Project: tensorflow

2017-03-08T10:06:12Z doncat99 102comments

ValueError: Attempt to reuse RNNCell with a different variable scope than its first use.

Web url: https://github.com/tensorflow/tensorflow/issues/8191

API url: https://api.github.com/repos/tensorflow/tensorflow/issues/8191

I am not sure if I am the first who met the following error:

ValueError: Attempt to reuse RNNCell <tensorflow.contrib.rnn.python.ops.core\_rnn\_cell\_impl.BasicLSTMCell object at 0x10210d5c0> with a different variable scope than its first use. First use of cell was with scope 'rnn/multi\_rnn\_cell/cell\_0/basic\_lstm\_cell', this attempt is with scope 'rnn/multi\_rnn\_cell/cell\_1/basic\_lstm\_cell'. Please create a new instance of the cell if you would like it to use a different set of weights. If before you were using: MultiRNNCell([BasicLSTMCell(...)] \* num\_layers), change to: MultiRNNCell([BasicLSTMCell(...) for \_ in range(num\_layers)]). If before you were using the same cell instance as both the forward and reverse cell of a bidirectional RNN, simply create two instances (one for forward, one for reverse). In May 2017, we will start transitioning this cell's behavior to use existing stored weights, if any, when it is called with scope=None (which can lead to silent model degradation, so this error will remain until then.)

with the code fragment:

import tensorflow as tf

from tensorflow.contrib import rnn

hidden\_size = 100

batch\_size = 100

num\_steps = 100

num\_layers = 100

is\_training = True

keep\_prob = 0.4

input\_data = tf.placeholder(tf.float32, [batch\_size, num\_steps])

lstm\_cell = rnn.BasicLSTMCell(hidden\_size, forget\_bias=0.0, state\_is\_tuple=True)

if is\_training and keep\_prob < 1:

lstm\_cell = rnn.DropoutWrapper(lstm\_cell)

cell = rnn.MultiRNNCell([lstm\_cell for \_ in range(num\_layers)], state\_is\_tuple=True)

\_initial\_state = cell.zero\_state(batch\_size, tf.float32)

iw = tf.get\_variable("input\_w", [1, hidden\_size])

ib = tf.get\_variable("input\_b", [hidden\_size])

inputs = [tf.nn.xw\_plus\_b(i\_, iw, ib) for i\_ in tf.split(input\_data, num\_steps, 1)]

if is\_training and keep\_prob < 1:

inputs = [tf.nn.dropout(input\_, keep\_prob) for input\_ in inputs]

outputs, states = rnn.static\_rnn(cell, inputs, initial\_state=\_initial\_state)

I had googled around with no luck, can anyone show me a way out?

-------------------------------------------------------------------------

2017-03-08T11:20:20Z Wojova

I am getting the same error when trying to run the translate example (even when doing the small self test) which can be found here: https://github.com/tensorflow/models/tree/master/tutorials/rnn/translate

-------------------------------------------------------------------------

2017-03-08T16:35:15Z tongda

I met the same issue. If you are all using compiled version on master branch, I believe that we are the same issue caused by the [recent commit](https://github.com/tensorflow/tensorflow/commit/54d50ffec8df4f748694632dbe5ebde9971e2c9e). As the commit message says:

> Make all RNNCells in tf.contrib.rnn act like tf.layers Layers, but with stricter semantics for no

w:

>

> 1. Upon first use of \_\_call\_\_, the used scope is stored in the cell. The RNNCell tries to create weights in that scope but if some are already set, an error is raised unless the RNNCell was constructed with argument reuse=True.

>

> 2. A subsequent use of \_\_call\_\_ of the same cell instance must be in the same scope.

> If it is not, an error is raised.

From my case, which is running the [ptb tutorial](https://github.com/tensorflow/models/tree/master/tutorials/rnn/ptb), the solution is just to add a parameter named with `reuse` like this at line 112:

def lstm\_cell():

return tf.contrib.rnn.BasicLSTMCell(

size, forget\_bias=0.0, state\_is\_tuple=True, reuse=tf.get\_variable\_scope().reuse)

Then it works.

-------------------------------------------------------------------------

2017-03-08T17:37:12Z prb12

@ebrevdo Could you please take a look at this?

-------------------------------------------------------------------------

2017-03-08T23:56:29Z tomwanzek

The issue replicates for me when using the Windows/GPU build 105 on the [Shakespeare RNN Repo](https://github.com/martin-gorner/tensorflow-rnn-shakespeare).

When running the code with the Win 1.0.0/GPU Release, there is no issue.

-------------------------------------------------------------------------

2017-03-09T01:03:35Z ebrevdo

That repo looks like it's targeted at tf 1.0, not intermediate releases.

On Mar 8, 2017 3:56 PM, "Tom Wanzek" <notifications@github.com> wrote:

> The issue replicates for me when using the Windows/GPU build 105 on the Shakespeare

> RNN Repo <https://github.com/martin-gorner/tensorflow-rnn-shakespeare>.

>

> When running the code with the Win 1.0.0/GPU Release, there is no issue.

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-285209555>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim5ansaL1KN51T4nCaqLnqw2QHN4Wks5rj0BBgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-03-09T02:58:15Z doncat99

@tongda , I am using the Release Version of Tensorflow 1.0, working on MacOS in cpu mode. I will switch to the master branch to see if it work by adding the "reuse" parameter, thanks.

-------------------------------------------------------------------------

2017-03-09T03:34:20Z ebrevdo

doncat99: if you do, please ensure your code queries the tensorflow version

and raises a flag if the version is lower than the master branch version.

you may need to check against:

from tensorflow.core import versions

versions.GIT\_VERSION

On Wed, Mar 8, 2017 at 6:58 PM, doncat99 <notifications@github.com> wrote:

> @tongda <https://github.com/tongda> , I am using the Release Version of

> Tensorflow 1.0, working on MacOS in cpu mode. I will switch to the master

> branch to see if it work by adding the "reuse" parameter, thanks.

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-285240438>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim66cU9e16lgD-2D0QLGcQCiHbV0zks5rj2rbgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-03-10T15:43:51Z tomwanzek

@ebrevdo So what would be the suggested changes to the Shakepeare RNN to allow it to work with the intermediate stable release?

Here is the key architectural section of the code, which now fails with build#105:

```python

#

# the model (see FAQ in README.md)

#

lr = tf.placeholder(tf.float32, name='lr') # learning rate

pkeep = tf.placeholder(tf.float32, name='pkeep') # dropout parameter

batchsize = tf.placeholder(tf.int32, name='batchsize')

# inputs

X = tf.placeholder(tf.uint8, [None, None], name='X') # [ BATCHSIZE, SEQLEN ]

Xo = tf.one\_hot(X, ALPHASIZE, 1.0, 0.0) # [ BATCHSIZE, SEQLEN, ALPHASIZE ]

# expected outputs = same sequence shifted by 1 since we are trying to predict the next character

Y\_ = tf.placeholder(tf.uint8, [None, None], name='Y\_') # [ BATCHSIZE, SEQLEN ]

Yo\_ = tf.one\_hot(Y\_, ALPHASIZE, 1.0, 0.0) # [ BATCHSIZE, SEQLEN, ALPHASIZE ]

# input state

Hin = tf.placeholder(tf.float32, [None, INTERNALSIZE\*NLAYERS], name='Hin') # [ BATCHSIZE, INTERNALSIZE \* NLAYERS]

# using a NLAYERS=3 layers of GRU cells, unrolled SEQLEN=30 times

# dynamic\_rnn infers SEQLEN from the size of the inputs Xo

onecell = rnn.GRUCell(INTERNALSIZE)

dropcell = rnn.DropoutWrapper(onecell, input\_keep\_prob=pkeep)

multicell = rnn.MultiRNNCell([dropcell for \_ in range(NLAYERS)], state\_is\_tuple=False)

multicell = rnn.DropoutWrapper(multicell, output\_keep\_prob=pkeep)

Yr, H = tf.nn.dynamic\_rnn(multicell, Xo, dtype=tf.float32, initial\_state=Hin)

# Yr: [ BATCHSIZE, SEQLEN, INTERNALSIZE ]

# H: [ BATCHSIZE, INTERNALSIZE\*NLAYERS ] # this is the last state in the sequence

```

I do not seem to find any documentation regarding a `reuse` flag?

Thanks in advance.

-------------------------------------------------------------------------

2017-03-15T05:51:03Z ebrevdo

Use:

multicell = rnn.MultiRNNCell([rnn.DropoutWrapper(rnn.GRUCell(INTERNALSIZE),

input\_keep\_prob=pkeep) for \_ in range(NLAYERS)], state\_is\_tuple=False)

Which creates a separate grucell object for each layer.

On Mar 10, 2017 7:44 AM, "Tom Wanzek" <notifications@github.com> wrote:

> @ebrevdo <https://github.com/ebrevdo> So what would be the suggested

> changes to the Shakepeare RNN to allow it to work with the intermediate

> stable release?

>

> Here is the key architectural section of the code, which now fails with

> build#105:

>

> ## the model (see FAQ in README.md)#

> lr = tf.placeholder(tf.float32, name='lr') # learning rate

> pkeep = tf.placeholder(tf.float32, name='pkeep') # dropout parameter

> batchsize = tf.placeholder(tf.int32, name='batchsize')

> # inputs

> X = tf.placeholder(tf.uint8, [None, None], name='X') # [ BATCHSIZE, SEQLEN ]

> Xo = tf.one\_hot(X, ALPHASIZE, 1.0, 0.0) # [ BATCHSIZE, SEQLEN, ALPHASIZE ]# expected outputs = same sequence shifted by 1 since we are trying to predict the next character

> Y\_ = tf.placeholder(tf.uint8, [None, None], name='Y\_') # [ BATCHSIZE, SEQLEN ]

> Yo\_ = tf.one\_hot(Y\_, ALPHASIZE, 1.0, 0.0) # [ BATCHSIZE, SEQLEN, ALPHASIZE ]# input state

> Hin = tf.placeholder(tf.float32, [None, INTERNALSIZE\*NLAYERS], name='Hin') # [ BATCHSIZE, INTERNALSIZE \* NLAYERS]

> # using a NLAYERS=3 layers of GRU cells, unrolled SEQLEN=30 times# dynamic\_rnn infers SEQLEN from the size of the inputs Xo

>

> onecell = rnn.GRUCell(INTERNALSIZE)

> dropcell = rnn.DropoutWrapper(onecell, input\_keep\_prob=pkeep)

> multicell = rnn.MultiRNNCell([dropcell for \_ in range(NLAYERS)], state\_is\_tuple=False)

> multicell = rnn.DropoutWrapper(multicell, output\_keep\_prob=pkeep)

> Yr, H = tf.nn.dynamic\_rnn(multicell, Xo, dtype=tf.float32, initial\_state=Hin)# Yr: [ BATCHSIZE, SEQLEN, INTERNALSIZE ]# H: [ BATCHSIZE, INTERNALSIZE\*NLAYERS ] # this is the last state in the sequence

>

> I do not seem to find any documentation regarding a reuse flag?

>

> Thanks in advance.

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-285702372>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim6MOOCbx3RJEJe8PQBDXGVIXTGPmks5rkW\_jgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-03-15T14:27:30Z BSVogler

I don't understand why I am getting this error with the [seq2seq tutorial model](https://github.com/tensorflow/models/blob/master/tutorials/rnn/translate/seq2seq\_model.py):

```python

cell = tf.contrib.rnn.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

```

[Source](https://github.com/tensorflow/models/blob/master/tutorials/rnn/translate/seq2seq\_model.py#L129)

where the cell is created with

```python

def single\_cell():

return tf.contrib.rnn.GRUCell(size)

```

-------------------------------------------------------------------------

2017-03-15T16:15:09Z tomwanzek

@ebrevdo Thanks for getting back to this issue. Unfortunately, the suggested change leaves matters as they are, with the aforementioned error. Given the above comment regarding the \*\*seq2seq tutorial\*\*, I suspect we are all in the same boat?

-------------------------------------------------------------------------

2017-03-16T04:25:17Z ebrevdo

Are you sure it's the exact same error? Please copy and paste it here.

-------------------------------------------------------------------------

2017-03-16T13:30:24Z tomwanzek

My bad, I just went through the change process to the relevant code again (from scratch) and re-ran it as proposed. The error has indeed been removed and the Old Bard is hallucinating just fine now üëç

So, thx, not sure where I went wrong yesterday, but it was clearly on me.

-------------------------------------------------------------------------

2017-03-17T03:29:06Z bingfengyiren

I met the same problem when using the Release Version of Tensorflow 1.0 and working on MacOS in cpu mode.Even if add the "reuse" parameter

```

def cell():

return tf.contrib.rnn.BasicLSTMCell(rnn\_size,state\_is\_tuple=True,reuse=tf.get\_variable\_scope().reuse)

muticell = tf.contrib.rnn.MultiRNNCell([cell for \_ in range(num\_layers)], state\_is\_tuple=True)

```

-------------------------------------------------------------------------

2017-03-17T05:28:33Z ebrevdo

your multicell looks wrong... you should be using "cell() for \_ in

range(...)"

On Thu, Mar 16, 2017 at 8:29 PM, cuiming <notifications@github.com> wrote:

> I met the same problem when using the Release Version of Tensorflow 1.0

> and working on MacOS in cpu mode.Even if add the "reuse" parameter

>

> def cell():

> return tf.contrib.rnn.BasicLSTMCell(rnn\_size,state\_is\_tuple=True,reuse=tf.get\_variable\_scope().reuse)

>

> muticell = tf.contrib.rnn.MultiRNNCell([cell for \_ in range(num\_layers)], state\_is\_tuple=True)

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-287257629>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim3A6JQr8ptRKrdiDW\_kgNRIFkHGlks5rmf4WgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-03-26T18:15:16Z bowu

I was trying to run the translate example: python2.7 translate.py --data\_dir data/ --train\_dir train/ --size=256 --num\_layers=2 --steps\_per\_checkpoint=50

It seems the way to use MultiRNNCell is correct:

cell = tf.contrib.rnn.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

But I got the same error:

ValueError: Attempt to reuse RNNCell <tensorflow.contrib.rnn.python.ops.core\_rnn\_cell\_impl.GRUCell object at 0x7fba0683de90> with a different variable scope than its first use. First use of cell was with scope 'embedding\_attention\_seq2seq/embedding\_attention\_decoder/attention\_decoder/multi\_rnn\_cell/cell\_0/gru\_cell', this attempt is with scope 'embedding\_attention\_seq2seq/rnn/multi\_rnn\_cell/cell\_0/gru\_cell'. Please create a new instance of the cell if you would like it to use a different set of weights. If before you were using: MultiRNNCell([GRUCell(...)] \* num\_layers), change to: MultiRNNCell([GRUCell(...) for \_ in range(num\_layers)]). If before you were using the same cell instance as both the forward and reverse cell of a bidirectional RNN, simply create two instances (one for forward, one for reverse). In May 2017, we will start transitioning this cell's behavior to use existing stored weights, if any, when it is called with scope=None (which can lead to silent model degradation, so this error will remain until then.)

-------------------------------------------------------------------------

2017-03-29T17:21:44Z robmsylvester

@bowu - did you have any luck with this? if you haven't tried it yet, reinstall tensorflow from the latest source. there were some changes to some of the core\_rnn files, among a few others. works for me now.

-------------------------------------------------------------------------

2017-03-30T03:05:00Z oxwsds

@robmsylvester I reinstall tensorflow from the latest source, still the same error. I was on branch master and the latest commit is `commit 2a4811054a9e6b83e1f5a2705a92aab50e151b13`. What's the latest commit when you build your repo?

-------------------------------------------------------------------------

2017-04-03T21:46:37Z prashantserai

Hi, I am using Tensorflow r1.0 using GPU built using source. I am trying to follow the unmodified Seq2Seq translation tutorial, but I'm getting the same error. i.e.

> ValueError: Attempt to reuse RNNCell <tensorflow.contrib.rnn.python.ops.core\_rnn\_cell\_impl.GRUCell object at 0x7f0fb51ebb00> with a different variable scope than its first use. First use of cell was with scope 'embedding\_attention\_seq2seq/embedding\_attention\_decoder/attention\_decoder/multi\_rnn\_cell/cell\_0/gru\_cell', this attempt is with scope 'embedding\_attention\_seq2seq/rnn/multi\_rnn\_cell/cell\_0/gru\_cell'.....

The relevant portion of the code in my seq2seq\_model.py is:

```

# Create the internal multi-layer cell for our RNN.

def single\_cell():

return tf.contrib.rnn.GRUCell(size)

if use\_lstm:

def single\_cell():

return tf.contrib.rnn.BasicLSTMCell(size)

cell = single\_cell()

if num\_layers > 1:

cell = tf.contrib.rnn.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

```

What can I do to solve the problem?

adding "reuse=tf.get\_variable\_scope().reuse" to the call where the GRUCell is created doesn't help.

Thanks a ton!

-------------------------------------------------------------------------

2017-04-03T23:18:43Z robmsylvester

@prashantserai - see what happens if you remove the MultiRNNCell line from above, effectively making your network just one layer. Does it work then? It might be a bug somewhere in MultiRNNCell. I've read about that somewhere recently, probably on stack overflow.

If you implement the stacked lstm/gru yourself, you don't get this error, and you can implement the same functionality (actually more, because you're free to do whatever you want with bidirectional architectures, weird residual and skip connections, etc.)

-------------------------------------------------------------------------

2017-04-04T13:43:33Z prashantserai

@robmsylvester The same error persisted even when I tried with num\_layers=1 which should effectively skip that line. Any other ideas? Thanks for the input.

-------------------------------------------------------------------------

2017-04-04T17:59:13Z robmsylvester

Hmmm. One thing that stands out to me is in the referenced legacy seq2seq file:

`encoder\_cell = copy.deepcopy(cell)`

This line appears to be used because the same architecture is used on both the encoder and decoder side. They make a copy of the cell, then pass the cell argument along to the attention decoder embedding function, then to the attention decoder itself.

What happens if you explicitly create the encoder cell AND the decoder cell in your seq2seq model file and pass both along to the legacy library file, making the small adjustments to the functions and their arguments?

-------------------------------------------------------------------------

2017-04-04T18:30:33Z iamgroot42

@robmsylvester shouldn't making changes in the scopes of the cells work? It's working for the other two examples as well. In my opinion, this would be a very ugly workaround; a cleaner solution must exist; maybe we are missing something? ( I got the same error on the seq2seq tutorial as well, tried all of the above solutions).

-------------------------------------------------------------------------

2017-04-04T18:38:34Z robmsylvester

@iamgroot42 - Yeah, that 'solution' is admittedly very ugly, but more so just trying to hunt down where an issue might be. I'll play with it in a few hours and see if I can track something down.

-------------------------------------------------------------------------

2017-04-05T23:16:55Z ebrevdo

In fact, the copy.deepcopy is there because these are legacy functions and

we don't have the resources to maintain/update them. If you'd like to

introduce a backwards-compatible change that allows the user to provide a

second cell for the decoding step, and if it's None then to fallback on the

deepcopy, then I would be happy to review the PR. Keep in mind it would

have to be a backwards compatible change.

On Tue, Apr 4, 2017 at 11:38 AM, Rob Sylvester <notifications@github.com>

wrote:

> @iamgroot42 <https://github.com/iamgroot42> - Yeah, that 'solution' is

> admittedly very ugly, but more so just trying to hunt down where an issue

> might be. I'll play with it in a few hours and see if I can track something

> down.

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-291593289>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim1QHTDhOC\_zT6cKtmUFPOit5Yjn7ks5rso5CgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-04-05T23:32:50Z robmsylvester

@ebrevdo - I'll think about it. I do have a translator that works pretty similar to this one but creates cells through a separate class that allows for inserting bidirectional layers where you want, residuals where you want, merging inputs with concat vs. sum, and a few other things. I think I could migrate my class over to this tutorial pretty easily by using static RNN's. I'll let you know.

-------------------------------------------------------------------------

2017-04-06T00:32:18Z prashantserai

@ebrevdo i am running Tensorflow r1.0 (tensorflow-1.0.1-cp36-cp36m-linux\_x86\_64) on Red Hat and have the latest version of the translation tutorial from Github.. is there a way you know to make this work currently?

-------------------------------------------------------------------------

2017-04-06T04:08:26Z ebrevdo

It's unfortunate that the translation tutorial does not work with TF 1.0. We should fix that. @lukaszkaiser can you take a look? We're working on a new tutorial but it's still a few weeks off and will require a nightly version of TensorFlow (or TF 1.1 or 1.2) to work.

-------------------------------------------------------------------------

2017-04-06T04:09:40Z ebrevdo

(lukasz; it's hard for me to identify from the various comments which part of the tutorial is faulty in TF 1.0. any chance you could identify the line and i can help get it working?)

-------------------------------------------------------------------------

2017-04-06T04:56:36Z iamgroot42

@ebrevdo It's [this ](https://github.com/tensorflow/models/tree/master/tutorials/rnn/translate) tutorial. The error is in [this](https://github.com/tensorflow/models/blob/master/tutorials/rnn/translate/seq2seq\_model.py#L122) cluster of lines. The cells passed here are used for both the backward and forward phase of the legacy seq2seq model, which throws an error because of same cells being used with different scopes.

-------------------------------------------------------------------------

2017-04-06T05:01:35Z lukaszkaiser

@iamgroot42 do you want to make a PR with the needed changes? That would be great, I currently don't have the cycles to do that myself. Thanks!

-------------------------------------------------------------------------

2017-04-06T13:11:53Z oxwsds

I noticed that the TF 1.0 works fine with the newest version of translation tutorial if compiled from the source on branch remotes/origin/r1.0

```

$ git clone https://github.com/tensorflow/tensorflow

$ cd tensorflow

$ git checkout remotes/origin/r1.0

```

then build and install TensorFlow, it works fine.

On branch remotes/origin/r1.1 it has the "different variable scope" error.

I modified the code as @robmsylvester suggested

> What happens if you explicitly create the encoder cell AND the decoder cell in your seq2seq model file and pass both along to the legacy library file, making the small adjustments to the functions and their arguments?

and it works for me now.

-------------------------------------------------------------------------

2017-04-08T14:00:37Z prashantserai

@oxwsds the Tensorflow I'm using is 1.0.1 so maybe that's having an error..

I had tried what @robmsylvester suggested then actually.. and the training had begun (2 days 13 hours done now).. it fails during decoding though with the error:

```

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 883, in embedding\_attention\_seq2seq

initial\_state\_attention=initial\_state\_attention)

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 787, in embedding\_attention\_decoder

initial\_state\_attention=initial\_state\_attention)

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 686, in attention\_decoder

cell\_output, state = cell(x, state)

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/rnn/python/ops/core\_rnn\_cell\_impl.py", line 796, in \_\_call\_\_

% (len(self.state\_size), state))

ValueError: Expected state to be a tuple of length 3, but received: Tensor("model\_with\_buckets/embedding\_attention\_seq2seq/rnn/gru\_cell\_4/add:0", shape=(?, 1024), dtype=float32)

```

Did you try decoding?

-------------------------------------------------------------------------

2017-04-10T08:20:50Z oxwsds

@prashantserai Don't exactly know, but what you met seems to be another issue.

-------------------------------------------------------------------------

2017-04-10T09:00:44Z robmsylvester

@prashantserai If it fails only when you decode, perhaps it has something to do with using a batch size of one? Does the model still train if you lower the batch size to one during training?

-------------------------------------------------------------------------

2017-04-10T15:30:41Z soloice

@bowu Same error here. Mac OX Sierra, TensorFlow 1.1.0-rc1, Python 2.7.10 & Python 3.6.1.

-------------------------------------------------------------------------

2017-04-10T15:38:20Z prashantserai

@robmsylvester it did train successfully with a batch size of one too, but failed during decoding in the same way or similar way.. here's a full traceback.. the reason I was thinking of this as a connected error was because of the reference to seq2seq\_f (which was one of the modified functions) (the #prashant comment from my code to signify a modified line is part of the trace)

```

2017-04-10 11:32:27.447042: I tensorflow/core/common\_runtime/gpu/gpu\_device.cc:887] Found device 0 with properties:

name: GeForce GTX 780 Ti

major: 3 minor: 5 memoryClockRate (GHz) 0.928

pciBusID 0000:42:00.0

Total memory: 2.95GiB

Free memory: 2.88GiB

2017-04-10 11:32:27.447094: I tensorflow/core/common\_runtime/gpu/gpu\_device.cc:908] DMA: 0

2017-04-10 11:32:27.447102: I tensorflow/core/common\_runtime/gpu/gpu\_device.cc:918] 0: Y

2017-04-10 11:32:27.447118: I tensorflow/core/common\_runtime/gpu/gpu\_device.cc:977] Creating TensorFlow device (/gpu:0) -> (device: 0, name: GeForce GTX 780 Ti, pci bus id: 0000:42:00.0)

Traceback (most recent call last):

File "translate.py", line 322, in <module>

tf.app.run()

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/python/platform/app.py", line 48, in run

\_sys.exit(main(\_sys.argv[:1] + flags\_passthrough))

File "translate.py", line 317, in main

decode()

File "translate.py", line 248, in decode

model = create\_model(sess, True)

File "translate.py", line 136, in create\_model

dtype=dtype)

File "/data/data6/scratch/serai/models/tutorials/rnn/translate/seq2seq\_model.py", line 168, in \_\_init\_\_

softmax\_loss\_function=softmax\_loss\_function)

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 1203, in model\_with\_buckets

decoder\_inputs[:bucket[1]])

File "/data/data6/scratch/serai/models/tutorials/rnn/translate/seq2seq\_model.py", line 167, in <lambda>

self.target\_weights, buckets, lambda x, y: seq2seq\_f(x, y, True),

File "/data/data6/scratch/serai/models/tutorials/rnn/translate/seq2seq\_model.py", line 144, in seq2seq\_f

dtype=dtype) #prashant

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 883, in embedding\_attention\_seq2seq

initial\_state\_attention=initial\_state\_attention)

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 787, in embedding\_attention\_decoder

initial\_state\_attention=initial\_state\_attention)

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 686, in attention\_decoder

cell\_output, state = cell(x, state)

File "/homes/3/serai/.conda/envs/tensorflow\_r1.0\_gpu/lib/python3.6/site-packages/tensorflow/contrib/rnn/python/ops/core\_rnn\_cell\_impl.py", line 796, in \_\_call\_\_

% (len(self.state\_size), state))

ValueError: Expected state to be a tuple of length 3, but received: Tensor("model\_with\_buckets/embedding\_attention\_seq2seq/rnn/gru\_cell\_4/add:0", shape=(?, 1024), dtype=float32)

```

@oxwsds does your opinion change on the basis of the full trace above?

-------------------------------------------------------------------------

2017-04-11T02:56:25Z oxwsds

@prashantserai I tried decoding and it works fine. I just simply add a `encoder\_cell` arg to function `tf.contrib.legacy\_seq2seq.embedding\_attention\_seq2seq` and in `translate/seq2seq\_model.py` create the cell and pass it to the function, which were called in function `seq2seq\_f`. How did you change your code?

-------------------------------------------------------------------------

2017-04-11T04:04:23Z prashantserai

@oxwsds @robmsylvester @ebrevdo

I finally have something that's working now (I mean, results for my single layer 256 unit network are kind of appalling, but that's probably just because the network is ultra light weight and I didn't tune params AT ALL)

Thank you so much everyone...!!!!!

\_Here's my thoughts at the end of this:\_

@oxwsds comment \*\*that the tutorial (in it's current form) works without any need for modification when Tensorflow is compiled from the branch remotes/origin/r1.0 was TRUE\*\*. Although, the sad bit was that the version of Tensorflow I had for which modifications within Tensorflow code were needed, and the version in remotes/origin/r1.0 were both identically labelled.

@robmsylvester 's fix in the comment (copied below) DID WORK for my version of Tensorflow where the Tutorial didn't work out of the box (and should work for TF 1.1 too I guess). It is slightly messy to implement, but I could do it, which is saying something :-P

The error in my last two comments before this was due to my mistake. Like a dummy, I was specifying the layers and hidden units parameters only during training, I was leaving the code to use defaults during decoding. \*\*(this portion of the tutorial could be slightly more dummy proof: https://www.tensorflow.org/tutorials/seq2seq#lets\_run\_it )\*\*

> Hmmm. One thing that stands out to me is in the referenced legacy seq2seq file:

>

> encoder\_cell = copy.deepcopy(cell)

>

> This line appears to be used because the same architecture is used on both the encoder and decoder side. They make a copy of the cell, then pass the cell argument along to the attention decoder embedding function, then to the attention decoder itself.

>

> What happens if you explicitly create the encoder cell AND the decoder cell in your seq2seq model file and pass both along to the legacy library file, making the small adjustments to the functions and their arguments?

-------------------------------------------------------------------------

2017-04-11T18:46:33Z ebrevdo

Thanks for the feedback! Seems there's something different between the TF

on pypi and at that tag? Gunhan, is that possible?

On Mon, Apr 10, 2017 at 9:05 PM, prashantserai <notifications@github.com>

wrote:

> @oxwsds <https://github.com/oxwsds> @robmsylvester

> <https://github.com/robmsylvester> @ebrevdo <https://github.com/ebrevdo>

> I finally have something that's working now (I mean, results for my single

> layer 256 unit network are kind of appalling, but that's probably just

> because the network is ultra light weight and I didn't tune params AT ALL)

>

> Here's my bottomline:

>

> @oxwsds <https://github.com/oxwsds> comment \*that the tutorial (in it's

> current form) works without any need for modification when Tensorflow is

> compiled from the branch remotes/origin/r1.0 was TRUE\*. The sad bit

> although being that the version of Tensorflow I had for which modifications

> within Tensorflow code were needed, and the version in remotes/origin/r1.0

> were both identically labelled.

>

> @robmsylvester <https://github.com/robmsylvester> 's fix in the comment

> (copied below) DID WORK for my version of Tensorflow where the Tutorial

> didn't work out of the box (and should work for TF 1.1 too I guess). It is

> slightly messy to implement, but I could do it, which is saying something

> :-P

> The error in my last two comments before this was due to my mistake. Like

> a dummy, I was specifying the layers and hidden units parameters only

> during training, I was leaving the code to use defaults during decoding. \*(this

> portion of the tutorial is could be slightly more dummy proof:

> https://www.tensorflow.org/tutorials/seq2seq#lets\_run\_it

> <https://www.tensorflow.org/tutorials/seq2seq#lets\_run\_it> )\*

>

> Hmmm. One thing that stands out to me is in the referenced legacy seq2seq

> file:

>

> encoder\_cell = copy.deepcopy(cell)

>

> This line appears to be used because the same architecture is used on both

> the encoder and decoder side. They make a copy of the cell, then pass the

> cell argument along to the attention decoder embedding function, then to

> the attention decoder itself.

>

> What happens if you explicitly create the encoder cell AND the decoder

> cell in your seq2seq model file and pass both along to the legacy library

> file, making the small adjustments to the functions and their arguments?

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-293143828>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtimxvcfFnbWbpj7aUs3BUjwGEFj6p5ks5ruvvygaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-04-13T10:11:20Z pltrdy

For information I had this issue while trying to stack LSTM cells:

My orginial code was:

```

lstm\_cell = tf.nn.rnn\_cell.BasicLSTMCell(hidden\_size, forget\_bias=0.0, state\_is\_tuple=True)

if is\_training and keep\_prob < 1:

lstm\_cell = tf.nn.rnn\_cell.DropoutWrapper(

lstm\_cell, output\_keep\_prob=keep\_prob)

cell = tf.nn.rnn\_cell.MultiRNNCell([lstm\_cell] \* num\_layers, state\_is\_tuple=True)

```

Then, with the following code, creating the model was ok, but I couldn't share the variable with another model. (for instance if you create a train\_model and a valid\_model supposed to share tensors, it will fail)

```

lstm\_creator = lambda: tf.contrib.rnn.BasicLSTMCell(

hidden\_size,

forget\_bias=0.0, state\_is\_tuple=True)

if is\_training and keep\_prob < 1:

cell\_creator = lambda:tf.contrib.rnn.DropoutWrapper(

lstm\_creator(), output\_keep\_prob=keep\_prob)

else:

cell\_creator = lstm\_creator

cell = tf.contrib.rnn.MultiRNNCell([cell\_creator() for \_ in range(num\_layers)], state\_is\_tuple=True)

```

So finally I used `lstm\_creator` to be the function like `lstm\_cell` in [tensorflow/models/tutorials/rnn/ptb/ptb\_word\_lm.py#L112](https://github.com/tensorflow/models/blob/master/tutorials/rnn/ptb/ptb\_word\_lm.py#L112). I now have:

```

def lstm\_cell():

# With the latest TensorFlow source code (as of Mar 27, 2017),

# the BasicLSTMCell will need a reuse parameter which is unfortunately not

# defined in TensorFlow 1.0. To maintain backwards compatibility, we add

# an argument check here:

if 'reuse' in inspect.getargspec(

tf.contrib.rnn.BasicLSTMCell.\_\_init\_\_).args:

return tf.contrib.rnn.BasicLSTMCell(

size, forget\_bias=0.0, state\_is\_tuple=True,

reuse=tf.get\_variable\_scope().reuse)

else:

return tf.contrib.rnn.BasicLSTMCell(

size, forget\_bias=0.0, state\_is\_tuple=True)

attn\_cell = lstm\_cell

lstm\_creator = lstm\_cell

if is\_training and keep\_prob < 1:

cell\_creator = lambda:tf.contrib.rnn.DropoutWrapper(

lstm\_creator(), output\_keep\_prob=keep\_prob)

else:

cell\_creator = lstm\_creator

cell = tf.contrib.rnn.MultiRNNCell([cell\_creator() for \_ in range(num\_layers)], state\_is\_tuple=True)

```

It is now fully working

-------------------------------------------------------------------------

2017-04-16T21:42:34Z aep

trying to get this thing running, which results in the same error:

https://gist.github.com/danijar/c7ec9a30052127c7a1ad169eeb83f159#file-blog\_tensorflow\_sequence\_classification-py-L38

@pltrdy 's solution didn't do it for me oddly. I'm getting

```

ValueError: Variable rnn/multi\_rnn\_cell/cell\_0/basic\_lstm\_cell/weights does not exist, or was not created with tf.get\_variable(). Did you mean to set reuse=None in VarScope?

```

-------------------------------------------------------------------------

2017-04-18T09:48:42Z pltrdy

@aep did you use the function of https://github.com/tensorflow/models/blob/master/tutorials/rnn/ptb/ptb\_word\_lm.py#L112 I mention at the end of my post (now edited to be more clear)

-------------------------------------------------------------------------

2017-04-28T02:12:32Z Tshzzz

cells=[]

for \_ in range(15):

cell = create\_lstm\_cell(config)

cells.append(cell)

lsmt\_layers = rnn.MultiRNNCell(cells)

it solved my problem

-------------------------------------------------------------------------

2017-04-28T07:21:09Z dsoiM

Managed to fix this issue by installing older version of Tensorflow:

`pip install -Iv tensorflow==1.0`

I was receiving the error when executing the seq2seq tutorial

-------------------------------------------------------------------------

2017-05-01T15:06:56Z kyteague

In regards to what @ebrevdo said, I think the solution is not to fix the legacy seq2seq code, but to update the tutorial to use the `contrib.seq2seq` package instead, which is actively maintained. It is quite demoralizing when the first tensorflow program you ever run spits out a bunch of errors. If I have some time this week, I'll submit a PR.

-------------------------------------------------------------------------

2017-05-01T18:03:33Z ebrevdo

We're working on a new seq2seq tutorial. We had hoped to release by end of

last month but are getting delayed. It will use the new API.

On May 1, 2017 8:07 AM, "Kyle Teague" <notifications@github.com> wrote:

> In regards to what @ebrevdo <https://github.com/ebrevdo> said, I think

> the solution is not to fix the legacy seq2seq code, but to update the

> tutorial to use the contrib.seq2seq package instead, which is actively

> maintained. It is quite demoralizing when the first tensorflow program you

> ever run spits out a bunch of errors. If I have some time this week, I'll

> submit a PR.

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-298350307>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim587xZx9Gi4-yXmwccSum8\_Trc1oks5r1fUogaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-05-05T04:16:16Z njuzrs

@ebrevdo I meet the same error when running the sequence\_to\_sequence model on the tensorflow1.1 website. And I have try to use 'reuse' parameter but failed. Could you tell me when the new seq2seq tutorial will be released?

-------------------------------------------------------------------------

2017-05-05T14:47:33Z ebrevdo

Looks like at the same time as tf 1.2, since we will rely on some new

features of that release.

On May 4, 2017 9:16 PM, "njuzrs" <notifications@github.com> wrote:

> @ebrevdo <https://github.com/ebrevdo> I meet the same error when running

> the sequence\_to\_sequence model on the tensorflow1.1 website. And I have try

> to use 'reuse' parameter but failed. Could you tell me when the new seq2seq

> tutorial will be released?

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-299366774>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim8\_kFTM7-SsXQAA-Ar0dfhHMGT0Zks5r2qKngaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-05-08T08:40:15Z PratsBhatt

@ebrevdo I am as well facing the same issue and unable to progress with seq2seq. It will be really helpful if you could let us/me know what is a probable date for a new tutorial.

Thanks a lot for your help.

-------------------------------------------------------------------------

2017-05-09T04:10:11Z tanmayb123

Installing using `pip install tensorflow==1.0` (Tensorflow 1.0) is working for me (translate tutorial).

-------------------------------------------------------------------------

2017-05-09T06:40:42Z PratsBhatt

I have version 1.1.0-rc2.

-------------------------------------------------------------------------

2017-05-15T17:28:28Z mmmmming18

TF1.2 will solve this problem? Please help me how to continue training the model. TF 1.0 works but doesn't have devicewrapper api for multiple GPUs.

-------------------------------------------------------------------------

2017-05-19T18:21:42Z thomasqjohns

Having the same problem with tensor flow 1.1. Still working on a solution

-------------------------------------------------------------------------

2017-05-20T00:14:26Z jtubert

I tried several things, at the end I was able to use tensorflow 1.1 but had to make these changes: (based on Tshzzz above)

Remove this:

`multicell = rnn.MultiRNNCell([dropcell]\*NLAYERS, state\_is\_tuple=False)`

And add this:

cells=[]

for \_ in range(NLAYERS):

cell = rnn.DropoutWrapper(tf.contrib.rnn.GRUCell(INTERNALSIZE), input\_keep\_prob=pkeep)

cells.append(cell)

multicell = rnn.MultiRNNCell(cells, state\_is\_tuple=False)

-------------------------------------------------------------------------

2017-05-20T02:02:16Z prashantserai

@ebrevdo Congratulations, TF 1.2 just got released - was the new tutorial also released somewhere or is it being released anytime soon?

Thanks

-------------------------------------------------------------------------

2017-05-20T15:17:52Z ebrevdo

We'll plan to have an announcement when it's released. Working on it.

On May 19, 2017 7:02 PM, "prashantserai" <notifications@github.com> wrote:

> @ebrevdo <https://github.com/ebrevdo> Congratulations, TF 1.2 just got

> released - was the new tutorial also released somewhere or is it being

> released anytime soon?

>

> Thanks

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-302844002>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim0RWDzNCXk-bIjKSyHLvgFxUvq2lks5r7km7gaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-05-24T17:51:17Z ajaanbaahu

For anyone using tensorflow-gpu==1.1.0 and getting this error, switching to 1.0.0 via pip install tensorflow-gpu==1.0.0 is not going to fix the problem, at least didn't work for me.

I ran into this issue on both mac and ubuntu and compiling from source worked both times. So:

pip install https://storage.googleapis.com/tensorflow/linux/gpu/tensorflow\_gpu-1.0.0-cp34-cp34m-linux\_x86\_64.whl

-------------------------------------------------------------------------

2017-05-25T03:44:51Z mmmmming18

@ajaanbaahu Still waiting for tf1.2 new seq2seq tutorial.

-------------------------------------------------------------------------

2017-05-26T06:48:21Z saching270

It worked for me using `pip install tensorflow==1.0`.

-------------------------------------------------------------------------

2017-05-26T09:25:00Z Vimos

For tf r1.2, got deepcopy error. As listed in [sequence to sequence model error #1050](https://github.com/tensorflow/models/issues/1050)

-------------------------------------------------------------------------

2017-05-29T03:52:35Z SunnerLi

As the rookie, I raise some of my opinion.

The following code will make this similar mistake occure:

(Piece of my code)

```python

lstm\_cell = self.LSTMCell(self.num\_hidden)

lstm\_entity = tf.contrib.rnn.DropoutWrapper(lstm\_cell, output\_keep\_prob=0.5)

layer = tf.contrib.rnn.MultiRNNCell([lstm\_entity] \* self.num\_layer)

\_\_, \_ = tf.nn.dynamic\_rnn(layer, self.data, dtype=tf.float64)

```

The error dump as the following:

```

Traceback (most recent call last):

File "IntentNet.py", line 71, in <module>

net = Net(data, target, 5, 1)

File "IntentNet.py", line 45, in \_\_init\_\_

\_\_, \_ = tf.nn.dynamic\_rnn(layer, self.data, dtype=tf.float64)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/ops/rnn.py", line 553, in dynamic\_rnn

dtype=dtype)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/ops/rnn.py", line 720, in \_dynamic\_rnn\_loop

swap\_memory=swap\_memory)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/ops/control\_flow\_ops.py", line 2623, in while\_loop

result = context.BuildLoop(cond, body, loop\_vars, shape\_invariants)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/ops/control\_flow\_ops.py", line 2456, in BuildLoop

pred, body, original\_loop\_vars, loop\_vars, shape\_invariants)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/ops/control\_flow\_ops.py", line 2406, in \_BuildLoop

body\_result = body(\*packed\_vars\_for\_body)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/ops/rnn.py", line 705, in \_time\_step

(output, new\_state) = call\_cell()

File "/usr/local/lib/python2.7/dist-packages/tensorflow/python/ops/rnn.py", line 691, in <lambda>

call\_cell = lambda: cell(input\_t, state)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/contrib/rnn/python/ops/core\_rnn\_cell\_impl.py", line 953, in \_\_call\_\_

cur\_inp, new\_state = cell(cur\_inp, cur\_state)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/contrib/rnn/python/ops/core\_rnn\_cell\_impl.py", line 713, in \_\_call\_\_

output, new\_state = self.\_cell(inputs, state, scope)

File "/usr/local/lib/python2.7/dist-packages/tensorflow/contrib/rnn/python/ops/core\_rnn\_cell\_impl.py", line 235, in \_\_call\_\_

with \_checked\_scope(self, scope or "basic\_lstm\_cell", reuse=self.\_reuse):

File "/usr/lib/python2.7/contextlib.py", line 17, in \_\_enter\_\_

return self.gen.next()

File "/usr/local/lib/python2.7/dist-packages/tensorflow/contrib/rnn/python/ops/core\_rnn\_cell\_impl.py", line 77, in \_checked\_scope

type(cell).\_\_name\_\_))

ValueError: Attempt to reuse RNNCell <tensorflow.contrib.rnn.python.ops.core\_rnn\_cell\_impl.BasicLSTMCell object at 0x7fe4fc7bd150> with a different variable scope than its first use. First use of cell was with scope 'rnn/multi\_rnn\_cell/cell\_0/basic\_lstm\_cell', this attempt is with scope 'rnn/multi\_rnn\_cell/cell\_1/basic\_lstm\_cell'. Please create a new instance of the cell if you would like it to use a different set of weights. If before you were using: MultiRNNCell([BasicLSTMCell(...)] \* num\_layers), change to: MultiRNNCell([BasicLSTMCell(...) for \_ in range(num\_layers)]). If before you were using the same cell instance as both the forward and reverse cell of a bidirectional RNN, simply create two instances (one for forward, one for reverse). In May 2017, we will start transitioning this cell's behavior to use existing stored weights, if any, when it is called with scope=None (which can lead to silent model degradation, so this error will remain until then.)

```

But after I do the revision, It can work.

```python

"""

lstm\_cell = self.LSTMCell(self.num\_hidden)

lstm\_entity = tf.contrib.rnn.DropoutWrapper(lstm\_cell, output\_keep\_prob=0.5)

layer = tf.contrib.rnn.MultiRNNCell([lstm\_entity] \* self.num\_layer)

"""

layer = []

for i in range(self.num\_layer):

lstm\_cell = self.LSTMCell(self.num\_hidden)

lstm\_entity = tf.contrib.rnn.DropoutWrapper(lstm\_cell, output\_keep\_prob=0.5)

layer.append(lstm\_entity)

layer = tf.contrib.rnn.MultiRNNCell(layer)

\_\_, \_ = tf.nn.dynamic\_rnn(layer, self.data, dtype=tf.float64)

```

-------------------------------------------------------------------------

2017-06-02T02:29:57Z philipperemy

None of those workarounds worked for me with Tensorflow 1.1

I'm using `seq2seq` model with `MultiRNNCell` cells.

I had to reverse back to 1.0.1: `pip3 install tensorflow==1.0`

-------------------------------------------------------------------------

2017-06-06T17:20:20Z rileyedmunds

Anyone have these issues when working with legacy\_seq2seq.rnn\_decoder()?

-------------------------------------------------------------------------

2017-06-07T05:50:50Z supermeatboy82

@oxwsds As you said, I change input args cell of tf.contrib.legacy\_seq2seq.embedding\_attention\_seq2seq to two different cell {encoder\_cells, decoder\_cells}. Finally, I get seq2seq model worked. After 73200 setps, I get perplexity 5.54.

Then I run decode part,

>> Who is the president of the United States?

Qui est le pr√©sident des √âtats-Unis ?

Problem solved. Thanks.

-------------------------------------------------------------------------

2017-06-15T13:28:28Z ypruan

@doncat99

It seems that `copy.deepcopy(cell)` in `seq2seq.py` doesn't make effect.

So I change the related part in `seq2seq\_model.py` to

```

if num\_layers > 1:

cell\_enc = tf.contrib.rnn.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

cell\_dec = tf.contrib.rnn.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

# The seq2seq function: we use embedding for the input and attention.

def seq2seq\_f(encoder\_inputs, decoder\_inputs, do\_decode):

return seq2seq.embedding\_attention\_seq2seq(

encoder\_inputs,

decoder\_inputs,

cell\_enc,

cell\_dec,

num\_encoder\_symbols=source\_vocab\_size,

num\_decoder\_symbols=target\_vocab\_size,

embedding\_size=size,

output\_projection=output\_projection,

feed\_previous=do\_decode,

dtype=dtype)

```

-------------------------------------------------------------------------

2017-06-19T10:04:31Z martinambition

@supermeatboy82 , Could you share your code?

-------------------------------------------------------------------------

2017-06-21T12:32:48Z cpury

Upgrading to Tensorflow 1.2.0 and generating the cells in a loop instead of list multiplication fixed this for me.

-------------------------------------------------------------------------

2017-06-22T01:21:13Z syw2014

Got the error with TF1.2 when running translate.py, details:

name: GeForce GTX 1080 Ti

major: 6 minor: 1 memoryClockRate (GHz) 1.582

pciBusID 0000:02:00.0

Total memory: 10.91GiB

Free memory: 10.76GiB

2017-06-22 09:15:04.485252: I tensorflow/core/common\_runtime/gpu/gpu\_device.cc:961] DMA: 0

2017-06-22 09:15:04.485256: I tensorflow/core/common\_runtime/gpu/gpu\_device.cc:971] 0: Y

2017-06-22 09:15:04.485265: I tensorflow/core/common\_runtime/gpu/gpu\_device.cc:1030] Creating TensorFlow device (/gpu:0) -> (device: 0, name: GeForce GTX 1080 Ti, pci bus id: 0000:02:00.0)

Creating 3 layers of 1024 units.

Traceback (most recent call last):

File "translate.py", line 322, in <module>

tf.app.run()

File "/home/lscm/opt/anaconda2/lib/python2.7/site-packages/tensorflow/python/platform/app.py", line 48, in run

\_sys.exit(main(\_sys.argv[:1] + flags\_passthrough))

File "translate.py", line 319, in main

train()

File "translate.py", line 178, in train

model = create\_model(sess, False)

File "translate.py", line 136, in create\_model

dtype=dtype)

File "/data/research/github/dl/tensorflow/tensorflow/models/tutorials/rnn/translate/seq2seq\_model.py", line 179, in \_\_init\_\_

softmax\_loss\_function=softmax\_loss\_function)

File "/home/lscm/opt/anaconda2/lib/python2.7/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 1206, in model\_with\_buckets

decoder\_inputs[:bucket[1]])

File "/data/research/github/dl/tensorflow/tensorflow/models/tutorials/rnn/translate/seq2seq\_model.py", line 178, in <lambda>

lambda x, y: seq2seq\_f(x, y, False),

File "/data/research/github/dl/tensorflow/tensorflow/models/tutorials/rnn/translate/seq2seq\_model.py", line 142, in seq2seq\_f

dtype=dtype)

File "/home/lscm/opt/anaconda2/lib/python2.7/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 848, in embedding\_attention\_seq2seq

encoder\_cell = copy.deepcopy(cell)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 174, in deepcopy

y = copier(memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/site-packages/tensorflow/python/layers/base.py", line 476, in \_\_deepcopy\_\_

setattr(result, k, copy.deepcopy(v, memo))

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 230, in \_deepcopy\_list

y.append(deepcopy(a, memo))

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 230, in \_deepcopy\_list

y.append(deepcopy(a, memo))

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 230, in \_deepcopy\_list

y.append(deepcopy(a, memo))

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 237, in \_deepcopy\_tuple

y.append(deepcopy(a, memo))

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/lscm/opt/anaconda2/lib/python2.7/copy.py", line 343, in \_reconstruct

y.\_\_dict\_\_.update(state)

AttributeError: 'NoneType' object has no attribute 'update'

-------------------------------------------------------------------------

2017-06-23T03:30:15Z Miopas

I also met the error caused by `copy.deepcopy(cell)` in `embedding\_attention\_seq2seq()` when running `self\_test()` in the translate model in tutorial.

I tried to change the codes in `seq2seq\_f()` in `Seq2SeqModel` as follows:

```

def seq2seq\_f(encoder\_inputs, decoder\_inputs, do\_decode=False):

tmp\_cell = copy.deepcopy(cell) #new

return tf.contrib.legacy\_seq2seq.embedding\_attention\_seq2seq(

encoder\_inputs,

decoder\_inputs,

tmp\_cell, #new

num\_encoder\_symbols=source\_vocab\_size,

num\_decoder\_symbols=target\_vocab\_size,

embedding\_size=size,

output\_projection=output\_projection,

feed\_previous=do\_decode,

dtype=dtype)

```

Then there is no error now.

BUT as a rookie I don't know whether the codes here work as before and it seems the changes make the model run slower.

-------------------------------------------------------------------------

2017-06-23T09:22:58Z PratsBhatt

I would like to update everyone that I downgraded the tensorflow to 1.0.0 (tensorflow-GPU) and it is working for me. The models are performing as expected. I assume that the CPU version of 1.0.0 should function as expected? Or?.

Thanks :)

-------------------------------------------------------------------------

2017-06-25T17:37:25Z fabiofumarola

Hi guys, I don't know if you're still interested on it, but I found that the problem is related to the operation of copying the cell passed as params to the `embedding\_attention\_seq2seq` function. This is because the same cell definition is used both for encoder and decoder. I think the tutorial is deprecated since it uses a seq2seq model with bucketing in contrast to a dynamic seq2seq. But, I'm pasting a modified function that works. The function is updated in the file `tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py`.

thanks,

Fabio

```!python

def embedding\_attention\_seq2seq(encoder\_inputs,

decoder\_inputs,

enc\_cell,

dec\_cell,

num\_encoder\_symbols,

num\_decoder\_symbols,

embedding\_size,

num\_heads=1,

output\_projection=None,

feed\_previous=False,

dtype=None,

scope=None,

initial\_state\_attention=False):

"""Embedding sequence-to-sequence model with attention.

This model first embeds encoder\_inputs by a newly created embedding (of shape

[num\_encoder\_symbols x input\_size]). Then it runs an RNN to encode

embedded encoder\_inputs into a state vector. It keeps the outputs of this

RNN at every step to use for attention later. Next, it embeds decoder\_inputs

by another newly created embedding (of shape [num\_decoder\_symbols x

input\_size]). Then it runs attention decoder, initialized with the last

encoder state, on embedded decoder\_inputs and attending to encoder outputs.

Warning: when output\_projection is None, the size of the attention vectors

and variables will be made proportional to num\_decoder\_symbols, can be large.

Args:

encoder\_inputs: A list of 1D int32 Tensors of shape [batch\_size].

decoder\_inputs: A list of 1D int32 Tensors of shape [batch\_size].

cell: tf.nn.rnn\_cell.RNNCell defining the cell function and size.

num\_encoder\_symbols: Integer; number of symbols on the encoder side.

num\_decoder\_symbols: Integer; number of symbols on the decoder side.

embedding\_size: Integer, the length of the embedding vector for each symbol.

num\_heads: Number of attention heads that read from attention\_states.

output\_projection: None or a pair (W, B) of output projection weights and

biases; W has shape [output\_size x num\_decoder\_symbols] and B has

shape [num\_decoder\_symbols]; if provided and feed\_previous=True, each

fed previous output will first be multiplied by W and added B.

feed\_previous: Boolean or scalar Boolean Tensor; if True, only the first

of decoder\_inputs will be used (the "GO" symbol), and all other decoder

inputs will be taken from previous outputs (as in embedding\_rnn\_decoder).

If False, decoder\_inputs are used as given (the standard decoder case).

dtype: The dtype of the initial RNN state (default: tf.float32).

scope: VariableScope for the created subgraph; defaults to

"embedding\_attention\_seq2seq".

initial\_state\_attention: If False (default), initial attentions are zero.

If True, initialize the attentions from the initial state and attention

states.

Returns:

A tuple of the form (outputs, state), where:

outputs: A list of the same length as decoder\_inputs of 2D Tensors with

shape [batch\_size x num\_decoder\_symbols] containing the generated

outputs.

state: The state of each decoder cell at the final time-step.

It is a 2D Tensor of shape [batch\_size x cell.state\_size].

"""

with variable\_scope.variable\_scope(

scope or "embedding\_attention\_seq2seq", dtype=dtype) as scope:

dtype = scope.dtype

# Encoder.

encoder\_cell = enc\_cell

encoder\_cell = core\_rnn\_cell.EmbeddingWrapper(

encoder\_cell,

embedding\_classes=num\_encoder\_symbols,

embedding\_size=embedding\_size)

encoder\_outputs, encoder\_state = rnn.static\_rnn(

encoder\_cell, encoder\_inputs, dtype=dtype)

# First calculate a concatenation of encoder outputs to put attention on.

top\_states = [

array\_ops.reshape(e, [-1, 1, encoder\_cell.output\_size]) for e in encoder\_outputs

]

attention\_states = array\_ops.concat(top\_states, 1)

# Decoder.

output\_size = None

if output\_projection is None:

dec\_cell = core\_rnn\_cell.OutputProjectionWrapper(dec\_cell, num\_decoder\_symbols)

output\_size = num\_decoder\_symbols

if isinstance(feed\_previous, bool):

return embedding\_attention\_decoder(

decoder\_inputs,

encoder\_state,

attention\_states,

dec\_cell,

num\_decoder\_symbols,

embedding\_size,

num\_heads=num\_heads,

output\_size=output\_size,

output\_projection=output\_projection,

feed\_previous=feed\_previous,

initial\_state\_attention=initial\_state\_attention)

# If feed\_previous is a Tensor, we construct 2 graphs and use cond.

def decoder(feed\_previous\_bool):

reuse = None if feed\_previous\_bool else True

with variable\_scope.variable\_scope(

variable\_scope.get\_variable\_scope(), reuse=reuse):

outputs, state = embedding\_attention\_decoder(

decoder\_inputs,

encoder\_state,

attention\_states,

dec\_cell,

num\_decoder\_symbols,

embedding\_size,

num\_heads=num\_heads,

output\_size=output\_size,

output\_projection=output\_projection,

feed\_previous=feed\_previous\_bool,

update\_embedding\_for\_previous=False,

initial\_state\_attention=initial\_state\_attention)

state\_list = [state]

if nest.is\_sequence(state):

state\_list = nest.flatten(state)

return outputs + state\_list

outputs\_and\_state = control\_flow\_ops.cond(feed\_previous,

lambda: decoder(True),

lambda: decoder(False))

outputs\_len = len(decoder\_inputs) # Outputs length same as decoder inputs.

state\_list = outputs\_and\_state[outputs\_len:]

state = state\_list[0]

if nest.is\_sequence(encoder\_state):

state = nest.pack\_sequence\_as(

structure=encoder\_state, flat\_sequence=state\_list)

return outputs\_and\_state[:outputs\_len], state

```

-------------------------------------------------------------------------

2017-06-26T09:05:53Z PratsBhatt

@fabiofumarola Thank you for the function. Looks really helpful. I also saw that the tutorial is deprecated. I am still waiting for an official tutorial release. Looks like you have used the new api. Do you have any code that can be looked up to start coding on the new api?

Any help is well appreciated. Thank you once again :)

-------------------------------------------------------------------------

2017-06-26T21:37:35Z w268wang

@syw2014 Did you fix your issue?

-------------------------------------------------------------------------

2017-06-27T00:59:36Z syw2014

@w268wang not yet, still waiting for other solutions, but comments of @Miopas may have a try, and I am trying the solution of @fabiofumarola

-------------------------------------------------------------------------

2017-07-02T16:13:43Z sachinh35

it says `TypeError: embedding\_attention\_seq2seq() missing 1 required positional argument: 'dec\_cell' `

after using the update that @fabiofumarola posted. Can you guys please help me?

-------------------------------------------------------------------------

2017-07-02T16:42:44Z fabiofumarola

Yes because the update I have proposed require you to change the

embedding\_attention\_seq2seq Function. If you go to the source file in you

tensorflow release you can change the method definition you re self.

On Sun, 2 Jul 2017 at 18:15, sachinh35 <notifications@github.com> trote

> it says TypeError: embedding\_attention\_seq2seq() missing 1 required

> positional argument: 'dec\_cell'

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-312500996>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABepUEc3W8m5CVDQGnCLu4dcJVFwwLDZks5sJ8IOgaJpZM4MWl4f>

> .

>

--

Sent from Gmail Mobile

-------------------------------------------------------------------------

2017-07-02T16:49:12Z sachinh35

Yes i did the same thing. I changed the function in seq2seq.py file in the tensorflow release. Still i am getting the same error. Is there one more argument to the function?

-------------------------------------------------------------------------

2017-07-02T18:55:56Z fabiofumarola

Yes, now in you code you need to specify to rnn\_cells. One for the encoder

and another for the decoder.

On Sun, 2 Jul 2017 at 20:54, fabio fumarola <fabiofumarola@gmail.com> wrote:

> Yes

>

> On Sun, 2 Jul 2017 at 18:50, sachinh35 <notifications@github.com> wrote:

>

>> Yes i did the same thing. I changed the function in seq2seq.py file in

>> the tensorflow release. Still i am getting the same error. Is there one

>> more argument to the function?

>>

>> ‚Äî

>> You are receiving this because you were mentioned.

>> Reply to this email directly, view it on GitHub

>> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-312503106>,

>> or mute the thread

>> <https://github.com/notifications/unsubscribe-auth/ABepUOXTQC\_mzLuhcwW0iZRVkLmmr8yIks5sJ8pugaJpZM4MWl4f>

>> .

>>

> --

Sent from Gmail Mobile

-------------------------------------------------------------------------

2017-07-03T15:45:37Z sachinh35

I am totally new to this. Maybe this a pretty basic question but could you tell what argument to be passed as the decoder cell in this code? I am trying to develop the seq2seq as shown in the tensorflow tutorial using own dataset.

`

from \_\_future\_\_ import absolute\_import

from \_\_future\_\_ import division

from \_\_future\_\_ import print\_function

import random

import numpy as np

from six.moves import xrange # pylint: disable=redefined-builtin

import tensorflow as tf

import data\_utils

class Seq2SeqModel(object):

def \_\_init\_\_(self,

source\_vocab\_size,

target\_vocab\_size,

buckets,

size,

num\_layers,

max\_gradient\_norm,

batch\_size,

learning\_rate,

learning\_rate\_decay\_factor,

use\_lstm=False,

num\_samples=512,

forward\_only=False,

dtype=tf.float32):

self.source\_vocab\_size = source\_vocab\_size

self.target\_vocab\_size = target\_vocab\_size

self.buckets = buckets

self.batch\_size = batch\_size

self.learning\_rate = tf.Variable(

float(learning\_rate), trainable=False, dtype=dtype)

self.learning\_rate\_decay\_op = self.learning\_rate.assign(

self.learning\_rate \* learning\_rate\_decay\_factor)

self.global\_step = tf.Variable(0, trainable=False)

output\_projection = None

softmax\_loss\_function = None

if num\_samples > 0 and num\_samples < self.target\_vocab\_size:

w\_t = tf.get\_variable("proj\_w", [self.target\_vocab\_size, size], dtype=dtype)

w = tf.transpose(w\_t)

b = tf.get\_variable("proj\_b", [self.target\_vocab\_size], dtype=dtype)

output\_projection = (w, b)

def sampled\_loss(labels, inputs):

labels = tf.reshape(labels, [-1, 1])

local\_w\_t = tf.cast(w\_t, tf.float32)

local\_b = tf.cast(b, tf.float32)

local\_inputs = tf.cast(inputs, tf.float32)

return tf.cast(

tf.nn.sampled\_softmax\_loss(local\_w\_t, local\_b, local\_inputs, labels,

num\_samples, self.target\_vocab\_size),

dtype)

softmax\_loss\_function = sampled\_loss

def single\_cell():

return tf.nn.rnn\_cell.GRUCell(size)

if use\_lstm:

def single\_cell():

return tf.nn.rnn\_cell.BasicLSTMCell(size)

cell = single\_cell()

if num\_layers > 1:

cell = tf.nn.rnn\_cell.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

def seq2seq\_f(encoder\_inputs, decoder\_inputs, do\_decode):

return tf.contrib.legacy\_seq2seq.embedding\_attention\_seq2seq(

encoder\_inputs,

decoder\_inputs,

cell,

num\_encoder\_symbols=source\_vocab\_size,

num\_decoder\_symbols=target\_vocab\_size,

embedding\_size=size,

output\_projection=output\_projection,

feed\_previous=do\_decode,

dtype=dtype)

self.encoder\_inputs = []

self.decoder\_inputs = []

self.target\_weights = []

for i in xrange(buckets[-1][0]): # Last bucket is the biggest one.

self.encoder\_inputs.append(tf.placeholder(tf.int32, shape=[None],

name="encoder{0}".format(i)))

for i in xrange(buckets[-1][1] + 1):

self.decoder\_inputs.append(tf.placeholder(tf.int32, shape=[None],

name="decoder{0}".format(i)))

self.target\_weights.append(tf.placeholder(dtype, shape=[None],

name="weight{0}".format(i)))

# Our targets are decoder inputs shifted by one.

targets = [self.decoder\_inputs[i + 1]

for i in xrange(len(self.decoder\_inputs) - 1)]

# Training outputs and losses.

if forward\_only:

self.outputs, self.losses = tf.contrib.legacy\_seq2seq.model\_with\_buckets(

self.encoder\_inputs, self.decoder\_inputs, targets,

self.target\_weights, buckets, lambda x, y: seq2seq\_f(x, y, True),

softmax\_loss\_function=softmax\_loss\_function)

# If we use output projection, we need to project outputs for decoding.

if output\_projection is not None:

for b in xrange(len(buckets)):

self.outputs[b] = [

tf.matmul(output, output\_projection[0]) + output\_projection[1]

for output in self.outputs[b]

]

else:

self.outputs, self.losses = tf.contrib.legacy\_seq2seq.model\_with\_buckets(

self.encoder\_inputs, self.decoder\_inputs, targets,

self.target\_weights, buckets,

lambda x, y: seq2seq\_f(x, y, False),

softmax\_loss\_function=softmax\_loss\_function)

# Gradients and SGD update operation for training the model.

params = tf.trainable\_variables()

if not forward\_only:

self.gradient\_norms = []

self.updates = []

opt = tf.train.GradientDescentOptimizer(self.learning\_rate)

for b in xrange(len(buckets)):

gradients = tf.gradients(self.losses[b], params)

clipped\_gradients, norm = tf.clip\_by\_global\_norm(gradients,

max\_gradient\_norm)

self.gradient\_norms.append(norm)

self.updates.append(opt.apply\_gradients(

zip(clipped\_gradients, params), global\_step=self.global\_step))

self.saver = tf.train.Saver(tf.global\_variables())

def step(self, session, encoder\_inputs, decoder\_inputs, target\_weights,

bucket\_id, forward\_only):

# Check if the sizes match.

encoder\_size, decoder\_size = self.buckets[bucket\_id]

if len(encoder\_inputs) != encoder\_size:

raise ValueError("Encoder length must be equal to the one in bucket,"

" %d != %d." % (len(encoder\_inputs), encoder\_size))

if len(decoder\_inputs) != decoder\_size:

raise ValueError("Decoder length must be equal to the one in bucket,"

" %d != %d." % (len(decoder\_inputs), decoder\_size))

if len(target\_weights) != decoder\_size:

raise ValueError("Weights length must be equal to the one in bucket,"

" %d != %d." % (len(target\_weights), decoder\_size))

# Input feed: encoder inputs, decoder inputs, target\_weights, as provided.

input\_feed = {}

for l in xrange(encoder\_size):

input\_feed[self.encoder\_inputs[l].name] = encoder\_inputs[l]

for l in xrange(decoder\_size):

input\_feed[self.decoder\_inputs[l].name] = decoder\_inputs[l]

input\_feed[self.target\_weights[l].name] = target\_weights[l]

# Since our targets are decoder inputs shifted by one, we need one more.

last\_target = self.decoder\_inputs[decoder\_size].name

input\_feed[last\_target] = np.zeros([self.batch\_size], dtype=np.int32)

# Output feed: depends on whether we do a backward step or not.

if not forward\_only:

output\_feed = [self.updates[bucket\_id], # Update Op that does SGD.

self.gradient\_norms[bucket\_id], # Gradient norm.

self.losses[bucket\_id]] # Loss for this batch.

else:

output\_feed = [self.losses[bucket\_id]] # Loss for this batch.

for l in xrange(decoder\_size): # Output logits.

output\_feed.append(self.outputs[bucket\_id][l])

outputs = session.run(output\_feed, input\_feed)

if not forward\_only:

return outputs[1], outputs[2], None # Gradient norm, loss, no outputs.

else:

return None, outputs[0], outputs[1:] # No gradient norm, loss, outputs.

def get\_batch(self, data, bucket\_id):

encoder\_size, decoder\_size = self.buckets[bucket\_id]

encoder\_inputs, decoder\_inputs = [], []

# Get a random batch of encoder and decoder inputs from data,

# pad them if needed, reverse encoder inputs and add GO to decoder.

for \_ in xrange(self.batch\_size):

encoder\_input, decoder\_input = random.choice(data[bucket\_id])

# Encoder inputs are padded and then reversed.

encoder\_pad = [data\_utils.PAD\_ID] \* (encoder\_size - len(encoder\_input))

encoder\_inputs.append(list(reversed(encoder\_input + encoder\_pad)))

# Decoder inputs get an extra "GO" symbol, and are padded then.

decoder\_pad\_size = decoder\_size - len(decoder\_input) - 1

decoder\_inputs.append([data\_utils.GO\_ID] + decoder\_input +

[data\_utils.PAD\_ID] \* decoder\_pad\_size)

# Now we create batch-major vectors from the data selected above.

batch\_encoder\_inputs, batch\_decoder\_inputs, batch\_weights = [], [], []

# Batch encoder inputs are just re-indexed encoder\_inputs.

for length\_idx in xrange(encoder\_size):

batch\_encoder\_inputs.append(

np.array([encoder\_inputs[batch\_idx][length\_idx]

for batch\_idx in xrange(self.batch\_size)], dtype=np.int32))

# Batch decoder inputs are re-indexed decoder\_inputs, we create weights.

for length\_idx in xrange(decoder\_size):

batch\_decoder\_inputs.append(

np.array([decoder\_inputs[batch\_idx][length\_idx]

for batch\_idx in xrange(self.batch\_size)], dtype=np.int32))

# Create target\_weights to be 0 for targets that are padding.

batch\_weight = np.ones(self.batch\_size, dtype=np.float32)

for batch\_idx in xrange(self.batch\_size):

# We set weight to 0 if the corresponding target is a PAD symbol.

# The corresponding target is decoder\_input shifted by 1 forward.

if length\_idx < decoder\_size - 1:

target = decoder\_inputs[batch\_idx][length\_idx + 1]

if length\_idx == decoder\_size - 1 or target == data\_utils.PAD\_ID:

batch\_weight[batch\_idx] = 0.0

batch\_weights.append(batch\_weight)

return batch\_encoder\_inputs, batch\_decoder\_inputs, batch\_weights`

-------------------------------------------------------------------------

2017-07-03T16:04:53Z ebrevdo

This is a good question for stack overflow.

On Jul 3, 2017 8:46 AM, "sachinh35" <notifications@github.com> wrote:

> I am totally new to this. Maybe this a pretty basic question but could you

> tell what argument to be passed as the decoder cell in this code? I am

> trying to develop the seq2seq as shown in the tensorflow tutorial using own

> dataset.

> `# Copyright 2015 The TensorFlow Authors. All Rights Reserved.

> Licensed under the Apache License, Version 2.0 (the "License"); you may

> not use this file except in compliance with the License. You may obtain a

> copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless

> required by applicable law or agreed to in writing, software distributed

> under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES

> OR CONDITIONS OF ANY KIND, either express or implied. See the License for

> the specific language governing permissions and limitations under the

> License. ============================================================

> ==================

>

> """Sequence-to-sequence model with an attention mechanism."""

>

> from \*future\* import absolute\_import

> from \*future\* import division

> from \*future\* import print\_function

>

> import random

>

> import numpy as np

> from six.moves import xrange # pylint: disable=redefined-builtin

> import tensorflow as tf

>

> import data\_utils

>

> class Seq2SeqModel(object):

> """Sequence-to-sequence model with attention and for multiple buckets.

>

> This class implements a multi-layer recurrent neural network as encoder,

> and an attention-based decoder. This is the same as the model described in

> this paper: http://arxiv.org/abs/1412.7449 - please look there for

> details,

> or into the seq2seq library for complete model implementation.

> This class also allows to use GRU cells in addition to LSTM cells, and

> sampled softmax to handle large output vocabulary size. A single-layer

> version of this model, but with bi-directional encoder, was presented in

> http://arxiv.org/abs/1409.0473

> and sampled softmax is described in Section 3 of the following paper.

> http://arxiv.org/abs/1412.2007

> """

>

> def \*init\*(self,

> source\_vocab\_size,

> target\_vocab\_size,

> buckets,

> size,

> num\_layers,

> max\_gradient\_norm,

> batch\_size,

> learning\_rate,

> learning\_rate\_decay\_factor,

> use\_lstm=False,

> num\_samples=512,

> forward\_only=False,

> dtype=tf.float32):

> """Create the model.

>

> Args:

> source\_vocab\_size: size of the source vocabulary.

> target\_vocab\_size: size of the target vocabulary.

> buckets: a list of pairs (I, O), where I specifies maximum input length

> that will be processed in that bucket, and O specifies maximum output

> length. Training instances that have inputs longer than I or outputs

> longer than O will be pushed to the next bucket and padded accordingly.

> We assume that the list is sorted, e.g., [(2, 4), (8, 16)].

> size: number of units in each layer of the model.

> num\_layers: number of layers in the model.

> max\_gradient\_norm: gradients will be clipped to maximally this norm.

> batch\_size: the size of the batches used during training;

> the model construction is independent of batch\_size, so it can be

> changed after initialization if this is convenient, e.g., for decoding.

> learning\_rate: learning rate to start with.

> learning\_rate\_decay\_factor: decay learning rate by this much when needed.

> use\_lstm: if true, we use LSTM cells instead of GRU cells.

> num\_samples: number of samples for sampled softmax.

> forward\_only: if set, we do not construct the backward pass in the model.

> dtype: the data type to use to store internal variables.

> """

> self.source\_vocab\_size = source\_vocab\_size

> self.target\_vocab\_size = target\_vocab\_size

> self.buckets = buckets

> self.batch\_size = batch\_size

> self.learning\_rate = tf.Variable(

> float(learning\_rate), trainable=False, dtype=dtype)

> self.learning\_rate\_decay\_op = self.learning\_rate.assign(

> self.learning\_rate \* learning\_rate\_decay\_factor)

> self.global\_step = tf.Variable(0, trainable=False)

>

> # If we use sampled softmax, we need an output projection.

> output\_projection = None

> softmax\_loss\_function = None

> # Sampled softmax only makes sense if we sample less than vocabulary size.

> if num\_samples > 0 and num\_samples < self.target\_vocab\_size:

> w\_t = tf.get\_variable("proj\_w", [self.target\_vocab\_size, size], dtype=dtype)

> w = tf.transpose(w\_t)

> b = tf.get\_variable("proj\_b", [self.target\_vocab\_size], dtype=dtype)

> output\_projection = (w, b)

>

> def sampled\_loss(labels, inputs):

> labels = tf.reshape(labels, [-1, 1])

> # We need to compute the sampled\_softmax\_loss using 32bit floats to

> # avoid numerical instabilities.

> local\_w\_t = tf.cast(w\_t, tf.float32)

> local\_b = tf.cast(b, tf.float32)

> local\_inputs = tf.cast(inputs, tf.float32)

> return tf.cast(

> tf.nn.sampled\_softmax\_loss(local\_w\_t, local\_b, local\_inputs, labels,

> num\_samples, self.target\_vocab\_size),

> dtype)

> softmax\_loss\_function = sampled\_loss

>

> # Create the internal multi-layer cell for our RNN.

> def single\_cell():

> return tf.nn.rnn\_cell.GRUCell(size)

> if use\_lstm:

> def single\_cell():

> return tf.nn.rnn\_cell.BasicLSTMCell(size)

> cell = single\_cell()

> if num\_layers > 1:

> cell = tf.nn.rnn\_cell.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

>

> # The seq2seq function: we use embedding for the input and attention.

> def seq2seq\_f(encoder\_inputs, decoder\_inputs, do\_decode):

> return tf.contrib.legacy\_seq2seq.embedding\_attention\_seq2seq(

> encoder\_inputs,

> decoder\_inputs,

> cell,

> num\_encoder\_symbols=source\_vocab\_size,

> num\_decoder\_symbols=target\_vocab\_size,

> embedding\_size=size,

> output\_projection=output\_projection,

> feed\_previous=do\_decode,

> dtype=dtype)

>

> # Feeds for inputs.

> self.encoder\_inputs = []

> self.decoder\_inputs = []

> self.target\_weights = []

> for i in xrange(buckets[-1][0]): # Last bucket is the biggest one.

> self.encoder\_inputs.append(tf.placeholder(tf.int32, shape=[None],

> name="encoder{0}".format(i)))

> for i in xrange(buckets[-1][1] + 1):

> self.decoder\_inputs.append(tf.placeholder(tf.int32, shape=[None],

> name="decoder{0}".format(i)))

> self.target\_weights.append(tf.placeholder(dtype, shape=[None],

> name="weight{0}".format(i)))

>

> # Our targets are decoder inputs shifted by one.

> targets = [self.decoder\_inputs[i + 1]

> for i in xrange(len(self.decoder\_inputs) - 1)]

>

> # Training outputs and losses.

> if forward\_only:

> self.outputs, self.losses = tf.contrib.legacy\_seq2seq.model\_with\_buckets(

> self.encoder\_inputs, self.decoder\_inputs, targets,

> self.target\_weights, buckets, lambda x, y: seq2seq\_f(x, y, True),

> softmax\_loss\_function=softmax\_loss\_function)

> # If we use output projection, we need to project outputs for decoding.

> if output\_projection is not None:

> for b in xrange(len(buckets)):

> self.outputs[b] = [

> tf.matmul(output, output\_projection[0]) + output\_projection[1]

> for output in self.outputs[b]

> ]

> else:

> self.outputs, self.losses = tf.contrib.legacy\_seq2seq.model\_with\_buckets(

> self.encoder\_inputs, self.decoder\_inputs, targets,

> self.target\_weights, buckets,

> lambda x, y: seq2seq\_f(x, y, False),

> softmax\_loss\_function=softmax\_loss\_function)

>

> # Gradients and SGD update operation for training the model.

> params = tf.trainable\_variables()

> if not forward\_only:

> self.gradient\_norms = []

> self.updates = []

> opt = tf.train.GradientDescentOptimizer(self.learning\_rate)

> for b in xrange(len(buckets)):

> gradients = tf.gradients(self.losses[b], params)

> clipped\_gradients, norm = tf.clip\_by\_global\_norm(gradients,

> max\_gradient\_norm)

> self.gradient\_norms.append(norm)

> self.updates.append(opt.apply\_gradients(

> zip(clipped\_gradients, params), global\_step=self.global\_step))

>

> self.saver = tf.train.Saver(tf.global\_variables())

>

> def step(self, session, encoder\_inputs, decoder\_inputs, target\_weights,

> bucket\_id, forward\_only):

> """Run a step of the model feeding the given inputs.

>

> Args:

> session: tensorflow session to use.

> encoder\_inputs: list of numpy int vectors to feed as encoder inputs.

> decoder\_inputs: list of numpy int vectors to feed as decoder inputs.

> target\_weights: list of numpy float vectors to feed as target weights.

> bucket\_id: which bucket of the model to use.

> forward\_only: whether to do the backward step or only forward.

>

> Returns:

> A triple consisting of gradient norm (or None if we did not do backward),

> average perplexity, and the outputs.

>

> Raises:

> ValueError: if length of encoder\_inputs, decoder\_inputs, or

> target\_weights disagrees with bucket size for the specified bucket\_id.

> """

> # Check if the sizes match.

> encoder\_size, decoder\_size = self.buckets[bucket\_id]

> if len(encoder\_inputs) != encoder\_size:

> raise ValueError("Encoder length must be equal to the one in bucket,"

> " %d != %d." % (len(encoder\_inputs), encoder\_size))

> if len(decoder\_inputs) != decoder\_size:

> raise ValueError("Decoder length must be equal to the one in bucket,"

> " %d != %d." % (len(decoder\_inputs), decoder\_size))

> if len(target\_weights) != decoder\_size:

> raise ValueError("Weights length must be equal to the one in bucket,"

> " %d != %d." % (len(target\_weights), decoder\_size))

>

> # Input feed: encoder inputs, decoder inputs, target\_weights, as provided.

> input\_feed = {}

> for l in xrange(encoder\_size):

> input\_feed[self.encoder\_inputs[l].name] = encoder\_inputs[l]

> for l in xrange(decoder\_size):

> input\_feed[self.decoder\_inputs[l].name] = decoder\_inputs[l]

> input\_feed[self.target\_weights[l].name] = target\_weights[l]

>

> # Since our targets are decoder inputs shifted by one, we need one more.

> last\_target = self.decoder\_inputs[decoder\_size].name

> input\_feed[last\_target] = np.zeros([self.batch\_size], dtype=np.int32)

>

> # Output feed: depends on whether we do a backward step or not.

> if not forward\_only:

> output\_feed = [self.updates[bucket\_id], # Update Op that does SGD.

> self.gradient\_norms[bucket\_id], # Gradient norm.

> self.losses[bucket\_id]] # Loss for this batch.

> else:

> output\_feed = [self.losses[bucket\_id]] # Loss for this batch.

> for l in xrange(decoder\_size): # Output logits.

> output\_feed.append(self.outputs[bucket\_id][l])

>

> outputs = session.run(output\_feed, input\_feed)

> if not forward\_only:

> return outputs[1], outputs[2], None # Gradient norm, loss, no outputs.

> else:

> return None, outputs[0], outputs[1:] # No gradient norm, loss, outputs.

>

> def get\_batch(self, data, bucket\_id):

> """Get a random batch of data from the specified bucket, prepare for step.

>

> To feed data in step(..) it must be a list of batch-major vectors, while

> data here contains single length-major cases. So the main logic of this

> function is to re-index data cases to be in the proper format for feeding.

>

> Args:

> data: a tuple of size len(self.buckets) in which each element contains

> lists of pairs of input and output data that we use to create a batch.

> bucket\_id: integer, which bucket to get the batch for.

>

> Returns:

> The triple (encoder\_inputs, decoder\_inputs, target\_weights) for

> the constructed batch that has the proper format to call step(...) later.

> """

> encoder\_size, decoder\_size = self.buckets[bucket\_id]

> encoder\_inputs, decoder\_inputs = [], []

>

> # Get a random batch of encoder and decoder inputs from data,

> # pad them if needed, reverse encoder inputs and add GO to decoder.

> for \_ in xrange(self.batch\_size):

> encoder\_input, decoder\_input = random.choice(data[bucket\_id])

>

> # Encoder inputs are padded and then reversed.

> encoder\_pad = [data\_utils.PAD\_ID] \* (encoder\_size - len(encoder\_input))

> encoder\_inputs.append(list(reversed(encoder\_input + encoder\_pad)))

>

> # Decoder inputs get an extra "GO" symbol, and are padded then.

> decoder\_pad\_size = decoder\_size - len(decoder\_input) - 1

> decoder\_inputs.append([data\_utils.GO\_ID] + decoder\_input +

> [data\_utils.PAD\_ID] \* decoder\_pad\_size)

>

> # Now we create batch-major vectors from the data selected above.

> batch\_encoder\_inputs, batch\_decoder\_inputs, batch\_weights = [], [], []

>

> # Batch encoder inputs are just re-indexed encoder\_inputs.

> for length\_idx in xrange(encoder\_size):

> batch\_encoder\_inputs.append(

> np.array([encoder\_inputs[batch\_idx][length\_idx]

> for batch\_idx in xrange(self.batch\_size)], dtype=np.int32))

>

> # Batch decoder inputs are re-indexed decoder\_inputs, we create weights.

> for length\_idx in xrange(decoder\_size):

> batch\_decoder\_inputs.append(

> np.array([decoder\_inputs[batch\_idx][length\_idx]

> for batch\_idx in xrange(self.batch\_size)], dtype=np.int32))

>

> # Create target\_weights to be 0 for targets that are padding.

> batch\_weight = np.ones(self.batch\_size, dtype=np.float32)

> for batch\_idx in xrange(self.batch\_size):

> # We set weight to 0 if the corresponding target is a PAD symbol.

> # The corresponding target is decoder\_input shifted by 1 forward.

> if length\_idx < decoder\_size - 1:

> target = decoder\_inputs[batch\_idx][length\_idx + 1]

> if length\_idx == decoder\_size - 1 or target == data\_utils.PAD\_ID:

> batch\_weight[batch\_idx] = 0.0

> batch\_weights.append(batch\_weight)

> return batch\_encoder\_inputs, batch\_decoder\_inputs, batch\_weights`

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-312679587>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim0l5UMHHtbL1sz7meXserV8NVS7cks5sKQzXgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-07-03T16:53:12Z sachinh35

Okay! thanks though! :)

-------------------------------------------------------------------------

2017-07-03T17:08:09Z PratsBhatt

@ebrevdo is there any update on when the new tutorial of seq2seq using new api will come out?

Thank you. Amazing work!.

-------------------------------------------------------------------------

2017-07-03T17:15:43Z prashantserai

yeah waiting for the new tutorial... would be great to know if it's planned to be released anytime soon.. @ebrevdo

tried to take code in the kernel tests and retrofit the beam search with the legacy seq2seq, but it was challenging...

-------------------------------------------------------------------------

2017-07-03T17:25:53Z ebrevdo

We're hoping for this coming week!

On Jul 3, 2017 10:16 AM, "prashantserai" <notifications@github.com> wrote:

> yeah waiting for the new tutorial... would be great to know if it's

> planned to be released anytime soon.. @ebrevdo

> <https://github.com/ebrevdo>

>

> tried to take code in the kernel tests and retrofit the beam search with

> the legacy seq2seq, but it seemed challenging...

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-312697274>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtim45-HTuQrIRDhphqqHjqkKOKTe53ks5sKSHYgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-07-20T06:39:42Z tshi1983

Hi guys,

Any update to this issue, I'm experiencing the same on tensorflow 1.1-gpu for mac os x

-------------------------------------------------------------------------

2017-07-24T13:15:49Z selinachenxi

@tshi1983

I got the same problem with tensorflow 1.1-gpu for ubuntu.

I upgrade to tf 1.2. It still doesn't work.

Then I change the function embedding\_attention\_seq2seq in file

tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py

to the one as @fabiofumarola suggested above.

Now it starts training. I haven't tested decoding yet.

-------------------------------------------------------------------------

2017-07-31T06:36:59Z huxuanlai

Move the code on cell definition into seq2seq\_f:

```

def seq2seq\_f(encoder\_inputs, decoder\_inputs, do\_decode):

def single\_cell():

return tf.contrib.rnn.GRUCell(size)

if use\_lstm:

def single\_cell():

return tf.contrib.rnn.BasicLSTMCell(size)

cell = single\_cell()

if num\_layers > 1:

cell = tf.contrib.rnn.MultiRNNCell([single\_cell() for \_ in range(num\_layers)])

return tf.contrib.legacy\_seq2seq.embedding\_attention\_seq2seq(

...

)

```

Then "python translate.py --data\_dir data/ --train\_dir checkpoint/ --size=256 --num\_layers=2 --steps\_per\_checkpoint=50" can work.

-------------------------------------------------------------------------

2017-07-31T11:50:23Z a111xushuai

@huxuanlai it works! At least it's training now, thx!

-------------------------------------------------------------------------

2017-08-05T17:23:24Z btchncamaro

@huxuanlai Works for me as well.

-------------------------------------------------------------------------

2017-08-10T20:33:28Z jkarimi91

I am receiving the same `AttributeError: 'NoneType' object has no attribute 'update'` but with `tf.contrib.legacy\_seq2seq.model\_with\_buckets`. I am running tf 1.2.1 (GPU) on ubuntu 16.04 lts.

This only seems to occur when I have more than 1 bucket.

full traceback:

```

Traceback (most recent call last):

File "chatbot.py", line 262, in <module>

main()

File "chatbot.py", line 257, in main

train()

File "chatbot.py", line 138, in train

model.build\_graph()

File "/home/jkarimi91/Projects/cs20/code/hw/a3/model.py", line 134, in build\_graph

self.\_create\_loss()

File "/home/jkarimi91/Projects/cs20/code/hw/a3/model.py", line 102, in \_create\_loss

softmax\_loss\_function=self.softmax\_loss\_function)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 1206, in model\_with\_buckets

decoder\_inputs[:bucket[1]])

File "/home/jkarimi91/Projects/cs20/code/hw/a3/model.py", line 101, in <lambda>

lambda x, y: \_seq2seq\_f(x, y, False),

File "/home/jkarimi91/Projects/cs20/code/hw/a3/model.py", line 76, in \_seq2seq\_f

feed\_previous=do\_decode)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/site-packages/tensorflow/contrib/legacy\_seq2seq/python/ops/seq2seq.py", line 848, in embedding\_attention\_seq2seq

encoder\_cell = copy.deepcopy(cell)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 174, in deepcopy

y = copier(memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/site-packages/tensorflow/python/layers/base.py", line 476, in \_\_deepcopy\_\_

setattr(result, k, copy.deepcopy(v, memo))

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 230, in \_deepcopy\_list

y.append(deepcopy(a, memo))

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 230, in \_deepcopy\_list

y.append(deepcopy(a, memo))

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 230, in \_deepcopy\_list

y.append(deepcopy(a, memo))

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 237, in \_deepcopy\_tuple

y.append(deepcopy(a, memo))

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 334, in \_reconstruct

state = deepcopy(state, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 163, in deepcopy

y = copier(x, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 257, in \_deepcopy\_dict

y[deepcopy(key, memo)] = deepcopy(value, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 190, in deepcopy

y = \_reconstruct(x, rv, 1, memo)

File "/home/jkarimi91/Apps/anaconda2/envs/tf/lib/python2.7/copy.py", line 343, in \_reconstruct

y.\_\_dict\_\_.update(state)

AttributeError: 'NoneType' object has no attribute 'update'

```

-------------------------------------------------------------------------

2017-08-16T01:23:30Z LevineHuang

@Tshzzz @jtubert

thx, your solution worked for me. My tf verstion is 1.1.0.

I changed from:

```python

lstm\_cell = tf.contrib.rnn.BasicLSTMCell(HIDDEN\_SIZE, state\_is\_tuple=True)

cell = tf.contrib.rnn.MultiRNNCell([lstm\_cell() for \_ in range(NUM\_LAYERS)])

output, \_ = tf.nn.dynamic\_rnn(cell, X, dtype=tf.float32)

```

to:

```python

cells=[]

for \_ in range(NUM\_LAYERS):

cell = tf.contrib.rnn.BasicLSTMCell(HIDDEN\_SIZE, state\_is\_tuple=True)

cells.append(cell)

multicell = tf.contrib.rnn.MultiRNNCell(cells, state\_is\_tuple=True)

output, \_ = tf.nn.dynamic\_rnn(multicell, X, dtype=tf.float32)

```

-------------------------------------------------------------------------

2017-08-18T15:03:01Z saurabhvyas

This is still not fixed , tried all possible solutions , ones mentioned in this thread and stackoverflow , it doesn't work with tensorflow 1.3 or 1.2 or 1.1

-------------------------------------------------------------------------

2017-08-18T19:21:17Z comsian106

I'm facing this error:

```TypeError: embedding\_attention\_seq2seq() missing 1 required positional argument: 'dec\_cell' ```

The error points to this function in seq2seq\_model.py which is line 142 in seq2seq\_model.py:

```def seq2seq\_f(encoder\_inputs, decoder\_inputs, do\_decode):

return tf.contrib.legacy\_seq2seq.embedding\_attention\_seq2seq(

encoder\_inputs,

decoder\_inputs,

cell,

num\_encoder\_symbols=source\_vocab\_size,

num\_decoder\_symbols=target\_vocab\_size,

embedding\_size=size,

output\_projection=output\_projection,

feed\_previous=do\_decode,

dtype=dtype)

```

Anyone who came across with this error and managed to solve this, please help me correct this issue.

-------------------------------------------------------------------------

2017-08-25T04:01:25Z ybdx

ValueError: Attempt to reuse RNNCell <tensorflow.contrib.rnn.python.ops.core\_rnn\_cell\_impl.GRUCell object at 0x11d32cbd0> with a different variable scope than its first use. First use of cell was with scope 'rnn/multi\_rnn\_cell/cell\_0/gru\_cell', this attempt is with scope 'rnn/multi\_rnn\_cell/cell\_1/gru\_cell'. Please create a new instance of the cell if you would like it to use a different set of weights. If before you were using: MultiRNNCell([GRUCell(...)] \* num\_layers), change to: MultiRNNCell([GRUCell(...) for \_ in range(num\_layers)]). If before you were using the same cell instance as both the forward and reverse cell of a bidirectional RNN, simply create two instances (one for forward, one for reverse). In May 2017, we will start transitioning this cell's behavior to use existing stored weights, if any, when it is called with scope=None (which can lead to silent model degradation, so this error will remain until then.)

the origin code:

from tensorflow.contrib import rnn

inputs = tf.placeholder(dtype=tf.int32, shape=[None, None], name="inputs")

keep\_prob = tf.placeholder(dtype=tf.float32, name="keep\_prob")

cell = rnn.GRUCell(10)

cell = rnn.DropoutWrapper(cell=cell, input\_keep\_prob=keep\_prob)

cell = rnn.MultiRNNCell([cell for \_ in range(5)], state\_is\_tuple=True)

outs, states = tf.nn.dynamic\_rnn(cell=cell, inputs=look\_up, dtype=tf.float32)

solution:

inputs = tf.placeholder(dtype=tf.int32, shape=[None, None], name="inputs")

keep\_prob = tf.placeholder(dtype=tf.float32, name="keep\_prob")

cell = rnn.MultiRNNCell([rnn.DropoutWrapper(rnn.GRUCell(10), input\_keep\_prob=keep\_prob) for \_ in range(5)] , state\_is\_tuple=True)

-------------------------------------------------------------------------

2017-10-01T16:11:16Z ebrevdo

Do you have this issue with the tf nightlies?

On Oct 1, 2017 8:34 AM, "Baohua Zhou" <notifications@github.com> wrote:

> I have the same issue when using tensorflow 1.1 on cpu with ios.

>

> ‚Äî

> You are receiving this because you were mentioned.

> Reply to this email directly, view it on GitHub

> <https://github.com/tensorflow/tensorflow/issues/8191#issuecomment-333384725>,

> or mute the thread

> <https://github.com/notifications/unsubscribe-auth/ABtimwOv7vf5vvFXBllbZryjCFwmJcU6ks5sn7DxgaJpZM4MWl4f>

> .

>

-------------------------------------------------------------------------

2017-10-23T10:30:51Z PR-Iyyer

AttributeError: 'NoneType' object has no attribute 'update'

in tf=1.3

-------------------------------------------------------------------------

2017-11-27T09:31:33Z rashmishrm

ValueError: Attempt to reuse RNNCell <tensorflow.contrib.rnn.python.ops.core\_rnn\_cell\_impl.GRUCell object at 0x117f7cbd0> with a different variable scope than its first use. First use of cell was with scope 'embedding\_attention\_seq2seq/rnn/multi\_rnn\_cell/cell\_0/gru\_cell', this attempt is with scope 'embedding\_attention\_seq2seq/rnn/multi\_rnn\_cell/cell\_1/gru\_cell'. Please create a new instance of the cell if you would like it to use a different set of weights. If before you were using: MultiRNNCell([GRUCell(...)] \* num\_layers), change to: MultiRNNCell([GRUCell(...) for \_ in range(num\_layers)]). If before you were using the same cell instance as both the forward and reverse cell of a bidirectional RNN, simply create two instances (one for forward, one for reverse). In May 2017, we will start transitioning this cell's behavior to use existing stored weights, if any, when it is called with scope=None (which can lead to silent model degradation, so this error will remain until then.)

-------------------------------------------------------------------------

2017-12-22T07:48:59Z tensorflowbutler

It has been 14 days with no activity and the `awaiting tensorflower` label was assigned. Please update the label and/or status accordingly.

-------------------------------------------------------------------------

2018-01-05T19:13:58Z tensorflowbutler

Nagging Awaiting TensorFlower: It has been 14 days with no activityand the `awaiting tensorflower` label was assigned. Please update the label and/or status accordingly.

-------------------------------------------------------------------------

2018-01-05T19:32:31Z ebrevdo

The solution is to move to a newer version of TF. This thread has drastically diverged from its original issue. Closing.

-------------------------------------------------------------------------

2018-04-13T08:28:20Z monk1337

If you want instant solution you can try what i tried :

`pip install tensorflow==1.0

`

The issue is with tenorflow 1.1 version , it worked for me.

-------------------------------------------------------------------------

-------------------------------------

2017-12-22T07:48:59Z tensorflowbutler

It has been 14 days with no activity and the `awaiting tensorflower` label was assigned. Please update the label and/or status accordingly.

-------------------------------------------------------------------------

2018-01-05T19:13:58Z tensorflowbutler

Nagging Awaiting TensorFlower: It has been 14 days with no activityand the `awaiting tensorflower` label was assigned. Please update the label and/or status accordingly.

-------------------------------------------------------------------------

2018-01-05T19:32:31Z ebrevdo

The solution is to move to a newer version of TF. This thread has drastically diverged from its original issue. Closing.

-------------------------------------------------------------------------

2018-04-13T08:28:20Z monk1337

If you want instant solution you can try what i tried :

`pip install tensorflow==1.0

`

The issue is with tenorflow 1.1 version , it worked for me.

-------------------------------------------------------------------------