- Activation. Your choice of activation function.
- Layer 5: Fully Connected (Logits). This should have 10 outputs.

Output

- Return the result of the 2nd fully connected layer.

In [17]:

```
### architecture
import tensorflow as tf
from tensorflow.contrib.layers import flatten

# for tf.truncated_normal
mu = 0.0
sigma = 0.1

# for weights -- height, width, depth_in, depth_out
# for padding -- 'VALID'
# out_height = ceil(float(in_height - filter_height + 1) / float(strides[1])
weights = {
    'c1': tf.Variable(tf.truncated_normal(shape = (5,5,1,12), mean = mu, stddev = sigma)),
    'c2': tf.Variable(tf.truncated_normal(shape = (5,5,12,32), mean = mu, stddev = sigma)),
    'f1': tf.Variable(tf.truncated_normal(shape = (5*5*32,120), mean = mu, stddev = sigma)),
    'f2': tf.Variable(tf.truncated_normal(shape = (120, 84), mean = mu, stddev = sigma)),
    'f3': tf.Variable(tf.truncated_normal(shape = (84,43), mean = mu, stddev = sigma))
}

biases = {
    'bc1':tf.Variable(tf.zeros(12)),
    'bc2':tf.Variable(tf.zeros(32)),
    'bf1':tf.Variable(tf.zeros(120)),
    'bf2':tf.Variable(tf.zeros(84)),
    'bf3':tf.Variable(tf.zeros(43))
}


def covNet(x, weights, biases, dropout = 1.0):
    # layer 1: convolutional - Input = 32x32x1. Output = 28x28x12
    strides = [1,1,1,1]
    conv1 = tf.nn.conv2d(x, weights['c1'], strides=strides, padding='VALID')
    conv1 = tf.nn.bias_add(conv1, biases['bc1'])
    # activation
    conv1 = tf.nn.relu(conv1)
    # Pooling. Input = 28x28x12. Output = 14x14x12
    ksize = [1,2,2,1]
    strides = [1,2,2,1]
    conv1 = tf.nn.max_pool(conv1, ksize=ksize, strides=strides, padding='VALID')
```