This repository Search Pull requests Issues Marketplace Gist

articuno144 / DMT2017

Unwatch 2 Star 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Settings Insights

Branch: master DMT2017 / main / CNN_functions_MMGonly.py Find file

articuno144 report draft gesture part 41ace33 3

1 contributor

56 lines (46 sloc) 1.75 KB Raw Blame History

```
import tensorflow as tf
import numpy as np
import pandas as pd

cali_d = pd.read_csv('cali_d.csv',sep = ',', header = None).values



n_input = 50 # input size
n_hidden_1 = 512 # 1st layer number of features
n_hidden_2 = 512 # 2nd layer number of features
n_classes = 9 # 5 gestures, 1 noise

def standardize_set(arr):
    a3 = np.multiply(np.add(arr[3,50::5],-250),0.01)
    a4 = np.multiply(np.add(arr[4,50::5],-250),0.01)
    a5 = np.multiply(np.add(arr[5,50::5],-250),0.01)
    return np.array([a3,a4,a5])

def conv2d(x, W, b, strides=1):
    # Conv2D wrapper, with bias and relu activation
    x = tf.nn.conv2d(x, W, strides=[1, strides, strides, 1], padding='VALID')
    x = tf.nn.bias_add(x, b)
    return tf.nn.relu(x)

def maxpool2d(x, k=2):
    # MaxPool2D wrapper
    return tf.nn.max_pool(x, ksize=[1, k, 1, 1], strides=[1, k, 1, 1],
                          padding='SAME')

# Create model
def conv_net(x, weights, biases, dropout):
    # Reshape input
    x = tf.reshape(x, shape=[-1, n_input,3, 1])
    # Convolution Layer
    conv3 = conv2d(x, weights['wc3'], biases['bc3'])
    # Max Pooling (down-sampling)
    conv3 = maxpool2d(conv3, k=2)
    conv3 = tf.nn.relu(conv3)
    # Convolution Layer
    conv4 = conv2d(conv3, weights['wc4'], biases['bc4'])
    # Max Pooling (down-sampling)
    conv4 = maxpool2d(conv4, k=2)
    conv4 = tf.nn.relu(conv4)
    # Fully connected layer
    # Reshape conv2 output to fit fully connected layer input
    fc2 = tf.reshape(conv4, [-1, weights['wd2'].get_shape().as_list()[0]])
```

```python
    fc2 = tf.add(tf.matmul(fc2, weights['wd2']), biases['bd2'])
    fc2 = tf.nn.relu(fc2)
    # Apply Dropout
    fc2 = tf.nn.dropout(fc2, dropout)
    # Output, class prediction
    out = tf.add(tf.matmul(fc2, weights['out']), biases['out'])
    return out

```