

Data Science: Machine Learning: Tensorflow API

CPSC 501: Advanced Programming Techniques
Fall 2022

Jonathan Hudson, Ph.D
Assistant Professor (Teaching)
Department of Computer Science
University of Calgary

Thursday, October 13, 2022



UNIVERSITY OF
CALGARY

Tensors

Constants

```
import tensorflow as tf  
a = tf.constant([2, 2], name='a')  
b = tf.constant([[0, 1], [2, 3]], name='b')
```

```
tf.constant(  
    value,  
    dtype=None,  
    shape=None,  
    name='Const',  
    verify_shape=False  
)
```

Tensors filled with a specific value

- `tf.zeros(shape, dtype=tf.float32, name=None)`
- creates a tensor of shape and all elements will be zeros

- `tf.zeros([2, 3], tf.int32) ==> [[0, 0, 0], [0, 0, 0]]`

Similar to `numpy.zeros`

Tensors filled with a specific value

- `tf.zeros_like(input_tensor, dtype=None, name=None, optimize=True)`
- creates a tensor of shape and type (unless type is specified) as the `input_tensor` but all elements are zeros.

- # `input_tensor` is `[[0, 1], [2, 3], [4, 5]]`
- `tf.zeros_like(input_tensor) ==> [[0, 0], [0, 0], [0, 0]]`

Similar to `numpy.zeros_like`

Tensors filled with a specific value

- `tf.ones(shape, dtype=tf.float32, name=None)`
- `tf.ones_like(input_tensor, dtype=None, name=None, optimize=True)`

Similar to `numpy.ones`,
`numpy.ones_like`

Tensors filled with a specific value

- `tf.fill(dims, value, name=None)`
- creates a tensor filled with a scalar value.

- `tf.fill([2, 3], 8) ==> [[8, 8, 8], [8, 8, 8]]`

Similar to NumPy.full

Constants as sequences

- `tf.lin_space(start, stop, num, name=None)`
`tf.lin_space(10.0, 13.0, 4) ==> [10. 11. 12. 13.]`
- `tf.range(start, limit=None, delta=1, dtype=None, name='range')`
`tf.range(3, 18, 3) ==> [3, 6, 9, 12, 15]`
`tf.range(5) ==> [0, 1, 2, 3, 4]`

NOT THE SAME AS NUMPY SEQUENCES

Tensor objects are not iterable

```
for _ in tf.range(4): # TypeError
```


Randomly Generated Constants

- `tf.random_normal`
- `tf.truncated_normal`
- `tf.random_uniform`
- `tf.random_shuffle`
- `tf.random_crop`
- `tf.multinomial`
- `tf.random_gamma`

Randomly Generated Constants

- `tf.set_random_seed(seed)`

Operations

Operations

Category	Examples
Element-wise mathematical operations	Add, Sub, Mul, Div, Exp, Log, Greater, Less, Equal, ...
Array operations	Concat, Slice, Split, Constant, Rank, Shape, Shuffle, ...
Matrix operations	MatMul, MatrixInverse, MatrixDeterminant, ...
Stateful operations	Variable, Assign, AssignAdd, ...
Neural network building blocks	SoftMax, Sigmoid, ReLU, Convolution2D, MaxPool, ...
Checkpointing operations	Save, Restore
Queue and synchronization operations	Enqueue, Dequeue, MutexAcquire, MutexRelease, ...
Control flow operations	Merge, Switch, Enter, Leave, NextIteration

Arithmetic Ops

- `tf.abs`
- `tf.negative`
- `tf.sign`
- `tf.reciprocal`
- `tf.square`
- `tf.round`
- `tf.sqrt`
- `tf.rsqrt`
- `tf.pow`
- `tf.exp`

Wizard of Div

```
a = tf.constant([2, 2], name='a')
```

```
b = tf.constant([[0, 1], [2, 3]], name='b')
```

```
print(tf.divide(b, a))           ⇒ [[0. 0.5] [1. 1.5]]
```

```
print(tf.truediv(b, a))        ⇒ [[0. 0.5] [1. 1.5]]
```

```
print(tf.truncatediv(b, a))    ⇒ [[0 0] [1 1]]
```

Data Types

TensorFlow Data Types

- TensorFlow takes Python natives types: boolean, numeric (int, float), strings

```
t_0 = 19  
tensors
```

scalars are treated like 0-d

```
tf.zeros_like(t_0)
```

==> 0

```
tf.ones_like(t_0)
```

==> 1

TensorFlow Data Types

- TensorFlow takes Python natives types: boolean, numeric (int, float), strings

```
t_1 = [b"apple", b"peach"]    # 1-d arrays are like 1-d tensors
```

```
tf.zeros_like(t_1)          # ==> [b"", b""]
```

```
tf.ones_like(t_1)           # ==> TypeError: Expected string, got 1 of type
```

TensorFlow Data Types

- TensorFlow takes Python natives types: boolean, numeric (int, float), strings

```
t_2 = [[True, False, False],  
       [False, False, True],  
       [False, True, False]]
```

2-d arrays are treated like 2-d tensors

```
tf.zeros_like(t_2)
```

==> 3x3 tensor, all elements are False

```
tf.ones_like(t_2)
```

==> 3x3 tensor, all elements are True

TensorFlow Data Types

`tf.float16` : 16-bit half-precision floating-point.

`tf.float32` : 32-bit single-precision floating-point.

`tf.float64` : 64-bit double-precision floating-point.

`tf.bfloat16` : 16-bit truncated floating-point.

`tf.complex64` : 64-bit single-precision complex.

`tf.complex128` : 128-bit double-precision complex.

`tf.int8` : 8-bit signed integer.

`tf.uint8` : 8-bit unsigned integer.

`tf.uint16` : 16-bit unsigned integer.

`tf.int16` : 16-bit signed integer.

`tf.int32` : 32-bit signed integer.

`tf.int64` : 64-bit signed integer.

`tf.bool` : Boolean.

`tf.string` : String.

`tf.qint8` : Quantized 8-bit signed integer.

`tf.quint8` : Quantized 8-bit unsigned integer.

`tf.qint16` : Quantized 16-bit signed integer.

`tf.quint16` : Quantized 16-bit unsigned integer.

`tf.qint32` : Quantized 32-bit signed integer.

`tf.resource` : Handle to a mutable resource.

TF vs NP Data Types

- TensorFlow integrates seamlessly with NumPy
- `tf.int32 == np.int32` `# ⇒ True`

- Can pass numpy types to TensorFlow ops
- `tf.ones([2, 2], np.float32)` `# ⇒ [[1.0 1.0], [1.0 1.0]]`

Use TF DType when possible

- Python native types: TensorFlow has to infer Python type
- NumPy arrays: NumPy is not GPU compatible

Constants?

Constants?

What's wrong with constants ...

... other than being constant?

What's wrong with constants?

- Constants are stored in the graph definition

Print out the graph def (shown in TF1 format)

- `my_const = tf.constant([1.0, 2.0], name="my_const")`
- with `tf.Session()` as `sess`:
`print(sess.graph.as_graph_def())`

```
attr {  
  key: "value"  
  value {  
    tensor {  
      dtype: DT_FLOAT  
      tensor_shape {  
        dim {  
          size: 2  
        }  
      }  
      tensor_content: "\000\000\200?\000\000\000@"  
    }  
  }  
}
```

What's wrong with constants?

- This makes loading graphs expensive when constants are big
- Only use constants for primitive types.
- Use variables or readers for more data that requires more memory

Variables

Variables

```
# create variables with tf.Variable  
s = tf.Variable(2, name="scalar")  
m = tf.Variable([[0, 1], [2, 3]], name="matrix")  
W = tf.Variable(tf.zeros([784,10]))
```

Variables

- Why `tf.constant` but `tf.Variable`?
- `tf.Variable` is a class with many ops

`tf.Variable` holds several ops:

`x = tf.Variable(...)`

`x.initializer` # init op

`x.value()` # read op

`x.assign(...)` # write op

`x.assign_add(...)` # and more

Shapes

Shapes

A tuple of dimensions

- Often reported when asking about what data is 'like' in a graph (tensor)

(28,28) is a matrix of 28 entries by 28 entries (grayscale image)

(5) Is a one-dimensional array of length 5

() Is a scalar

(28,28,3) is a matrix of 28 by 28 entries with each entry being having three items
(3d matrix) -> RGB image

Onward to ... tensorflow linear regression.

Jonathan Hudson
jwhudson@ucalgary.ca
<https://pages.cpsc.ucalgary.ca/~jwhudson/>



UNIVERSITY OF
CALGARY