Iterative Particle Swarm Optimisation – Hyperparameter Tuning

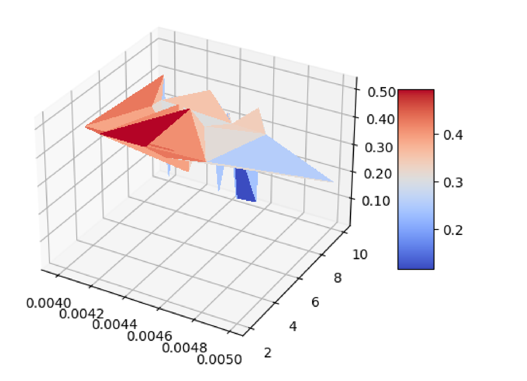
Grey Churchill

Simon Parsons (Lincoln Centre for Autonomous Systems / UoL), Michal Mackiewicz (Colour and Imaging Lab / UEA)

Introduction

The overarching project for this work investigates vision techniques using machine learning for the context of green-on-green spraying. So far this has involved ensemble networks and multi-spectral imaging, however a common theme is the need to train neural networks.

Summary



This work employs Particle Swarm Optimisation to automatically tune hyperparameters such as learning rates and weight decay. This is achieved using an estimate of a gaussian distribution of each weight, combined with gradient descent and iterative resampling.

Problem

The outcome of training a neural network is dependent on the hyperparameters used. The ideal set varies with each dataset; it is usually estimated via trial and error with expert knowledge. The surrounding PhD work involves training multiple networks on multiple datasets, making automation very valuable.

Particle Swarm Optimisation (PSO)

Particle swarms are a common method used to estimate probability distributions. A common example is in Monte Carlo Localization[1] for mobile robots, where each particle represents a potential position. As more information is gathered about the robot's locale, particles are evaluated on the likelihood of being the same position as the robot. Over time, the distribution reflects a more and more confident estimate of the robot's position, to the point it can be assumed accurate. Sampling methods such as this are favourable compared to grid-based methods, because of the lower memory and processing costs. They also scale well, and hence can be run in parallel simply.

Sub-Heading/References

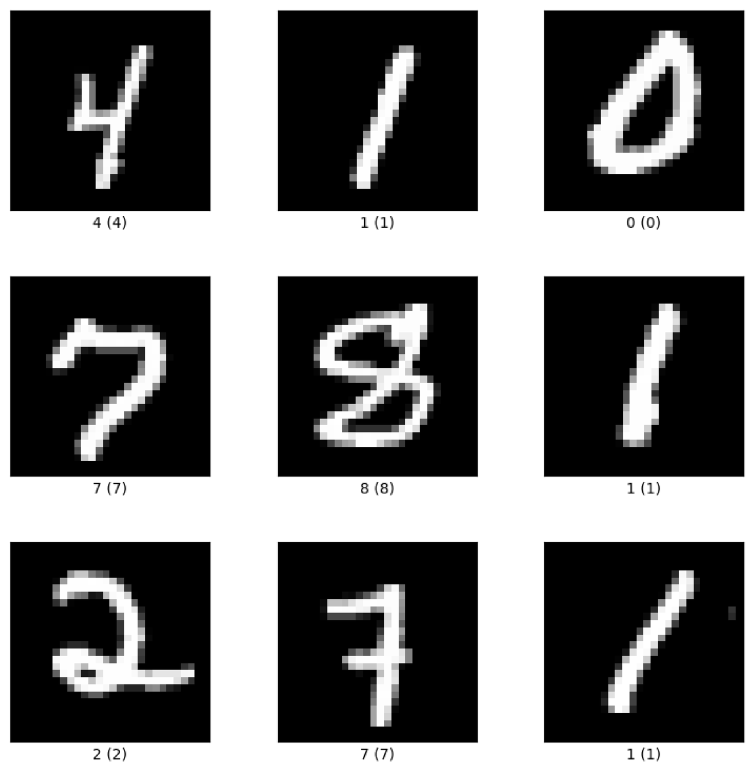
[1] Dellaert *et al.*, ICRA **2**, 1322 (1999).

[2] Deng., IEEE Signal Proc. Magazine ***29***, 141 (2012).

Method

When applying PSO to neural networks, thinking of each hyperparameter to be optimised as a dimension simplifies the explanation. First, arbitrary boundaries are set for each parameter. These are used to constrain the particles, as well as initialise them in the first iteration. Each particle is assigned a value in each dimension based on a uniform distribution between the boundaries, and the network is trained using these parameters. The final training loss value is used as the cost function to be optimised by the particle swarm.

After this first iteration, and at regular intervals onward, the swarm is reinitialised. This is done by approximating the distribution of each parameter as a gaussian, with the mean set to the value found by the best performing particle, and a standard deviation (arbitrarily) of *0.05*. A gaussian distribution is used because it still allows for outliers, while most particles remain near the current best position. After runs between resamples, the gradient is estimated using the particle's current and last position, and gradient descent is applied with a decreasing rate. Each particle only changes 3 randomly chosen dimensions, else too many variables change and the gradient descent fails to improve the model.



For the purposes of this work, the MNIST dataset[2] was used with as well as a very simple classifier. This was done to reduce the time taken to train a network, which could also be done by using a reflective sample of a larger dataset in the case of a much larger network.

Results and Conclusions

The initial results are promising, with the network performance gradually improving as expected during training and an accuracy of *95.91%* being achieved on the MNIST test set. One difficulty with examining the output is the dimensionality; while testing a network trained with the generated hyperparameters shows it has worked, the distribution of particles still needs to be checked to validate boundaries and other parameters such as particle count, standard deviation and the rate used for gradient descent.

Future work includes applying this methodology to a reflective subset of an agricultural dataset, as well as visualising the multidimensional output of hyperparameters as they change over time.

Sub-Heading/Collaborators

Grey Churchill <17642848@students.lincoln.ac.uk>*

* AgriFoRwArdS CDT, Robotics Department, University of Lincoln