What is a TensorFlow Session?

I've seen a lot of confusion over the rules of `tf.Graph` and `tf.Session` in TensorFlow. It's simple:

- A graph defines the computation. It doesn't compute anything, it doesn't hold any values, it just defines the operations that you specified in your code.
- A session allows to execute graphs or part of graphs. It allocates resources (on one or more machines) for that and holds the actual values of intermediate results and variables.

Let's look at an example.

Update 2020-03-04: Sessions are gone in TensorFlow 2. There is one global runtime in the background that executes all computation, whether run eagerly or as a compiled `tf.function`.

Defining the Graph

We define a graph with a variable and three operations: `variable` returns the current value of our variable. `initialize` assigns the initial value of 42 to that variable. `assign` assigns the new value of 13 to that variable.

```
graph = tf.Graph()
with graph.as_default():
  variable = tf.Variable(42, name='foo')
  initialize = tf.global_variables_initializer()
  assign = variable.assign(13)
```

On a side note: TensorFlow creates a default graph for you, so we don't need the first two lines of the code above. The default graph is also what the sessions in the next section use when not manually specifying a graph.

Running Computations in a Session

To run any of the three defined operations, we need to create a session for that graph. The session will also allocate memory to store the current value of the variable.

```
with tf.Session(graph=graph) as sess:
  sess.run(initialize)
  sess.run(assign)
  print(sess.run(variable))
# Output: 13
```

As you can see, the value of our variable is only valid within one session. If we try to query the value afterwards in a second session, TensorFlow will raise an error because the variable is not initialized there.

```
with tf.Session(graph=graph) as sess:
  print(sess.run(variable))
# Error: Attempting to use uninitialized value foo
```

Of course, we can use the graph in more than one session, we just have to initialize the variables again. The values in the new session will be completely independent from the first one:

```
with tf.Session(graph=graph) as sess:
  sess.run(initialize)
  print(sess.run(variable))
# Output: 42
```

Hopefully this short workthrough helped you to better understand `tf.Session`. Feel free to ask questions in the comments.

Updated 2017-07-12: The operation to initialize variables changed in TensorFlow 1.0.

You can use this post under the open CC BY-SA 3.0 license and cite it as:

```
@misc{hafner2016tfsession,
  author = {Hafner, Danijar},
  title = {What is a TensorFlow Session?},
  year = {2016},
  howpublished = {Blog post},
  url = {https://danijar.com/what-is-a-tensorflow-session/}
}
```

30 Comments **Danijar Hafner** 1 **Login**

Recommend 44 Tweet Share **Sort by Best**

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

imtithal • 3 years ago

very useful short lesson

2 | • Reply • Share ›

Govindam Agrawal • 4 years ago

Can't we print the "variable" directly simply by print(variable) as the sess.run(initialize) already assign value 42 to it?

1 | • Reply • Share ›

Danijar Mod Govindam Agrawal • 4 years ago

No, the value of the `tf.Variable` is never stored in the Python variable (`variable`). The Python variable is just a reference to the TensorFlow graph. And the session holds the values of elements in the graph. We need to ask the session for the value and provide the Python variable (`sess.run(variable)`) so that it knows which graph element's value we want to read.

3 ^ | ⌄ • Reply • Share ›

santanu • 4 years ago • edited

Excellent. Very insightful.

What is the key purpose of having such a programming model? I am sure some deep thinking is going on there. Is it about decoupling the model from the runtime? or is it for allowing parallel processing or trying to mimic spark RDDs and Operations?

1 ^ | ⌄ • Reply • Share ›

Danijar Mod ➦ santanu • 4 years ago

It's like defining your program (compute graph) and running it (session). Specifying the graph beforehand allows for optimizations and distributed execution. When running on a cluster, TensorFlow adds send and receive operations into the graph to automatically handle communication for you. The graph then gets sent to all workers before the first operation gets executed (as part of the first `sess.run(...)` call).

1 ^ | ⌄ • Reply • Share ›

Rafael Casio • 9 months ago

dsdss

^ | ⌄ • Reply • Share ›

MPROXX • a year ago

Great tutorial! I have a question: am I only allowed to sess.run(op1) at one place or
can I also sess.run(op1) and sess.run(op2) at a later point? Or is that bad? There is no tutorial really mentioning that.

^ | ⌄ • Reply • Share ›

Danijar Mod ➦ MPROXX • a year ago

Yes, you can run as many different ops in the session as you like. Each sess.run() call has some overhead, so ideally you do fewer large calls rather than more small ones. Also, I'd recommend using TF 2 which is quite stable now, where sessions are no longer needed.

^ | ⌄ • Reply • Share ›

Malhar Jajoo • a year ago

The bit on default graph seems a little incomplete.

Reply • Share ›

Yunsoo Jung • 2 years ago

Thank you for nice summary.

Reply • Share ›

akshat dashore • 2 years ago

what should i do if i use pretrained model and got error ==> Error running session: Invalid argument: Session was not created with a graph before Run()!

Reply • Share ›

Vijay Ram • 2 years ago

Nice summary

Reply • Share ›

Tianxiao Jiang • 3 years ago

Nice summary ~

Reply • Share ›

Əli Nuraliyev • 3 years ago

What is Weight and Bias, could you explain it thoroughly

Reply • Share ›

Əli Nuraliyev • 3 years ago

```
import tensorflow as tf
state=tf.Variable(0,name='counter')
one=tf.constant(1)
new_value=tf.add(state,one)
update=tf.assign(state,new_value)
init=tf.global_variables_initializer()
with tf.Session() as sess:
sess.run(init)
for _ in range(3):
sess.run(update)
print(sess.run(state))
```

The code that I stated above works, but my question is that why we dont need to write sess.run(new_value) before the sess.run(update).

Reply • Share ›

Danijar Mod ➦ Əli Nuraliyev • 3 years ago

`new_value` is a dependency of `update` because it's value is used to the assign. When you call `sess.run()` of a tensor or operation, all its dependencies are executed as well in order to compute the result.

1 ^ | ⌄ • Reply • Share ›

Cun Mu • 3 years ago

thanks for the clear explanation!

^ | ⌄ • Reply • Share ›

boray tek • 3 years ago • edited

Hi,

I know your explanation would run in the same scope but my question is how to get active running session in another scope.

lets say...

you build the graph in one scope

def method_build_the_graph():

....

an_intermediate_tensor= tf.random('an_intermediate_tensor'...)

...
return output_tensor

def
callback_intermediate_tensor_to_a_file_after_every_batch(batch
logs):

an_intermediate_tensor=tf.get_variable('an_intermediate_tensor
tf.train.saver(get_the_session_where_tensor_was_initialized,
an_intermediate_tensor)

How to get the session where the tensor was initialized?

^ | ⌄ • Reply • Share ›

Danijar Mod ➦ boray tek • 3 years ago

Sorry, but I don't understand this question. A tensor is defined in exactly one graph and can be initialized in any number of sessions for that graph. Typically you only use one graph and one session.

^ | ⌄ • Reply • Share ›

boray tek ➦ Danijar • 3 years ago • edited

I am sorry for the clutter in my question.

I create the graph in one method. Then I train it

"model.compile and model.fit_generator".
I think model.fit_generator initializes the variables and runs the session. I am wondering if it is possible to get a handle to active running session that is probably executed in fit_generator.

Perhaps, still asking the wrong question. But my problem is that I wrote a callback which is supposed to dump and intermediate tensor value to a file. But that requires a session handle, which I can't have because keras wraps the initialization and the session.

Reply • Share ›

Danijar Mod boray tek • 3 years ago

No worries. Unfortunately, I don't know how Keras manages the session. However, a quick Google search brought up `tf.keras.backend.get_session()`, maybe that helps you.

Reply • Share ›

Yang Li • 3 years ago

Very good and concise explanation! It helped me get a better understanding of TensorFlow, thank you!

Reply • Share ›

Meziane HADJADJ • 4 years ago

Simple, short and very useful!!

Reply • Share ›

Fong Way • 4 years ago

Great explanation. I had encounter one problem in tensorflow. Is it possible to reuse the variable value (which had been trained) in a new session? Thank you

Reply • Share ›

Paul Gradie Fong Way • 4 years ago

Yeah totally! You can use the tf.train.Saver() class to save trained variables to a file. This class has '.save' and '.restore' methods.

For example:

# The file path to save the data
save_file = './model.ckpt'

```
#Class used to do the saving
saver = tf.train.Saver()

###Start session and train model/varibles###
with tf.Session() as sess:
...#train model
...#Save the model (and its variables) using the save
method
...saver.save(sess, save_file)
```

:D You can then load them back up using the load method.

```
#start session
with tf.Session() as sess:
...#load previous session
...saver.restore(sess, save_file)
```

1 ^ | ⌄ • Reply • Share ›

Taotao Li • 4 years ago • edited

I find that some modern frameworks are designed from the same way, say tensorflow and spark, they both have DAG, Distributed, Hardware-portable, lazy executing, seperate define from execution, and so on. that's very nice.