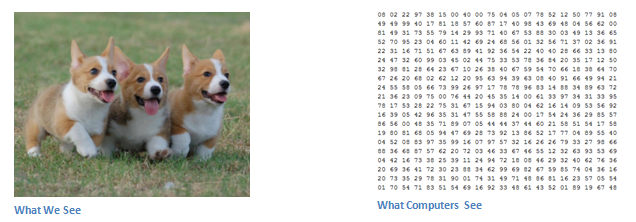
different architecture samja dena. And flask se webapp ka jitna likhna hai utna likh lena

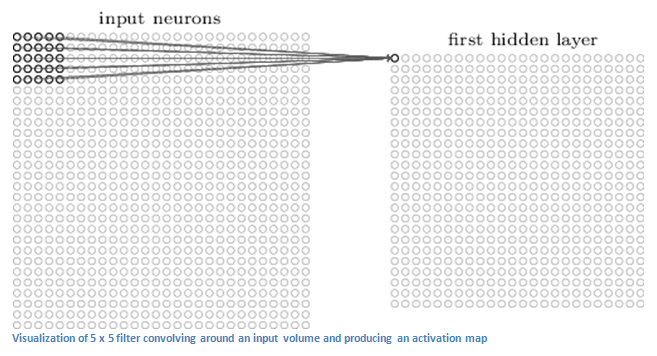
**Understanding Convolutional Neural Networks**



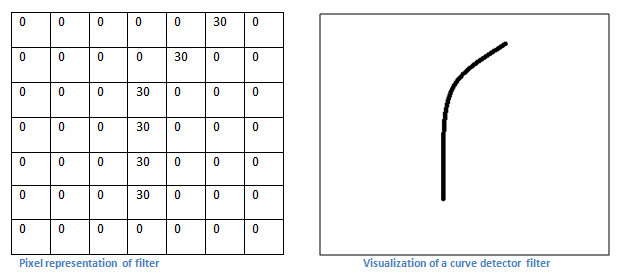
Image classification is the task of taking an input image and outputting a class or a probability of classes that best describes the image. When a computer sees an image it will see an array of pixel values as shown below. Each of these numbers is given a value from 0 to 255 which describes the pixel intensity at that point.

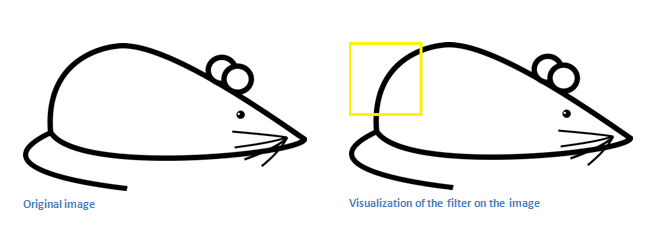


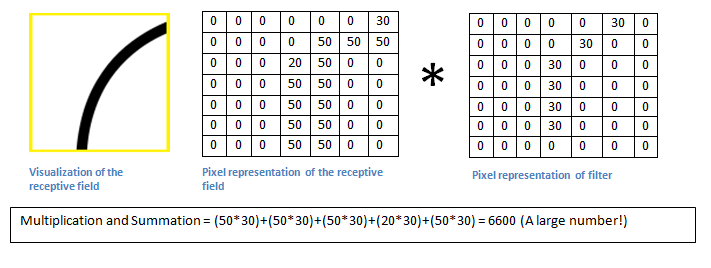
Now, when we look at a picture of a dog, we can classify it because the picture has identifiable features such as paws or 4 legs. In a similar way, the computer is able perform image classification by looking for low level features such as edges and curves, and then building up to more abstract concepts through a series of convolutional layers.The first layer in a CNN is always a **Convolutional Layer**. Now, imagine a conv layer is a flashlight that is shining over the top left of the image. Let’s say that the light this flashlight shines covers a 5 x 5 area. And now, let’s imagine this flashlight sliding across all the areas of the input image. In machine learning terms, this flashlight is called a **filter**. Now this filter is also an array of **weights** or **parameters**. As the filter is sliding, or **convolving**, around the input image, it is multiplying the values in the filter with the original pixel values of the image (ie, computing **element wise multiplications**). These multiplications are all summed up. So now you have a single number. Now, we repeat this process for every location on the input volume. Every unique location on the input volume produces a number. After sliding the filter over all the locations, you will find out that what you’re left with is a 28 x 28 x 1 array of numbers, which we call an **activation map** or **feature map**.

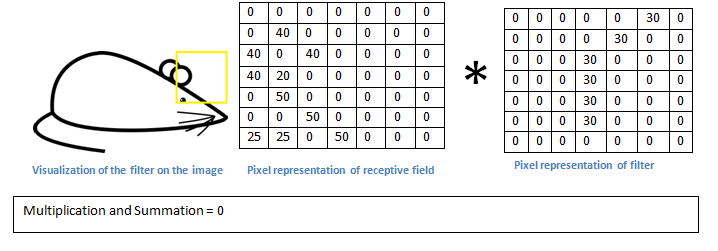


Each of these filters can be thought of as **feature identifiers**. These features can be straight edges, simple colours, and curves. Let’s say our first filter is a curve detector. As a curve detector, the filter will have a pixel structure in which there will be higher numerical values along the area that is a shape of a curve.





Basically, in the input image, if there is a shape that generally resembles the curve that this filter is representing, then all of the multiplications summed together will result in a large value! Now let’s see what happens when we move our filter.



The value is much lower! This is because there wasn’t anything in the image section that responded to the curve detector filter. The output of this conv layer is an activation map. So, in the simple case of a one filter, the activation map will show the areas in which there at mostly likely to be curves in the picture.

The output of the first conv layer becomes the input of the 2nd conv layer. The input of the first layer was the original image. However, for the 2nd conv layer, the input is the activation map(s) that result from the first layer. So, each layer of the input is basically describing the locations in the original image for where certain low-level features appear. Now when you apply a set of filters on top of that (pass it through the 2nd conv layer), the output will be activations that represent higher level features. As you go through the network and go through more conv layers, you get activation maps that represent more and more complex features. Types of these features could be semicircles (combination of a curve and straight edge) or squares (combination of several straight edges). By the end of the network, you may have some filters that activate when there is handwriting in the image, filters that activate when they see pink objects, etc. Another interesting thing to note is that as you go deeper into the network, the filters begin to have a larger and larger receptive field, which means that they are able to consider information from a larger area of the original input volume.

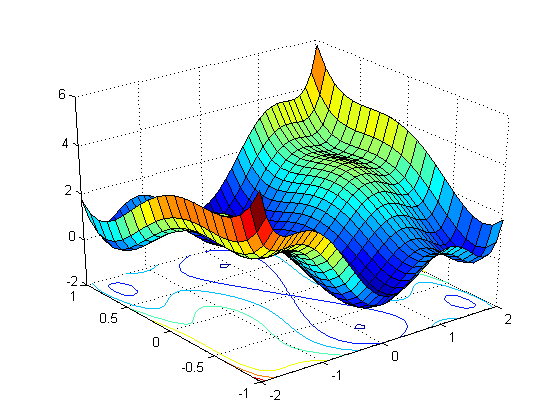
The end of the CNN is a **fully connected layer**. This layer basically takes an input and outputs an N dimensional vector where N is the number of classes that the program has to choose from.

**How does the CNN know what should be the weights in a filter?**

Before training, all the weights or filter values are randomized. During the **forward pass**, you take a training image and pass it through the whole network. Since all of the weights were randomly initialized, the output will probably be something like [.1 .1 .1 .1 .1 .1 .1 .1 .1 .1], this goes to the **loss function**. A loss function can be defined in many different ways but a common one is MSE (mean squared error), which is ½ times (actual - predicted) squared.

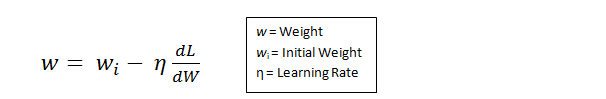


Initially the loss will be extremely high for the first couple of training images. Now, we want to minimize the amount of loss we have.



The loss function could look like this. Our objective is to find the lowest point of this curve.

This is the mathematical equivalent of a **dL/dW** where W are the weights at a particular layer. Now, what we want to do is perform a **backward pass** through the network, which is determining which weights contributed most to the loss and finding ways to adjust them so that the loss decreases. Once we compute this derivative, we then go to the last step which is the **weight update**. This is where we take all the weights of the filters and update them so that they change in the opposite direction of the gradient.



**Flask Web-app**

[Flask](http://flask.pocoo.org/) is a web development framework. A [framework](https://www.fullstackpython.com/web-frameworks.html) "is a code library that makes a developer's life easier when building reliable, scalable, and maintainable web applications" by providing reusable code or extensions for common operations. There are a number of frameworks for Python, including, [Tornado](http://www.tornadoweb.org/en/stable/), [Pyramid](https://trypyramid.com/), and [Django](https://www.djangoproject.com/). We have used Flask and Django to develop separate web apps.