Convolution Neural Network Based Image Recognition

Sofia Hamid¹, Mrigana Walia²

¹M-Tech Scholar, Universal Group of Institutions, Lalu, Punjab, India  
sofihamid166[at]gmail.com

²Assistant Professor, Universal Group of Institutions, Lalu, Punjab, India  
mriganawalia[at]ugichd.edu.com

Abstract: The identification of an image is one thorny task in computer vision problems. Using the python programming language and a few other machine learning algorithms and python libraries, this image recognition and feature extraction can be achieved. In pattern- and image-recognition problems, convolutional neural networks (CNNs) are commonly used as they have a range of benefits compared to other techniques. The basics of CNNs are covered in this document, including a description of the different layers used. In this paper, we presented a model for cat and dog image recognition from the Convolutional Neural Network (CNN). Before feeding it to the CNN model, the cat and dog dataset is pre-processed.

Keywords: Image processing, pooling layer, convolutional layer, Deep Learning, Tensor flow etc

1. Introduction

In the world, there is an immense amount of picture data, and the rate of growth itself is growing. Many of the images are processed or distributed on the Internet on cloud services. For a wide range of image-related activities, automatic processing of image information is useful. This means, for computer systems, crossing the so-called semantic distance between the information contained in the image at the pixel level and the human understanding of the same images. Object detection from the image library is a difficult task in the field of computer vision. In terms of memory and speed, CNNs have a high computational cost in the learning stage, but can achieve some degree of invariance of shift and deformation.

Image processing was described by Lillsand and Kiefer as involving the manipulation of digital images using computers. It is a broad topic and includes mathematically complex processes. Some simple operations are involved in image processing, such as image restoration/rectification, image enhancement, image classification, fusion of images, etc. The classification of images forms an essential part of processing images. Automatic allocation of images to thematic groups is the purpose of image classification. Supervised classification and unsupervised classification are two types of classification. The image classification process involves two stages, followed by testing and training of the system. The training method makes taking the characteristic characteristics of the images (forming a class) and forming a special definition for a specific class. Depending on the form of classification issue, the process is performed for all classes; binary classification or multi-class classification. The testing phase involves classifying the test images into different groups for which the system has been educated. This class assignment is done on the basis of the partitioning of the training features between classes.

2. Convolution Neural Network

One of the most common approaches to deep learning, where multiple layers are robustly fitted, is Convolutionary Neural Networks (CNN). It has been found to be highly effective, and is also the most commonly used in different computer vision applications. In general, a CNN is made up of three central neural layers, which are fully connected convolutionary layers, pooling layers, and layers. Different layer forms perform various functions. In two steps, the network is trained: a forward and a backward step. Second, the main aim of the forward step is to represent the input image with the current parameters in each layer (weights and bias). The predictive output is then used to measure the cost of the loss with the ground-truth labels. Second, the backward step with chain rules calculates the gradients of each parameter based on the cost of loss. All parameters are updated and prepared based on gradients for the next forward calculation [4]. After sufficient iterations of the forward and backward stages, the network learning can be stopped. The convolutionary neural network consists of single or multiple convolutionary and sub-sampling layer blocks, which often include completely linked layers and finally an output block.

![Figure 1: Block diagram of CNN](image)

Convolutional Layer: The most significant layer in CNN is the convolution layer, since this layer converges with the input and transfers the output to the next layer. This is close to the concept of human vision. In the upcoming layer, this layer only links to unique (smaller) neurons. As a machine uses pixels to define an image, each pixel is provided with a

Volume 10 Issue 2, February 2021

www.ijsr.net

Licensed Under Creative Commons Attribution CC BY

Paper ID: SR21225214136  DOI: 10.21275/SR21225214136  1673
value. This layer has three mandatory characteristics, such as convolutional kernels, the number of input channels and the filter depth. A smaller image is considered and the pattern is compared with all the pixels in the given input (Large image) (Large image). Every single pixel is filtered by following certain steps: I Firstly, the filters are applied to each and every pixel of the images. (ii) each pixel is multiplied with the corresponding image (iii) all the pixels values have to be added. (iv) the value has to be divided by the feature pixel size. Hence, this layer outputs the lesser number of free parameters, allowing the network to be deeper with the free parameters.

**Pooling layer:** The input for this layer is taking from a small portion of the previous layer. The input is sampled into smaller proportions and produces a single output. Though there exist several non-linear functions such as max-pooling, mean pooling and average pooling for sub sampling, max pooling is widely used technique. This technique considers the maximum value of the pixel in the considerable window. This states the importance of ideal recognition of the related pattern, rather than the exact match. It continues pooling minimises the spatial size representation and amount of computation in the memory.

**Fully connected layer:** Final part of the CNN is fully connected layer; this layer comes into existence after several layers of convolutional and pooling layers. The high-level reasoning is performed in this layer. As like in artificial neural networks, neurons in this layer fully connected with the previous layers.

**Implementation Steps**

*Image Normalization* The first step in our approach is to normalise the input image. Images are given as a set of RGB values. These values are given as channels. Value of 1 equals grey scale and of 3 equals that of RGB. So, we need to normalise these values in a common range. Hence in the first layer we have normalise our images to RGB using scale.

*Rectified Linear Unit* We use ReLU to threshold the input. These are usually defined in the convolutional layer itself. In tensor flow there are two ways to define it. First to define it in the convolutional layer and secondly add a different layer that contains the activation function. We use the first option in our approach. The formula for rectified linear unit is given as:

\[ f(x) = \max(0, x) \]

*Convolutional Layer* The convolutional layer convolves a set of filters over the input. A high filter response indicates similarity between the filter and the input and vice-versa. The filter outputs obtained in this layer enables it to make a decision about the class of the input image. It does linear transformation from input to output without changing the dimensions of the input image but changing the number of channels in the output image. The convolutional layer takes the input with some pre-defined bias. Weights are defined for each input as well. These weights help us reduce the errors when we back propagate the error value and adjust it so as to improve the model.

Max pool layer, Themax pool operation is used to down-sample the images. This takes in an input of size and turns it into our desired size. In this step we change the number of rows and columns but the depth remains same. The max pool operation is important because it will not over fit to our data.

**3. Result and Discussion**

*Image 1*
Accuracy
Epoch 24/25
250/250 -------------------------- [767s 3s/step]
- loss: 0.2495 - accuracy: 0.8969 - val_loss: 0.5687
- val_accuracy: 0.8134

Epoch 25/25
250/250 -------------------------- [778s 3s/step]
- loss: 0.2363 - accuracy: 0.9031 - val_loss: 0.4470
- val_accuracy: 0.8189

Image 2

Predicted

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Size</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>prediction</td>
<td>str</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>result</td>
<td>float32</td>
<td>(1, 1)</td>
<td>[[0.]]</td>
</tr>
<tr>
<td>test_image</td>
<td>float32</td>
<td>(1, 64, 64, 3)</td>
<td>[[[239, 239, 239.], [239, 239, 239.]]</td>
</tr>
</tbody>
</table>

Accuracy
250/250 -------------------------- [375s 2s/step] - loss: 0.3572 - accuracy: 0.8411 - val_loss: 0.3720 - val_accuracy: 0.7915
Epoch 14/25
250/250 -------------------------- [372s 1s/step] - loss: 0.3436 - accuracy: 0.8503 - val_loss: 0.5882 - val_accuracy: 0.7845
Epoch 15/25
250/250 -------------------------- [369s 1s/step] - loss: 0.3309 - accuracy: 0.8556 - val_loss: 0.6287 - val_accuracy: 0.7426
Epoch 16/25
250/250 -------------------------- [375s 1s/step] - loss: 0.3139 - accuracy: 0.8624 - val_loss: 0.4741 - val_accuracy: 0.7804
Epoch 17/25
250/250 -------------------------- [395s 1s/step] - loss: 0.2983 - accuracy: 0.8675 - val_loss: 0.3586 - val_accuracy: 0.8001
Epoch 18/25
250/250 -------------------------- [369s 1s/step] - loss: 0.2741 - accuracy: 0.8827 - val_loss: 0.8830 - val_accuracy: 0.7801
Epoch 19/25
250/250 -------------------------- [373s 1s/step] - loss: 0.2654 - accuracy: 0.8860 - val_loss: 0.5239 - val_accuracy: 0.8102
Epoch 20/25
250/250 -------------------------- [370s 1s/step] - loss: 0.2572 - accuracy: 0.8923 - val_loss: 0.7086 - val_accuracy: 0.7830
Epoch 21/25
250/250 -------------------------- [373s 1s/step] - loss: 0.2520 - accuracy: 0.8947 - val_loss: 0.6816 - val_accuracy: 0.7781
Epoch 22/25
250/250 -------------------------- [369s 1s/step] - loss: 0.2335 - accuracy: 0.9015 - val_loss: 1.1730 - val_accuracy: 0.7936
Epoch 23/25
250/250 -------------------------- [376s 2s/step] - loss: 0.2212 - accuracy: 0.9106 - val_loss: 0.8443 - val_accuracy: 0.7989
Epoch 24/25
250/250 -------------------------- [372s 1s/step] - loss: 0.2150 - accuracy: 0.9133 - val_loss: 0.5440 - val_accuracy: 0.7915
4. Conclusion

Image classification based on their classes is easy for humans but difficult for machines. We plan to train a model on many images from various classes to build a model for prediction. In this research we develop an algorithm for classifying images of different classes like Cat and Dog. We used the Convolutional Neural networks (CNN) to extract and learn features of the images and train our model for classification. We tried various experiments to improve the accuracy on our test dataset and finally achieved an accuracy of 91% by this approach.

5. Future Work

We would like to intend to use more advanced networks in future work, which will help to train deep architectures and allow us to analyse the accuracy of our image recognition system.

References


