Convolutional Neural Networks

James Della-Giustina, Johnathan Fisher, Harry Pham

March 31, 2021

What exactly is a Convolutional Neural Network??

Convolutional Neural Networks (CNN) are a technique in deep learning commonly used for image analysis and classification. They can be utilized to identify faces, tumors, street signs and garner other important information from images, regardless of perspective or sizing. [3]

CNN Overview

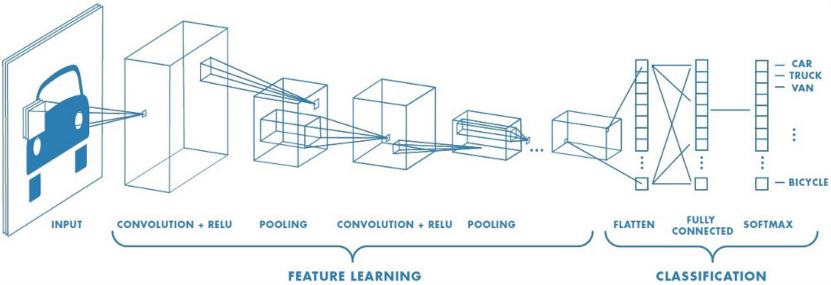


Figure 1: Map of the CNN processes. [4]

CNN: Step by Step

We begin with the input layer that will read in an image. The dimensions of the image may be altered, either up or down but usually to facilitate a square image i.e. some dimension of $n \times n$. For each pixel in a gray-scale image a value representing brightness is stored in a $n \times n$ matrix that preserves the spatial relationships of the image. In a full color RGB image, three separate matrices are formed, one for each pixels components that make up its RGB value.

Input Decomposition

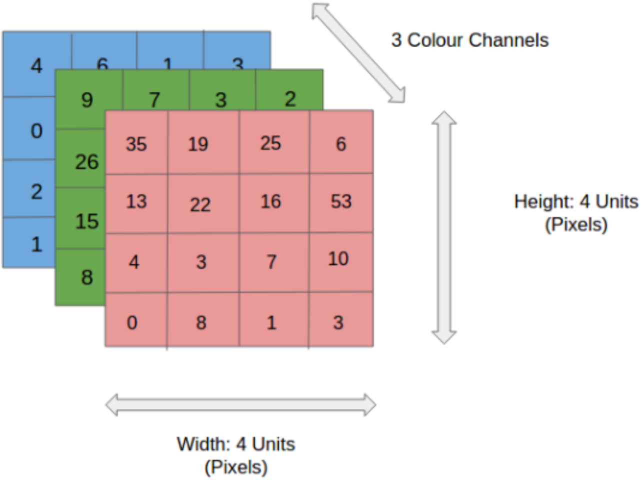


Figure 2: Three separate matrices for each associated RGB value [2].

Convolution Stage

Next we begin to pass our $n \times n$ matrices to our convolution layer. Here, we will take a **filter/kernel** which is a **small** $m \times m$ matrix that is populated with weights, and compute the dot product of a $m \times m$ sample of our original matrix, eventually passing over the entire image in a uniform manner. These dot product results are stored within what is called an **activation map**, another $m \times m$ matrix.

Non-Linearity

As each activation map is generated, a non linear function such as ReLU is applied to each entry in the map:

$$\text{ReLU}\,(x_{i,j}) = \begin{cases} x_{i,j} \text{ if } x_{i,j} > 0 \\ 0 \text{ otherwise} \end{cases}$$

Convolutional Formula

Mathematically, given a input image f and a kernel h, then for our activation map [5];

$$G[m,n] = (f \times h)[m,n] = \sum_j \sum_k h[j,k] f[m-j, n-k]$$

*Note that though we have been denoting our activation map as $m \times m$, but for clarity of the operation above we choose the dimensions of our activation map to be $m \times n$.

Populating an Activation Map

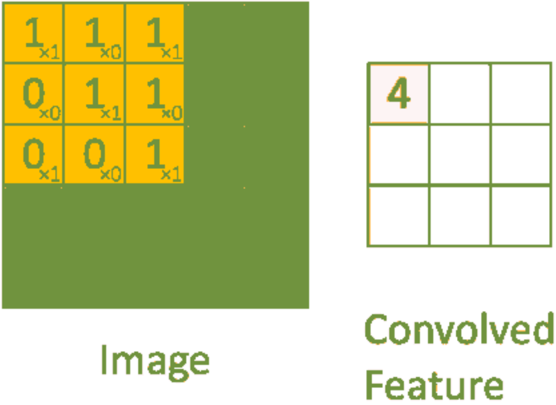


Figure 3: Passing a filter over an image to populate an activation map or convolved feature [2]

Activation Maps and Max Pooling

When all k filters are passed over the image, we stack each $m \times m$ activation map on top of each other, thus resulting in a $m \times m \times k$ 3-D volume.

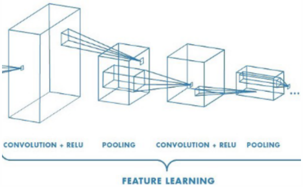
A second **smaller** filter will be passed over each activation map, let's say some 2×2 filter that will then evaluate and find the **maximum** of those 4 entries in the original $m \times m$ activation map.

The resulting entries are stored in a $\frac{m}{2} \times \frac{m}{2} \times k$ 3-D volume, which results in reduced computations while preserving the spatial characteristics of the original image.

Rinse and Repeat

At this stage there are several options available. We could:

- Add more convolution layers and pass additional filters over our reduced activation map. After a new activation volume is produced we once again perform a pooling technique to continually decrease the volume.



- We can **flatten** our activation volume. So for our $\frac{m}{2} \times \frac{m}{2} \times k$ volume we adjoin every column of each map in our k-depth volume to a single column, which results in a $(\frac{m}{2} \cdot \frac{m}{2} \cdot k) \times 1$ vector.

CNN Processes

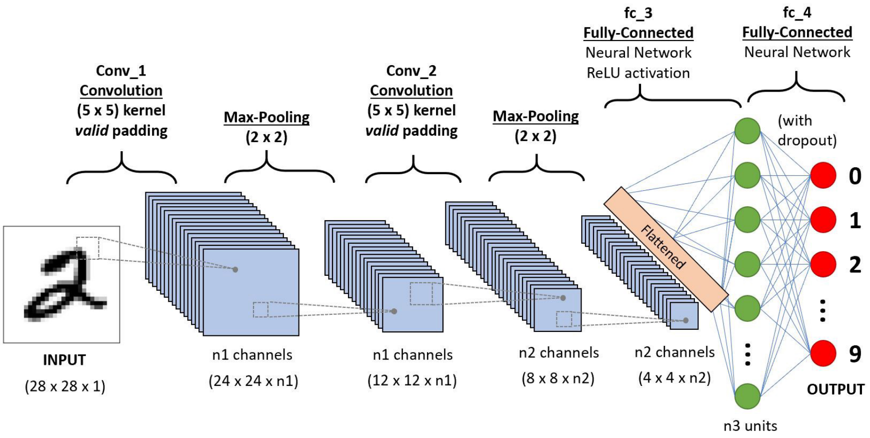


Figure 4: A typical overview of the processes a CNN takes while classifying an image, [2]

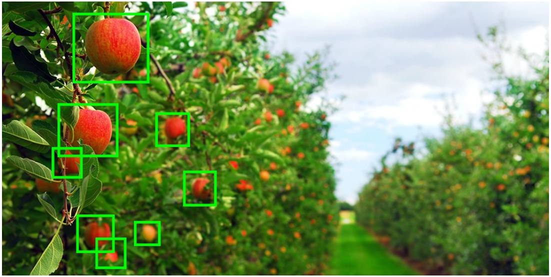
Benefits of a CNN

One extremely important aspect of convolutional neural networks is their ability to 'learn' characteristics or patterns within an image without explicit instruction from the engineer. The **filters** are created by hand, but eventually **backpropogation** with **stochastic gradient descent** computes the total loss and then working backwards from the last convolution layer, updates weights in our filters as needed to minimize the loss. This process is what truly allows the network to 'learn' filters and identify objects and patterns within an image.

Applications

Utilizing the ideas of CNN allows one for countless applications! An immediate application is facial recognition and reCAPTCHAs, while a subtle application is the prediction of chemical reactions [1]. More applications include:

- Recreating decayed/destroyed art
- Precision farming
- Social media facial filters



References

[1] Jure Gasteiger Johann; Zupan. *Neural Networks in Chemistry*. 1993. URL: http://web.uni-plovdiv.bg/plamenpenchev/mag/files/ang_chem2.pdf.

[2] Elnaz Jahani Heravi Hamed Habibi Aghdam. *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. Springer, 2017. ISBN: 978-3-319-57550-6.

[3] Chris Nicholson. *A beginner's guide to convolutional neural Networks (CNNs)*. URL: https://wiki.pathmind.com/convolutional-network.

[4] Sumit Saha. *A Comprehensive Guide to Convolutional Neural Networks-the ELI5 way*. 2018. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[5] Piotr Skalski. *Gentle Dive into Math Behind Convolutional Neural Networks*. 2019. URL: https://towardsdatascience.com/gentle-dive-into-math-behind-convolutional-neural-networks-79a07dd44cf9.

Questions?