Tensorflow 개발 팁

목차


```
In [1]: %load_ext do_not_print_href
```

GPU 사용 방식 설정

`tf.ConfigProto`

- http://devdocs.io/tensorflow~python/tf/configproto
- https://github.com/tensorflow/tensorflow/blob/r1.2/tensorflow/core/protobuf/config.proto

```
config = tf.ConfigProto()
config.gpu_options.allow_growth = True
...
session = tf.Session(config=config)
```

CUDA_VISIBLE_DEVICES

- 여러개의 GPU가 장착된 시스템에서 사용시, 특정 GPU만 선택해서 사용하도록 설정

```
CUDA_VISIBLE_DEVICES=0,1
```

- GPU가 있는 시스템에서, CPU만 사용하도록 만들 때도 사용 가능

```
CUDA_VISIBLE_DEVICES=-
```

- 환경변수로 저장

```
export CUDA_VISIBLE_DEVICES=0,1
python ...
```

- 필요시 ~/.bashrc 파일에 추가하면 로그인 할 때마다 적용
- 참고: http://acceleware.com/blog/cudavisibledevices-masking-gpus

주의사항

- *GPU 번호가 `nvidia-smi`에서 보여주는 번호와 다를 수 있습니다.*

모델의 저장과 복원

모델 파라메터의 저장과 복원

tf.train.Saver

```
tf.train.Saver(
    var_list=None,
    reshape=False,
    sharded=False,
    max_to_keep=5,
    keep_checkpoint_every_n_hours=10000.0,
    name=None,
    restore_sequentially=False,
    saver_def=None,
    builder=None,
    defer_build=False,
    allow_empty=False,
    write_version=tf.train.SaverDef.V2,
    pad_step_number=False,
    save_relative_paths=False
)
```

모델 파라메터 저장

***saver*.save()**

```
save(
    sess,
    save_path,
    global_step=None,
    latest_filename=None,
    meta_graph_suffix='meta',
    write_meta_graph=True,
    write_state=True
)
```

> Returns: A string: path at which the variables were saved. If the saver is sharded, this string ends with: '-?????-of-nnnnn' where 'nnnnn' is the number of shards created. If the saver is empty, returns None.

모델 파라메터 복원

***saver*.restore()**

```
restore(
    sess,
    save_path
)
```

저장 복원 예시

MNIST 데이터 준비 (1주 3일차 데이터 활용)

In [2]:
```
%%bash
test -s ./mnist/train-images-idx3-ubyte || (
 mkdir -p ./mnist
 cd ./mnist
 echo "$(pwd)"
 wget -q \
  http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz \
  http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz \
  http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz \
  http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz
 gzip -d *.gz
)
```

MNIST 데이터 로딩

In [3]:
```
from __future__ import print_function, division
import numpy as np
import tensorflow as tf
```

In [4]:
```
data_dir = './mnist'
data_dir

images          = np.fromfile(data_dir +
                      '/train-images-idx3-ubyte',dtype=np.uint8)
images          = images[16:].reshape([-1,28,28]).astype(np.float32)
images          = images / 127.0 - 1.0

labels          = np.fromfile(data_dir +
                      '/train-labels-idx1-ubyte',dtype=np.uint8)
labels          = labels[8:].astype(np.int64)
```

In [5]:
```
images.shape, images.dtype, labels.shape, labels.dtype
```

Out[5]:
```
((60000, 28, 28), dtype('float32'), (60000,), dtype('int64'))
```

MNIST 훈련 네트워크 정의

In [6]:
```
class Model:
    '''
    simple 1-layer fully-connected network for MNIST problem
    '''
    def __init__(self):
        learning_rate = 0.05
        input_size    = 28 * 28
        output_size   = 10

        with tf.name_scope('data'):
            input_        = tf.placeholder(
                shape=[None, input_size],
                dtype=tf.float32, name="input")
            label         = tf.placeholder(
                shape=[None],
                dtype=tf.int64, name="label")
        with tf.name_scope('fcn'):
            weights       = tf.Variable(
                tf.zeros([input_size, output_size]))
            biases        = tf.Variable(
                tf.zeros([output_size]))
            output        = tf.matmul(input_, weights) + biases

        with tf.name_scope('optimize'):
            loss          = \
                tf.losses.sparse_softmax_cross_entropy(
                    label,
                    output)
            optimize      = \
                tf.train.AdamOptimizer(learning_rate) \
                    .minimize(loss)

        with tf.name_scope('prediction'):
            pred          = tf.argmax(output,
                                      axis=1,
                                      name='pred')
            accuracy      = \
                1.0 - \
                tf.cast(
                    tf.count_nonzero(pred-label),
                    tf.float32) / \
                tf.cast(tf.size(label),tf.float32)

        self.input    = input_
        self.label    = label
        self.loss     = loss
        self.optimize = optimize
        self.pred     = pred
        self.accuracy = accuracy
        self.weights  = weights
        self.biases   = biases
```

In [7]:
```
model = Model()
```

MNIST 훈련

In [8]:
```
def train(model,max_epochs=20):

    batch_size    = 128
    batch_count   = 60000 // batch_size

    step          = 1

    config = tf.ConfigProto(gpu_options={'allow_growth': True})
    with tf.Session(config=config) as session:

        saver = tf.train.Saver()

        session.run(tf.global_variables_initializer())
        for ep in range(max_epochs):
            total_loss       = 0
            total_acc_v      = 0
            for i in range(batch_count):
                img = np.reshape(
                        images[i*batch_size:(i+1)*batch_size],
                        [batch_size, 28 * 28])
                lbl = (labels[i*batch_size:(i+1)*batch_size])
                _, loss_v, acc_v = \
                  session.run(
                    [model.optimize,
                     model.loss,
                     model.accuracy],
                    feed_dict= {
                        model.input: img,
                        model.label: lbl})
                step        += 1
                total_loss  += loss_v
                total_acc_v += acc_v

                # <<<=== 주기적으로 save ===>>>
                ################################################
                # 2000 step 마다 save/example 에 저장
                ################################################
                if step % 2000 == 0:
                    checkpoint = saver.save(session,
                                            'save/example',
                                            global_step=step)
                    print('Saved: %s'%(checkpoint,))


            print('ep %d: loss: %.5f acc: %.3f%%' % (
                ep+1,
                total_loss / batch_count,
                total_acc_v / batch_count * 100))

        ################################################
        # save/graph 에 저장
        ################################################
```

```
        tf.train.export_meta_graph(filename='save/graph')

        ##################################################
        # save/example 에 저장
        ##################################################

        checkpoint = saver.save(session,
                                'save/example',
                                global_step=step)
        print('Saved: %s'%(checkpoint,))


        writer = tf.summary.FileWriter(
            'save',
            tf.get_default_graph())

    return step
```

In [9]:
```
!rm -fr save
```

In [10]:
```
train(model,20);
```

```
ep 1: loss: 2.14669 acc: 82.175%
ep 2: loss: 1.65718 acc: 85.445%
ep 3: loss: 1.94722 acc: 85.882%
ep 4: loss: 2.05991 acc: 86.485%
Saved: save/example-2000
ep 5: loss: 1.99479 acc: 86.988%
ep 6: loss: 2.04258 acc: 87.023%
ep 7: loss: 2.02497 acc: 87.336%
ep 8: loss: 2.11884 acc: 87.238%
Saved: save/example-4000
ep 9: loss: 2.04744 acc: 87.570%
ep 10: loss: 2.03975 acc: 87.690%
ep 11: loss: 1.98439 acc: 87.857%
ep 12: loss: 2.10790 acc: 87.630%
Saved: save/example-6000
ep 13: loss: 2.15829 acc: 87.759%
ep 14: loss: 2.23853 acc: 87.727%
ep 15: loss: 2.06537 acc: 88.418%
ep 16: loss: 2.12343 acc: 88.103%
ep 17: loss: 2.09931 acc: 88.303%
Saved: save/example-8000
ep 18: loss: 2.14287 acc: 88.134%
ep 19: loss: 2.19149 acc: 88.086%
ep 20: loss: 2.12249 acc: 88.361%
Saved: save/example-9361
```

In [11]:
```
!ls -altr save
```

```
total 1808
-rw-r--r--   1 rhee  staff  52981 Sep  8 11:03 example-2000.meta
-rw-r--r--   1 rhee  staff    343 Sep  8 11:03 example-2000.index
-rw-r--r--   1 rhee  staff  94208 Sep  8 11:03 example-2000.data-0
0000-of-00001
drwxrwxr-x  29 rhee  staff    986 Sep  8 11:03 ..
-rw-r--r--   1 rhee  staff  52981 Sep  8 11:03 example-4000.meta
-rw-r--r--   1 rhee  staff    343 Sep  8 11:03 example-4000.index
-rw-r--r--   1 rhee  staff  94208 Sep  8 11:03 example-4000.data-0
0000-of-00001
-rw-r--r--   1 rhee  staff  52981 Sep  8 11:03 example-6000.meta
-rw-r--r--   1 rhee  staff    343 Sep  8 11:03 example-6000.index
-rw-r--r--   1 rhee  staff  94208 Sep  8 11:03 example-6000.data-0
0000-of-00001
-rw-r--r--   1 rhee  staff  52981 Sep  8 11:03 example-8000.meta
-rw-r--r--   1 rhee  staff    343 Sep  8 11:03 example-8000.index
-rw-r--r--   1 rhee  staff  94208 Sep  8 11:03 example-8000.data-0
0000-of-00001
-rw-r--r--   1 rhee  staff  52911 Sep  8 11:03 graph
-rw-r--r--   1 rhee  staff  52981 Sep  8 11:03 example-9361.meta
-rw-r--r--   1 rhee  staff    343 Sep  8 11:03 example-9361.index
-rw-r--r--   1 rhee  staff  94208 Sep  8 11:03 example-9361.data-0
0000-of-00001
-rw-r--r--   1 rhee  staff    253 Sep  8 11:03 checkpoint
drwxr-xr-x  20 rhee  staff    680 Sep  8 11:03 .
-rw-r--r--   1 rhee  staff  98509 Sep  8 11:03 events.out.tfevents
.1504836214.rhee-mbp.local
```

여기서 잠깐, 텐서보드 그래프 기능 한 번 확인 하겠습니다.

In [12]:
```
!tensorboard --logdir save
```

```
Starting TensorBoard 54 at http://rhee-mbp.local:6006
(Press CTRL+C to quit)
^C
```

저장된 모델 파라메터의 복원

모델 파라메터 텐서는 문자열로 표현 가능

In [13]:
```
model.pred.name, model.input.name
```

Out[13]:
```
(u'prediction/pred:0', u'data/input:0')
```

In [14]:
```
tf.reset_default_graph()
model = Model()
```

In [15]:
```
model.pred.name, model.input.name
```

Out[15]:
```
(u'prediction/pred:0', u'data/input:0')
```

In [16]:
```
from __future__ import print_function
import numpy as np
import tensorflow as tf

def infer(image, label):

    config = tf.ConfigProto(gpu_options={'allow_growth': True})
    with tf.Session(config=config) as session:

        session.run(tf.global_variables_initializer())

        ##################################################
        # save/example 에 저장한 파일로 부터 복원
        ##################################################

        saver = tf.train.Saver()
        saver.restore(session, 'save/example-9361')

        pred = session.run(
            # model.pred
            'prediction/pred:0',
            # model.input
            {'data/input:0':[image.reshape([28*28])]})

        print('infer: label={}, pred={}'.format(label,pred[0]))
```

In [17]:
```
infer(images[1234],labels[1234])
```

```
INFO:tensorflow:Restoring parameters from save/example-9361
infer: label=3, pred=3
```

그래프의 저장과 복원

그래프 저장

tf.train.export_meta_graph('save/graph')

```
tf.train.export_meta_graph(
    filename=None,
    meta_info_def=None,
    graph_def=None,
    saver_def=None,
    collection_list=None,
    as_text=False,
    graph=None,
    export_scope=None,
    clear_devices=False,
    **kwargs
)
```

그래프 복원

tf.train.import_meta_graph('save/graph')

```
import_meta_graph(
    meta_graph_or_file,
    clear_devices=False,
    import_scope=None,
    **kwargs
)
```

저장된 그래프와 모델 파라메터 복원

In [18]:
```
tf.reset_default_graph()
# model = Model()  # <<<==== 만들 필요 없음
tf.train.import_meta_graph('save/graph')
```

Out[18]: <tensorflow.python.training.saver.Saver at 0x11a7169d0>

In [19]:
```
from __future__ import print_function
import numpy as np
import tensorflow as tf

def infer(image, label):

    config = tf.ConfigProto(gpu_options={'allow_growth': True})
    with tf.Session(config=config) as session:

        session.run(tf.global_variables_initializer())

        ###################################################
        # save/example 에 저장한 파일로 부터 복원
        ###################################################

        saver = tf.train.Saver()
        saver.restore(session, 'save/example-9361')

        model_input = 'data/input:0'
        model_pred  = 'prediction/pred:0'

        pred = session.run(
                    model_pred,
                    {model_input:[image.reshape([28*28])]})

        print('infer: label={}, pred={}'.format(label,pred[0]))


infer(images[1234],labels[1234])
```

```
INFO:tensorflow:Restoring parameters from save/example-9361
infer: label=3, pred=3
```

실제로 전혀 다른 파이썬 인터프리터로 확인

In [20]:
```
%pycat restore_test.py
```

In [21]:
```
!python restore_test.py
```

```
2017-09-08 11:03:53.163645: W tensorflow/core/platform/cpu_feature
_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.2
instructions, but these are available on your machine and could sp
eed up CPU computations.
2017-09-08 11:03:53.163666: W tensorflow/core/platform/cpu_feature
_guard.cc:45] The TensorFlow library wasn't compiled to use AVX in
structions, but these are available on your machine and could spee
d up CPU computations.
2017-09-08 11:03:53.163670: W tensorflow/core/platform/cpu_feature
_guard.cc:45] The TensorFlow library wasn't compiled to use AVX2 i
nstructions, but these are available on your machine and could spe
ed up CPU computations.
2017-09-08 11:03:53.163674: W tensorflow/core/platform/cpu_feature
_guard.cc:45] The TensorFlow library wasn't compiled to use FMA in
structions, but these are available on your machine and could spee
d up CPU computations.
infer: label=3, pred=3
```

모델의 저장과 복원 정리

파라메터 저장

```
# 모든 파라메터 저장 (trainable/non-trainable 구별은 하지 않음)
saver = tf.train.Saver()
saver_checkpoint = saver.save(sess, 'save/my-model', global_step=step)

# 선택한 파라메터만 저장
selected_saver = tf.train.Saver([model.w1,model.w2])
selected_saver.save(sess, 'save2/weights', global_step=step)

# scope 속성을 활용한 파라메터 선택
export_saver = tf.train.Saver( \
    tf.get_collection(tf.GraphKeys.GLOBAL_VARIABLES, scope='my_block'))
export_saver.save(sess, 'export/my-block', global_step=step)
```

> tf.get_collection 을 이용해서 원하는 텐서의 목록을 얻음
>
> tf.GraphKeys.GLOBAL_VARIABLES 를 지정해서 모든 글로벌 파라메터의 목록을 얻을 수 있음

파라메터 복원

```
# checkpoint 경로를 미리 알고 있다면
saver = tf.train.Saver()
saver.restore(sess, saver_checkpoint)

# 지정 디렉토리 'save'에서 가장 최근의 체크포인트 경로를 읽어옴
checkpoint = tf.train.latest_checkpoint('save')
saver_for_restore = tf.train.Saver()
saver_for_restore.restore(sess, checkpoint)

# 모든 파라메터를 다 읽어오지 않고, 정해진 파라메터들만 읽어오고 싶다면
import_checkpoint = tf.train.latest_checkpoint('export')
import_saver = tf.train.Saver( \
    tf.get_collection(tf.GraphKeys.GLOBAL_VARIABLES, \
                      scope='my_block'))
import_saver.restore(sess,import_checkpoint)
```

그래프 저장

```
tf.train.export_meta_graph(filename='save/graph')
```

그래프 복원

```
tf.train.import_meta_graph('save/graph')
```

저장된 모델을 텐서플로우 외부에서 확인

- inspect_checkpoint.py
- https://github.com/tensorflow/tensorflow/blob/master/tensorflow/python/tools/inspect_checkpoint.py

In [22]:
```
!python -mtensorflow.python.tools.inspect_checkpoint
```

```
Usage: inspect_checkpoint --file_name=checkpoint_file_name [--tens
or_name=tensor_to_print]
```

In [23]:
```
%%bash
python -mtensorflow.python.tools.inspect_checkpoint \
  --file_name 'save/example-9361'
```

```
fcn/Variable (DT_FLOAT) [784,10]
fcn/Variable/Adam (DT_FLOAT) [784,10]
fcn/Variable/Adam_1 (DT_FLOAT) [784,10]
fcn/Variable_1 (DT_FLOAT) [10]
fcn/Variable_1/Adam (DT_FLOAT) [10]
fcn/Variable_1/Adam_1 (DT_FLOAT) [10]
optimize/beta1_power (DT_FLOAT) []
optimize/beta2_power (DT_FLOAT) []
```

그래프의 finalize()

`tf.get_default_graph().finalize()`

> Finalizes this graph, making it read-only.
>
> After calling g.finalize(), no new operations can be added to g. This method is used to ensure that no operations are added to a graph when it is shared between multiple threads, for example when using a tf.train.QueueRunner.

In [24]:
```
tf.get_default_graph().finalize()
one_ = tf.constant(1.0)
```

```
---------------------------------------------------------------------
---------
RuntimeError                                 Traceback (most recent c
all last)
<ipython-input-24-b1f25a1fad0d> in <module>()
      1 tf.get_default_graph().finalize()
----> 2 one_ = tf.constant(1.0)

/opt/conda/envs/tensorflow/lib/python2.7/site-packages/tensorflow/
python/framework/constant_op.pyc in constant(value, dtype, shape,
name, verify_shape)
    104   const_tensor = g.create_op(
    105       "Const", [], [dtype_value.type],
--> 106       attrs={"value": tensor_value, "dtype": dtype_value},
name=name).outputs[0]
    107   return const_tensor
    108

/opt/conda/envs/tensorflow/lib/python2.7/site-packages/tensorflow/
python/framework/ops.pyc in create_op(self, op_type, inputs, dtype
s, input_types, name, attrs, op_def, compute_shapes, compute_devic
e)
   2456
   2457     """
-> 2458     self._check_not_finalized()
   2459     for idx, a in enumerate(inputs):
   2460       if not isinstance(a, Tensor):

/opt/conda/envs/tensorflow/lib/python2.7/site-packages/tensorflow/
python/framework/ops.pyc in _check_not_finalized(self)
   2179     """
   2180     if self._finalized:
-> 2181       raise RuntimeError("Graph is finalized and cannot be
modified.")
   2182
   2183   def _add_op(self, op):

RuntimeError: Graph is finalized and cannot be modified.
```

그래프 reset()

tf.reset_default_graph()

> Clears the default graph stack and resets the global default graph.
>
> NOTE: The default graph is a property of the current thread. This function applies only to the current thread. Calling this function while a tf.Session or tf.InteractiveSession is active will result in undefined behavior. Using any previously created tf.Operation or tf.Tensor objects after calling this function will result in undefined behavior.

```
In [25]: tf.reset_default_graph()
         one_ = tf.constant(1.0)
```

텐서 수치오류 체크

tf.check_numerics(tensor,messge)

```
tf.check_numerics(
    tensor,
    message,
    name=None
)
```

> Checks a tensor for NaN and Inf values.
>
> When run, reports an InvalidArgument error if tensor has any values that are not a number (NaN) or infinity (Inf). Otherwise, passes tensor as-is.

```
In [26]: config = tf.ConfigProto(gpu_options={'allow_growth': True})
         sess = tf.InteractiveSession(config=config)
         sess
```

```
Out[26]: <tensorflow.python.client.session.InteractiveSession at 0x11a72731
         0>
```

```
In [27]: tf.get_default_session()
```

```
Out[27]: <tensorflow.python.client.session.InteractiveSession at 0x11a72731
         0>
```

In [28]:
```
zero_ = tf.constant(0.0)
minus_one_ = tf.constant(-1.0)
c_minus_one_ = tf.constant(-1.0,dtype=tf.complex64)
```

In [29]:
```
sess.run(zero_)
```

Out[29]: 0.0

In [30]:
```
minus_one_.eval(), c_minus_one_.eval()
```

Out[30]: (-1.0, (-1+0j))

0 으로 나눈다고 exception 이 바로 발생하지 않습니다

In [31]:
```
div_by_zero_ = tf.div(tf.constant(1.0), zero_)
div_by_zero_.eval()
```

Out[31]: inf

-1 제곱근을 구해도 exception 발생하지 않습니다

In [32]:
```
sqrt_minus_one_ = tf.sqrt(tf.constant(-1.0))
sqrt_minus_one_.eval()
```

Out[32]: nan

잠깐 주의: 텐서플로우는 복소수 타입도 지원합니다.

- 하지만 모델 작성에 사용하실 때는 복소수 타입이 지원 안되는 연산이 많다는 것에 유의 (https://stackoverflow.com/questions/42284904/complex-gradients-on-gpu-in-tensorflow)

In [33]:
```
sqrt_c_minus_one_ = tf.sqrt(tf.constant(-1.0,dtype=tf.complex64))
sqrt_c_minus_one_.eval()
```

Out[33]: (7.5497901e-08+1j)

0 의 로그 값을 계산해도, 음수의 로그값을 계산해도 ... (중략)

In [34]:
```
log_zero_ = tf.log(zero_)
log_zero_.eval()
```

Out[34]: -inf

In [35]:
```
minus_one_ = tf.constant(-1.0)
log_minus_one_ = tf.log(minus_one_)
log_minus_one_.eval()
```

Out[35]: nan

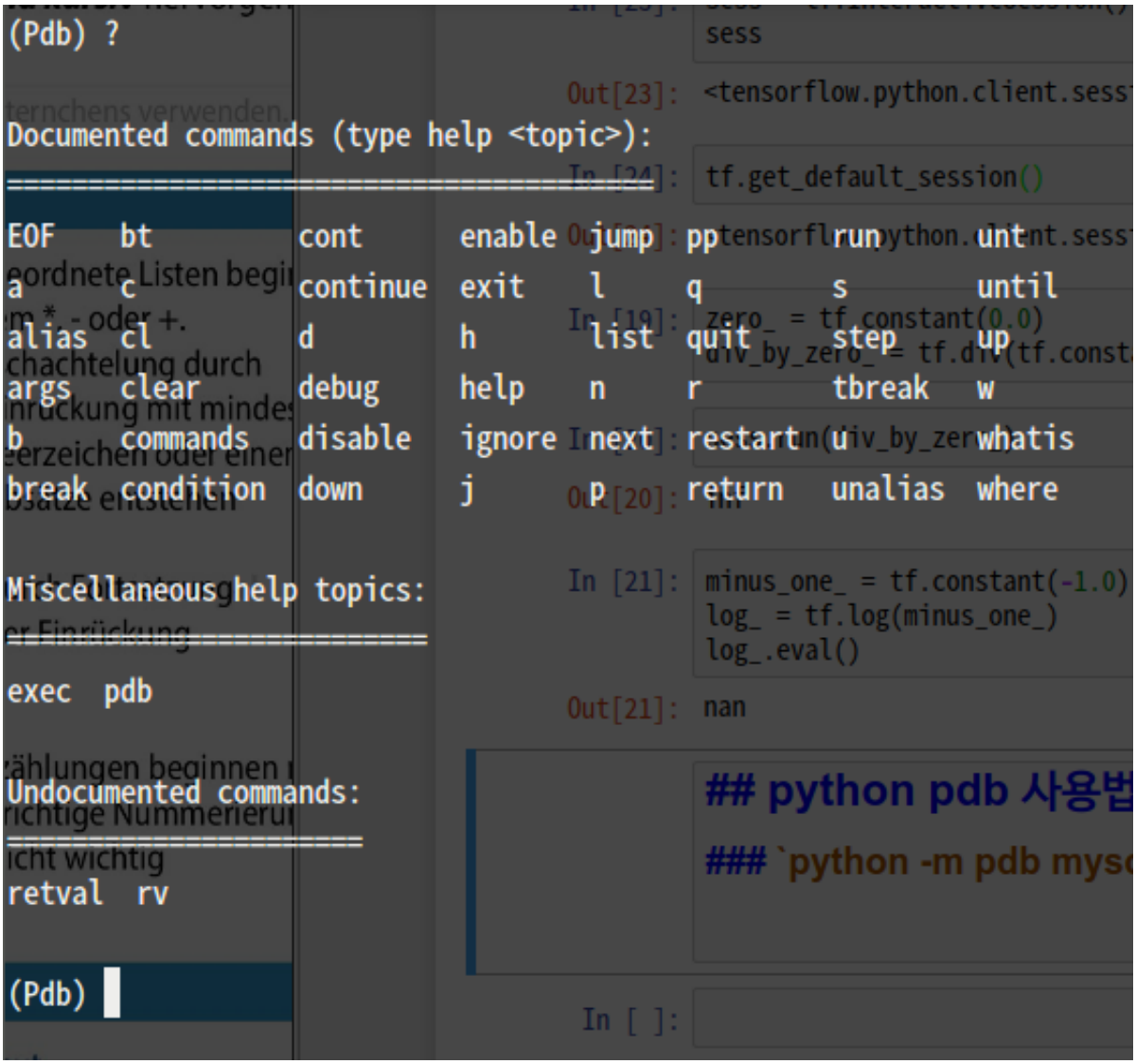
모델 학습 중에 Inf 또는 NaN 이 발생했다면, 외관상 뭔가 계속 학습하는 것 같아도 결과적으로 쓸모없을 수 있음

- `tf.check_numerics(a_tensor)`
  - `a_tensor` 에 NaN 이나 Inf 값이 들어 있는지 확인. 들어있으면 예외발생.
- `tf.add_check_numerics_op()`
  - 모든 부동 소숫점 텐서에 `tf.check_numerics()` 계산을 추가

In [ ]:
```
sess.run([ \
          tf.check_numerics(v,'check_numerics {}'.format(v.name)) \
          for v in [
              zero_,
              minus_one_,
              div_by_zero_,
              sqrt_minus_one_,
              log_zero_,
              minus_one_]])
```

파이썬 디버거 pdb 사용법

- `python -m pdb myscript.py`



- `w(here)`
- `d(own)`
- `u(p)`
- `l(ist)` [*first*[, *last*]]

- `s(tep)`
- `n(ext)`
- `unt(il)`

- `c(ont(inue))`
- `r(eturn)`

- `b(reak)` [[*filename*:]*lineno*|*function*[,*condition*]]

  > Without argument, list all breaks, including for each breakpoint, the number of times that breakpoint has been hit, the current ignore count, and the associated condition if any.

- `disable` [*bpnumber* [*bpnumber* ...]]
- `enable` [*bpnumber* [*bpnumber* ...]]

- `[!]`*statement*

  > Execute the (one-line) statement in the context of the current stack frame. The exclamation point can be omitted unless the first word of the statement resembles a debugger command. To set a global variable, you can prefix the assignment command with a global command on the same line, e.g.:

  ```
  (Pdb) global list_options; list_options = ['-l']
  (Pdb)
  ```

- `p expression`

  > Note print can also be used, but is not a debugger command — this executes the Python print statement.

- `pp expression`

- `a(rgs)`

  > Print the argument list of the current function.

- `run` [*args* ...]

  > Restart the debugged Python program. If an argument is supplied, it is split with “shlex” and the result is used as the new sys.argv. History, breakpoints, actions and debugger options are preserved. “restart” is an alias for “run”.

- `q(uit)`

주피터 노트북에서도 pdb 를 쓸 수 있습니다

- **`debug`**

In [37]:
```
def function_1(x,y):
    return x / y

def function_2(x,y):
    return function_1(x,y) * x

def function_3(x,y):
    return function_2(x,y) + x

function_3(99,0)
```

```
---------------------------------------------------------------------
---------
ZeroDivisionError                         Traceback (most recent c
all last)
<ipython-input-37-161ffdb3c76d> in <module>()
      8     return function_2(x,y) + x
      9
---> 10 function_3(99,0)

<ipython-input-37-161ffdb3c76d> in function_3(x, y)
      6
      7 def function_3(x,y):
----> 8     return function_2(x,y) + x
      9
     10 function_3(99,0)

<ipython-input-37-161ffdb3c76d> in function_2(x, y)
      3
      4 def function_2(x,y):
----> 5     return function_1(x,y) * x
      6
      7 def function_3(x,y):

<ipython-input-37-161ffdb3c76d> in function_1(x, y)
      1 def function_1(x,y):
----> 2     return x / y
      3
      4 def function_2(x,y):
      5     return function_1(x,y) * x

ZeroDivisionError: division by zero
```

In [39]:
```
debug
```

```
> <ipython-input-37-161ffdb3c76d>(2)function_1()
      1 def function_1(x,y):
----> 2     return x / y
      3
      4 def function_2(x,y):
      5     return function_1(x,y) * x

ipdb> q
```

텐서플로우 디버거 tfdbg

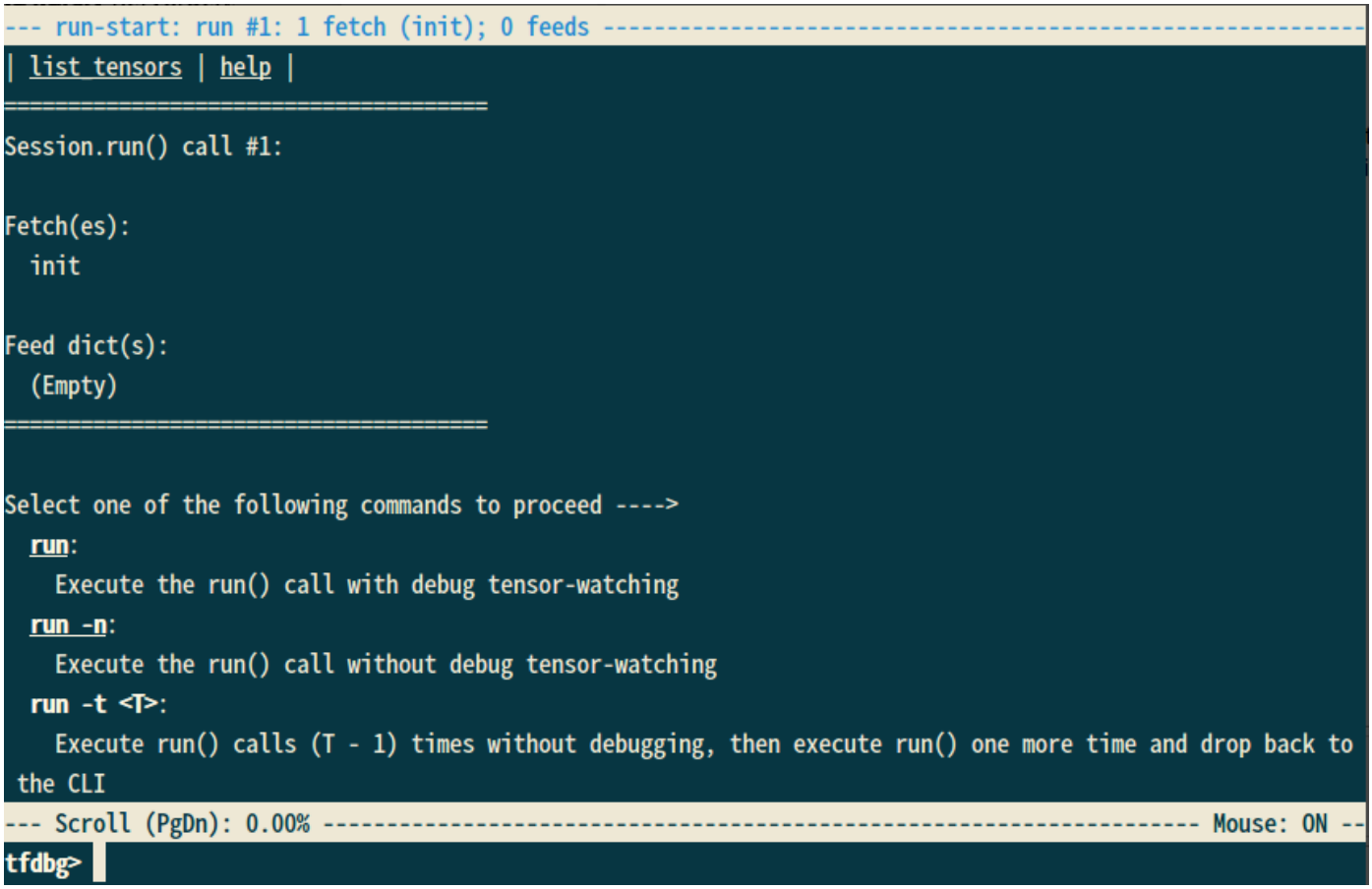
```
from tensorflow.python import debug as tf_debug
```

파일을 열어 보면 training 할 때 다음과 같이 session 을 새로 만들어 쓰는 것을 볼 수 있습니다

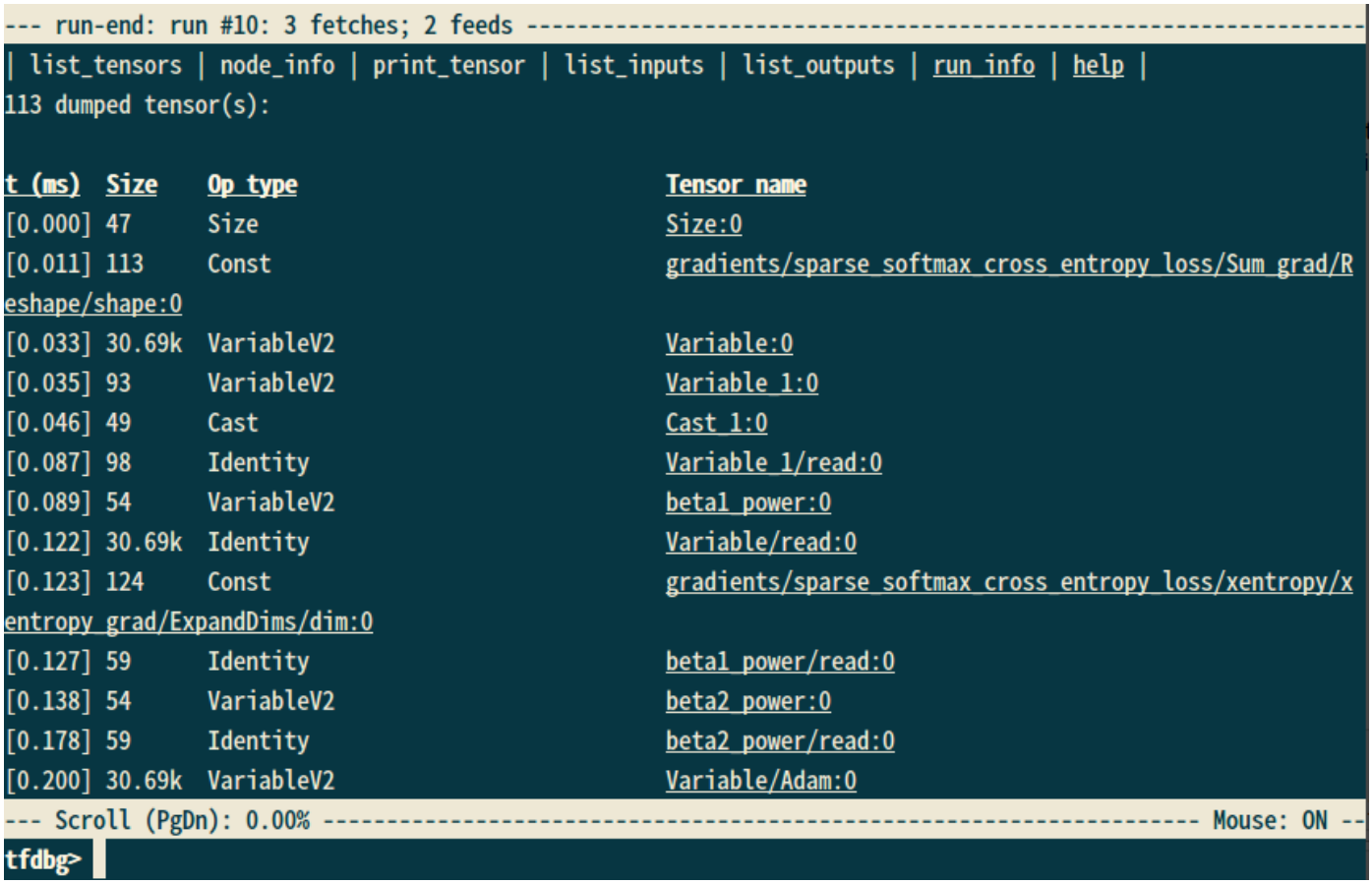
```
config = tf.ConfigProto(gpu_options={'allow_growth': True})
with tf.Session(config=config) as session:

session = tf_debug.LocalCLIDebugWrapperSession(session)
  session.add_tensor_filter('has_inf_or_nan', tf_debug.has_inf_or_nan)


session.run(tf.global_variables_initializer())
</code>
```

실행

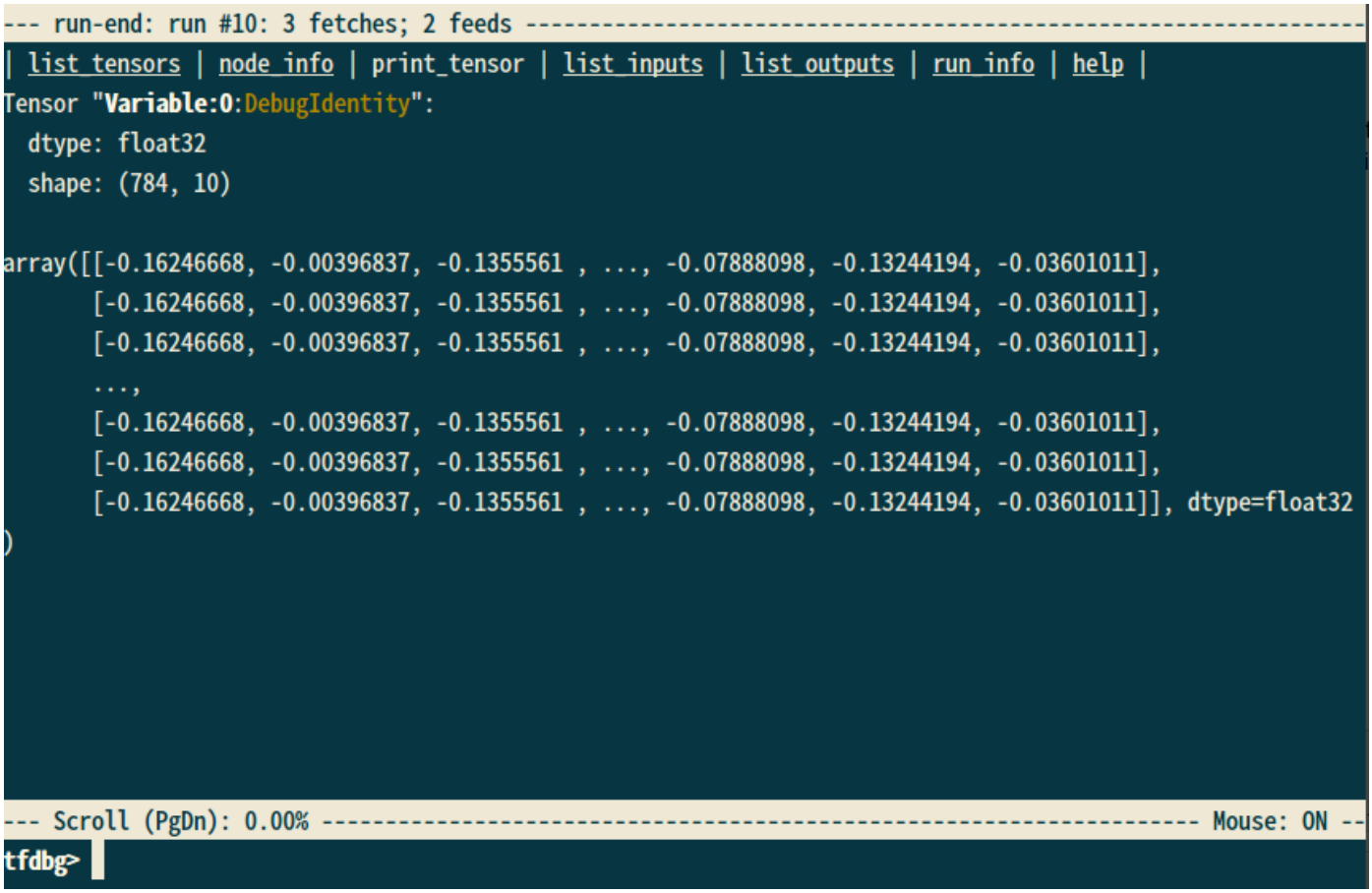
```
python mnist_with_tf_debug.py --debug
```



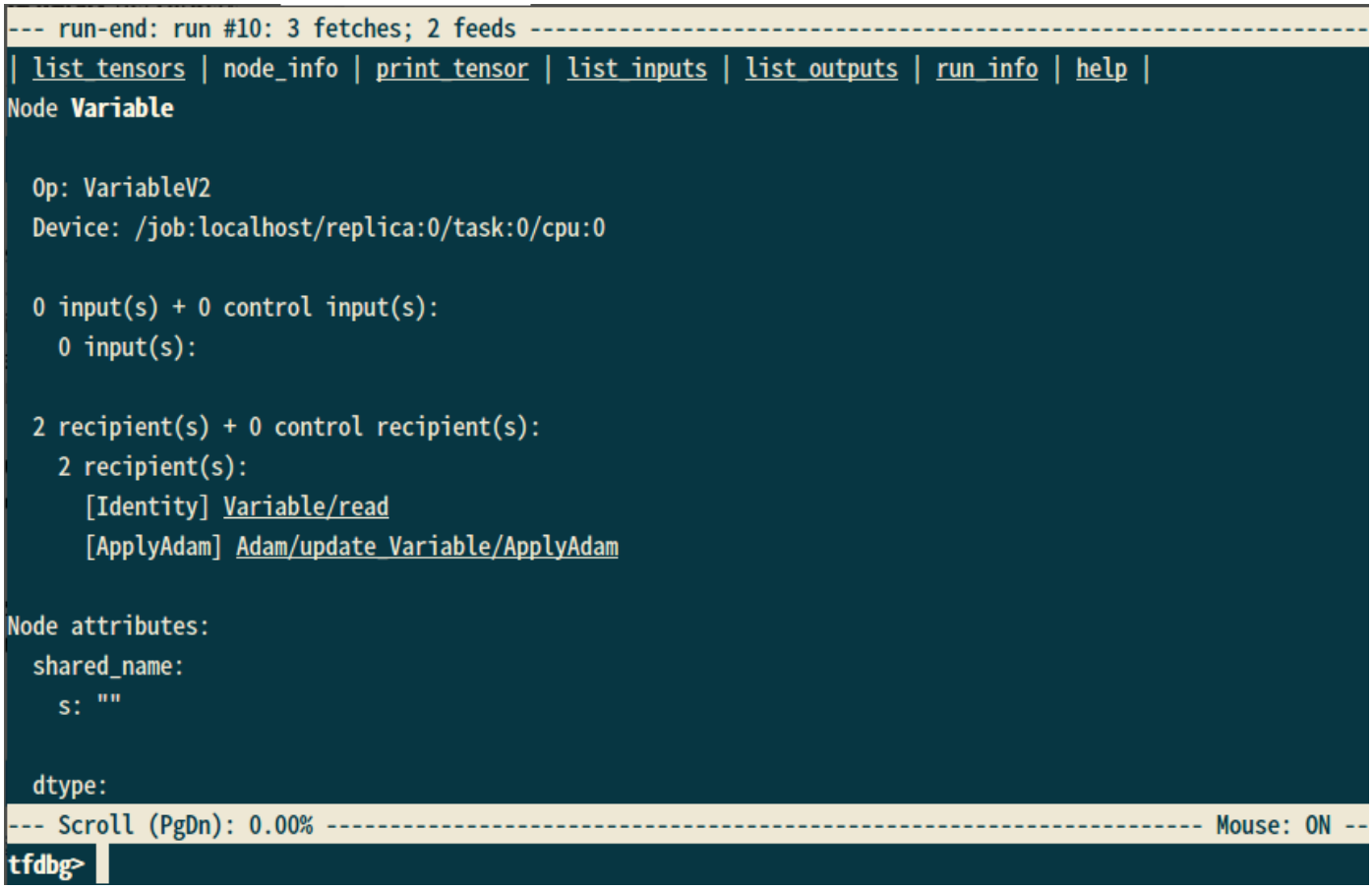

```
tfdbg> run -t 10
```



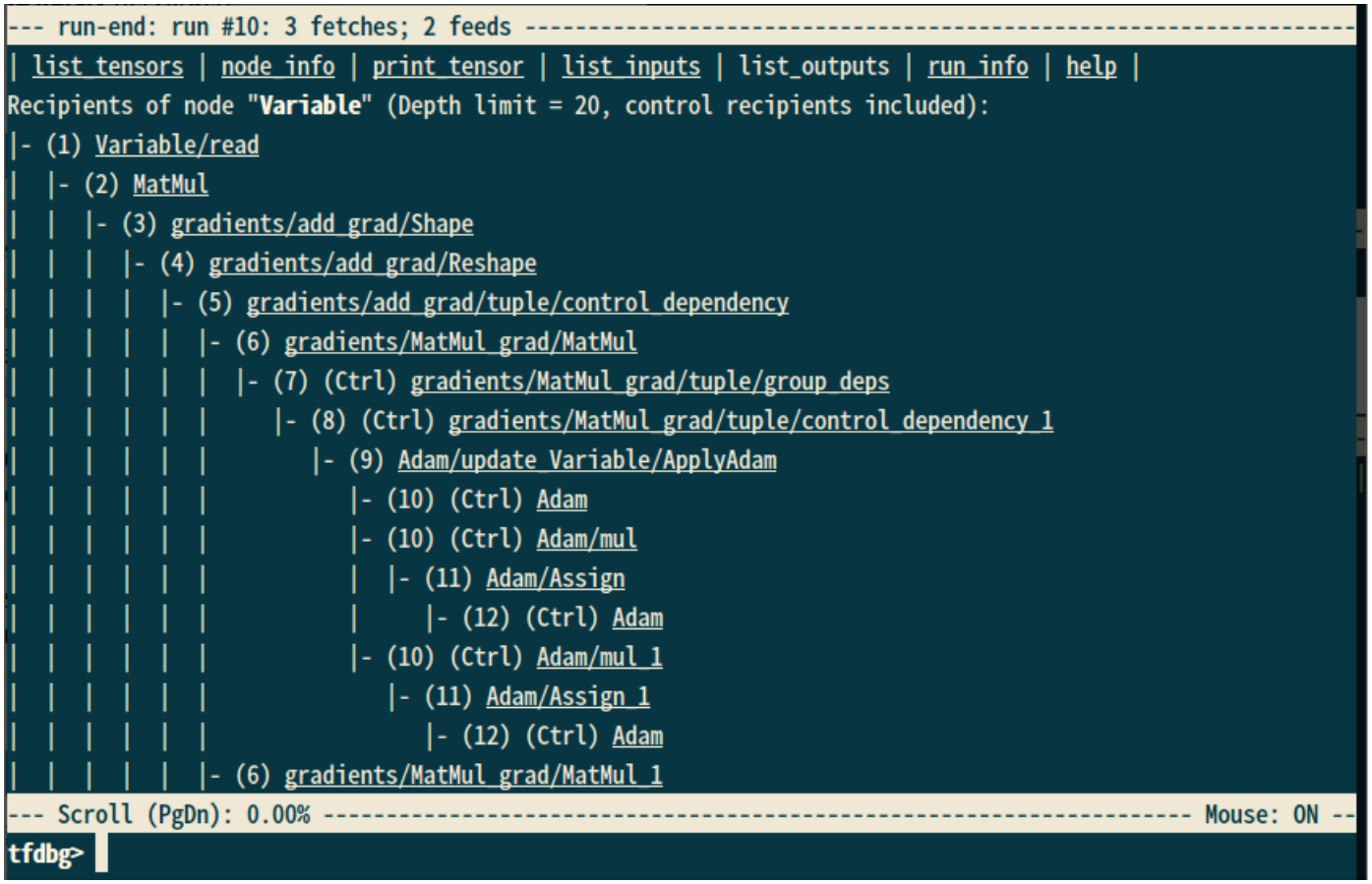

```
tfdbg> print_tensor Variable:0
```



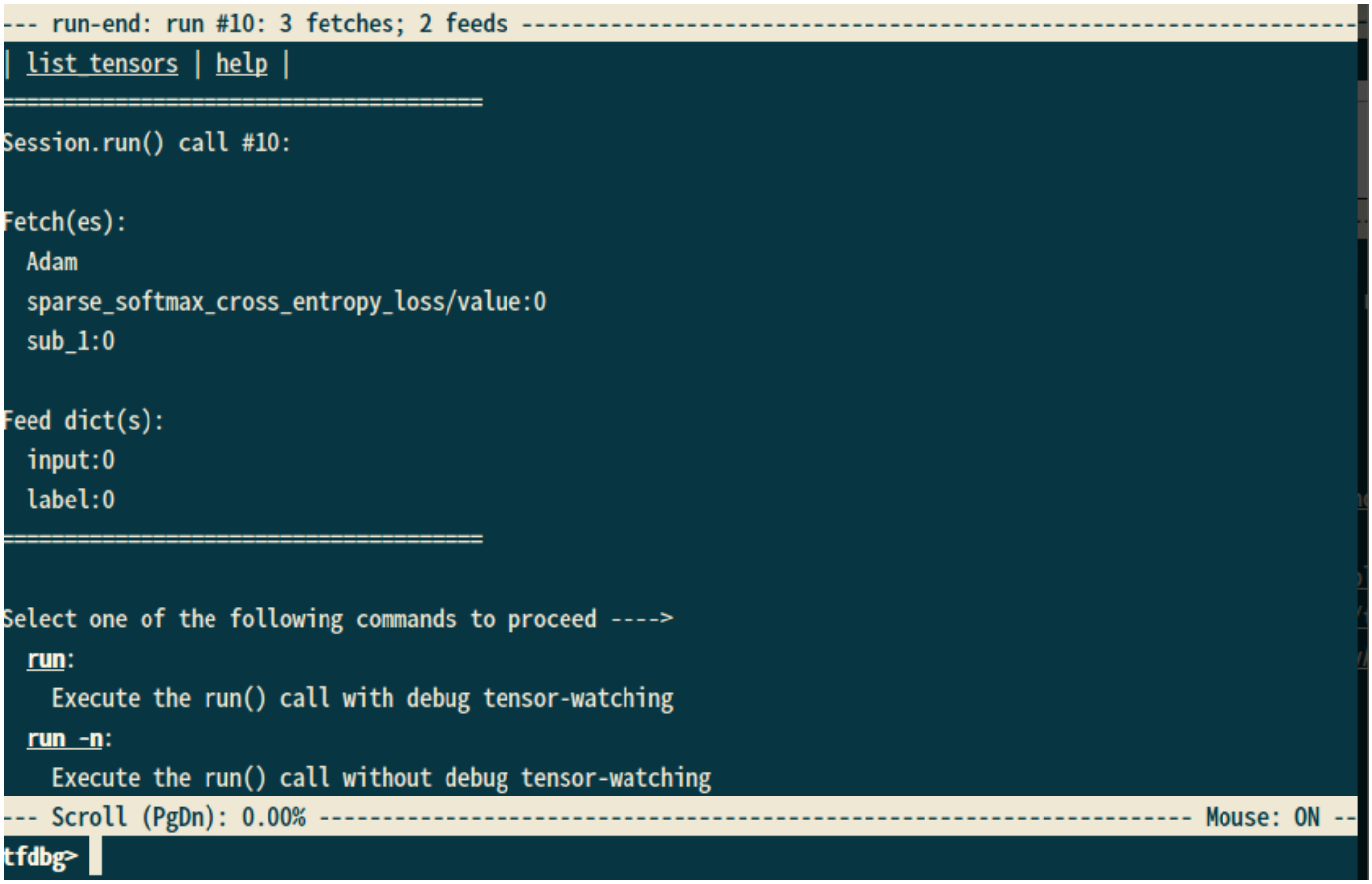

```
tfdbg> node_info Variable:0
```




```
tfdbg> list_outputs Variable:0
```




```
tfdbg> run_info
```


--- run-end: run #10: 3 fetches; 2 feeds ---
| list tensors | help |
Session.run() call #10:
Fetch(es):
Adam
sparse_softmax_cross_entropy_loss/value:0
sub_1:0
Feed dict(s):
input:0
label:0
Select one of the following commands to proceed ---->
run:
Execute the run() call with debug tensor-watching
run -n:
Execute the run() call without debug tensor-watching
--- Scroll (PgDn): 0.00% --- Mouse: ON ---
tfdbg>

기타 참고자료

- A Practical Guide for Debugging TensorFlow Codes, Jongwook Choi
- Jupyter notebooks features
- IPython built-in magic commands
- %quickref, %magic