**TensorFlow的模型格式**

1.**CheckPoint（.ckpt）**

在训练TensorFlow模型时，每迭代若干轮需要保存一次权值到磁盘，称为“checkpoint”。这种格式文件是由tf.train.Saver()对象调用saver.save()生成的，只包含若干Variables对象序列化后的数据，**不包含图结构**，所以只给checkpoint模型不提供代码是无法重新构建计算图的。载入checkpoint时，调用saver.restore(session, checkpoint\_path)。

2.**MetaGraphDef（.meta）**

类：MetaGraphDef，包含MetaInfoDef、GraphDef、SaverDef、CollectionDef。

序列化存储格式：protobuf，.meta文件。

3.**GraphDef（.pb）**

这种格式文件包含protobuf对象序列化后的数据，**包含了计算图**，可以从中得到所有运算符（operators）的细节，也包含张量（tensors）和Variables定义，**但不包含Variable的值**，因此**只能从中恢复计算图，但一些训练的权值仍需要从checkpoint中恢复**。

利用.pb文件构建计算图：

def load\_graph(model\_file):

graph = tf.Graph()

graph\_def = tf.GraphDef()

with open(model\_file, "rb") as f:

graph\_def.ParseFromString(f.read())

with graph.as\_default():

tf.import\_graph\_def(graph\_def)

return graph

4.**FrozenGraphDef（.pb）**

TensorFlow一些例程中用到.pb文件作为预训练模型，这**和上面GraphDef格式稍有不同**，属于**冻结（Frozen）后的GraphDef文件**，简称FrozenGraphDef格式。这种文件格式**不包含Variables节点**。将GraphDef中所有**Variable节点转换为常量**（其值从checkpoint获取），就变为**FrozenGraphDef格式**。代码可以参考 tensorflow/python/tools/freeze\_graph.py。.pb 为**二进制文件**，实际上protobuf也支持文本格式（.pbtxt），但包含权值时文本格式会占用大量磁盘空间，一般不用。

=----------------------------------------------------------------------------------------------------=

获取一个分类模型的三种方法

最新的物体识别模型可能含有数百万个参数，将耗费几周的时间去完全训练。因此我们采用迁移学习的方法，在已经训练好的模型（基于ImageNet）上调整部分参数，以实现对新类别的分类。关于迁移学习的理论，可以参考[DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition](https://link.jianshu.com?t=https://arxiv.org/pdf/1310.1531v1.pdf" \t "_blank)。  
事实上，获取一个分类模型有三种方式：

* Train from scratch  
  从头开始训练
* Fine-tune a model  
  对一个网络调优
* Retrain a model  
  对一个网络重训

其难度由上至下递减，当然，执行效果也是逐级递减。目前我们所要实现的迁移学习对应于最后一个部分：Retrain a model。其与Fine-tune a model方法的差距还是比较明显的，以Inception\_v3模型为例：

* Retrain a model是利用基于ImageNet图像训练的Inception\_v3模型所导出的pb文件，更改最后的softmax layer为自己需要的分类器，然后对这一更改的softmax layer进行训练。除开最后一层，其他层的参数全部固化，无法更新。因此，在实际的Retrain中，往往先将数据集（包含训练集、验证集与测试集）中的所有图片导入到Inception\_v3模型中，获取最后一层的输入，或者说是倒数第二层的输出，定义为Bottlenecks。然后直接使用Bottlenecks对最后更改的softmax layer进行训练，将大幅度提升训练速度。
* Fine-tune a model是利用基于ImageNet图像训练的Inception\_v3模型所导出的Ckpt文件，在训练过程中，整个网络的参数都可以随之修改，不仅仅局限于被替换掉的softmax layer。
  + Fine-tune例子
  + 使用freeze\_graph.py文件可以将ckpt文件中所有参数固化，转为pb文件。

=----------------------------------------------------------------------------------------------------=

一些自带的python工具

1. 图片分类模型测试：tensorflow/examples/label\_image.py
2. 当retrain完后需要参数固定否则会出现类似错误：

Node 'final\_result:0' does not exist in model

：

|  |
| --- |
| python tensorflow/python/tools/strip\_unused.py \  --input\_graph=tensorflow/examples/label\_image/data/output\_graph2.pb \  --output\_graph=tensorflow/examples/label\_image/data/tmp/new.pb \  --input\_node\_names=Mul \  --output\_node\_names=final\_result \  --input\_binary=true |

处理后得到一个新的pb文件，在代码中需修改成下面这样：

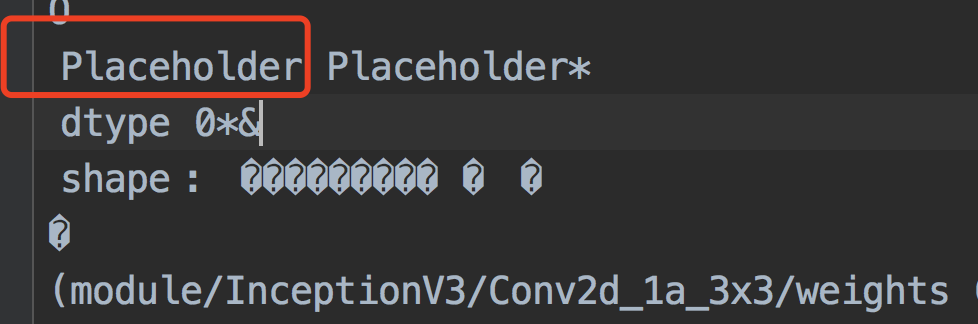
//注意，这里的input\_size要与训练时的对齐否则会报错：//ValueError: Cannot feed value of shape (1, 299, //299, 3) for Tensor 'import/input:0', which has //shape '(1, 128, 128, 3)',也就是说大小要与训练时的输入//图片大小一致，使用label\_image.py测试时如果//input\_node\_names是对的，但没有对齐就可以看到这个错提，//跟据提示进行修正就可以了

private static final int *INPUT\_SIZE* = 299;//这里//的大小要与模型训练时的大小对齐，怎么知道这个大小，在训练时//模型名称如“mobilenet\_1.0\_224”对齐尺寸为224，那我们应//该设置INPUT\_SIZE为224，inception模型类似  
private static final int *IMAGE\_MEAN* = 128;  
private static final float *IMAGE\_STD* = 128;  
private static final String *INPUT\_NAME* = "Mul";//这//个是inception\_v3老版编译时设定的节点名，在新版训练代码//中使用的是Placeholder，可以在pb文件中看到，而//mobilenet中使用的是input  
private static final String *OUTPUT\_NAME* = "final\_result";

部份必要的解释：

1、--input\_node\_names对应的是create\_model\_info方法中的resized\_input\_tensor\_name字段定义的节点名，

--output\_mode\_names对应的是--final\_tensor\_name，在老版的python训练代码中是可以找到resized\_input\_tensor\_name，在训练完的pb文件打开后看到第一个字段就是，比如Placeholder这样的



1. 防止gradle对pb文件进行压缩，破坏文件

aaptOptions { noCompress 'pb' }

参数链接：<https://blog.csdn.net/leifengpeng/article/details/78754405>

=----------------------------------------------------------------------------------------------------=