Ankit Mishra

Senior Analyst, Mphasis NEXTlabs

Ankit Mishra, a Senior Analyst at Mphasis NEXTlabs, works primarily on conceptualizing and developing different algorithms for document image processing and Natural Language Processing based projects. He has experience in working on Deep Learning technologies and Semantic Web.

Ankit holds a Master of Technology degree in Data Sciences from International Institute of Information Technology, Bengaluru (IIITB) and Bachelors of Engineering from Institute of Engineering and Technology, Devi Ahilya University, Indore.

About Mphasis

Mphasis is a global technology services and solutions company specializing in the areas of Digital, Governance, Risk & Compliance. Our solution focus and superior human capital propels our partnership with large enterprise customers in their digital transformation journeys. We partner with global financial institutions in the execution of their risk and compliance strategies. We focus on next generation technologies for differentiated solutions delivering optimized operations for clients.

Mphasis
Unleash the Next

Deep Learning in Computer Vision

Ankit Mishra
Senior Analyst, Mphasis NEXTlabs

NEXT labs

Introduction

Neural Networks are graph-like structures constituting electronic neurons as nodes with billions of connections between them. These require a lot of computational power and storage to operate them, discontinuing them back in the 90s when compute power was scarce. With the advent of powerful Graphical Processing Units and tensor manipulating libraries such as TensorFlow, neural nets and their variants are able to solve some of the complex problems in terms of intelligence and automation with decent speed and unbeatable accuracy.

In this paper we try to throw some light on Computer Vision, different methods used for it using deep learning methodologies. We will start our discussion with the overview of CV, feature representation of an image, then we move to feed forward NN, CNN and see how to use these methods for CV.

Abstract

Computer Vision is a field that deals with how computers and machines can be made proficient in understanding images and videos. Before deep learning this was done by image pre and post processing but with the dawn of deep learning this has become much more accurate and fast at such a scale that computer can now process live images as in self-driving cars.

This paper presents the concept and steps of usage of deep learning in CV. We present ways to prepare data for deep learning and the use of CNNs to automatically extract hundreds of features and use those to understand and recognize images.

We finally present results of a classification example performed on an image dataset containing images of cats and dogs.

Keywords: CV- Computer Vision, NN: Neural networks, ANN: artificial neural networks, CNN: convolutional neural networks.

What is Computer Vision?

Computer vision is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images. It involves the development of a theoretical and algorithmic basis to achieve automatic visual understanding.

Having a long history, studies in CV started in 60s which led to the development of algorithms that are still used e.g. edge detection, line labeling etc. These algorithms are manual ways of making the computer to understand images. Machine learning in CV started in previous decade when statistical machine learning algorithms like SVM came into picture. Here, image features were extracted manually and fed into the algorithm.

When deep learning became feasible, Convolutional Neural Networks emerged as an efficient way to process and learn images, which, by far, is the state-of-the-art method for image understanding. CV tasks in deep learning include image classification and recognition, text spotting, image caption generation to name a few.

Image Representation

Images are a matrix of pixels where each pixel can take a value from 0 to 255, in each channel. These numbers signify the intensity of color in that channel.

Channel: Every image pixel is composed of combination of primary color components. A single channel is the grayscale image of one of the primary color components. An image has such multiple channels of different color components E.g. a RGB image constitutes Red, Green and Blue channels. A grayscale image consists of only 1 channel.

Since image itself is a stack of matrices (1 for each channel) of numbers, it is represented as a *tensor* (channels x length x breadth). To a neural net, image is passed in this format after which various features are extracted out of it.

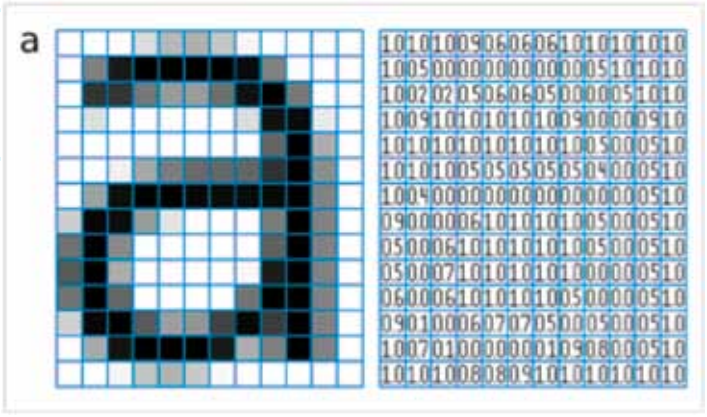

Figure 1

Image Feature Extraction

An image is assumed to have large regions of similar properties such as color, gradient etc. Feature extraction from an image is based on generalizing these regions based on different properties such as color, edges etc. [1] e.g. large region of an image can be approximated of having same color and so that feature of color can be extracted from that image.

Image in a matrix form is passed to the Convolutional Neural Network for extracting features out of the image. CNNs use different filters to extract a feature peculiar to that filter. These features in matrix form are called *feature maps*. These are then compressed into smaller vectors which can be passed to fully- connected neural layer to perform e.g. classification.

Convolutional Neural Networks

Convolutional Neural Networks or CNNs are a 1-dimensional or 2-dimensional structure (practically) of artificial neurons that work on the principle of local receptive fields and weight sharing [2] [3].

Hubel and Wiesel [2] analyzed the visual cortex of a monkey and concluded the cortex's structure to follow the structure as CNN with the two properties. These two properties are captured by the convolution filter that we use in the CNN.

Local Receptive Fields

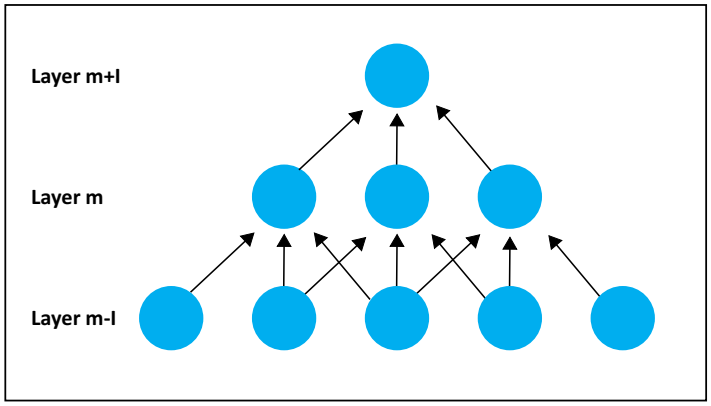

Figure 2

Imagine that layer **m-1** is the input layer. We will call this layer as retina. In the above figure, units in layer **m** have receptive fields of width 3 in the input retina and are thus only connected to 3 adjacent neurons in the retina layer. Units in layer **m+1** have a similar connectivity with the layer below. We can say that their receptive field with respect to the layer below is also 3, but their receptive field is larger with respect to the input i.e. 5. Each unit is unresponsive to variations outside of its receptive field with respect to the retina. The architecture thus ensures that the learnt "filters" produce the strongest response to a spatially local input pattern [4].

Thus, the inputs of hidden units in layer **m** are from a subset of units in layer **m-1**, units that have spatially contiguous

receptive fields.

Shared Weights

In CNNs, each filter is replicated across the entire visual

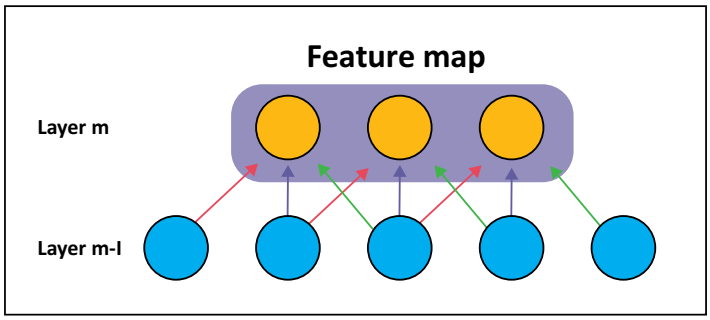

Figure 3

field. These replicated units share the same parameters (weight vector and bias) and form a feature map.

In the above figure, we show 3 **units in the hidden layer** belonging to the same feature map. Weights of the same color are shared—constrained to be identical.

Replicating units in this way allows for features to be detected *regardless of their position in the visual field*. Additionally, weight sharing increases learning efficiency by greatly reducing the number of free parameters being learnt. The constraints on the model enable CNNs to achieve better generalization on computer vision problems [4].

CNN Architecture and Working Example

The CV model of CNNs, for different problems, is trained in a particular sequence only, using which learning can happen with a satisfiable accuracy.

Here, *we are going to perform image classification* between images of dogs and cats. For this problem, the CNN architecture comprises of following steps or layers:

1. Convolution

This is the layer where **N** no. of filters are chosen of size *m x n* (where **N** = Max number of features desired) and each filter is made to slide over the image of size *a x b* (where *m, n < a, b*) thereby performing convolution operation i.e. multiplying each pixel of the image by each cell of filter and adding all these values which are covered by the filter in image area.

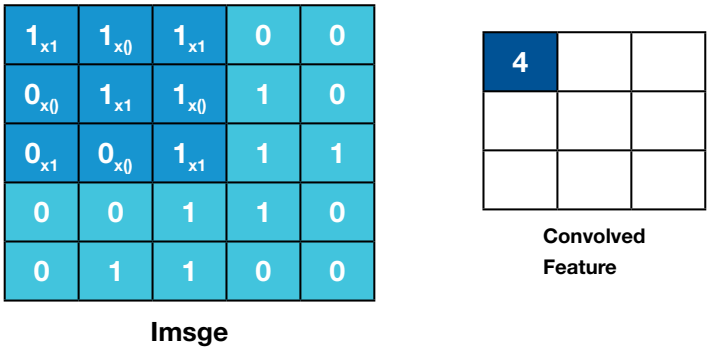

Figure 4

Here, a filter, *101010101* is made to slide (convolved) over the image to get output 4.

2. Pooling

This layer is placed to reduce the size of convolved image, yet keeping the features obtained after convolution, intact. It does this by sliding a small window (size < image-size)

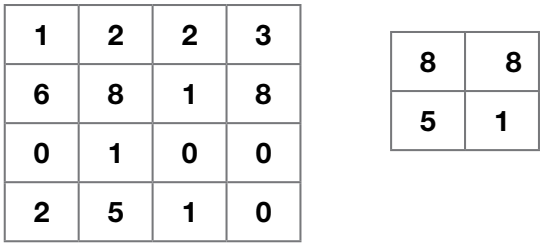

Figure 5

over the image, *extracting the max value* out of the area under the window. In this way, the max values in the feature map get preserved while the map's size also gets reduced.

3. Flatten

Pooled (reduced) feature maps can't be passed to the fully connected layer in matrix form. To convert matrix form of pooled feature maps to single dimensional array, flatten layer is used.

4. Fully Connected Layer

At last, fully connected layer is used to take the single dimensional array of pooled features to classify images.

In the given classification problem, training set contained 8000 images; test set contained 2000 images.

Sample Images

For this problem, two architectures were tested:

1. *Single Convolution-Pooling*

Only 1 convolution layer and pooling layer is used.

Results

Training Set Accuracy - **84.5%**

Test Set Accuracy- **75%**

Loss Function - *Binary Cross-entropy*

Optimizer - *Adam*

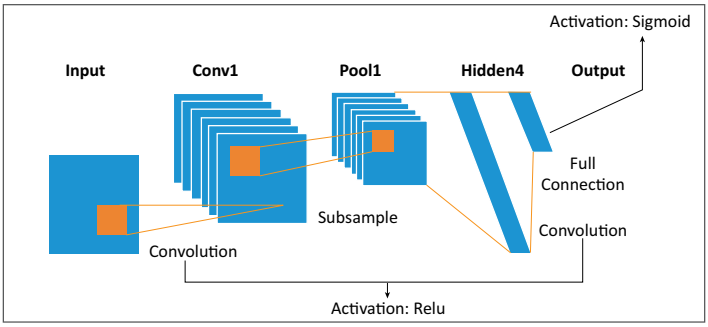

Figure 6

2. *Double Convolution-Pooling*

2 layers each of convolution and pooling is used. Its architecture is same as that of LeNet.

Results

Training Set Accuracy - **85%**

Test Set Accuracy - **82%**

Loss Function - *Binary Cross-entropy*

Optimizer - *Adam*

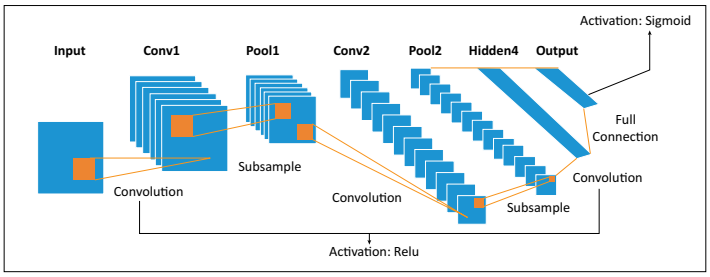

Figure 7

References

[1] B. Yoshua, "Learning Deep Architectures for AI," Foundations and Trends in Machine Learning, vol. 2, no. 1, p. 71, 2009.

[2] D. Hubel and T. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," Journal of Physiology, p. 32, 1968.

[3] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in IEEE, 1998.

[4] DeepLearning.net, "LeNet," DeepLearning.net, [Online]. Available: http://deeplearning.net/tutorial/lenet.html. [Accessed March 2017].