MLP Coursework 1: Activation Functions

s1784849

Abstract

The abstract should be 100–200 words long, providing a concise summary of the contents of your report.

1. Introduction

As neural networks have become become increasingly popular over the last decade (reference: https://arxiv.org/pdf/1206.5533.pdf) it has become important to investigate and understand the effects popular activation functions have on the training process, and how different activation functions can be used to achieve state of the art level performance on well understood research problems. Furthermore, the emergence of deep learning (Hinton et al., 2006; Bengio et al., 2007; Ranzato et al., 2007) has provided ample motivation for investigating the effects of employing many hidden layers within neural network architectures.

This report provides investigations into three variable aspects of neural networks that effect the potential accuracy and error achieved by these models. Initially, before any investigations are presented, an introduction to the set of activation functions that are employed by neural networks utilised within this report is provided. An experimental comparison of these activation functions is given, along with a sensitivity analysis that provides tuning for the hyper parameter usually labelled as the step size (or learning rate)[insert reference here]. In addition, an investigation into varying the depth of a neural network is presented and follow up conclusions about the effects this has on the potential accuracy attained by the networks is proposed. A final inquire into varying the network weight initialisation method is also touched upon.

The neural networks used for the experiments within this report are trained on the MNIST dataset[reference goes here]. The training set is made up of 50,000 examples and the test set (which the validation examples are drawn from) is of length 10,000.

The fundamental research questions this report aims to address are outlined below.

- Two hidden layer network: what is an optimal setting for the hyper parameter known as the step size (or learning rate) on the given activation functions?
- Two hidden layer network: which activation function provides the fastest rate of convergence towards a minimum error and maximum accuracy?
- Two hidden layer network: which activation functions provides the highest ceiling with regards to potential accuracy that can be acquired?
- N-hidden layer network: how does varying the number of hidden layers within our neural network architecture affect the overall model accuracy?
- N-hidden layer network: how can modifying the weight initialisation for a deep neural network affect the training process?

2. Activation functions

The activation functions utilised within this report are leaky rectified linear units, exponential linear units (https://arxiv.org/pdf/1511.07289.pdf) and scaled exponential linear units (https://arxiv.org/pdf/1706.02515.pdf). Comparisons against one another, in addition to baseline rectified linear units and sigmoid units will be presented in the next section. This section provides an introduction the aforementioned activation functions.

2.1. Sigmoid

Within this report a standard sigmoid function is used to provide a simple baseline to compare other activation functions to. The sigmoid activation function is defined as the following[insert reference here]:

INSERT MATH HERE f(x) = 1 / 1+e$^{(} - x)$

With gradient fâĂŹ(x) = x * (1 - x)

2.2. Rectified Linear Unit

$$\text{relu}(x) = \max(0, x), \tag{1}$$

which has the gradient:

$$\frac{d}{dx}\text{relu}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0. \end{cases} \tag{2}$$

2.3. Leaky Rectified Linear Unit

$$\text{relu}(x) = \max(0, x), \tag{3}$$

which has the gradient:

$$\frac{d}{dx}\text{relu}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0. \end{cases} \tag{4}$$

2.4. Exponential Linear Unit

$$\text{relu}(x) = \max(0, x), \tag{5}$$

which has the gradient:

$$\frac{d}{dx}\text{relu}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0. \end{cases} \tag{6}$$

2.5. Scaled Exponential Linear Unit

$$\text{relu}(x) = \max(0, x), \tag{7}$$

which has the gradient:

$$\frac{d}{dx}\text{relu}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0. \end{cases} \tag{8}$$

3. Experimental comparison of activation functions

The neural network model used within the initial experiment that compares activations functions was trained using 2 hidden layers, each containing 100 units. A soft-max layer was applied to the output which contained 10 units in order to classify digits from the MNIST dataset from 0 to 9. A batch size of 50 and stats interval[footnote: the interval at which statistics are recorded across the experiment, measured in epochs.] of 1 was used across all experiments.

Throughout the experiments the sigmoid and ReLU activation functions were used as baselines to compare with the performance of the other activation functions. However, it is shown in figure X.X that any ReLU variant function[FOOTNOTE: ReLU variant functions reference ReLU, LReLU, ELU and SELU in this report.] consistently outperforms a standard sigmoid function[FOOTNOTE: By standard sigmoid, this report is referring to the function described within the second section, as a sigmoid with a tuneable exponent may perform better.]. For this reason, the sigmoid baseline is only shown in figures and tables presented within the appendices.

[FIGURE X.X] <- 4x4 gridplot

It is shown that although ReLU variant functions consistently outperform a sigmoid baseline, there are occurrences of overfitting the test set when measuring error on the validation set. This is conveyed in figure X.X within the validation error column. The sigmoid line that is plotted (marked in blue) crosses over the ReLU variant function lines, demonstrating that the ReLU variant functions are more sensitive to overtraining. This suggests that employing a technique such as early stoping is particularly important for these activation functions, for this reason measurements in error and accuracy will be taken between the range of epoch 15 and 20.

In order to find an appropriate value for the step size, which is a tuneable hyperparameter, a sensitivity analysis was performed. The values 0.05, 0.1, 0.2 and 0.3 were used evaluate the performance of the network. In order to gauge a more reliable value for the step size, the experiments were repeated a total of five times[FOOTNOTE: Ideally, experiments of this nature should be performed more than five times in order find a more reliable value for any particular hyperparameter, in this instance the experiments were repeated five times due to time constraining factors.] and the mean value for error and accuracy was taken across all of these experiments for each epoch. The following seeds were used for each experiment:

1. 6102016 2. 1237878 3. 2673679 4. 8978283 5. 5627351

A value of 0.2 was chosen for the step size because this value provided the highest ceiling with regards to the potential accuracy that could be reached, see figure X.Y and appendix X.Z.

[FIGURE X.Y] <- sensitivity$_a$*nalysis*$_v$*alidation*$_a$*ccuracy*

Figure X.A shows the training and validation error and and figure X.B presents the validation accuracy of all four ReLU variant functions

[FIGURE X.A] <- 02$_r$*elu*$_v$*ariants*$_e$*rror*

[FIGURE X.B] <-02$_r$*elu*$_v$*ariants*$_a$*ccuracy*

4. Deep neural network experiments

This section should report on your experiments on deeper networks for MNIST. The two sets of experiments are to explore the impact of the depth of the network (number of hidden layers), and a comparison of different approaches to weight initialisation.

5. Conclusions

ashjdhajshdjahskjd