Deep Learning Using TensorFlow

Dr. Ash Pahwa

Lesson 2: TensorFlow

Lesson 2.2: Writing TensorFlow Programs

Outline

TensorFlow Version Number

```
import tensorflow as tf


import numpy as np

print(tf.__version__)
1.9.0
```

NumPy and TensorFlow

```
#######################################
a = np.array([5,3,8])
b = np.array([3,-1,2])
c = np.add(a,b)

print(c)
[ 8  2 10]
```

```
a = tf.constant([5,3,8])
print(a)
Tensor("Const:0", shape=(3,), dtype=int32)

b = tf.constant([3,-1,2])
print(b)
Tensor("Const_1:0", shape=(3,), dtype=int32)

c = tf.add(a,b)
print (c)
Tensor("Add:0", shape=(3,), dtype=int32)

with tf.Session() as sess:
    result = sess.run(c)
    print(result)


[ 8  2 10]
```

Same Result Using Placeholder

```
a = tf.placeholder(dtype=tf.int32, shape=(None,))
b = tf.placeholder(dtype=tf.int32, shape=(None,))
c = tf.add(a,b)

with tf.Session() as sess:
    result = sess.run(c,
                      feed_dict={a:[3,4,5],
                                 b:[-1,2,3]})
    print(result)


[2 6 8]
```

Area of a Triangle Using Constants

- Triangle#1
  - Side length = 5.0, 3.0 7.1
- Triangle#2
  - Side length = 2.3, 4.1, 4.8

$$Area\ of\ a\ Triangle = \sqrt{s(s-a)(s-b)(s-c)}\ where\ s = \frac{a+b+c}{2}$$

```
########################################
# Area of a triangle
def compute_area(sides):
    a = sides[:,0]
    b = sides[:,1]
    c = sides[:,2]

    s = (a+b+c)*0.5
    areaSq = s*(s-a)*(s-b)*(s-c)
    return( tf.sqrt(areaSq))

with tf.Session() as sess:
    area = compute_area(tf.constant([
              [5.0, 3.0, 7.1],
              [2.3, 4.1, 4.8]
              ]))
    result = sess.run(area)
    print (result)

[ 6.27849722  4.70913887]
```

Area of a Triangle Using Placeholders

- Triangle#1
  - Side length = 5.0, 3.0 7.1
- Triangle#2
  - Side length = 2.3, 4.1, 4.8

$$Area\ of\ a\ Triangle = \sqrt{s(s-a)(s-b)(s-c)}\ where\ s = \frac{a+b+c}{2}$$

```
# Using Placeholder
def compute_area(sides):
    a = sides[:,0]
    b = sides[:,1]
    c = sides[:,2]

    s = (a+b+c)*0.5
    areaSq = s*(s-a)*(s-b)*(s-c)
    return( tf.sqrt(areaSq))

with tf.Session() as sess:
    sides = tf.placeholder(tf.float32, shape=(None,3))
    area = compute_area(sides)
    result = sess.run(area,
                      feed_dict={sides:
                          [[5.0, 3.0, 7.1],[2.3, 4.1, 4.8]]})
    print(result)

[ 6.27849722  4.70913887]
```

Eager Solution
Area of a Triangle: Using Constants

```
import tensorflow as tf
from tensorflow.contrib.eager.python import tfe
tfe.enable_eager_execution()

########################################
# Area of a triangle
def compute_area(sides):
    a = sides[:,0]
    b = sides[:,1]
    c = sides[:,2]

    s = (a+b+c)*0.5
    areaSq = s*(s-a)*(s-b)*(s-c)
    return( tf.sqrt(areaSq))

area = compute_area(tf.constant([
              [5.0, 3.0, 7.1],
              [2.3, 4.1, 4.8]
              ]))

print (area)
tf.Tensor([ 6.27849722  4.70913887], shape=(2,), dtype=float32)
```

- Triangle#1
  - Side length = 5.0, 3.0 7.1
- Triangle#2
  - Side length = 2.3, 4.1, 4.8

Eager Solution
Area of a Triangle: Using Placeholders

```
import tensorflow as tf
from tensorflow.contrib.eager.python import tfe
tfe.enable_eager_execution()

def compute_area(sides):
    a = sides[:,0]
    b = sides[:,1]
    c = sides[:,2]

    s = (a+b+c)*0.5
    areaSq = s*(s-a)*(s-b)*(s-c)
    return( tf.sqrt(areaSq))

with tf.Session() as sess:
    sides = tf.placeholder(tf.float32, shape=(None,3))
    area = compute_area(sides)
    result = sess.run(area,
                      feed_dict={sides:
                          [[5.0, 3.0, 7.1],[2.3, 4.1, 4.8]]})
    print(result)


[ 6.27849722  4.70913887]
```

- Triangle#1
  - Side length = 5.0, 3.0 7.1
- Triangle#2
  - Side length = 2.3, 4.1, 4.8

TensorFlow Math Functions

TensorFlow Math Functions

#	TensorFlow Operator	Description
1	tf.add	Returns the sum
2	tf.sub	Returns subtraction
3	tf.mul	Returns the multiplication
4	tf.div	Returns the division
5	tf.mod	Returns the module
6	tf.abs	Returns the absolute value
7	tf.neg	Returns the negative value
8	tf.sign	Returns the sign
9	tf.inv	Returns the inverse
10	tf.square	Returns the square

#	TensorFlow Operator	Description
11	tf.round	Returns the nearest integer
12	tf.sqrt	Returns the square root
13	tf.pow	Returns the power
14	tf.exp	Returns the exponential
15	tf.log	Returns the logarithm
16	tf.maximum	Returns the maximum
17	tf.minimum	Returns the minimum
18	tf.cos	Returns the cosine
19	tf.sin	Returns the sine

How to Debug TensorFlow Programs

How to Debug a TensorFlow Program?

- Lazy Execution
  - You cannot find bugs in your code till you run the code
  - During the graph building stage bugs cannot be found
- Debugging a TensorFlow program
  - During the program development stage
    - Run the TF program in Eager mode
    - It will help you to debug your program

Debugging in Lazy Mode

- Read Error Message
- Isolate the method in question
- Send “made up” data into the method
- Know how to solve common problems

Types of Errors

- Shape Errors
- Scalar Vector Mismatch
- Data Type Mismatch

Shape Errors

Debugging

```
def some_method (data):
    a = data[:,0:2]
    print(a.get_shape())
    c = data[:,1]
    #c = data[:,1:3]
    print(c.get_shape())
    s = ( a + c )
    return tf.sqrt(tf.matmul(s,tf.transpose(s)))
```

```
with tf.Session() as sess:
    fake_data = tf.constant ([
            [5.0, 3.0, 7.1],
            [2.3, 4.1, 4.8],
            [2.8, 4.2, 5.6],
            [2.9, 8.3, 7.3]
            ])
    print
(sess.run(some_method(fake_data)))


(4, 2)
(4,)
```

Error Message

```
Traceback (most recent
call last):

  File "<ipython-input-4-
558f2d6141f1>", line 8,
in <module>
    print
(sess.run(some_method(fak
e_data)))

  File "<ipython-input-3-
2c14a1ed09fc>", line 7,
in some_method
    s = ( a + c )

  File
"C:\ProgramData\Anaconda3
\lib\site-
packages\tensorflow\pytho
n\ops\math_ops.py", line
847, in binary_op_wrapper
    return func(x, y,
name=name)

  File
"C:\ProgramData\Anaconda3
\lib\site-
packages\tensorflow\pytho
n\ops\gen_math_ops.py",
line 296, in add
    "Add", x=x, y=y,
name=name)
```

```
File
"C:\ProgramData\Anaconda3\lib\site
-
packages\tensorflow\python\framewo
rk\op_def_library.py", line 787,
in _apply_op_helper
    op_def=op_def)

  File
"C:\ProgramData\Anaconda3\lib\site
-
packages\tensorflow\python\framewo
rk\ops.py", line 3414, in
create_op
    op_def=op_def)

  File
"C:\ProgramData\Anaconda3\lib\site
-
packages\tensorflow\python\framewo
rk\ops.py", line 1756, in __init__
    control_input_ops)

  File
"C:\ProgramData\Anaconda3\lib\site
-
packages\tensorflow\python\framewo
rk\ops.py", line 1592, in
_create_c_op
    raise ValueError(str(e))

ValueError: Dimensions must be
equal, but are 2 and 4 for 'add'
(op: 'Add') with input shapes:
[4,2], [4].
```

Debugging: Fixing the Problem

```
import tensorflow as tf

def some_method (data):
    a = data[:,0:2]
    print(a.get_shape())
    #c = data[:,1]
    c = data[:,1:3]
    print(c.get_shape())
    s = ( a + c )
    return tf.sqrt(tf.matmul(s,tf.transpose(s)))
```

```
with tf.Session() as sess:
    fake_data = tf.constant ([
            [5.0, 3.0, 7.1],
            [2.3, 4.1, 4.8],
            [2.8, 4.2, 5.6],
            [2.9,8.3, 7.3]
            ])
    print (sess.run(some_method(fake_data)))
```

```
(4, 2)
(4, 2)
[[ 12.88448715  11.87813091  12.44909573  15.72132301]
 [ 11.87813091  10.96220779  11.48999596  14.50930595]
 [ 12.44909573  11.48999596  12.04325485  15.20789242]
 [ 15.72132301  14.50930595  15.20789242  19.20416641]]
```

Shape Problems

- Can also occur batch size mismatch
- First batch has 64 observations
- Second batch has 64 observations
- But the last batch may have only 42 observations

Shape Problems can be Fixed Using

- "tf.reshape()"
- "tf.expand_dims()"
- "tf.slice()"
- "tf.squeeze()"

Example: “tf.expand_dims”

- Example: “tf.expand_dims ” converts a tensor
  - From 3/2
  - To 3/1/2

```
import tensorflow as tf
x = tf.constant([ [3,2],
                  [4,5],
                  [6,7] ])


print("x.shape", x.shape)
x.shape (3, 2)

expanded = tf.expand_dims(x,1)

print("expanded shape",expanded.shape)
expanded shape (3, 1, 2)

with tf.Session() as sess:
    print("expanded:\n", expanded.eval())


expanded:
 [[[3 2]]
 [[4 5]]
 [[6 7]]]
```

Example: “tf.slice”

```
x = tf.constant([ [3,2],
                  [4,5],
                  [6,7] ])


print("x.shape", x.shape)
x.shape (3, 2)

sliced = tf.slice(x,[0,1],[2,1])

print("sliced shape",sliced.shape)
sliced shape (2, 1)

with tf.Session() as sess:
    print("Sliced Shape\n", sliced.eval())


Sliced Shape
 [[2]
 [5]]
```

Example: “tf.squeeze”

- Squeeze is an inverse of “expand_dims”
- Example: “tf.squeeze” converts a tensor
  - From 1/2/4
  - To 2/4

```
import tensorflow as tf

t = tf.constant([  [[1],[2],[3],[4]],  [[5],[6],[7],[8]]   ])

with tf.Session() as sess:
    print("t")
    print(sess.run(t))
    print("t Squeezed")
    print(sess.run(tf.squeeze(t)))
```

```
t
[[[1]
  [2]
  [3]
  [4]]

 [[5]
  [6]
  [7]
  [8]]]
t Squeezed
[[1 2 3 4]
 [5 6 7 8]]
```

Data Type Problems

Datatype Problems

```
def some_method(a,b):
    s = (a+b)
    return tf.sqrt(tf.matmul(s,tf.transpose(s)))



with tf.Session() as sess:
    fake_a = tf.constant ([
            [5.0, 3.0, 7.1],
            [2.3, 4.1, 4.8]
            ])
    fake_b = tf.constant ([
            [2, 4, 5],
            [2, 8, 7]
            ])
    print( sess.run(some_method(fake_a,
fake_b)))
```

```
Traceback (most recent call last):

  File "<ipython-input-9-7039eda1e8b3>", line
10, in <module>
    print( sess.run(some_method(fake_a,
fake_b)))

  File "<ipython-input-8-c5828ee59b1f>", line
2, in some_method
    s = (a+b)

  File "C:\ProgramData\Anaconda3\lib\site-
packages\tensorflow\python\ops\math_ops.py",
line 847, in binary_op_wrapper
    return func(x, y, name=name)

  File "C:\ProgramData\Anaconda3\lib\site-
packages\tensorflow\python\ops\gen_math_ops.py
", line 296, in add
    "Add", x=x, y=y, name=name)

  File "C:\ProgramData\Anaconda3\lib\site-
packages\tensorflow\python\framework\op_def_li
brary.py", line 546, in _apply_op_helper
    inferred_from[input_arg.type_attr]))

TypeError: Input 'y' of 'Add' Op has type
int32 that does not match type float32 of
argument 'x'.
```

Both arrays ‘a’ and ‘b’ should be same datatype

Solution: Cast Array ‘b’ to ‘float32’

```
def some_method(a,b):
    b = tf.cast(b,tf.float32)
    s = (a+b)
    return tf.sqrt(tf.matmul(s,tf.transpose(s)))



with tf.Session() as sess:
    fake_a = tf.constant ([
            [5.0, 3.0, 7.1],
            [2.3, 4.1, 4.8]
            ])
    fake_b = tf.constant ([
            [2, 4, 5],
            [2, 8, 7]
            ])
    print( sess.run(some_method(fake_a, fake_b)))


[[ 15.63361835  16.04929924]
 [ 16.04929924  17.43960953]]
```

General Guidance for Debugging TensorFlow Programs

How to Debug the TF Program

- ‘tf.Print()’
  - Print out values of tensor when specific conditions are met
- ‘tfdbg’
  - Interactive Debugger
  - Runs from a terminal and attach to a local or remote TensorFlow session

How to Debug the TF Program

- TensorBoard is a visual monitoring tool
  - Verbosity Levels
    - Fatal – quite
    - Error
    - Warn Default level
    - Info
    - Debug – most verbose

Terminal Debugger

- Terminal Window
  - Python xyz.py –debug
- Options during debugging
  - Step through the code
  - Set break points

Summary

- TensorFlow Version Number
- NumPy and TensorFlow
- Same Result Using Placeholder
- Area of a Triangle Using Constants + Place Holder
- Eager Solution: Area of a Triangle Using Constants + Place Holder
- Math Functions
- Debugging a TensorFlow Program
- Shape errors
- Data Type Problems
- General Guidance in Debugging TensorFlow Programs