Project Report

Convolutional Neural Network

Convolutional Neural Network or CNN is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex

Convolutional Neural Networks have the following layers:

-Convolution

-ReLU Layer

-Pooling

-Fully Connected

CNN compares the images piece by piece. The pieces that it looks for are called features. By finding rough feature matches, in roughly the same position in two images, CNN gets a lot better at seeing similarity that whole-image matching schemes.

**CNN**

CNN will move the feature/filter to every possible position of the image

Step One - Line up the feature and the image

Step Two - Multiply each image pixel by the corresponding feature pixel.

Step Three - Add them up

Step Four - Divide by the total number of pixels in the feature

**Creating a Map to Put the Value of the Filter**

-Now to keep track of where that feature was, we create a map and put the value of the filter at that place.

**Sliding the Filter Throughout the image**

-Now, using the same feature and move it to another location and perform the filtering again.

**Convolution Layer Output**

Similarly, we will move the feature to all the other positions of the image and will see how the feature matches that area. Finally we will get an output of values in a matrix.

**Convolution Layer Output**

Similarly, we will perform the same convolution with all the other filters

**ReLU Layer**

-In this layer we remove every negative value from the filtered images and replace them with zeros.

-This is done to avoid the values from summing up to zero

**Rectified Linear Unit** (ReLU) transform function only activates a node if the input is above a certain quantity, while the input is below zero, the output is zero, but when the input rises above a certain threshold, it has a linear relationship with the dependent variable.

**Pooling Layer**

In this layer we shrink the image stack into a smaller size

Steps:

1. Pick a window size (usually 2 or 3).

2. Pick a stride (usually 2).

3. Walk your window across your filtered images.

4. From each window, take the maximum value.

**Calculating the Maximum Value in each Window**

Start with the first filtered image

In the first window the maximum or highest value is 1, so we track that and move the window two strides.

**Output After Passing Through Pooling Layer**

Reduces the input matrix to a smaller output matrix

**Stacking up the Layers**

Adding additional layers of Convolution, ReLU, and Pooling will reduce the output matrix further.

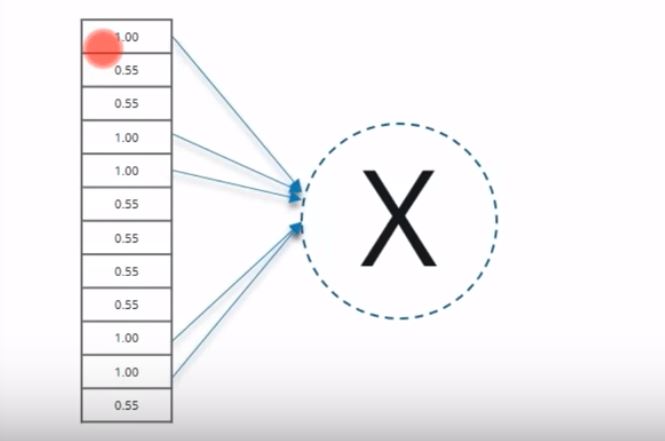
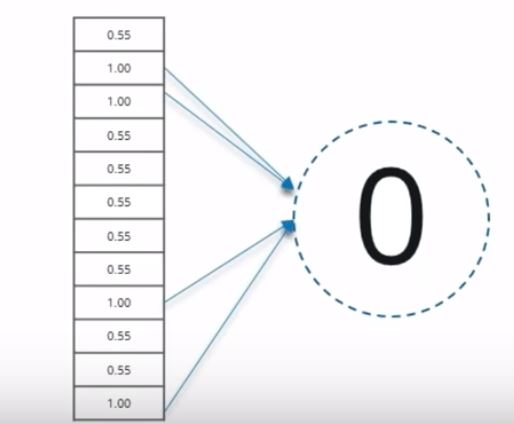
**Fully Connected Layer**

This is the final layer where the actual classification happens.

Here we take our filtered and shrunken images and put them in a single list

**Output**

When we feed in, ‘X’ and ‘O’. Then there will be some element in the vector that will be high. Consider the image below, as you can see for ‘X’ there are different elements that are high and similarly, for ‘O’ we have different elements that are high.

**Prediction**

Consider the list of a new input image

A new input image passes through all of the layers

**Comparing the Input Vector with X**

This new list is compared with the trained list that has been separated into different classes.

**Result**

After this is done the input image is identified with some level of accuracy.

**Implementing the Use-Case**

Download the dataset

Function to encode the labels

Resize the image (X) x (X) pixel and read as greyscale

Split the data 50% for training and 50% for testing

Reshape the data appropriately for TensorFlow

Build the Model

Calculate loss function, it is categorical cross entropy

Optimizer with learning rate set to 0.001

Train the Deep Neural Net for (X) epochs

Make predctions