A CONVOLUTIONAL NEURAL NETWORK FOR THE MNIST DATASET USING TENSORFLOW

Project report

FABIO LOCHE - STEFANO TEDESCHI

Corso di Reti Neurali
Anno Accademico 2016/2017

Corso di Laurea Magistrale in Informatica
Scuola di Scienze della Natura
Università degli Studi di Torino

ABSTRACT

Qua scriveremo l'abstract. . .

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth (Knuth, 1974)

ACKNOWLEDGMENTS

Put your acknowledgments here.

Many thanks to everybody who already sent me a postcard!

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Jim Young, Lorenzo Pantieri and Enrico Gregorio[1], Jörg Sommer, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, Roland Winkler, Jörg Weber, Henri Menke, Claus Lahiri, Clemens Niederberger, Stefano Bragaglia, Jörn Hees, and the whole LaTeX-community for support, ideas and some great software.

Regarding LyX: The LyX port was intially done by *Nicholas Mariette* in March 2009 and continued by *Ivo Pletikosić* in 2011. Thank you very much for your work and for the contributions to the original style.

1 Members of GuIT (Gruppo Italiano Utilizzatori di TeX e LaTeX)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

CNN	Convolutional Neural Network
MLP	Multylayer Perceptron

1 INTRODUCTION

2 CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Networks (CNNs) are a kind of artificial feed-forward neural network in which the organization of the connections between the neurons is inspired by the animal visual cortex. Actually, individual cortical neurons respond to stimuli in a restricted region of space known as the **receptive field**. The receptive fields of different neurons partially overlap such that they tile the visual field. Similarly, in CNNs each neuron is connected with only a small subset of inputs from the previous layer.

These networks are extremely useful when dealing with data with a grid-like topology, such as time-series data or images.

The name "convolutional neural network" indicates that the network employs a specialized kind of linear operation called **convolution**. In short, as stated in [Goodfellow, Bengio, and Courville, 2016], *convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers*.

2.1 MAIN FEATURES

While traditional Multylayer Perceptron (MLP) models were successfully used in the past for image recognition, due to the full connectivity between nodes they suffer from the curse of dimensionality and thus do not scale well to higher resolution images.

For instance, in the CIFAR-10 dataset, images are of size 32x32x3 (32 wide, 32 high, 3 color channels), so a single fully connected neuron in the first hidden layer of a regular MLP would have $32 * 32 * 3 = 3,072$ weights. A 200x200 image, however, would lead to neurons that have $200 * 200 * 3 = 120,000$ weights. Such network architecture does not take into account the spatial structure of data, treating input pixels which are far apart or close together exactly in the same way. The full connectivity of neurons is wasteful in the framework of image recognition, and the huge number of parameters quickly leads to overfitting.

As said before, convolutional neural networks are biologically inspired variants of multilayer perceptrons, designed to emulate the behaviour of a visual cortex. These models mitigate the challenges posed by the MLP architecture by exploiting the strong spatially local correlation present in natural images. In particular, CNNs have the following distinguishing features:

3D VOLUMES OF NEURONS The layers of a CNN have neurons arranged in 3 dimensions: width, height and depth. Neurons inside a

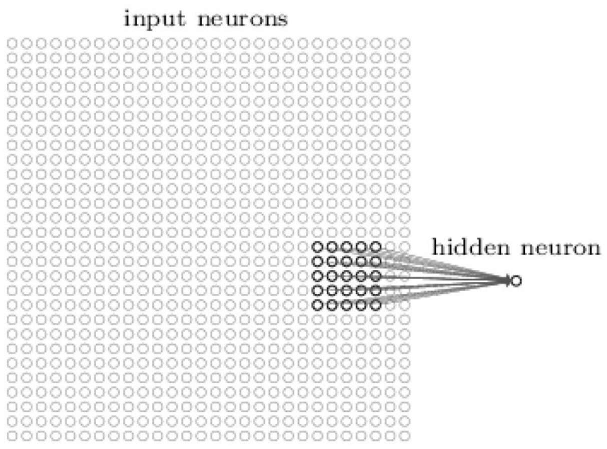

Figure 1: Implementation of a 5x5 receptive field in a CNN

layer are only connected to a small region of the layer before it, called a receptive field. Figure 1, for instance, shows a 5x5 receptive field from the input neurons to the first hidden layer, with a 28x28 input. Distinct types of layers, both locally and completely connected, are stacked to form the CNN architecture.

LOCAL CONNECTIVITY Following the concept of receptive field, CNNs exploit spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. The architecture thus ensures that every **filter** (i. e. weight patch) learnt produces the strongest response to a spatially local input pattern. Stacking many of such layers leads to non-linear filters that become increasingly "global" (i. e. responsive to a larger region of input space). This allows the network to first create good representations of small parts of the input, then assemble representations of larger areas from them.

SHARED WEIGHTS In CNNs, each filter is replicated across the entire visual field. These replicated units share the same parameterization (weight vector and bias) and form a feature map. This means that all the neurons in a given convolutional layer detect exactly the same features. Replicating units in this way allows for features to be detected regardless of their position in the visual field, thus constituting the property of translation invariance.

Together, these properties allow convolutional neural networks to achieve better generalization performances in vision problems. Moreover, th weight sharing helps by dramatically reducing the number of free parameters being learnt, thus lowering the memory requirements for running the network. Decreasing the memory footprint allows the training of larger, more powerful networks.

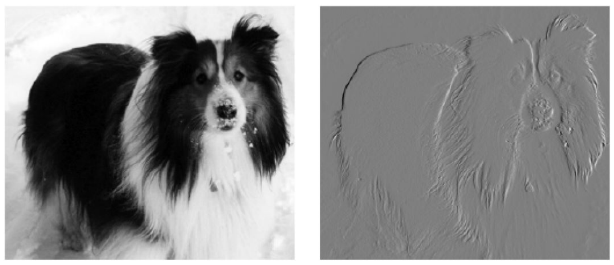

Figure 2: Application of a filter for edge detection (Photo credit: Paula Goodfellow)

The output of a convolutional layer, i. e. the the matrix formed by sliding a given filter over the input and and computing the dot product, is called **feature map**. It is evident that different filters will produce different feature maps for the same input. Figure 2 shows the result of the application of a filter for edge detection to a given image. The image on the right was formed by taking each pixel in the original image and subtracting the value of its neighboring pixel on the left. This shows the strength of all of the vertically oriented edges in the input image, which can be a useful operation for object detection.

In order to recognize an image, we'll need more than one filter. For this reason, the output of a full convolutional layer will be a set of feature maps with the following structure:

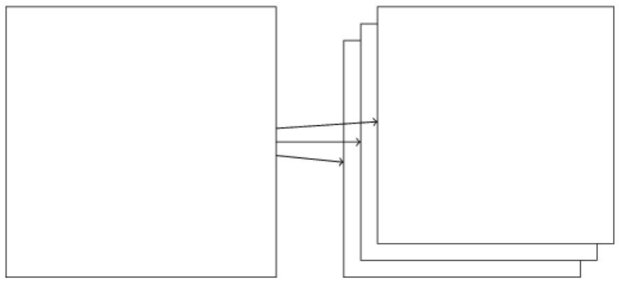

Figure 3: Application of three filters to a given input producing three different feature maps

The amount by which the filter shifts is called **stride** and it controls how the filter convolves around the input volume.

In order to have an output of the same size of the input, it is necessary to add some additional pixels to the input image, this is called **padding**. For instance, a zero padding like the one in figure 4 pads the input volume with zeros all around the border.

In general, the formula for calculating the output size for any given convolutional layer is:

$$O = \frac{W - K + 2P}{S} + 1$$

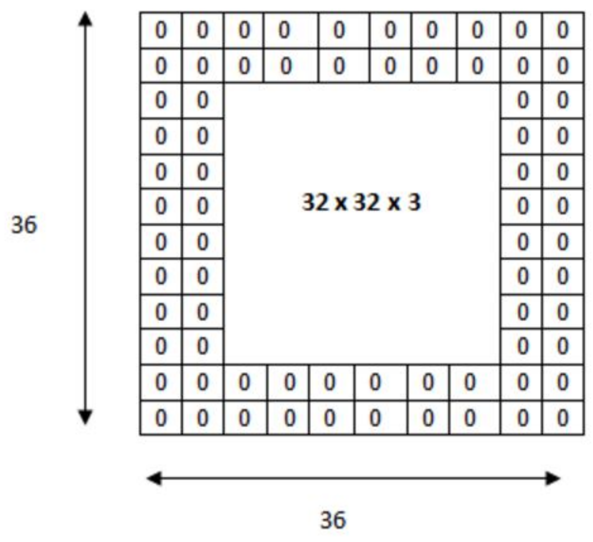

Figure 4: Zero padding of two applied to a 32x32x3 picture

where O is the output height/length, W is the input height/length, K is the filter size, P is the padding, and S is the stride.

3 TENSORFLOW BASICS

REFERENCES

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press (cit. on p. 2).

Knuth, D. E. (1974). "Computer Programming as an Art." In: *Communications of the ACM* 17.12, pp. 667–673 (cit. on p. iii).