C++ 编译...

michael2008bj : 请问博主，Tensor类和C++自有的类型转换方便吗？比如Tensor和array或Tensor和...

Tensorflow 自动文摘: ...

doggyche : [reply]fcl346[reply] 请问，现在github上404了，能分享下代码么？多谢。

Tensorflow进行POS词性...

胡椒CSDN : 楼主，您好，可以提供一下train dev和test这几个文件吗

```

1. # Mac下安装包管理工具homebrew
2. ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
3.
4. # 安装Bazel, Google 的一个编译工具
5. brew install bazel
6.
7. # 安装Protobuf, 参考 http://blog.csdn.net/wwq_1111/article/details/50215645
8. git clone https://github.com/google/protobuf.git
9. brew install automake libtool
10. ./autogen.sh
11. ./configure
12. make check
13. make && make install
14.
15. # 安装 Eigen, 用于矩阵运算
16. brew install eigen

```



2



下载编译tensorflow源码

[python]

```

1. # 从github下载tensorflow源代码
2. git clone --recursive https://github.com/tensorflow/tensorflow
3.
4. ## 进入根目录后编译
5. # 编译生成.so文件, 编译C++ API的库 (建议)
6. bazel build //tensorflow:libtensorflow_cc.so
7.
8. # 也可以选择, 编译C API的库
9. bazel build //tensorflow:libtensorflow.so

```

在等待30多分钟后, 如果编译成功, 在tensorflow根目录下出现 bazel-bin, bazel-genfiles 等文件夹, 按顺序执行以下命

Tensorflow C++ 编译...

IceBear6 : [reply]u012545225[/reply] 你好，请问你的问题解决了没？我也遇到了这个问题

Tensorflow C++ 编译...

yuanwx_ : 博主好，最新的tensorflow存储文件变了，请问步骤需要变吗？



app开发报价单



联系我们



请扫描二维码联系客服

✉ webmaster@csdn.net

☎ 400-660-0108

🗣 QQ客服 🗣 客服论坛

关于 招聘 广告服务 🐼 百度

©1999-2018 CSDN版权所有

京ICP证09002463号

经营性网站备案信息

网络110报警服务

中国互联网举报中心

令将对应的libtensorflow_cc.so文件和其他文件拷贝进入 /usr/local/lib/ 目录

[python]

```
1. mkdir /usr/local/include/tf
2. cp -r bazel-genfiles/ /usr/local/include/tf/
3. cp -r tensorflow /usr/local/include/tf/
4. cp -r third_party /usr/local/include/tf/
5. cp -r bazel-bin/tensorflow/libtensorflow_cc.so /usr/local/lib/
```



2



这一步完成后，我们就准备好了libtensorflow_cc.so文件等，后面在自己的C++编译环境和代码目录下编译时链接这些库即可。

1. Python线下定义模型和训练

我们写了一个简单的脚本，来训练一个包含1个隐含层的ANN模型来对Iris数据集分类，模型每层节点数：[5, 64, 3]，具体脚本参考项目：

<https://github.com/rockingdingo/tensorflow-tutorial>

1.1 定义Graph中输入和输出tensor名称

为了方便我们在调用C++ API时，能够准确根据Tensor的名称取出对应的结果，在python脚本训练时就要先定义好每个tensor的tensor_name。如果tensor包含命名空间namespace的如"namespace_A/tensor_A" 需要用完整的名称。(Tips: 对于不清楚tensorname具体是什么的，可以在输出的.pbtxt文件中找对应的定义)；这个例子中，我们定义以下3个tensor的tensorname

[python]

```

1. class TensorNameConfig(object):
2.     input_tensor = "inputs"
3.     target_tensor = "target"
4.     output_tensor = "output_node"
5.     # To Do

```

1.2 输出graph的定义文件*.pb和参数文件 *.ckpt



2

我们要在训练脚本nn_model.py中加入两处代码：第一处是将tensorflow的graph保存成./models/目录下一个文件nn_model.pbtxt, 里面包含有图中各个tensor的定义名称等信息。第二处是在nn_model.py代码中加入保存参数文件的代码，将训练好的ANN模型的权重Weight和Bias同时保存到./ckpt目录下的*.ckpt, *.meta等文件。最后执行 python nn_model.py 就可以完成训练过程

[python]

```

1. # 保存图模型
2. tf.train.write_graph(session.graph_def, FLAGS.model_dir, "nn_model.pbtxt", as_text=True)
3.
4. # 保存 Checkpoint
5. checkpoint_path = os.path.join(FLAGS.train_dir, "nn_model.ckpt")
6. model.saver.save(session, checkpoint_path)
7.
8. # 执行命令完成训练过程
9. python nn_model.py

```

1.3 使用freeze_graph.py小工具整合模型freeze_graph

最后利用 tensorflow 自带的 freeze_graph.py 小工具把 .ckpt 文件中的参数固定在 graph 内，输出 nn_model_frozen.pb

```
[python]
1. # 运行freeze_graph.py 小工具
2. # freeze the graph and the weights
3. python freeze_graph.py --input_graph=./model/nn_model.pbtxt --input_checkpoint=./ckpt/nn_model.ckpt --output_graph=./model/nn_model_frozen.pb --output_node_names=output_node
4.
5. # 或者执行
6. sh build.sh
7.
8. # 成功标志:
9. # Converted 2 variables to const ops.
10. # 9 ops in the final graph.
```



2



脚本中的参数解释：

--input_graph: 模型的图的定义文件nn_model.pb（不包含权重）；

--input_checkpoint: 模型的参数文件nn_model.ckpt；

--output_graph: 绑定后包含参数的图模型文件 nn_model_frozen.pb；

-- output_node_names: 输出待计算的tensor名字【重要】；

发现tensorflow不同版本下运行freeze_graph.py 脚本时可能遇到的Bug挺多的，列举一下：

```
[python]
1. # Bug1: google.protobuf.text_format.ParseError: 2:1 : Message type "tensorflow.GraphDef" has no field named "J".
2. # 原因: tf.train.write_graph(,as_text=False) 之前写出的模型文件是Binary时,
3. # 读入文件格式应该对应之前设置参数 python freeze_graph.py [***] --input_binary=true,
4. # 如果as_text=True则可以忽略, 因为默认值 --input_binary=false.
5. # 参考: https://github.com/tensorflow/tensorflow/issues/5780
6.
7. # Bug2: Input checkpoint '...' doesn't exist!
8. # 原因: 可能是命令行用了 --input_checkpoint=data.ckpt ,
```

```

9. # 运行 freeze_graph.py 脚本, 要在路径参数前加上 "./" 貌似才能正确识别路径。
10. # 如文件的路径 --input_checkpoint=data.ckpt 变为 --input_checkpoint=./data.ckpt
11. # 参考: http://www.it1me.seriousdigitalmedia.com/it-answers?id=42439233&ttl=How+to+use+free
    eze_graph.py+tool+in+TensorFlow+v1
12.
13. # Bug3: google.protobuf.text_format.ParseError: 2:1 : Expected identifier or number.
14. # 原因: --input_checkpoint 需要找到 .ckpt.data-000*** 和 .ckpt.meta等多个文件,
15. # 因为在 --input_checkpoint 参数只需要添加 ckpt的前缀, 如: nn_model.ckpt, 而不是完整的路径nn_mode
    l.ckpt.data-000***
16. # .meta .index .data checkpoint 4个文件
17.
18. # Bug4: # you need to use a different restore operator?
19. # tensorflow.python.framework.errors_impl.DataLossError: Unable to open table file ./pos.c
    kpt.data-000000-of-000001: Data loss: not an sstable (bad magic number) perhaps your file i
    s in a different file format and you need to use a different restore operator?
20. # Saver 保存的文件用格式V2, 解决方法更新tensorflow....
21.
22. # 欢迎补充

```



2



最后如果输出如下: Converted variables to const ops. * ops in the final graph 就代表绑定成功了! 发现绑定了参数的的.pb文件大小有10多MB。

2. C++API调用模型和编译

在C++预测阶段, 我们在工程目录下引用两个tensorflow的头文件:

2.1 C++加载模型

?




[cpp]

```
1. #include "tensorflow/core/public/session.h"
```

```
2. | #include "tensorflow/core/platform/env.h"
```

在这个例子中我们把C++的API方法都封装在基类里面了。FeatureAdapterBase 用来处理输入的特征，以及ModelLoaderBase提供统一的模型接口load()和predict()方法。然后可以根据自己的模型可以继承基类实现这两个方法，如本demo中的ann_model_loader.cpp。可以参考下，就不具体介绍了。

a) 新建Session, 从model_path 加载*.pb模型文件，并在Session中创建图。预测的核心代码如下：

?  2  

```
[cpp]
1. // @brief: 从model_path 加载模型，在Session中创建图
2. // ReadBinaryProto() 函数将model_path的protobuf文件读入一个tensorflow::GraphDef的对象
3. // session->Create(graphdef) 函数在一个Session下创建了对应的图；
4.
5. int ANNModelLoader::load(tensorflow::Session* session, const std::string model_path) {
6.     //Read the pb file into the graphdef member
7.     tensorflow::Status status_load = ReadBinaryProto(Env::Default(), model_path, &graphdef);
8.     if (!status_load.ok()) {
9.         std::cout << "ERROR: Loading model failed..." << model_path << std::endl;
10.        std::cout << status_load.ToString() << "\n";
11.        return -1;
12.    }
13.
14.    // Add the graph to the session
15.    tensorflow::Status status_create = session->Create(graphdef);
16.    if (!status_create.ok()) {
17.        std::cout << "ERROR: Creating graph in session failed..." << status_create.ToString() << std::endl;
18.        return -1;
19.    }
20.    return 0;
21. }
```


b) 预测阶段的函数调用 `session->Run(input_feature.input, {output_node}, {}, &outputs);`

参数 `const FeatureAdapterBase& input_feature`, 内部的成员 `input_feature.input` 是一个 Map 型, `std::vector<std::pair>`, 类似于python里的 `feed_dict={"x":x, "y": y}`, 这里的C++代码中的输入 `tensor_name`也一定要和python训练脚本中的一致, 如上文中设定的"inputs", "targets" 等。调用基类 `FeatureAdapterBase`中的方法 `assign(std::string, std::string tname, std::vector* vec)` 函数来定义。

参数 `const std::string output_node`, 对应的就是在python脚本中定 输出节点的名称, 如" `name_scope/output_node`"

[cpp]

```
1. int ANNModelLoader::predict(tensorflow::Session* session, const FeatureAdapterBase& input_
   feature,
2.     const std::string output_node, double* prediction) {
3.     // The session will initialize the outputs
4.     std::vector<tensorflow::Tensor> outputs;           //shape [batch_size]
5.
6.     // @input: vector<pair<string, tensor>>, feed_dict
7.     // @output_node: std::string, name of the output node op, defined in the protobuf file
8.
9.     tensorflow::Status status = session->Run(input_feature.input, {output_node}, {}, &outp
   uts);
10.    if (!status.ok()) {
11.        std::cout << "ERROR: prediction failed..." << status.ToString() << std::endl;
12.        return -1;
13.    }
14.    // ...
15. }
```

2.1 C++编译的方法

记得我们之前预先编译好的 `libtensorflow_cc.so`文件, 要成功编译需要链接那个库。运行下列命令:

[cpp]

```
1. # 使用g++
2. g++ -std=c++11 -o tfcpp_demo \
3. -I/usr/local/include/tf \
4. -I/usr/local/include/eigen3 \
5. -g -Wall -D_DEBUG -Wshadow -Wno-sign-compare -w \
6. `pkg-config --cflags --libs protobuf` \
7. -L/usr/local/lib/libtensorflow_cc \
8. -ltensorflow_cc main.cpp ann_model_loader.cpp
```



2



参数含义:

- a) -I/usr/local/include/tf # 依赖的include文件
- b) -L/usr/local/lib/libtensorflow_cc # 编译好的libtensorflow_cc.so文件所在的目录
- c) -ltensorflow_cc # .so文件的文件名

为了方便调用, 尝试着写了一个Makefile文件, 将里面的路径换成自己的, 每次直接用make命令执行就好

[python]

```
1. make
```

此外, 在直接用g++来编译的过程中可能会遇到一些Bug, 现在记录下来

[python]

```
1. # Bug1: main.cpp:9:10: fatal error: 'tensorflow/core/public/session.h' file not found
2. # include "tensorflow/core/public/session.h"
3. # 原因: 这个应该就是编译阶段没有找到之前编译好的tensorflow_cc.so 文件, 检查-I和-L的路径参数
4.
5. # Bug2: fatal error: 'google/protobuf/stubs/common.h' file not found
```

```
6. # 原因：没有成功安装 protobuf文件
7. # 参考：http://blog.csdn.net/wwq_1111/article/details/50215645
8.
9. # Bug3: /usr/local/include/tf/third_party/eigen3/unsupported/Eigen/CXX11/Tensor:1:10: fatal
   # error: 'unsupported/Eigen/CXX11/Tensor' file not found
10. # 原因：没有安装或找到Eigen的路径
11. # 参考之前安装Eigen的步骤
```



2



3. 运行

最后试着运行一下之前编译好的可执行文件 tfcpp_demo

```
[python]
1. # 运行可执行文件，输入参数 model_path指向之前的包含参数的模型文件 nn_model_frozen.pb
2. folder_dir=`pwd`
3. model_path=${folder_dir}/model/nn_model_frozen.pb
4. ./tfcpp_demo ${model_path}
5.
6. # 或者直接执行脚本：
7. sh run.sh
```

?

我们试着预测一个样本[1,1,1,1,1]，输出该样本对应的分类和概率。进行到这一步，我们终于成功完成了在python中定义模型和训练，然后在C++生产代码中进行编译和调用的流程。

参考资料和延伸阅读

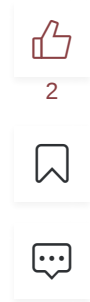
Github模型源代码地址

medium: loading-a-tensorflow-graph-with-the-c-api

exporting-trained-tensorflow-models-to-c-the-right-way

deepnlp 0.1.6 : Python Package Index

-



- [上一篇](#) ANN神经网络模型的置信区间的预估-R包nnetpredint应用(一)

体验TeamCity持续集成工具



支持JAVA, .NET, C++, Python, PHP等多种语言., 即刻下载并注册

您还没有登录, 请[\[登录\]](#)或[\[注册\]](#)

查看评论



yuanmartin

8楼 2018-03-07 11:30发表

请问博主：

我的编译的时候出现如下报错：

1> main.cpp

```
1>g:\study\tensorflow\src_code\tensorflow\tensorflow\google\protobuf\io\coded_stream.h(870): warning C4800: "google::protobuf::internal::AtomicWord": 将值强制为布尔值"true"或"false"(性能警告)
```

```
1>g:\study\tensorflow\src_code\tensorflow\tensorflow\google\protobuf\io\coded_stream.h(874): warning C4800: "google::protobuf::internal::Atomic32":
```

[查看更多评论](#)

2



如何用C++实现自己的Tensorflow



dev_csdn 2017年11月10日 15:59 13885

原文：How To Write Your Own Tensorflow in C++ 作者：Ray Zhang 翻译：无阻我飞扬 摘要：TensorFlow是由谷歌基于DistBelief...

Tensorflow C++ 学习（一）搭建环境



jmh1996 2017年06月14日 00:38 11035

前言Tensorflow 网上大部分是python的资料较多，而C++方面的极少，因此，接下来会有一些博客用于学习tensorflow,记录学习的过程。加油！搭建环境既然使用C++的API，那第一...

程序员不会英语怎么行？

北大猛男教你：不背单词和语法，一个公式学好英语



c++调用tensorflow教程



qq_34484472 2017年08月02日 21:33 11779

目前深度学习越来越火，学习、使用tensorflow的相关工作者也越来越多。但是目前绝大部分的python都是拥有着丰富的python的API，而c++的API不够完善。这就导致绝大多是使用tenso...

如何实现Visual Studio C++调用TensorFlow代码

对机器学习感兴趣的VS老司机们，在Windows上搭建好TensorFlow开发环境后，应该...
+程序调用TensorFlow代码。只要能调用成功，你就可以将机器学习融入到你的...



ShadowN1ght 2017年09月11日 15:45 4429



Tensorflow C++学习(二)



jmh1996 2017年06月14日 10:24 8558

前言本节主要介绍（一）中的代码，了解如何使用C++创建图和tensor，并使用它们进行计算。代码// tensorflow/cc/example/example.cc#include "tensorf...

英语文档看不懂？教你一个公式秒懂英语！

软件工程出身的英语老师，教你用数学公式读懂天下英文→



Tensorflow C++ 编译和调用图模型



c2a2o2 2017年09月25日 16:36 1536

简介 最近在研究如何打通tensorflow线下 python 的脚本训练建模, 利用freeze_graph工具输出.pb图文件，之后再线上生产环境用C++代码直接调用预先训练好的模型完成预测...

C++ API载入tensorflow graph



elaine_bao

2017年12月03日 16:19 478

通过C++ API载入tensorflow graph在tensorflow repo中，和C++相关的tutorial远没有python的那么详尽。这篇文章主要介绍如何利用C++来载入一个预训练好的...

64位Windows10下tensorflow的安装



mslddzm

2017年12月18日 14:22 1530

网上有很多关于Windows系统下安装tensorflow的教程，方法大致有几种，但是细节却同。我看了很多同学写的安装教程，实践后都遇到过各种各样的小问题（其实主要还是自己对TF不熟悉）。在对很...



tensorflow c++编译



fireflychh

2017年12月23日 17:55 1115

1. 安装bazel，可以在bazel官网上查找到具体安装方法 2. git clone 一份tensorflow的源码，然后上官网看，（需要clone相对稳定的版本如r1.3,否则可能...

TensorFlow学习笔记(一)



dd864140130

2017年05月12日 23:15 14886

最近致力于深度学习,希望在移动领域能够找出更多的应用点.其中TensorFlow作为目前的一个热点值得我们重点关注....

Tensorflow C++库的编译和使用方法



xinchen1234

2017年12月08日 11:52 317

Tensorflow C++库的编译和使用方法网上编译或者安装Tensorflow的教程大多数都是Python相关的，有少量C或者C++的，但版本又比较老不适合最新的Tensorflow教程，本博记录...

如何学习python自动化测试

python自动化

百度广告

tensorflow c++接口编译



junxiacaocao 2018年01月25日 17:11 80

源码编译安装部分省，c++接口编译参考：<http://www.deeplnlp.org/blog/tensorflow-cpp-b> r-production/ ...

TensorFlow 研究实践 三



forest_world 2016年05月10日 12:48 2544

C++ API的使用方法 用C++运行同样的Inception-v3模型 一、下载定义模型的GraphDef文件（在TensorFlow的根目录下运行）：`wget https://storage...`

[tensorflow应用之路]模型的存储、读取和预测（c++/python）

之前的文章中讲了如何使用tensorflow源码编译一个c++版的动态库。同时留下了一个问题：能否在C++中读取预先训练好的模型呢？——答案是肯定的。下面，就来一一介绍tensorflow模型在...



h8832077 2018年01月08日 18:55 234

tensorflow debug c++ code



mydear_11000 2016年08月19日 16:56 1625

TensorFlow's C++ code executes in the same process as the Python code that calls it (or, if you are ...

程序猿不会英语怎么行？英语文档都看不懂！

软件工程出身的英语老师，教你用数学公式读懂天下英文 →



TensorFlow 基本使用

Tensorflow的基本使用简介。



YhL_Leo



2016年02月02日 13:35

44720



Ubuntu1604 下编译并使用tensorflow c++库

----- 安装tensorflow c++库 ----- 1. 安装bazel 具体怎么安装可以在bazel官网看看 有直接的...



lovekkss

2017年08月11日 19:04

1224



Loading a TensorFlow graph with the C++ API

Check out the related post: Loading TensorFlow graphs via host languages (using the C API). The cur...



mydear_11000

2016年08月09日 08:36

2645

tensorflow1.4 c++编译以及API使用



zwx1995zwx

2018年01月15日 14:20

281

摘要：最近在研究如何使用tensorflow c++ API调用tensorflow python环境下训练得到的网络模型文件。参考了很多博客，文档，一路上踩了很多坑，现将自己的方法步骤记录下来，希...

深度学习（五十六）tensorflow项目构建流程



hjimce

2016年07月13日 17:05

25616

import tensorflow.nn.rnn_cell lstm = rnn_cell.BasicLSTMCell(lstm_size)#创建一个lstm cell单元类，隐藏层神经元个数为l...

