

## **Manual: Tensorflow Debugging and Visualization Environments**

### **Tensorflow Visualisation**

Tensorboard is essentially a visualization toolkit that can be used to track the evolution of various parameters during training a machine learning system.

The variables that are to be tracked are written into a log directory after having generated summary data in the form of a summary writer;

```
file_writer = tf.summary.FileWriter('/path/to/logs', sess.graph)
```

Once the event files (files where the variables are logged into) are ready, use the command

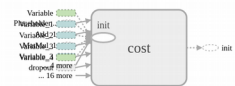
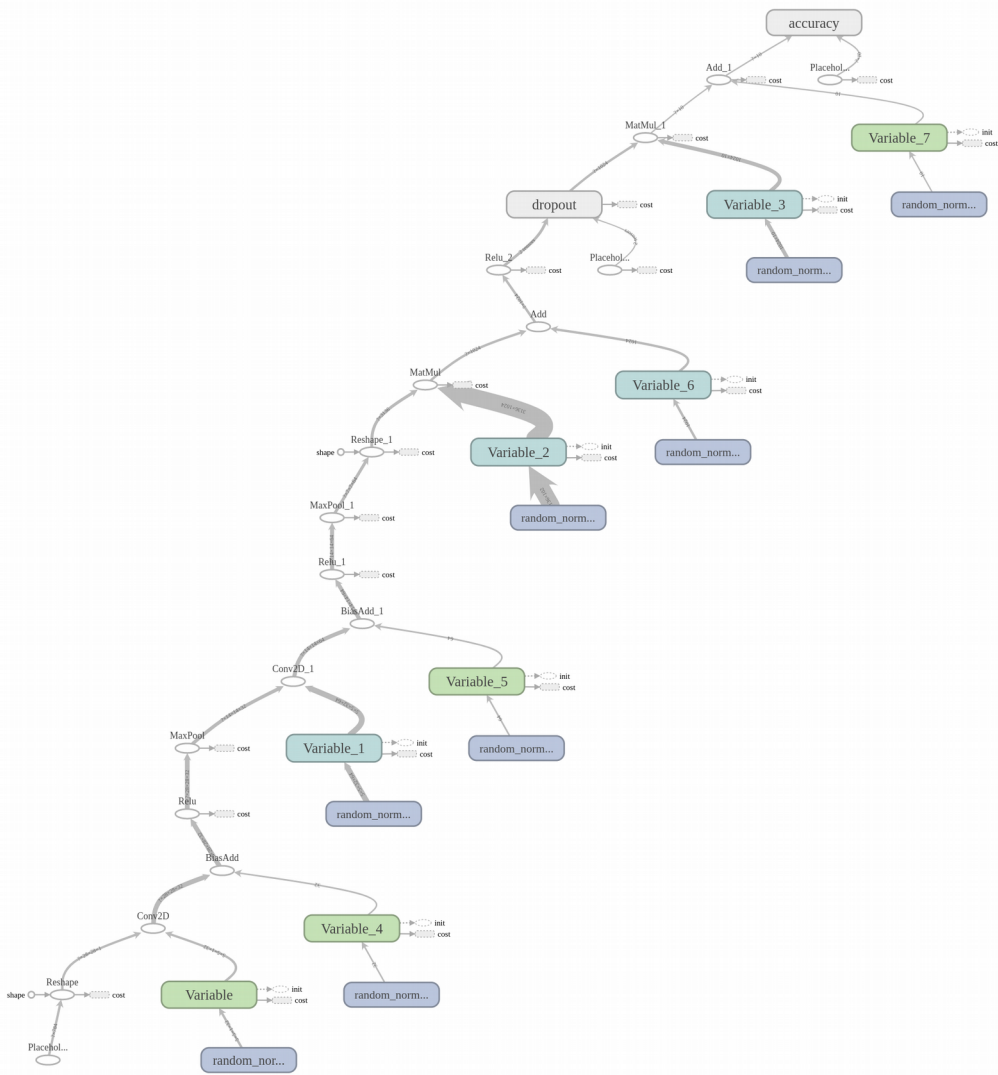
```
$ tensorboard --logdir=path/to/logs --port 6060
```

(The entire command is important as we'll be using another instance of tensorboard for debugging on port 6006)

Also, keep in mind to launch the above command from the Terminal (Linux and Mac OS) or Command Prompt (Windows). While running either Jupyter notebook or Python script.

*Key Concepts:*

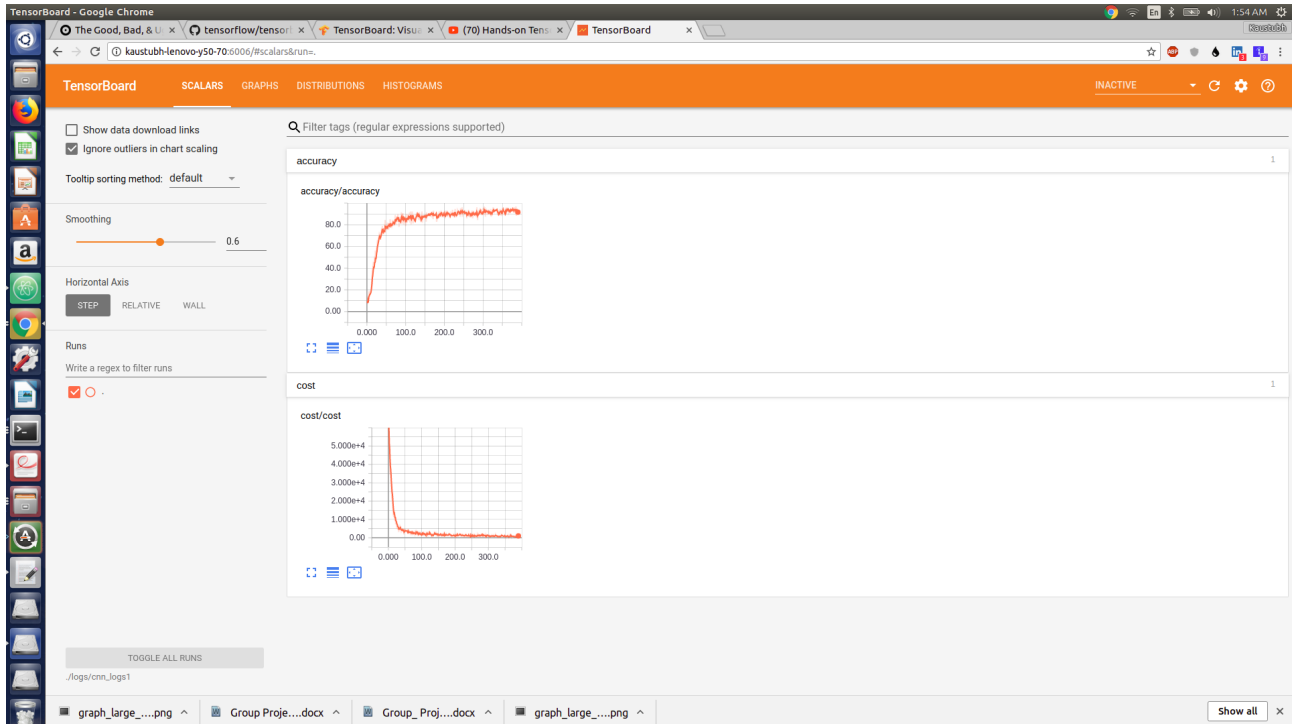
For the *handwritten\_digits\_recognition\_cnn\_5layer.ipynb*, the different (graph, scalar, histogram) visualization are illustrated in the following screenshots.



## Scalar Dashboard

TensorBoard's Scalar Dashboard visualizes scalar statistics that vary over time; for example, you might want to track the model's loss or learning rate.

Accuracy and loss have been tracked.



The `tf.summary.scalar` object is used to log these changes for visualization.

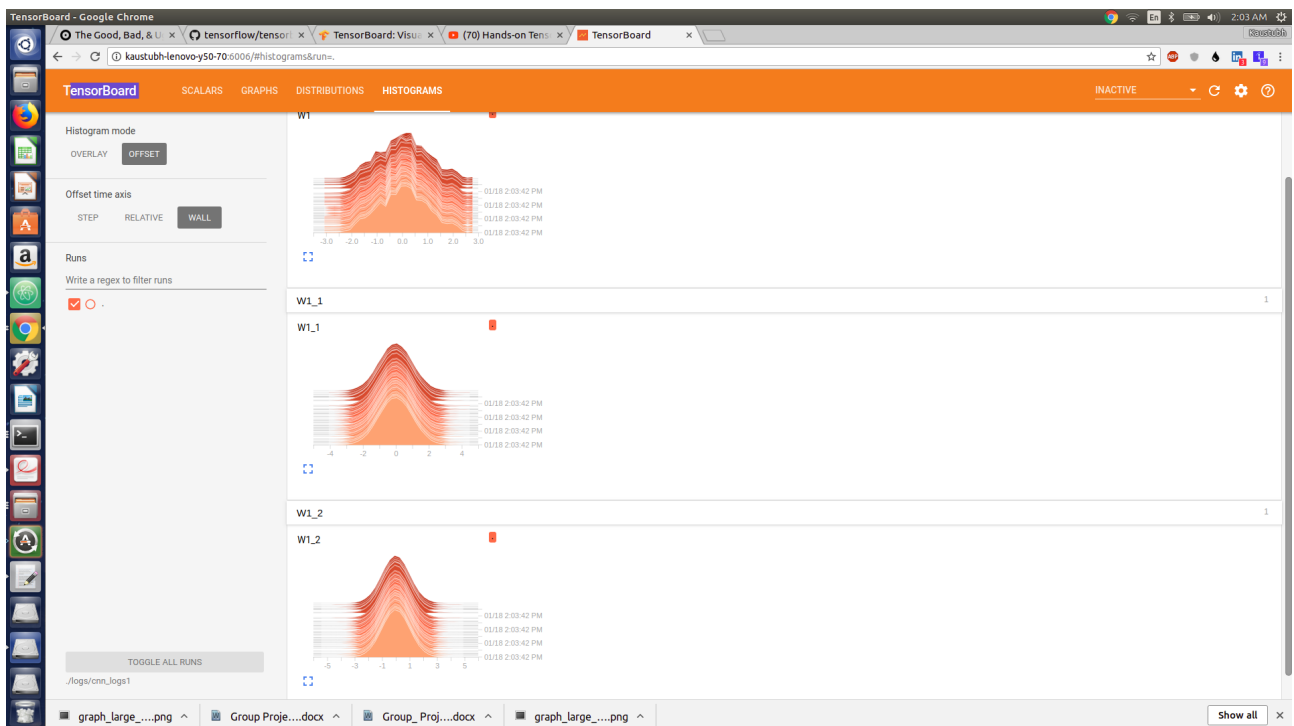
## Histogram Dashboard

The HistogramDashboard displays how the statistical distribution of a Tensor has varied over time. It visualizes data recorded through the `tf.summary.histogram` object.

Each chart shows temporal "slices" of data, where each slice is a histogram of the tensor at a given step.

It's organized with the oldest timestep in the back, and the most recent timestep in front.

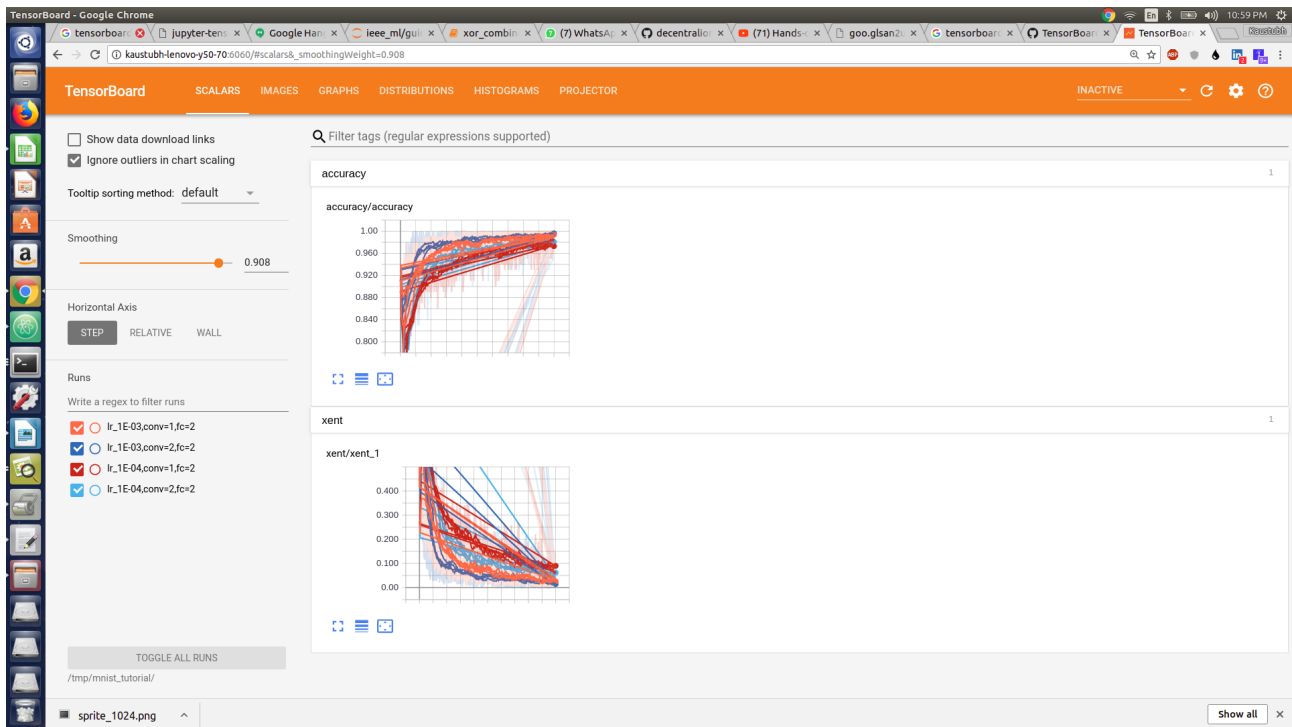
By changing the Histogram Mode from "offset" to "overlay", the perspective will rotate so that every histogram slice is rendered as a line and overlaid with one another.



## Hyperparameter Search

Tensorboard also provides option to visualize the performance of a model for different choices of hyperparameters.

This is done on the file `mnist_hyperparameters.py`



The different color codes correspond to different hyperparameter settings which in this case are [learning rate, number of convolution\_layers, number of fully\_connected\_layers].

For a more detailed instructions for using various functionalities of tensorboard; go to <https://github.com/tensorflow/tensorflow/blob/r1.2/tensorflow/tensorboard/README.md>

## Tensorflow Debugger

### CLI Debugger

#### *XOR\_combined.py*

Firstly, the Tensorflow CLI debugger can be used to debug any running computational graph in Tensorflow very succinctly. This is difficult to achieve with standard debuggers like Python's pdb.

It is best recommended to use TF CLI debugger on .py files.

The *tf\_debug* (*TensorflowDebugger*) is a debugger built for tensorflow that enables you to view the internal structure and states of the Tensorflow computational graphs during training and inference.

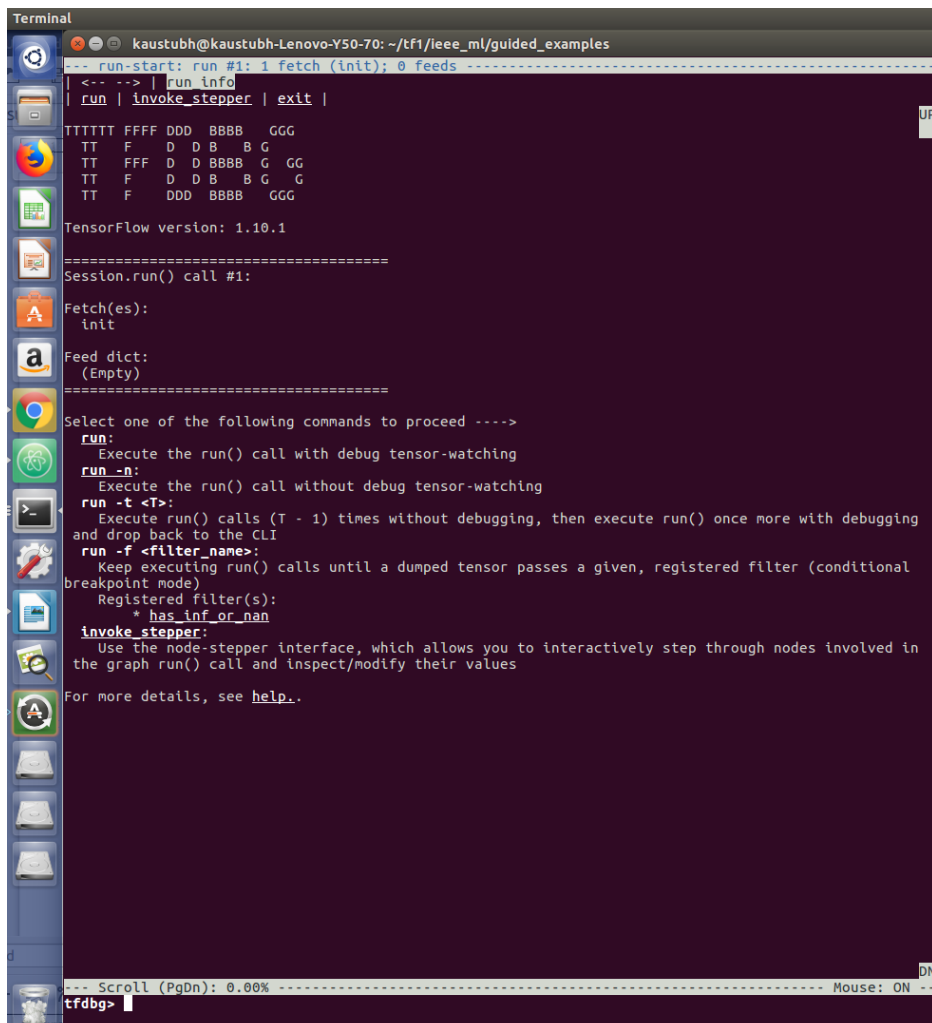
You will have to wrap TF sessions with the tfdbg as:

```
sess = tf_debug.LocalCLIDebugWrapperSession(sess)
```

This wrapper will same interface as Session thus enabling debugging needs no extra code changes.

To launch a TF CLI Debugger on the xor\_combined.py;

```
$ python3 xor_combined.py -debug
```



```
Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/ieee_ml/guided_examples
--- run-start: run #1: 1 fetch (init); 0 feeds ---
<-- --> | run_info
| run | invoke_stepper | exit |
TTTTT FFFF DDD BBBB GGG
TT F D D B B G
TT FFF D D BBBB G GG
TT F D D B B G G
TT F DDD BBBB GGG

TensorFlow version: 1.10.1
=====
Session.run() call #1:
Fetch(es):
  init
Feed dict:
  (Empty)
=====
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
run -t <T>:
  Execute run() calls (T - 1) times without debugging, then execute run() once more with debugging
  and drop back to the CLI
run -f <filter_name>:
  Keep executing run() calls until a dumped tensor passes a given, registered filter (conditional
  breakpoint mode)
  Registered filter(s):
    * has_inf_or_nan
invoke_stepper:
  Use the node-stepper interface, which allows you to interactively step through nodes involved in
  the graph run() call and inspect/modify their values
For more details, see help..

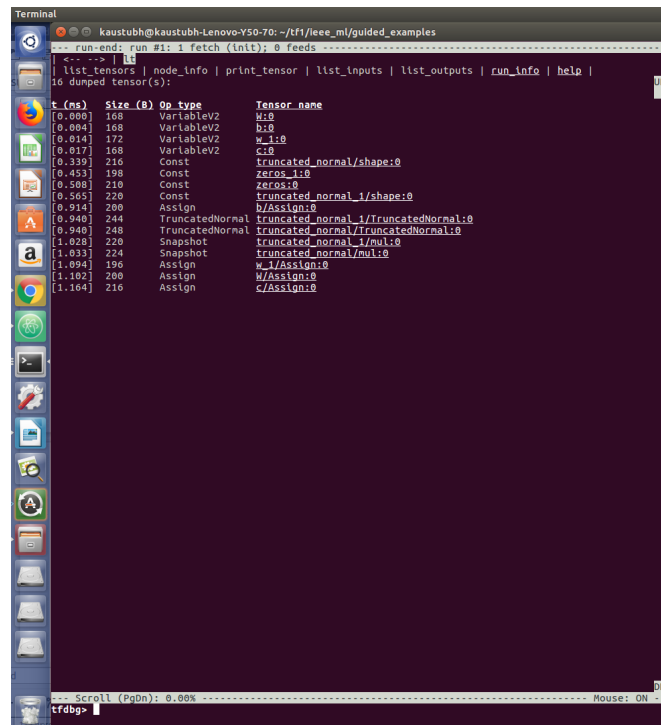
--- Scroll (PgDn): 0.00% --- Mouse: ON ---
tfdbg>
```

The debug wrapper session will prompt you before the first Session.run() call is about to be executed. This is also the run-start CLI. It lists the fetches and the feeds to the current Session.run() call.

Now enter the run command in the terminal;

```
tfdbg> run
```

The run command makes the tf\_debug execute until the end of the next Session.run() call.



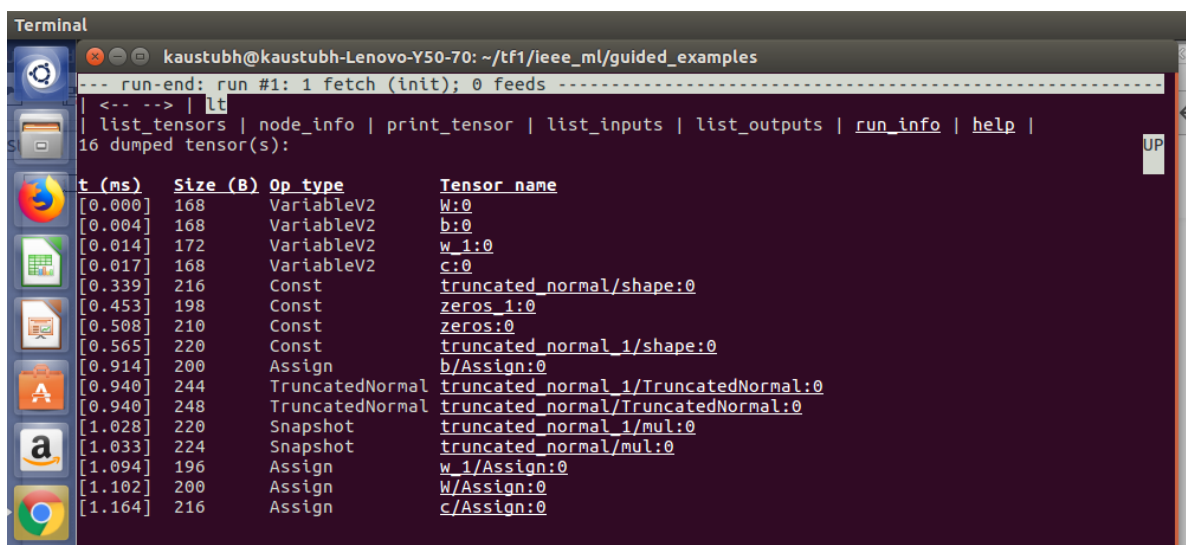
```
Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/ieee_ml/guided_examples
--- run-end: run #1: 1 fetch (init); 0 feeds ---
| <-- --> | lt
| list_tensors | node_info | print_tensor | list_inputs | list_outputs | run_info | help |
16 dumped tensor(s):

t (ms)  Size (B) Op type      Tensor name
[0.000] 168   VariableV2  W:0
[0.004] 168   VariableV2  b:0
[0.014] 172   VariableV2  w_1:0
[0.017] 168   VariableV2  c:0
[0.339] 216   Const      truncated_normal/shape:0
[0.453] 198   Const      zeros_1:0
[0.508] 210   Const      zeros:0
[0.565] 220   Const      truncated_normal_1/shape:0
[0.914] 200   Assign     b/Assign:0
[0.940] 244   TruncatedNormal truncated_normal_1/TruncatedNormal:0
[0.940] 248   TruncatedNormal truncated_normal/TruncatedNormal:0
[1.028] 220   Snapshot   truncated_normal_1/mul:0
[1.033] 224   Snapshot   truncated_normal/mul:0
[1.094] 196   Assign     w_1/Assign:0
[1.102] 200   Assign     W/Assign:0
[1.164] 216   Assign     c/Assign:0

--- Scroll (PgDn): 0.00% --- Mouse: ON ---
tfdbg>
```

The above screenshot shows the intermediate tensors from the previous Session.run() call.

For one such tensor, truncated\_normal/mul



```
Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/ieee_ml/guided_examples
--- run-end: run #1: 1 fetch (init); 0 feeds ---
| <-- --> | lt
| list_tensors | node_info | print_tensor | list_inputs | list_outputs | run_info | help |
16 dumped tensor(s):

t (ms)  Size (B) Op type      Tensor name
[0.000] 168   VariableV2  W:0
[0.004] 168   VariableV2  b:0
[0.014] 172   VariableV2  w_1:0
[0.017] 168   VariableV2  c:0
[0.339] 216   Const      truncated_normal/shape:0
[0.453] 198   Const      zeros_1:0
[0.508] 210   Const      zeros:0
[0.565] 220   Const      truncated_normal_1/shape:0
[0.914] 200   Assign     b/Assign:0
[0.940] 244   TruncatedNormal truncated_normal_1/TruncatedNormal:0
[0.940] 248   TruncatedNormal truncated_normal/TruncatedNormal:0
[1.028] 220   Snapshot   truncated_normal_1/mul:0
[1.033] 224   Snapshot   truncated_normal/mul:0
[1.094] 196   Assign     w_1/Assign:0
[1.102] 200   Assign     W/Assign:0
[1.164] 216   Assign     c/Assign:0
```

Now, we can use the `node_info` option to check up the type and attributes of this graph node as shown

```
Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/ieee_ml/guided_examples
--- run-end: run #1: 1 fetch (init); 0 feeds ---
| <-- --> | node info -a -d -t truncated_normal/mul
| list_tensors | node info | print_tensor | list_inputs | list_outputs | run_info | help |
Node truncated_normal/mul
Op: Snapshot
Device: /job:localhost/replica:0/task:0/device:CPU:0
1 input(s) + 0 control input(s):
1 input(s):
[TruncatedNormal] truncated_normal/TruncatedNormal
1 recipient(s) + 0 control recipient(s):
1 recipient(s):
[Assign] W/Assign
Node attributes:
T:
type: DT_FLOAT
1 dumped tensor(s):
Slot 0 @ DebugIdentity @ 1.033 ms
Traceback of node construction:
0: xor_combined.py
Line: 55
Function: <module>
Text: "W = tf.Variable(tf.truncated_normal([2,2]), name = "W")"
1: /usr/local/lib/python3.5/dist-packages/tensorflow/python/ops/random_ops.py
Line: 175
Function: truncated_normal
Text: "mul = rnd * stddev_tensor"
2: /usr/local/lib/python3.5/dist-packages/tensorflow/python/ops/math_ops.py
Line: 850
Function: binary_op_wrapper
Text: "return func(x, y, name=name)"
3: /usr/local/lib/python3.5/dist-packages/tensorflow/python/ops/math_ops.py
Line: 1094
Function: _mul_dispatch
Text: "return gen_math_ops.mul(x, y, name=name)"
4: /usr/local/lib/python3.5/dist-packages/tensorflow/python/ops/gen_math_ops.py
Line: 4936
Function: mul
Text: "'Mul', x=x, y=y, name=name)"
5: /usr/local/lib/python3.5/dist-packages/tensorflow/python/framework/op_def_library.py
Line: 787
Function: _apply_op_helper
Text: "op_def=op_def)"
6: /usr/local/lib/python3.5/dist-packages/tensorflow/python/util/deprecation.py
--- Scroll (PgDn): 0.00% --- Mouse: ON ---
tfdbg>
```



The *list\_inputs* and *list\_outputs* give the transitive inbound and outbound tensors of a given node.

```

Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/leee_ml/guided_examples
run-end: run #1: 1 fetch (init); 0 feeds -----
<--> | list_outputs -c -r truncated_normal/mul
| list_tensors | node_info | print_tensor | list_inputs | list_outputs | run_info | help |
Recipients of node "truncated_normal/mul" (Depth limit = 20, control recipients included):
|- (1) WAssign
|- (2) (Ctrl) init
Legend:
(d): recursion depth = d.
(Ctrl): Control input.
tfdbg>

```

*list\_inputs*

```

Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/leee_ml/guided_examples
run-end: run #1: 1 fetch (init); 0 feeds -----
<--> | list_inputs -c -r truncated_normal/mul
| list_tensors | node_info | print_tensor | list_inputs | list_outputs | run_info | help |
Inputs to node "truncated_normal/mul" (Depth limit = 20, control inputs included):
|- (1) truncated_normal/truncated_normal
|- (2) truncated_normal/shape
Legend:
(d): recursion depth = d.
(Ctrl): Control input.
tfdbg>

```

*list out\_puts*

Also, conditional breakpoints feature of *tf\_debug* can be used to let code run until certain cases/conditions are satisfied on the graph.

In this case, let us consider the case the model runs until values like *inf* and *nan* are encountered. For this, in the command line enter;

```
tfdbg> run -f has_inf_or_nan
```

For this example, on *xor\_combined.py*, all the epochs are run as XOR mapping using an ANN is relatively simple than image classification tasks and hence did not run into any issues.

```

Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/leee_ml/guided_examples
kaustubh@kaustubh-Lenovo-Y50-70:~/tf1$ cd leee_ml/
kaustubh@kaustubh-Lenovo-Y50-70:~/tf1/leee_ml$ cd guided_examples/
kaustubh@kaustubh-Lenovo-Y50-70:~/tf1/leee_ml/guided_examples$ python3 xor_combined.py
2018-09-21 18:59:18.359200: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
^Z
[1]+  Stopped                  python3 xor_combined.py
kaustubh@kaustubh-Lenovo-Y50-70:~/tf1/leee_ml/guided_examples$ python3 xor_combined.py
2018-09-21 19:02:39.492652: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
^Z
[2]+  Stopped                  python3 xor_combined.py
kaustubh@kaustubh-Lenovo-Y50-70:~/tf1/leee_ml/guided_examples$ python3 xor_combined.py
2018-09-21 19:03:02.357818: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
^Z
Epoch: 0
y_estimated:
[0.49984378]
[0.85242045]
[0.6522666]
[0.9152757]
W:
[-0.31868675  0.42915654]
[-0.01389174  1.1991655]
C:
[0. 0.]
[0. 0.00013563]
[0. 0.00057755]
[0. 0.00051821]
W:
[-1.4329072]
[1.4622012]
b
[-0.000623]
[0.273729e-05]
[0.00039491]
[-0.00035433]
loss: 0.30757833
^Z
[3]+  Stopped                  python3 xor_combined.py
kaustubh@kaustubh-Lenovo-Y50-70:~/tf1/leee_ml/guided_examples$ python3 cnn_mnist.py
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow/contrib/learn/python/learn/datasets/_init_.py:80: load_mnist (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use tf.data
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow/contrib/learn/python/learn/datasets/_init_.py:80: load_mnist (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use alternatives such as official/mnist/dataset.py from tensorflow/models.
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:260: read_data_sets (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use alternatives such as official/mnist/dataset.py from tensorflow/models.
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:260: maybe_download (from tensorflow.contrib.learn.python.learn.datasets.base) is deprecated and will be removed in a future version.

```

***cnn\_mnist.py***

```
kaustubh@kaustubh-Lenovo-Y50-70: ~/f1/f1/leeds/examples$ python run.py --run_name=run #1: 2 epochs; 8 feeds
--> | List outputs -c -r conv2d_1/kernel
list_tensors | node_info | print_tensor | list_inputs | list_outputs | run_info | help |
Recipients of node "conv2d_1/kernel" (Depth limit = 20, control recipients included):
(1) conv2d_1/kernel/read
|- (2) conv2d_1/conv2d
|- (3) conv2d_1/BiasAdd
|- (4) conv2d_1/Relu
|- (5) max_pooling2d_1/MaxPool
|- (6) Reshape_1
|- (7) dense/RatMul
|- (8) dense/BiasAdd
|- (9) dense/Relu
|- (10) gradients/dense/Relu_grad/ReluGrad
|- (11) gradients/dense/RatMul_grad/RatMul
|- (12) gradients/Reshape_1_grad/Reshape
|- (13) gradients/max_pooling2d_1/MaxPool_grad/MaxPoolGrad
|- (14) gradients/conv2d_1/Relu_grad/ReluGrad
|- (15) gradients/conv2d_1/Conv2D_grad/Conv2DBackpropFilter
|- (16) GradientDescent/update_conv2d_1/kernel/ApplyGradientDescent
|- (17) (Ctrl) GradientDescent/value
|- (18) GradientDescent
|- (16) (ctrl) gradients/max_pooling2d/MaxPool_grad/MaxPoolGrad
|- (17) gradients/conv2d/Relu_grad/ReluGrad
|- (18) gradients/conv2d/BiasAdd_grad/BiasAddGrad
|- (19) GradientDescent/update_conv2d/Bias/ApplyGradientDescent
|- (20) (ctrl) GradientDescent/value
|-. ...
|- (19) (ctrl) gradients/conv2d/Conv2D_grad/Conv2DBackpropFilter
|- (20) GradientDescent/update_conv2d/kernel/ApplyGradientDescent
|-. ...
|- (18) gradients/conv2d/conv2d_grad/Conv2DBackpropFilter
|- (19) GradientDescent/update_conv2d/kernel/ApplyGradientDescent
|- (20) (ctrl) GradientDescent/value
|-. ...
|- (15) gradients/conv2d_1/Conv2D_grad/Conv2DBackpropInput
|- (16) gradients/max_pooling2d/MaxPool_grad/MaxPoolGrad
|- (17) gradients/conv2d/Relu_grad/ReluGrad
|- (18) gradients/conv2d/BiasAdd_grad/BiasAddGrad
|- (19) GradientDescent/update_conv2d/Bias/ApplyGradientDescent
|- (20) (ctrl) GradientDescent/value
|-. ...
|- (19) (ctrl) gradients/conv2d/conv2d_grad/Conv2DBackpropFilter
|- (20) GradientDescent/update_conv2d/kernel/ApplyGradientDescent
|-. ...
|- (18) gradients/conv2d/conv2d_grad/Conv2DBackpropFilter
|- (19) GradientDescent/update_conv2d/kernel/ApplyGradientDescent
|- (20) (ctrl) GradientDescent/value
|-. ...
|- (16) (ctrl) GradientDescent/update_conv2d_1/kernel/ApplyGradientDescent
|- (17) (ctrl) GradientDescent/value
|- (18) GradientDescent
|- (15) gradients/conv2d_1/BiasAdd_grad/BiasAddGrad
|- (16) GradientDescent/update_conv2d_1/bias/ApplyGradientDescent
|- (17) (ctrl) GradientDescent/value
|- (18) GradientDescent
|- (16) (ctrl) gradients/conv2d_1/conv2d_grad/Conv2DBackpropFilter
|- (17) GradientDescent/update_conv2d_1/kernel/ApplyGradientDescent
|- (18) (ctrl) GradientDescent/value
```

```
> tfdbg run -f has_inf_or_nan
```

Also, regex search can be used to find desired tensors in the CLI debug window.

```
tfdbg> (drop) ---> Searching for the regular expression "drop"
```

```

kaustubh@kaustubh-Lenovo-Y50-70: ~/tf/lee_m/guided_examples
-- run-end: run #1: 2 fetches; 0 feeds -----
tfdbg>
tfdbg> list_tensors | node_info | print_tensor | list_inputs | list_outputs | run_info | help |
[1.646] 252 Const enqueue_input/concat/div_grad/Shape:0
[1.773] 227 IsVariableInitialized global_step/IsVariableInitialized:0
[1.792] 3.33k Identity conv2d/kernel/read:0
[1.968] 4.19k Identity dense_1/bias/read:0
[2.026] 40.20k VariableV2 dense_1/kernel:0
[2.295] 222 Const GradientDescent/learning_rate:0
[2.326] 224 RefSwitch global_step/cond/read/Switch:1
[2.339] 228 VariableV2 dense_1/bias:0
[2.426] 371 RandomShuffleQueueV2 enqueue_input/random_shuffle_queue:0
[2.599] 419 Const enqueue_input/queue/enqueue_input/random_shuffle_queuefraction_over_250_of_750_full/tags:0
[2.617] 238 Identity enqueue_input/queue:0
[2.900] 242 QueueSizeV2 enqueue_input/random_shuffle_queue_Size:0
[2.993] 40.21k Identity dense_1/kernel/read:0
[3.144] 210 Const enqueue_input/Maximun/x:0
[3.307] 202 Const enqueue_input/sub/y:0
[3.433] 200.22k Identity conv2d_1/kernel/read:0
[3.510] 202 Const enqueue_input/mul/y:0
[3.562] 198 Sub enqueue_input/sub:0
[3.658] 182 Const loss/tags:0
[3.778] 266 Maximum enqueue_input/Maximun:0
[3.784] 226 Const global_step/initializer/zeros:0
[3.790] 400.25k RandomUniform dropout/dropout/random_uniform/RandomUniform:0
[3.899] 206 Const Reshape/shape:0
[3.912] 200 Cast enqueue_input/Cast:0
[4.010] 202 Const Reshape_1/shape:0
[4.063] 198 Mul enqueue_input/mul:0
[4.134] 214 Const dropout/dropout/keep_prob:0
[4.182] 418 ScalarSummary enqueue_input/queue/enqueue_input/random_shuffle_queuefraction_over_250_of_750_full:0
[4.262] 232 Const random_shuffle_queue_DequeueMany/n:0
[4.371] 250 Const sparse_softmax_cross_entropy_loss/Const_1:0
[4.489] 252 Const sparse_softmax_cross_entropy_loss/zeros_like:0
[4.570] 243 Const sparse_softmax_cross_entropy_loss/zeros_like:0
[4.674] 218 Switch global_step/cond/Switch_1:1
[4.783] 208 Merge global_step/cond/Merge:1
[4.860] 242 Merge global_step/cond/Merge:0
[4.939] 198 Snapshot global_step/add:0
[5.078] 1.01k QueueDequeueManyV2 random_shuffle_queue_DequeueMany:0
[5.153] 630 QueueDequeueManyV2 random_shuffle_queue_DequeueMany:2
[5.240] 400.24k Snapshot dropout/dropout/random_uniform/mul:0
[5.305] 306.48k QueueDequeueManyV2 random_shuffle_queue_DequeueMany:1
[5.417] 400.23k Snapshot dropout/dropout/random_uniform:0
[5.515] 306.44k Reshape Reshape:0
[5.600] 400.21k Add dropout/dropout/add:0
[5.680] 400.21k Floor dropout/dropout/Floor:0
[5.760] 12.25M VariableV2 dense/kernel:0
[5.840] 9.57M Conv2D conv2d/conv2d:0
[5.920] 12.25M Identity conv2d/kernel/read:0
[6.000] 9.57M BiasAdd conv2d/BiasAdd:0
[6.080] 9.57M Relu conv2d/Relu:0
[6.160] 2.39M MaxPool max_pooling2d/MaxPool:0
[6.240] 4.79M Conv2D conv2d_1/conv2d:0
[6.320] 4.79M BiasAdd conv2d_1/BiasAdd:0
[6.400] 4.79M Relu conv2d_1/Relu:0
[6.480] 1.20M MaxPool max_pooling2d_1/MaxPool:0
[6.560] 1.20M Reshape Reshape_1:0
[6.640] 400.19k Reshape Reshape/mul:0
tfdbg>

```

Now we shall see a way to step through the nodes of graph one-by-one in a manner analogous to procedural languages debuggers like GDB and PDB.

*tfdbg> invoke\_stepper*

*tfdbg> s* ----> Call once for stepping through each node in the graph.

```

kaustubh@kaustubh-Lenovo-Y50-70: ~/tf/lee_m/guided_examples
tfdbg> invoke_stepper
tfdbg> s
Topologically-sorted transitive input(s) and fetch(es):
--> (1 / 201) | | gradients/Shape
(2 / 201) | | sparse_softmax_cross_entropy_loss/assert_broadcastable/static_scalar_check_success
(3 / 201) | | sparse_softmax_cross_entropy_loss/Equal/y
(4 / 201) | | sparse_softmax_cross_entropy_loss/Greater/y
(5 / 201) | | sparse_softmax_cross_entropy_loss/num_present/ones_like/Const
(6 / 201) | | sparse_softmax_cross_entropy_loss/num_present/zeros_like
(7 / 201) | | sparse_softmax_cross_entropy_loss/ones_like/Const
(8 / 201) | | sparse_softmax_cross_entropy_loss/Const_1
(9 / 201) | | sparse_softmax_cross_entropy_loss/num_present/broadcast_weights/assert_broadcastable/static_scalar_check_success
(10 / 201) | | sparse_softmax_cross_entropy_loss/num_present/broadcast_weights/ones_like/Const
(11 / 201) | | sparse_softmax_cross_entropy_loss/num_present/broadcast_weights/ones_like/Shape
(12 / 201) | | sparse_softmax_cross_entropy_loss/num_present/broadcast_weights/ones_like
(13 / 201) | | sparse_softmax_cross_entropy_loss/num_present/Equal/y
(14 / 201) | | sparse_softmax_cross_entropy_loss/zeros_like
(15 / 201) | | sparse_softmax_cross_entropy_loss/ones_like/Shape
(16 / 201) | | sparse_softmax_cross_entropy_loss/ones_like
(17 / 201) | | sparse_softmax_cross_entropy_loss/Const_2
(18 / 201) | | sparse_softmax_cross_entropy_loss/num_present/ones_like/Shape
(19 / 201) | | sparse_softmax_cross_entropy_loss/num_present/ones_like
(20 / 201) | | sparse_softmax_cross_entropy_loss/num_present/Const
(21 / 201) | | GradientDescent/learning_rate
(22 / 201) | | gradients/sparse_softmax_cross_entropy_loss/Sum_1_grad/Const
(23 / 201) | | conv2d_1/bias
(24 / 201) | | conv2d_1/bias/read
(25 / 201) | | gradients/sparse_softmax_cross_entropy_loss/Sum_1_grad/Reshape/shape
(26 / 201) | | dropout/dropout/random_uniform/min
(27 / 201) | | dense_1/bias
(28 / 201) | | dense_1/bias/read
(29 / 201) | | random_shuffle_queue_DequeueMany/n
(30 / 201) | | Reshape_1/shape
(31 / 201) | | gradients/grad_ys_0
(32 / 201) | | gradients/Full
(33 / 201) | | conv2d/kernel
(34 / 201) | | conv2d/kernel/read
(35 / 201) | | gradients/sparse_softmax_cross_entropy_loss/div_grad/Shape_1
(36 / 201) | | enqueue_input/random_shuffle_queue
(37 / 201) | | random_shuffle_queue_DequeueMany
(38 / 201) | | gradients/sparse_softmax_cross_entropy_loss/Sum_grad/Const
(39 / 201) | | gradients/sparse_softmax_cross_entropy_loss/xentropy/xentropy_grad/ExpandDims/dim
(40 / 201) | | conv2d/bias
(41 / 201) | | conv2d/bias/read
(42 / 201) | | dense/kernel
(43 / 201) | | dense/kernel/read
(44 / 201) | | gradients/sparse_softmax_cross_entropy_loss/Sum_grad/Reshape/shape
(45 / 201) | | gradients/dropout/dropout/div_grad/Shape_1
(46 / 201) | | gradients/dropout/dropout/div_grad/Shape
(47 / 201) | | gradients/dropout/dropout/div_grad/BroadcastGradientArgs
(48 / 201) | | gradients/sparse_softmax_cross_entropy_loss/Mul_grad/Shape_1
(49 / 201) | | dense/bias
(50 / 201) | | dense/bias/read
(51 / 201) | | dense_1/kernel
(52 / 201) | | dense_1/kernel/read
(53 / 201) | | Reshape/shape
(54 / 201) | | Reshape
(55 / 201) | | gradients/conv2d/conv2d_grad/Shape/m
tfdbg>

```

The above screenshot shows the CLI after running the *invoke\_stepper*.

UP

```

Terminal
kaustubh@kaustubh-Lenovo-Y50-70: ~/tf1/ieee_ml/guided_examples
--- Node Stepper: run #2: 2 fetches; 0 feeds -----
| <-- --> | S
(21 / 201) [ H ] GradientDescent/learning_rate
(22 / 201) [ H ] gradients/sparse_softmax_cross_entropy_loss/Sum_1_grad/Const
-->(23 / 201) [ H ] conv2d_1/bias
(24 / 201) [ ] conv2d_1/bias/read
(25 / 201) [ ] gradients/sparse_softmax_cross_entropy_loss/Sum_1_grad/Reshape/shape

Continued to conv2d_1/bias:

Stepper used feeds:
(No feeds)

Tensor "conv2d_1/bias":
dtype: float32
shape: (64,)

array([ 6.4245737e-03, -1.1049911e-03, -2.8435481e-03, -6.0856170e-03,  4.4578407e-03, -2.9437926e-03,
        1.1797058e-03, -3.3700862e-04,  4.4678403e-03,  7.9403818e-04, -1.3935991e-03,  2.3202647e-03,
        5.4761763e-03,  1.2291918e-03, -3.6602267e-03, -1.5621104e-03, -3.4348869e-03,  2.2799787e-03,
        3, -1.6903636e-03,  7.9052988e-03,  4.7243210e-03,  4.1664601e-03,  7.6617762e-03,  1.3869540e-03,
        1.2804555e-03,  5.9276596e-03,  1.9799273e-03, -3.6244935e-03, -2.1704400e-03, -9.4626172e-03,
        5,  6.4960658e-03,  3.4955924e-03,  6.6145868e-03, -1.5650952e-03, -3.9124405e-03,  5.1908474e-04,
        3.0861902e-03,  4.5332010e-03, -6.9009978e-04,  1.8244556e-03, -1.8225196e-03, -4.2091244e-03,
        5,  5.5355150e-03,  2.1685592e-03, -3.3132746e-03, -2.4143842e-03, -1.4805236e-03, -1.2613695e-04,
        7.0909485e-03,  2.3233686e-03, -1.1860991e-03, -2.6365105e-04, -1.4719635e-05,  1.6924310e-03,
        3,  5.6557666e-04, -4.1795475e-03, -1.2175994e-03,  5.5604633e-03,  8.2645088e-04, -7.4623473e-04,
        1.8561778e-03, -1.8244114e-04,  5.0964090e-03,  2.7790812e-03], dtype=float32)

```

To modify the value of a given tensor at a step stage while preserving the values of all other tensors we use;

```
tfdbg> inject "tensor_name" "value_to_be_given"
```

Executed:

```
tfdbg> inject conv2d_1/bias np.zeros(64,)
```

Obtained:

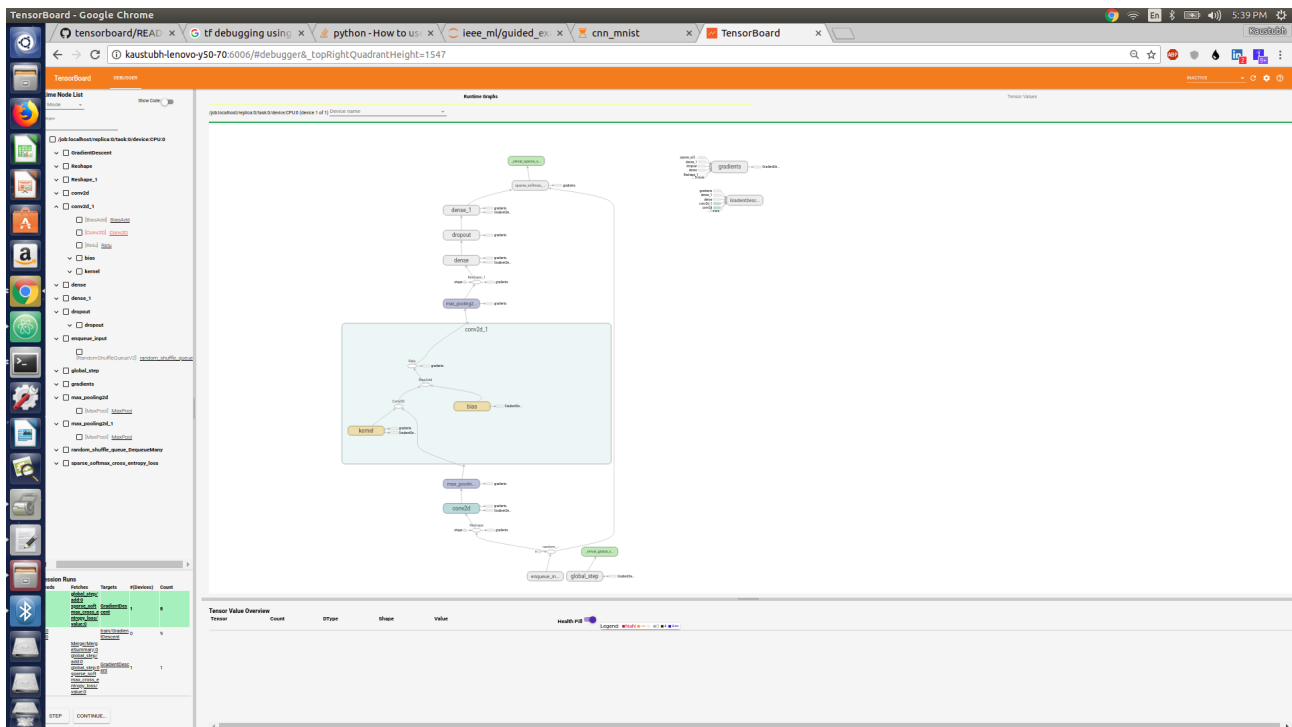
[illegible]

This command can be used to check if any specific tensor causes issues in a specific run, its value can be changed and set to a meaningful value while leaving all other tensors in the Session.run call unchanged.

## Tensorboard debugger – Not supported on Windows yet!!

Tensorboard offers a GUI plugin for debugging. In the CLI mode, we have to traverse the computational graph to go to any node of interest.

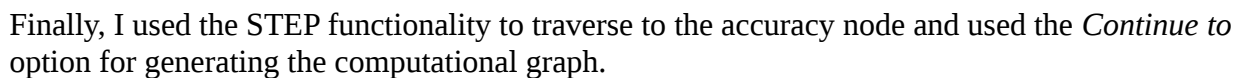
The following screenshot is the layout of the Tensorboard Debugger session for the file `cnn_mnist.py`.



On running the file; `cnn_mnist.py` either on Jupyter notebook or as a .py file, a computational graph is generated initially. Any node can be double-clicked to magnify and analyse the various elements of the magnified node. Here, the node `dense conv2d_1` is magnified and one can see the convolution operation, addition of bias in the node.

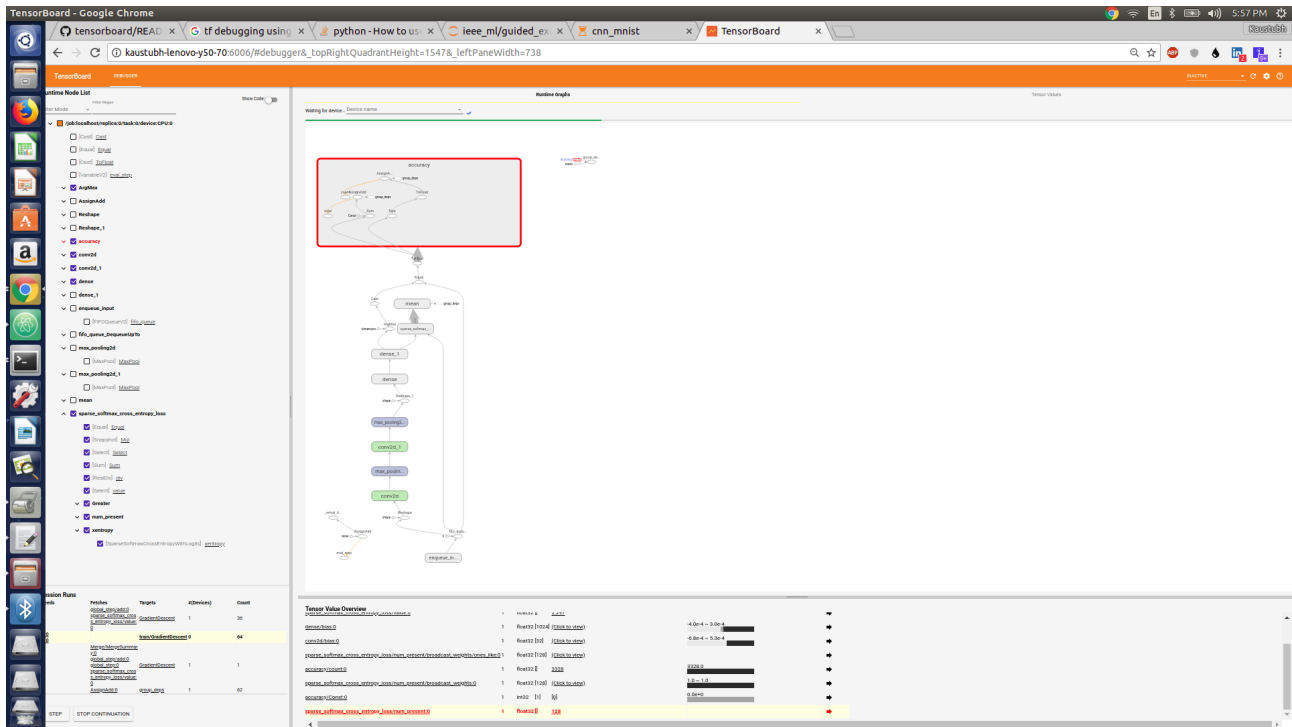
To move to next node use the STEP button in the bottom-left corner. Also, the CONTINUE button shall help you to conditionally run the Session calls.

In the following example, I reached the dense node by clicking *Continue to* after right-clicking.



Once the `Session.run()` call execution is paused, the values of tensors for all the selected nodes is displayed in the bottom half of the screen. Each tensor is also attributed with a *health pill* which visualises the proportion of values within the tensor that fall under each of the six categories noted in the legend. A user might use health pills to for instance pinpoint nodes that are culprits for producing undesired values (such as infinity & NaN).

The following screenshot illustrates this.



We have covered debugging on Tensorboard for our example in a brief manner.

However, please refer to the document;

<https://github.com/tensorflow/tensorboard/tree/master/tensorboard/plugins/debugger>

for detailed instructions for using the Tensorboard debugger.

The below *SUMMARY MATRIX* gives the Tensorflow functionalites of visualizer and debugger that are enabled with each file in the *ieee\_ml* directory.

File ./iee_ml/	Tensorboard – visualization	Tensorboard – Debugger	CLI Debugger
guided_examples/cnn_mnist.ipynb	No	Yes	No
guided_examples/cnn_mnist_no_debugger.ipynb	No	No	No
guided_examples/kmeans.ipynb	No	No	
guided_examples/nearest_neighbor.ipynb	No	No	No
guided_examples/mnist_hyperparameter.ipynb	Yes – hyperparameter search	No	No
guided_examples/xor_combined.ipynb	Yes	No	No
guided_examples/cnn_mnist.py	No	Yes	Yes
guided_examples/cnn_mnist_no_debugger.ipynb	No	No	No
guided_examples/kmeans.py	No	No	Yes
guided_examples/nearest_neighbor.py	No	No	Yes
guided_examples/mnist_hyperparameter.py	Yes – hyperparameter search	No	No
guided_examples/xor_combined.py	Yes	No	Yes
problem_sets/hdr_om/ ALL FILES	No	No	No
problem_sets/hdr_tf/handwritten_digits_recognition_cnn_5layer.ipynb	Yes	No	No
problem_sets/hdr_tf/handwritten_digits_recognition_cnn_5layer_solution.ipynb	No	No	No
problem_sets/hdr_tf/handwritten_digits_recognition_cnn_5layer.py	Yes	No	No
problem_sets/hdr_tf/handwritten_digits_recognition_cnn_5layer_solution.py	No	No	No



basics_directory/ ALL FILES	No	No	No
--------------------------------	----	----	----