Decision Making for Computation Offloading in Mobile Cloud Computing

Anil Kumar, Aniruddha Shrikhande and Anand Deshmukh

June 29, 2018

Group No - 15

**Supervisor Name - Prof. Shashikala Tapaswi,
Dr. Neetesh Kumar**

1 Problem Description

We aim to improve the performance of the currently available decision making methods for Computation offloading in Mobile Cloud Computing.

These days, mobile devices are used to perform heavy tasks. This means if the smartphone tries to deliver the output using its local resources, it will deteriorate its performance. Its battery life, processing capabilities and storage space will degrade. On the other hand, if we let all the tasks to be offloaded on to the cloud, it will cause other significant problems such as high time response and more data usage for communicating with the servers on the cloud. In short, a user will not be satisfied with his experience.

Hence, a need for a dynamic and efficient decision maker arises which will try to predict when to offload a resource intensive task on to the cloud as accurately as possible.

2 Methodology

2.1 Perceptron

It refers to a learning algorithm which works on a set of inputs and classifies it into binary results. The input is represented in the form of a vector of real numbers. It is a linear classifier and works on a linear combination of the input parameters. Shortcomings of this learning algorithm includes an instance when the decision boundary is not linear. In this case, the algorithm doesnt terminate since the classification of learning points never happen.

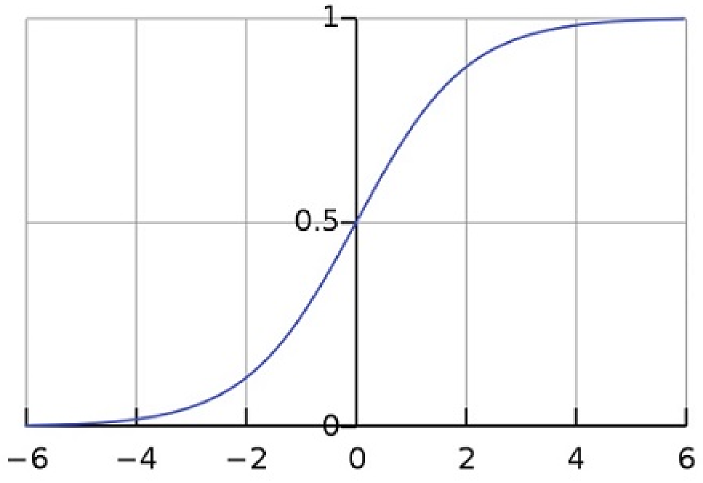
2.2 MLP

To overcome this limitation, we use a multilayer perceptron architecture which can capture the non-linear decision boundaries in the training data. It consists of multiple layers of perceptron where the outputs of the perceptron of one layers acts as an input to the next. The weights of the MLP network can be learned iteratively using gradient descent and backpropagation.

2.3 Sigmoid Function

A typical perceptron produces a binary output by application of the Signum function on the linear combination of the inputs. This limits the flexibility of the classifier. Therefore, we choose to employ the sigmoid function which outputs the probabilities instead of Boolean values.

$$f(x) = \frac{1}{1 + e^{-x}}$$



3 Experimental Setup (Dataset, Software, Hardware Used)

In our approach, we are developing an application in which we will implement our decision maker.

We are generating a data set which will contain the following key parameters:

- Mobile CPU Usage
- Battery Percentage

- Latency
- Network Strength/ Bandwidth
- Problem Size

In order to generate the dataset, we will develop an application which we will first run on our smartphone using its local resources and note down the values of the above key parameters including output response time. Next, we will offload the same task to the cloud server and again note down the values of the key parameters. Then, we will compare the output response time of both the scenarios and select the one which has smaller value. By this we will finally achieve our base data set.

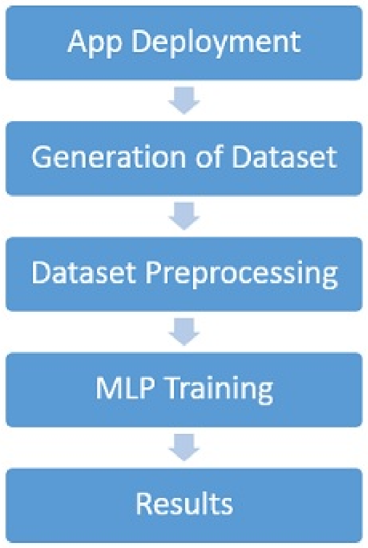
After having the dataset generated, it will be split into three parts namely training, validation and test sets. The MLP will be trained on the training set to learn the parameters. The hyper-parameters of the MLP such as number of layers and the number of units in each layer will be decided using the validation set. The trained model will be evaluated on the test set to give the final accuracy.

We are using cloud services to offload the data on a high-end system to avoid the bottleneck of traffic at server side.

3.1 Software and Libraries Used

- Android Studio: For App development and deployment
- Android CPU Profiler: For data gathering and dataset generation
- Tensorflow: For MLP implementation
- NumPy: Maths operation library
- Pandas: For data analysis

3.2 Flowchart


App Deployment
Generation of Dataset
Dataset Preprocessing
MLP Training
Results