395 Machine Learning
— Assignment 2 —

Group 11
Malon AZRIA, Alexandre CODACCIONI, Benjamin MAI, Laura HAGEGE.
ma12917@ic.ac.uk, afc17@ic.ac.uk, bm2617@ic.ac.uk, lmh1417@ic.ac.uk
CBC helper: Pingchaun MA
Course: CO395, Imperial College London

28th February, 2018

Introduction

The aim of this coursework is to implement a Fully-Connected Neural Network from scratch and train it in order to identify the best parameters according to emition recognition. This report will present how the backpropagation algorithm has been implemented and the analysis of a such algorithm One of the goal of backpropagation is to optimize the weights and then the neural network would be able to learn how correctly map arbitrary inputs to outpus.

To do so we implemented first all the necessary keystones function to create the Network such as:
- Linear and ReLu Layers
- Dropout Layers
- SoftMax Classifiers

Finally we implemented hyper-parameter optimisation in order to train our Network.
In this report we will explain and discuss choices we made on our function implementation and the results we got from the training.

Linear and ReLu Layers implementation (forward/ backward pass)

Forward and backward layers' pass depend on the activation function.

Linear function

- Forward pass :
  For a given set of data X, weight W and bias b computing the forward pass for the linear function consist in calculating the conresponding $net = \sum_{k=0}^{N} w_i * x_i + b$. Then the activation function σ is just as $\sigma\ (net) = net$.

  The idea of the implementation is to apply this transformationt to all data included in the matrix input, and repeat the process for each transformed output.

  Our code has been created to compute every kind of data which includes shape differencies.

- Backward pass :

Once the transformations have been made, we can calculate the error for each output neuron. We can use the squarred error function and sum the erros:

$E = \sum_{k=0}^{N} 1/2 * (target - output)^2$.

One of the main idea behind the backpropagation algorithm is to reduce the error rate according to the results expected. We need then to send the output back into the neural network to update parameters such as weight and biais.

Our algorithm calculates dW and db according to dout, which the first neural network output. We then need to send the "re-estimated" value in other hidden layer by calculating dX.

ReLu activation

- Forward pass :

  Still in order to achieve a re-definition of hyper-parameters, we can apply to our input Data for the linear forward pass, an activation function. In this algorithm, we use $ReLU(X) = max(0, X)$.

  ReLU functions are particurlarly efficient for Non-linear complex functional mappings between the inputs and response variable.

  If we do not apply a Activation function then the output signal would simply be a simple linear function.A linear function is just a polynomial of one degree. Now, a linear equation is easy to solve but they are limited in their complexity and have less power to learn complex functional mappings from data.A Neural Network without Activation function would simply be a Linear regression Model.

- Backward pass :

Dropout implepentation

Softmax classifier

Fully-Connected Neural NetworkCreation

Hyper-parameter optimisation

Conclusion