PiCNN

C++ Convolutional Neural Network Library for Raspberry Pi

1 Introduction

Convolutional Neural Networks (CNNs) have been one of the best performing sets of **computer vision algorithms** since their discovery and are mainly used for image classification. In essence, they build on traditional neural networks and introduce features that are **inspired by biological processes**. Benefits include a lower number of learnable parameters compared to traditional neural networks, shift in-variance, and automatic learning of distinguishing features. A comparison of the structure of traditional neural networks and CNNs is shown in *Figure 1* below.

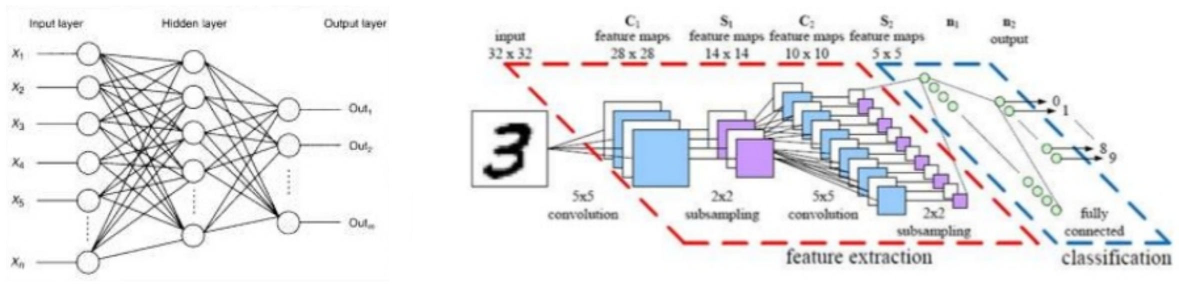

Figure 1: Comparison of Network Structures

CNNs are **trained using supervised learning**. A dataset consisting of training images in combination with the correct training labels is fed through the network and the internal weights updated to match the input-output representation. The goal is to minimize an error function - i.e. a measure of how many classifications the network performed incorrectly. The weight update is performed by a technique known as *Backpropagation* in combination with *Gradient Descent*.

2 PiCNN

2.1 What is PiCNN?

PiCNN, as the name implies, is a convolutional neural network implementation for the Raspberry Pi. It is a **fast, lightweight, single-header, and easy-to-use convolutional neural network library written in C++ specifically designed for the Raspberry Pi 3** and its limited computational capabilities compared to conventional desktop computers. It enables practical machine learning in Raspberry Pi programming projects, without the hassle, size, and computational requirements of installing and running larger machine learning frameworks.

2.2 Why PiCNN?

Traditional machine learning frameworks, such as TensorFlow or Theano, are very large and resource-intensive environments that are tedious to install - even on regular desktop machines. These are unfeasible to run on single-board computers such as the Raspberry Pi, due to the limited resources available. Other, smaller C++ neural network libraries exist, however after extensive research and testing, these also fail to deliver a quick and easy way to implement CNNs on single-board computers.

Therefore, **PiCNN was developed to provide a light-weight but powerful alternative** to the above mentioned packages. PiCNN can be included into any C++ project using a single include statement. Various neural network architectures, with flexible parameter settings, can be achieved easily using PiCNN.

2.3 Performance and Future Improvements

A standard measure of performance for CNNs is the MNIST dataset, which consists of roughly 60,000 images of handwritten digits between 0 and 9. PiCNN in just a few lines of code, quick compilation and execution time ($<$ 2 minutes), achieved an **accuracy of over 90%**.

PiCNN in its current form is fully functional, however a large future improvement can be made. This is to enable **parallel training of clusters of Raspberry Pis across a network**, so that each Raspberry Pi only performs a simple neural network operation, but in combination they outperform the training time and accuracy compared to running on a single desktop machine.

3 Conclusions

After having developed and tested PiCNN, the following conclusions can be made:

- PiCNN provides a strong, light-weight alternative to large machine learning frameworks for single-board computers.
- Compilation, implementation, and execution time of PiCNN outperforms most frameworks available.
- Training accuracy on conventional datasets is on par with other frameworks.
- A large future improvement is to implement parallel training of clusters of Raspberry Pis running PiCNN.

Philip Salmony | University of Cambridge | pms67@cam.ac.uk