Tensorflow: how to save/restore a model?

After you train a model in Tensorflow:

1. How do you save the trained model?
2. How do you later restore this saved model?

python machine-learning tensorflow

edited 8 hours ago
Sun Qingyao
6,166 6 20 49

asked Nov 17 '15 at 14:37
mathetes
1,066 3 12 22

Were you able to restore variables used in inception model? I am also trying the exact same problem but I am unable to write set of variables that were used while training the inception model (of which I have ckpt file) – Sangram Oct 11 '16 at 17:52

I haven't tried with the inception model. Do you have the model's network structure with its names? You have to replicate the network and then load the weights and biases (the ckpt file) as Ryan explains. Maybe something has changed since Nov'15 and there's a more straightforward approach now, I'm not sure – mathetes Oct 11 '16 at 18:22

Ohh okay. I have loaded other pre-trained tensorflow models previously but was looking for variable specifications of inception model. Thanks. – Sangram Oct 11 '16 at 18:30

8 Answers

In(and After) TensorFlow version 0.11.0RC1, you can save and restore your model directly by calling `tf.train.export_meta_graph` and `tf.train.import_meta_graph` according to https://www.tensorflow.org/programmers_guide/meta_graph

save model:

```
w1 = tf.Variable(tf.truncated_normal(shape=[10]), name='w1')
w2 = tf.Variable(tf.truncated_normal(shape=[20]), name='w2')
tf.add_to_collection('vars', w1)
tf.add_to_collection('vars', w2)
saver = tf.train.Saver()
sess = tf.Session()
sess.run(tf.global_variables_initializer())
saver.save(sess, 'my-model')
# `save` method will call `export_meta_graph` implicitly.
# you will get saved graph files:my-model.meta
```

restore model:

```
sess = tf.Session()
new_saver = tf.train.import_meta_graph('my-model.meta')
new_saver.restore(sess, tf.train.latest_checkpoint('./'))
all_vars = tf.get_collection('vars')
for v in all_vars:
    v_ = sess.run(v)
    print(v_)
```

edited Feb 19 at 0:18
KT12
35 5

answered Nov 23 '16 at 13:24
lei du
1,146 4 4

That's great! I'm selecting your answer instead as it will serve people better from now on – mathetes Nov 23 '16 at 17:04

3 how to load variables from the saved model? How to copy values in some other variable? – neel Dec 19 '16 at 8:58

3 I am unable to get this code working. The model does get saved but I cannot restore it. It is giving me this error. `<built-in function TF_Run> returned a result with an error set` – Saad Qureshi Jan 8 at 9:05

When after restoring I access the variables like shown above, it works. But I cannot get the variables more directly using `tf.get_variable_scope().reuse_variables()` followed by `var = tf.get_variable("varname")` . This gives me the error: "ValueError: Variable varname does not exist, or was not created with tf.get_variable()." Why? Should this not be possible? – Johsm Jan 12 at 14:16

If I add `print sess.run([w1, w2])` in the save section of the code it correctly prints the variables. But if I add that line at the end of the restore code I get an error: `NameError: name 'w1' is not defined` . If the graph and variables are restored then what is wrong here? – Ron Cohen Jan 12 at 17:41

For TensorFlow version < 0.11.0RC1:

The checkpoints that are saved contain values for the `Variable` s in your model, not the model/graph itself, which means that the graph should be the same when you restore the checkpoint.

Here's an example for a linear regression where there's a training loop that saves variable checkpoints and an evaluation section that will restore variables saved in a prior run and compute predictions. Of course, you can also restore variables and continue training if you'd like.

```
x = tf.placeholder(tf.float32)
y = tf.placeholder(tf.float32)

w = tf.Variable(tf.zeros([1, 1], dtype=tf.float32))
b = tf.Variable(tf.ones([1, 1], dtype=tf.float32))
y_hat = tf.add(b, tf.matmul(x, w))
```

```
...more setup for optimization and what not...

saver = tf.train.Saver()  # defaults to saving all variables - in this case w and b

with tf.Session() as sess:
    sess.run(tf.initialize_all_variables())
    if FLAGS.train:
        for i in xrange(FLAGS.training_steps):
            ...training loop...
            if (i + 1) % FLAGS.checkpoint_steps == 0:
                saver.save(sess, FLAGS.checkpoint_dir + 'model.ckpt',
                           global_step=i+1)
    else:
        # Here's where you're restoring the variables w and b.
        # Note that the graph is exactly as it was when the variables were
        # saved in a prior training run.
        ckpt = tf.train.get_checkpoint_state(FLAGS.checkpoint_dir)
        if ckpt and ckpt.model_checkpoint_path:
            saver.restore(sess, ckpt.model_checkpoint_path)
        else:
            ...no checkpoint found...

        # Now you can run the model to get predictions
        batch_x = ...load some data...
        predictions = sess.run(y_hat, feed_dict={x: batch_x})
```

Here are the docs for `Variable` s, which cover saving and restoring. And here are the docs for the `Saver` .

edited Mar 3 at 15:43

answered Nov 17 '15 at 16:30

Ryan Sepassi
1,088 1 8 4

Thank you Ryan, that's exactly what I was looking for! – mathetes Nov 17 '15 at 18:31

2 what FLAGS?? where do they come from?? – Nomi Mar 23 '16 at 20:02

1 FLAGS are user-defined. Here's an example of defining them: github.com/tensorflow/tensorflow/blob/master/tensorflow/… – Ryan Sepassi Mar 26 '16 at 1:19

in which format does `batch_x` need to be? Binary? Numpy array? – pepe Jun 5 '16 at 16:27

@pepe Numpy arrary should be fine. And the element's type should correspond to the type of the placeholder. [link]tensorflow.org/versions/r0.9/api_docs/python/… – Donny Jun 9 '16 at 16:46

There are two parts to the model, the model definition, saved by `Supervisor` as `graph.pbtxt` in the model directory and the numerical values of tensors, saved into checkpoint files like `model.ckpt-1003418` .

The model definition can be restored using `tf.import_graph_def` , and the weights are restored using `Saver` .

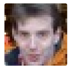
However, `Saver` uses special collection holding list of variables that's attached to the model Graph, and this collection is not initialized using import_graph_def, so you can't use the two together at the moment (it's on our roadmap to fix). For now, you have to use approach of Ryan Sepassi -- manually construct a graph with identical node names, and use `Saver` to load the weights into it.

(Alternatively you could hack it by using by using `import_graph_def` , creating variables manually, and using `tf.add_to_collection(tf.GraphKeys.VARIABLES, variable)` for each variable, then using `Saver`)

edited Aug 15 '16 at 0:27

David Silva-Barrera
101 8

answered Nov 17 '15 at 17:22



Yaroslav Bulatov
20.1k 8 52 101

Perfectly clear, thanks for your help! – mathetes Nov 17 '15 at 18:30

In the classify_image.py example that uses inceptionv3, only the graphdef is loaded. Does it mean that now the GraphDef also contains the Variable ? – jrabary Feb 5 '16 at 20:42

@jrabary The model has probably been frozen. – Eric Platon Mar 21 '16 at 2:27

As Yaroslav said, you can hack restoring from a graph_def and checkpoint by importing the graph, manually creating variables, and then using a Saver.

I implemented this for my personal use, so I though I'd share the code here.

Link: https://gist.github.com/nikitakit/6ef3b72be67b86cb7868

(This is, of course, a hack, and there is no guarantee that models saved this way will remain readable in future versions of TensorFlow.)

answered Jan 5 '16 at 10:35

nikitakit
98 4

You can also take this easier way.

Step.1 - Initialize all your variables

```
W1 = tf.Variable(tf.truncated_normal([6, 6, 1, K], stddev=0.1), name="W1")
B1 = tf.Variable(tf.constant(0.1, tf.float32, [K]), name="B1")

Similarly, W2, B2, W3, .....
```

Step.2 - Save the list inside Model Saver and Save it

```
model_saver = tf.train.Saver()

# Train the model and save it in the end
model_saver.save(session, "saved_models/CNN_New.ckpt")
```

Step. 3 - Restore the model

```
with tf.Session(graph=graph_cnn) as session:
    model_saver.restore(session, "saved_models/CNN_New.ckpt")
    print("Model restored.")
    print('Initialized')
```

Step. 4 - Check Variable

```
W1 = session.run(W1)
print(W1)
```

edited Feb 6 at 21:27

answered Feb 6 at 20:21

Himanshu Babal
111 2 9

Upvoted because it's the simplest working code that answers OPs question. – Hlynur Davíð Hlynsson Feb 7 at 14:06

Hi, How can I save the model after suppose 3000 iterations, similar to Caffe. I found out that tensorflow save only last models despite that I concatenate iteration number with model to differentiate it among all iterations. I mean model_3000.ckpt, model_6000.ckpt, --- model_100000.ckpt. Can you kindly explain why it doesn't save all rather saves only last 3 iterations. – khan Apr 4 at 10:32

If it is an internally saved model, you just specify a restorer for all variables as

```
restorer = tf.train.Saver(tf.all_variables())
```

and use it to restore variables in a current session:

```
restorer.restore(self._sess, model_file)
```

For the external model you need to specify the mapping from the its variable names to your variable names. You can view the model variable names using the command

```
python /path/to/tensorflow/tensorflow/python/tools/inspect_checkpoint.py --
file_name=/path/to/pretrained_model/model.ckpt
```

The inspect_checkpoint.py script can be found in './tensorflow/python/tools' folder of the Tensorflow source.

To specify the mapping, you can use my Tensorflow-Worklab, which contains a set of classes and scripts to train and retrain different models. It includes an example of retraining ResNet models, located here

answered Jul 4 '16 at 7:32

Sergey Demyanov
577 5 8

You can also check out examples in TensorFlow/skflow, which offers `save` and `restore` methods that can help you easily manage your models. It has parameters that you can also control how frequently you want to back up your model.

answered Feb 17 '16 at 3:21

Yuan Tang
323 1 9

As described in issue 6255:

```
use '**./**model_name.ckpt'
saver.restore(sess,'./my_model_final.ckpt')
```

instead of

```
saver.restore('my_model_final.ckpt')
```

edited Feb 2 at 10:04

Grisha Levit
3,539 1 14 28

answered Feb 2 at 9:57

user7505159
11 1