Scalable Machine Learning and Deep Learning- Project Report

1. INTRODUCTION

Breast cancer is nowadays very common among males and females. It is a type of cancer which is formed at the cells located at the breast. After skin cancer it is one of the most commonly diagnosed diseases among women. The commonly found symptoms are lump or thickening that feels different from the surrounding tissue, change in the size, shape or appearance of a breast. Thanks to the advances in imaging technology, early detection and treatment made possible to increase the survival rates drastically. It is very important to identify the presence of cancer based on the clinical tests and for this reason we decided to to work on breast cancer database.

2. WORK DONE

In the project, in order to predict the presence of breast cancer we have implemented a Recurrent Neural Network which uses its memory to process sequences of inputs. Further, LSTM model is trained to classify benign and malignant tumor.

3. DATASET

The dataset has been extracted from UCI Machine Learning Repository, the link for which is attached below: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+The dataset consists of following features (32 attributes):

- radius
- texture
- perimeter
- area
- smoothness
- compactness
- concavity
- concave points
- symmetry and
- fractal dimension

All feature values are recorded with four significant digits and there are no missing attribute values. The class distribution is: 357 benign, 212 malignant (in total: 569). The mean, standard error (SE), and "worst" or largest (mean of the three largest values) of these features were computed for each image,resulting in 30 features. For instance, fields 3 to 12 correspond to Mean Radius, fields 13 to 22 correspond to the standard error (SE) and, field 23 to 32 correspond to the "worst" measurements.

4. METHOD

- Step 1: Import all the libraries that we need in order to run the code. Then we load the data (csv file) from the previously stated repository into the program.
- Step 2: Next, we separate the data into label column and feature columns. In the dataset, there are 32 columns out of which the 1st column is the id of the measurements and the 2nd column is diagnosis (benign or malignant). As the 2nd column gives information about the output we want to predict, presence of cancer, we set it as the label column for our code.
- Step 3: Once the data is separated as label and features, then it is further divided into training (80%) and test sets (20%).
- Step 4: The constants used during the training and testing of the model are initialized:
  1. epochs: the number of iterations to run the data set through the model.
  2. n_classes: the number of classes for classification. In this case we are assigning 0 if it is benign and 1 if it is malignant.Thus,the number of classes is 1.
  3. n_units: the size of the hidden state.
  4. n_features: represents the number of features in the dataset.
  5. batch_size: refers to the size of each batch of data you are going to feed into the model. Back-propagation will be done once per batch.
  6. learning rate: magnitude of our weight updates so that the loss function is minimized.
- Step 5: Definition of the placeholder for the batch of data which is fed into the model
- Step 6: Building of the RNN Model using LSTM cells for an improved memory performance of the network
- Step 7: Lastly, training and evaluation of the model.

5. RESULTS

We calculate F1 Score which is the weighted average of Precision and Recall. Recall is the ratio of correctly predicted positive observations to all positive observations. Then, the Accuracy of the model and precision is also calculated. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Our results are the following:

Epoch 500 completed out of 500 loss: 6.467866743449122

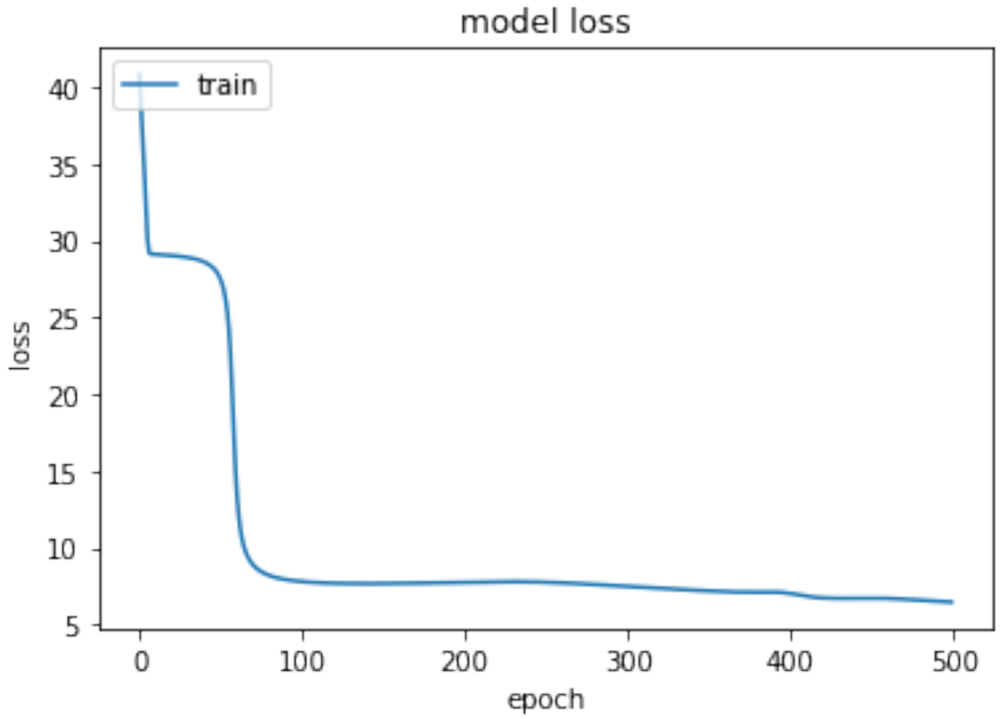
F1 Score: 0.8420151626314503

Accuracy Score: 0.8508771929824561

Recall: 0.8974358974358975

Precision: 0.7291666666666666

Confusion Matrix: $\begin{bmatrix} 62 & 13 \\ 4 & 35 \end{bmatrix}$


model loss
train
loss
epoch
40
35
30
25
20
15
10
5
0
100
200
300
400
500