REPORT

Name: Arun Kumar S

S.R.No: 15656

In Software1.0 , the task is very trivial . I have just written the divisibility conditions for 3 , 5 and both. Since the problem is very simple , we are able to design the rules and hence are able to solve the problem perfectly with 1.0 accuracy using simple modular arithmetic .

In Software2.0 , we want the machine to learn the algorithm of divisibility tests intrinsically . The training data is the numbers from 101 to 1000 . The representation of the data is very important for the learning of algorithms by the model . Expressing the numbers in binary will makes it easier for the algorithm to learn the divisibility since the binary representation gives better information about the divisibility of number . So all the data will be represented in binary . Since numbers used for training are <= 1000 , number of digits in binary will be less than 10 .

In the model , I have 3 hidden layers and one output layer . I tried with different number of layers . With more than 3 hidden layers , I found overfitting . Training data is learnt with 1.0 accuracy with number of layers >=2 for epochs>50 . But with more than 3 hidden layers, the accuracy on the validation set was less around 0.69 for 4 layers with >50 epochs . Even after trying with different number of epochs I found that there was overfitting . With 3 hidden layers it was working well but accuracy for the validation set was not the best . With only one hidden layer , accuracy was good around 0.8 for the validation set only when number of hidden units was large (>300). So the capacity of the model had to be increased . Two hidden layers gave good accuracy on the validation set around 0.95 for around 80-100 epochs on the validation set . So I decided to keep 2 hidden layers .

The first layer has 250 neurons and second layer has 125 neurons . The number of units is significantly high implying that the required capacity of the model is significantly high . The number of units was chosen by random sampling . With decrease of number of neurons in consecutive layers by 0.5 (approx.), the accuracy was always seemed to improve . So I have chosen 250 followed by 125 units .

The activation function used in all the layers except the last output layer is the relu activation function . This is most commonly used in most of the ML algorithms and I have chosen it for the same reason .

Now to decide the number of epochs to run on so that there is a good trade off between bias and variance . With different values of epochs , the model was run and both training and validation error was plotted against the number of epochs . So the epoch number was chosen such that both the training errors and validation errors are less .

The output layer has softmax activation . Softmax is the activation function commonly used when dealing with multiclass classifications . Even in my model , the output is a vector with four values representing the probabilities of being "fizzbuzz" , "fizz" , "buzz" and the number itself, i.e divisible by both 3 and 5 , divisible by 3 , divisible by 5 , not divisible by any of 3 or 5 . So the model written is a solving a typical multiclass classification problem of classifying the number as divisible by 3 or 5 or both or none .