Convolutional Neural Networks

Aditya Chakka & Shrey Gupta

What are convolutional neural networks?

Convolutional neural networks(CNNs) are a variant of the deep learning algorithm, neural networks, which capture the spatial features of an image. Their use of spatial relationships enables them to play a large role in computer vision and image recognition. Though CNNs are designed for image classification tasks, they have been used in natural language processing, genomics, and astrophysics. For the purpose of this article, we will focus on image classification, and any examples given will be in the context of image classification.

CNN Architecture

The architecture of a CNN includes multiple layers, including the convolutional layer, which are all used to identify features of an image. The standard CNN includes input data, convolutional layer, the pooling layer, the fully connected layer input and output layer, and the error. There are multiple algorithms used to train neural networks, but two very common algorithms are forward propagation and backward propagation, both of which utilize the layers mentioned.

Before going into depth about the various layers, lets go over a general overview of the various layers and how they fit together in CNNs.

- Convolutional Layer - The goal of the convolutional layer is to extract the features from the input data to a feature map. It essentially scans over an image with a kernal, which is basically a "filter,", performs dot product calculations, and stores the data it extracts on a matrix. The resulting matrices scanned by the various filters are the convolutional layers and are used as parameters. Multiple convolutional layers allow models to learn low level features early on and more complex features deeper into the network.

- Pooling Layer - Since convolutional layers record the exact position of feautures, slight changes to the feature positions creates a new feature map. To solve this problem, the pooling layer creates a downsampled or pooled feature map containing essentially a summary of the important features so that slight changes in feature location won't result in a new feature map. It

- Fully Connected Layers(Input and Output) - These layers process the data for inputting to the next layer or classify the image for the output. They take the data from the

Key Concepts

Feature Maps or Activation Maps

A feature map is the result of applying filters to an image, they map the output of the filter at every position which the filter strides over. The feature map corresponds to where a feature is in the image and the activation The number of feature maps is the same as the number of layers.

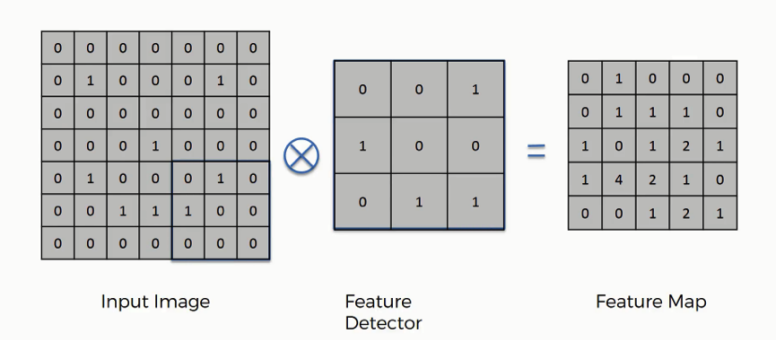

Figure 1: Feature Map

Strides & Padding

The stride controls how the filter moves across the input image, meaning it determines how many pixels the filter shifts when convolving. For example, if the stride was 1, then the filter would shift by 1 pixel. Figure 5 and figure 6 show a stride of 1 and 2 respectively.

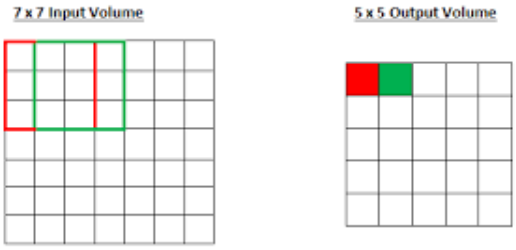

Figure 2: 3x3 filter with a stride of 1.

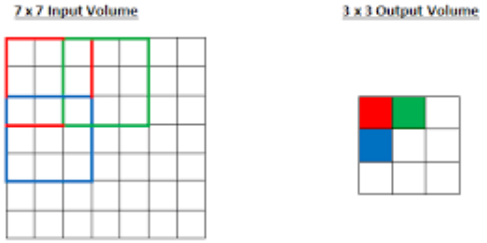

Figure 3: 3x3 filter with a stride of 2.

Padding is when a layer of zeroes are added to the border of the input image. This is done to ensure that each pixel is at the center of the filter. Pixels at the edges of an image will not get the chance to be in the center of the filter, therefore adding additional zeroes to the sides means that the filter can start off the image and then it can begin stepping across the image which results in the edge pixels becoming the center of a filter. By padding the input image, the output will also retain a larger portion of information of the input image, which helps extract lower level features.

ReLU(Rectified Linear Unit)

ReLU or rectified linear unit is a common activation function that is passed over the input data after the filter is applied. All of the negative cells that were created as a result of the filter are turned into zeros and the positive values remain the same. It is strictly that simple, if the value is below 0 return 0, otherwise return the number. These are examples of what ReLU does.

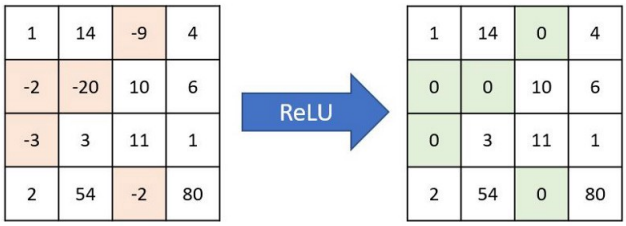

Figure 4: Image data after ReLU function is applied

Pooling

The purpose of pooling is to reduce the size of the inputted image, which greatly reduces the computational time required to train the model. It also maintains the dominant features while eliminating the unrequired ones. This allows for

the model to stay as effective as it was before pooling. There are two kinds of pooling: average pooling and max pooling.

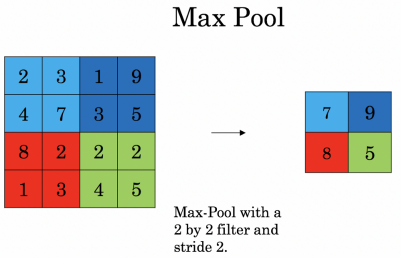

Figure 5: Example of max pooling

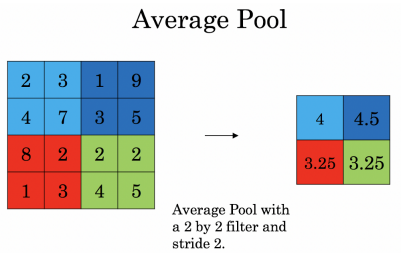

Figure 6: Example of average pooling

While the two choices may seem arbitrary, each has their own benefits. Max pooling is useful when dealing with a dark background and are primarily looking at the lighter pixels because this kind of pooling highlights the lighter pixels. This also means max pooling is preferred when dealing with specific features of an image. Average pooling smooths the image out and is useful for when every aspect of the image is useful in classification.

Fully Connected

The fully connected layer represents the last few layers in a convolutional neural network. Each of the neurons in these layers are connected to the other layers' neurons, hence the name fully connected. The input to this layer is flattened, which means that it is turned from its 3-dimensional matrix form to a vector, reducing the dimensionality of the data. Not all models are fully connected because it greatly increases computational time and in some cases can increase loss. The output of this process in the neural network is what determines how the model classifies the image.

The Convolutional Layer

The convolutional neural network differs from other neural networks in that it performs an operation known as convolution. Convolutional involves extracting the features from an image through the use of a filter or a kernel.

To understand how the input image is processed we have to understand channels. Images containing color are represented as 3D and have 3 channels, one for red, one for green, and one for blue, while grayscale images are represented as 2D have only 1 channel. The input of the convolutional layer contains the width and height of an image but adding the color component adds a 3rd dimension representing the color. An important thing to know is that the filter has the same number of channels as the input image.

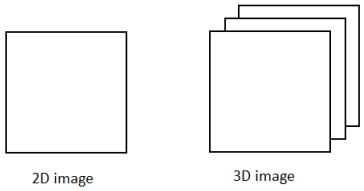

Figure 7: Example of multiple channels

So far we mentioned that the convolutional layer is for extracting the features from an image, meaning they work with the original image and its pixel data. Convolutional layers are not necessarily limited to only being used on the input image, they are also used at the output of other layers, usually the pooling layer. Consider Figure 8, the process of feature extraction, applying the activation function, and pooling is repeated numerous times before reaching a fully connected layer for classification.

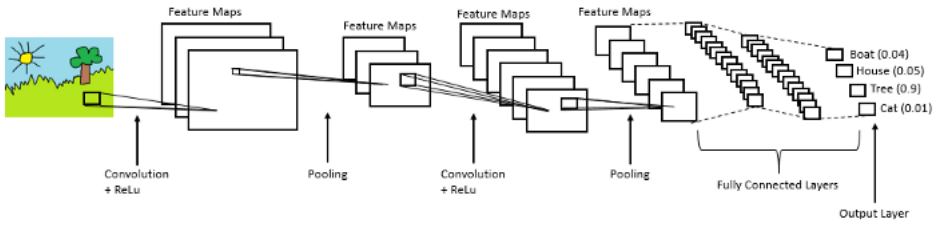

Figure 8: CNN Architecture

The layers first begin extracting lower level features and eventually, through a combination of multiple low level features, begin to extract higher level features which can be used for classification.

The convolutional layer works by stepping a filter, which is a matrix of weights, over a matrix containing the light intensity levels of each pixel of an image, and multiplying the corresponding values of the filter and input image

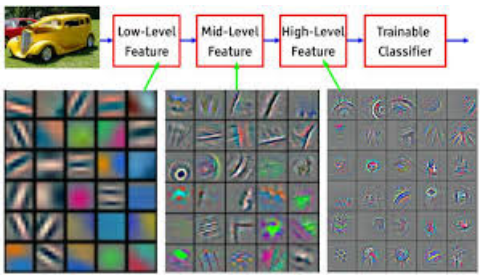

Figure 9: Low and high level features

matrix, summing them, and putting the result of the operations performed on a feature map. There are multiple filters applied to the image each used to extract different features of it.

Figure 10: Filter stepping across image.

Forward Propagation

Forward propagation is the action of feeding the data in a forward direction through the neural network. Depending on how many hidden layers you have, the data will be processed for each one based on the activation function and passed on the next layer. Essentially, it is how the data changes and moves through each layer until your desired output. This is present in every neural network as each involves manipulating input data to create reliable and accurate output data. In a convolutional neural network, the first step is to load the input images into a variable. Then, these images are put into the convolutional layer where a filter is used to extract the relevant information. The data then is put through an activation function which maps the input data to a relevant output dependent on the activation function. Next, it is linearly transformed with a bias matrix and then another activation function is applied. These steps happen

after each layer to properly process the data and then is passed onto the next layer.

Backward Propagation

Now that the data has been passed through, back propagation is used to tweak the weights, biases, and filters so that the predictions of the images are more accurate. It would take forever if random tweaks were made, so that is clearly not practical. This is where gradient descent comes into play. Gradient means the direction of steepest increase and if we take the opposite direction, we get the term, gradient descent.

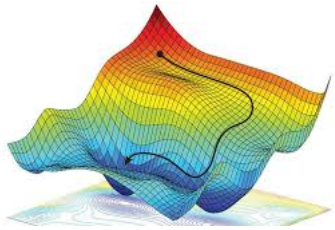

Figure 11: Gradient descent visual

To better understand this let's imagine we are at the top of a bowl and our goal is trying to get to the bottom, but we don't know where the bottom is. Our first step is going to be to calculate the gradient, or the direction we want to go, and then take a small step in that direction. We keep repeating this process until we are unable to notice any progress. To compare this with a more practical scenario, let's take an ML model with 10,000 weights. We run through our model and get a high cost value. Then we start the process of gradient descent which would tell us what we need to change to each weight in order for us to get closer to a lower cost. We keep repeating this process until the model is not showing any improvement. This ties to CNNs because the weights, filters, and biases involved, are all parameters that are part of gradient descent.

References

1. CNN vs. RNN vs. ANN – Analyzing 3 Types of Neural Networks in Deep Learning by Aravind Pai
2. Demystifying the Mathematics Behind Convolutional Neural Networks (CNNs) by Aishwarya Singh
3. Convolutional Neural Network Architecture: Forging Pathways to the Future

4. How Do Convolutional Layers Work in Deep Learning Neural Networks? by Jason Brownlee

5. An introduction to Convolutional Neural Networks by Christopher Thomas