Machine Learning Project Report

By
Surya Alla

PROJECT DESCRIPTION

Goal is to work on Malaria Cell Images Dataset (This dataset is taken from the official NIH Website and Kaggle website). The aim of collection of datasets was to reduce the burden for macroscopics in resource-constrained regions and improve diagnostic accuracy using AI based algorithm to detect and segment the red blood cells.

Background

The latest WHO report showed that the number of malaria cases climbed to 219 million last year, two million higher than last year. The global efforts to fight malaria have hit a plateau and most significant underlying reason is international funding has declined. Malaria, which is spread to people through the bites of infected female mosquitoes, occurs in 91 countries but about 90% of the cases and deaths are in sub-Saharan Africa. The disease killed 4,35,000 people last year, the majority of them being children under five in Africa. We want to obtain the highest level of detection of case, resultant we must also consider making the model as small and computationally efficient as possible and also able to be deployed to edge and Internet of Things devices.AI backed technology has revolutionized the malaria detection in some regions of Africa and the future impact of such work can be revolutionary.

Introduction

We have seen lots of algorithm came for classification and this paper we tried to perform different classification technique and evaluated how different classifier works with image classification. Image classification is one of the hot topics and it can solve lots of problem statement in real life. We have used these classifiers to classify the cell image and train the model to classify the infected and not infected cell image. We have used three different model to train over same dataset and recorded how these models behave and which model we should choose in this scenario. We have used Naïve Bayes, Random Forest classifier and Convolutional neural network. During the process we have learned a lot about the complex mathematics going behind each classifier. Later in the paper you find the detail description of each model.

Naïve Bayes

It is a conditional probability model, when given a problem instance to be classified (vector representation of features), it assigns to this instance probabilities. This model is completely based on Bayes theorem, which states:

$P(C_k \mid x) = P(C_K)\ P(x \mid C_K) / P(x)$

Approach: -

1. I used Kaggle kernel to easily access the Malaria Cell dataset and also to explore the dataset of the malaria cell images first and code the same in the kernel.
2. There were two folders (Parasitized and Uninfected), which had around 13700 images in each folder. First, I iterated all the images in each folder and assigned a label whether it belonged to positive (Parasitized, labeled as 1) or negative (Uninfected, labeled as 0).
3. When iterating the images, the images are converted to multidimensional array with RGB values and resized the image to 64 * 64 width and height for less memory intensive computation. The array is converted to NumPy array and stored in X vector and Y is list of all labels assigned to that pictures in 1d array.
4. After preprocessing the data, the data is now divided into 80 percent training and 20 percent testing.

5. Before putting the train data into the model, I need to first flatten the multidimensional X array to be compatible with the labels 1d array.
6. With the help of scikit learn's Gaussian Naïve Bayes model, I was able to train the model with and without priors and calculating their respective accuracies.
7. The priors are as follows: - 0.1 for Positive class and 0.9 for Negative class 8. training models with priors gave the highest accuracies of 63.60%

Random Forest Classifier

Random forest is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forest creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.

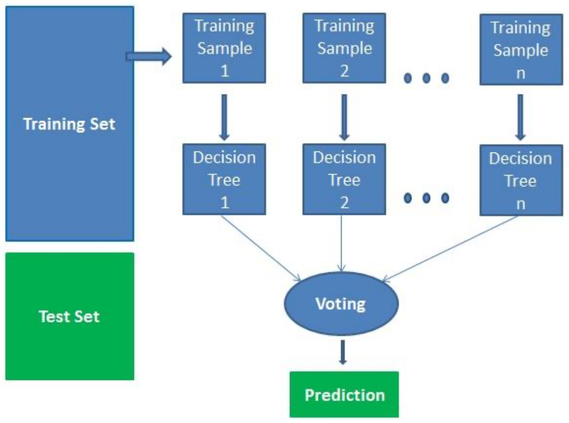
It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. How the algorithm works:

1. Select random samples from a given dataset.
2. Construct a decision tree for each sample and get a prediction result from each decision tree.
3. Perform a vote for each predicted result.
4. Select the prediction result with the most votes as the final prediction.

Approach: -

1. I used Kaggle kernel to access the Malaria Cell dataset. In the dataset, there are two folders (Parasitized and Uninfected), which have around 13700 images in each folder.
2. The images are converted to multidimensional array with RGB values and resized the image to 64 * 64 width and height for less memory intensive computation. The array is converted to NumPy array and stored in X vector and Y is list of all labels assigned to that pictures in 1d array where 1 is assigned to parasitized and 0 is assigned to uninfected.
3. Using sklearn train_test_split, the data is divided into 90 % training data and 10% test data.
4. The array of images is than flatten to a 1d array.

• The training data was used to train the random forest classifier and the accuracy was calculated on the prediction made by the model



Advantages:

- Random forest is considered as a highly accurate and robust method because of the number of decision trees participating in the process.
- It does not suffer from the overfitting problem. The main reason is that it takes the average of all the predictions, which cancels out the biases.

- The algorithm can be used in both classification and regression problems.
- Random forests can also handle missing values. There are two ways to handle these: using median values to replace continuous variables and computing the proximity-weighted average of missing values.
- You can get the relative feature importance, which helps in selecting the most contributing features for the classifier.

Disadvantages:

- Random forest is slow in generating predictions because it has multiple decision trees. Whenever it makes a prediction, all the trees in the forest have to make a prediction for the same given input and then perform voting on it. This whole process is time-consuming.
- The model is difficult to interpret compared to a decision tree, where you can easily make a decision by following the path in the tree.

Accuracy using Random Forest Classifier = 81.79%

Convolutional neural network

Convolutional neural networks have performed really well in recent years in their ability to automatically extract features and learn filters and acted as very good classifier of images. In previous machine learning solutions, features had to be manually programmed in for example, size, color, the morphology of the cells. Utilizing convolutional neural networks (CNN) will greatly speed up prediction time while mirroring (or even exceeding) the accuracy of clinicians.

Classification of cell images is an interesting problem and have great utility. There are already country and medical universities which are leveraging this AI backed technology which can detect diseases (refer related works). I choose Keras sequential model as the starting model, which can work with gray scale images. Initially training model was taking lots of time so I used Maxpolling to reduce the input for my CNN and it have improved my training time at great extent. I got the accuracy of 95% or more on test data-set.

How the algorithm works:

- Create the model and define the number of neurons in input, output and hidden layer.
- Perform forward propagation using initially picked weight on neuron.
- Calculate the final output of forward propagation using activation function. If classifier is producing incorrect result update the weight on the network in back propagation.
- Back propagation adjusts the network weight by bubbling down the error from output to input layer.
- Once weight updated on neurons perform forward propagation again and do all the previous step.
- Stopping criteria: user can keep a threshold of maximum number of iterations. The model stops either if it converges or reach the threshold.

Approach:

- I used Kaggle kernel to access the Malaria Cell dataset. In the dataset, there are two folders (Parasitized and Uninfected), which have around 13799 images in each folder.
- The images are converted to multidimensional array with RGB values and set label for each dataset.
- Using sklearn.model_selection train_test_split the data is divided into 80 % training data and 20% test data.
- Use Keras to create sequential model for CNN and then break image in pixel in form of batch.
- Performed maxpooling to reduce the dimension and size of input. Created two hidden layers, first of 512 neuron and second of 256 neurons using Relu activation function.

- Created an output layer with two output unit using Sigmoid activation function.
- Use adam optimizer for optimization. Set the metrics attribute you want. Train the model with labeled dataset.
- Once model is fit on given datasets, we can retrieve the accuracy using model mastics.

Learning Curve:

This model helps me to learn how to create deep neural network and understand the mathematics beneath the feed forward and back ward learning. I have tried pooling which clear my idea about how to truncate the dataset.

Observation: I have run the model over 50 echos and model error keep improving with each iteration and I have achieved following accuracy.

Accuracy of model: 94.47%

Conclusion

After training our model on different algorithm we observed a difference in accuracy. Accuracy increase once we evaluate model using Naïve Bayes then Random forest and then to convolutional neural network.

Below we mentioned our statistic to support our observation:

Model	Accuracy
Naïve Bayes (With prior)	63.57%
Naïve Bayes (Without Prior)	63.60%
Random Forest	81.97%
CNN	94.47%

REFERENCES

[1] "Nih data set." [Online]. Available: https://ceb.nlm.nih.gov/repositories/ malaria-datasets/

[2] Data set from Kaggle

[2] P. Ballester and R. M. Araujo, "On the performance of google net and Alex net applied to sketches," in Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[3] G. P. Gopakumar, M. Swetha, G. Sai Siva, and G. R. K. Sai Subrahmanyam, "Convolutional neural network-based malaria diagnosis from focus stack of blood smear images acquired using custom-built slide scanner," Journal of bio photonics, vol. 11, no. 3, p. e201700003, 2018.

[4] "Uganda ai laboratory." [Online]. Available: https://www.cnn.com/2018/12/14/health/ugandas-first- ai-lab-develops-malaria-detection-app-intl/index.htm

[5] https://www.datacamp.com/community/tutorials/random-forests-classifier-python