# Tensors

TensorFlow, as the name indicates, is a framework to define and run computations involving tensors.

A **tensor** is a generalization of vectors and matrices to potentially higher dimensions.

Internally, TensorFlow represents tensors as n-dimensional arrays of base datatypes.

A [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) has the following properties:

* a data type (float32, int32, or string, for example)
* a shape

Each element in the Tensor has the same data type, and the data type is always known.

The shape (that is, the number of dimensions it has and the size of each dimension) might be only partially known.

Some types of tensors are:

* [tf.Variable](https://www.tensorflow.org/api_docs/python/tf/Variable)
* [tf.constant](https://www.tensorflow.org/api_docs/python/tf/constant)
* [tf.placeholder](https://www.tensorflow.org/api_docs/python/tf/placeholder)
* [tf.SparseTensor](https://www.tensorflow.org/api_docs/python/tf/sparse/SparseTensor)

With the exception of [tf.Variable](https://www.tensorflow.org/api_docs/python/tf/Variable), the value of a tensor is immutable, which means that in the context of a single execution tensors only have a single value.

However, evaluating the same tensor twice can return different values; for example that tensor can be the result of reading data from disk, or generating a random number.

## Rank

The **rank** of a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) object is its number of dimensions.

Synonyms for rank include **order** or **degree** or **n-dimension**.

| Rank | Math entity |
| --- | --- |
| 0 | Scalar (magnitude only) |
| 1 | Vector (magnitude and direction) |
| 2 | Matrix (table of numbers) |
| 3 | 3-Tensor (cube of numbers) |
| n | n-Tensor (you get the idea) |

### Rank 0

The following snippet demonstrates creating a few rank 0 variables:

mammal = tf.Variable("Elephant", tf.string)  
ignition = tf.Variable(451, tf.int16)  
floating = tf.Variable(3.14159265359, tf.float64)  
its\_complicated = tf.Variable(12.3 - 4.85j, tf.complex64)

**Note:** A string is treated as a single object in TensorFlow, not as a sequence of characters. It is possible to have scalar strings, vectors of strings, etc.

### Rank 1

To create a rank 1 [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) object, you can pass a list of items as the initial value:

mystr = tf.Variable(["Hello"], tf.string)  
cool\_numbers  = tf.Variable([3.14159, 2.71828], tf.float32)  
first\_primes = tf.Variable([2, 3, 5, 7, 11], tf.int32)  
its\_very\_complicated = tf.Variable([12.3 - 4.85j, 7.5 - 6.23j], tf.complex64)

### Higher ranks

A rank 2 [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) object consists of at least one row and at least one column:

mymat = tf.Variable([[7],[11]], tf.int16)  
myxor = tf.Variable([[False, True],[True, False]], tf.bool)  
linear\_squares = tf.Variable([[4], [9], [16], [25]], tf.int32)  
squarish\_squares = tf.Variable([ [4, 9], [16, 25] ], tf.int32)  
rank\_of\_squares = tf.rank(squarish\_squares)  
mymatC = tf.Variable([[7],[11]], tf.int32)

Higher-rank Tensors, similarly, consist of an n-dimensional array.

For example, during image processing, many tensors of rank 4 are used, with dimensions corresponding to example-in-batch, image width, image height, and color channel.

my\_image = tf.zeros([10, 299, 299, 3])  # batch x height x width x color

### Getting a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) object's rank

To determine the rank of a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) object, call the [tf.rank](https://www.tensorflow.org/api_docs/python/tf/rank) method.

r = tf.rank(my\_image)  
# After the graph runs, r will hold the value 4.

### Referring to [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) slices

Since a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) is an n-dimensional array of cells, to access a single cell in a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) you need to specify n indices.

For a rank 0 tensor (a scalar), no indices are necessary, since it is already a single number.

For a rank 1 tensor (a vector), passing a single index allows you to access a number:

my\_scalar = my\_vector[2]

Note that the index passed inside the [] can itself be a scalar [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor), if you want to dynamically choose an element from the vector.

my\_scalar = my\_matrix[1, 2]

Passing a single number, however, returns a subvector of a matrix, as follows:

my\_row\_vector = my\_matrix[2]  
my\_column\_vector = my\_matrix[:, 3]

The : notation is python slicing syntax for "leave this dimension alone".

## Shape

The **shape** of a tensor is the number of elements in each dimension.

The TensorFlow documentation uses three notational conventions to describe tensor dimensionality: rank, shape, and dimension number.

| Rank | Shape | Dimension number | Example |
| --- | --- | --- | --- |
| 0 | [] | 0-D | A 0-D tensor. A scalar. |
| 1 | [D0] | 1-D | A 1-D tensor with shape [5]. |
| 2 | [D0, D1] | 2-D | A 2-D tensor with shape [3, 4]. |
| 3 | [D0, D1, D2] | 3-D | A 3-D tensor with shape [1, 4, 3]. |
| n | [D0, D1, ... Dn-1] | n-D | A tensor with shape [D0, D1, ... Dn-1]. |

Shapes can be represented via Python lists / tuples of ints, or with the [tf.TensorShape](https://www.tensorflow.org/api_docs/python/tf/TensorShape).

### Getting a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) object's shape

There are two ways of accessing the shape of a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor).

This can be done by reading the shape property of a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) object.

This method returns a TensorShape object, which is a convenient way of representing partially-specified shapes (since, when building the graph, not all shapes will be fully known).

It is also possible to get a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) that will represent the fully-defined shape of another [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) at runtime. This is done by calling the [tf.shape](https://www.tensorflow.org/api_docs/python/tf/shape) operation.

This way, you can build a graph that manipulates the shapes of tensors by building other tensors that depend on the dynamic shape of the input [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor).

For example, here is how to make a vector of zeros with the same size as the number of columns in a given matrix:

zeros = tf.zeros(my\_matrix.shape[1])

### Changing the shape of a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor)

The **number of elements** of a tensor is the product of the sizes of all its shapes.

The number of elements of a scalar is always 1. Since there are often many different shapes that have the same number of elements, it's often convenient to be able to change the shape of a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor), keeping its elements fixed. This can be done with [tf.reshape](https://www.tensorflow.org/api_docs/python/tf/reshape).

The following examples demonstrate how to reshape tensors:

rank\_three\_tensor = tf.ones([3, 4, 5])  
matrix = tf.reshape(rank\_three\_tensor, [6, 10])  # Reshape existing content into  
                                                 # a 6x10 matrix  
matrixB = tf.reshape(matrix, [3, -1])  #  Reshape existing content into a 3x20  
                                       # matrix. -1 tells reshape to calculate  
                                       # the size of this dimension.  
matrixAlt = tf.reshape(matrixB, [4, 3, -1])  # Reshape existing content into a  
                                             #4x3x5 tensor  
  
# Note that the number of elements of the reshaped Tensors has to match the  
# original number of elements. Therefore, the following example generates an  
# error because no possible value for the last dimension will match the number  
# of elements.  
yet\_another = tf.reshape(matrixAlt, [13, 2, -1])  # ERROR!

## Data types

In addition to dimensionality, Tensors have a data type.

It is possible to cast [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor)s from one datatype to another using [tf.cast](https://www.tensorflow.org/api_docs/python/tf/dtypes/cast):

# Cast a constant integer tensor into floating point.  
float\_tensor = tf.cast(tf.constant([1, 2, 3]), dtype=tf.float32)

To inspect a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor)'s data type use the Tensor.dtype property.

When creating a [tf.Tensor](https://www.tensorflow.org/api_docs/python/tf/Tensor) from a python object you may optionally specify the datatype.

## Evaluating Tensors

Tensors that depend on placeholders can't be evaluated without providing a value for the placeholder.

import tensorflow as tf

import os

session = tf.Session()

p = tf.placeholder(tf.float32)

t = p + 1.0

#t.eval() # This will fail, since the placeholder did not get a value.

result = session.run(t,feed\_dict={p:2.0})

# This will succeed because we're feeding a value

# to the placeholder.

print( result )

#3.0

@