

TensorFlow Quick Reference Sheet

by
www.TensorFlow.uk

Import TensorFlow:

```
import tensorflow as tf
```

Basic math operations:	
<code>tf.add()</code>	sum
<code>tf.subtract()</code>	subtraction
<code>tf.multiply()</code>	multiplication
<code>tf.div()</code>	division
<code>tf.mod()</code>	module
<code>tf.abs()</code>	absolute value
<code>tf.negative()</code>	negative value
<code>tf.sign()</code>	return sign
<code>tf.reciprocal()</code>	reciprocal
<code>tf.square()</code>	square
<code>tf.round()</code>	nearest integer
<code>tf.sqrt()</code>	square root
<code>tf.pow()</code>	power
<code>tf.exp()</code>	exponent
<code>tf.log()</code>	logarithm
<code>tf.maximum()</code>	maximum
<code>tf.minimum()</code>	minimum
<code>tf.cos()</code>	cosine
<code>tf.sin()</code>	sine

Basic operations on tensors:	
<code>tf.string_to_number()</code>	converts string to numeric type
<code>tf.cast()</code>	casts to new type
<code>tf.shape()</code>	returns shape of tensor
<code>tf.reshape()</code>	reshapes tensor
<code>tf.diag()</code>	creates tensor with given diagonal values
<code>tf.zeros()</code>	creates tensor with all elements set to zero
<code>tf.fill()</code>	creates tensor with all elements set given value
<code>tf.concat()</code>	concatenates tensors
<code>tf.slice()</code>	extracts slice from tensor
<code>tf.transpose()</code>	transpose the argument
<code>tf.matmul()</code>	matrices multiplication
<code>tf.matrix_determinant()</code>	determinant of matrices
<code>tf.matrix_inverse()</code>	computes inverse of matrices

Control Flow:	
<code>tf.while_loop()</code>	repeat body while condition true
<code>tf.case()</code>	case operator
<code>tf.count_up_to()</code>	increments ref until limit
<code>tf.tuple()</code>	groups tensors together

Logical/Comparison Operators:	
<code>tf.equal()</code>	returns truth value element-wise
<code>tf.not_equal()</code>	returns truth value of X!=Y
<code>tf.less()</code>	returns truth value of X<Y
<code>tf.less_equal()</code>	returns truth value of X<=Y
<code>tf.greater()</code>	returns truth value of X>Y
<code>tf.greater_equal()</code>	returns truth value of X>=Y
<code>tf.is_nan()</code>	returns which elements are NaN
<code>tf.logical_and()</code>	returns truth value of 'AND' for given tensors
<code>tf.logical_or()</code>	returns truth value of 'OR' for given tensors
<code>tf.logical_not()</code>	returns truth value of 'NOT' for given tensors
<code>tf.logical_xor()</code>	returns truth value of 'XOR' for given tensors

Working with Images:	
<code>tf.image.decode_image()</code>	converts image to tensor type uint8
<code>tf.image.resize_images()</code>	resize images
<code>tf.image.resize_image_with_crop</code>	resize image by cropping or padding
<code>tf.image.flip_up_down()</code>	flip image horizontally
<code>tf.image.rot90()</code>	rotate image 90 degrees counter-clockwise
<code>tf.image.rgb_to_grayscale()</code>	converts image from RGB to grayscale
<code>tf.image.per_image_standardization</code>	scales image to zero mean and unit norm

Neural Networks:	
<code>tf.nn.relu()</code>	rectified linear activation function
<code>tf.nn.softmax()</code>	softmax activation function
<code>tf.nn.sigmoid()</code>	sigmoid activation function
<code>tf.nn.tanh()</code>	hyperbolic tangent activation function
<code>tf.nn.dropout</code>	dropout
<code>tf.nn.bias_add</code>	adds bias to value
<code>tf.nn.all_candidate_sampler()</code>	set of all classes
<code>tf.nn.weighted_moments()</code>	returns mean and variance
<code>tf.nn.softmax_cross_entropy_with_logits()</code>	softmax cross entropy
<code>tf.nn.sigmoid_cross_entropy_with_logits()</code>	sigmoid cross entropy
<code>tf.nn.l2_normalize()</code>	normalization using L2 Norm
<code>tf.nn.l2_loss()</code>	L2 loss
<code>tf.nn.dynamic_rnn()</code>	RNN specified by given cell
<code>tf.nn.conv2d()</code>	2D convolutions given 4D input
<code>tf.nn.conv1d()</code>	1D convolution given 3D input
<code>tf.nn.batch_normalization()</code>	batch normalization
<code>tf.nn.xw_plus_b()</code>	computes matmul(x,weights)+biases

High level Machine Learning:	
<code>tf.contrib.keras</code>	Keras API as high level API for TensorFlow
<code>tf.contrib.layers.one_hot_column()</code>	one hot encoding
<code>tf.contrib.learn.LogisticRegressor()</code>	logistic regression
<code>tf.contrib.learn.DNNClassifier()</code>	DNN classifier
<code>tf.contrib.learn.DynamicRnnEstimator()</code>	Rnn Estimator
<code>tf.contrib.learn.KMeansClustering()</code>	K-Means Clustering
<code>tf.contrib.learn.LinearClassifier()</code>	linear classifier
<code>tf.contrib.learn.LinearRegressor()</code>	linear regressor
<code>tf.contrib.learn.extract_pandas_data()</code>	extract data from Pandas dataframe
<code>tf.contrib.metrics.accuracy()</code>	accuracy
<code>tf.contrib.metrics.auc_using_histogram()</code>	AUC
<code>tf.contrib.metrics.confusion_matrix()</code>	confusion matrix
<code>tf.contrib.metrics.streaming_mean_absolute_error()</code>	mean absolute error
<code>tf.contrib.rnn.BasicLSTMCell()</code>	basic lstm cell
<code>tf.contrib.rnn.BasicRNNCell()</code>	basic rnn cell

Placeholders and Variables:	
<code>tf.placeholder()</code>	defines placeholder
<code>tf.Variable(tf.random_normal([3, 4], stddev=0.1))</code>	defines variable
<code>tf.Variable(tf.zeros([50]), name='x')</code>	defines variable
<code>tf.global_variables_initializer()</code>	initialize global variables
<code>tf.local_variables_initializer()</code>	initialize local variables

```
with tf.device("/cpu:0"):  
    v = tf.Variable()
```

```
with tf.device("/gpu:0"):  
    v = tf.Variable()
```

```
sess = tf.Session()  
sess.run()  
sess.close()
```

```
with tf.Session() as session:  
    session.run()
```

```
saver=tf.train.Saver()  
saver.save(sess,'file_name')  
saver.restore(sess,'file_name')
```

Working with Data:	
<code>tf.decode_csv()</code>	converts csv to tensors
<code>tf.read_file()</code>	reads file
<code>tf.write_file()</code>	writes to file
<code>tf.train.batch()</code>	creates batches of tensors

If you need more detailed information please visit WWW.TENSORFLOW.ORG all above information have been sourced there.