TensorFlow 1 - Convolutional Layers

March 27, 2018

0.1 Convolutional Layers in TensorFlow

TensorFlow provides the `tf.nn.conv2d()`, `tf.nn.bias_add()`, and `tf.nn.relu()` functions to create our own convolutional layers.

```
In [ ]: import tensorflow as tf

        # input/image
        input = tf.placeholder(
            tf.float32,
            shape=[None, image_height, image_width, color_channels])
```

We define a placeholder for input of type `tf.float32`. The shape of the input is defined as $NxHxWxC$ in which N is batch size, HxW is the height and width of images (same size across the batch) and C is the number of image channels i.e. 3 for an RGB image.

Note that in **Caffe**, the shape of the input is defined as $NxCxHxW$.

```
In [ ]: # convolution filter dimensions
        filter_size_width = 5
        filter_size_height = 5

        # weight and bias
        weight = tf.Variable(
            tf.truncated_normal([filter_size_height, filter_size_width, color_channels, k_output
        bias = tf.Variable(tf.zeros(k_output))
```

We define weight and bias as variables as they will be changed.

`weight` is constructed using `tf.truncated_normal()`.

From the TensorFlow document, `tf.truncated_normal()` outputs a tensor of the specified shape filled with random values from a truncated normal distribution. That is the generated values follow *a normal distribution with specified mean and standard deviation*, except that values whose magnitude is more than 2 standard deviations from the mean are dropped and re-picked.

The constructor of the `tf.truncated_normal()` is defined as

```
tf.truncated_normal(
  shape,
  mean=0.0,
  stddev=1.0,
  dtype=tf.float32,
  seed=None,
  name=None
)
```

```
In [ ]: # apply convolution
        # padding can be set either 'SAME' or 'VALID'
        conv_layer = tf.nn.conv2d(input, weight, strides=[1,2,2,1], padding='SAME')
        # add bias
        conv_layer = tf.nn.bias_add(conv_layer, bias)
        # apply activation function
        conv_layer = tf.nn.relu(conv_layer)
```

The code above is similar to the math notation `conv = relu(input * weight + bias)`. We use the `tf.nn.conv2d()` function to compute the convolution with `weight` in which the strides are `[1,2,2,1]` i.e. the stride for the image is 2 in the `x` and `y` direction. - TensorFlow uses a stride for each `input` dimension, stride: `[batch, input_height, input_width, input_channels]`. - The stride for `batch` and `input_channel` are always set to 1. This ensures that the model uses all batches and input channels. - The input_height and input_width strides are for striding the filter over `input`. In the code above, we uses a stride of 2 with 5x5 over `input`.