Convolutional Neural Networks for image classification

Julio Nicolás Reyes Torres
Universidad de los Andes
jn.reyes10@uniandes.edu.co

Juan David Triana
Universidad de los Andes
jd.triana@uniandes.edu.co

Abstract

This article presents a concept of "deep learning" to achieve high accuracy in image classifications problems, this the "Convolutional Neural Network (CNN)". The net is designed and implemented to analyze a specific problem of attribute classification. This work presents the concepts, techniques, components of a CNN and the description of evaluated architecture.

1. Introduction

A recognition problematic for computer vision relies on face detection with multiple features regarding specific people. Among these, certain characteristics are needed to be understood by recognizing different features, such as glasses, hair color, gender, among others. Various algorithms have been implemented to solve this problematic. HOG-SVM, can certainly obtain accurate descriptions of each image and can classify thoroughly each one with a specific discriminate hyper plane. Moreover, the state of the art algorithm relies on the use of Convolutional Neural Networks (CNN), which performs a series of convolutions over images, evaluating its results with the presented groundtruth and optimizing the function through different loss functions [2]. In this work, a face recognition algorithm using CNN is going to be evaluated on the CelebA dataset.

2. Methods

2.1. Data set

The dataset used in this work was the CelebA dataset, which has 200 thousand images of celebrities with 40 different attributes. Some of the classes attained for each image are: arched eyebrows, attractiveness, hair color, among others. [4]

2.2. Alternative Method

A way to address this problematic would be using a HOG-SVM method, where a description with histograms of oriented gradients over each image can be the target for further classification. Certainly, various problems could be seen using this method, since support vector machines are binary discriminative classifiers. This means that for every class or attribute adjacent to an image a support vector machine should be used, meaning a total of 40 SVMs. In general, the computation of this problem is expensive and unaccurate.

3. Convolutional Neural Network

The concept of convolutional neural network born from deep learning and has shown a high accuracy in areas of classification and recognition [6]. Many applications in machine learning are attacked by the Convolutional Neural Network (CNNs).

A CNN is understood as a combination of multiple algorithms that work well together [5]. The main difference between CNNs and a conventional neural net is the preprocessing. CNNs have two main parts:

1. **Feature engineering / preprocessing** turn our images into something that the algorithm can more efficiently interpret

2. **Classification:** train the algorithm to map our images to the given classes and understand the underlying relationship

The main parameters to develop a CNN are:

Convolution: The convolution process extract features from the input image using a kernel. This "filter" is used to learn image features preserving spatial relationship between pixels. The output of the convolution process is called: "convolved feature" or "feature map". This is important because this process involves a more optimal representation of the input image [6, 3]. In PytTorch this function is given

by the sentence: "*torch.nn.Conv2d()*". The kernel is defined with the following parameters:

- Kernel Size: size of the filter
- Kernel Type: values of the filter
- Stride: the rate at which the kernel run over the image. For example, a stride = 2 moves the kernel in 2-pixel increment.
- Padding: Extra columns and rows to make sure that the kernel properly passes over all the edges of the image.
- Output Layers: Number of different kernels applied to the image.

ReLU: A big problem in CNNs field is to add new layers, it could come across different problems as "vanish gradient problem". ReLU is a sensible nonlinear activation function that solves that issue [1]. It helps to approximate the relationship in the underlying data and it is given by: $Max(0, input)$.

Max Pooling: It is the last part of the feature engineering step in CNNs. The idea is to pass over sections of the image and pool them into the highest value for each one. In figure (1) is presented this process.

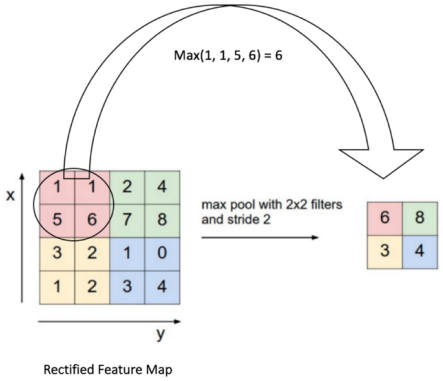

Figure 1: Max pooling to an image. Taken from: [3]

Fully Connected layer: The Fully Connected layer is known as the traditional Multi Layer Perceptron, which uses a softmax activation function in the output layer. The main idea is that the neurons of the previous layer are connected with the neurons of next layer.

The output from the convolutional and pooling layers represent high-level features of the input image and is the beginning to the Fully Connected layer. The sum of output probabilities from the Fully Connected Layer is 1 and is given by the Softmax activation function. The main idea of Softmax function is to take a vector of arbitrary real-valued scores and squashes it to a vector of values between zero and one that sum to one. [6]

References

[1] Adventures in Machine Learning. Convolutional Neural Net Tutorial in PyTorch.

[2] P. Agrawal, R. B. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. *CoRR*, abs/1407.1610, 2014.

[3] Algorithmia Blog. Convolutional Neural Nets in PyTorch.

[4] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[5] PyTorch Tutorials. Training a Classifier.

[6] Ujjwalkarn. An Intuitive Explanation of Convolutional Neural Networks, 2016.