Danmarks Tekniske Universitet

Katrine Bay, s183910
Lukas Leindals, s183920
Matilde de Place, s183960
Sunniva Olsrud Punsvik, s183924

02463 - Aktiv machine learning og agency F20

Project 1 - Optimisation of ANN for classifying wine by cultivar

5th March 2020
Danmarks Tekniske Universitet

Introduction

It can be a challenging task to find the input or parameter values of a model that results in the optimal output either by finding the minimum or maximum of a certain cost function. This task is an optimisation problem and it can be both time consuming and rather difficult. Bayesian optimisation has proven to be able to solve this kind of problem in a very effective way, and has proven to cause better overall performance of various machine learning algorithms and to reduce the time required for optimisation.

This project will focus on improving the classification performance of an artificial neural network (ANN) by optimising different parameters of the network with Bayesian optimisation (BO) where Gaussian processes (GPs) are applied. The ANN is trained and tested on subsets of the CIFAR10 data set loaded directly into pyTorch with the package torchvision [1].

Description of data

As mentioned, this report makes use of the data set CIFAR10 which consists of 60,000 RGB images (continuous) of size 32x32 pixels, and their corresponding class label (categoriacal). The images are divided into 10 classes of 6,000 images: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. Furthermore, the CIFAR10 are divided into 5 training batches and one test batch of 10,000 images. The CIFAR10 classification problem, in this report, uses a Neural Network in order to classify the images as one of the ten classes.

Method & Analysis

The model implemented in this project to apply Bayesian optimisation on is a two-layered convolutional neural network with ReLU and Sigmoid as the activation functions, as well as the two optimisers which are to be tuned; ADAM and Stochastic Gradient Descent (SGD).

Bayesian optimisation is a powerful tool for optimising model configuration - to put it simply, Bayesian optimisation is a model-based hyperparameter optimisation which builds a probability model of an objective function and use it to select the most promising hyperparameters to evaluate in the true objective function. For the ANN the parameters which are tuned: *learning rate*, choice between the two *optimisers* ADAM and SGD, and lastly the choice between the two *activation functions* ReLU or Sigmoid (see table 2). Building a probabilistic model, one needs an acquisition function to sample from the objective function and in this project it was chosen to be *Expected Improvement* (EI), this method tries to quantify the amount of improvement for each $x \in \mathcal{X}$ with respect to the current best sample.

To evaluate the ANN; BO was run five times with different exploration weights (see table 1), which optimises the objective function which generates the accuracy for the ANN where the loss is measured in cross entropy. The learning rate ranges `np.arange(0.0001, 0.011, 0.0001)`.

Results

Exploration weight	0.01	0.3	0.8	1	1.5
Accuracy	53.17	49.41	54.16	53.17	53.17

Table 1: Accuray of the five BOs with different explorations weights

Exploration weight	Optimizer	Activation function	Learning rate	Accuracy
0.8	Adam	ReLu	0.0009	54.16
1 and 1.5	Adam	ReLu	0.0009	53.17

Table 2: Hyperparameters of 2 models with highest accuracy

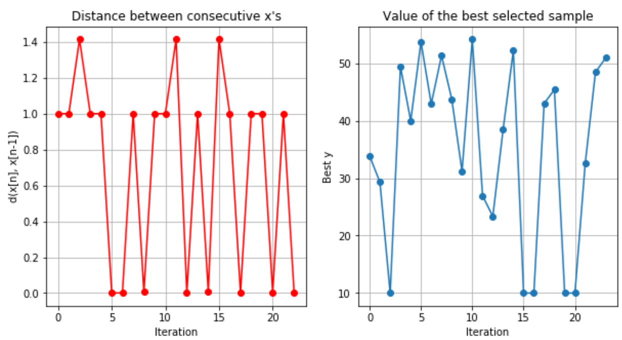

Figure 1: BO with accuracy 54.16, `exploration weight = 0.8`, `Activation function = ReLu`, `Learning rate = 0.0009`

Discussion & Conclusion (learning outcome)

This paper examines BO with five different values of exploration weights (i.e five choices of how much the acquisition function is to explore); 0.01, 0.3, 0.8, 1 and 1.5. In table 1, the `max(objective function)` (i.e. accuracy of neural network) are shown for each exploration weight respectively. Table 2 shows the parameters that maximises the objective function, along with `max(objective function)` and the exploration weight. Figure 1 contains two figures. The first visualises the distance between to consecutive **x**'s, i.e. the parameters, chosen by the acquisition function. The second figure visualises the values of the objective function, at the point chosen by the acquisition function.

Table 1 indicates that the accuracy does not depend hugely on the exploration weight, since all five accuracies do not vary a lot. At least not with the range of numbers examined in this report. However, table 2 shows, that the objective function is maximised by the parameters: `Optimizer = Adam`, `Activation function = ReLu` and `Learning rate = 0.0009`, by the

the four best accuracies. Even though the distances in 1 (the first figure), often are equal to one, indicating that the acquisition function often alternates between either the two different activation functions, or the two different optimisers. Figure 1 shows that evaluations of the objective functions varies a lot all the way through. Even though the tables gives some usefull results, the accuracies are all relatively low, since it informs, that the neural network only classifies 54.16% of 1000 test images correctly. Therefor, it is relevant to optimise other parameters. Alternatives could be the drop out rate or a regularisation parameter. Introducing a regularisation parameter, and thereby penalise large weights, could possibly improve the accuracy by penalising weights which cause some features (pixels) to be favoured by the neural network and cause overfitting.

By optimising the ANN using BO and Gaussian Processes we have come to a better understanding of these techniques, their efficiency and how to implement them.

References

[1] Mathieu Fortin. *Nonlinear regression*. URL: http://www.nwfps.eu/wp-content/uploads/2012/07/Fortin-NonlinearModels.pdf.

Appendix

Link to code in GitHub

Github link