

4.2 Prediction Analysis Phase

In the whole process, a lot of data will be collected and stored on the cloud, which can be used to make predictions about levels of several gases. Predictive analysis uses statistical techniques from data mining, predictive modeling and machine learning that analyze the current and historical facts to make predictions let's consider one of the regions where our setup is established, for example, Area 1. Various sensor sets are placed that detect the changes in values of gases. To make predictions regarding the Methane gas levels, analysis is applied to the cloud data. The dependency on Methane gas and some fixed parameters like weight. Preprocessing is done on this data. The data obtained is analyzed for patterns and trends. In our Methane gas example, the dataset shows a dependency on the weights of wastes with the Methane gas. The prediction algorithm that best suits this case is regression. In our pre-processed dataset available, the modeling can be done to identify what sort of dependencies are present. We get that there is a linear dependency between the weight of wastes dumped (kgs) and gas released. In the linear regression model, a line is found that most closely fits the data according to the specific mathematical criterion.

In our Methane gas case from the graph we realize that a linear line can be fitted the equation can be shown as:

$$MG = a + b * (W) \quad (1)$$

MG—Methane Gas levels, W—Weight of wastes dumped, a —y intercept, b —slope of line. A generalized equation for any gas can be written as:

$$G = a + b * (W) \quad (2)$$

The weight (W) is an independent variable while the Methane gas levels (MG) depend on it. Thus depending on the trends and patterns in our dataset, the analysis is done and the regression technique is decided. The modeling contains data training thus finding the values of unknowns a and b of the equation. The values of a and b can be found by:

$$a = \frac{(\sum G) * (\sum (W)^2) - ((\sum W)^* (\sum (W * G)))}{n * (\sum (W^2)) - (\sum W)^2} \quad (3)$$

$$b = \frac{n * (\sum (G * W)) - ((\sum W) * (\sum G))}{n * (\sum (W^2)) - (\sum W)^2} \quad (4)$$

where G is generalized for all gas levels, W is weight of wastes dumped, n is the number of datasets. Thus calculating a and b we get the line equation which is used to predict a gas level value given the weight of wastes dumped in the region. The predicted values are sent to the dashboard for display.

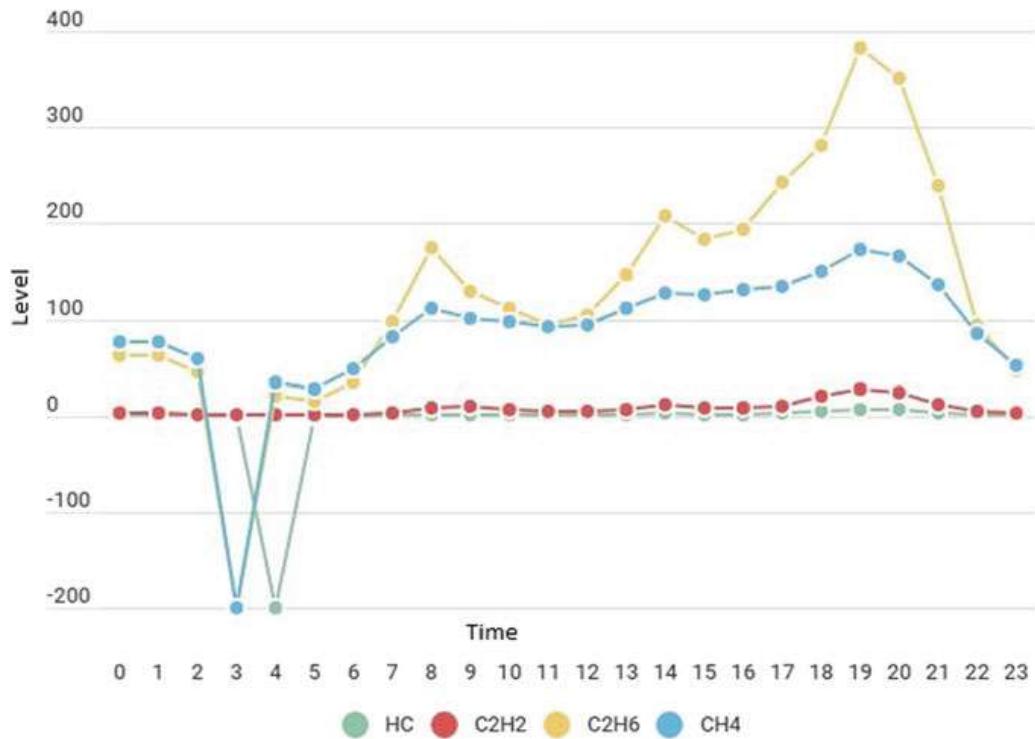


Fig. 4 Line chart for different levels of gases

4.3 Dashboard Phase

The results can be represented in different formats like line-chart, radial-chart, pie-chart, bar-chart. Here the collected data is displayed using a line-chart as shown in Fig. 4. Here the data focus called markers and it showed by straight-line fragments.

The representation can also be done as a radial-Chart or Spider Chart. It is a graphical strategy that shows multiple information in two-dimensional formats. In this at least we have to provide three factors. Here, we have given 4 quantitative factors i.e. HC, C₂H₂, C₂H₆, CH₄ is shown in Fig. 5.

5 Conclusion

The system has been designed as per the requirements. The system successfully retrieves sensor datasets from the source of gas sensors, prediction analyses the benchmark values of gas and delivers a prediction in an easily understandable manner to the user with a dashboard with graphs. The single predicted value and the graphs displayed help the user to understand gas levels better. The dashboard interface is minimal and easy to use.

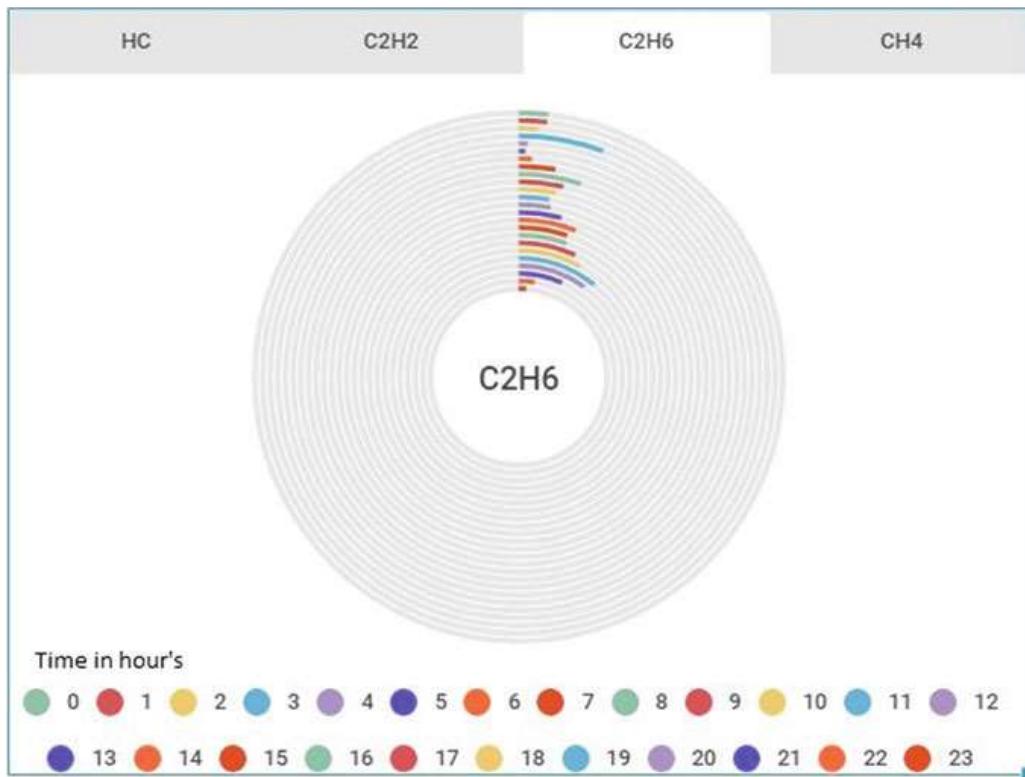


Fig. 5 Radial chart for C_2H_6 gas

For future enhancement, we can consider improving the prediction mechanism by applying neural, and support for user-level analysis. There can be a notification system that can alert the subsequent phases of the waste management system to get alert based on the gas values. Finally, sensors, prediction analysis efficiency can be increased further so that the dataset retrieval can be quick.

References

1. Mehmood, Y., Ahmad, F., Yaqoob, I., Adnane, A., Imran, M., Guizani, S.: Internet-of-things-based smart cities: recent advances and challenges. *IEEE Commun. Mag.* (2017)
2. Wetchakun, K., Samerjai, T., Tamaekong, N., Liewhiran, C., Siriwonga, C., Kruefu, V., Wisitsoraat, A., Tuantranont, A., Phanichphant, S.: Semiconducting metal oxides as sensors for environmentally hazardous gases. Elsevier (2011)
3. Resmi, N.G., Fasila, K.A.: E-waste management and refurbishment prediction (EMARP) model for refurbishment industries. *J. Environ. Manag.* (2017)
4. Sedrakyan, G., Mannens, E., Verbert, K.: Guiding the choice of learning dashboard visualizations: linking dashboard design and data visualization concepts. *J. Vis. Lang. Comput.* (2018)
5. Rovetta, A., Xiumin, F., Vicentini, F., Minghua, Z., Giusti, A., Qichang, H.: Early detection and evaluation of waste through sensorized containers for a collection monitoring application. **29**(12), 2939–2949 (2009)

6. Internet of things: challenges and state-of-the-art solutions in internet-scale sensor information management and mobile analytics. In: 2015 16th IEEE International Conference on Mobile Data Management (2015)
7. Bringing IoT and cloud computing towards pervasive healthcare. In: 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (2012)
8. Bilal, M., Oyedele, L.O., Akinade, O.O., Ajayi, S.O., Alaka, H.A., Owolabi, H.A., Qadir, J., Pasha, M., Bello, S.A.: Big data architecture for construction waste analytics (CWA): a conceptual framework. **6**, 144–156 (2016)
9. Kekalainen, F.: IOT and big data solving problems for the waste and recycling industry (2016)
10. Niska, H., Serkkola, A.: Data Analytics approach to create waste generation profiles for waste management create waste generation profiles for waste management and collection. Finland (2018)

Brain Tumour Detection in MRI Using Deep Learning



S. Shanmuga Priya, S. Saran Raj, B. Surendiran, and N. Arulmurugaselvi

Abstract Tumour is the assortment or mass growth of abnormal cells within the brain. Individuals are still dying because of brain tumour. So early and accurate detection of brain tumour will scale back the death rate. The computer-aided application will help to give accurate detection of brain tumour. The process of performing some operations on image to get useful information is called image processing. At present, image processing is rapidly growing technology. It consists of many types of imaging methods, like MRI, CT scans, X-rays. By this, we can find small abnormalities in the human body and brain. The main process of image processing is to extract accurate information from the image. In this paper, the GLCM texture feature and Haralick texture features of the images are extracted. Then, the calculated features are given as an input to various machine learning classifiers to classify the MRI images of the brain. This work carried out with three steps, preprocessing, feature extraction, and classification. Finally, the methods are compared, and it has been found that MLP is the best accuracy classifier.

Keywords Feature extraction · GLCM · Haralick texture feature · Support vector machine · MLP

1 Introduction

The goal of image processing is to extract the characteristics and features from the corresponding image. It plays a key role in MRI medical image processing. Detecting any disease in advance will help for better treatment and increasing the immortality rate. Identification, detection, and classification of brain tumour from MRI images are

S. Shanmuga Priya (✉) · S. Saran Raj · B. Surendiran
Department of Computer Science and Engineering, National Institute of Technology Puducherry,
Karaikal, India
e-mail: shanmugapriyasukumaran30@gmail.com

N. Arulmurugaselvi
Department of ECE, GPT, Coimbatore, India

a tedious and time-consuming task. So we are going for computer-aided application. The human brain is the most critical and composite organ of our central nervous system. The brain is protected by the skull and is very rigid. The cerebral cortex contains 15–33 billion neurons, each neuron connected to a thousand other neurons. Any growth inside such a restricted complex part can cause problems. Reason for the brain tumour is incalculable [1, 2]. But there are two major reasons for brain tumour: (i) radiation and (ii) rare genetic conditions. The most common symptoms of brain tumours are headache, numbness, tingling in the arms or legs seizures, memory problems, mood and personality changes, vomiting and problems in speech, hearing, or vision. There are two common classifications of brain tumour: (i) primary brain tumours originate and stay at the brain. (ii) A secondary brain tumour is so common [3]. In this cancer cells originate from somewhere in the body like lungs, breasts, kidney and travel to the brain. Some brain tumours contain cancer cells. Benign brain tumours do not have cancer cells. They are coming under a primary brain tumour class. They grow slowly and can often be removed and rarely spread to the brain cells around. Depending on where they are located in the brain, they can be life-threatening. Malignant tumours may have cancer cells, and they come under primary or secondary tumours. Malignant tumours rarely spread beyond the brain or spinal cord. Detecting malignant tumour is more difficult than a benign tumour [4].

There are so many different algorithms which have been developed for brain tumour detection and classification. Normally, the automatic classification and detection of the image is challenging. The major aim of this system is detecting, identifying, classifying whether the MRI of the brain is normal or tumour. The paper analyses the performance and accuracy of the classifier-based brain tumour classification against deep neural networks. In the proposed work, the brain tumour detection is carried out in three steps: (i) preprocessing, (ii) feature extraction, and (iii) classification. Preprocessing includes feature detection and feature extraction followed by training the images on different classifier based on the extracted features and finally measuring the accuracy and performance of different classifiers. The results of the work are to classify an image with a tumour or not. Also analysing the classification accuracy of the SVM and MLP classifiers [5].

2 Literature Survey

Telrandhe et al. [1] proposed the work to identify the region of the tumour and detailed diagnosis of the tumour. Image processing is used in the medical tools for the detection of the tumour. In this K-means segmentation is used with preprocessed image. Support vector machine (SVM) is used in an unsupervised manner to maintain the pattern for future use.

Gurbina et al. [6] studied some types of brain tumours using MR Images. In their paper brain they carried out tumour detection and classification of MRI using wavelet transforms and SVM.

Halder et al. [3] proposed a method to extract the tumour using K-means algorithm and followed by object labelling algorithm. In this paper, preprocessing step is carried out by median filtering and morphological operation.

Nandha et al. [7] explained the importance of the image processing and diagnosing the brain tumour in advance stage. The algorithm proposed in this paper is image processing clustering algorithms and fuzzy C-Means and optimization tools as genetic algorithm and particle swarm optimization. In this, they performed the detection of tumour in two phases: (i) preprocessing and enhancement and (ii) segmentation and classification in the second phase.

Badran et al. [4] proposed a computer-based method for defining tumour region in the brain using MRI. The algorithm proposed in this paper was carried out in the following steps: preprocessing, segmentation, and feature extraction. This paper explains the feature extraction clearly.

3 Proposed Method

The key purpose of this paper is to identify the tumour in brain MRI. In this paper, different types of MR images are used to detect the normal and abnormal brain. The dataset is taken as input having two folders of images, one is normal brain MRI and another folder contains abnormal tumour brain MRI. Totally image dataset has 256 images, 152 normal brain MRI, and 94 brain MRI which have a tumour. Figure 1 shows the complete procedure for the proposed algorithm [8].

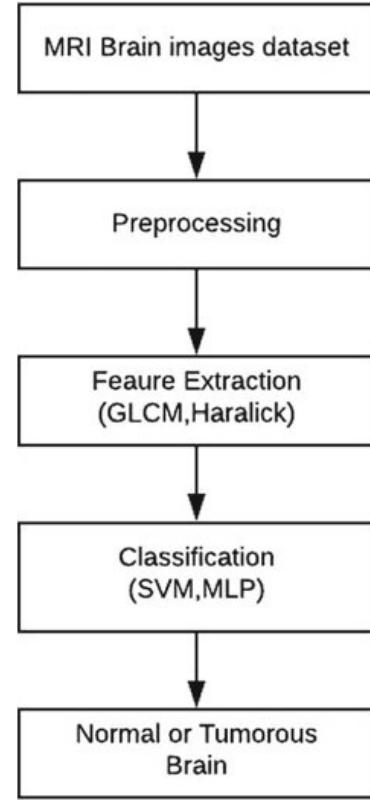
3.1 Preprocessing

Data preprocessing is transforming the normal raw data into an understandable data format. Inconsistent data will cause so many errors. Data preprocessing is a wise method of resolving unwanted errors in the dataset.

3.1.1 Grayscale Imaging

In this paper, MR image dataset is used as input. Mainly, this brain MRI is converted to a grayscale image. In this normal MRI is converted into a grayscale image for better accuracy. The grayscale image is basically black-and-white image. In the grayscale image, red, green, and blue equally spread. It contains only luminance information and not colour information. That is the main reason grayscale imaging method is used. Luminance is more important in distinguishing visual features. To identify important edges or feature grayscale image helps wisely.

Fig. 1 Algorithm of detection and classification



3.2 Feature Extraction

Feature extraction is a set of methods to extract important features from corresponding MRI. The feature extraction is a process to represent the image is a composite and unique form of values or matrix-vector. Features extracted in this paper are GLCM+Haralick texture features [9].

3.2.1 GLCM Texture Feature

Gray-Level Co-Occurrence Matrix (GLCM) is the method to extract the second-order statistical texture [9]. The input image is composed of the pixel, each pixel with an intensity. The GLCM is a tabulation of how often different combinations of gray-level co-occurrence in an image or image section. It gives the estimation of the change in intensity. The GLCM stored as $i \times j \times n$ matrix, where n is the number of GLCM calculated because of different positioning and displacements used in the algorithm and i, j is the rows and columns of the matrix [10]. In GLCM number of rows and number of columns are equal to the number of gray level in the image. By using GLCM method the basic features contrast, correlation, energy, homogeneity are extracted, and other features like mean, RMS, standard deviation, entropy also extracted.

Table 1 Extracted GLCM texture feature from the image data set

Id	Contrast	Correlation	Energy	Entropy	Homogeneity	Mean	RMS	Std	Class
img1.jpg	1.36	0.67	0.10	0.34	0.76	504	928	1210	Yes
img2.jpg	1.23	0.75	0.29	0.00	0.87	504	1160	2130	No
img3.jpg	0.71	0.79	0.27	0.20	0.88	254	586	1020	No
img4.jpg	0.15	0.60	0.55	0.66	0.93	254	631	1500	No

GLCM considers the spatial relationship of the pixel.

a. *Syntax:*

```
glcms = graycomatrix(I)
glcms = graycomatrix(I,Name,Value,...)
```

b. *Description:*

`glcms = graycomatrix(I)` creates a gray-level co-occurrence matrix(GLCM) from image I. Although there are a function `graycoprops()` in MATLAB Image Processing Toolbox that computes four parameters Contrast, Correlation, Energy, and Homogeneity. Table 1 shows the extracted GLCM texture feature from the image data set.

3.2.2 Haralick Texture Feature

Haralick Texture feature is well-known texture feature is the function of normalized GLCM. This also depends on the gray levels in the image. It is a widely used texture measure algorithm. The starting point for these features is the GLCM. This matrix is square with dimension N_g , where N_g is the number of gray levels in the image. Value[i, j] of the corresponding matrix is calculated by counting the total number of times value of i adjacent with value j and then furcate the total matrix by the count of the comparison made [11]. The GLCM created when ‘Symmetric’ is set to true is symmetric across its diagonal, and is equivalent to the GLCM described by Haralick (1973). The GLCM produced by the following syntax, with ‘Symmetric’ set to true. Table 2 shows the extracted Haralick texture feature from the image data set.

3.3 Classification

After the features are extracted and selected, the classification step is performed on the resulting feature vector of corresponding images given as input to the classifiers SVM, MLP. Classification is performed by using a tenfold cross-validation technique. Also for evaluating the performance of the selected classifier, WEKA is used to import other machine learning classification algorithm. SMO-SVM and MLP are the classification algorithm we used here with tenfold cross-validation [12].

Table 2 Extracted Haralick texture feature from the image data set

Id	Contrast_H	Correlation_H	Energy_H	Entropy_H	Homogeneity_H	Mean_H	RMS_H	Std_H	Class
img1.jpg	1.36	0.67	0.10	0.34	0.76	504	928.01	1211.28	Yes
img2.jpg	1.23	0.75	0.29	0.00	0.87	504	1155.38	2128.46	No
img3.jpg	1.31	0.62	0.24	0.00	0.84	504	1109.80	1941.92	No
img4.jpg	0.21	0.46	0.52	0.66	0.90	504	1212.55	2887.41	No

3.3.1 Support Vector Machine

Support Vector Machine is a discriminative classifier defined by separating a hyper lane. SVM is majorly many, as it gives noteworthy accuracy with less computation time. It can be used for classification and as well as regression. It is an algorithm to find a hyper lane and bifurcates all data points [13]. Hyperlane means which one has the largest margin between a couple of classes. In this paper, SVM classifier classifies the normal and tumorous brain MRI with 74% accuracy. First labelled data set obtained and split into test set. Then classifier is trained using the extracted features from the proposed image dataset.

3.3.2 Multilayer Perceptron

Multilayer Perceptron (MLP) is a feedforward Artificial Neural Network which takes a set of inputs and generates a set of outputs. MLP especially suits for simpler dataset. It connects the multiple layers in a directed graph, it means signal passes through perceptron in only one way, backpropagation suits for supervised learning. It resembles the characteristics of a fully connected layer where each perceptron in the network connects to every other perceptron. It consists of three layers: (i) Input layer, (ii) Output layer, (iii) Hidden layer.

4 Experimental Results

4.1 Dataset

The dataset contains 2 folders: yes and no which contains 256 Brain MRI Images. The folder yes contains 152 Brain MRI Images that are tumorous and the folder no contains 94 Brain MRI Images that are non-tumorous. Each image resolution is 128 × 128.

4.2 Results

Here all the algorithms implemented and their confusion matrix is added. Result of all algorithms implemented is shown in Table 3.

Results of SVM algorithm implementation:
Confusion Matrix

Table 3 Classification results

S. No.	Algorithm	Accuracy (%)	Error rate (%)
1	SVM	76.12	23.7
2	MLP	82.18	17.81

a	b	← classified as
130	20	$a = \text{yes}$
37	60	$b = \text{no}$

Results of MLP algorithm implementation: Confusion Matrix

a	b	← classified as
125	25	$a = \text{yes}$
19	78	$b = \text{no}$

5 Conclusion and Results

This paper developed a different classifier to classify brain MRI into 2 types of normal and abnormal. The proposed approach gives very promising results in classifying the healthy and pathological brain with 82% high accuracy. The approach is limited by the fact that it necessitates fresh training each time whenever there is a change in the image database. The good results achieved using different Feature Extraction method and Classifiers could be employed with CNN in the future and compare the results.

References

1. Giraddi, S., Vaishnavi, S.V.: Detection of brain tumor using image classification. In: 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), Mysore, pp. 640–644 (2017)
2. Kapoor, L., Thakur, S: A survey on brain tumor detection using image processing techniques. In: 2017 7th International Conference on Cloud Computing, Data Science & Engineering—Confluence, Noida, pp. 582–585 (2017)
3. Halder, A., Giri, C., Halder, A.: Brain tumor detection using segmentation based Object labeling algorithm. In: International Conference on Electronics, Communication and Instrumentation (ICECI), Kolkata, pp. 1–4 (2014)
4. Badran, E.F., Mahmoud, E.G., Hamdy, N.: An algorithm for detecting brain tumors in MRI images. In: The 2010 International Conference on Computer Engineering & Systems, Cairo, pp. 368–373 (2010)

5. Iftekharuddin, K.M., Zheng, J., Islam, M.A., Ogg, R.J., Lanningham, F.: Brain tumor detection in MRI: technique and statistical validation. In: 2006 Fortieth Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, pp. 1983–1987 (2006)
6. Gurbină, M., Lascu, M., Lascu, D.: Tumor detection and classification of MRI brain image using different wavelet transforms and support vector machines. In: 2019 42nd International Conference on Telecommunications and Signal Processing (TSP), Budapest, Hungary, pp. 505–508 (2019)
7. Gopal, N.N., Karnan, M.: Diagnose brain tumor through MRI using image processing clustering algorithms such as Fuzzy C Means along with intelligent optimization techniques. In: 2010 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, pp. 1–4 (2010)
8. Latha, M., Surya, R.: Brain tumour detection using neural network classifier and k-means clustering algorithm for classification and segmentation. IIR J. Soft Comput. **1**(1), 27–32 (2016)
9. Shree, N.V., Kumar, T.N.R.: Identification and classification of brain tumor MRI images with feature extraction using DWT and probabilistic neural network. **5**(1), 23–30 (2018)
10. Mohanaiah, P., Sathyanarayana, P., Guru Kumar, L.: Image texture feature extraction using GLCM approach. Int. J. Sci. Res. Publ. **3**(5), 1 (2013)
11. Haralick, R.M., Shanmugam, K., Dinstein, I.: Textural features for image classification. IEEE Trans. Syst. Man Cybern. **SMC-3**(6), 610–621 (1973)
12. Anjali, R., Priya, S.: An efficient classifier for brain tumor classification. Int. J. Comput. Sci. Mob. Comput. **6**(8), 40–48 (2017)
13. Kharrat, A., Gasm, K.: Hybrid approach for automatic classification of brain MRI using genetic algorithm and support vector machine. Leonardo J. Sci. **17**, 71–82 (2010)

Parallel Implementation of Luhn's Algorithm for Credit Card Validation Using MPI and CUDA



P. Karthik G. Kudva M. L. Shreyas B. Ashwath Rao ,
Shwetha Rai , and N. Gopalakrishna Kini

Abstract The Luhn's algorithm is the first line of defense in various e-commerce sites and is utilized to validate credit card numbers. With increase in usage of credit cards validation process also needs to be faster. This fast processing is achievable by parallel processing. This paper intends to make use of MPI and CUDA programming to enhance the computation time of Luhn's algorithm of validation of multiple credit cards in parallel.

Keywords Luhn's algorithm · Parallel programming · Credit card validation · MPI programming · CUDA programming

1 Introduction

In recent times, Credit cards have become a dominant mode of online transactions. By the end of 2018, there were more than 44.2 million credit cards and 958.2 million debit cards in India [1]. But, as the credit card market has grown significantly in recent years, so too have crimes such as credit card fraud, due to which banks and cardholders suffer huge losses. Development in the detection and avoidance of credit

P. K. G. Kudva · M. L. Shreyas · B. A. Rao () · S. Rai · N. G. Kini
Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal
Academy of Higher Education, Manipal, Karnataka 576104, India
e-mail: ashwath.rao.b@gmail.com

P. K. G. Kudva
e-mail: karthikkudva@gmail.com

M. L. Shreyas
e-mail: shreyasml95@gmail.com

S. Rai
e-mail: shwetharai.cse@gmail.com

N. G. Kini
e-mail: ng.kini@manipal.edu

card fraud has become the primary concern of banks in order to prevent security breaches. One of the most common errors that occur in any e-commerce site is typing errors.

With the increasing usage of credit cards, the processors validating the credit cards have to handle a huge load. Implementing parallel processing using MPI and CUDA will succeed in reducing the credit card number validity processing time. In real-time the system has to validate millions of credit cards together at any given instance.

MPI stands for Message passing Interface. MPI is a communication protocol used in parallel computation to achieve interaction between different processors. MPI supports both point to point and collective communication. MPI makes use of inbuilt APIs to provide communication. These APIs can be called using C, C++, Fortran, etc. and is able to interface with languages like java, python, and C# [2].

CUDA stands for Compute Unified Device architecture. Nvidia created CUDA to act as both platform and API for parallel computing on its own GPU. CUDA works by copying data from the main memory to the GPU memory and running kernels on the set of data in parallel [3].

2 Literature Survey

With the increased usage of credit cards, there is also a significant rise in credit card fraud and traffic. There are multiple types of credit card frauds one of which is credit card generators [4]. In this method, one program will generate multiple valid credit card numbers for a single account. The software uses Luhn's algorithm for creating valid fake credit card numbers. When a fraudster tries to use these multiple cards online, the validation system checks each and every one of those credit cards and sends it to the next stage of verification. However, when there is a huge surge of fake credit cards, all of them should be validated which is a big load on the validation system.

In recent times, multiple research is done to avoid credit card fraud using different techniques. One of them is the use of decision trees with the combination of Luhn's and Hunt's algorithm in the validation of mailing and billing addresses [5]. There are several researches done on improving Luhn's algorithm based on the number of digits entered for validation [6], which filters out invalid cards but still has to process fake cards with a valid number of digits.

Luhn's algorithm is also used as an application for wired and wireless internet which is used for credit card validation [7].

The survey indicates that there are multiple efforts made to improve the validation process of credit cards using Luhn's algorithm but none of them deals with the heavy load and optimization of processing time of Luhn's algorithm.

This paper aims to improve the processing time of credit card validation using Luhn's algorithm with a large data set using MPI and CUDA programming.

3 Methodology

3.1 Luhn's Algorithm

Luhn's algorithm [1] is used to create and validate credit card numbers from all different vendors. It was developed by scientist Hans Peter Luhn in 1954. It is a checksum algorithm extensively used by the public domain to protect consumers and companies against accidental errors like skipping or entering wrong numbers (Fig. 1).

Steps Followed [5]:

Step 1: From the unit place of the number, multiply the value of every second digit by 2.

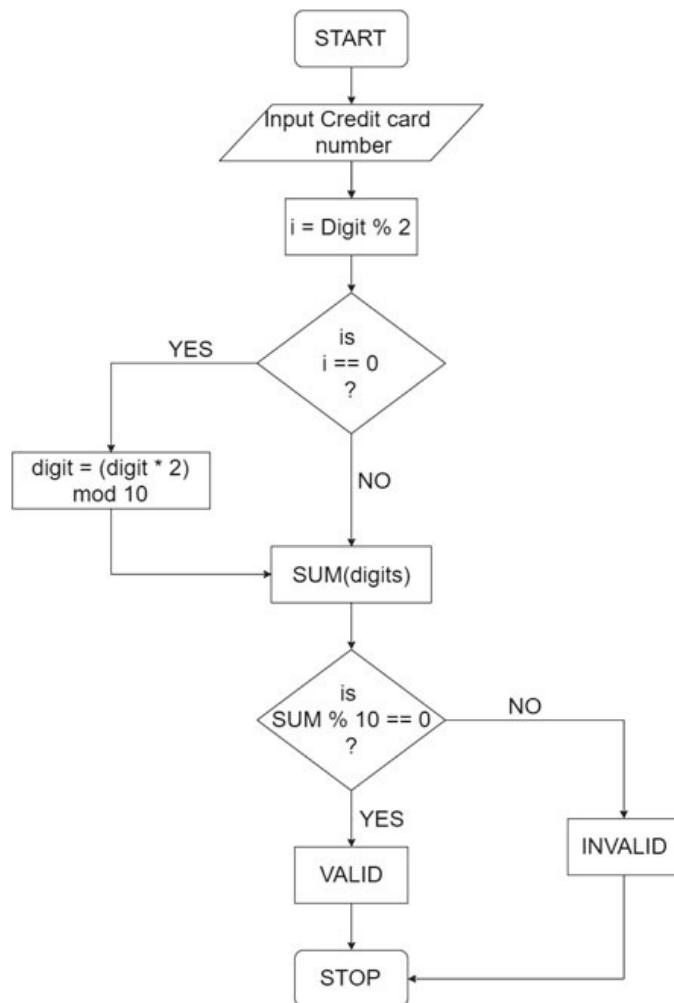


Fig. 1 Flowchart of Luhn's algorithm

Step 2: If the results obtained in step 1 gives a 2-digit number (e.g., $9 \times 2 = 18 > 9$), then the digits of the product obtained are to be added (e.g., $18: 1 + 8 = 9$), to obtain a single-digit number.

Step 3: Perform summation of the digits.

Step 4: If the summation modulo 10 gives 0 (if the sum is a multiple of 10) then the number is valid; else invalid.

3.2 Parallelism on Luhn's Algorithm

The programs are run on the Ubuntu OS on a system with 8 GB RAM INTEL Octa-core processor. Both sequential and parallel programs are run in similar environments and with the same set of data.

Using MPI

MPI header file “mpi.h” and MPI APIs: MPI_Scatter, MPI_Barrier, and MPI_Wtime are used in the program. MPI_Scatter API divides a big array into a number of smaller parts equal to the number of processes and sends each process (including the root) a part of the array in rank order. The program takes n number of credit cards as input, the system with k processor will allocate n/k number of credit cards to each processor by making use of MPI API “MPI_Scatter”. Each processor will execute the Luhn's credit card validation algorithm in parallel on each set of data input, thereby enhancing the performance and reducing the computing time. MPI_Wtime returns a floating-point number of seconds, representing elapsed wall-clock time since some time in the past. MPI_Wtime is used to calculate the time consumed by the processors for processing. MPI_Barrier API blocks until all processes in the communicator have reached that routine. MPI_Barrier API is used to synchronize all the processors before calculating the time so that all of the processors reflect the same time. The number of processors to be used can be defined by the user at compile time [2].

Using CUDA

In the CUDA program the header file “cuda_runtime.h” and “device_launch_parameters.h” are used. The number of threads to be created is predefined in the program as ‘ n ’. The program takes n number of credit cards as input. Each thread will execute Luhn's credit card validation algorithm in parallel on each set of data input thereby enhancing the performance and reducing the computing time. The kernel function is used in order to execute the main logic. This allocates a thread unit for every input, hence making it faster and more efficient for large sets of data. In the program, the threads are created as one-dimensional grid and one-dimensional block. This can be modified to extend the number of threads based on system configuration and data set size [3].

In this paper, sample credit card numbers in the range of 100–1000 are used as input for evaluating the performance for different sizes of data sets.

4 Results

Processing time of execution of Luhn's algorithm in parallel using sequential, MPI, and CUDA programming is given in Tables 1, 2, and 3, respectively. Figures 2 and 3

Table 1 Processing time for sequential program

Input count	100	200	400	500	1000
Processing time (in ms)	0.638	2.521	4.944	5.621	12.673

Table 2 Processing time for MPI program

No. of processors	Input count	100	200	400	500	1000
5	Processing time (in ms)	0.051	0.077	0.108	0.137	0.170
		0.039	0.057	0.066	0.081	0.094
		0.049	0.061	0.067	0.072	0.079
		0.033	0.040	0.043	0.051	0.053
		0.031	0.034	0.039	0.042	0.043
		0.034	0.036	0.039	0.047	0.052

Table 3 Processing time for CUDA program

Input count	100	200	400	500	1000
Processing time (in ms)	0.043	0.043	0.045	0.049	0.045

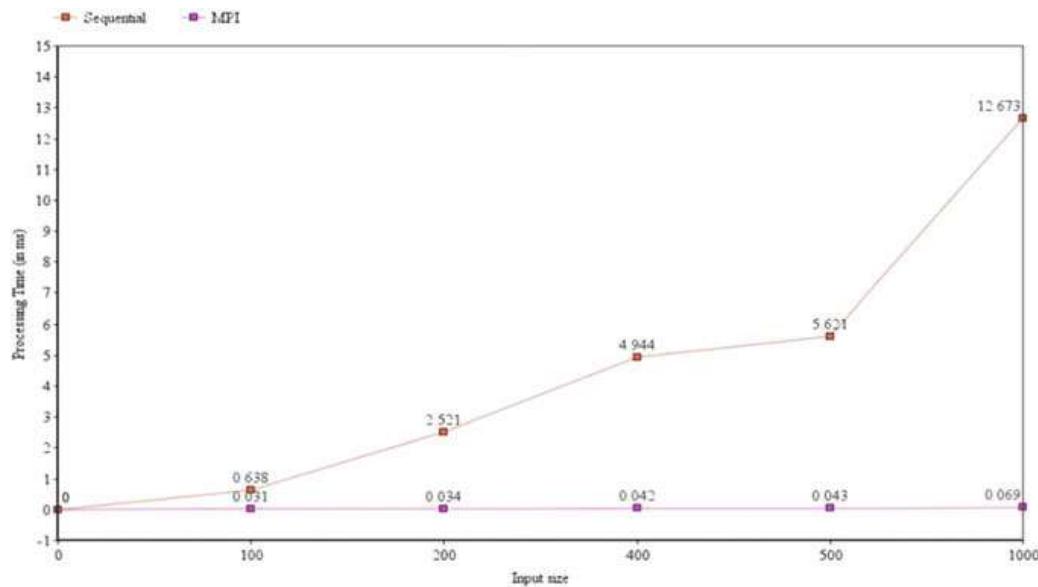


Fig. 2 Processing time of sequential versus MPI programming

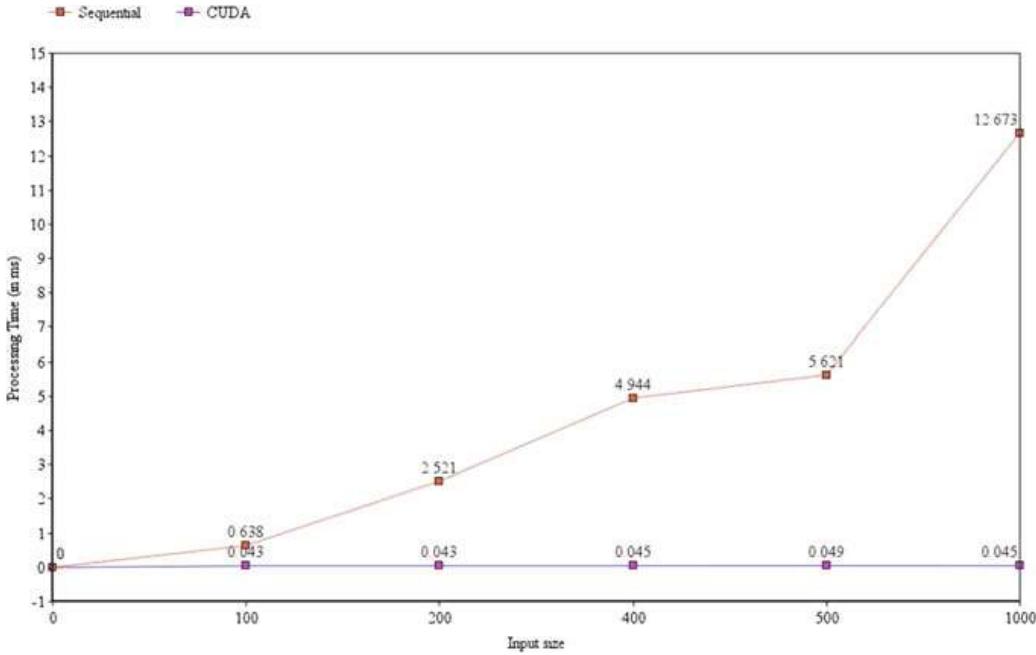


Fig. 3 Processing time of sequential versus CUDA programming

give the graphical comparisons of processing time between Sequential and MPI and also Sequential and CUDA techniques respectively.

From tables and figures, it is inferred that using MPI and CUDA, the processing time of Luhn's algorithm can be improved significantly for large data sets. Parallel programs are evaluated by their speed-up percentages. Higher the speed-up, the higher the improvement in processing time. The speed-up of Sequential and MPI and Sequential and CUDA programs are shown in Tables 4 and 5 respectively.

From tables, it is evident that there is a significant speed-up in parallel programming compared to sequential programming. The improvement is exponential as the size of the data set increases.

Table 4 Speed-up of sequential versus MPI

Input size	Sequential program run-time (in ms)	MPI program run-time (in ms)	Speed-up = (Sequential processing time/MPI processing time) × 100 (%)
100	0.638	0.031	2058
200	2.521	0.034	7414
400	4.944	0.042	11,771
500	5.621	0.043	13,072
1000	12.673	0.069	18,366

Table 5 Speed-up of sequential versus CUDA

Input size	Sequential program run-time (in ms)	CUDA program run-time (in ms)	Speed-up = (Sequential processing time/CUDA processing time) × 100 (%)
100	0.638	0.043	1483
200	2.521	0.045	5602
400	4.944	0.049	10,089
500	5.621	0.045	12,491
1000	12.673	0.043	29,472

5 Conclusion and Future Scope

This paper shows that using MPI and CUDA programming, the processing time of Luhn's algorithm for credit card validation can be significantly improved as compared to the traditional sequential method.

In the current market, there are multiple credit card vendors all around the world. Each vendor can be identified by the first digit of the credit card number. Similarly, there are other parameters like Bank number, account number, etc. [8]. Using these details, the algorithm can be extended to identify specific details of the credit card and make a decision as to which database the data should be transmitted for further validation and verification.

References

- Verma, S.: India had 44.2 M credit cards, 958.2 M debit cards in December 2018—MediaNama. [online] MediaNama. Available at <https://www.medianama.com/2019/03/223-india-had-44-2m-credit-cards-958-2m-debit-cards-in-December-2018>. Accessed 14 Sep 2019
- Gropp, W., Lusk, E., Skjellum, A.: *Using MPI*. The MIT Press, Cambridge, MA (2014)
- Sanders, J., Kandrot, E.: *CUDA by Example*. Pearson, India (2010)
- Save, P., Tiwarekar, P., Ketan, N., Mahyavanshi, N.: A novel idea for credit card fraud detection using decision tree. *Int. J. Comput. Appl.* **161**(13), 6–9 (2017)
- Hussein, K., Sani, N., Mahmod, R., Abdullah, M.: Enhance Luhn algorithm for validation of credit cards numbers. *Int. J. Comput. Sci. Mob. Comput.* **2**(7), 262–272 (2019)
- Bhatla, T.P., Prabhu, V., Dua, A.: *Understanding Credit Card Frauds* (2003)
- Li, C., Yao, Z.: The validation of credit card number on the wired and wireless internet. *J. Netw.* **6**(3) (2011)
- Digit, W. (2019). What Is A Credit Card Number? The Meaning of Each Digit. [online] WalletHub. Available at <https://wallenthub.com/edu/cc/what-is-a-credit-card-number/44066>. Accessed 19 Sep 2019

Malayalam POS Tagger—A Comparison Using SVM and HMM



K. Usha and S. Lakshmana Pandian

Abstract Many Parts Of Speech (POS) taggers for the Malayalam language has been implemented using Support Vector Machine (SVM), Memory-Based Language Processing (MBLP), Hidden Markov Model (HMM) and other similar techniques. The objective was to find an improved POS tagger for the Malayalam language. This work proposed a comparison of the Malayalam POS tagger using the SVM and Hidden Markov model (HMM). The tagset used was the popular Bureau of Indian Standard (BIS) tag set. A manually created data set which has around 52,000 words has been taken from various Malayalam news sites. The preprocessing steps that have done for news text are also mentioned. Then POS tagging has been done using SVM and HMM. As POS tagging requires the extraction of multiple class labels, a multi-class SVM is used. It also performs feature extraction, feature selection, and classification. The word sense disambiguation and misclassification of words are the two major issues identified in SVM. Hidden Markov Model predicts the hidden sequence based on maximum observation likelihood which reduces ambiguity and misclassification rate.

Keywords Natural language processing · POS tagger · Support vector machine · Hidden Markov model · Tagset · Feature extraction · Document term matrix · Feature selection and classification · Maximum likelihood · Viterbi

1 Introduction

Language can be treated as one of the basic means of communication among people and its primary purpose is information sharing across millions of people in the world.

K. Usha · S. L. Pandian

Department of Computer Science and Engineering, Pondicherry Engineering College, Puducherry 605014, India

e-mail: usha.dileep@pec.edu

S. L. Pandian

e-mail: lpandian72@pec.edu

The growth of natural language processing has brought forward a lot of achievements to European languages whereas Asian languages like Chinese, Hindi, and Japanese are still looking for improvements. The major difficulty these languages suffer is the complexity in character set, syntactic representation, and semantics of the language. So we have manually created a text which is copied from different Malayalam news sites. We have tokenized the text for the ease of processing.

This multi-class classification is applicable for text categorization [1, 2] and in information retrieval systems. The work by Manjusha et al. [3] proposed the issues related to Malayalam literature and data set availability. There is no standard benchmark data set available for Malayalam text. The work proposed by Karthik et al. gives a detailed description of different issues with Indian language scripts and its transformations [4]. Another work explained a Deep belief Network (DBN) for character recognition in which Restricted Boltzmann Machines (RBM) are trained using the Contrastive Divergence (CD) algorithm. A recent paper suggested Arabic Character Recognition using Convolution Neural Network with pooling and Rectified Linear Units (RELU) [5] nonlinearity as the activation function. A Stochastic Gradient Descent (SGD) Algorithm was applied to minimize the error function.

In Sect. 2 explained about Malayalam Literature and Sect. 3 gives a brief explanation about related work. In Sect. 4 a comparison between SVM and HMM is described. Finally, Sect. 5 gives the results and discussion. SVM is a hyperplane with a maximum margin [2] to separate positive samples from negative samples. HMM is a probabilistic model that makes use of observed states and current state, and thereby predicts the hidden states of a sequence. The tagset used is the BIS tag set. We make use of almost all tags in the set. The data set is a manually created one from different Malayalam news sites and contains around 52,000 words.

2 Malayalam Literature

Malayalam is the native language of Kerala and has spoken by around 35 million native speakers in India and abroad. Malayalam text character set comprises of 15 vowels (swaram), 36 consonants (vyanjanam), 5 chillu alphabets (chillu aksharam). In addition to that, it has a list of conjuncts [3] (Koottaksharams) and separate diacritic symbols which are present for all vowels other than ‘‘അ’’ equivalent to ‘a’ alphabet in English. Due to the presence of diacritics and to represent any consonant character require 1 or 2-byte storage in its complete form. Malayalam has a list of suffixes and affixes attached to the respective word. Certain suffixes can be applicable to one or more classes of words. But every word is associated with a list of suffixes. One example of the suffix is vibhakthi prathayams (ae, odu, kku, nu, aal, nte, ude, il) attached to a noun. To represent negation (illa), affirmation (undu), singular and plural form (maar, kal, gal, kar) suffixes are used. Stemming is usually performed to get the base form of a verb. To get a generalization on language architecture the suffixes and affixes present along with a word were removed. Similarly, stop words are also less important words in a document and does not have any significance with

the matter prescribed in a document and hence removed. Preprocessing was done to reduce complexity. The tag set used is the BIS standard which has around it can handle almost all classes of words in Malayalam. Malayalam language mainly uses four types of sandhi. When compounded words occur it is difficult to find the POS tag if the subsequent base word belongs to different class labels. Hence a sandhi splitter helps to resolve problems related to compound word splitting. Tokenization in Malayalam is the same as in English where white space characters act as the separator of every word.

3 Related Work

SVM was a linear classification technique in which, kernel-based methods were also employed that maps the problem into a different high dimensional feature space so as to handle nonlinear data sets. A multi-class classification of Malayalam text followed by POS tag assignment using SVM and HMM were discussed here. Multi-class classification can be implemented in any one of the following ways: one-against-one classifier (OAO), one-against-all (OAA), and decision directed acyclic graph (DDAG). The main concentration was on the use of multi-class classification to classify the documents based on OAA. HMM model was used to make a structured prediction based on generative sequence models.

In the recent work proposed an ensemble model [6] combining five basic classifiers including SVM were used to classify the data that signifies the cause of accidents. In addition to that Sequential Quadratic Programming (SQP) is used to optimize the weight of respective classifier and unsupervised chunking was also used to extract the common objects that lead to the cause of accidents. Other SVM classifiers [7–9] based models existed which make use of Term Frequency, Inverse Document frequency (TF_IDF), vectorization, training, and classification algorithm, and word feature, POS features, and lexicalized features respectively. There were proposed works [10–12] in which a CRF based POS tagger model which makes use of word-level features for predicting the POS tag. The work on the Sinhala language created a tag set and a POS tagger [9] using SVM in which tag set assignment was done on three levels.

The latest work [13] proposed HMM to relate a sequence of words from an activity label to its semantic components with the help of transition and emission probability matrices. In these works [14–18] proposed a POS tagger using HMM where n-gram approach is implemented using Viterbi algorithm. Another work proposed a context-aware point-of-interest category prediction scheme [5] that uses the natural language processing model. A Bayesian estimate computes the posterior probability using Gibbs Sampling. A work that [18] builds an infinite HMM with Pitman-Yor Dirichlet process with hyperparameters that control the number of states in infinite HMM. This study leads to an inference that HMM is difficult to implement for Indian Languages because of the absence of annotated corpus whereas English provides 94% annotated corpus and the number of corpus for research work is more there. Another work [12]

POS tagging and chunking was done using CRF and Transformation-Based Learning. CRF works better with window size 4, previous state, and current state providing an accuracy of 73.47%. Johnson [19] proposed an HMM-based model which computes maximum likelihood via Expectation-Maximization.

4 Proposed Work

The proposed work concentrates on classifying the sequence of words from a new text corpus into different parts of speech categories and assigns a tag based on the structure and context of the word. For making the data appropriate for classification we have performed some preprocessing techniques like tokenization, stop word removal, sandhi splitting, affix removal, and stemming. Here we are making use of sandhi splitter [20] to split complex and compounded words that occur in Malayalam news text. Now we apply SVM and HMM on the text corpus to see which is having better accuracy.

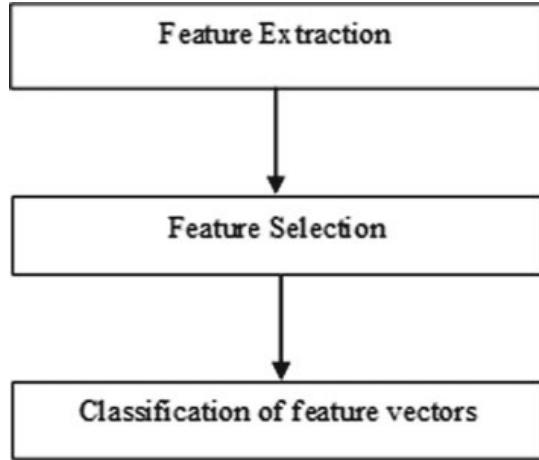
4.1 Support Vector Machine

In this work, a multi-class SVM [8, 21] was taken as it requires multiple class labels to categorize the words in the given news text corpus. So, we have as many SVMs as per the number of categories in the POS tag set. The reason, why we have chosen SVM is, SVM overfitting does not occur as it has the capacity equal to the margin of separation between the support vectors and hyperplane, hence it does not get affected by the dimensionality of the data. Also, SVM can handle high dimensional data using kernels. But it has the problem of high training time and classification time associated with high dimensional data. Training SVM is a quadratic optimization problem in which the hyperplane is represented by $w^T x + b$, where w is the vector of hyperplane co-efficient and b is the bias term.

Training an SVM with a linear kernel is faster than with any other kernel. But if the number of categories is more than two, we need a multi-class classification scheme. The extracted feature is fed into a multi-class SVM which will categorize the words into appropriate class viz. Nouns, verbs, adverbs, adjectives. The collection of text is converted into feature vectors and each vector value is assigned a Euclidean score which is always less than 1.0. There is a training phase followed by a testing phase. Test data set contains unknown words that help us to validate the accuracy.

Another major trouble we face here is the conversion of text data to numerical data. We have chosen this TF_IDF [21] scheme as it vectorizes easily. Feature Extraction, Feature Selection, and classification have been incorporated with this multi-class SVM as shown (see Fig. 1). Feature extraction is implemented using Document Term Matrix (DTM) that represents the dimension in which the row number specifies the number of unique words selected from the text and the column number specifies the

Fig. 1 Classification using multi-class SVM



features selected from those words. The features selected may be the stem or part of a stem that in turn depends on the Euclidean value greater than 0.5. The class labels are assigned as per the BIS tag set which contains around 36 tags.

4.2 Hidden Markov Model

Hidden Markov Model is based on Markov Chain and is the best statistical model to assign POS tags to text data in natural language processing when there is an annotated corpus. Hidden Markov Model predicts the probability of hidden events based on a set of observable events T that makes the transition through a set of N states using transition probability matrix, observation probability, and initial probability distribution. It has three basic problems viz., likelihood, decoding, and learning. The likelihood is based on forward algorithm, decoding through Viterbi algorithm, and training through forward–backward algorithm.

Due to the unavailability of labeled corpus, the HMM model is very difficult to implement because all stochastic taggers require large annotated corpus. In this work, the corpus is labeled using BIS tag set, and incorrectly tagged words are corrected applying HMM. Also, non-tagged words were classified correctly. This work first computed an observation likelihood and then choose the hidden sequence based on the highest observation likelihood. HMM is a generative sequence model in which the probability was first decomposed by the Bayesian estimate. Here the initial probability was assigned as given [22]:

$$P(Y) \approx \pi_{i=1}^{l+1} P_T(y_i|y_{i-1})$$

$$P(X|Y) \approx \pi_1^l P_E(X_i|Y_i)$$

5 Results and Discussion

While implementing the SVM and HMM there are a lot of challenges and issues. First of all, taking the case of SVM, encoding of Malayalam was the major trouble we faced while implementing the dataset as CSV and its access in python. But DTM helps in feature extraction very effectively and also it helps in the conversion of text data to numeric data. Dimensionality issue was resolved by both SVM and HMM as it is giving better results for high dimensional data.

The SVM classifier is functioning quite well with both the test set and training set. Initially trained with a test data of 2500 and a test data of around 500 words. It is giving an accuracy of 89%. Later trained using 25,000 words as training data and 1000 as test data which shows 90% accuracy. When the training size was increased as 50,000 then it is showing an accuracy of 91% for a test data of size 2000 (see Table 1 and Fig. 2).

In text, we consider the commonly used performance measure viz., precision, recall, and F-measure. In a multi-class classifier, the performance of binary classifiers over multiple categories has to be averaged as a micro average. The micro average recall is computed as a sum of the overall documents that are correctly accepted and wrongly rejected.

Table 1 SVM accuracy for a given test and train set

S. No.	Train	Test	Accuracy
1.	2500	500	0.894566
2.	25,000	1000	0.902918
3.	25,000	2000	0.896500
4.	50,000	2000	0.915042

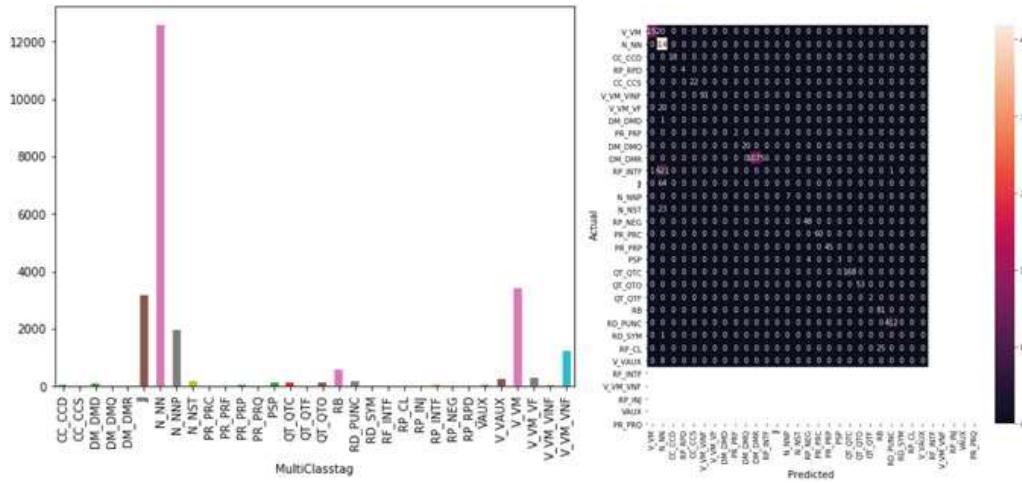


Fig. 2 Multi-class classification of Malayalam news text containing 27,000 words using SVM and respective confusion matrix

$$\widehat{R}_e^U = \frac{\sum_{i=1}^m |\text{TP}_i|}{\sum_{i=1}^m (|\text{TP}_i| + |\text{FN}_i|)}$$

HMM is showing a better classification accuracy. HMM gives good results for newly occurring words. If we increase the size of the annotated corpus it will obviously give good results. It is quite difficult to implement. But for incorrectly classified words it's giving 90% accuracy. HMM is giving better results than SVM in spite of its implementation difficulties. POS with HMM makes use of transition probability and word emission probability. HMM shows an accuracy of 96%. But it has the disadvantage of the three algorithms used. Forward algorithm, Viterbi, and backward algorithm.

6 Conclusion and Future Work

The work shows a comparison of SVM with HMM. Again, the application of HMM reduces the misclassification and ambiguity rate. It showed that HMM is giving better accuracy than SVM when worked with the manually created news corpus inspite of its implementation complexities. The algorithm can be extended by applying deep learning to choosing tags. The effect of manually extracted features in machine learning can be compared with the system extracted features in deep learning and hence the performance and accuracy of the system.

References

1. Menga, J., Lin, H., Yu, Y.: A two-stage feature selection method for text categorization. *Comput. Math. Appl.* **62**(7), 2793–2800 (2011)
2. Wang, T.-Y., Chiang, H.-M.: Fuzzy support vector machine for multi-class text categorization. *Inform. Process. Manag.* **43**, 914–929 (2007)
3. Manjusha, K., Anand Kumar, M., Soman, K.P.: On developing handwritten character image database for Malayalam language script. *Eng. Sci. Technol. Int. J.* **22**(2), 637–645 (2019)
4. Karthik, S., Srikantha Murthy, K.: Deep belief network-based approach to recognize handwritten Kannada characters using distributed average of gradients. *Cluster Comput.* **22**(2), 4673–4681 (2019)
5. El-Sawy, A., Loey, M., Hazem, E.B.: Arabic handwritten characters recognition using convolutional neural network. *WSEAS Trans. Comput. Res.* **5**, 11–19 (2017)
6. Zhang, F., et al.: Construction site accident analysis using text mining and natural language processing techniques. *Autom. Constr.* **99**, 238–248 (2019)
7. Suresh, A., Jha, M.: Automated essay grading using natural language processing and support vector machine. *Int. J. Comput. Technol.* **5**(2) (2018)
8. Kumar, D., Josan, G.: Prediction of part of speech tags for Punjabi using support vector machines. *Int. Arab. J. Inform. Technol. (IAJIT)* **13**(6) (2016)
9. Fernando, S., et al.: Comprehensive part-of-speech tag set and SVM based POS tagger for Sinhala. In: Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing (WSSANLP 2016) (2016)

10. Ghosh, S., Ghosh, S., Das, D.: Part-of-speech tagging of code-mixed social media text. In: Proceedings of the Second Workshop on Computational Approaches to Code Switching (2016)
11. Pandian, S.L., Geetha, T.V.: CRF models for Tamil part of speech tagging and chunking. In: International Conference on Computer Processing of Oriental Languages. Springer, Berlin, Heidelberg (2009)
12. Avinesh, P.V.S., Karthik, G.: Part-of-speech tagging and chunking using conditional random fields and transformation-based learning. In: Shallow Parsing for South Asian Languages, vol. 21 (2007)
13. Leopold, H., et al.: Using hidden Markov models for the accurate linguistic analysis of process model activity labels. *Inform. Syst.* **83**, 30–39 (2019)
14. Khorsheed, M.S.: Diacritizing Arabic text using a single hidden Markov model. *IEEE Access* **6**, 36522–36529 (2018)
15. van der Aa, H., et al.: Transforming unstructured natural language descriptions into measurable process performance indicators using hidden Markov models. *Inform. Syst.* **71**, 27–39 (2017)
16. Paul, A., Purkayastha, B.S., Sarkar, S.: Hidden Markov model based part of speech tagging for Nepali language. In: 2015 International Symposium on Advanced Computing and Communication (ISACC). IEEE (2015)
17. Joshi, N., Darbari, H., Mathur, I.: HMM based POS tagger for Hindi. In: Proceeding of 2013 International Conference on Artificial Intelligence, Soft Computing (AISC-2013) (2013)
18. van Gael, J., Vlachos, A., Ghahramani, Z.: The infinite HMM for unsupervised POS tagging. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, vol. 2. Association for Computational Linguistics (2009)
19. Johnson, M.: Why doesn't EM find good HMM POS-taggers? In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (2007)
20. Devadath, V.V., Dipti Misra, S.: Significance of an accurate sandhi splitter in shallow parsing of dravidian languages. In: Proceedings of the 54th Annual Meeting. Association for Computational Linguistics (2016)
21. Yang, L., Li, C., Ding, Q., Li, L.: Combining lexical and semantic features for short text classification. *Procedia Comput. Sci.* **22**, 78–86 (2013). 17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems—KES 2013
22. Neubig, G.: NLP Programming Tutorial 5—Part of Speech Tagging with Hidden Markov Models (2016)

Comparison of CutShort: A Hybrid Sorting Technique Using MPI and CUDA



Harshit Yadav , Shraddha Naik , B. Ashwath Rao , Shwetha Rai , and Gopalakrishna Kini

Abstract Many sorting algorithms have been developed over the years and the main aim is to reduce the time and space complexity for sorting the worst and average-case scenarios. Parallel computing greatly decreases the processing time and increases the processing speed. In this paper, we compare the results of a hybrid algorithm named CutShort algorithm using a parallel processing framework namely CUDA and MPI. We tested the proposed technique with random samples of large sample data. 30% speedup is achieved with parallel processing as compared to the sequential program.

Keywords CutShort algorithm · Message passing interface · Parallel processing · CUDA

1 Introduction

Applications of sorting are found almost everywhere. Sorting plays a very important role in almost all the algorithms in Computer Science. There are various sorting algorithms developed each serving its own purpose. Sorting helps in increasing the time complexity for searching, Insertion, deletion in many data structures. In this

H. Yadav · S. Naik · B. A. Rao () · S. Rai · G. Kini

Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka 576104, India
e-mail: ashwath.rao.b@gmail.com

H. Yadav

e-mail: harshit591@outlook.com

S. Naik

e-mail: shraddhanaik1995@gmail.com

S. Rai

e-mail: shwetharai.cse@gmail.com

G. Kini

e-mail: ng.kini@manipal.edu

paper, we will talk about the CutShort algorithm which works on the bit count operation. We have been learning from school days that any number having more digits is always greater than a number having less number of digits. The main principle of this algorithm is that two numbers can be compared based on the number of digits. This comparison can be applied to elements in an array, i.e., it helps in sorting an array of elements. CutShort Algorithm is a hybrid algorithm which is a combination of the CutShort with Merge sort or Insertion sort or Quicksort. The word “CutShort” is self-explanatory, i.e., cutting an input array into smaller pieces of the array. These sub-arrays are then fed to Merge sort or quick sort or Insertion sort algorithm. Thus, the time complexity is reduced by a small amount. Message Passing Interface (MPI) and CUDA create a parallel environment for processing the computations in parallel. In MPI, messages are passed among the various processes created during runtime in order to perform the specific task by each process. These frameworks work quite well with sorting techniques like Merge sort as the sorting application sorts the data locally on processes and later passes the data to its neighbors to process the merged lists.

2 Literature Survey

While doing a literature survey, we went through various algorithms like Quicksort, Merge Sort [1], Insertion sort [2], and the time complexities of each one of them. The traditional Insertion Sorting algorithm has worst-case and average-case complexity of $O(n * n)$ [3]. When this algorithm is used with CutShort Algorithm the time complexity is reduced to $O(n \log n)$ in average and worst-case time complexity.

Further studying the parallel computing framework using MPI, we proposed a model that would combine the CutShort Algorithm with MPI and CUDA framework, respectively, which would greatly improve the efficiency and achieve scale-up.

3 Methodology

In this section, we will discuss in detail the working of CutShort algorithm. We will see in detail the proposed framework of using CutShort Algorithm using MPI or CUDA and observe how the speedup can be achieved in a parallel environment.

3.1 CutShort Algorithm

CutShort algorithm [4] works on four steps as mentioned below.

1. Initial step

2. Range defining step
 3. Rearranging
 4. Sub-array sorting
- (i) Initial step

In the following step, we count the no. of bits of a digit in its binary equivalent form. An array of decimal numbers is given as the input. These decimal values are converted into their binary equivalent and their count is stored in an array named Bitband array. With the help of this operation, we get to know the count of values in the input array having the same no. of binary digits used to represent them.

Consider the input array shown in Table 1.

Calculation of the BitCount operation is as shown in Table 2.

In the above example we have one integer with 3 bits, i.e., (5), two integers with 5 bits, i.e., (22, 18), five integers with 6 bits, i.e., (40, 60, 52, 58, 46) and 2 integers with 7 bits, i.e., (78, 82).

So, the BitBand Array will be formed as shown by the author in Table 3:

Table 1 Input array

0	1	2	3	4	5	6	7	8	9
22	40	60	52	78	58	18	7	46	82

Table 2 Output of BitCount operation

Number	Binary representation	O/P of BitCount operation
22	10110	5
40	101000	6
60	111100	6
52	110100	6
78	1001110	7
58	111010	6
18	10010	5
7	111	3
46	101110	6
82	1010010	7

Table 3 Bitband array

0	1	2	3
3	5	6	7

Table 4 Resultant Bitband for ranges

[0,3)	[3,5)	[5,6)	[6,7)
3	5	6	7

(ii) Range defining step

In this step, the range defines the upper and lower boundary of sub-array having only the elements that are having the equal bit count. It is calculated using the following procedure.

The ranges of these sub-arrays can be written as follows: Element 7 will have range from [0, 3), Elements {0, 6} have range from [3, 5), Elements {1, 2, 3, 5, 8} have range from [5, 6), Elements {4,9} have range from [6, 7). The range value is calculated using the method given above and is stored in the BitBand Array as shown in Table 4.

(iii) Rearranging

In this step, the elements of array are rearranged according to the no. of bit count values. Integers with same bit count are placed together with one after the other in a sequential manner in an array. After this step, the input array would look as given in Table 5.

With the help of this step, the integers with higher bit count are present on one side of the array while the integers with less bit count values are present on the other side of the array. In this step, the integers with the same bit count values are not sorted. After the final step of Sub-array Sorting, the entire input array will be sorted.

(iv) Sub-array Sorting

Each of these sub-arrays can be sorted using any sorting technique by using the Bitband array to define the different range of bits present in the input array. Merge Sort, Quick Sort or Insertion Sort can be used for sorting [5] with this BitBand Array.

After this step, we get the sorted input array as shown in Table 6.

Table 5 Resultant input array after rearranging step

0	1	2	3	4	5	6	7	8	9
7	22	18	40	60	52	58	46	78	82

Table 6 Sorted input array

0	1	2	3	4	5	6	7	8	9
7	18	22	40	46	52	58	60	78	82

A. Proposed Parallel execution Model (CutShort Algorithm with MPI)

In the proposed MPI implementation of the CutShort algorithm, we parallelize the sequential algorithm [6] using the Message Passing Interface which is a framework that creates an environment for parallel programming by providing libraries [7] in C/C++ programming. We considered MPI to be our one of communication interfaces as it can also be used on a variety of hardware from a large-scale supercomputer to a single compute node for inter process-communication in a desktop setup for performance reasons [8].

To execute the CutShort parallelly we divide the total input array into sub-arrays of equal sizes based upon the number of processors passed as an argument during the runtime in the MPI, then BitCount function (Initial step of CutShort Algorithm) is performed on every value stored at index of the sub-array in parallel by each processing element and the BitCount value is stored in separate bitmap array for each process which is then collected together in the root process and combined together to obtain the final bitmap array for the given input array the obtained bitmap array is used for rearranging step(step ii of the CutShort Algorithm) serially and resultant can be sorted with any sequential or parallel sorting algorithm of choice.

B. Proposed Parallel execution Model (CutShort Algorithm with CUDA)

In the proposed CUDA (CUDA is a parallel computing platform and API Model created by Nvidia [9]) implementation of the CutShort algorithm, we parallelize the sequential algorithm by sending the input array to the kernel where each thread process the BitCount function (Initial step of CutShort Algorithm) on its respective thread element in parallel and the result is stored in separate bitmap array in global memory which is shared among all the kernel and contain bitmap value obtained from all the parallel execution, the final bitmap array obtained after completion of all threads is processed sequentially in rearranging step of the CutShort algorithm and its result can be sorted with any sequential or parallel sorting algorithm of choice.

4 Experiment

For the Experimental setup for the proposed model, a Linux machine was used having Intel i5-3210 M 2.5 GHz processor with 6 GB of 1600 MHz DDR3 RAM and Ubuntu operating system.

For the tests, a sample space of ten thousand elements was taken which was timed by running sequentially as shown in Table 7 which was divided into three categories horizontally for effective analysis of the algorithm in all cases.

A. Worst Case [test case 1–3]

All elements of sample space are in range from 2^i to 2^{i+1}

Table 7 Time taken by different serial algorithms (in s)

Test case	Quicksort + CutShort	Insertion sort + CutShort	Merge sort + CutShort
1	0.004342	0.035820	0.006833
2	0.004319	0.033692	0.006644
3	0.004367	0.033484	0.006166
4	0.004256	0.028709	0.005831
5	0.004086	0.031815	0.006000
6	0.004119	0.029502	0.005073
7	0.003823	0.026361	0.005370
8	0.003522	0.028389	0.005399
9	0.003773	0.027373	0.005233

- B. Random Values [test case 4–6]
Containing all randomly chosen numbers
- C. Best Case [test case 7–9].

All the elements can be equally divided into equal bit range buckets.

Now for parallel tests, the sample space of ten thousand elements was taken which was divided into three categories as mentioned with the use of 4 processors [10] in MPI for the experiment.

5 Results and Analysis

By comparing the timing observed as seen in Table 8 and Table 9 with the sequential CutShort algorithm time readings as shown in Table 7 the performance improvement in the timing of CutShort algorithm in parallel [11] using CUDA or MPI individually in the best test case values is shown in graphical form in Fig. 1 respectively.

Table 8 Time taken by different parallel algorithms in MPI (in s)

Test case	Quicksort + CutShort	Insertion sort + CutShort	Merge sort + CutShort
1	0.002298	0.027697	0.003462
2	0.002864	0.027697	0.003855
3	0.002310	0.025781	0.003534
4	0.002770	0.022981	0.003526
5	0.002460	0.025772	0.004421
6	0.002301	0.024669	0.003821
7	0.001903	0.024542	0.003662
8	0.002016	0.023974	0.003214
9	0.001900	0.024508	0.003343

Table 9 Time taken by different parallel algorithms in CUDA (in s)

Test case	Quicksort + CutShort	Insertion sort + CutShort	Merge sort + CutShort
1	0.001706	0.016513	0.003116
2	0.002139	0.016508	0.003144
3	0.001684	0.016697	0.003272
4	0.002232	0.015384	0.003096
5	0.001546	0.018439	0.003192
6	0.001604	0.017452	0.003074
7	0.001108	0.015622	0.002464
8	0.001832	0.014504	0.002258
9	0.001296	0.014912	0.002402

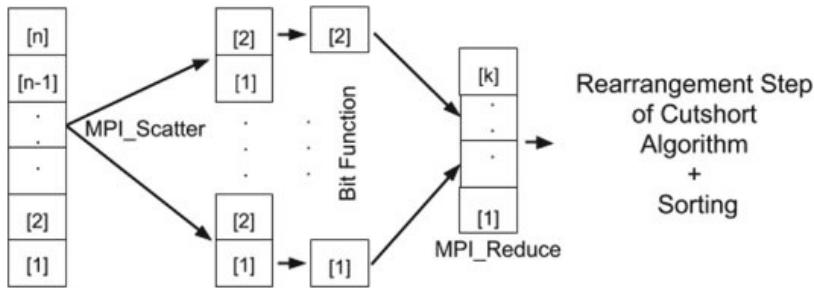


Fig. 1 Parallel execution of CutShort using MPI

6 Conclusion

In this paper, we proposed a parallel implementation of the CutShort Algorithm. Parallel processing increases the processing speed of the algorithm [12] which can be concluded from the various test case we tested using different parallel computing methods.

We achieved a speedup of greater than 30% as shown in Fig. 2 by implementing it parallelly [13] in CUDA or MPI individually as compared to running the algorithm sequentially with the same data set.

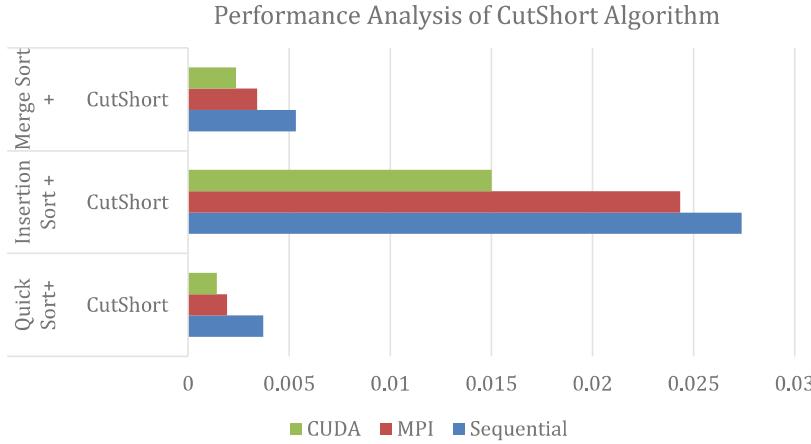


Fig. 2 Sequential and parallel execution time of algorithms in best case in MPI and CUDA (in s)

References

1. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: *Introduction to Algorithms*, 3rd edn. MIT Press (2009)
2. Traditional insertion algorithm—[en.wikipedia/wiki/Insertion_sort](https://en.wikipedia.org/wiki/Insertion_sort). Three lines implementation and five-lines optimized version by Bentley, J.: *Programming Pearls*. Addison-Wesley Professional (1999)
3. Bender, M.A., Farach-Colton, M., Mosteiro, M.: Insertion sort is $O(n \log n)$. SUNYSB (2006). <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.60.3758>
4. Garg, A., Goswami, S., Garg, V.: CutShort: a hybrid sorting technique. In: 2016 International Conference on Computing, Communication and Automation (ICCCA) (2016). Available <https://doi.org/10.1109/ccaa.2016.7813705>
5. Sorting Algorithm. En.wikipedia.org (2020). [Online]. Available https://en.wikipedia.org/wiki/Sorting_algorithm. Accessed 19 Jan 2020
6. Kirk, D., Hwu, W.: *Programming Massively Parallel Processors*, 3rd edn. (2016)
7. Using MPI: Portable parallel programming with the message-passing interface. Comput. Math. Appl. **40**(2–3), 419 (2000). [https://doi.org/10.1016/s0898-1221\(00\)90207-4](https://doi.org/10.1016/s0898-1221(00)90207-4)
8. Lee, R.C.T., Tseng, S.S., Chang, R.C., Tsai, Y.T.: *Introduction to the Design and Analysis of Algorithms*. Mc Graw Hill (2005)
9. Kazennov, A.: Basic concepts of CUDA technology. Comput. Res. Model. **2**(3), 295–308 (2010). <https://doi.org/10.20537/2076-7633-2010-2-3-295-308>
10. Gropp, W., Lusk, E., Skjellum, A.: *Using MPI*. TheMIT Press, Cambridge, Massachusetts (2014)
11. Shen, Z., Song, J., Zhuang, W.: Speedup improvement on general connectivity computation by algorithmic techniques and parallel processing. In: Proceedings High Performance Computing on the Information Superhighway. HPC Asia ‘97 (2016). Available <https://doi.org/10.1109/hpc.1997.592241>
12. Rastogi, S., Zaheer, H.: Significance of parallel computation over serial computation. In: 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT) (2016). Available <https://doi.org/10.1109/iceeot.2016.7755106>
13. Sun, X., Ni, L.: Another view on parallel speedup. In: Proceedings Supercomputing ‘90 (2016). Available <https://doi.org/10.1109/superc.1990.130037>

Self-Learning and Self-Organizing Log Files by Generating Recursive Associations



K. Indra Gandhi and A. Balaji

Abstract The main aim of the system log is to track the events that occur at any given point of time so as to analyze if any problem exists in the system. Although many automated testing tools are available, other forms of testing are required in order to improve the accuracy of the result continuously. In this work, we have proposed a level-based categorizer by applying self-learning techniques to process the log data by parsing, preprocessing the initial data, and categorizing the logs to ease the monitoring process. The features are extracted and the model is trained such that the model can be used for learning dynamically and determining the events that trigger the consecutive log generation. This is performed in a recursive fashion until the logs can be fine-tuned to understand the causal effect. Finally, the impact of the metrics that influences the log events are verified on the different log data set. The process can be streamlined by learning the pattern of data continuously and finally modeling the data using multinomial Naive Bayes classification so as to pay more attention to the precise set of information.

Keywords Multinomial Naive Bayes algorithm · Log files · Classification · Causal effect · Self-learning

1 Introduction

The increase in the effective usage of computer systems has paved the way for the generation of enormous logs. These log information provides necessary details which are of major use for managing a larger group of systems. This is because the log information may not be able to identify a sudden variation point especially if the

K. I. Gandhi (✉) · A. Balaji

Department of Information Science and Technology, Anna University, Chennai 600025, TamilNadu, India

e-mail: jeevaindra@gmail.com

A. Balaji

e-mail: balaji.foo404@gmail.com

end-users are a homogenous group. The classification of the logs may provide greater insight into the data that is collected continuously. There are various classification techniques. The voluminous data, the types of log, the change of format, and the inference made makes the system even more complex since a common classification and the corresponding inference may not provide a generalized view of the system [1]. The detection of failure may not correlate with the same set of events that have been classified for the logs taken into consideration.

For error detection or failure prediction, various forms of classification, and their approach towards identifying the errors or failures have been determined [2, 3]. However, these set of classifications may vary depending upon the logs, the environment within which the logs have been collected, the usage of the systems whether continuous or discrete may have an impact on the logs that have been collected. Our approach is a novel approach where we try to cluster the group automatically by learning the pattern and their association with the other information available. This is a continuous process where the identification of the groups and their correlation may vary depending upon the timestamp and the event that is being processed at that particular point.

System logs generate voluminous data that can be effectively handled by continuously learning the pattern in which the logs are repeated. The logs generated may have repetitive data that are preprocessed and the duplication is eliminated. In this work, the log files are collected from the specialized labs and the preprocessing of the voluminous information is done by identifying the stop words. At the outset, for the purpose of initial classification, we create a vocabulary out of all commonly occurring events. This vocabulary helps in filtering out records that are prone to repetition. This technique actually minimizes the record size without deviating from actual information to be observed. To assign a category to those reduced record set, frequency of each token is found and its probabilities are compared to get frequently recurring tokens.

The logs are collected from specialized labs and the cluster of events that have been happening are analyzed. The relevant features are identified and the categories are determined accordingly. The occurrence of these categories across timestamps is examined so as to observe the change in the group of categories determined. The system is modeled and trained using multinomial Naive Bayesian classifier. The classified events and their relationship with the other sub-events are generated as a tree so as to access the events associated with a particular log efficiently.

2 Related Work

The log analysis has been evolving continuously since the networking components have been emerging with the addition of mobile computing, cloud computing, edge computing, etc. For a larger network, the amount of log information is complex, and also there may be a repetition of the same information being stored continuously. The log information consists of various types which need to be identified, classified,

and detected effectively. The classification of logs and the accuracy of determining the root cause of the fault has been done through fault keyword matrix generation [4].

The challenges in addressing the detection of attacks in major security operations Centre has been investigated [5]. The log data have been trained using machine learning algorithms after which the attacks have been classified. Due to the increase in the volume of data especially in the cloud environment, an automatic log classification system to improve the classification by applying deep convolutional neural networks has been developed [6]. The root cause analysis for the agile software testing environment has been tested for analyzing the performance of the system. Kahles et al. [7] explored the issues involved in identifying the relevant features, categorizing the root cause in software failures and the learning period required for training the raw log data.

Du et al. [8] have proposed a DeepLog verification system that learns the pattern of data, model the pattern, and any deviation in the pattern lead to the detection of anomalies. Boutaba et al. [9] have provided an extensive survey on how machine learning techniques can be applied to the various sub-domains of network. Log files provide information regarding the actions that have been performed. Analyzing logs based on data mining techniques enables the processing of a large amount of data with minimum error rate [10, 11]. Nagaraj et al. [12] explored the system behavior through the logs and the association between the performance components which are presented to the developers. This may help in improving the stability of the system by understanding the components that impact the performance.

Landauer et al. [13] formulated a cluster evolution technique based on the timestamp of the events. The influence of parameters has been studied by developing self-learning algorithms that help in the evolution of clusters. Seyyar et al. [14] have detected the web vulnerability scans by analyzing the log files by applying rule-based methodology. Their investigation also revealed that static rules can detect these scans at a very high accuracy rate. A binary-based approach for computing the frequency of the information in the log files have been proposed [15]. The database log consists of the user-based queries as table entries. The log analysis is an evolving system where the continuous monitoring of events is a challenging process.

3 Problem Description

The system log information is collected for further classification of the logs based on multinomial Naive Bayes classifier by extracting the major features from the log information. The major categories are classified at the first run of the algorithm. We propose a novel method to generate decision trees in order to identify the causal-relationship across the various subcategories that are identified within a category. This provides an insight into a recursive relationship such that the root cause of the log generation can be matched appropriately. Apart from finding the root cause analysis of the information which has been extensively explored in the literature,

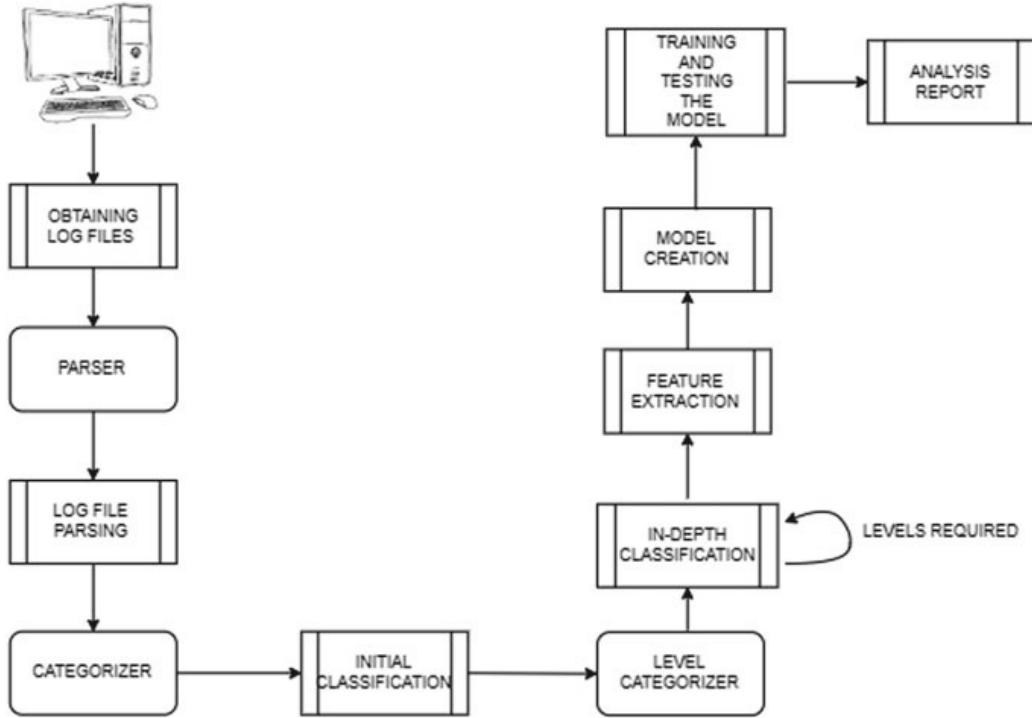


Fig. 1 Flow analysis of the log information

this work focuses on examining these categories as a cluster of information. The lifetime of these clusters is observed so as to understand the pattern in which the log information is generated. Further from the lifetime of the cluster, the shifting of information across various categories is also analyzed so as to develop a system that provides information relevant to the current timestamp.

The generation of events and their association is illustrated in Fig. 1. At the initial set, the log files are real-time datasets collected from the specialized programming labs preprocessed for further analysis. The categories are extracted through recursive classification and the model is generated.

Therefore, Automatic classification of the categories and their corresponding sub-categories helps in associating the links that have to be tracked for the root cause of the generation. The sample of log messages are shown in Fig. 2.

3.1 Parsing Phase

The system logs are parsed with reference to the common tag that occurs for a given time period. Initially, the model is trained by computing the term frequency of the logs. The categorization is done in an iterative fashion by updating from the logs

```

Jun 15 04:06:20 combo logrotate: ALERT exited abnormally with [1]
Jun 15 04:12:42 combo su(pam_unix)[22644]: session opened for user news by (uid=0)
Jun 15 04:12:43 combo su(pam_unix)[22644]: session closed for user news
Jun 15 12:12:34 combo sshd(pam_unix)[23397]: check pass; user unknown
Jun 15 12:12:34 combo sshd(pam_unix)[23397]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=218.188.2.4
Jun 15 12:12:34 combo sshd(pam_unix)[23395]: check pass; user unknown
Jun 15 12:12:34 combo sshd(pam_unix)[23395]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh
ruser= rhost=218.188.2.4

```

Fig. 2 Sample event generation

that are being generated. A dictionary is created that stores the categories and their corresponding sub-categories. The pseudo-code for the training model is specified below.

Pseudocode 1: Dictionary Creation

```

Step 1. Retrieve all log files from dataset directory (algorithm: 'glob ()')
Step 2. for each file in files
        open each file under UTF-encoding (algorithm: 'open ()')
        read every line of each file (algorithm:'readlines ()')
Step 3. for each line in lines
        Tokenize them into different fields (timestamp, process, event, category, category id)
Step 4. Write dictionary of {file, data frame}
Step 5. Create a data frame out of dictionary created earlier(algorithm:'pandas.DataFrame()')

```

The log files are preprocessed and duplicates are removed so as to precise the categorization. The information as per the requirement can be extracted from the log files.

3.2 Initial Categorization

Since the logs have been parsed, the event fields of all records are accumulated. The normalization of raw logs is processed by performing the following pseudo-code:

Pseudocode 2: Label Formulation

```

Step 1. Replace all non-characters [symbols] (algorithm:'re.sub()')
Step 2. Replace all punctuation (algorithm:'string.replace()')
Step 3. Remove stop words specific to English language (algorithm: 'set(stopwords.words('english'))')
Step 4. Assign categories to those records that contain most frequently repeated tokens
Step 5. Remove duplicates in event field
Step 6. Group by category for temporary usage

```

For the purpose of initial classification, we create a vocabulary out of all commonly occurring anomaly. This vocabulary helps in filtering out records that are prone to anomalies. To assign a category to those reduced record sets, frequency of each token is found and its probabilities are compared to get frequently recurring tokens. With this category list, categorization of reduced record set is done. This procedure is performed in a recursive fashion obtaining another level of categorization which is termed as the sub-category in order to understand the cause of these related logs.

3.3 Feature Generation

The features are extracted based on the term frequency (tf) and inverse document frequency. Word count that counts the occurrences of each token is referred to as term frequency. The inverse document frequency is meant for diminishing the weight of terms that occur very frequently in the document set and increases the weight of term that occurs rarely. The product of term frequency and inverse document frequency (idf) is used to emphasize the importance of a keyword or phrase within dataset. As the features constitute multiple attributes, a feature matrix is generated with the list of attributes for the corresponding log records. To reflect the nature of keywords that is related to anomaly we use a feature called *N*-gram which is a sequence of n items combined from given record text. The pseudocode for feature generation is defined as below:

Pseudocode 3: Feature Extraction

```

Step 1. Find term frequency and inverse document frequency
Step 2. Fit it and transform the event field according to tf-idf score
Step 3. Fetch labels for category id
Step 4. For each element Category dictionary generated in previous phase
Step 5. do
    Generate features for it (algorithm: 'chi2()')
    Recognize features as N-grams (N=2)
    Generate Unigram and Bigram out of it
end do

```

3.4 Model Creation and Training

The model is created by applying Multinomial Naïve Bayes classification so that multiple categories can be generated based upon the constraints specified. The NB models cannot represent complex behavior since the classes grow as the events are generated dynamically. The dataset is split for the purpose of training and testing. At the initial stage, the training set is generated as vectors. The training set is fit into the model and formulated as per the categories and their sub-categories. After labeling

the categories we train the model as that fits into the tf-idf score. The corresponding pseudocode is specified as below.

Pseudocode 4: Model Formulation

- Step 1. Split the obtained dataset for Training and Testing purpose
- Step 2. Fit and transform the training set according to Count Vectorizer
- Step 3. Train the partitioned Trained dataset with tf-idf Transformer
- Step 4. Create a Multinomial Naïve Bayes model
- Step 5. Fit the tf-df score with Multinomial Naïve Bayes model

4 Results and Analysis

The model for analyzing log events is developed using Multinomial Naïve Bayes model. The dataset is split for training and testing. The multinomial distribution uses the tf-idf count for building the model. The training process includes the computation of prior probabilities of a particular category in the syslog dataset.

$$P(\text{category}) = \frac{\text{Number of events classified into that category}}{\text{Total number of events}} \quad (1)$$

The conditional probability for calculating the subcategory is the probability of the events that occur in a particular category. By computing the categories and subcategories, an n -ary tree is generated which is illustrated in Fig. 3.

The initial categorization comprises *device state change*, *error*, *host name conflict*, *server misbehaving*, and *warning*. These categories are further sub-classified based upon the events that are generated with reference to the super-category.

The training model for categorizing the log events using Multinomial Bayesian method has been generated with the log events collected. The model was built with the following events collected from specialized programming labs. After identifying the categories, the associated subcategories are generated. The sample of categorization that has been achieved for the logs has been illustrated in Table 1. In a similar fashion, the enumerated log files have been split into a 60% training and a 40% testing dataset.

The system logs have been generated from various system labs so as to ensure the validity of the classification. Figures 4 and 5 demonstrate the classification of logs and their consecutive categorization of events. The category with the higher count

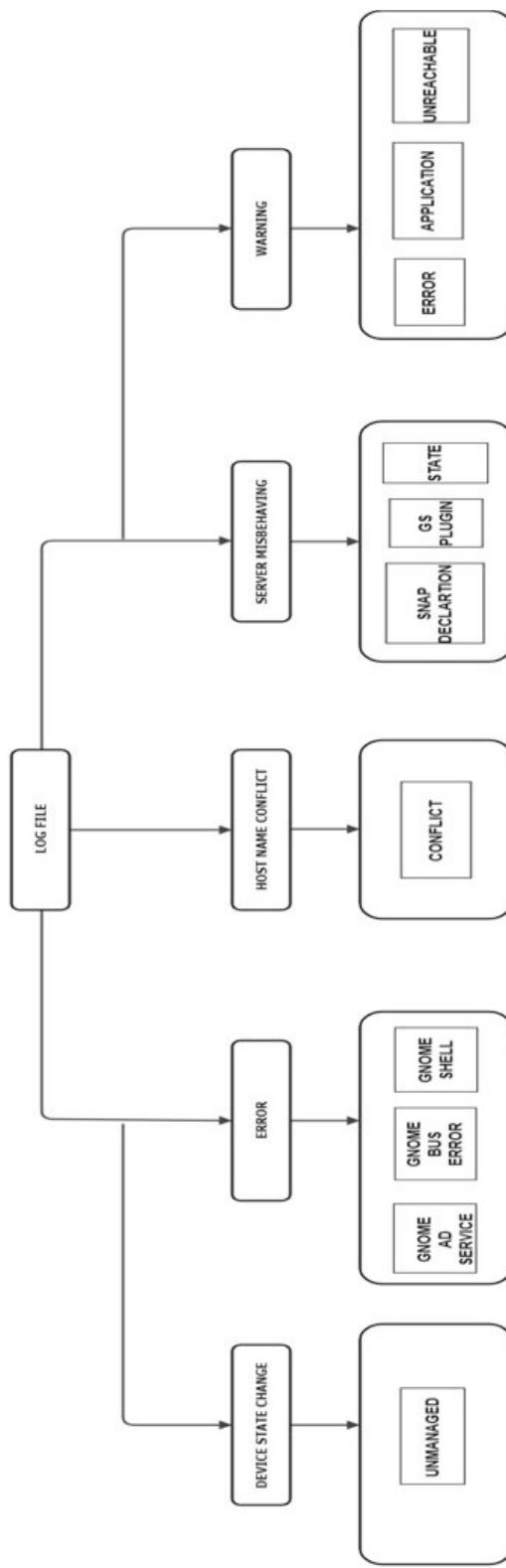


Fig. 3 Association generation

Table 1 Log files collection

Name of file	Category name	Number of logs	Size (KB)
Log 5.txt (syslog)	Total logs	4600	110,400
	Error	39	89
	Server misbehaving	23	56
	Hostname conflict	10	26
	Warning	25	59
	Device state change	8	12
Log6.txt (syslog 1)	Total logs	2800	76,400
	Error	30	80
	Server misbehaving	18	50
	Hostname conflict	6	16
	Warning	22	50
	Device state change	1	2
sample.txt (HP 15q syslog)	Total logs	3300	98,500
	Error	16	22
	Server misbehaving	8	15
	Hostname conflict	17	26
	Warning	8	10
	Device state change	12	24
Total			285,300

implies a greater number of sub-categories. The sub-categorization is illustrated in Fig. 5.

The log events classified based on the categories identified are captured for a certain period of time which is shown in Fig. 6. By applying multinomial Naive Bayesian algorithm, the categories can be generated by continuously learning from the events that are being generated. These categories with a higher count of events provide an insight into the subcategories that need to be analyzed. This is because the higher number of categories imply a higher number of subcategories to be taken into consideration. These subcategories may change as per the user activities and

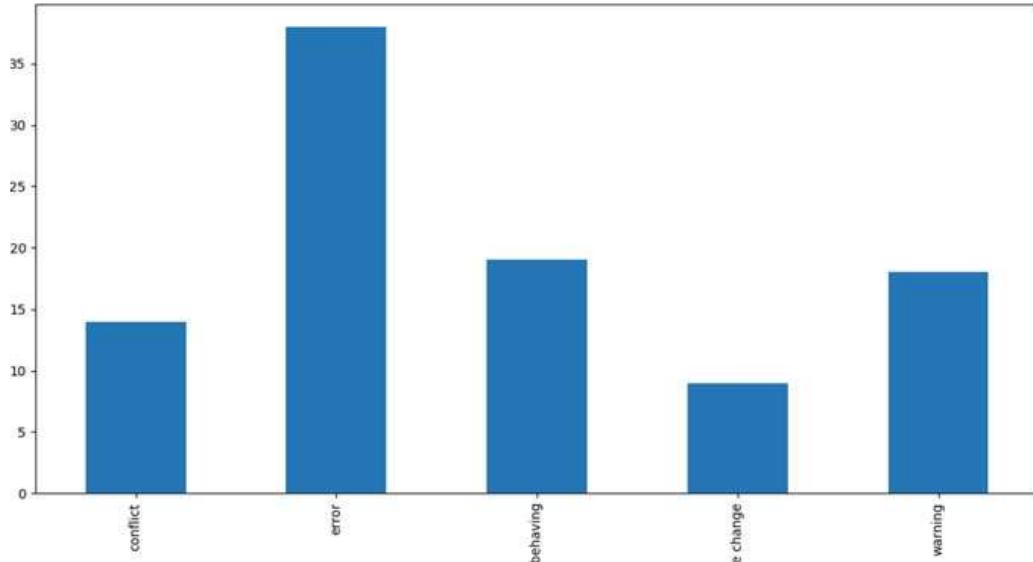


Fig. 4 Classification of logs

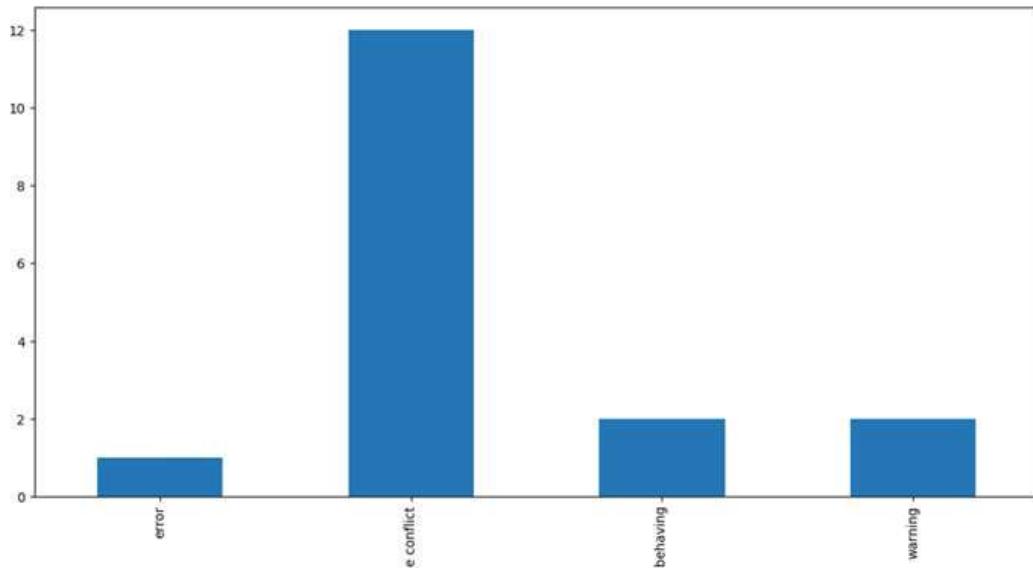


Fig. 5 Sub-classification of log files

hence the change in the subcategories and their relevance can vary depending upon the events being generated.

Irrespective of the voluminous logs that are being generated, the association between the categories and their subcategories which has been generated as a tree structure provides more flexibility in order to trace the specific cause of the event.

In order to test the efficiency of the model, the system logs were collected from a completely different log file set, an accuracy of 96% was observed on the trained dataset (Fig. 7).

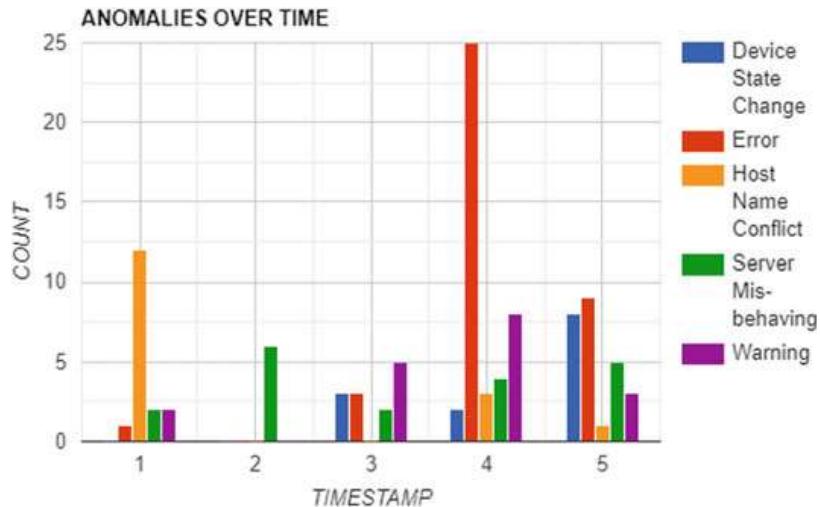


Fig. 6 Sampling of various Log files

Accuracy score : 0.96					
	precision	recall	f1-score	support	
device state change	1.00	1.00	1.00	2	
error	1.00	0.90	0.95	10	
host name conflict	1.00	1.00	1.00	1	
server misbehaving	1.00	1.00	1.00	7	
warning	0.83	1.00	0.91	5	
accuracy			0.96	25	
macro avg	0.97	0.98	0.97	25	
weighted avg	0.97	0.96	0.96	25	

Fig. 7 Score generation

The model was effective to handle the classification of the logs as well as to generate the associated subcategories. This implies that the self-learning techniques applied to the voluminous logs create the categorization of logs such that the root cause of a particular log can be identified precisely. This shows that extensive self-learning techniques can be applied to the logs such that the classification learns the pattern in which the subcategories are generated.

5 Conclusion

The occurrence of any event on the network has to be recorded so as to ensure transparency and security in the network. The tracking of these events is a complex task since the collection of data increases across time. The logging information process also becomes time-consuming since the details may be a periodic repetition of the same data. The events generated are classified and a recursive sub-classification is done vigorously so as to obtain the precise set of subcategories at various levels. The model is trained and tested which shows an accuracy of 96% fit. Since the associations can be generated depending upon the log files, the accuracy improves as the sub-classes grow. The self-learning structure of the log files will help in analyzing the precise generation of sequence of logs and their respective associations.

References

1. Fu, X., Ren, R., McKee, S.A., Zhan, J., Sun, N.: Digging deeper into cluster system logs for failure prediction and root cause diagnosis. In: 2014 IEEE International Conference on Cluster Computing (CLUSTER), Madrid, pp. 103–112 (2014). <https://doi.org/10.1109/cluster.2014.6968768>
2. Salfner, F., Tscharpke, S.: Error log processing for accurate failure prediction. In: USENIX conference on Analysis of system logs (WASL'08). USENIX Association, USA, p. 4 (2008)
3. Chuah, E., et al.: Diagnosing the root-causes of failures from cluster log files. In: 2010 International Conference on High Performance Computing, Dona Paula, pp. 1–10 (2010). <https://doi.org/10.1109/hipc.2010.5713159>
4. Zou, D., Qin, H., Jin, H., Qiang, W., Han, Z., et al.: Improving log-based fault diagnosis by log classification. In: 11th IFIP International Conference on Network and Parallel Computing (NPC), Sep 2014, Ilan, Taiwan, pp. 446–458 (2014). https://doi.org/10.1007/978-3-662-44917-2_37.hal-01403114
5. Vásquez, V., Giancarlo, E.: Classification of Logs Using Machine Learning Technique. M.S. Thesis. Norwegian University of Science & Technology (2018)
6. Ren, R., et al.: Deep convolutional neural networks for log event classification on distributed cluster systems. In: 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, pp. 1639–1646 (2018). <https://doi.org/10.1109/bigdata.2018.862261>
7. Kahles, J., Törrönen, J., Huuhtanen, T., Jung, A.: Automating root cause analysis via machine learning in agile software testing environments. In: 2019 12th IEEE Conference on Software Testing, Validation and Verification (ICST), Xi'an, China, pp. 379–390 (2019). <https://doi.org/10.1109/icst.2019.00047>
8. Du, M., Li, F., Zheng, G., Srikumar, V.: DeepLog: anomaly detection and diagnosis from system logs through deep learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17), pp. 1285–1298. ACM, New York, NY, USA (2017). <https://doi.org/10.1145/3133956.3134015>
9. Boutaba, R., Salahuddin, M.A., Limam, N., et al.: A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *J. Internet Serv. Appl.* **9**, 16 (2018). <https://doi.org/10.1186/s13174-018-0087-2>
10. Breier, J., Branišová, J.: Anomaly detection from log files using data mining techniques. *Inf. Sci. Appl.* **339**, 449–457 (2015)
11. Breier, J., Branišová, J.: A dynamic rule creation based anomaly detection method for identifying security breaches in log records. *J. Wirel. Pers. Commun.* **94**, 497 (2017). <https://doi.org/10.1007/s11277-015-3128-1>

12. Nagaraj, K., Killian, C., Neville, J.: Structured comparative analysis of systems logs to diagnose performance problems. In: Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation (NSDI'12). USENIX Association, Berkeley, CA, USA, p. 26 (2012)
13. Landauer, M., Wurzenberger, M., Skopik, F., Settanni, G., Filzmoser, P.: Dynamic log file analysis: an unsupervised cluster evolution approach for anomaly detection. *Comput. Sec.* **79**, 94–116 (2018). ISSN 0167-4048. <https://doi.org/10.1016/j.cose.2018.08.009>
14. Seyyar, M.B., Çatak, F.O., Gül, E.: Detection of attack-targeted scans from the Apache HTTP server access logs. *Appl. Comput. Inform.* **14**(1), 28–36 (2018). ISSN 2210-8327
15. Fageeri, S.O., Ahmad, R.: An efficient log file analysis algorithm using binary-based data structure. *Procedia Soc. Behav. Sci.* **129**, 518–526 (2014). ISSN 1877-0428

Analysis and Prediction of Fantasy Cricket Contest Winners Using Machine Learning Techniques



K. Karthik, Gokul S. Krishnan, Shashank Shetty, Sanjay S. Bankapur,
Ranjit P. Kolkar, T. S. Ashwin, and Manjunath K. Vanahalli

Abstract Cricket is one of the well-known sports across the world. The increasing interest of cricket in recent years resulted in different forms like T20, T10 from test and one day format. The craze of all these formats of cricket matches today has come into online fantasy cricket league games. Dream11 is one such app that is most popular in this context, along with many similar apps. Creating a dream team of 11 players from playing 11 of both teams involves skills, ideas and luck. Predicting a winner among all the joined contestants based on the previous historical data is a challenging task. In this paper, we used a feed-forward deep neural network (DNN) classifier for predicting the winning contestant for the top three positions in a fantasy league cricket contest. The performance of the DNN approach was compared against that of state-of-the-art machine learning approaches like k-nearest neighbours (KNN), logistic regression (LR), Naive Bayes (NB), random forest (RF), support vector machines (SVM) and in predicting the fantasy cricket contest winners. Among the methods used, DNN showed the best results for all three positions, showing its consistency in predicting the winners and outperforms the state-of-the-art machine learning classifiers by 13%, 8% and 9%, respectively, for first, second and third winning positions, respectively.

Keywords Cricket data analysis · Data mining · Fantasy game · Indian premier league · Performance measurement · Prediction

K. Karthik (✉) · G. S. Krishnan · S. Shetty · S. S. Bankapur · R. P. Kolkar · T. S. Ashwin · M. K. Vanahalli

Department of Information Technology, National Institute of Technology Karnataka
Surathkal, Mangaluru 575025, India
e-mail: 2karthik.bhat@gmail.com

S. Shetty
Department of Computer Science and Engineering, NMAM Institute of Technology, Nitte,
Karnataka 574110, India

1 Introduction

Cricket¹ can be defined as a bat-and-ball game played between two teams, each including 11 players. The main aim of the game is to score runs during batting and to dismiss the batsmen one at a time, by the opposite team during fielding and thus restricting the runs scored by the batting team in an innings. Batting and bowling are the crucial factors affecting the performance of a team to its success. Apart from this, there are other factors like fielding, captaincy skills, home ground advantage, etc. [6]. Based on the duration of playing, Twenty20 attained more attraction and commercial success compared to other formats that lead to the launch of most popular tournaments like *World Championship*,² *Big Bash*³ and *Indian Premier League (IPL)*.⁴

With cricket being the prominent sport in India, it was never surprising that it became the first online fantasy game. Initially, Dream11 was one of the popular fantasy leagues which grew up, and today there are many other fantasy cricket league games like Howzat, Myteam11, Fanfight, and so on. Dream11⁵ is one of India's latest unicorn, a term used for a privately held start-up company valued over \$1 billion. The app has over 52.5 million registered users and expected to reach 100 million by the end of 2020, making it India's largest fantasy cricket platform, in terms of total subscribers.

Even though the fantasy game seems to be gambling, it is the game of strategies; because a fantasy player needs to play a combined role of a coach, team owner/manager and statistician. An efficient team needs to be selected by the fantasy cricket contestant within the fixed credits. Fantasy contestants cannot recruit all the best players within the limited credits, but intelligent picking needs to be done based on the player's past and current performance. Hence, we can adapt this to artificial intelligence(AI) and can be modelled using machine learning. Obtaining the best result every day is indeed tricky due to the unpredictable conditions involved in the outcome of the game. Supposably in an innings, the last batsman of the team top scores or a bowler who is not expected to take a wicket gets five-wicket haul, or a batsman with perfect technique might struggle during a particular match proves the unlikely situation of the game. Lots of extrinsic factors such as weather conditions, home pitch conditions, the spell of rain, a sudden gush of wind and crowd behaviour have a high impact on the game. Hence, these uncertainties and an intricacy involved in a player selection make fantasy cricket game challenging [8].

Machine learning (ML) is one of the intelligent techniques that has shown promising outcomes in the field of classification and prediction. The main objective of classification is to build the classification model and to predict the target variable (class) from the test data. The model to classify and predict the future outcomes in cricket,

¹<https://www.cricket-rules.com>.

²https://en.wikipedia.org/wiki/World_championship.

³https://en.wikipedia.org/wiki/Big_Bash_League.

⁴https://en.wikipedia.org/wiki/Indian_Premier_League.

⁵<https://timesofindia.indiatimes.com/companies/the-rise-rise-of-dream11-and-fantasy-sports-gaming-in-india/articleshow/68543816.cms>.

it uses the data of the previous historical games, performance of the players, opposition team details, etc. This paper focuses on providing a critical analysis of fantasy cricket winners by leveraging ML and comparing the results with the traditional ML techniques. In doing so, we identify the winning contestants in the Dream11 competition.

The rest of this paper is arranged as follows: In Sect. 2, a brief review of different performance-related works on cricket is briefed out. In Sect. 3, the materials and methods used in this work are presented. The experimental results are discussed in Sect. 4, followed by a conclusion and future work.

2 Literature Review

There are many approaches to cricket that were addressed by researchers with different formats of cricket at various levels. The current literature review focuses on the works that are related to the prediction of player performance, including good team selection. A J Lewis [7] proposed the performance measures of players in ODI cricket. The assessment involved key players of various countries considering several series of matches and measures like innings, batting average, batting contribution and so on. Though the data used for the experiment was not exhaustive, it had an adequate quantity in providing strong evidence of player ranking for long-term viability.

Hemanta Saikia and Dibyojoyoti Bhattacharjee [9] proposed a Bayesian classification approach on classifying all-rounders of IPL. In their work, strike and economy rates was considered for the classification of all-rounder performance. By analysing the above parameters, all-rounders were classified into four classes, namely batting all-rounder, bowling all-rounder, performer and under-performer. The training samples consisted of 35 all-rounders from the first three seasons of IPL. Amin and Sharma [2] proposed a linear programming model considering several factors that are related to the player's performance. Statistics of the player's scores are combined from IPL season 4 so that the model could be used to build a national cricket team from top cricketers. Saikia et al. [10] proposed a novel approach in simplifying the selector's job by assessing the cricketer's performance into a single numerical value, which measures the efficiency of the cricketer. The data from IPL-2012 was considered to analyse the player's performance under various expertise. The outcome from this study showed that the several non-performers were selected leaving behind the good performers. Ahmed et al. [1] introduced a new gene representation model and multi-objective technique for optimizing the strength of overall batting and bowling of a team. Another factor like fielding performance was used for making a better decision. As a performance indicator, the case study included historical data of player performance and the set of players auctioned during the fourth season of IPL.

With the player performance analysis, there has been a significant contribution to the selection of the best team. Dibyojoyoti Bhattacharjee and Hemanta Saikia [4] in their work developed an objective approach using binary integer programming for selection of optimum balanced team from a set of players. The data from the fifth

season of IPL was considered to validate the proposed model. Aqil Burney et al. [5] proposed a genetic algorithm-based technique to select the balanced team in a multiplayer game. The analysis was carried out on Pakistani cricket players for the test team selection based on their previous outcome (i.e., win or loose) considering recent matches.

Most of the existing work focused on the best team selection and the player performance from the cricket data by using statistical approaches. To best of our knowledge, it was found that the fantasy league data is not explored in analysing and predicting the winning positions by using DNN. In our proposed model, the winning positions are analysed using ML techniques in the fantasy cricket league played during IPL-2019. To the best of our knowledge, this is the first of its kind research work carried out on predicting the winning positions in the Dream11 fantasy cricket league data.

3 Materials and Method

The overall workflow for the proposed approach to predict winning positions in fantasy cricket contests is illustrated in Fig. 1.

The data collected from a fantasy cricket contest application is initially fed into a preprocessing module where various strategies are used to prepare the data to be fed into the deep neural network-based classifier for training and prediction. The methodology is explained in detail in subsequent sections.

3.1 Dataset

For the experimentation, the data was collected from a private contest played in the Dream11 app during the 12th IPL season in 2019. Eight teams participated in the IPL 2019 tournament—Mumbai Indians (MI), Chennai Super Kings (CSK), Delhi Capitals (DC), Sunrisers Hyderabad (SRH), Kolkata Knight Riders (KKR), Kings XI Punjab (KXIP), Rajasthan Royals (RR) and Royal Challengers Bangalore (RCB). The private contest in the Dream11 app consisted of 7 participants, who played for winning the first three positions.

When a match is being played between two teams, we, the contestants, create our Dream11 team from playing 11 cricketers of both sides. Constraints in picking

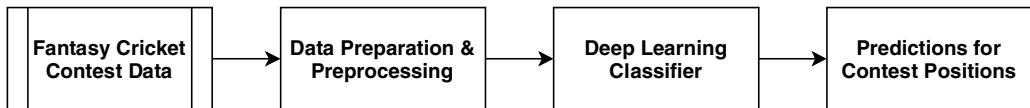


Fig. 1 Proposed approach for fantasy cricket contest winner positions prediction



Fig. 2 Sample scored points of contestants. <https://www.dream11.com/games/fantasy-cricket/point-system>

up players are as follows: A maximum of 7 and a minimum of 4 players have to be selected from each team, with one wicket-keeper, 3–5 batsmen, 1–3 all-rounder and 3–5 bowlers. As the match progresses, each contestant gains or losses points for each player. Initially, for each playing cricketer, 2 points are awarded for being in the playing 11. Later, the points start increasing depending on the performance of the player in batting, bowling and fielding performance w.r.t fantasy point system.⁶ Supposably, a batsman when he hits a boundary, six, a half-century or century, and a bowler for picking up 4 or 5 wickets, also for a maiden over bowled additional bonus points will be provided. Apart from these, two players are chosen as captain and vice-captain by each contestant from their respective Dream11 team. And their points get increased by $\times 2$ and $\times 1.5$ times of the points scored by the player during the match.

Once the match completes, each contestant will have a scored points w.r.t their dream team as shown in Fig. 2. Depending on the scored points (from highest to lowest), each contestant will be ranked from first to seventh accordingly. A total of 60 matches, which included 56 league matches, three play-offs and one final match. Hence, the dataset consists of 60 rows, each row pertaining to a game played in the 12th season of IPL. The primary dataset includes various attributes such as teams played, winning team, losing team, venue of the game and finally, the seven positions of contestants.

3.2 Data Preparation and Preprocessing

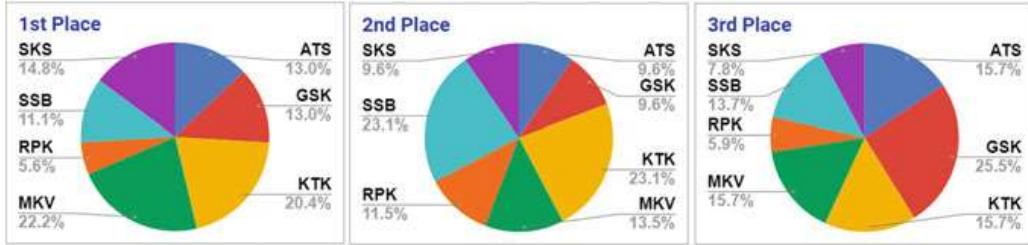
Data preparation and preprocessing are one of the major tasks involved in transforming, integrating and preparing raw data into a relevant ML model. Our raw data

⁶<https://www.dream11.com/games/fantasy-cricket/point-system>.

Table 1 Total no. of times contestants positioned

Cont.	1st	2nd	3rd	4th	5th	6th	7th
ATS	7	5	8	5	5	6	1
GSK	7	5	13	10	7	7	2
KTK	11	12	8	6	10	4	2
MKV	12	7	8	7	8	6	2
RPK	3	6	3	5	6	6	3
SKS	8	5	4	17	8	5	1
SSB	6	12	7	4	9	10	2

Note Cont. = Contestant's name

**Fig. 3** Overall contestant first, second and third positions

contains 60 instances with 420 values, out of which we found 80 values were missing. It could be because individual contestants did not take part in the contest due to some reasons or because of the cricket match getting abandoned (i.e., RCB vs. RR match). In case of no result or match abandoned, we ignored the row. The missing values were filled with a contestant name who has positioned the maximum number of times in that respective position. Observing to the condition that he should not be present in that particular row (i.e., he did not participate in that particular contest). The number of times each contestant was ranked at each position (column-wise) are shown in Table 1, and Fig. 3 shows the overall percentage of winning in first three positions. Table 2 shows the preference ranking of the contestants for IPL teams. Filling of missing values with statistical mode values of classes for labels ensured minimal bias for the dataset. One such example of missing data handled is illustrated in Table 3 (filled missing values are highlighted in yellow colour).

Additionally, preference ranking for the 8 IPL teams was recorded from all the participating contestants as extra information. This factor was also fed into the classifier, along with playing teams and venue information. The winner and loser columns were not considered for feeding into the classifier, as this information will not be available before a match. Finally, all string values in features (match details) and classes in labels (i.e., the contestants) were encoded as distinct positive integers while vectorizing to be fed into the deep learning classifier. To ensure that there are enough data points for the classifier to learn, we oversampled the data instances to twice its number. Therefore, the final data consisted of 118-row instances with 59 features, which included the two teams and the 56 preferences (8 preferences pertaining to

Table 2 Preference ranking of contestants for IPL teams

Cont.	CSK	RCB	DC	SRH	MI	KXIP	KKR	RR
ATS	3	1	4	5	2	8	6	7
GSK	2	3	6	4	1	7	8	5
KTK	1	4	2	3	5	8	7	6
MKV	8	1	2	6	5	3	7	4
RPK	2	1	6	3	4	5	8	7
SKS	4	1	2	6	3	7	8	5
SSB	8	1	2	7	3	5	6	4

Note Cont. = Contestant's name

Table 3 Sample data of fantasy cricket league after preprocessing

Match	Venue	1 st	2 nd	3 rd	4 th	5 th	6 th	7 th
RCB vs MI	Bengaluru	MKV	SKS	KTK	ATS	RPK	SSB	GSK
CSK vs RR	Chennai	ATS	SSB	GSK	MKV	KTK	SKS	RPK ^a
MI vs RCB	Mumbai	RPK	GSK	SSB	MKV	SKS	ATS	KTK
SRH vs CSK	Hyderabad	SKS	ATS	RPK	SSB	KTK	GSK	MKV
RR vs MI	Jaipur	GSK	KTK	ATS	SKS	RPK	MKV	SSB ^a
KXIP vs CSK	Punjab	SSB	KTK	GSK	MKV	RPK	SKS ^a	ATS ^a
KKR vs DC	Kolkata	KTK	ATS	GSK	SSB	MKV	SKS	RPK

^a Highlighted cells indicate filled missing data

each contestant) and 7 labels (7 winning positions) with each label consisting of 7 classes.

3.3 Prediction Model

For the effective prediction of winning positions of the contests, we adopted the usage of a feed-forward deep neural network (DNN) model. The input layer of the neural network consisted of 59 neurons, corresponding to the number of features fed into the classifier. The model consisted of three hidden layers, each with 64, 32 and 16 neurons, respectively. Finally, the output layer consisted of 7 neurons to predict one of the seven classes for a position. A learning rate of 0.3 is used for training 500 epochs. Categorical cross-entropy loss function with sigmoid activation functions in hidden layers and a softmax activation function in the output layer is used for training the classifier. The basic hyperparameters (i.e., no. of layers, neurons and learning rate) were empirically optimized using the grid search approach [3]. In this work, we have considered the neural network model as separate classifier for various winning positions.

4 Experimental Results and Discussion

A DNN approach for prediction of winning positions of Dream11 contestants was designed. The features and respective labels were fed into the classifier and the classification performance of the proposed DNN approach was observed and analysed.

4.1 Performance Analysis for Respective Winning Positions

The performance was observed in terms of soft computing metrics such as accuracy, precision, recall, F-score, Matthews correlation coefficient (MCC) and area under receiver operating characteristic curve (AUROC) for the prediction of first, second and third positions for the Dream11 contest. The class-wise results obtained for classification of first, second and third winning position are tabulated in Tables 4, 5 and 6 respectively.

Table 4 CP of DNN for first position

Class	Pre.	Recall	F-Sc.	MCC	AUROC
ATS	1	0.86	0.93	0.92	0.95
GSK	0.94	1	0.97	0.97	1
KTK	0.91	0.91	0.91	0.89	0.96
MKV	0.89	0.94	0.91	0.88	0.96
RPK	1	0.84	0.91	0.91	0.99
SKS	1	1	1	1	1
SSB	1	1	1	1	1
Avg.	0.94	0.94	0.94	0.92	0.97

Note Pre. = Precision, F-Sc. = F-Score, Pos. = Position

Table 5 CP of DNN for second position

Class	Pre.	Recall	F-Sc.	MCC	AUROC
ATS	0.73	0.8	0.77	0.74	0.93
GSK	0.91	1	0.96	0.95	1
KTK	1	0.92	0.96	0.94	0.96
MKV	0.82	0.75	0.79	0.76	0.93
RPK	0.92	0.79	0.85	0.83	0.94
SKS	0.75	0.86	0.80	0.78	0.99
SSB	0.89	0.96	0.92	0.90	0.98
Avg.	0.88	0.88	0.88	0.86	0.95

Note Pre. = Precision, F-Sc. = F-Score, Pos. = Position

Table 6 CP of DNN for third position

Class	Pre.	Recall	F-Sc.	MCC	AUROC
ATS	1	1	1	1	1
GSK	1	0.94	0.97	0.96	0.96
KTK	0.95	1	0.97	0.97	1
MKV	1	1	1	1	1
RPK	0.5	0.33	0.4	0.38	0.87
SKS	0.8	1	0.88	0.88	1
SSB	0.94	1	0.97	0.96	1
Avg.	0.94	0.94	0.94	0.93	0.98

Note Pre. = Precision, F-Sc. = F-Score, Pos. = Position

4.2 Baseline Comparison

The performance of the proposed DNN approach was compared with the traditional baseline machine learning classifiers: Logistic regression (LR), k-nearest neighbours (KNN), Naive Bayes (NB), support vector machines (SVM), random forest (RF) and Adaboost on the Dream11 contest dataset. Best performance of KNN implementations with K values 11, 7 and 5 were considered for classification. Random forest classifier was implemented with a depth value of 3. SVM classifier's best performance identified with the C parameter value of 11. Here, C is a regularization parameter of SVM classifier that can generalize the classifier for hidden data and tries to achieve low training and testing error. All other classifiers were implemented with default parameters. Tenfold cross-validation was performed for all classifiers and the performance results are tabulated in Table 7.

Table 7 Baseline comparison of proposed DNN approach w.r.t prediction accuracy

Classifier	First Pos.	Second Pos.	Third Pos.
LR	52.54	51.69	60.16
KNN	33.89	33.05	27.11
NB	24.57	38.13	23.72
SVM	83.05	81.35	86.44
RF	75.42	77.11	83.05
Adaboost	33.05	28.81	33.89
Proposed	94.06	88.13	94.91

Note Pre. = Precision, F-Sc. = F-Score, Pos. = Position

4.3 Discussion

The proposed DNN approach for predicting Dream11 contest-winning positions outperformed all other baseline classifiers by at least 13, 8 and 9% in terms of prediction accuracy for first, second and third winning positions. With the help of effective data preparation strategies that included preprocessing and missing data handling, the DNN-based classifier is able to generalize features effectively, thereby making good predictions on the contest-winning positions compared to other baseline classifiers.

It was observed from the contestant's preferences that whenever their favourite team (first preference) won, the chances of winning at least third position was more. However, when their favourite team lost, mostly the respective contestants also lost. This indicates that whenever contestant's favourite teams are playing, the emotion or spirit towards the team is more evident and becomes a bias than logical team creation. Even this has been effectively captured by the DNN classifier to predict the winning positions even more accurately.

5 Conclusion and Future Work

In this paper, we have used a feed-forward DNN classifier predicting the contestant winning positions for a Dream11 contest. Its performance was benchmarked against different kinds of existing machine learning algorithms. The experimental evaluation was performed using a real dataset that the contestants created and participated in the Dream11 app during the IPL 2019 season. Effective data preparation strategies that involved preprocessing and missing data handling ensured effective generalization by the DNN classifier to get promising results. The experiment shows that the DNN model outperforms other ML techniques by at least 13, 8 and 9% in terms of prediction accuracy for first, second and third winning positions, which indicates that this DNN-based model can be applied for accurately finding out the winners in fantasy cricket leagues.

As deep learning-based approaches have proven to work effectively with the huge amount of data, as a future work, we have planned to increase and enhance the dataset further by including additional features like contestant total points, the performance of the players in terms of points and in terms of cricket like runs, wickets, catches, etc. With the enhanced data, we also plan to investigate the effectiveness of multi-class multi-label prediction models for fantasy cricket winning positions as part of future work.

Dataset Consent

The data is collected from a private contest played by seven contestants in the Dream11 app. All seven contestants provide their consent to use this data for the research purpose and the data is made available at: <https://github.com/2karthikbhat/Dream-11-Dataset>

References

1. Ahmed, F., Deb, K., Jindal, A.: Multi-objective optimization and decision making approaches to cricket team selection. *Appl. Soft Comput.* **13**(1), 402–414 (2013)
2. Amin, G.R., Sharma, S.K.: Cricket team selection using data envelopment analysis. *Eur. J. Sport Sci.* **14**(sup1), S369–S376 (2014)
3. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **13**(Feb), 281–305 (2012)
4. Bhattacharjee, D., Saikia, H.: An objective approach of balanced cricket team selection using binary integer programming method. *Opsearch* **53**(2), 225–247 (2016)
5. Burney, S.M.A., Mahmood, N., Rizwan, K., Amjad, U.: A generic approach for team selection in multi-player games using genetic algorithm. *Int. J. Comput. Appl.* **40**(17), 11–17 (2012). Full text available
6. Lemmer, H.H.: A measure of the current bowling performance in cricket. *S. Afr. J. Res. Sport Phys. Educ. Recreation* **28**(2), 91–103 (2006)
7. Lewis, A.: Extending the range of player-performance measures in one-day cricket. *J. Oper. Res. Soc.* **59**(6), 729–742 (2008)
8. Naha, S.: Flight of fantasy or reflections of passion? knowledge, skill and fantasy cricket. *Sport in Society*, pp. 1–13 (2019). <https://doi.org/10.1080/17430437.2019.1607012>
9. Saikia, H., Bhattacharjee, D.: On classification of all-rounders of the Indian premier league (IPL): a Bayesian approach. *Vikalpa* **36**(4), 51–66 (2011)
10. Saikia, H., Bhattacharjee, D., Radhakrishnan, U.K.: A new model for player selection in cricket. *Int. J. Perform. Anal. Sport* **16**(1), 373–388 (2016)

IoT Stream Data Compression Using LDPC Coding



Rajni Jindal, Neetesh Kumar, and Sanjay Patidar

Abstract Internet of Things is an extension of Internet connectivity to physical devices in heterogeneous networks. IoT network consists of the Internet and low-speed IoT. In IoT, with the help of the Internet, many devices can communicate and interact with each other. The devices can be monitored remotely. A large amount of data stream is transmitted from IoT devices to the Internet. Therefore, it is observed that there is a difference in the speed and (MTU) maximum transfer unit of IoT which in turn results in the overhead. Overhead also depends on the data, as the data size increases the value of overhead increases. This is a major problem in IoT devices as they are sensitive toward power consumption. This needs to be handled in real-time. To solve this problem there is an approach that compresses the data stream. The approach is Low-density parity-check code (LDPC). LDPC compression technique can also be applied to encrypted data. Therefore this method is also beneficial in privacy and confidentiality. For the implementation, we have used Raspberry Pi, collectors and a computer. Raspberry Pi is used to generate data stream, collector is used to collect the data. To calculate efficiency, we focused on specific data in different sizes. Efficiency is calculated using transmission time and compression time. The dataset used for the experiment is temperature sensor data. As a result, it is observed that the data stream transmission time is decreased by 45%. As a result of this study, the data transmission time from IoT to the collector can be reduced after compression using the LDPC code.

Keywords IoT · LDPC · 6LoWPAN · DTLS · BT-LE · IoT SDK

R. Jindal · S. Patidar (✉)
Delhi Technological University, New Delhi, India
e-mail: sanjaypatidarcs@gmail.com

R. Jindal
e-mail: rajnijindal@dce.ac.in

N. Kumar
ABV-Indian Institute of Information Technology and Management, Gwalior, India
e-mail: dgoldneetesh15@gmail.com

1 Introduction

IoT (Internet of things) is an emerging technology that deals with streaming data which results into increasing volume of the data [2]. It deals with the data related to various fields which can be processed online or in real-time. So, generated data varies and expanded [3]. To get valuable information from collected data, data stream collector plays an important role as it collects the data stream, process it and extracts relevant information from data stream [4]. The data steam generated by an IoT device traverses a network comprised of a number of different things having varying communication speeds. While data stream transmitted from IoT to collector travels through a heterogenous network consist of IoT and Internet. In terms of speed of the network, IoT having limited resources like Bluetooth Low Energy (BT-LE) are slow networks, while traditional devices like laptops and phones use speedy networks such as Ethernet or Wi-Fi [5].

It has been observed that the size of data transmitted is directly dependent on the number of fragmentations which affects maximum transfer unit as it returns a small value when various sizes of data are transmitted using Bluetooth and Fast Ethernet. Because of the stark difference in speeds, the transmission time of the IoT network and device is very high.

We use a handler which improves the usability of the data stream. The technique used is LDPC(low-density parity-check) Coding. It can be applied to data streams that are linear in nature; i.e., they don't change much and sudden changes in the data are uncommon. For example: Temperature data of a room. A reduction in transmission time over Bluetooth will help to satisfy the data stream characteristics, which is processed in real time.

1.1 Speeds of Internet and IoT Networks

Table 1 presented a comparison of Internet and IoT in terms of speeds and MTUs (maximum transfer unites) [6, 7].

Table 1 Speeds of Internet and IoT networks

Network	Standard/Protocol	Speed	MTU
Internet	802.3bs (Gigabit Ethernet)	Up to 400 Gb/s	1500
	802.11ax (Wi-Fi)	Up to 10 Gb/s	2303
IoT	Z-Wave	40–100 Kb/s	127
	802.15.4 (ZigBee)	250 Kb/s	127
	Bluetooth Low Energy	1 Mb/s	27

Table 2 Specification and platform of IoT prototype

Name	CPU	RAM	Power
Raspberry Pi 4B	ARM CORTEX-A72	4 GB	5 V USB-C
Arduino MEGA 2560	ATMEGA 2560	256 kB	7-12 V USB
BeagleBoard, Pocket Beag	OSD3358SM ARM Cortex	512 MB	5 V
Phidget SBC4	Allwinner A20	512 MB	5 V

1.2 Specification and Platform of IoT Prototype

Table 2 shows device specification which is helpful for the implementation of IoT prototype [8]. From table it has been observed that, desktop or laptop are faster than CPU, smaller size of Random Access Memory and at even low voltage level power operates. We have used Raspberry Pi 4B in our experiment having ARM CORTEX-A72 CPU, 4 GB RAM, and 5 V/USB-C power. Similarly Arduino MEGA 2560, BeagleBoard, Pocket Beagle and Phidget SBC4 are the mentioned in the below table with their specifications.

2 Background

This section describes the study of the IoT network architecture and structure to understand the basic theories and terminologies used.

2.1 IoT Architecture

We need to develop a process flow for a definite framework over which an IoT solution is built [9]. Figure 1 shows the architecture of IoT. The IoT Architecture generally comprises of these 4 stages:

Physical layer

It consists of sensors that can be used to collect data. Data is collected from the environment. It identifies the smart objects and collects physical parameters from the objects [10, 11]. An actuator affects change in the environment. For example if there is change in temperature, sensor will sense and the light is low, then an actuator will switch on the street lights automatically.

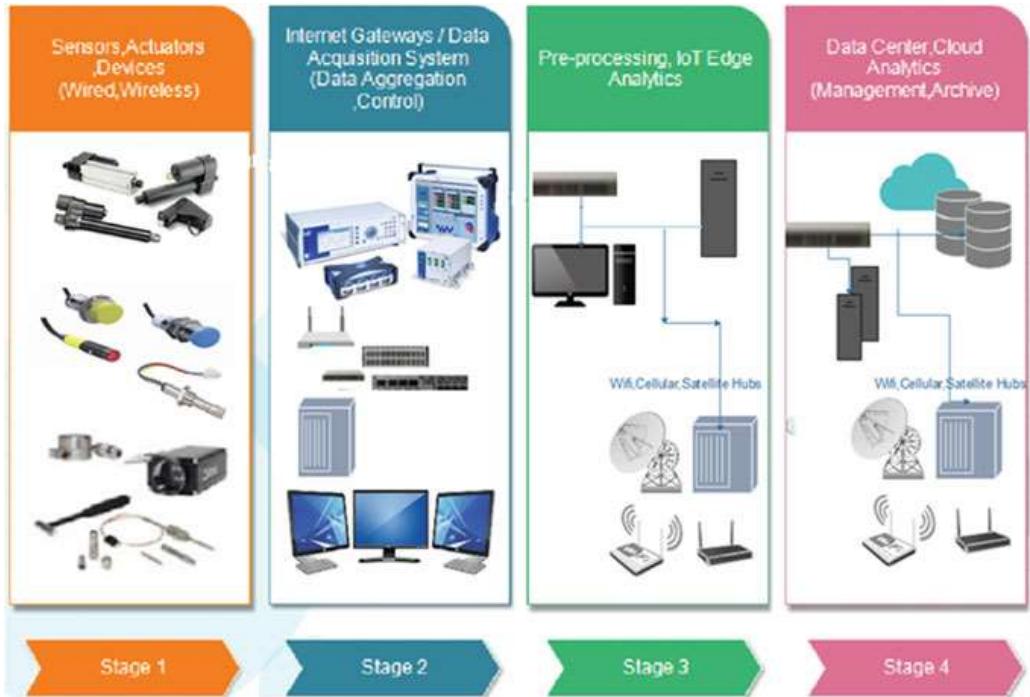


Fig. 1 Architecture of IoT

Internet Gateway

In this layer data generated by sensors should be preprocessed. When it enters the processing stage, the data is received in the analog form which needs to be aggregated and converted into digital. In this layer, the Internet gateway routes the data over WLANs or networks which can be processed further. The data collection and preprocessing is done in real time using this and layer above it. The gateways have some additional functionalities as data management, data analytics, and malware protection.

Data Preprocessing

In this stage the data is preprocessed by edge computing or some IT infrastructure. In edge computing further analysis of the data is performed by IT infrastructure [11]. The above two layers situated at device side and edge IT processing is situated in other edge or remote offices. Usually, the IoT data is sent to server or data center which use huge resources and network bandwidth. So edge computing system perform data analytics to reduce IT infrastructure burden.

Analytics and Management

In this stage the data is managed, stored and analyzed in IT system. The data needs more processing and does not need immediate feedback. Therefore a cloud-based system or data center is suitable for this purpose perfectly.

2.2 Hardware Deployments Factors

During deployment, we generally consider the following factors, which show how the hardware is deployed.

Cost

For the value of provided data what cost can be supported for each device.

I/O roles

The device can be a sensor, an actuator or it can be the combination of the both roles.

Power budget

The device can use electricity or power. So think for the device will require either battery or solar power.

Networking environment

In Internet using TCP/IP route, the device can be wired directly. Some types of connection can be expensive with high traffic, such as cellular. So think for a network which is reliable. The reliability will effect directly to the latency and throughput.

Out of four hardware deployment factor last two can affect the compression technique. Raspberry pi may need battery to operate and networking environment may also affect the transmission as it can be wired or wireless also reliability can be one of the issue.

3 Related Work

In [12], researcher proposed a model which uses variable key length of 128, 64 and 32 bits was defined which helps to provide real-time security verification to the data stream. It has been observed that, in case of unbounded data stream use of stream ciphers is more reasonable. So, they uses cipher stream that possess similar characteristics of stream data. As a result, the encryption and compression decreases the transmission time from IoT device to the collector and improves the efficiency.

In [13], researcher studied the IPsec protocol header compression in 6LoWPAN(6 Low Power Area Network). IPsec ensures integrity and confidentiality of transport layer header as it operated at network layer. Data integrity and authentication is handled by Authentication Header (AH) and Encapsulating Security Payload (ESP) of IPsec. Authentication header uses authentication function which is compressed from 24 to 16 bytes while Encapsulating Security Payload uses some cryptographic function which compresses the data from 18 to 12 bytes. Therefore some bytes are saved like AH save 8 bytes, ESP with encryption save 6 bytes and ESP with authentication and encryption saves 6 bytes. But ESP has some limitation that is the upper layer header is encrypted and hence cannot be compressed.

In [14], researcher focuses on the DTLS (datagram transport layer security) header compression. DTLS is a protocol for security which uses constrained application protocol (CoAP). DTLS is further divided into two layers: record and handshake Layers. Record layer is compressed from 104 to 40 bytes and handshake layer is compressed from 96 to 24 bytes. Therefore, 64 bytes are saved in record layer and 72 bytes are saved in the Handshake layer.

In [15], researcher focused on LDPC codes for compression but their focus is on compression of general data and not on the specific data. In decompression technique unattributed data is not described.

In [16], author proposed solution of distributed source encoding and decoding problem. Distributed source encoding is helpful in managing network traffic such as sensor network. For which author proposed a block code which can achieve on any point in Slepian Wolf region which is helpful to determine the region with lossless compression coding rate.

In [17], first time researchers studied about encrypted data compression. Encrypted data with key used as sources in distributed source coding. At receiver end, receiver decompresses and decrypts the data simultaneously using received key and encrypted data. For experiment and simulation, they have taken an unrealistic data having a binary image of 100*100 which consists of 706 pixels of value 1 which are compressed at $\frac{1}{2}$ rate and decompressed with no loss. Here the encryption is performed on bitwise XOR operation by random key stream.

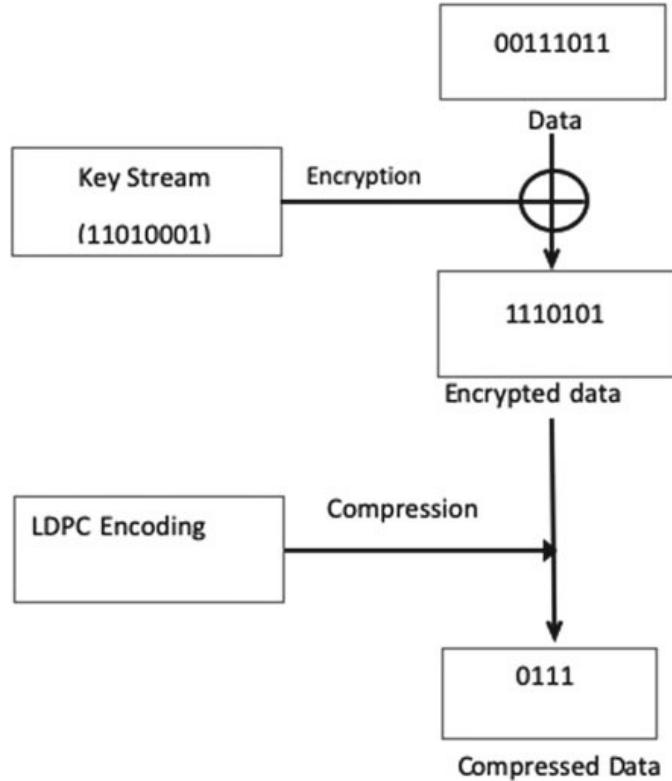
In [18], proposed resolution progressive compression of the type of lossless compression to compress encrypted grayscale image more efficiently. When the resolution of the image gradually increases, then it uses the statistical value of sampled picture of low resolution. Image encrypted using XOR operation and the key stream is in bits.

In [19], the data stream is transmitted from the IoT device. Among stream cipher and block cipher, selecting stream cipher has more advantages. As stream cipher has fast processing speed and in case of limited resources in IoT stream simple structure of stream cipher is sufficient for improvement of efficiency. For the device which have limited resource as Grain, Trivium, MICKEY, and WG-8, apply stream ciphers which are lightweight.

4 Proposed Work

This section includes details of data stream handler to improve utilization of data stream. Section 4.1 includes data stream transmission from IoT device and Sect. 4.2 discuss the decompression of data stream received at collector.

Fig. 2 Data transmission in IoT device



4.1 Transmission of Data in IoT Device

In Fig. 2 the data with the value 00111011 (in binary) are compressed using LDPC codes and encrypted with key streams (11010001).The encryption is done by the XOR operation. In example, the data is read in binary and processed also in binary. Then input value is XOR with key streams 11010001. The above procedure is applied to all the data streams which perform zero padding if required. The applied algorithm for encryption is required as needed. And then using LDPC coding the encrypted data after Ex-OR with key stream compressed into 0111.

4.2 Receiving of Data in Data Stream Collector

Figure 3 shows that after receiving the compressed data from IoT boarder router, the data starts decompressing. The example shows the decompressing process for received 4 bit data to the original data. Each bit of the received data stream (0111) may have two outputs 01 and 10 for 1 and 00 and 11 for 0. Four bits are needed to decompress 1 byte and then combination of total 16 outputs are generated. The key stream XOR with the generated 16 outputs to perform decryption. With encryption key the data can be decrypted to correct value and then it decide the correct value by sensor characteristics of IoT device so the correct output is generated. In example the

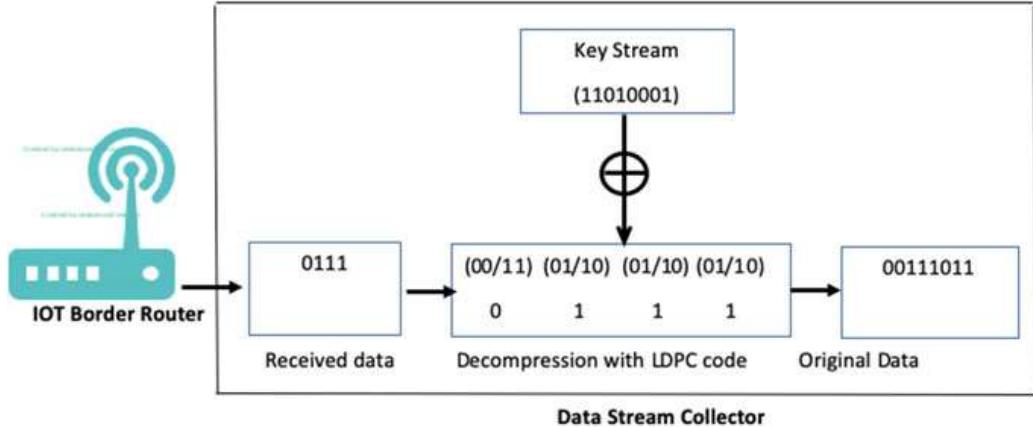


Fig. 3 Receiving data in data stream collector

experiment used temperature value of data. Temperature range can be in 25–120°C with SHT75 [20]. So it is possible to remove the values which fall outside of the range.

5 Parameters/Experimentation Setting

5.1 Implementation

This section includes implementation details of IoT device, IoT border router, data stream collector and dataset.

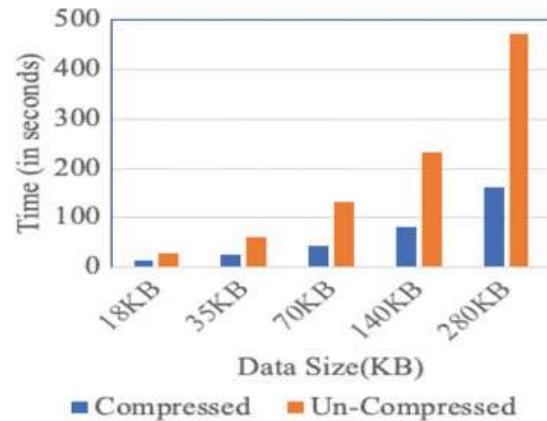
IoT Device: Raspberry Pi 4B used as IoT device for implementation. We have used 6LoWPAN that utilizes BT-LE (Bluetooth low energy), and IoT SDK(software development kit) [21]. Python is used to write Sending and receiving program and the compression was also written using Python language. We have used some library for compression using LDPC coding.

IoT Border Router: Raspberry Pi 4B used for implementation of IoT border router. IoT SDK installed [21] for supporting BT-LE and 6LoWPAN. Ethernet wired connection set up to the data stream collector.

Data Stream Collector: We use desktop computer having operating system Ubuntu 18.04.6 LTS to make data stream collector.

Dataset: The dataset consist of eight gas, temperature and humidity sensors [22].

Fig. 4 Comparision of transmission time



5.2 Result and Analysis

To prove the efficiency of LDPC code following parameters are used and analyzed the result.

Transmission Time

See Fig. 4.

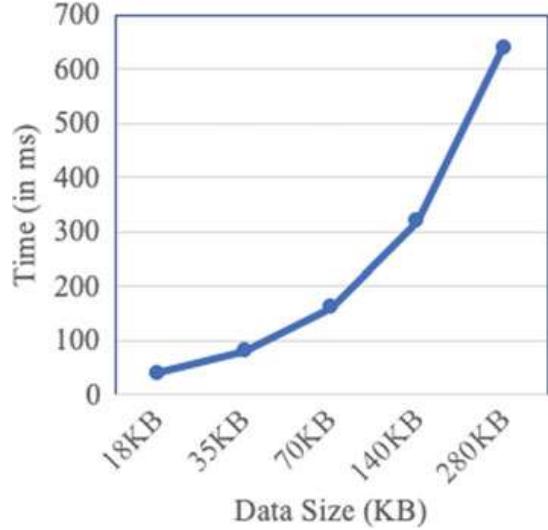
Above is the comparison between Transmission times for Compressed and Un-Compressed data. We divided the dataset into data size of 18 KB, 35 KB, 70 KB, 140 KB and 280 KB and use LDPC coding to compress this data and transmitted it to the Internet from IoT device. Then compare the transmission time of original data with compressed data using LDPC which gives around 5% reduction in transmission time.

Transmission Time Reduction Rate:

As shown in Table 3, there is decrease in transmission time for data size of 18 KB, 35 KB, 70 KB, 140 KB, and 280 KB. The average transmission time reduction is approx. 45% which may be depend on data size chosen and hardware used for implementation.

Table 3 Transmission time reduction rate (%)

Data Size (KB)	Rate of reduction (%)
18	58.52
35	53.12
70	40.23
140	36.32
280	38.43

Fig. 5 Compression time

Compression Time:

Shows the graphical representation of the increase in compression time with increasing data size. The compression time linearly increases with increasing dataset size. However, even for large amounts of data, the compression time is minuscule. The compression time for smallest and largest data 18 KB and 280 KB takes 0.05 and 0.64 s.

Also, the transmission time for each data size was much larger than the compression time and the results obtained don't change even if the compression time is added to the transmission times of the Compressed Data (Fig. 5).

6 Conclusion and Future Scope

In this paper, we used a handler for Data Streams that reduced the transmission time of data from IoT device to data stream collector. LDPC coding is used to reduce the size of the actual payload and overcome the difference in performance of speed between Internet and the IoT device. Using the Raspberry Pi 4B and a laptop, IoT devices are implemented which can be used to generate stream data. We have taken the temperature data from dataset created by temperature sensor. As a result average transmission time is reduced by 45% using LDPC coding in our work. This result is achieved after applying the technique on selected data set using Raspberry pi 4 and updated operating system as compared to previous research by Jaejin Jang [1]. This result may vary from machine and hardware used. In future we can compress the data based on range of input data set and can reduce more transmission time.

We had examined the stream data of IoT devices which can generate a range of values. The implementation of this method required prior knowledge of device, data set and their characteristics. The applications of IoT are diverse and types of data

collected are vastly different. This limits the application of this study as it is more suited to linear datasets. Future studies will be able to work on this problem and figure out a way to make the results of this study better.

References

1. Jang, Jaejin, Jung, Im Y., Park, Jong Hyuk: An effective handling of secure data stream in IoT. *Appl. Soft Comput.* **68**, 811–820 (2018)
2. Samira Amira Pouyanfar, Yimin Yang, Shu-Ching Chen, Mei-Ling Shyu, S. S. Iyengar, Multi-media Big Data Analytics: A Survey, *ACM Computing Surveys*, Vol. 51, No. 1, Article 10. Publication date: January 2018, pp. 1–34
3. Muhammad Usman, Mian Ahmad Jan, Xiangjian He, and Jinjun Chen. 2019. A Survey on Big Multimedia Data Processing and Management in Smart Cities. *ACM Comput. Surv.* **52**, 3, Article 54 (June 2019), pp. 1–29
4. C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, A framework for clustering evolving data streams, in *Proc. 29th Int. Conf. Very Large Data Bases*, Berlin, Germany, 2003, pp. 81–92
5. A. Pughat, V. Sharma, A review on stochastic approach for dynamic power management in wireless sensor networks, *Hum. Cent. Comput. Inf. Sci.* **5** (4) (2015)
6. Freschi, Valerio, Lattanzi, Emanuele: A Study on the Impact of Packet Length on Communication in Low Power Wireless Sensor Networks Under Interference. *IEEE Internet of Things Journal* **6**(2), 1–11 (2019)
7. Christensen, K., Reviriego, P., Nordman, B., Bennett, M., Mostowfi, M., Maestro, J.A.: IEEE 802.3az: the road to energy efficient ethernet. *IEEE Commun. Mag.* 1–7 (2010)
8. Wang, W., He, S., Sun, L., Jiang, T., Zhang, Q.: Cross-technology communications for heterogeneous IoT devices through artificial doppler shifts. *IEEE Trans. Wireless Commun.* **18**(2), 1–11 (2019)
9. Azar, J., Makhoul, A., Barhamgi, M., Couturier, R.: An energy efficient IoT data compression approach for edge machine learning. *Future Gener. Comput. Syst.* **96**, 168–175 (2019)
10. Schneider, S., Hirzel, M., Gedik, B.: Tutorial: stream processing optimizations. In: *ACM DEBS*, pp. 249–58 (2013)
11. Tsai, C.-W., Lai, C.-F., Chao, H.-C., Vasilakos, A.V.: Big data analytics: a survey. *J. Big Data* **2**(1), 1–32 (2015)
12. Puthal, D., Ranjan, R., Chen, J.: DLSeF: a dynamic key-length-based efficient real time security verification model for big data stream. In: *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 6, no. 2, (2016)
13. Gomes, T., Salgado, F., Pinto, S., Cabral, J., Tavares, A.: A 6LoWPAN accelerator for Internet of things endpoint devices. *IEEE Internet Things J.* **5**(1), 1–7 (2018)
14. Park, C.-S., Park, W.-S.: A Group-oriented DTLS handshake for secure IoT applications, *IEEE Trans. Autonom. Sci. Eng.* **15**(4), (2018)
15. Liu X., Xiong F., Wang Z., Liang, S.: Design of binary LDPC codes with parallel vector message passing. *IEEE Trans. Commun.* **66**(4), (2018)
16. Nomura, R., Han T.S.: Second-order Slepian-Wolf coding theorems for non-mixed and mixed sources. *IEEE Trans. Inform. Theory* **60**(9), (2014)
17. Johnson, M., Ishwar, P., Prabhakaran, V., Schonberg, D., Ramchandran, K.: on compressing encrypted data. *EEE Trans. Signal Process* **52**(10), 2992–3006 (2004)
18. Kishore, P., Nagendra, N., Reddy, K., Murthy, V.: Smoothing and optimal compression of encrypted gray scale images. *Int. J. Eng. Res. Appl. (IJERA)* **2**(3), 23–28 (2012)
19. Zeng, G., Dong, X., Bornemann, J.: Reconfigurable feedback shift register based stream cipher for wireless sensor networks. *IEEE Wireless Commun. Lett.* **2**(5), 1–4 (2013)
20. SHT75 Datasheet. <http://legacy.caricoos.org>

21. Semiconductor, N.: NRF5 SDK for IoT. <https://www.nordicsemsemi.com/eng/products/Bluetooth-low-energy/nRF5-SDK-for-IoT>
22. Huerta, R., Mosqueiro, T., Fonollosa, J., Rulkov, N., Rodriguez-Lujan, I.: Online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. *Chemometr. Intell. Lab. Syst.* (2016)

Improved PSO for Task Scheduling in Cloud Computing



Richa and Bettahally N. Keshavamurthy

Abstract An improved Particle Swarm Optimization (PSO) is used for performing task scheduling in cloud computing with the aim of distributing uniform load on each Virtual Machine (VM) in a datacenter. It is achieved using an objective function which tries to enhance the candidate solutions iteratively and thus finds an optimal mapping of task set to VM set. Experimental results have shown that the improved PSO performs better than the original PSO by maintaining the consistency in scheduling.

Keywords Cloud computing · Particle swarm optimization · Task scheduling · Objective function

1 Introduction

Cloud computing is a computing paradigm in which users are increasingly accessing software, computing infrastructure and files over the Internet instead of on their desktops. It has benefited users by reducing the cost and complexity of owning and operating servers and networks. Because of its enormous benefits, number of cloud users are significantly increasing resulting in increase in load. Figure 1 shows the model of load balancing where ‘ n ’ number of user requests are received by the load balancer and are distributed to ‘ m ’ number of virtual machines based on the logic defined by the task scheduling algorithm.

Optimal allocation of user requests on computing nodes in the cloud environment is an NP-hard optimization problem [1]. Deterministic scheduling algorithms like First Come First Served (FCFS), Shortest Job First (SJF) and Round Robin (RR) are not usually capable of finding optimal solution [2]. A lot of research has tended toward metaheuristic algorithms to solve optimization problems.

Richa · B. N. Keshavamurthy (✉)
National Institute of Technology, Goa, India
e-mail: bnkeshav.fcse@nitgoa.ac.in

Richa
e-mail: richaverma21193@gmail.com

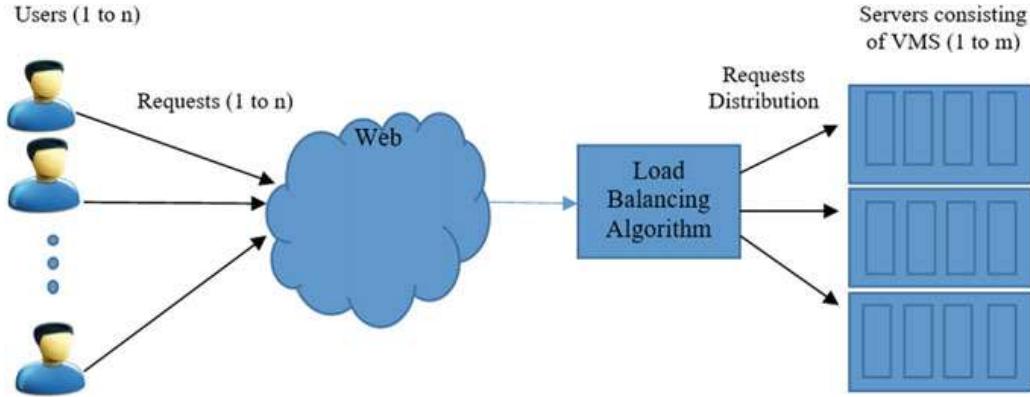


Fig. 1 Cloud load balancing model

These algorithms are shown capable in providing near to optimal solutions but have other problems like slow convergence and complexity [3]. Particle swarm optimization (PSO) algorithm is one of the popular metaheuristic scheduling algorithms which is simple to use and has better convergence [4]. PSO is used in task scheduling [4–9] with an objective to minimize the makespan for a given set of tasks to be performed on a given set of virtual machines.

In this paper, PSO is discussed with respect to load balancing in cloud computing. The reason why PSO is inconsistent in distributing the load uniformly and an improved PSO is presented which maintains the consistency of uniform task distribution.

2 Background

In PSO, each member of the population represents a particle and the population forms a swarm. The swarm is randomly initialized in beginning, each particle travels in the search space such that it memorizes the best previous positions of its neighbors and itself and learns its new position from this knowledge. Thus, all particles tend to move toward better positions over the search process until the swarm moves close enough to an optimum position. In PSO the velocity, V_{i+1} and position, P_{i+1} of i th particle changes in each iteration according to the following equations:

$$V_{i+1} = W V_i + C_1 R_1 (P_{\text{best}} - P_i) + C_2 R_2 (G_{\text{best}} - G_i) \quad (1)$$

$$P_{i+1} = V_{i+1} + P_i \quad (2)$$

P_{best} is its best position till now and G_{best} is the position of the best particle in the candidate solutions till now. C_1 and C_2 are the coefficients for cognitive and social behavior. R_1 and R_2 are random numbers used to create the effect of randomness

of the real bird flocking. Furthermore Eqs. 3 and 4 are used to add nonlinearity to update the position of each particle [4].

$$s(P(i+1)) = \frac{1}{1 + e^{(-P_{i+1})}} \quad (3)$$

$$P(i+1) = \begin{cases} 0, & s(P_{i+1}) \leq r \\ 1, & \text{otherwise} \end{cases} \quad (4)$$

where r is a random number between 0 and 1.

Random Inertia Weight (RIW) strategy is used to adjust the balance between local search and global search at any moment during algorithm run [10]. It uses simulated annealing method. Simulated Annealing (SA) is a probabilistic technique for approximating the global optimum of a given function. In RIW strategy, inertia weight adjusts adaptively along with changing fitness value, which can adjust local and global search. In this work the hybrid PSO_SA algorithm with improved objective function is used to give consistent results for uniform task scheduling.

3 Algorithm Improvement Discussion

Since PSO is a population-based metaheuristic algorithm, it maintains multiple candidate solutions and uses their characteristics to guide the search. The objective of using PSO in task scheduling is to minimize the makespan and is given by the formula 5 and 6:

$$\text{Makespan}(P) = \max(\text{CT}_{idl}(ti) \dots \text{CT}_{idM}(tj)) \quad (5)$$

$$\begin{aligned} \text{Objective function} &= \min(\text{Makespan}(P)) \\ \text{Where } P &\in \text{Candidate Solution Set} \end{aligned} \quad (6)$$

where Makespan (P) is the fitness value of P th candidate solution, which is the maximum completion time at any VM. $\text{CT}_{idx}(ty)$ is the completion time of x th virtual machine to complete all the tasks ‘y’ assigned to it. Likewise for all other particles makespan is calculated and finally the particle with minimum makespan is selected as the best particle.

3.1 Problem Statement

In a given heterogeneous cloud environment of N independent tasks of different sizes and M virtual machines (VMs) of different computing speed, N tasks are to

be scheduled on M VMs by using PSO in such a way that the load is properly balanced across all the given VMs in limited number of iterations. Can minimization of makespan guarantee optimal solution always?

Particles movement is highly influenced by the initial particles which are generated randomly in the beginning. There may be a case that all particles are generated similar to each other and are in great imbalance. Then their movement will be very slow toward convergence. And it may happen that minimum time at some VM for all particles is very less and similar to each other. It may get stuck into one minimum value and other VMs will only be the part of load balancing, which is highly undesirable.

Pictorial Representation of the Movement of Particles in the search space-

3.2 Possible Solution

Since the PSO objective function is focusing only on the minimization of the maximum completion time at any VM, PSO may not schedule the load in an efficient way.

In the present work, the objective function is modified to consider maximization of minimum completion time at any VM. Modified objective function is as follows:

$$\begin{aligned} \text{Difference}(P) = & (\max(CT_{idl}(ti) \dots CT_{idM}(tj))) \\ & - (\min(CT_{idl}(ti) \dots CT_{idM}(tj))) \end{aligned} \quad (7)$$

$$\begin{aligned} \text{Objective function} = & \min(\text{Difference}(P)) \\ \text{Where } P \in & \text{ Candidate Solution Set} \end{aligned} \quad (8)$$

where Difference (P) is the fitness value of P th candidate solution, which is the difference of the maximum and minimum completion times. Equation (7) shows that for particle P , the difference between the VM which is taking maximum time and the VM which is taking minimum time is calculated. Likewise for all the particles the difference is calculated and the particle which has the minimum difference will be selected as the best particle. The difference value of the best particle will be the value of the Objective function according to the Eq. (8).

In Fig. 2 (right), same particles as of Fig. 2 (left) are taken into consideration. Since the completion time difference between orange color particle is least, it will be selected as best particle and all other particles will try to move toward it. In this scenario all particles will be in a moving state toward each other in successive iterations and convergence will be faster. The minimum and maximum ends will try to move toward each other and in the end all other VMs completion time will also be confined within a smaller gap. Thus, it will help in giving a solution in which tasks spread is proper. The completion time at each VM will be nearly same thus makespan will be automatically minimized.

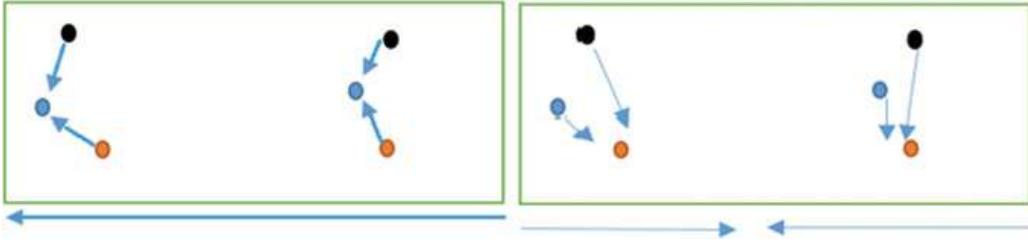


Fig. 2 Particles movement in original objective function (left) and in modified objective function (right)

3.3 Algorithm Flow

The PSO algorithm in load balancing in cloud computing is shown in Fig. 3.

4 Experiment and Results

PSO-SA was implemented on Cloudsim3.0.3 tool [11] to evaluate the result of original objective function and modified objective function. The experiment was performed on High Performance Computing and Networking (HPCN) workload [12] on the task set which required one processing element. Heterogeneous environment consisting of VMs of different computing power (million instruction per second) and RAM size was used in the simulation.

The metrics taken for evaluating the performance of the modified algorithms are Degree of Imbalance (DI) and Standard Deviation (σ). The DI metric measures the imbalances among the completion times of the VMs [8]. Standard deviation σ shows the amount of dispersion from the average. DI and σ are given by the formula 9 and 10.

$$DI = m \times \frac{\max(CTi) - \min(CTi)}{\sum_{i=1}^m CTi} \quad \text{where } i = 1, \dots, m \quad (9)$$

$$\sigma = \sqrt{\frac{1}{m} \times \sum_{i=0}^m \left(CTi - \frac{\sum_{i=1}^m CTi}{m} \right)^2} \quad \text{where } i = 1, \dots, m \quad (10)$$

The experimental results are obtained by running improved PSO-SA and existing PSO-SA for two cases. First for scheduling the workload on 5 VMs and second, for scheduling the workload on 10 VMs. In all the cases 20 particles and 20 iterations are considered.

From the Fig. 4a, b it can be seen that the modified algorithm maintains a near to minimum standard deviation that means VMs are having almost equal completion time. But with the original algorithm, a peak can be seen in the 6th run of Fig. 4a

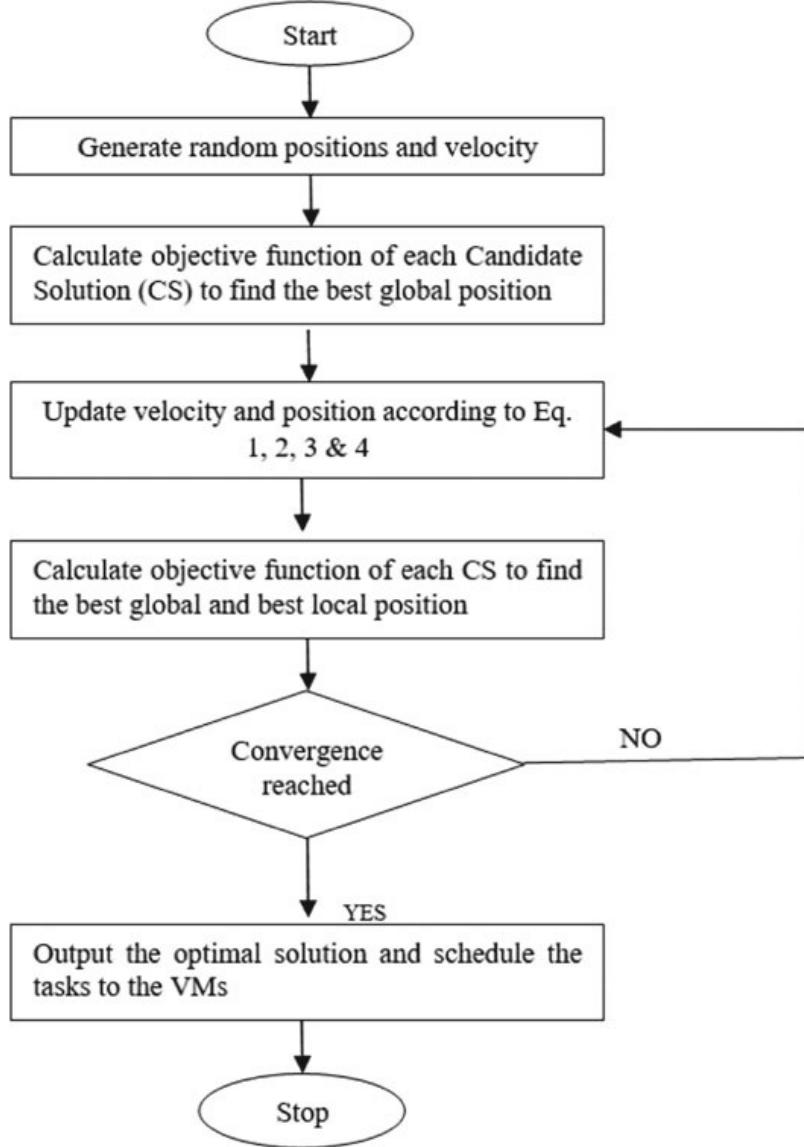


Fig. 3 Algorithm flow

and 3rd and 7th run of Fig. 4b which means that in that runs VMs are having a lot of difference between the completion times which is a sign that load imbalance has occurred. The DI information from Tables 1 and 2 that the imbalance ranges from 0.118 to 0.355 for 5 VMs, from 0.211 to 0.419 for 10 VMs in original algorithm and from 0.117 to 0.199 for 5 VMs, from 0.172 to 0.282 for 10 VMs in modified algorithm concludes that the modified algorithm is consistent in giving the solution which is having lesser standard deviation in completion times. Thus, indicating a proper spread of the tasks where all resources will be utilized with a balance.



Fig. 4 **a** Standard deviation comparison result for 5 VMs, **b** standard deviation comparison result for 10 VMs

Table 1 Experiment result for DI for 5 VMs

Run algorithm	1	2	3	4	5	6	7	8	9	10
Basic	0.193	0.118	0.226	0.140	0.139	0.355	0.181	0.126	0.122	0.234
Modified	0.118	0.129	0.140	0.117	0.140	0.129	0.120	0.191	0.199	0.176

Table 2 Experiment result for DI for 10 VMs

Run algorithm	1	2	3	4	5	6	7	8	9	10
Basic	0.250	0.220	0.399	0.219	0.258	0.211	0.419	0.212	0.200	0.235
Modified	0.199	0.212	0.172	0.250	0.224	0.236	0.211	0.215	0.282	0.232

5 Conclusion

From the experiments and results the conclusion can be drawn that the improved PSO gives consistent results for task scheduling. The modified objective function outperforms the original objective function used in PSO-SA in terms of maintaining the consistency to give optimal solution. Also, when both maximum and minimum

completion time try to reach toward each other the final gap between the two can be minimized as shown by the standard deviation results and thus a load balanced system is achieved.

References

1. Mokoto, E.: Scheduling to minimize the makespan on identical parallel machines: an LP-based algorithm. In: *Investigacion Operativa*, pp. 97–107 (1999)
2. Yu, J., Buyya, R., Ramamohanarao, K.: Workflow scheduling algorithms for grid computing. In: *Metaheuristics for Scheduling in Distributed Computing Environments* Berlin Germany: Springer pp. 173–214 (2008)
3. Geetha, P., Robin, C.R.R.: A comparative-study of load-cloud balancing algorithms in cloud environments. In: *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, Chennai, pp. 806–810 (2017)
4. Khalili, A., Babamir, S.M.: Makespan improvement of PSO-based dynamic scheduling in cloud environment. In: *2015 23rd Iranian Conference on Electrical Engineering, Tehran*, pp. 613–618 (2015)
5. Salman, A., et al.: Particle swarm optimization for task assignment problem. *Microprocess. Microsyst.* **10**, 1–8 (2011)
6. Zhang, L., Chen, Y., Sun, R.: A task scheduling algorithm based on PSO for grid computing. *Int. J. Comput. Intell. Res.* **4**(1), 37–43 (2008)
7. Al-Olimat, H.S., Alam, M., Green, R., Lee, J.K.: Cloudlet scheduling with particle swarm optimization. In: *Proceedings of IEEE 5th International Conference on Communication Systems and Network Technologies*, pp. 991–995 (2015)
8. Saleh, H., Nashaat, H., Saber, W., Harb, H.M.: IPSO task scheduling algorithm for large scale data in cloud computing environment. *IEEE Access* **7**, 5412–5420 (2019)
9. Izakian, H., Ladani, B.T., Zamanifar, K., Abraham, A.: A novel particle swarm optimization approach for grid job scheduling. In: *Inf. Syst. Technol. Manag. Commun. Comput. Inf. Sci.* **31**, 100–109
10. Chong-min, L., Yue-lin, G., Yu-hong, D.: A new particle swarm optimization algorithm with random inertia weight and evolution strategy. *J. Commun. Comput.* **5**(11), 42–48 (2008)
11. Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A.F.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw. Pract. Experience* **41**(1), 23–50 (2010)
12. Feitelson, D.G., Tsafrir, D., Krako, D.: Experience with using the parallel workloads archive. *J Parallel Distrib. Comput.* **7**(1), 2967–2982 (2014)

Parallel Implementation of kNN Algorithm for Breast Cancer Detection



Suhas Athani , Shreesha Joshi , B. Ashwath Rao , Shwetha Rai , and N. Gopalakrishna Kini

Abstract Current advances in parallel processing technology aims at providing unmatched degree of computational power in upcoming days. Parallel computation is an efficient form of information processing which exploits the concurrency of execution. This paper investigates the use of parallel programming, when applied on k Nearest Neighbors (kNN) algorithm which is intended for classification and prediction of the large dataset. Breast cancer dataset is used for classification and prediction which consists of two labels namely, malignant and benign. kNN is a non-parametric algorithm which makes use of similarity measure to classify the dataset into different categories. The similarity between the data points is computed by using Euclidean distance formula. Multiple threads are created for parallel processing and an appropriate kNN graph is constructed, which helps in easier implementation. Finally, execution speeds for sequential and parallel programs is recorded. The results are verified by using frameworks namely, Message Passing Interface (MPI) and Compute Unified Device Architecture (CUDA) highlighting that parallel execution takes less time when compared to sequential execution.

Keywords Euclidean distance · K nearest neighbor · Parallel processing

S. Athani · S. Joshi · B. A. Rao () · S. Rai · N. G. Kini

Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, Karnataka, India
e-mail: ashwath.rao.b@gmail.com

S. Athani

e-mail: suhas2012athani@gmail.com

S. Joshi

e-mail: jshreesha@gmail.com

S. Rai

e-mail: shwetharai.cse@gmail.com

N. G. Kini

e-mail: ng.kini@manipal.edu

1 Introduction

It is important to extract only the dominant data relevant for concerned application. In order to do so, it requires analyzing dataset in the structured manner. The current trend is now to handle large datasets through a computer assisted process of pattern searching and analysis. This process is popularly known as Data Mining. Data Mining requires lot of processing and consumes ample time to produce the output. Through data mining, it is possible to find the important patterns which can be further used for classification and clustering of data points which have common properties. Solving such problems requires more usage of basic computing resources [1]. Thus, parallel computation of tasks will provide an efficient result to such problems.

The important task is to select an appropriate algorithm for data mining. Data mining algorithms tend to find out hidden and closely related patterns in the dataset and then output is predicted. Classification and regression algorithms, which are the most popular type of data mining algorithms, help in identifying the relationships among data elements [2, 3]. One among those algorithms is kNN, known for its non-parametric, supervised and lazy learning nature.

Out of all the classification algorithms, kNN was chosen for several different reasons. This algorithm is extremely helpful as mathematically theoretical assumptions cannot always be followed in real-world datasets. The entire training data is utilized for testing, resulting in a faster training phase but slow and costly testing phase. In the worst case, more time is spent to iterate all the data in kNN, which in turn increases the space complexity for storing training data. This leads to the need to parallelize the kNN algorithm for better efficiency, throughput, usage of resources and quicker execution.

2 Literature Review

In the recent past, many researchers have used parallel programming models to solve data mining, artificial intelligence and machine learning problems. Parallelism of various algorithms belonging to the above domains is implemented using MPI, OpenCL and CUDA parallel processing environments.

In [4], automatic text classification procedure was performed to classify the Web pages containing high volume of information. This was implemented parallelly using kNN and there was increase in the speed-up to 40 times in comparison with sequential kNN program.

In [5], kNN was implemented parallelly on CUDA-enabled GPU to optimize and to achieve maximum performance. CUDA multi-threading concept was used.

In [6], initially kNN graph was constructed. kNN algorithm was parallelized using MPI/OpenMP mixed mode codes. Parallelized kNN was able to take the advantage of multiple processors. Results showed that parallel algorithm was not only faster but also achieved fine quality when compared with sequential algorithm.

In [7], kNN algorithm was used in various fields, particularly for face recognition. Parallel CUDA implementation of kNN was proposed and compared with conventional serial implementation of kNN.

In [8], kNN was parallelly computed using threads, MPI, grid respectively. Three-level architectural model performed better in comparison with serial execution of kNN with speed-up near to 90.

In [9], to reduce the time-consuming process of kNN which involves distance classification by computing the ratio of test data to train data, training data was clustered in an unsupervised manner and then an iterative method was applied to assign a training sample to one of the clusters.

In [10], gene expression data is utilized for classification of cancer, wherein kNN algorithm is implemented to classify cancer types. Accuracy of classification seen to be increased in the range of 7.72 and 16.41%. Mammograms are X-ray pictures used in identifying prior signs of breast cancer. In [11], mammograms are processed by merging high-pass and low-pass filters. Here, the equation of nonlinear polynomial filters (NPF) are expressed as a function of linear term which removes the noise and quadratic term performing mammogram enhancement. The results have shown that this method is efficient in detecting breast cancer. In [12], mammograms are enhanced by using sigmoidal transformation along with gray-level-co-occurrence matrix (GLCM) and classification using Support Vector Machine (SVM) classifier. It further supports two way (malignant and benign) and multi-level classification of tumor types.

3 Proposed Method

The method described here, involves serial and parallel computation of kNN algorithm. When computed in a serial fashion, the running time of an algorithm is equal to the data points present in the dataset. This generates a possibility to replicate the process using parallel processing in order to improve the computational time of the algorithm.

4 kNN Algorithm

Nearest-neighbor classifiers compares similarity of any given tuple to that of the trained tuples. N attributes are used to represent training data tuple, where are represented in an n-dimensional space. When introduced with a new tuple, a kNN classifier searches the pattern space for the k training tuples with least distance from the new tuple. The term closeness used to describe the k ‘nearest neighbors’ of the introduced tuple is with respect to distance value.

The type of distance measure chosen for calculation depends on the type of variable used and the requirement. All the distance measures are used for continuous

Table 1 Various distance measures

Euclidean	$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$
Manhattan	$\sum_{i=1}^k x_i - y_i $
Minkowski	$\left(\sum_{i=1}^k (x_i - y_i)^{q^{1/q}} \right)$

variables and Hamming distance is used for categorical variables. In the proposed system, Euclidean distance is computed using the formula as shown in the Table 1.

In kNN, k represents the number of nearest neighbors, which is the core deciding factor. If the label of a point ‘X’ has to be predicted, the closest point to X is found whose label is known, and the same label is assigned to X as well.

kNN algorithm has basically three steps, as shown in Fig. 1. [13]. It involves calculating distance, finding closest neighbors and voting for labels.

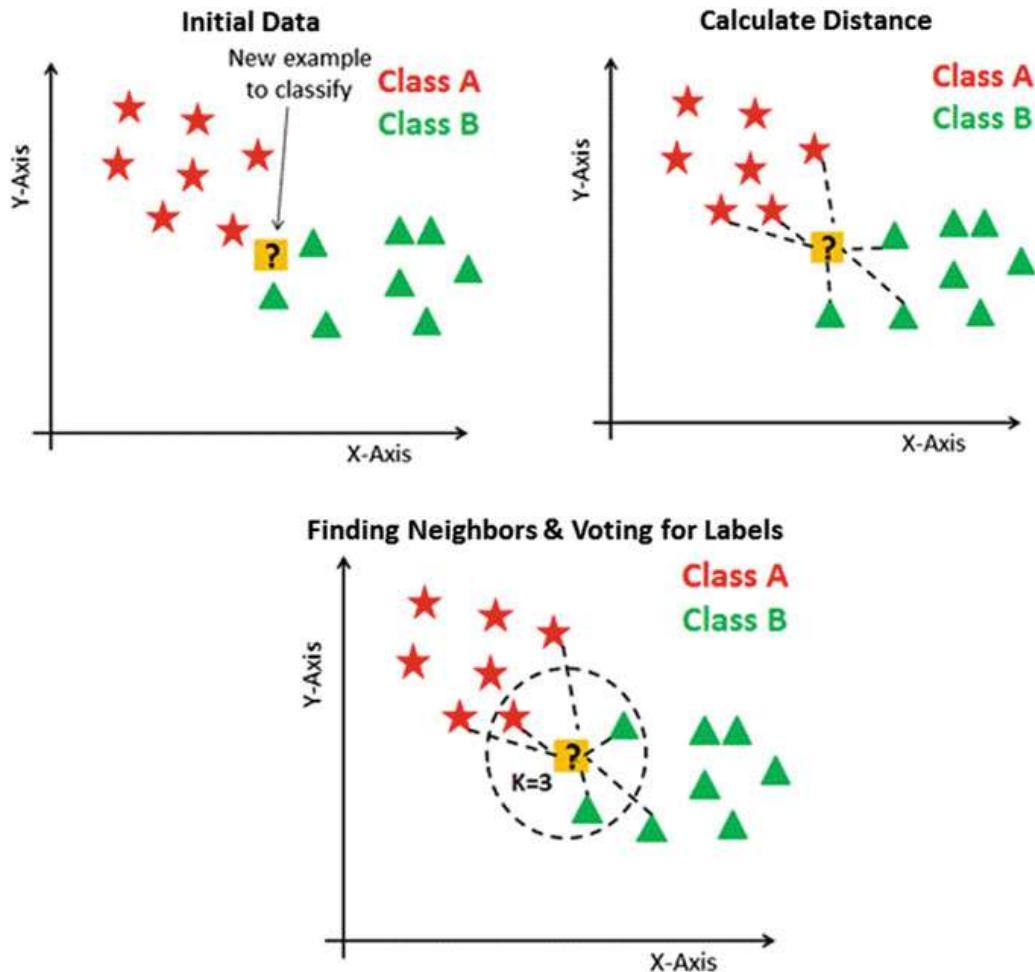


Fig. 1 Steps involved in the kNN algorithm

```

Algorithm 1: To compute Euclidean distance
Input: first, second, size
Output: distance
    distance = 0
    for x, y in range(size)
        distance = (first[x] - second[x])2 + (first[y] - second[y])2
        result =  $\sqrt{distance}$ 
    return result

```

Fig. 2 Algorithm to compute euclidean distance

5 Methodology

The Wisconsin Breast Cancer dataset [14] is split into two CSVs, one as training dataset and the other as test dataset to the kNN algorithm. For processing using MPI, the training dataset is scattered to all the available processes evenly. Each processor creates separate threads which work on the kNN algorithm simultaneously improving efficiency, resource utilization and reduce the amount of time taken to execute. Each processor calculates the distance between the test datapoint and the existing datapoint by using Euclidean distance in parallel.

To determine the similarity between two instances, Euclidean distance function is used as shown in Fig. 2.

1D grid of 1D blocks are used in the CUDA kernel. Number of blocks required are determined by keeping the number of threads created in each block constant. The kernel is used to calculate the distance of the test datapoint to the other training datapoints.

Since all pairs of objects are independent of the others, it is possible to fully parallelize the distance computation. The concept of data-parallelism is used for implementation of the kernel defined for distance calculation. After transferring the data in the n-dimensional space, the proposed algorithm uses each thread created to calculate the distance between the object of interest and a reference object using the kernel function. A unique segment of the reference dataset is loaded into the shared memory from the global memory. Since the number of reference objects is huge, many threads and blocks will be launched in this kernel. The test datapoint, in both MPI and CUDA, will receive the label like the present majority labels of the k nearest neighbors. Figure 3, shows the parallel implementation of kNN algorithm.

6 Results and Discussion

The dataset consists of 32 attributes and there are two classes which are malignant and benign. Malignant and benign are the two types of tumor which are developed when cell reproduction rate is high. Tumors vary in size and type. Malignant tumors are dangerous as they are cancerous. The cells can grow and spread to the other parts

Fig. 3 Parallel algorithm for kNN

Algorithm 2: Algorithm to parallelize *knn*

Input:

- D_{tr} : training data
- D_{te} : testing data
- M : Malignant
- B : Benign
- p : probability value
- P_i : Processes where $i = 1, 2, 3 \dots n$

Output: Class Labels

```

for  $P_i$ ,  $0 < i < n$ 
     $P_i = D_{te}(i)$ 
    Euclidean Distance is computed parallelly
     $a = \text{euclidean distance}(D_{te}, D_{tr}[i])$ 
    merge sort( $a$ )
    for  $i = 0$  to  $K$ , where  $K$  is number of neighbors
         $b[i] = D_{tr}(\text{Label})$ 
        if ( $p(M) > p(B)$ ) then
            do  $D_{te}(\text{label}) = M$ 
        else
             $D_{te}(\text{label}) = B$ 

```

of the body. Benign tumors are non-cancerous. They neither can grow nor spread. If they are diagnosed, then there is a less probability of them recurring in future [15]. Serial and parallel kNN algorithm are used to classify the records into these two classes.

Figure 4, shows the n-dimensional data reduced to two-dimensional plane using the technique of linear discriminant analysis. Parameters passed for sequential execution of the kNN, are the dataset containing the 31 attributes, ratio of train and test dataset and number of clusters to be formed. Parameters passed for the parallel kNN module are same as that of serial execution except that, the threads are created for parallel execution in MPI and CUDA. Both these techniques utilize the available processors to distribute the workload and hence complete the task in much lesser time.

The execution time with parallel kNN was nearly half the time taken for sequential execution and execution of kNN using CUDA was further reduced. Table 2, shows the execution time of an algorithm in MPI and CUDA with $k = 3$.

Figure 5 shows the mean error rate of the parallel kNN algorithm for various values of k, where k value is varied from 0 to 40. It can be seen that the error rate

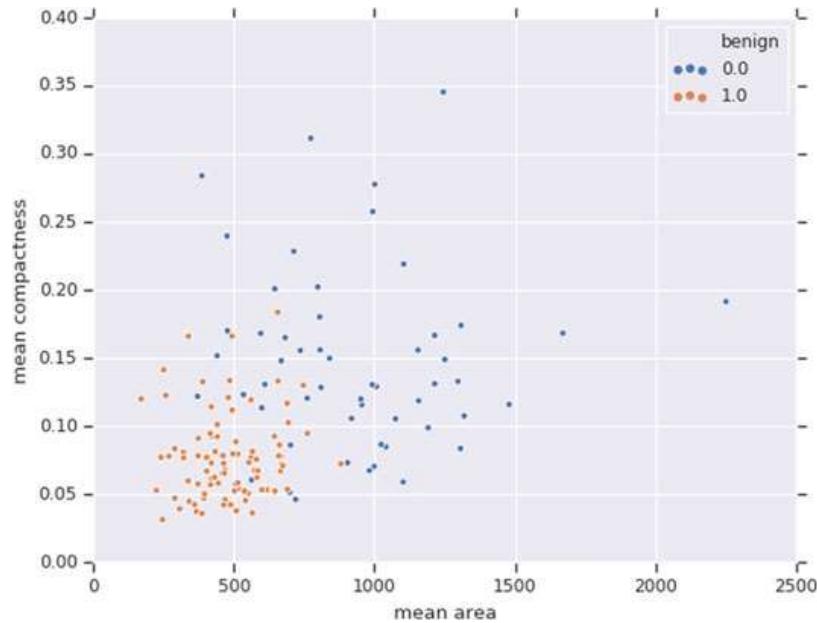


Fig. 4 Visualization of n-dimensional data using LDA

Table 2 Execution time for MPI versus CUDA with $k = 3$

Test dataset entries	MPI execution time (in ms)	CUDA execution time (in ms)
15	11.2	1.8
30	22	2.1
60	39.1	2.2
90	61.4	2.3
120	77.0	2.8

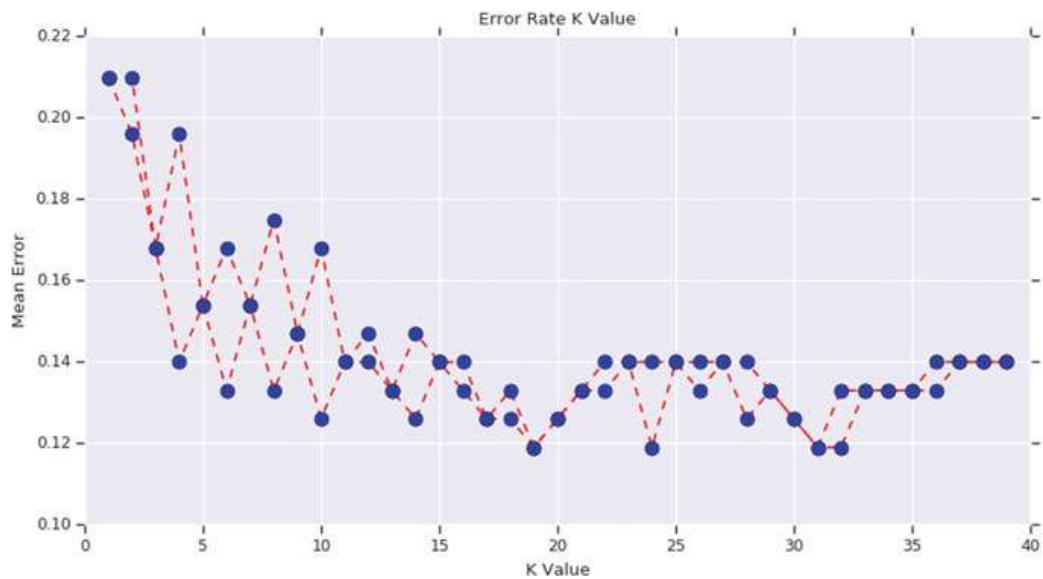


Fig. 5 Relation between k value and mean error rate

is reduced as the value of k increases. This is because the influence of noise reduces with the increase in the value of k .

7 Conclusion

kNN algorithm is parallelized using MPI and CUDA frameworks. The implementation of both the parallel techniques consumed less amount of time and resources, making them more efficient than the sequential version of the algorithm. In future, it can be designed and implemented to work in a parallel environment with lesser communication overhead.

References

1. Cheung, D.W., Lee, S.D., Xiao, Y.: Effect of data skewness and workload balance in parallel data mining. *IEEE Trans. Knowl. Data Eng.* **14**(3) (2002)
2. Gavahi, M., Mirzaei, R., Nazarbeygi, A., Ahmadzadeh, A., Gorgin, S.: High performance GPU implementation of k-NN based on mahalanobis distance. In: 2015 International Symposium Computer Science and Software Engineering (CSSE), pp. 1–6 (2015)
3. Cover, T., Hart, P.: Nearest neighbor pattern classification. *IEEE Trans. Inf Theory* **13**(1), 21–27 (1967)
4. Huang, L., Li, Z.: A novel method of parallel gpu implementation of knn used in text classification. In: 2013 Fourth International Conference on Networking and Distributed Computing (ICNDC), pp. 6–8 (2013)
5. Shenshen, L., et al.: CUKNN: a parallel implementation of K-nearest neighbor on CUDA-enabled GPU. In: IEEE Youth Conference on Information Computing and Telecommunication 2009, YC-ICT'09, (2009). IEEE
6. Wang, D., Zheng, Y., Cao, J.: Parallel construction of approximate kNN Graph. In: Proceeding of IEEE DCABES, pp. 22–26 (2012)
7. Despotovski, F., Gusev, M., Zdraveski, V.: Parallel implementation of k-nearest-neighbors for face recognition. In: 26th Telecommunications Forum (TELFOR), pp. 1–4 (2018)
8. Aparicio, G., Blanquer, I., Hernández, V.: A parallel implementation of the k nearest neighbours classifier in three levels: threads, MPI processes and the grid. In: Proceedings of the 7 International Conference on High Performance Computing For Computational Science, pp. 225–235 (2006)
9. Vajda, S., Santosh, K.C.: Fast k-nearest neighbor classifier using unsupervised clustering. In: Recent Trends in Image Processing and Pattern Recognition, pp. 185–193 (2016)
10. Chaudhari, P., et al.: Data augmentation for cancer classification in oncogenomics: an improved kNN based approach. In: Evolutionary Intelligence, Springer, Germany, pp. 1–10 (2019)
11. Bhateja, V., Misra, M., Urooj, S.: Non-Linear polynomial filters for contrast enhancement of mammograms. In: Non-Linear Filters for Mammogram Enhancement. Springer, Singapore, pp. 123–162 (2020)
12. Bhateja, V., et al.: Classification of mammograms using sigmoidal transformation and SVM. In: Smart Computing and Informatics. Springer, Singapore, pp. 193–199 (2018)
13. Navlani, A.: KNN Classification using Scikit-learn. DataCamp Community, 2019. (Online). Available: <https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn>. Accessed 01 Oct 2019

14. UCI machine learning repository: breast cancer wisconsin (Original) Data Set Archive.ics.uci.edu, (Online). Available: [https://archive.ics.uci.edu/ml/datasets/breast+can+cer+wisconsin+\(original\)](https://archive.ics.uci.edu/ml/datasets/breast+can+cer+wisconsin+(original)) (2019). Accessed: 17 Sep 2019
15. Christina Chun M.: Tumors: benign, premalignant, and malignant, Medical News Today, (Online). Available: <https://www.medicalnewstoday.com/articles/249141.php> (2019). Accessed 17 Sep 2019

Correlated High Average-Utility Itemset Mining



Krishan Kumar Sethi and Dharavath Ramesh

Abstract High average-utility itemset (HAUI) mining is an advancement over high utility itemset mining, where average-utility is used instead of utility measure to discover meaningful patterns. It has been discussed in several past studies that significance of utility-based patterns can be amplified if items in the patterns are correlated. In this paper, we propose a HAUI mining algorithm named correlated high average-utility itemset (CoHAI) miner which provides highly profitable and non-redundant correlated patterns. CoHAI miner is a one-phase algorithm which uses a vertical list structure to store utility and correlation information to apply pruning and candidate generation. Moreover, use of vertical list avoids repetitive dataset scans to calculate the utility of candidate itemsets. We performed rigorous experiments to compare the CoHAI miner with existing algorithm. The results show that the CoHAI miner performs efficiently and produces more meaningful patterns than existing algorithm.

Keywords High average-utility itemset mining · Correlation · Search space · Pattern mining

1 Introduction

High utility itemset (HUI) mining [3, 12, 17, 19] is an important pattern mining technique as it reveals most profitable patterns in a transaction dataset. HUI mining methods use per unit profit and quantity of item to define meaningful itemsets. The utility measure does not follow downward closure (DC) property. Therefore, a new term known as transaction weighted utility (TWU) [17] has been proposed to retain the DC property. Most of HUI mining algorithms use TWU measure to apply the

K. K. Sethi · D. Ramesh ()

Indian Institute of Technology (ISM) Dhanbad, Dhanbad, Jharkhand 826004, India
e-mail: ramesh.d.in@ieee.org

K. K. Sethi
e-mail: kksethi@ieee.org

pruning method. Various HUI mining algorithms [5, 10, 11, 16] discover potential candidate itemsets and HUIs in single phase using a list structure.

HUI mining has a major flaw that long itemsets have more chances to be HUI regardless their usefulness. Moreover, in the real-life applications, lengthy itemsets are not used for decision making. Therefore, to mitigate the impact of length constraints, a novel evaluator called as average-utility has been introduced. Pattern mining based on average-utility is known as high average-utility itemset mining [9, 14, 21]. HAUI mining techniques cannot follow the DC property using the TWU model. Therefore, a new upper-bound called average-utility upper-bound (AUUB) [9] has been proposed to acquire the DC property. Later, EHAUPM [15] has introduced two reduced upper bounds named looser upper-bound (LUB), revised tighter upper-bound (RTUB), and various efficient pruning methods to enhance the effectiveness of mining procedure.

In various studies, it has been argued that pattern mining using utility constraint suffers from a major limitation that items in the patterns may be weakly correlated. In real-life applications, it has been observed that tightly correlated patterns provide better platform for decision making. In the past, several correlation measures, e.g., support [1], frequency affinity [2], confidence [8], all-confidence [18] and coherence [18] have been widely used for pattern mining. In the area of utility-based pattern mining, HUIPM [2] and FDHUP [13] algorithms were introduced to discover HUIs with strong frequency affinity. Recently, a inherent correlated high utility pattern mining algorithm CoHUIM [7] has been proposed which uses a measure named kulg [20] as a correlation factor. Then, a one-phase algorithm named redundant correlated high utility pattern miner (CoUPM) [6] has been developed which uses positive correlation and utility measure to define correlated high utility patterns. It uses a vertical list structure to store the required information of itemsets which can be used to apply pruning and candidate generation without additional dataset scans.

However, it has been proved that high average-utility patterns are more meaningful than high utility patterns. In this paper, we introduce a one-phase correlated high average-utility itemset mining algorithm named CoHAI miner. The CoHAI miner introduces a vertical list structure named support average-utility (SAU) list to accumulate the information about utility and correlation of the itemset. We adopt the efficient looser upper-bound (LUB) [15] instead of AUUB model for pruning. Various detailed experiments have been executed to evaluate the performance of CoHAI miner compared to EHAUPM. The results show that the CoHAI miner outperforms EHAUPM and searches more meaningful patterns.

We organize remaining paper in the following sequence. Section 2 contains various prerequisites and problem definition. Section 3 describes the proposed CoHAI miner. Section 4 induces performance evaluation of proposed algorithm followed by conclusion in the last section.

2 Problem Definition

Let $D = \{t_1, t_2, \dots, t_n\}$ be a dataset which has n transactions and m total items $I = \{i_1, i_2, \dots, i_m\}$. A transaction has a transaction id, i.e., tid by which it is uniquely identified. In dataset D , every item has a quantity of purchase which is known as the internal utility. While a separate profit table called external utility is the part of input, where item to per unit profit mapping is stored. A collection of items is called itemset which is denoted as k -itemset, where k is cardinality of itemset. A sample dataset and profit table are depicted in Tables 1 and 2 which are employed as running example in rest of the paper.

Definition 1 Utility of an item a in transaction t_q is expressed as $u(a, t_q)$ and defined as: $u(a, t_q) = iu(a, t_q) \times eu(a)$, where, $iu(a, t_q)$ denotes internal utility of a in transaction t_q while $eu(a)$ refers to external utility of a . Utility of a k -itemset A in transaction t_q and in whole dataset can be computed as follows.

$$u(A, t_q) = \sum_{a \in A \wedge A \subseteq t_q} u(a, t_q) \quad (1)$$

$$u(A) = \sum_{A \subseteq t_q \wedge t_q \in D} u(A, t_q) \quad (2)$$

For example, in the running dataset, $u(ad, t_1) = u(a, t_1) + u(d, t_1) = 4 + 3 = 7$.

Table 1 Sample dataset

tid	Transaction
t_1	a1; c3; d1; f1
t_2	a2; b3; d1; f2
t_3	b4; d2; e1; f2
t_4	a3; b2; c1; d1; f 1
t_5	a2; b2; c1; f2
t_6	a1; b3; f1

Table 2 Profit table

Items	Per unit profit
a	4
b	2
c	1
d	3
e	1
f	3

Definition 2 Average-utility of k -itemset A in transaction t_q , i.e., $au(A, t_q)$ is calculated as follows.

$$au(A, t_q) = \frac{u(A, t_q)}{|A|} = \frac{\sum_{a \in A \wedge A \subseteq t_q} u(a, t_q)}{k} \quad (3)$$

Average-utility of A in dataset D is computed as follows.

$$au(A) = \sum_{A \subseteq t_q \wedge t_q \in D} au(A, t_q) \quad (4)$$

For example, $u(ad) = \frac{u(ad, t_1) + u(ad, t_2) + u(ad, t_4)}{2} = \frac{33}{2} = 16.5$.

Definition 3 Transaction maximum utility for a transaction t_q , i.e., $tmu(t_q)$ is defined as: $tmu(t_q) = \max\{u(a, t_q) | a \in t_q\}$.

Definition 4 Average-utility upper-bound for an itemset A is expressed as $auub(A)$ and calculated as follows.

$$auub(A) = \sum_{A \subseteq t_q \wedge t_q \in D} tmu(t_q) \quad (5)$$

For example, $auub(ad) = tmu(t_1) + tmu(t_2) + tmu(t_4) = 4 + 8 + 12 = 24$.

Property 1 AUUB downward closure property: Let B be a superset of itemset A , i.e., $A \subseteq B$ then AUUB of B cannot be larger than AUUB of A , $auub(B) \leq auub(A)$. It can also be used to avoid processing of weak candidate itemsets in the search space.

Definition 5 kulc measure: The correlation factor kulc for k -itemset A , i.e., $kulc(A)$ is calculated as follows.

$$kulc(A) = \frac{1}{k} \sum_{i_j \in A} \frac{\sup(A)}{\sup(i_j)} \quad (6)$$

where, i_j is j^{th} item in A and $\sup(A)$ is support value of A . Value of $kulc(A)$ varies in the range of $[0, 1]$ which can be used to find strong correlation in the patterns. For example, $kulc(ad) = \frac{1}{2} \cdot \left(\frac{\sup(ad)}{\sup(a)} + \frac{\sup(ad)}{\sup(d)} \right) = \frac{1}{2} \cdot \left(\frac{3}{5} + \frac{3}{4} \right) = 0.67$.

Definition 6 Correlated high average-utility itemset (CoHAI): An itemset A is called as CoHAI if A follows these two constraints: first, average-utility of A is no lesser than minimum average-utility threshold; second, correlation factor kulc of A is no less than minimum correlation threshold. Let CoHAI denotes the correlated HAUIs then CoHAI is defined as follows.

$$\text{CoHAI} \leftarrow \{A | au(A) \geq minUtil \wedge kulc(A) \geq minCor\} \quad (7)$$

3 Proposed Algorithm: CoHAI Miner

3.1 CoHAI Miner Properties

In CoHAI miner, we adopt a reduced upper-bound called looser upper-bound (LUB) from EHAUPM [15] which has been defined below.

Definition 7 Revised dataset: In the past HAUI mining algorithms, AUUB ascending order has been used to revise the dataset and also define the processing order of items. However, CoHAI miner uses the correlation factor kulg and requires to follow the sorted downward closure property for the same. Therefore, whole dataset is revised by sorting the items in ascending order of their support values.

Definition 8 Remaining Maximum Utility of an itemset $A = \{i_1, i_2, \dots, i_j\}$ in transaction t_q , i.e., $\text{remu}(A, t_q)$ is calculated as follows.

$$\text{remu}(A, t_q) = \max\{u(i_{j+1}), u(i_{j+2}), \dots, u(i_N)\} \quad (8)$$

where, N be the total number of items in transaction t_q . For example, $\text{remu}(cb, t_4) = 3 (< \text{tmu}(cb, t_2) = 12)$.

Definition 9 Looser upper-bound (LUB) for itemset A in transaction t_q , i.e., $\text{lub}(A, t_q)$ is computed as follows.

$$\text{lub}(A, t_q) = \frac{u(A, t_q) + |A| \times \text{remu}(A, t_q)}{|A|} \quad (9)$$

LUB for itemset A in entire dataset D is computed as follows.

$$\text{lub}(A) = \sum_{A \in t_q \wedge t_q \subseteq D} \text{lub}(A, t_q) \quad (10)$$

For example, $\text{lub}(cb, t_4) = \frac{5+2 \times \text{remu}(cb, t_4)}{2} = 5.5 (< \text{auub}(cb, t_4) = 12)$.

Property 2 Anti-monotonicity property of LUB: Let A and B be two itemsets such that B be the superset of A , i.e., $A \subseteq B$. LUB follows the anti-monotonicity property if it can be obtained $\text{au}(B) \leq \text{lub}(A)$.

Proof Proof to this property is given in [15].

3.2 Support Average-Utility List

In the proposed SAU list for an itemset, each tuple stores three measures named transaction ID (tid), utility (u) in tid, and remaining maximum utility (remu) in tid.

(a) SAU list of 1-itemsets {c, d, a}

c		
sup=3		
<i>tid</i>	<i>u</i>	<i>remu</i>
1	3	4
4	1	12
5	1	8

d		
sup=4		
<i>tid</i>	<i>u</i>	<i>remu</i>
1	3	4
2	3	8
3	6	8
4	3	12

a		
sup=5		
<i>tid</i>	<i>u</i>	<i>remu</i>
1	4	3
2	8	6
4	12	4
5	8	6
6	4	6

(b) SAU list of 2-itemsets {cd, ca}

cd		
sup=2		
<i>tid</i>	<i>u</i>	<i>remu</i>
1	6	4
4	4	12

ca		
sup=3		
<i>tid</i>	<i>u</i>	<i>remu</i>
1	7	3
4	13	4
5	9	6

Fig. 1 SAU list

Moreover, SAU list also stores the total support count of the itemset to calculate the kulc value. Initially, the dataset is scanned to construct the SAU list of each 1-itemset. Thereafter, SAU list of 2-itemset can be constructed by joining of SAU list of subset 1-itemset. Similarly, joining of SAU list of $(k - 1)$ -itemsets is performed to construct the SAU list of k -itemset. Let, $x.\text{SAUL}$ and $y.\text{SAUL}$ be SAU list of respective itemsets x and y . To construct the SAU list of itemset xy , join operation is performed in the following manner.

- All the common tids are found in $x.\text{SAUL}$ and $y.\text{SAUL}$ and assigned to *tid* field of $xy.\text{SAUL}$. The tids in the SAU list are sorted which effectively reduce the searching time.
- For each common tid, utility value in $xy.\text{SAUL}$ is the sum of utility in $x.\text{SAUL}$ and $y.\text{SAUL}$.
- The *remu* value of $y.\text{SAUL}$ is assigned to *remu* field in $xy.\text{SAUL}$.
- Count of *tids* in $xy.\text{SAUL}$ is assigned as support of itemset xy .

In the running example, SAU list of 1-itemsets {c, d, a} is depicted in Fig. 1a. SAU list of 2-itemsets {cd, ca} is constructed by joining of 1-itemsets as shown in Fig. 1b.

3.3 Pruning Strategies

The search space of the CoHAI miner is depicted in a tree structure called set enumeration tree [16]. It accommodates all the possible candidate itemsets(2^n for n items). In the absence of any pruning technique, whole search space has to be scanned to find the meaningful patterns which requires huge time and processing resources. Therefore, a set of pruning methods is required to discard various unpromising candidates in the tree. Following pruning methods are used in the proposed CoHAI miner.

Property 3 Sorted downward closure property of kulc: Let $A = \{i_1, i_2, \dots, i_j\}$ and $B = \{i_1, i_2, \dots, i_j, i_{j+1}\}$ be two itemsets such that $A \subseteq B$. The kulc measure follows the closure property if items in the transaction are sorted in ascending order of support

values [7], i.e., $\text{kulc}(B) \leq \text{kulc}(A)$. This property also ensures the anti-monotonicity property of kulc measure.

Proof A detailed proof is given in [7].

Pruning method 1 (Pruning using sorted DC property of kulc): While exploring search space in depth first search manner, if kulc of any itemset at a node is lesser than minCor then neither that itemset nor any extension from the child node is CoHAI. Therefore, such candidate itemset can be ignored while exploring the search space. Property 3 proves the validation of this pruning strategy.

Pruning method 2 (Pruning using LUB): A itemset A and all its extensions in the search space are discarded if LUB value of A is lesser than minUtil . Therefore, such weak candidate itemsets need not to be processed. Property 2 validates this pruning method.

Pruning method 3 (LA pruning): Let A_a and A_b be two itemsets. The join operation is only performed when value of LALU in the following equations is larger or equal to minUtil . The details of this strategy have been given in [15].

$$\text{LALU}(A_a) = \text{lub}(A_a) - \sum_{t_q \in \text{tids}(A_a) \wedge t_q \notin \text{tids}(A_b), t_q \in D} au(A_a, t_q) + \text{remu}(A_a, t_q) \quad (11)$$

3.4 CoHAIM Design

The main procedure of CoHAI miner has been depicted in Algorithm 1. Initially, CoHAI miner scans the dataset D and calculates the $auub$ value of each item $a \in I$ (line 1). The items with $auub$ lower than minimum utility threshold minUtil are discarded and remaining items are assigned to I^* (line 2). List of items in I^* are arranged in ascending order of their support values (line 3). Thereafter, dataset D is revised and SAU list for each items in I^* is constructed (lines 4–5). Then, a recursive method $\text{search}()$ is invoked to find the CoHAIs of various length and returned as output (lines 6–7).

Algorithm 1 CoHAI Miner

INPUT: D : Transaction dataset; minUtil : Minimum average-utility
 minCor : Minimum correlation

OUTPUT: CoHAI : List of correlated high average-utility itemsets

- 1: Scan dataset D and calculate the AUUB of all 1-itemsets $a \in I$.
 - 2: Search all the 1-itemset $a \in I$ which qualify the constrain $auub(a) \geq \text{minUtil}$ and accommodate these items in I^* .
 - 3: Sort the list I^* in support ascending order of items.
 - 4: Scan the dataset D again to revise the dataset.
 - 5: Construct SAU list of all the itemsets from I^* .
 - 6: Invoke $\text{search}(null, I^*, \text{minUtil}, \text{minCor}, 1)$
 - 7: Output (CoHAI)
-

The search() function takes five inputs and produces a set of CoHAIs in output as depicted in Algorithm 2. Initially, A is set to null because 1-itemsets have no prefix. The algorithm explores the search space in the depth first search manner for each itemset P in $\text{extensionOf}A$ (line 1). The $kulc$ value of itemset P is calculated using its support value (line 2). The itemset P is assigned to output set CoHAI if average-utility and $kulc$ value of P are no lesser than respective minUtil and minCor (lines 3–5). Thereafter, pruning methods are applied using looser upper-bound and $kulc$ value of P . Any extension of P is not checked if lub and $kulc$ values of P are lesser than respective minimum threshold values (line 6). Otherwise, another itemset Q from $\text{extensionOf}A$ is extended with P and SAU list of extended item APQ is constructed using a construct() function (lines 7–10). The pseudocode of construct() function is shown Algorithm 3. Thereafter, the extension list of P is calculated (line 12). Then, search() function is called recursively from new values of prefix and $\text{extensionOf}P$ (line 15).

Algorithm 2 Searching of CoHAIs: search()

INPUT: A : SAU list of itemset A , $\text{extensionOf}A$: SAU list of 1-extensions of A ,
 minUtil : Minimum average-utility, minSup : Minimum support,
 len : length of current itemset

OUTPUT: CoHAI : A set of correlated high average-utility itemsets

```

1: for each itemset  $P$  in  $\text{extensionOf}A$  do
2:   Calculate the  $kulc(P)$  using the  $P.\text{sup}$ 
3:   if  $\frac{\text{sum}(P.\text{u})}{\text{len}} \geq \text{minUtil}$  AND  $kulc(P) \geq \text{minCor}$  then
4:      $\text{CoHAI} \leftarrow \text{CoHAI} \cup P$ 
5:   end if
6:   if  $\text{sum}(P.\text{lub}) \geq \text{minUtil}$  AND  $kulc(P) \geq \text{minCor}$  then
7:      $\text{extensionOf}P \leftarrow \text{Null}$ 
8:     for each itemset  $Q$  in  $\text{extensionOf}A$  do
9:        $APQ \leftarrow AP \cup AQ$ 
10:       $APQ \leftarrow \text{construct}(A, P, Q, \text{minUtil})$ 
11:      if  $APQ$  is not empty then
12:         $\text{extensionOf}P \leftarrow \text{extensionOf}P \cup APQ$ 
13:      end if
14:    end for
15:     $\text{search}(P, \text{extensionOf}P, \text{minUtil}, \text{minCor}, \text{len} + 1)$ 
16:   end if
17: end for
```

4 Experiments and Performance Evaluation

We compare the performance of proposed CoHAI miner to EHAUPM. All experiments were performed on a workstation configured with core i7-7700HQ clocked @ 3.8 GHz, 16 GB DDR4 RAM, and Ubuntu 16.04 operating system. We use four benchmark datasets chess, accidents, retail, and mushroom for this experiment [4]. Various dataset properties are depicted in Table 3.

Algorithm 3 construct() method

INPUT: A = SAU list of itemset A , P = SAU list of extension of A with item x , Q = SAU list of extension of A with item y , $minUtil$ = minimum utility threshold

OUTPUT: APQ = SAU list of itemset APQ

```

1:  $APQ = null$ 
2:  $sup = 0$ 
3: for each tuple  $Ex \in P.SAUL$  do
4:   if  $\exists Ey \in Q.SAUL \wedge Ex.tid = Ey.tid$  then
5:      $sup = sup + 1$ 
6:     if  $A.SAUL \neq null$  then
7:       Find element  $E \in A.RAUL$  such that  $E.tid = Ex.tid$ 
8:        $Exy \leftarrow <Ex.tid, Ex.u + Ey.u - E.u, Ey.remu>$ 
9:     else
10:     $Exy \leftarrow <Ex.tid, Ex.u + Ey.u, Ey.remu>$ 
11:  end if
12:   $AQP.SAUL \leftarrow APQ.SAUL \cup Exy$ 
13: else
14:    $lubOfP = lub(P) - Ex.remu$ 
15:   if  $lubOfP < minUtil$  then
16:     return  $null$ 
17:   end if
18: end if
19: end for
20:  $APQ.support = sup$ 
21: return  $Pxy.RAUL$ .

```

Table 3 Datasets

Dataset	Transactions count	Items count	Average length	Type
Mushroom	8124	119	23	Dense
Chess	3196	75	37	Dense
Accidents	340,183	468	33.8	Dense
Retail	88,162	16,407	10.3	Sparse

4.1 Run-Time Test

Here, run-time performance of CoHAI miner and EHAUPM [15] has been compared by varying $minUtil$ and fixing value of $minCor$. Value of the $minCor$ for chess, mushroom, accidents, and retail dataset has been taken as 0.65%, 0.24%, 0.45% and 0.1%, respectively. The results are depicted in Fig. 2. It can be noticed that run time of both the algorithms is reduced as the value of $minUtil$ grows. The reason is that when $minUtil$ grows lesser itemsets qualify the threshold. Moreover, It can also be observed from the graphs that CoHAI miner outperforms to EHAUPM for all the dataset. For example, for chess dataset, EHAUPM terminates in 1150 s while CoHAI

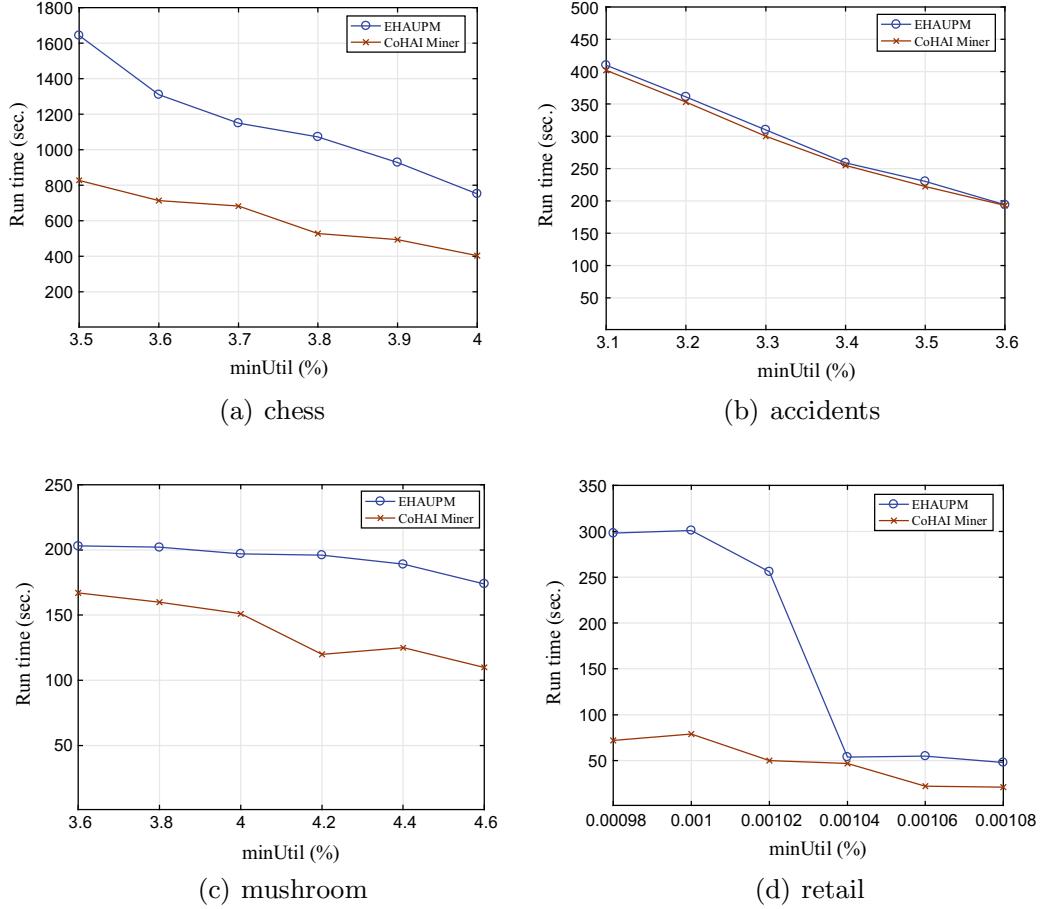


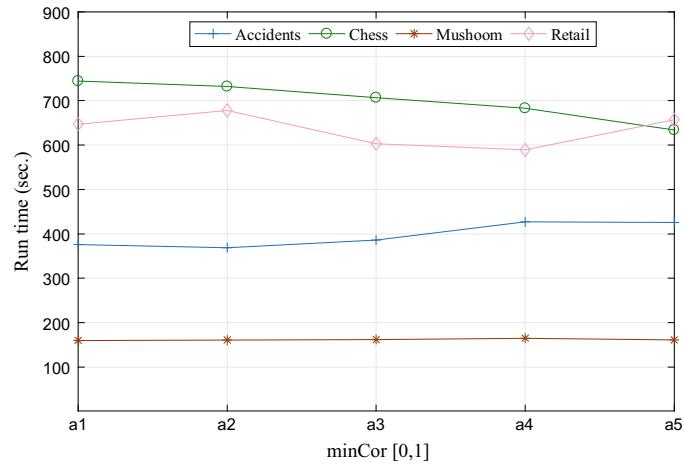
Fig. 2 Runtime performance for fixed $minCor$ and various $minUtil$

miner completes in 682s when $minUtil$ is set to 3.8%. The reason is that CoHAI miner applies one additional constrain of minimum correlation which discards a lot of candidate itemsets from the search space. However, it can be seen for the accidents dataset that run time difference for both the algorithm is very small. It is because of only a few candidates are pruned by correlation constraint in CoHAI miner. Therefore, it can be said that all the datasets have different characteristics and item distribution which affect the algorithm performance of different datasets.

We also performed the runtime evaluation by fixing the value of $minUtil$ and varying the values of $minCor$. We run both the algorithms for five different values of $minCor$ (a1, a2, a3, a4, a5) as shown in Table 4. The results for all the datasets are shown in Fig. 3. It can be observed from the results that run time of CoHAI miner decreases very small with the increase in value of $minCor$.

Table 4 Values of $minCor$

Dataset	$minUtil$ (%)	a1	a2	a3	a4	a5
Accidents	3.1	0.1	0.15	0.2	0.25	0.3
Chess	3.5	0.1	0.2	0.3	0.4	0.5
Retail	0.001	0.25	0.35	0.45	0.55	0.65
Mushroom	0.38	0.1	0.2	0.3	0.4	0.5

Fig. 3 Runtime test for fixed $minUtil$ and various $minCor$ 

4.2 Candidates Count

Here, we compare a number of candidate itemsets generated for both EHAUPM and CoHAI miner. We set value of $minCor$ constant and vary value of $minUtil$ to perform this experiment. Values of both $minUtil$ and $minCor$ have been taken same as the last experiment in 4.1. It can be observed from the results in Fig. 4 that the number of candidates for both the algorithms reduces as value of $minUtil$ increases. The reason is that lesser itemsets qualify the $minUtil$ constrain if value of $minUtil$ is increased. It can be also seen that CoHAI miner generates lesser number of candidate itemsets for all the datasets compared to EHAUPM. The reason is same as mentioned in Sect. 4.1.

5 Conclusions

In this paper, a correlation and average-utility-based pattern mining algorithm named CoHAI miner has been presented. The CoHAI miner is a novel algorithm in the field of pattern mining which uses correlation factor kulg to ensure the strong correlation in the patterns and average-utility measure to discover profitable itemsets. The algorithm is one phase in nature and uses a vertical list structure called SAU list to store the required information for pruning and average-utility calculation. Moreover, SAU

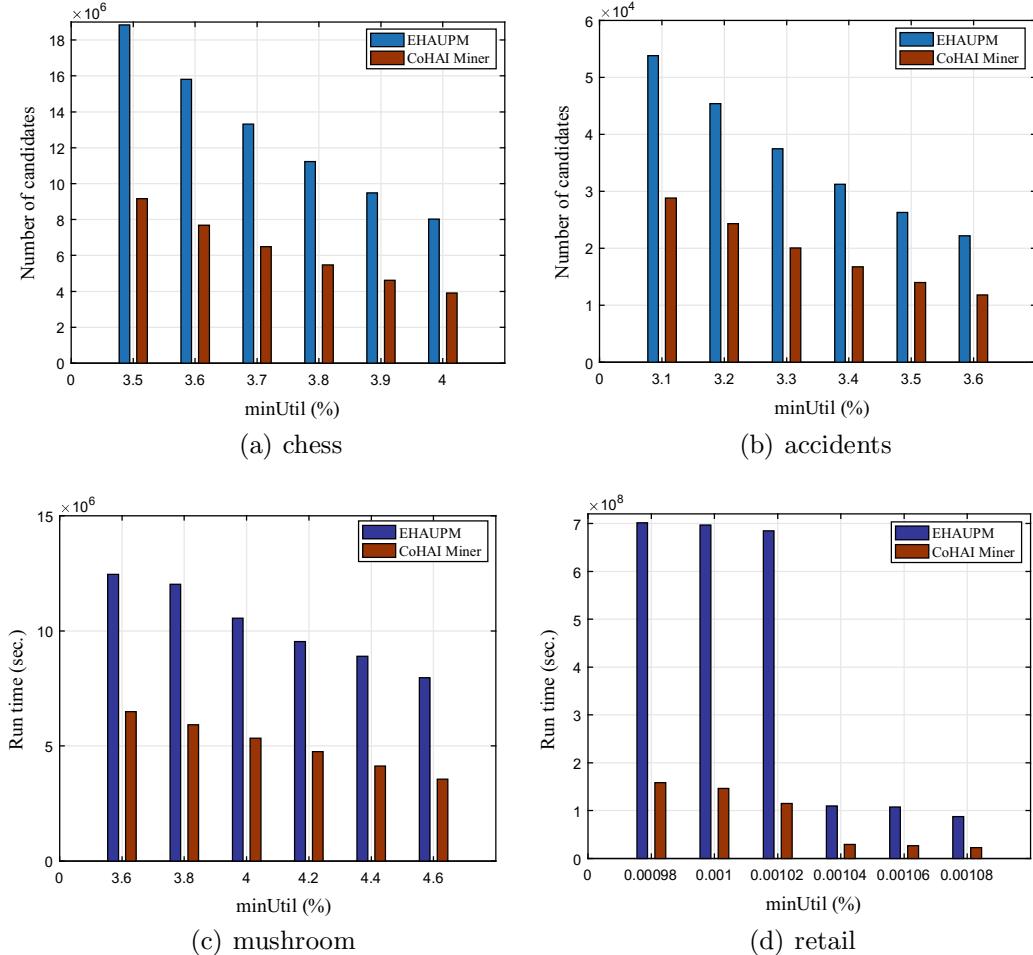


Fig. 4 Number of candidates

list of an itemset can be constructed by joining the SAU list of its subsets without any additional dataset scans. We performed comprehensive experiments to compare the performance of CoHAI miner with existing EHAUPM. It is observed from the results that CoHAI miner shows significant improvement in run time and produce more meaningful patterns.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: ACM Sigmod Record, vol. 22, pp. 207–216. ACM (1993)
2. Ahmed, C.F., Tanbeer, S.K., Jeong, B.-S., Choi, H.-J.: A framework for mining interesting high utility patterns with a strong frequency affinity. Inf. Sci. **181**(21), 4878–4894 (2011)
3. Chan, R., Yang, Q., Shen, Y.-D.: Mining high utility itemsets. In: Third IEEE International Conference on Data Mining, 2003. ICDM 2003, pp. 19–26. IEEE (2003)

4. Fournier-Viger, P., Gomariz, A., Gueniche, T., Soltani, A., Wu, C.-W., Tseng, V.S.: SPMF: a Java open-source pattern mining library. *J. Mach. Learn. Res.* **15**(1), 3389–3393 (2014)
5. Fournier-Viger, P., Wu, C.-W., Zida, S., Tseng, V.S.: FHM: faster high-utility itemset mining using estimated utility co-occurrence pruning. In: International Symposium on Methodologies for Intelligent Systems, pp. 83–92. Springer (2014)
6. Gan, W., Lin, J.C.-W., Chao, H.-C., Fujita, H., Yu, P.S.: Correlated utility-based pattern mining. *Inf. Sci.* **504**, 470–486 (2019)
7. Gan, W., Lin, J.C.-W., Fournier-Viger, P., Chao, H.-C., Fujita, H.: Extracting non-redundant correlated purchase behaviors by utility measure. *Knowl.-Based Syst.* **143**, 30–41 (2018)
8. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: a survey. *ACM Comput. Surv. (CSUR)* **38**(3), 9 (2006)
9. Hong, T.-P., Lee, C.-H., Wang, S.-L.: Effective utility mining with the measure of average utility. *Expert Syst. Appl.* **38**(7), 8259–8265 (2011)
10. Krishnamoorthy, Srikanth: Pruning strategies for mining high utility itemsets. *Expert Syst. Appl.* **42**(5), 2371–2381 (2015)
11. Krishnamoorthy, Srikanth: HMiner: efficiently mining high utility itemsets. *Expert Syst. Appl.* **90**, 168–183 (2017)
12. Li, Y.-C., Yeh, J.-S., Chang, C.-C.: Isolated items discarding strategy for discovering high utility itemsets. *Data Knowl. Eng.* **64**(1), 198–217 (2008)
13. Lin, J.C.-W., Gan, W., Fournier-Viger, P., Hong, T.-P., Chao, H.-C.: FDHUP: fast algorithm for mining discriminative high utility patterns. *Knowl. Inf. Syst.* **51**(3), 873–909 (2017)
14. Lin, J.C.-W., Li, T., Fournier-Viger, P., Hong, T.-P., Zhan, J., Voznak, M.: An efficient algorithm to mine high average-utility itemsets. *Adv. Eng. Inform.* **30**(2), 233–243 (2016)
15. Lin, J.C.-W., Ren, S., Fournier-Viger, P., Hong, T.-P.: EHAUPM: efficient high average-utility pattern mining with tighter upper bounds. *IEEE Access* **5**, 12927–12940 (2017)
16. Liu, M., Qu, J.: Mining high utility itemsets without candidate generation. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pp. 55–64. ACM (2012)
17. Liu, Y., Liao, W.-k., Choudhary, A.: A two-phase algorithm for fast discovery of high utility itemsets. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 689–695. Springer (2005)
18. Omiecinski, E.R.: Alternative interest measures for mining associations in databases. *IEEE Trans. Knowl. Data Eng.* **15**(1), 57–69 (2003)
19. Tseng, V.S., Wu, C.-W., Shie, B.-E., Yu, P.S.: Up-growth: an efficient algorithm for high utility itemset mining. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 253–262. ACM (2010)
20. Tianyi, W., Chen, Y., Han, J.: Re-examination of interestingness measures in pattern mining: a unified framework. *Data Min. Knowl. Discovery* **21**(3), 371–397 (2010)
21. Yun, U., Kim, D.: Mining of high average-utility itemsets using novel list structure and pruning strategy. *Future Gener. Comput. Syst.* **68**, 346–360 (2017)

Interactive Labelled Object Treemap: Visualization Tool for Multiple Hierarchies



Mahipal Jadeja, Hitarth Kanakia, and Rahul Muthu

Abstract Visualizing multiple hierarchies is necessary for a variety of applications., i.e. for a given set of objects, to identify object types as well as the relationship between different objects (whether objects are related to one other or not). As the simplest solution, two separate tree diagrams (object tree and taxonomy tree) can be used to display object connections and object types. We have designed an interactive visualization technique that effectively conveys both the necessary information using a single representation. ‘Labeled Object treemap’ is a very well-studied technique for visualizing multiple hierarchies in which objects are connected if and only if their corresponding set labels are disjoint and object types can be identified easily by the background structure, treemap. In this paper, we have designed an interactive version of this technique. In our interactive solution, for the given set of object types, the user can provide any size of input tree in the given textbox. A unique set label is generated for each node, and nodes are distributed and displayed (as small red circles) according to their object types in the rectangle areas of the treemap. Four checkboxes are provided to offer various features. These features are discussed in detail in the paper. Our proposed interactive visualization technique has all the good features of the existing ‘labelled object ‘Treemap’ technique including single representation, high compactness, cluster identification, etc., and solves the issues of edge crossings and continuity.

Keywords Human–computer interaction · Taxonomy · Visualizing multiple hierarchies · Treemapping · Interactive information visualization · Vertex labelling

M. Jadeja (✉)
Malaviya National Institute of Technology, Jaipur, India
e-mail: mahipaljadeja.cse@mnit.ac.in

H. Kanakia
University of Southern California, Los Angeles, CA, USA
e-mail: kanakia@usc.edu

R. Muthu
Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar, India
e-mail: rahul_muthu@daiict.ac.in

1 Introduction

In graph theory, a tree is a special type of graph which is connected and acyclic. Generally, trees are represented visually using node and link diagrams (tree diagrams). In trees, edges are present only between adjacent layers and hence, trees are most suitable to represent hierarchical data (where items are represented as being ‘below’ or ‘above’ or ‘at the same level’ to one another).

Taxonomies are used as a tool to structure information. Taxonomies classify objects into different groups. Taxonomy trees are used to represent hierarchical (classification) structure in which unique object type is represented by each leaf node and internal nodes classify object types.

For many applications, available information can be structured using two distinct trees namely (1) taxonomy tree (classification tree) and (2) object tree where object connections are shown and each object is related to the leaf nodes (object type) of a taxonomy tree. Note that actual objects are shown in object tree only and object types are shown as leaf nodes of taxonomy tree. More than one object of same object type is possible in the object tree.

The challenge is to draw the object tree without compromising details of taxonomy (position of object type in the taxonomy tree). The objective of this paper is to design an interactive visualization technique which can show details of both the trees precisely and that too, using a single representation.

The simplest solution is shown in Fig. 1 in which both the trees are drawn side by side. Here, it is not possible to identify frequencies as well as locations of different object types quickly in the underlying object tree.

2 Literature Review

In this section, existing methods of visualizing multiple hierarchies are discussed briefly.

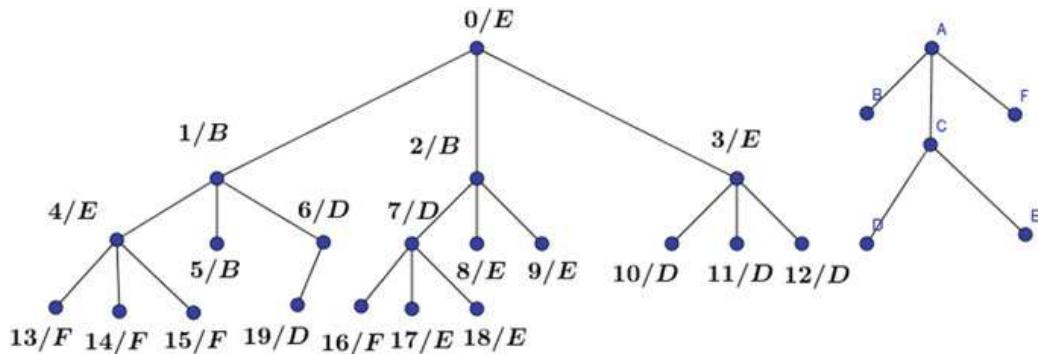


Fig. 1 Input for our proposed technique: object tree and taxonomy tree

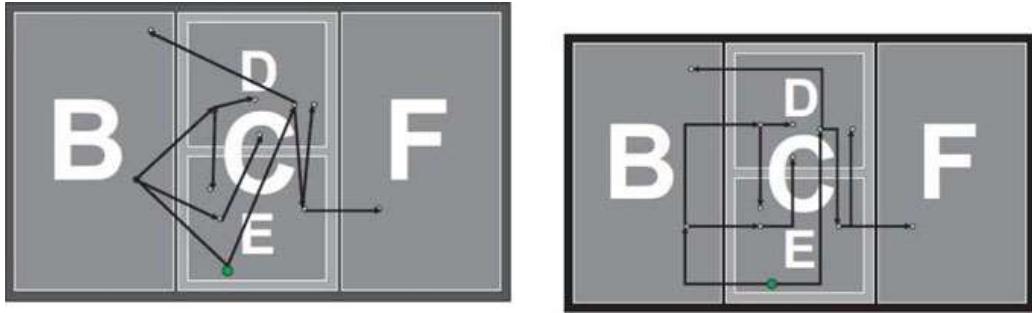


Fig. 2 Trees in a treemap visualization technique (from [3])

Most of the existing methods use treemaps along with different visualization techniques to represent multiple trees. Treemap is a visualization technique mainly suitable for large trees. It is possible to display a tree having millions of nodes in the limited available space using this technique. The core recursive idea behind this visualization technique is to allocate rectangles to parent nodes and for children, boxes(rectangles) are allocated within the corresponding parent's rectangle. Treemaps and its variants are very well studied in the literature [12, 13].

Trees in a treemap [3]: This visualization technique uses small circles to represent objects and edges (lines) to represent object connections. Taxonomy is displayed with the use of treemap (in the background). The technique has two major drawbacks when we consider big data: (a) presence of edge crossings (because of which it is difficult to identify object connections precisely) and (b) issue of continuity (very difficult to follow edges to identify object connections) (Fig. 2).

Labelled Object Treemap [9] provides a solution for the above-mentioned drawbacks by eliminating use of edges and construction of an equivalent labelled representation of the given input tree. In the case of complete binary trees and k-ary trees, a total number of elements used for labelling are asymptotically minimized. Objects are adjacent if and only if their corresponding labels are disjoint. Therefore, the method is indirectly dependent upon the user intelligence and observation skills which is the main disadvantage for this visualization technique since the visualization technique must be user-independent. For further details, please see [9].

2.1 Related Work

Visualizing multiple trees is very well studied in the literature [8]. One of the possible solutions is ‘Trees in a treemap’ technique in which object connections are shown using foreground tree, and object types (taxonomy) are shown using the background treemap. Other solutions are based on node link representations in which graphs are used to represent multiple trees. This type of representation is used in GLAD system [5]. Another technique is Cristalview [14] in which overlapping treemaps are used. The critical aspects related to success of any visualization technique mainly depend upon user interface design and target audience type [1]. Other related work includes multitrees [6] and ZTree [2].

3 Discussion on the Labelling Scheme

3.1 Proposed Labelling Scheme

A set labelling of a graph $G(V, E)$ is a function $f: V \rightarrow \mathcal{P}(\{1, 2, \dots, k\})$ where $k \in \mathbb{Z}^+$ such that

- f is one one. (Two distinct vertices must not get the same set label.)
- $\forall u, v \in V, (u, v) \in E \Leftrightarrow f(u) \cap f(v) = \emptyset$.

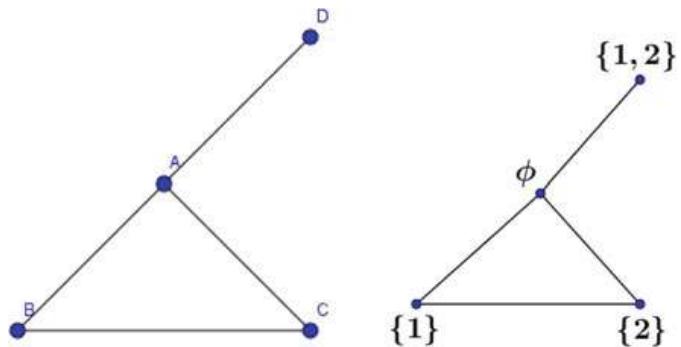
Universe size number (USN) of a graph is the least positive integer k such that a set labelling of G exists.

In other words, **Universe Size Number** (USN) of a graph is the number of elements in a smallest universal set S such that one can label the vertices of the graph with unique subsets of S such that if vertices are adjacent then their intersection of labels are disjoint and if the vertices are non-adjacent then their intersection of labels has at least one common element.

3.2 Underlying Concept for the Proposed Scheme:

The main motivation is **intersection graphs** for finite sets in which for each vertex a subset from the underlying universal set is associated and vertices are non-adjacent if and only if the corresponding subsets are disjoint. This was very well studied by Erdos et al. [4]. We use a slightly different framework in which vertex adjacency coincides with (sub)sets disjointness. A very well known example of graph family which obeys this framework is Kneser graphs. Since these two approaches are entirely different, for the given graph, they will give entirely different types of labellings (assuming that the graph is not self-complementary). In general, to find **Universe Size Number** (USN) of a graph is NP-complete problem. This is because the equivalent problem of finding intersection number is NP-complete [4, 7] (Fig. 3).

Fig. 3 Input graph H and $\text{USN}(H) = 2$



3.3 General Results on USN

Here, we present general bounds on USN.

Theorem 1 $\text{USN}(G) \geq \lceil \log_2 n \rceil$ where G has n vertices.

Proof. Case 1: $n = 2^i$ where $i \in N$. Using $\lceil \log_2 n \rceil$ elements, at most n subsets can be generated which, due to the requirement of label distinctness, can be used to label at most n vertices. This proves the claim.

Case 2: $n \neq 2^i$ where $i \in N$. Using $\lfloor \log_2 n \rfloor$ elements, at most $n - 1$ subsets can be generated which, due to the requirement of label distinctness, can be used to label at most $n - 1$ vertices. Here, total number of vertices is n . In order to assign a non-empty as well as unique label to n th vertex 1, additional element is required in the underlying universe.

Therefore, value of USN is at least $\lceil \log_2 n \rceil$ for any given graph.

Other results related to USN (and its variant UUSN) are discussed in [10, 11]. The rest of the paper is organized as follows: In Sect. 4, key results related to USN for trees are discussed. The next Sect. 5 describes all the key features of our proposed interactive visualization technique (with demo). In the same section, our proposed method is compared with the existing methods with respect to key visualization features. In the final section, we summarize our contributions along with the possible future research directions.

4 Results on USN for Trees

In this section, we list results on USN for the following classes of graphs: Binary trees, k -ary trees and trees since our proposed interactive visualization technique uses these results. All the results (excluding Algorithm 1) in this section are taken from [9].

Theorem 2 $\text{USN}(BT_n) = O(\log n)$. Where BT = complete binary tree and n denotes the total number of vertices of the BT .

Proof. For proof, please refer to [9].

Theorem 3 USN of any k -ary tree is $O(\log n)$ where number of vertices present in the T is n .

Proof. For proof, please refer to [9].

It is possible to generate a valid labelling for any tree with the use of following algorithm. **Algorithm 1: Valid-Tree-Labelling**

Step 1: For the given tree, calculate its maximum degree a .

Step 2: Consider a complete $(a - 1)$ -ary tree, with the same number of levels as the

given tree with a valid labelling (use Theorem 3 to obtain a valid labelling).

Step 3: Embed the given tree into this complete tree.

Step 4: For all the vertices which are present in the tree, copy the corresponding labels from the underlying $(a - 1)$ ary tree.

The resultant labelling is valid but total number of elements used is not asymptotically minimal.

These results are important for the development of interactive visualization of multiple hierarchies (as discussed, hierarchies are often displayed as trees).

5 Interactive Labelled Object Treemap

In order to make the existing visualization technique user-independent and interactive, we have implemented dynamic interactive version of our proposed visualization technique ‘labelled object treemap’. The same taxonomy tree (which is given in Fig. 1) is considered for our implementation. Treemap is used in the background in order to highlight taxonomy. Object classes having same parents in taxonomy tree are highlighted using same colour (Fig. 4).

5.1 Implementation Details

The interface is created using html/css with jQuery for handling click-based events. The input to the interface is a parent pointer representation of the tree. After parsing, we run the set labelling algorithm mentioned above to assign ids to all the nodes of the

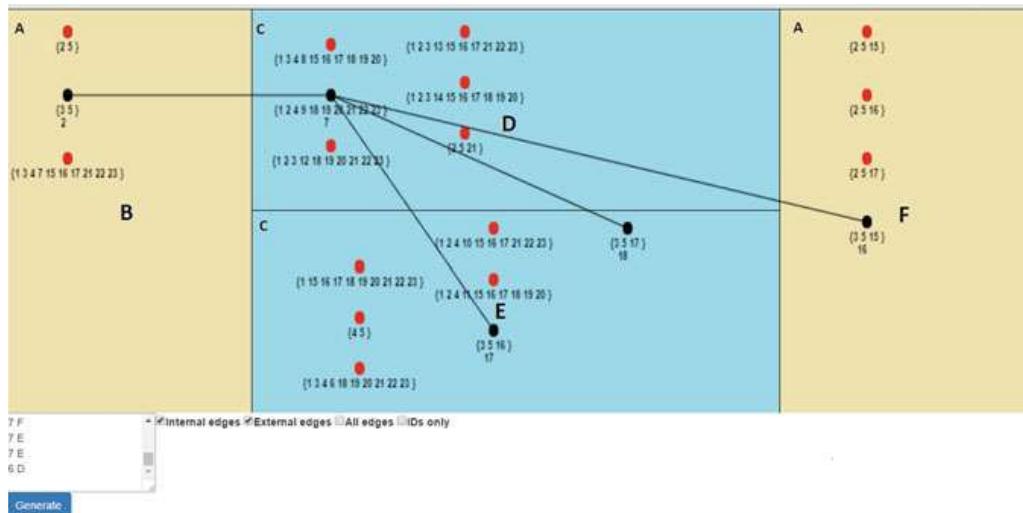


Fig. 4 Visualizing all the neighbours of object node 7

tree. We use a BFS-based approach since the labelling algorithm proceeds in a level-wise fashion. Then, depending on the mode of representation selected for the output (all edges, internal edges, etc.), we draw that tree with the nodes in the appropriate label area. The area to be occupied by a node and its set ids are computed and the nodes are then drawn so that inside every label, the minimum distance between 2 nodes is maximized to ensure good spacing.

For the given input (shown in Fig. 1), the overall layout of our proposed system is, as shown in Fig. 4.

- Click here for the demo. (Youtube link: <https://youtu.be/voUYxyI4P58>).
- Click here for the code (Figs. 5 and 6).

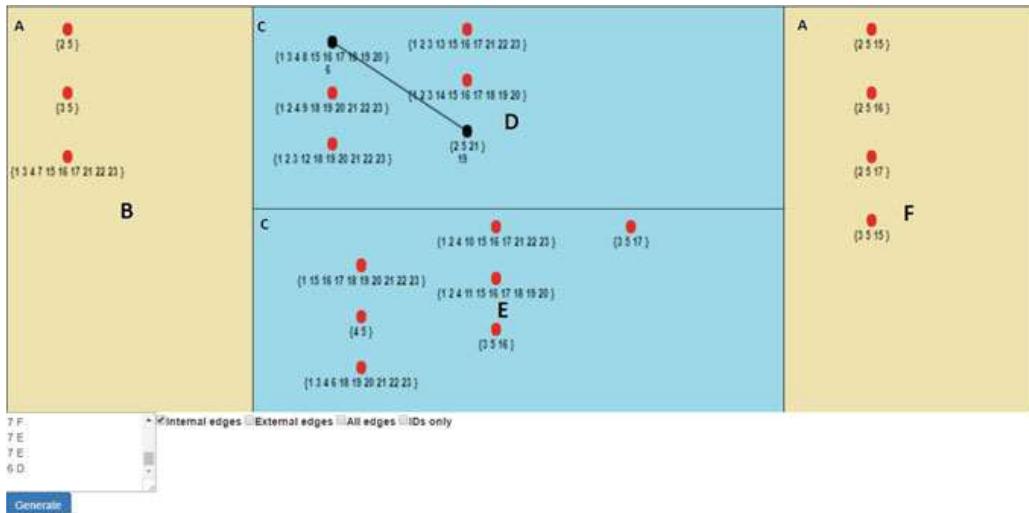


Fig. 5 Visualizing neighbours of similar object type for the object node 10

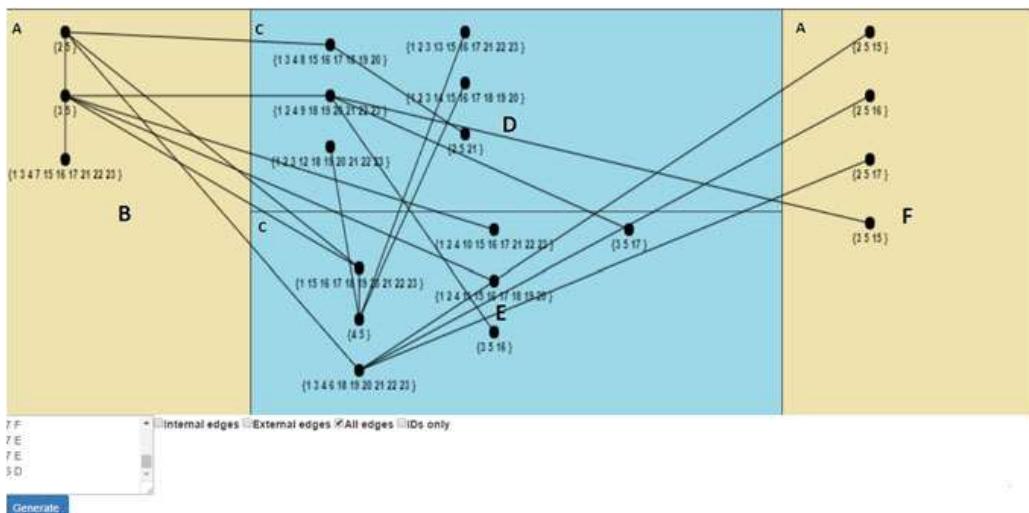


Fig. 6 All edges option

5.2 Key Features of Our Implementation

5.2.1 Visualization for the Any Given Input Tree:

The visualization technique uses label construction methods explained in the previous section. It is possible to identify object types and object relations using our proposed work. Users are supposed to give input in the textbox in (parent node id object type) format. For example, (2C) refers to the node with object type C and parent node 2. Here, the parent id is with respect to the object relationship tree. The root node does not have any parent so for the root node, we will use -1 as a parent node id. After giving valid input tree, **Generate** button should be clicked and user can see the visualization where nodes are positioned depending upon their object types along with their unique labels. Objects are connected if and only if their corresponding labels are disjoint.

5.2.2 Identification of Object Names:

By clicking on ids only button (see Fig. 7), labels will be replaced by ids (object names). Using this feature, a user can identify specific object(s) in which he/she is interested.

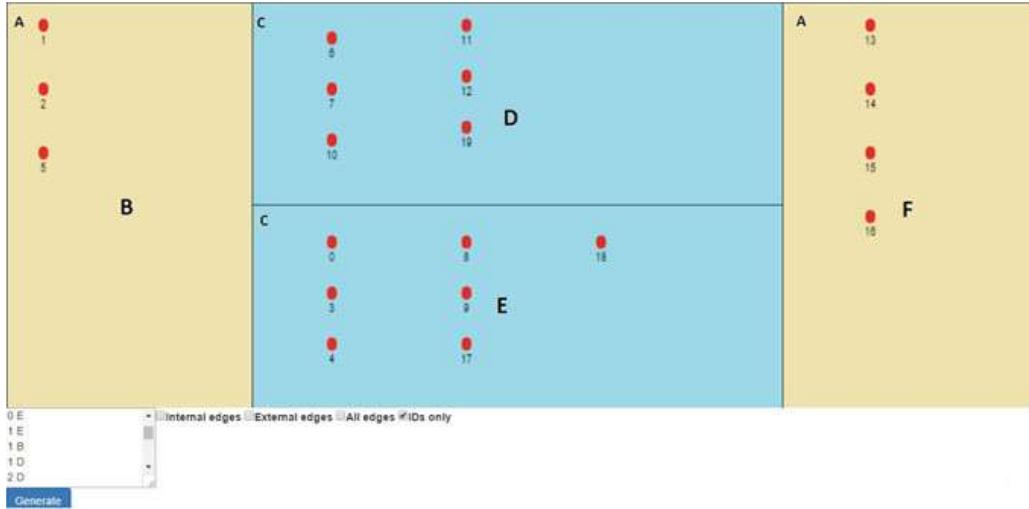


Fig. 7 Ids only option

5.2.3 Visualizing Connections

Two different checkboxes namely internal edges and external edges are provided to identify neighbours of a node. In the internal edges option (see Fig. 5): After clicking on any particular node (say v), v is highlighted in black along with its id. All the neighbours of the node v which belong to the same object type are also highlighted (with their ids) in black colour along with black edges to show connections. Colours of all other nodes (which are non-adjacent to v /nodes having different object types than v) will remain red. In the external edges option: After clicking on any particular node (say w), w is highlighted in black along with its id. All the neighbours of the node w which belong to other object type/types are also highlighted (with their ids) in black colour along with black edges to show connections. Colours of all other nodes (which are non-adjacent to w /nodes having the same object type as w) will remain red. In order to see all the neighbours of a particular node, both of the above-mentioned options must be selected before clicking on any particular object (see Fig. 4). It is also possible to visualize the whole tree by selecting ‘All edges’ option (see Fig. 6).

A brief comparison of our proposed technique with the existing techniques is shown in Table 1. We have considered parameters like single representation, crossing, continuity, compactness, etc., for comparison.

Table 1 Comparison with existing techniques

Visualization technique	Single model	Crossing	Continuity	Clusters outliers	Compact	Taxonomy
Separate tree diagrams	No	No	Straight	No	No	Tree Diagram
Linked tree diagrams	No	Yes	Straight	Difficult	No	Tree diagram
Coloured tree diagrams	No	No	Straight	Yes	Medium	Colour
Unsorted matrix	Yes	No	Not visible	No	High	Not visible
Sorted matrix	Yes	No	Not visible	Yes	High	Order
Sorted parallel coordinates	Yes	Yes	Straight	Yes	Medium	Order
Trees in TM (straight lines)	Yes	Yes	Straight	Yes	High	Treemap
Trees in TM (orthogonal lines)	Yes	Reduced	Orthogonal	Yes	High	Treemap
Interactive labelled object treemap (our proposed method)	Yes	No	Not visible	Yes	High	Treemap

6 Conclusions and Future Work

Our proposed interactive data visualization technique displays multiple hierarchies without any information loss and it can display large data sets coherently. Our proposed technique generates labelling of all objects automatically and a total number of elements used in labelling are asymptotically minimized (in the case of complete binary and k-ary trees). The interactive version offers various features using which it is easy to identify specific types of neighbours of a node just by clicking on it. Taxonomy is visible using treemap. The proposed interactive version has all the characteristics (single representation, compactness, clusters identification, etc.) to be a potential good visualization technique for visualizing multiple hierarchies. In future, one can think of an arrangement of nodes that is more compact so that a larger number of nodes can be clearly shown(along with the labels) in the same available space. Potentially, arranging the nodes in a zigzag pattern is one such solution.

References

1. García, P.A., Martín-Moncunill, D., Sánchez-Alonso, S., García, A.F.: A usability study of taxonomy visualisation user interfaces in digital repositories. *Online Inf. Rev.* **38**(2), 284–304 (2014)
2. Bartram, L., Uhl, A., Calvert, T.: Navigating complex information with the ztree. In: *Graphics Interface*, pp. 11–18. Citeseer (2000)
3. Burch, M., Diehl, S.: Trees in a treemap: visualizing multiple hierarchies. In: *Electronic Imaging 2006*, pp. 60600P–60600P. International Society for Optics and Photonics (2006)
4. Erdos, P., Goodman, A.W., Pósa, L.: The representation of a graph by set intersections. *Canad. J. Math.* **18**(106–112), 86 (1966)
5. Florentz, B., Muecke, T.: Unification and evaluation of graph drawing algorithms for different application domains. In: *Tenth International Conference on Information Visualization, 2006. IV 2006*, pp. 475–482. IEEE (2006)
6. Furnas, G.W., Zacks, J.: Multitrees: enriching and reusing hierarchical structure. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 330–336. ACM (1994)
7. Gary, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of np-Completeness (1979)
8. Graham, M., Kennedy, Jessie: A survey of multiple tree visualisation. *Inf. Visual.* **9**(4), 235–252 (2010)
9. Jadeja, M., Muthu, R.: Labeled object treemap: a new graph-labeling based technique for visualizing multiple hierarchies. *Ann. Pure Appl. Math.* **13**, 49–62 (2017)
10. Jadeja, M., Muthu, Rahul: Uniform set labeling vertices to ensure adjacency coincides with disjointness. *J. Math. Comput. Sci.* **7**(3), 537–553 (2017)
11. Jadeja, M., Muthu, R., Sunitha, V.: Set labelling vertices to ensure adjacency coincides with disjointness. *Electron. Notes Discrete Math.* **63**, 237–244 (2017)
12. Jadeja, M., Shah, K.: Tree-map: A visualization tool for large data. In: *GSB@ SIGIR*, pp. 9–13 (2015)
13. Kong, N., Heer, J., Agrawala, Maneesh: Perceptual guidelines for creating rectangular treemaps. *IEEE Trans. Visual. Comput. Graph.* **16**(6), 990–998 (2010)
14. Mank, F.W.M.: Cristalview-the Visualization of a Cristal. Master’s Thesis, Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, Netherlands, p. 115, (2005)

Medical Transcriptions and UMLS-Based Disease Inference and Risk Assessment Using Machine Learning



Thamizharuvi Arikrishnan and S. Swamynathan

Abstract An extensive set of applications and approaches for automatic disease inference with risk assessment system in medical fields objected to improve the efficacy and efficiency of patient care. Recent research studies are getting done in the area to predict patients' future health conditions based on the medical records, particularly full clinical texts, medical transcriptions, clinical measurements, or medical codes. Healthcare data are highly complex, multi-dimensional, and heterogeneous in nature which are the key challenges. In this paper, a system has been proposed for disease inference by extracting symptoms and mapping the metadata using Unified Medical Language System (UMLS) to have the disease code. The symptoms and metadata are extracted from medical transcripts using natural language processing (NLP) and the extracted information has been categorized into chief complaints, present illness, and past history. These extracted symptoms are mapped using UMLS for disease code inference. Based on the disease code, the risk classifier classifies the level of risk. The proposed system based on deep learning and UMLS for disease inference resulted with significant improvement in accuracy.

Keywords Healthcare analytics · Deep learning · UMLS · Natural language processing · Disease inference · Risk assessment

1 Introduction

Electronic health records (EHRs) have been given primary importance due to its rapid growth and voluminous amount of key information on patients' health conditions named medical history, medical test reports, social determinants, and diagnostic

T. Arikrishnan (✉)
IBM India / IST, CEG Anna University, Chennai, TN, India
e-mail: Thamizharuvi.A@in.ibm.com

S. Swamynathan
IST, CEG Anna University, Chennai, TN, India
e-mail: swamyns@annauniv.edu

results. Heterogenous nature of data in EHR provides its own set of challenges and a way to derive insight of medical knowledge by which define a system and strategy to improve the quality and efficiency of patient care. Medical transcriptions are considered as the primary input to the system, processing the transcripts for risk assessment and disease inference by extracting the symptoms [1] using natural language processing [2, 3] and machine learning techniques and deep learning techniques [4].

The Unified Medical Language System (UMLS) contains over one million biomedical concepts from over 100 source vocabularies. To identify symptom entities, the system workflow relies on existing NLP techniques [2] and a suite of technology stack with libraries for medical text analytics and statistical natural language processing and MetaMap which is used for recognizing Unified Medical Language System (UMLS) [5, 6, 9] in transcripts text. The association between symptoms provides key input for disease inference. With a massive influx of multimodality data, the role of data analytics in health informatics has grown rapidly in the last decade. Deep learning is emerging in recent years as a powerful tool for machine learning, promising to reshape the future of artificial intelligence [7, 8]. The symptom extraction and representation are the key process in disease inference system.

In this study, the proposed workflow has been designed to have disease inference integrated with the risk assessment flow, combining the bidirectional LSTM [6] model in association with TF-IDF approach for inferring disease code. The natural language processing approach with UMLS MetaMap was used to extract symptom concepts from medical transcripts dataset from [Kaggle.com](#) [7, 8, 9]. The solution model leverages the key features of the deep learning approaches and has been presented with evaluated measures of bidirectional LSTM with TF-IDF. The disease inference module was handled as core multi-class classification problem; in this study, we conducted experiments on the most common diseases. Our proposed models, bidirectional LSTM with TF-IDF, showed significant improvements in results compared with SVM [10] and sequential NN.

2 Literature Survey

Various research work and studies have been done on disease inference and risk assessment models in healthcare domain. Recent work with proposed approaches is based on machine learning and deep learning methodologies. Many works have been done in handling the challenges of electronic health record. The vast variety of structured and unstructured data such as handwritten doctor notes, medical records, medical diagnostic images [magnetic resonance imaging (MRI) [6, 10], computed tomography (CT)] [6, 10], and radiographic films and related work have been done by Feldman et al. Deep learning [8, 2] has been a noticeable topic in machine learning in this era when Bengio reviewed the motivation of deep learning and summarized the famous algorithms for deep learning. The applications in the Bengio and team cover image retrieval, music informatics, web search, and natural language processing. Regarding the algorithms, Hinton and team reported that deep supervised nets can

be trained by adding an unsupervised pre-training by RBM for parameter initialization. Lee proposed to scale unsupervised learning of hierarchical generative models for large datasets with high dimensions. An augmented AI system-based proposed methods were targeted to do the prediction of a single disease. Recently, more research work has been carried out to have inference of different diseases. Choi and team used medical codes in EHR to diagnose the disease and medical codes which are fed as primary inputs to the neural networks models to predict diseases. The proposed model have resulted in improved benchmark methods on the diagnosis of most frequent diseases.

3 Deep Learning and UMLS-Based Risk Assessment and Disease Inference System Architecture

The workflow describing the overview of deep learning and UMLS-based risk assessment and the system for disease inference in this study is shown in Fig. 1. In this study, the primary input which has been ingested into the system is the medical transcriptions. The medical reports would get treated as the next level of input. The medical transcriptions were processed using natural language processing module, applied with text categorization and segmentation techniques to extract the primary components as parts of consultations by which the system is enabled to segment the transcriptions to three segments as the chief complaint and present illness and past history. In the metadata extractor module, we process the medical transcriptions and do ingest the data into the three extractor modules and get them processed and stored in respective databases for chief complaints, present illness, and past history. Each

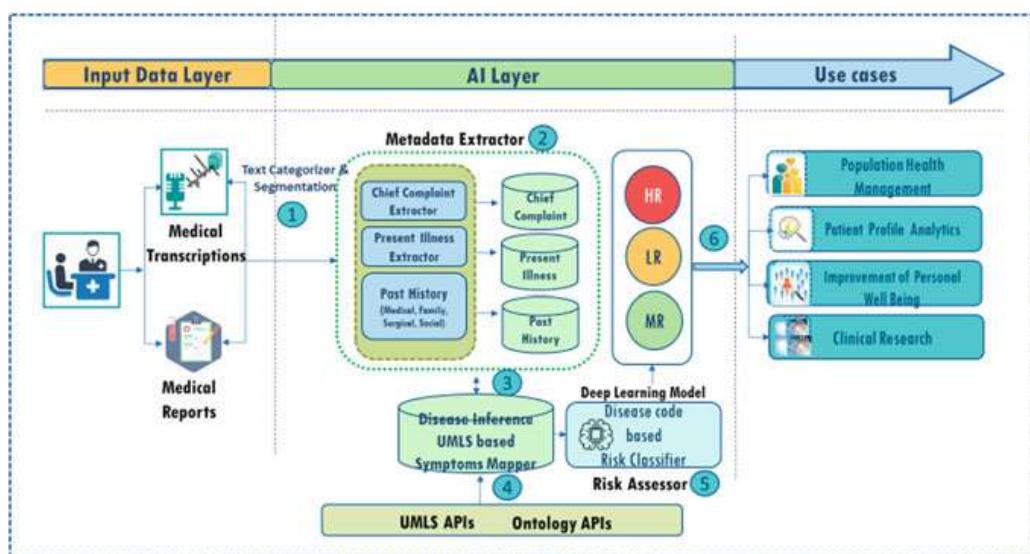


Fig. 1 Deep learning and UMLS-based risk assessment and disease inference system architecture

of the segments is distinctly stored in their respective DBs. After the preprocessing to filter out the non-symptom entities from the medical transcripts according to the structural characteristics of the dataset, the proposed system uses the existing natural language processing and UMLS MetaMap [2, 9, 10] in the workflow to extract the symptoms and the associated keywords from full clinical texts like medical transcriptions. Deep learning approaches were leveraged to do the vector representation of symptoms, and the processed clinical text was used to train word model to obtain the word vector which is utilized to generate the vector form of symptoms. The third part is a multi-label classifier module. In this part, system uses the keyword sequences of symptoms to train the multi-class classification model to identify the risk level. Finally, the system takes the outputs of deep learning model with varied symptom features extraction and representation approaches for predicting candidate diseases.

3.1 Metadata Extraction

The electronic medical transcriptions are the primary source of information of patient records in an unstructured text format in natural language. However, it is not very easy to extract symptoms and associated keywords [6, 12] from corpus due to their own set of challenges because of heterogenous nature and ambiguities. For example, ‘breath with difficulty’ and ‘difficult breathing’ referring the same about dyspnea. But the irregularity in writing the symptoms in medical transcripts brings in more difficulties to the recognition and extraction of symptoms from medical texts. It is very important to handle the heterogenous nature of clinical text and the extraction of the symptoms and the diversity of the statements when identifying symptoms. The proposed model detect and select the symptom concepts and its related keywords. Finally, filter out the concepts found in negated contexts.

3.2 Metadata Representation

The symptom representation approach using TF-IDF [4] gets the importance in the representation methodology compared to statistical methods. The symptoms and the associated keywords and the representation of symptoms associating the extracted keywords have a potential impact on the inference of disease; we use Word2Vec to get the representation of symptoms which has been used to validate the similarities between symptoms and the related entities. The statistical TF-IDF [3, 4] approach has been proposed to represent text documents as vectors of symptoms and its identifiers and this statistical approach has been used to represent a disease as vectors of different features. Risk assessment and disease inference disease is usually associated with multiple symptoms, the relationship and the association between symptoms can provide valid input for disease inference.

3.3 Disease Inference UMLS-Based Symptoms Mapper

The Symptoms Mapper module does the primary job of disease inference using the UML system. The UMLS [6, 9] contains over one million biomedical concepts from over 100 source vocabularies. The proposed workflow in this study mainly leverages the ‘The Metathesaurus’ component. The Metathesaurus has been built with over 100 vocabularies, code sets, and thesauri, or ‘source vocabularies’ are brought together to create the Metathesaurus. Terms from each source vocabulary are organized by meaning and assigned a concept unique identifier (CUI) [11]. The source vocabularies termed under many different types of biomedical vocabularies are included in the Metathesaurus. They have been categorized into different ways and some vocabularies fall into more than one category. The system in this study highly uses the diseases component of the source vocabularies and returns the inferred disease code as a resultant having ICD-10 [7] as the repository. The inferred disease code is fed into the machine learning classifier model.

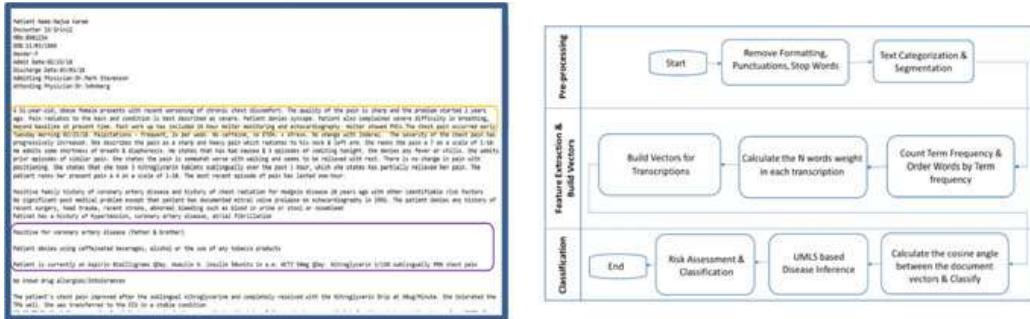
3.4 Risk Assessment and Classifier

Risk assessment and the classification of risk group of patient record, the medical transcripts of patients were processed through the natural language processing module to get them segmented as chief complaint, present illness history, and past history. The extracted text segments were injected to the symptom representation and mapper modules to retrieve the disease codes as primary resultants. The disease codes from the MetaMap [7] and Metathesaurus [11, 7] getting mined and inferred with the risk level. The risk level of the inferred disease code is divided into three subgroups high risk, medium risk, and low risk groups, who would target for respective medications. The patient records with the inferred disease codes are fed into the classifier model and the model has been trained with the features to define high risk, medium risk, and low risk groups. The high-risk group identified would be the immediate targeted population group for health management.

4 Implementation

In the implementation of the system proposed, the core input data considered in the system flow are the medical transcriptions which were available as open dataset in a technical forum Kaggle.com. The symptoms and the keywords entities extraction and representation technique are the primary step and play the critical role in disease inference module by determining the symptom keywords and its associated elements and features that the metadata extractor module can extract. With the extracted symptoms and the keywords, the relationship between symptoms and diseases defines the

mechanism to determine the candidate disease and diagnosis, when a patient visits consultant with certain symptoms. We use the deep learning algorithm bi-directional LSTM combining the mechanism TF_IDF for symptom representation and its association through statistical methods. Considering the association between symptoms which derives key input for the disease inference module, and also, we use Word2Vec to do the representation of symptoms and related keyword entities which would identify the inherent similarities between symptoms. The symptoms and the associated elements provide valued input to the disease inference system. For representing the similarities and differences between symptoms, we use Word2Vec as another symptom vector representation scheme [8]. The Word2Vec model is trained with medical transcriptions and discharge summary in which stop words are eliminated. The natural language processing flow enabled with Keras preprocessing layers, which handles all the required text preprocessing, stop words removal, tokenization, word constants, document constants. The classifier model has been trained with disease codes to identify the risk level of the patient records and they are grouped as high, medium, and low risk groups.



5 Results and Discussion

5.1 Data Description

In this work, we used the medical transcriptions dataset available in an open forum named Kaggle.com [8]. In this proposed system evaluation, we used only the medical transcriptions data that comprises of medical training of over 1600 patients' records. This work is primarily focusing on the medical transcriptions which comprise of not only the patient's conversation transcriptions and also other medical records which will get included in the study in future work of the proposed system. With the disease inference system, the patient records will get with a disease code which is a five-character International Classification of Diseases (ICD-10) [11, 7] codes. In that the first three characters of the disease code ICD-10 [7] specify the category of the disease and then the next two characters provide the subdivision for the disease.

In our study, we primarily look for only the diseases category. The diagnosis of the diseases for these varied diseases makes multi-label classification task difficult in disease inference.

5.2 Evaluation Metrics

The critical part evaluation methodology is in selecting the appropriate evaluation strategies. The disease inference system followed by risk assessment handled as a multi-label classification task and further handles the problem of sample imbalances and sparse labels, though there are varied evaluation strategies for multi-label classification problem. We used four micro-averaging measurements: precision (MiP), recall (MiR), F1 score (MiF1). F1 score is calculated as a weighted average of the precision and recall.

5.3 Experiments and Results

In the overall system, we defined the methodology and approach to define a strategy to do the inference of disease based on symptoms as a multi-label classification problem. As we describe above, there are many models based on medical transcriptions and associated them with ICD [5, 11] codes. In our method, symptoms and the associated elements play a vital role in disease inference. To validate the scenario, we proposed a model based on full text symptoms and its associated elements and compared their performances. In this study, we performed disease inference task for the most common diseases. In this evaluation exercise, the dataset has been randomly divided into training set (80%), validation set (10%), and test set (10%). The data distribution during evaluation against each model was captured and the results are as below. The SVM [3] model data was distributed with TP-589/1632, FP-197/1632, TN-486/1632, FN-360/1632. The sequential NN has been distributed with, TP-681/1632, 232/1632, 413/1632, 289/1632. Bidirectional LSTM model was performing good with these data points and has been distributed as, TP-756/1632, FP-246/1632, 394/1632, 243/1632. Table 1 captures the performance of different

Table 1 Disease inference model performance without UMLS

Deep learning model	Precision	Recall	AUC	F1 score (%)
BiLSTM	0.519	0.638	0.859	57.20
SVM	0.488	0.567	0.814	52.50
SeqNN	0.353	0.538	0.743	42.60

Table 2 Disease inference model performance using UMLS

Deep learning model	Precision	Recall	AUC	F1 score (%)
BiLSTM	0.746	0.675	0.924	71.60
SVM	0.619	0.587	0.878	63.10
SeqNN	0.573	0.512	0.783	56.70

machine learning models for disease inference without UMLS and measures as accuracy in terms of F1 score. And Table 2 compares the accuracy of the performance of the models with UMLS.

6 Conclusions and Future Enhancements

In this study, the proposed workflow has been designed to have disease inference integrated with the risk assessment flow, combining the bidirectional LSTM model with TF-IDF for inferring the disease code. The proposed solution model leverages the advantages of deep learning models. The patient records with the inferred disease codes are fed into the classifier model and the model has been trained with the features to define the risk level. In future, added to the medical transcriptions processing flow, enabling the system in processing varied other medical report as another level of core input to the system to extract features associated to the clinical reports that express the combination of symptoms which must be analyzed in detail to provide more accurate information for disease inference. To improve the accuracy of disease inference, the future work has been planned to consider introducing the medical reports and test results and biological data. With the improved and enhanced disease inference flow with the risk assessment is expected to get increased accuracy.

Acknowledgements The infrastructure facility provided by Department of Science and Technology, Government of India (DST-FIST) was used while implementing the solution model. The facility and support were provided by IBM for the execution of this study.

References

1. Guo, D., Li, M., Yu, Y., Li, Y., Duan, G., Wu, F.X., Wang, J.: Disease inference with symptom extraction and bidirectional recurrent neural network. In: IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE, pp. 864–868 (2018)
2. Fakoor, R., Ladhak, F., Nazi, A., Huber, M.: Using deep learning to enhance cancer diagnosis and classification. In: Proceedings of the International Conference on Machine Learning, Vol. 28. ACM, New York, USA (2013)
3. Shickel, B., Tighe, P.J., Bihorac, A., Rashidi, P.: Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. IEEE J. Biomed. Health. Inf. **22**(5), 1589–1604 (2014)

4. Ma, F., Chitta, R., Zhou, J., You, Q., Sun, T., Gao, J.: Dipole: diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1903–1911 (2017)
5. Zeng, M., Li, M., Fei, Z., Wu, F., Li, Y., Pan, Y., Wang, J.: A deep learning framework for identifying essential proteins by integrating multiple types of biological information. In: IEEE/ACM Transactions on Computational Biology and Bioinformatics (2019)
6. Yu, Y., Li, M., Liu, L., Li, Y., Wang, J.: Clinical big data and deep learning: applications, challenges, and future outlooks. *Big Data Min. Anal.* **2**(4), 288–305 (2019)
7. Fang, R., Pouyanfar, S., Yang, Y., Chen, S.C., Iyengar, S.S.: Computational health informatics in the big data age: a survey. *ACM Comput. Surv. (CSUR)* **49**(1), 1–36 (2016)
8. <https://www.kaggle.com/datasets>
9. UMLS system: <https://www.nlm.nih.gov/research/umls/index.html>
10. Choi, E., Bahadori, M.T., Schuetz, A., Stewart, W.F., Sun, J.: Doctor ai: predicting clinical events via recurrent neural networks. In: Machine Learning for Healthcare Conference, pp. 301–318 (2016, December)
11. Li, M., Fei, Z., Zeng, M., Wu, F.X., Li, Y., Pan, Y., Wang, J.: Automated ICD-9 coding via a deep learning approach. *IEEE/ACM Trans. Comput. Biol. Bioinf.* **16**(4), 1193–1202 (2018)
12. Martin, L., Battistelli, D., Charnois, T.: Symptom recognition issue. In: 13th Workshop on Biomedical Natural Language Processing (BioNLP), pp. 107–111 (2014)

Parallel Message Encryption Through Playfair Cipher Using CUDA



Saloni Goyal , Balie Shalomi Pacholi , B. Ashwath Rao ,
Shwetha Rai , and N. Gopalakrishna Kini

Abstract The well-known multi-letter encryption cipher is Playfair cipher although a wide variety of techniques have been employed for encryption and decryption. The Playfair cipher shows a great advancement over other encryption methods. This cipher technique is used for encrypting and decrypting the pair of characters together. This paper proposes an idea to solve the traditional Playfair cipher through parallel algorithm in order to encrypt the given message. A key is used to encrypt the message. The idea is to reduce the time complexity of sequential Playfair cipher by parallelizing it and using the power of GPUs to the maximum. Instead of making one process do the encryption, a number of threads are used to encrypt letters simultaneously. This is achieved using CUDA.

Keywords CUDA · Encryption · Parallel computation · Playfair cipher · Threads

S. Goyal · B. S. Pacholi · B. A. Rao () · S. Rai · N. G. Kini

Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka 576104, India
e-mail: ashwath.rao.b@gmail.com

S. Goyal
e-mail: saloni9628@gmail.com

B. S. Pacholi
e-mail: balipacholi1996@gmail.com

S. Rai
e-mail: shwetharai.cse@gmail.com

N. G. Kini
e-mail: ng.kini@manipal.edu

1 Introduction

Nvidia developed CUDA which is a parallel computing platform. CUDA [1] exploits the power of graphical processing units (GPUs) for parallel part of computation and uses CPU for sequential part of the program. Initially, GPUs were used for gaming purpose and now its capability is expanded to math, science, and all the other possible fields. Use of GPUs increases the performance to another level when the amount of computation to be done is very high.

This paper discusses parallel implementation of encryption method which is Playfair cipher. This encryption method is used to encrypt digraphs which are pairs of letters unlike Caesar cipher which encrypts every single letter. The starting step in encryption is always having the key, with which the message is going to be encrypted. This algorithm starts with taking a key as the input from the user. CUDA implemented approach uses a fixed key matrix size of 25 characters where the letter ‘j’ (any alphabet can be chosen according to the algorithm) is excluded and then key matrix is generated with unique alphabets taking the key as the initial characters of the matrix. Excluding one alphabet is required because the size of key matrix is fixed to 25 alphabets in basic Playfair cipher. To form key matrix, alphabetical order is followed excluding the key characters. Since this is a basic step with not much computation, CPU is used [2]. When it comes to encrypting the message, GPU is used. Key matrix size can be extended if the user wants to encrypt special characters also.

To understand key matrix generation better, assume the ‘key’ that user enters is ‘monarchy’ then the formed key matrix is given in Fig. 1.

Suppose the key had duplicate letters then only the unique alphabets will be considered while key matrix creation. Let us encrypt the message ‘conferences’. The encrypted message is shown in Fig. 2.

Key matrix is used to apply Playfair encryption rules [3]. These encryption rules are:

1. If the two letters that are to be encrypted together are present in the same column, then each letter is replaced by the letter specified just below it. When a letter is at bottom, then encrypt it with the top most letter in that column.

Fig. 1 Key matrix using monarchy as key

m	o	n	a	r
c	h	y	b	d
e	f	g	i	k
l	p	q	s	t
u	v	w	x	z

Fig. 2 Output cipher text

hmogkm gme1tx

2. If both the letters are present in the same row, then each letter is replaced by the letter specified just right to it. And when a letter is at leftmost column then replace it with rightmost letter in the row.
3. If both the rules specified above turn out to be false, form a rectangle using those two letters and replace each letter by the letter in the opposite corner of the rectangle.

The next step is taking a message to encrypt. Since the encryption is on two alphabets together, it is important that they are unique, and that the length of the message is divisible by 2 so that every thread created has two alphabets to encrypt. To achieve this, a filler letter (generally ‘x’) is padded between two identical characters and if the length is not even, ‘z’ is padded in the end of the message. Since ‘j’ is omitted from the key matrix, it is not possible to encrypt it using Playfair cipher encryption [4] rules mentioned above. Therefore, it is replaced with the letter ‘i’ and then encrypted.

Further, this paper consists of following sections: Section 2 broadly discusses the approach proposed in the paper. Section 3 gives the detailed analysis to support the proposed method used. Section 4 talks about the amortized analysis which gives the comparison between the traditional and parallel implementation of Playfair cipher following the conclusion and future scope in Sect. 5.

2 Methodology

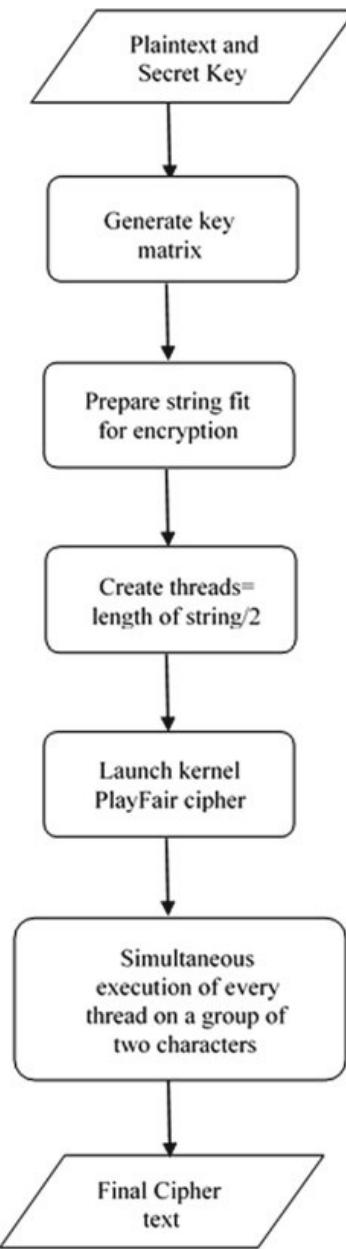
The GPU accelerates applications running on the CPU by offloading some of the compute-intensive and time-consuming portions of the code. The rest of the application still runs on the CPU. From a user’s perspective, the application runs faster because it is using the massively parallel processing power of the GPU to boost performance. This is known as ‘heterogeneous’ or ‘hybrid’ computing [5] (Fig. 3).

A CPU consists of four to eight CPU cores, while the GPU consists of hundreds of smaller cores. Together, they operate to crunch through the data in the application. This massively parallel architecture is what gives the GPU its high compute performance [6]. There are several GPU-accelerated applications that provide an easy way to access high-performance computing (HPC).

User enters the message he wants to encrypt and his secret message to create a key matrix on which rules can be applied for encryption.

If the length of the string is odd, then a character must be appended in the end as an additional bit to make the length even. This is done in prepare string step.

Now threads have been created in one direction that is in x direction where the parallelism is observed. Every thread is encrypting two letters simultaneously. Suppose there is a string of 1023 letters then prepare string will be called and it will append an additional character at the end of the string and that character ‘z’ is our program. Further length will be incremented by 1. Then threads will be created such that every thread works on 2 characters. It has one limitation that maximum number

Fig. 3 Parallel algorithm

of threads that can be created are fixed based on instruction size of the computer, although that is a huge string length.

Once every thread has done its execution, we obtain our desired result that is the cipher text. Cipher text is data that has been encrypted. Cipher text cannot be understood until it has been converted into plain text (decrypted) with a key.

Even if your computer does not have any GPU then also it can compute using CPU as mentioned in Nvidia. The results mentioned in the paper are based on the experiment conducted on hardware having Nvidia GPU which can have maximum of two 2-D grids with 1024×1024 threads possible.

3 Analysis

Time variation is observed while experimenting parallel computation for a given plain text while keeping the secret key same for all strings as shown in Table 1 (Fig. 4).

Let us consider the methodology used for sequential process that is executed with a single process as shown in Fig. 5. Since only one process is executing, it needs to run for (string length)/2 times therefore increasing the time constantly.

Table 2 gives the analysis of time variations for sequential code.

Now we calculate speedup in Eq. (1) for different values of string length in Table 3.

$$\text{Speedup} = \frac{\text{Time taken by sequential algorithm}}{\text{Time taken by parallel algorithm}} \quad (1)$$

There is a constant increase in speedup. This proves that CUDA works better when there is high computation. Even if we increase the length of string, time taken by CUDA program is almost same for every string length, but it is not the case in sequential algorithm. It keeps on increasing.

With the help of line graph in Fig. 4, we conclude that time complexity taken in parallel implementation of Playfair cipher using CUDA takes constant time that is $O(1)$ as mentioned in Table 4.

Table 1 Time taken for variable string lengths

String length	Time (ms) using CUDA
100	0.0296
300	0.0306
700	0.0367
1600	0.0318

Fig. 4 Time variation analysis

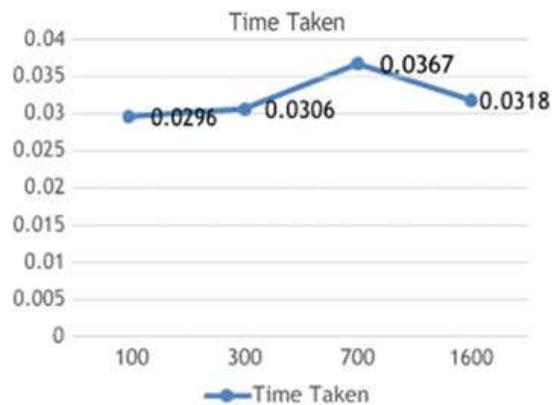
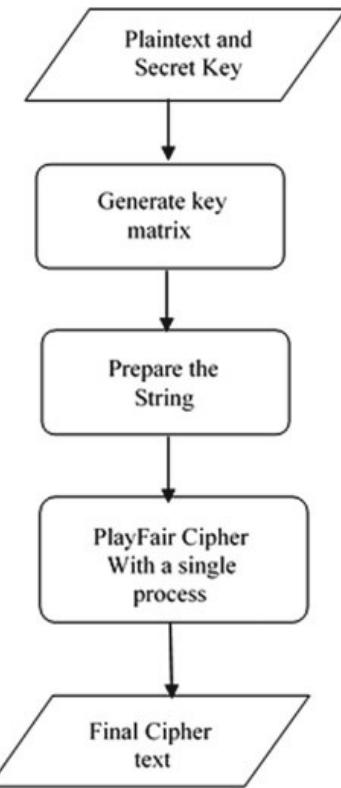


Fig. 5 Sequential algorithm**Table 2** Time taken for variable string lengths

String length	Time (ms) using sequential Playfair
100	0.0301
300	0.1384
700	0.2004
1600	0.7201

Table 3 Speedup corresponding to variable string lengths

String length	Speedup
100	1.016
300	4.522
700	5.460
1600	22.64

Table 4 Time complexity taken by each method

	Time complexity
Sequential Playfair cipher	$O(n)$
Parallel Playfair cipher	$O(1)$

4 Amortized Analysis

This analysis is used when most of the operations that are executing are fast but there are some occasional operations which are slow.

Once the user inputs the secret key and plain text, the first operation is key matrix generation and then the kernel is launched. Let us consider the amortized cost for each operation one by one.

Key matrix generation—this parallel implementation has a fixed size of key matrix which is 5×5 matrix of English alphabets excluding ‘j’.

Before matrix generation, we have

$$\vartheta(x) = E(\text{Key Length}) \quad (2)$$

This is because user enters key and then every character is checked for the letter ‘j’ to replace it with ‘i’, and therefore, the potential function is represented in (2).

After matrix generation,

$$\vartheta(x) = E(\text{Key Length}) + 25 \quad (3)$$

25 is the fixed because of the fixed size of matrix. In most of the cases, key length is less than 25, but occasionally, it can exceed that. Therefore, the amortized cost of matrix generation here is (3) –(2).

$$\vartheta'(x) = 25$$

$$\mathbb{C}' = \mathbb{C} + \vartheta'(x) \quad (4)$$

In (4) \mathbb{C}' is the amortized cost and \mathbb{C} is the actual cost which is 1 here $\mathbb{C}' \in O(1)$.

The next in the sequential code is launching kernel. The next operation is searching the position of two consecutive characters in the matrix. Suppose the character that must be searched is the last character in the matrix, in that case all the letters in the matrix must be compared with the letter to be searched. This is not the case every time. Once the characters are searched, they are encrypted using the encryption rules specified as in Playfair cipher algorithm. In (4) with this operation, \mathbb{C} is the actual cost which is 2 here because each thread is having 2 characters to encrypt.

$$\vartheta(x) = \text{KeyMatrix} \quad (5)$$

$$\vartheta(x) = \text{KeyMatrix} + \text{Search}(25) + \text{Encrypt}(2) \quad (6)$$

(5) gives previous potential function and (6) is the potential function after encryption. Potential difference is calculated by (6)–(5).

$$\vartheta'(x) = \text{Search}(25) + \text{Encrypt}(2) \quad (7)$$

From (4) and (7), we have amortized cost of encryption in (8).

$$\mathbb{C}' = 2 + \text{Search}(25) + \text{Encrypt}(2) \quad (8)$$

$$\mathbb{C}' \in O(1).$$

5 Conclusion and Future Scope

A better speedup is achieved using parallel implementation.

This paper discusses encryption of character by character using Playfair, and to improve the strength of encryption, it can be further extended to byte by byte encryption of each character. This will lead to increase in security.

Presently, an idea is proposed to develop alphabet encryption. This can be further extended [7, 8] to encrypt special characters using bigger key matrix which incorporates all the characters including special characters.

References

1. Stojanovic, N., Stojanovic, D.: High performance processing and analysis of geospatial data using CUDA on GPU. *Adv. Electr. Comput. Eng.* **14**(4), 109–114 (2014)
2. Liao, X., Yu, L., Yao, H.: A novel GPU resources management and scheduling system based on virtual machines. *Int. J. High Perform. Comput. Netw.* **9**(5/6), 423 (2016)
3. GeeksforGeeks: A computer science portal for geeks. GeeksforGeeks (2019). Online Available: <https://www.geeksforgeeks.org/>. Accessed 22 Oct 2019
4. Shakti Srivastava, S., Gupta, N.: A novel approach to security using extended playfair cipher. *Int. J. Comput. Appl.* **20**(6), 39–43 (2011). Available: <https://doi.org/10.5120/2435-3276>
5. Power, J., Hestness, J., Orr, M., Hill, M., Wood, D.: gem5-gpu: a heterogeneous CPU-GPU simulator. *IEEE Comput. Archit. Lett.* **14**(1), 34–36 (2015)
6. Jain, C., Flick, P., Pan, T., Green, O., Aluru, S.: An adaptive parallel algorithm for computing connected components. *IEEE Trans. Parallel Distrib. Syst.* **28**(9), 2428–2439 (2017)
7. Sharma, S., Shamhavi, S., Chaudhary, S., Khan, A.: Improvement of 16×16 playfair cipher using random number generator. *Int. J. Comput. Appl.* **94**(1), 1–8 (2014). Available: <https://doi.org/10.5120/16304-5527>
8. Dhenakaran, S.S., Ilayaraja, M.: Extension of playfair cipher using 16×16 matrix. *Int. J. Comput. Appl.* **48**(7), 37–41 (2012). Available: <https://doi.org/10.5120/7363-0192>

Text Classification Using Multilingual Sentence Embeddings



Anant Saraswat, Kumar Abhishek, and Sheshank Kumar

Abstract Training a machine learning model always requires an ample quantity of data as the effectiveness of the model depends on how good it is trained. It is not always that easy, using Zero-shot learning this paper is concentrated on intent classification in different speeches given the model is trained only in one language. Several machine learning models were developed and their outputs were compared to understand the concept of how Zero-shot learning (ZSL) works. It is grounded on the idea that the sentence embedding of two sentences of different languages having the same sense must be similar. In order to predict the intent of sentences in different languages, every sentence must be signified in the same manner and in the same vector space. Here comes the Language Agnostic Sentence Representations or LASER which uses a single encoder to give sentence embedding vector for sentences in any language. The motive of this paper is to assess how useful these sentence representations are in the context of a particular scenario.

Keywords Zero-shot learning · Sentence embeddings · Machine learning models · Vector space · LASER · Intent classification

1 Introduction

In the past decade, the trend of enabling a computer with tools to understand the human language has picked up a very high speed. Seeing the potential of understanding given text or speech in its application the automation of the task seemed

A. Saraswat (✉) · K. Abhishek · S. Kumar

Department of Computer Science and Engineering, National Institute of Technology, Patna, India
e-mail: anant.saraswat@live.com

K. Abhishek

e-mail: kumar.abhishek@nitp.ac.in

S. Kumar

e-mail: sheshank238mukul@gmail.com

very necessary. People always want ease in their life so the use of an automatic process is more favoured. The amount of data present in the form of text and speech is so high and is increasing at an exponential rate that the manual labour to process this much amount of data is not practically possible. Also, this understanding must be stored in a certified manner for others to use it in the future. This paper describes the process of dealing with text data to categorise them in particular groups.

Natural Language Processing (NLP) is the ability of a machine to interpret human language as it is written or verbalised. NLP is applied to interpret free text and make it analysable. Over the past decade, various models have been developed in this particular field. Ronan and Jason [1] in 2008 created a convolutional neural network, which extracts various features from a sentence such as parts of speech, chunks, named entities, etc. Young [2] in 2018 discussed various deep learning models and tasks performed by them in NLP and talked around the past, present and future of deep neural networks in NLP.

But the problem comes when for a simple computer various speakers or various texts are in dissimilar languages. Like humans, it is possible to train a computer program to understand several languages at once with some modifications in the training process. Hull [3] in 1996 tried to create a multilingual information retrieval system and examined the feasibility of such systems. He also expressed that the outcome of the efficiency of such multilingual systems is somewhat less than that of monolingual systems. He leaned out some features which can be tuned to create the multilingual information retrieval systems. McCarley and Jeffery [4] in 2002 had created a system that dispenses with the answer of a query where both answer and query may come from different languages. They designed the model such that given a query in one language answers in another language can be regained. Besides, seeing the answer in second language the query in third language can be created and so on. As the prospect of multiple languages came into the fold, a parallel corpus was needed in accordance to tell the program what a sentence in one language means in another language. It can be interpreted as a supervised approach in which the program is told the desired output along with the input. The issue which occurs with this requirement is the presence of such parallel corpora for different languages. On the one hand, first language can have a huge quantity of data labelled in particular classes, but on the other hand, the second language may not have enough data to train the model. One resolution for this problem is to assign a human being to translate the data from a first language to a second language with proper class labels. But this approach may suffer from human error and the cost of this overture is also very high.

Zero-Shot Learning (ZSL) comes into the picture at this degree. It grants us the ability to perform a task without training for that task. In ZSL, we consider two similar tasks, in this case, intent classification for two different languages. For first language which is English, there are enough data to develop a classifier to predict correct intent given a text. The same classifier will be utilised to predict intent in other languages such as Hindi or Spanish.

This paper describes the approaches required to create a classifier to predict intent in various languages given the classifier is trained only English data. This paper is divided into the following sections. Section 2 will contain related work. In Sect. 3,

motive and data collection are discussed. Section 4 deals with the architecture and includes the training process of the classifiers. Section 5 contains the results and Sect. 6 provides conclusions and future work.

2 Related Work

In the past, there had been many works performed in the field of Natural Language Processing, especially in intent classification. Conneau [5] in 2016 created a deep convolutional network for text classification. Lai and Siwei [6] in 2015 also created a convolutional neural network for text classification using the word representations. They used max-pooling layer to extract the important words from the text. Pang and Bo [7] classified a movie review as positive or negative using Naïve Bayes and Support Vector Machines.

Sebastian [8] in 2002 gave an inductive process to ascertain the characteristics of predefined categories by looking at the text belonging to them. These characteristics are then used to categorise the new text in classes. Socher and Richard [9] in 2013 created a model that identified objects in unseen images. The model was trained on text corpora which had descriptions of how objects look like. Conneau [10] in 2018 created an evaluation set for cross-lingual Language Understanding by extending the Multi-Genre Natural Language Inference Corpus to 15 different languages. Their thought was to produce similar embedding for a sentence in English and its translation in another language so that when a model is trained on the English dataset, it can be used to classify sentences in other languages as well. Grave and Edouard [11] in 2018 described how they trained word representations for 157 languages. Their training data consisted of the free online encyclopedia, Wikipedia. They introduced word analogy datasets for French, Hindi and Polish to evaluate these word vectors.

Correa and Renato [12] in 2002 compared the result of various classifiers including neural networks and Naïve Bayes Classifiers for text classification. Artetxe and Mikel [13] in 2018 produced a model to represent multilingual sentence representations for 93 languages. They used a BiLSTM encoder along with a shared BPE vocabulary for all the languages. These representations then used to train a model for English data and transferred to other languages for predictions without any modification. Cross-lingual language inference is one of the applications of such model. They achieved great accuracy in multilingual document classification. They innovated a new test set for 122 languages based on Tatoeba Corpus and showed that the sentence embedding provides strong results in a similarity of the same sentences in different languages.

3 Data Collection

The data employed for training purposes were collected from the conversation of a chat-bot which was being used to solve consumers' queries. The bot answered the

user's queries related to the services provided by some product manufacturers. The bot has been live for almost a year and the user queries and bot responses to the user queries were logged in the servers.

These conversation logs were curated to create an intent list and multiple question variants used by the users querying about that intent. The dataset had 48 intents and 920 question variants, i.e. Approximately 20 question variants per intent. The data set was balanced as it took in nearly the same number of examples per intent. During the curation process, care was taken to weed out repetitive question variants and make sure that the variants were proper representatives of the intent class they belonged to. Figure 1 shows the amount of questions for each intent. Here intent denotes the category to which any sentence or query can belong to.

The data collected was in English, it was converted to Hindi, Arabic and Spanish using Google Translate. The embedding of questions in each language was calculated using the LASER model. It was simulated that for one question the embedding of two languages is identical, i.e. their cosine similarity should be 1.0. To prove this point the cosine similarity between pairs of two languages was calculated which was as follows (Table 1).

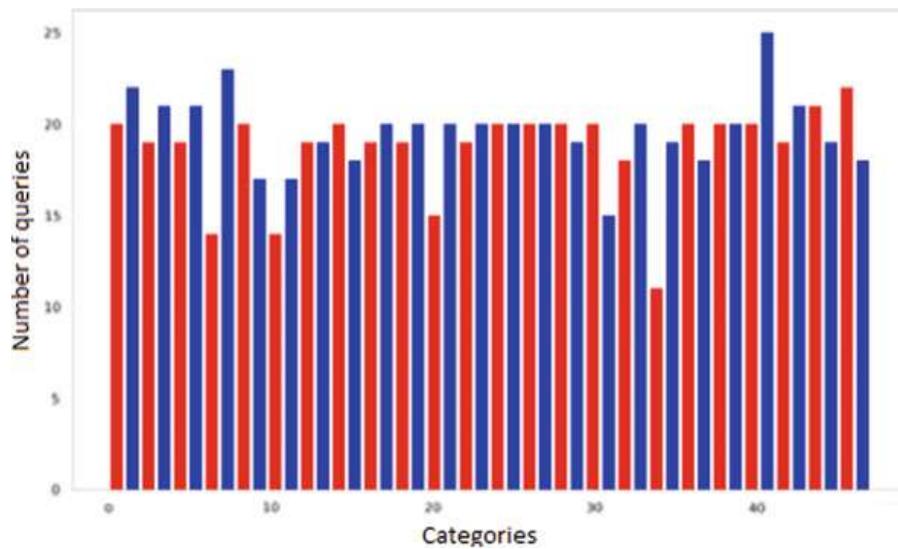


Fig. 1 Creating list of words

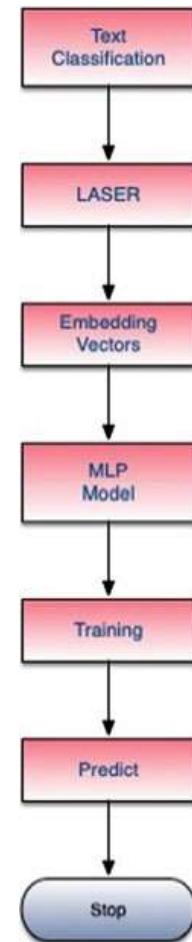
Table 1 Cosine similarity between language pairs

	English	Hindi	Arabic	Spanish
English	1.0	0.79	0.85	0.88
Hindi	0.79	1.0	0.78	0.78
Arabic	0.85	0.78	1.0	0.87
Spanish	0.88	0.78	0.87	1.0

4 Model Architecture

Once the data containing the questions related to various topics have been collected, the next step was to represent these questions in vector space for English language and other languages as well. The compiled data were in English. To translate these questions in various languages Google Translation was used. The other languages included Hindi, Spanish and Arabic. Artetxe and Mikel [13] had provided an encoder to convert the sentence of any language into a sentence embedding. Using that encoder the sentences were represented as vectors of dimension 1024. This vector now can be used as input features for each sentence. For training uses, the English sentence representations were used as input and their label as outputs. Three different models were trained over English data. They were Random Forest Classifier, Support Vector Machine and Multilayer Perceptron model. When the models were trained initially, the sentence representations of different languages were fed to the model to provide their labels (Fig. 2).

Fig. 2 Work flow



4.1 Training Classifiers

Random Forest is a classifier that uses decision trees as its constituent. The data set used in this paper contains the embedding vectors of queries. The common size of the vectors is 1024. These 1024 numeric values are treated as features. A decision tree takes the available set of features and successively selects the most important feature among them and divides the dataset into classes. At each iteration the size of unlabelled dataset decreases. In Random Forest there are various decision trees that take different subsets of features to predict the class.

In random forest first the number of decision trees to be used is decided. This number can vary among different random forest classifiers even for the same data. There is a trade-off between number of decision trees and number of features used by each decision tree to predict the categories. If the number of decision trees increases the number of features per decision tree decreases and vice versa. The choice of features for each decision tree and number of decision trees are vital elements in the classification process. So various combinations of features are used in successive training for a particular number of decision trees in a random forest classifier. Similarly, different random forest classifiers with a different number of decision trees are also trained.

The output class of random forest is the mode of all the classes predicted by the decision trees. The advantage of taking various decision trees is to leave out other features and try to predict the class based on a subset of features. Not all the trees will have the best-suited features that's why the collective output of all the decision trees is taken. In this approach, various numbers of decision trees are taken in creating a random forest classifier. It was observed that increasing the number of decision trees increases the accuracy significantly, but after a point, the accuracy becomes stagnant and stops improving shown in Fig. 3. The maximum accuracy that was achieved was 0.91 when the number of decision trees was 100. As observed when the number of decision trees was 110 and 120 the accuracy was almost similar. Random forest tends to show better accuracies when the number of trees increases.

Fig. 3 Random forest accuracy plot

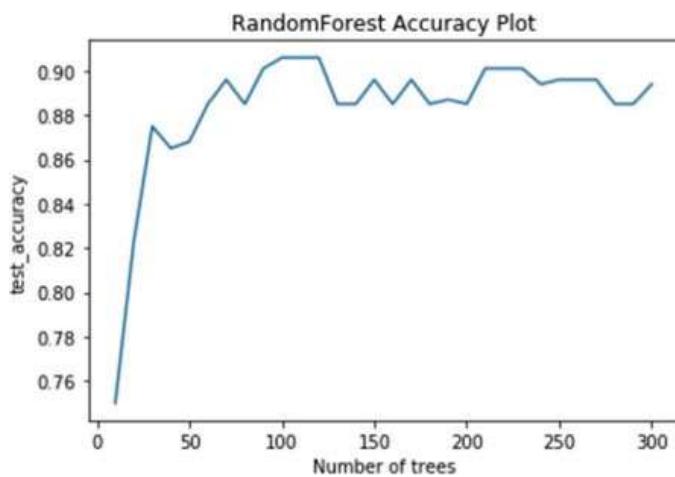
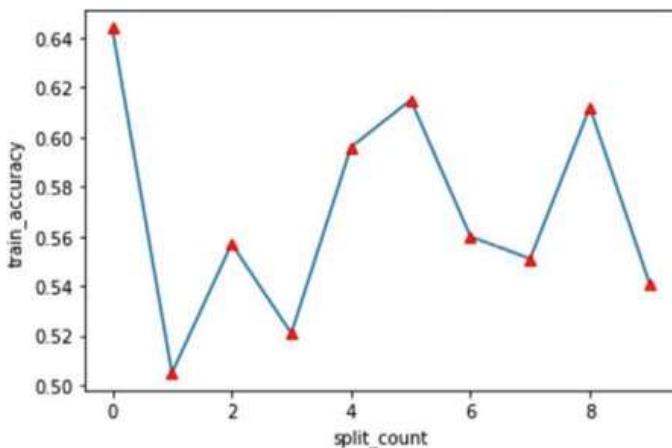


Fig. 4 Support vector machine accuracy plot



Support Vector Machine is used for classification purposes as well as in regression. In this algorithm, each data sample is plotted in n -dimensional vector space where n is the number of features for each sample. Then the classification is done by finding a hyper-plane that divides the vector spaces in the given number of classes. There are various kernels that can be used for classifying the input samples. In this approach ‘linear’ and ‘rbf’ kernels were used and it was observed that the accuracy of ‘linear’ kernel was way better than that of ‘rbf’ kernel. In the training process of Support Vector Machine for classification, ten different classifiers were trained. The difference among all these classifiers were the data set on which they were trained. For each classifier, a random set of datasets which was 80% of whole dataset was chosen. This selection was kept in mind that in each set the overall percentage of queries for each category is similar to the original dataset. This is known as Stratified Split. The accuracy of the classifier was 0.64. Figure 4 shows accuracy of models trained in different splits. By splits, it is meant the different set of dataset created on stratified random selection. This is a neural network architecture composed of various individual perceptron layers, and each layer is composed of McCulloch-Pitts neuron. It consists of an input layer, an output layer and probably one or more hidden layers. The number of neurons in the input layer is the number of features of the single input sample. The number of neurons in output layer is equal to the number of classes to which a single sample may belong to. The number of hidden layers and the number of nodes in each layer is decided by hit and try method of keeping track of accuracy and loss of the training process for different combinations. In this model, the input dimension is 1024 and the output dimension is 48. A wide range of combination of a number of layers and number of neurons in each layer were used while deciding the final architecture of the multi-layer perceptron model.

Figure 5 shows the final architecture used for training purposes. There are few more features which decide the accuracy of the model which includes learning-rate, dropout and number of epochs. These are known as hyperparameters. Stratified K Fold was used for training different models with different training data. It was made sure that the training and testing data for each model had a descent ratio of input samples for each class. Training on a multi-Layer perceptron model requires giving

Fig. 5 Model architecture

Layer (type)	Output Shape	Param #
dense_37 (Dense)	(None, 100)	102500
dropout_28 (Dropout)	(None, 100)	0
dense_38 (Dense)	(None, 100)	10100
dropout_29 (Dropout)	(None, 100)	0
dense_39 (Dense)	(None, 100)	10100
dropout_30 (Dropout)	(None, 100)	0
dense_40 (Dense)	(None, 48)	4848

Total params: 127,548
Trainable params: 127,548
Non-trainable params: 0

the same data again and again until the model learns up to an optimal level. Feeding whole training data once corresponds to one epoch. As the epoch increases the weight change using the cost function and the accuracy increases. To decide when the model starts overfitting the accuracy of training dataset is compared with the accuracy of testing dataset. As long as the training and testing accuracy keeps improving the epochs are running, but when the test accuracy stops to improve for more than a set limit, then the training is stopped. This is known as early stopping. The accuracy of the final model was 0.89. Figure 6 shows how the accuracy for train and test set changed as epochs were completed. It can be seen that in the last few epochs the test accuracy was not changing. This was the criteria of early stopping. In this figure, the horizontal axis denotes the epochs while the vertical axis denotes the test and train accuracy for successive epochs. It is observed that both train and test accuracy increases to a particular point and then the test accuracy stopped improving.

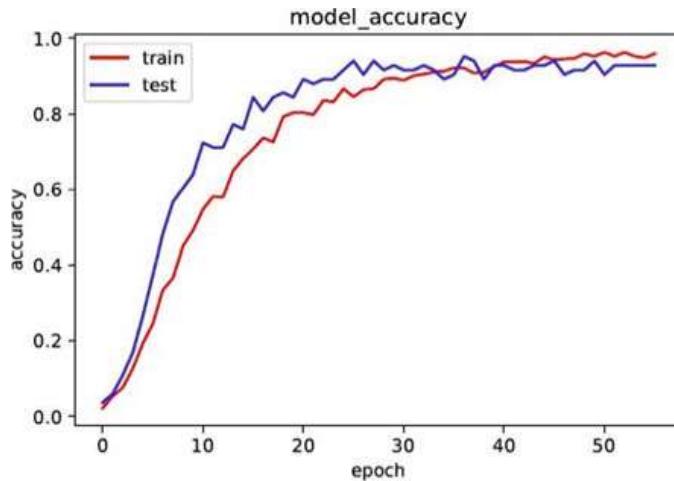
Fig. 6 Multi-Layer perceptron model accuracy plot

Table 2 Accuracies of classifiers for different languages

Classifier	English	Hindi	Arabic	Spanish
SVM	0.64	0.52	0.61	0.65
RFC	0.91	0.63	0.83	0.87
MLP	0.90	0.76	0.91	0.93

5 Results

After training the classifiers they were tested on three different languages for intent classification. Data in Hindi, Arabic and Spanish were created by translating the text of English using Google Translation. Embeddings for each sentence of all three languages were obtained using LASER. The stats showed (Table 2) that while training the random forest gave better accuracy, but in testing the accuracy of multi-layer perceptron network was very promising. SVM was not that efficient but consistency can be seen in training and testing accuracies. While in the case of random forest, the training accuracy was higher than testing accuracies. One more deduction that can be made is the accuracy of a particular language. It showed that Hindi was not performing well while on the other hand Spanish always gave better results.

6 Conclusion and Future Work

Considering the results of the classifiers it was observed that the embedding obtained by using the encoder from LASER it can be deduced that the embeddings are efficiently representing the sentence in various languages. Using different classifiers created a comparison between them and can be used as a reference while deciding which classifier to use in intent classification. The data used for creating the BPE vocabulary is highly grammatical. Thus, if the encoder is tested against real-time data its accuracy may degrade. Also the effectiveness of this model can be tested when two different terminologies are used in the same sentences, i.e. Writing words of one language in the alphabets of second language or when there are two different sets of alphabets belonging to two different languages in the same sentence. LASER can further be trained on such mix data to better understand the sentences hence accuracy can be improved.

Acknowledgements This publication is an outcome of the R&D work undertaken project under the Visvesvaraya Ph.D. Scheme of Ministry of Electronics and Information Technology, Government of India, being implemented by Digital India Corporation.

References

1. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning (ICML '08), pp. 160–167. ACM, New York, NY, USA (2008)
2. Young, T., Hazarika, D., Poria, S., Cambria, E.: Recent trends in deep learning based natural language processing (review article). *IEEE Comput. Intell. Mag.* **13**(3), 55–75 (2018)
3. Hull, D.A., Grefenstette, G.: Querying across languages: a dictionary-based approach to multilingual information retrieval. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (1996)
4. McCarley, J.S.: Multilingual information retrieval with a transfer corpus. US patent no. 6,349,276. 19 Feb 2002
5. Conneau, A., et al.: Very deep convolutional networks for text classification. Preprint at [arXiv:1606.01781](https://arxiv.org/abs/1606.01781) (2016)
6. Lai, S., et al.: Recurrent convolutional neural networks for text classification. In: Twenty-ninth AAAI Conference on Artificial Intelligence (2015)
7. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing, Vol. 10. Association for Computational Linguistics (2002)
8. Sebastiani, Fabrizio: Machine learning in automated text categorization. *ACM Comput. Surv. (CSUR)* **34**(1), 1–47 (2002)
9. Socher, R., et al.: Zero-shot learning through cross-modal transfer. *Adv. Neural Inf. Process. Syst.* (2013)
10. Conneau, A., et al.: Xnli: evaluating cross-lingual sentence representations. Preprint at [arXiv:1809.05053](https://arxiv.org/abs/1809.05053) (2018)
11. Grave, E., et al.: Learning word vectors for 157 languages. Preprint at [arXiv:1802.06893](https://arxiv.org/abs/1802.06893) (2018)
12. Corrêa, R.F., Ludermir, T.B.: Automatic text categorization: case study. In: VII Brazilian Symposium on Neural Networks, 2002. SBRN 2002. Proceedings. IEEE (2002)
13. Artetxe, M., Schwenk, H.: Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. Preprint at [arXiv:1812.10464](https://arxiv.org/abs/1812.10464) (2018)

Real-Time Yawn Extraction for Driver's Drowsiness Detection



Sumeet Saurav, Mehul Kasliwal, Raghav Agrawal, Sanjay Singh, and Ravi Saini

Abstract Driver's drowsiness is one of the major causes of increase in the number of road accidents. Therefore, design and implementation of a real-time driver's drowsiness detection system are considered as a crucial component of the Advanced Driver Assistance System (ADAS). Along with other physiological parameters, yawn is often considered as one of the most important indicators of drowsiness in human. Thus, in this work we propose an efficient and nonintrusive system for yawn detection targeted toward real-time drowsiness detection. The proposed algorithmic pipeline consists of a face and facial landmark detector for face detection and landmark localization, a scheme for extracting feature named mouth aspect ratio (MAR) characterizing the state of the mouth (open/close) in each frame, and a classifier to classify the state of the mouth in a temporal window of some fixed number of frames. The performance of the proposed approach has been validated on a manually annotated dataset extracted from the widely used yawn detection dataset called YawDD. The proposed approach has achieved an accuracy of 99.25% along with F1 score of 98.00% and runs at 30 frames per second (FPS).

Keywords Yawn detection · Extreme learning machine (ELM) · Advanced driver assistance system (ADAS) · Mouth Aspect Ratio (MAR) · SVM · DNN

S. Saurav (✉)

Academy of Scientific and Innovative Research (AcSIR), Chennai, India
e-mail: sumeet@ceeri.res.in

S. Saurav · S. Singh · R. Saini

CSIR-Central Electronics Engineering Research Institute, Pilani, India
e-mail: sanjay@ceeri.res.in

R. Saini

e-mail: ravi@ceeri.res.in

M. Kasliwal · R. Agrawal

BITS-Pilani, Goa Campus, Goa, India
e-mail: mehulkasliwal33@gmail.com

R. Agrawal

e-mail: f20160079@goa.bits-pilani.ac.in

1 Introduction

According to a report of the American National Highway Traffic Safety Administration (NHTSA) [1], in 2016 alone 803 fatalities were reported due to drowsy driving. Although the number of fatalities decreased by 3.5% from 832 in 2015 to 803 in 2016, this situation still needs to be addressed seriously because there is a major risk involved in such accidents. Development of driver fatigue and drowsiness detection systems can not only help in reducing the number of road accidents, but can also be of a major significance to professionals such as nuclear reactor operators, air traffic controllers, construction workers, wherein there is requirement of undivided attention.

Yawning can be best defined as an involuntary act of gaping of mouth and deep inhalation followed by shallow exhalation. It is an important indicator to detect drowsiness and fatigue at an early stage in order to prevent mishaps in work environments. People often yawn when fatigue or drowsiness occurs; hence, the detection of mouth state by a camera is an effective method for detecting fatigue or drowsiness.

While yawning, mouth opens wide and a change is observed in geometric features of mouth; therefore, previous researches focused on detecting geometric features of mouth. For example, in [2] the authors detected yawning from the openness of mouth using the ratio of the height of mouth to the width of the mouth, whereas in [3] the openness of mouth was calculated using the ratio of the height of mouth to width and yawning was asserted when the ratio was more than 0.5 in 20 frames out of 30. The work in [4] has made use of the geometric features of mouth region to make an eigenvector and input it to a back propagation artificial neural network, the output of which was categorized into three different mouth states—normal, yawning, or talking state, respectively. Various color-based approaches have also been proposed for mouth and lips detection. For example, the authors in [5] have proposed the detection of yawn based on difference image between successive images, and then yawn was detected using the distance between the mid-point of nostrils and chin. The localization of the chin and the nostrils are done using directional integral projection method. In [6], the authors have used gravity-center template for face detection followed by mouth corner extraction using gray projection and Gabor wavelets. At last, LDA was used to classify feature vectors to detect yawning. The authors in [7] segmented the image texture regions using stochastic region merging strategy, and then skin color and texture models are used for classification. The work reported in [8] used two cameras to their advantage: a low-resolution camera for the face and a high-resolution one for the mouth. Then, Haar-like features were used to detect yawning based on the ratio of the height to width of mouth. Most of the papers suggest color-based approach is not good for yawn detection when substantial changes in head pose and illumination are expected. Work presented in [9] used the static supervised classification based on log-polar signatures approach for open or closed mouth detection, whereas the authors in [10] used IPPG techniques to detect eye blinking and yawning. System presented in [11] used face-extraction-based support vector machines (SVM) and circular Hough transforms (CHT) to detect if the

driver's mouth is open or not. Authors in [12] detected rate and amount of changes in mouth using back projection theory through a modified implementation of the Viola–Jones algorithm for face and mouth detections in order to detect yawning. Another method presented in [13] proposed to detect yawning using geometric and appearance features of mouth and eye regions and were also able to detect hand-covered and uncovered yawns. In [14], authors have proposed neural networks and recurrent neural networks to extract spatial features and long short-term memory (LSTM) networks to analyze temporal characteristics.

Our proposed method for detection of a yawn, works in real time and is more practical to implement in real-world fatigue and drowsiness detection systems. In our method, a camera is installed on the dashboard of the car, which continuously captures the driver's movements. In order to detect a yawn, the first step is to detect the face of the driver and then extract the features of the mouth region using mouth aspect ratio (MAR) technique. The MAR from many frames is calculated using the landmarks of the mouth which gives an indication of the degree of openness of the mouth and also captures the temporal evolution during the act of yawning. Finally, a classifier is trained on these temporal features to classify it into either two mouth states—yawning and normal or three mouth states—yawning, talking, and normal. In the former case, the proposed approach has merged the other states of the mouth like talking, laughing, and singing into the normal state.

The remainder of this paper is divided as follows: Section 2 presents a detailed explanation of our proposed method. Experimental results and discussions are dealt with in Sect. 3. We finally conclude the paper with conclusive remarks in Sect. 4.

2 Proposed Method

The proposed yawn detection scheme as shown in Fig. 1 first extracts frames from video feed and passes them to the face and facial landmark detector, which detects the face and then localizes 68 landmarks of the face. Mouth aspect ratio (MAR) value is calculated for each frame using landmarks of the mouth, and then a temporal window of F -dimensional features is fed to a classifier for classifying the temporal window in either three states: normal, talking, and yawning or two states: normal and yawning. A brief detail of different components used in the proposed pipeline has been discussed below.

2.1 Face Detection and Landmark Localization

For face detection, we used OpenCV deep learning-based face detector. Once detected, faces are then passed to a landmark localizer made available by the author in [15] which is an open-sourced version of the work discussed in [16]. The landmark detector accepts initial input as the facial image and then localizes 68 face landmarks.

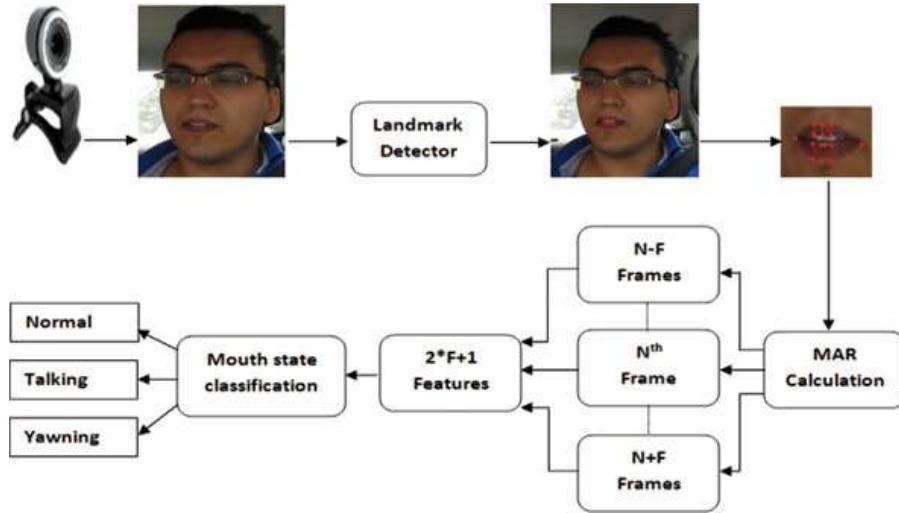


Fig. 1 Proposed yawn detection algorithmic pipeline

Points detected are located on the face such as the corners of the mouth, on the eyes, along the eyebrows and jawline and so forth. The landmarks detected on the mouth regions could be also seen in Fig. 1.

2.2 Mouth Aspect Ratio (MAR)

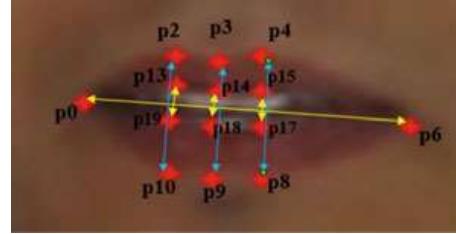
During a yawn, the degree of openness of the mouth varies from person to person; hence, a well-defined formula for calculating it is described in this section. We propose a formula called mouth aspect ratio (MAR) for calculating the degree of openness of the mouth by finding the ratio of the vertical height to the horizontal widths of the mouth as given in (1).

$$\text{MAR} = \frac{\|p2 - p10\| + \|p3 - p9\| + \|p4 - p8\| + \|p13 - p19\| + \|p14 - p18\| + \|p15 - p17\|}{6.0 * \|p0 - p6\|} \quad (1)$$

where $p1, p2, p3, \dots, p19$ are the 2D locations (x and y co-ordinates) of the facial landmarks from the mouth region as marked in Fig. 2.

The MAR value of the subject does not vary much when the mouth is closed or when the person is smiling or talking, but there is a significant surge in its value when person is yawning. Our proposed formula clearly distinguishes a laugh from a yawn, because when a person is laughing, the corners of his mouth tend to get farther apart as compared to when he is yawning. Hence, the horizontal distance between the corners of the mouth is larger when laughing as compared to yawning. This increase in horizontal distance during laughing tends to lower down the MAR

Fig. 2 2D location of the facial landmarks from the mouth region

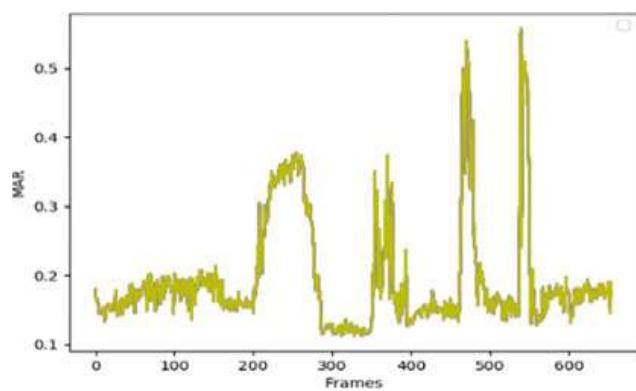


value. A similar feature called eye aspect ratio (EAR) was proposed in [17] which uses a drop in EAR value to detect whether the eye is closed or not. Like EAR, MAR is also partially person and head pose insensitive, and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face.

A plot of MAR values computed frame-wise using webcam captured frames has been shown in Fig. 3. Different areas of the plot signify different mouth states of the person. When the mouth is in normal state, the value of MAR typically lies below 0.20. This could be easily seen from the plot shown below (frames 0–200, frames 280–350, frames 400–460, etc.). However, when the person is laughing (frames 200–280), singing, and talking (frames 350–400), the MAR values lie in the range of 0.3–0.5. Finally, in yawn state the value of MAR is above 0.5 (two large peaks between 400 and 600 frames).

Now, the obvious question arises is if we are able to quantify the typical range of MAR values corresponding to different mouth state of a person, then what is the requirement of a classifier. By just setting different threshold values, we should be able to categorize the mouth state of a person. However, in real-world situation, the above logic might not hold true. This is because based on one frame MAR value we cannot predict the actual mouth state of the person. It generally does not hold true that a large value of the MAR means a person is yawning. A large value of the MAR may occur when a person opens his/her mouth intentionally or the MAR captures a short random fluctuation of the landmarks. Therefore, there is a requirement of classifier trained using MAR features obtained from a large temporal window comprising of a number of frames. The classifier learns inherent patterns associated with different mouth states using these feature vectors and should be able

Fig. 3 Plot showing variation of MAR values with frame



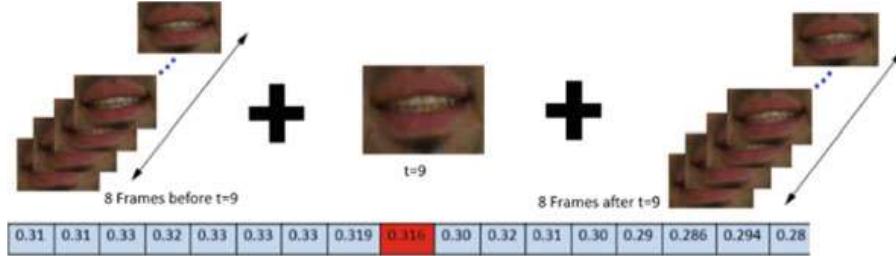


Fig. 4 Scheme used for temporal feature extraction based on MAR value

to have better generalization ability compared to simple threshold-based mouth state classification scheme.

2.3 Feature Extraction

As the degree of openness of the mouth varies from person to person during yawning, a single value of MAR for detection of a yawn can be misleading. To overcome this, we use a temporal window of adjacent frames to detect whether a person has yawned or not. We experimented with temporal window of different frame sizes wherein to calculate the state of the N th frame which is the frame of interest we use frames from N to F th frame to $N + F$ th frame, resulting in a $2 * F + 1$ -dimensional feature vector as shown in Fig. 4 for a value of F equal to 8. An important point to note here is that the ground truth label of the temporal window (which is essentially one sample training data) is empirically set equal to the ground truth label of the frame of interest.

2.4 Classifiers

We performed experiments with three well-known classifiers, viz. the support vector machine (SVM) classifier, the deep neural network (DNN), and the extreme learning machine (ELM) classifier.

An extreme learning machine (ELM) classifier as initially described in [18] is a machine learning technique which comes under the category of single-hidden-layer feed-forward neural network, wherein the connection weights and bias of the hidden neurons are defined randomly and are not learned as usually done in the case of traditional neural network training. Systematic representation of an extreme learning machine classifier is shown in Fig. 5, where one could find that there is only one hidden layer and the rest of the two layers are used as input and output layers. Regarding further details about the ELM classifier, we refer to [19].

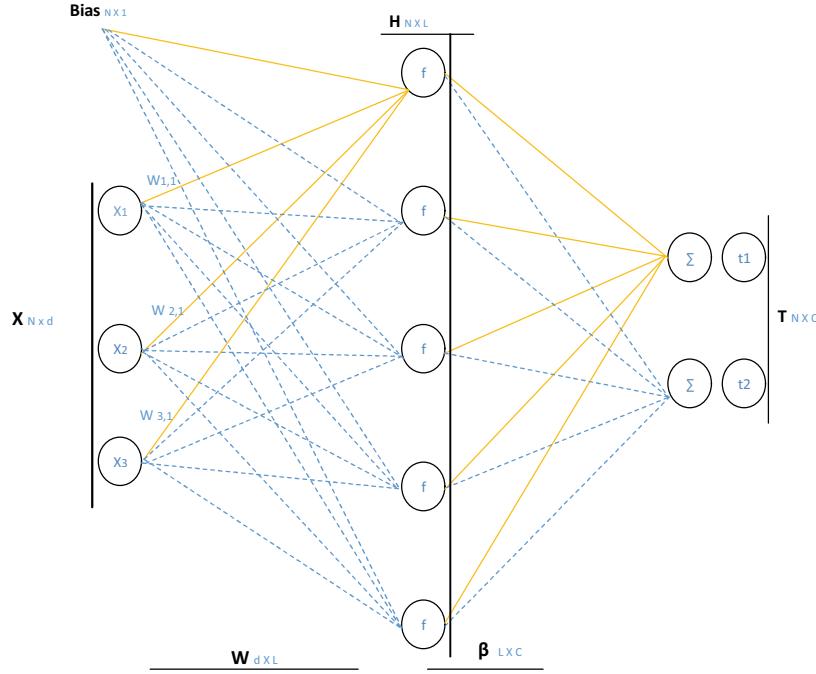
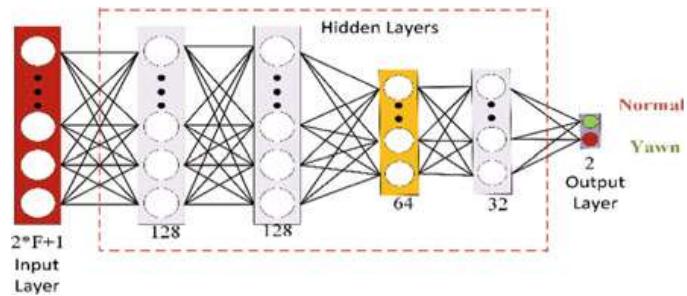


Fig. 5 Structure of a single-layer feed-forward network

Another classifier which has been used in the experiments is the deep neural network (DNN). As the name indicates, a DNN is a feed-forward neural network having a number of hidden layers apart from the usual input and output layers as shown in Fig. 6. The DNN used in this work contains four hidden layers having 128, 128, 64, and 32 neurons, respectively. Each hidden layer is followed by a batch normalization layer, and the final layer is the Softmax layer. For the nonlinear transformation between layers, the proposed DNN architecture has used exponential linear unit (ELU) [20] with default parameter.

The final classifier used is the support vector machine (SVM) classifier. The SVM classifier also called maximum-margin classifier comes under the category of discriminative classifier. Using labeled training data, it learns an optimal hyperplane separating different classes. More details about SVM can be found in [21].

Fig. 6 Deep neural network architecture used in the proposed work



2.5 Evaluation Metrics

In order to validate the performance of the proposed pipeline, several widely known evaluation metrics have been used in this work. Classification accuracy is widely used performance metric which is defined as the number of correct predictions made divided by the total number of predictions. In literature, there is an accuracy paradox which demands additional performance measures in addition to the classification accuracy for the purpose of evaluating a classifier. Therefore, in this work we also used other performance measures like precision, recall, and F1 score for the evaluation of the performance of different classifiers. Precision is defined as the ratio of the number of correct results (true positives) to the number of all obtained results (sum of true positives and false positive) and is expressed as shown in (2).

$$\text{Precision} = \frac{\#\text{true positives}}{\#\text{true positives} + \#\text{false positives}} \quad (2)$$

Precision indicates the exactness of a classifier, and a low value of precision is an indication of a large number of false positives. Another widely used performance evaluation metric is the recall which is defined as the ratio of the number of correct results to the total number of results that should have been obtained and is expressed as in (3).

$$\text{Recall} = \frac{\#\text{true positives}}{\#\text{true positives} + \#\text{false negatives}} \quad (3)$$

Recall is the indicator of classifiers completeness, and a low value of recall indicates many false negatives in the prediction. Therefore, a balance is often required between the precision and recall which in turn is measured by another evaluation metric called F1 score and is expressed in (4).

$$F1\text{-Score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}} \quad (4)$$

3 Experiments and Results

In this section, we discuss the dataset and different experiments which were performed in this work.

3.1 Dataset

We used YawDD dataset [22] in our experiments. The database consists of VGA resolution color videos captured at 30 frames per second (fps). There are a total of 322 videos which consist of both male and female drivers, from different ethnicities, with and without glasses/sunglasses, and in both normal and yawning conditions. The videos have been captured under varying illumination conditions (sunny, rainy, and cloudy environments) so that the algorithms trained using the dataset will be robust to the changes in the lightening conditions. During dataset preparation, the subjects were asked to perform three tasks: normal driving, talking, and singing while driving, and yawning while driving. Among these videos, we have used only the ones in which the camera was placed on the dashboard of the car because the results in [12] justify that the yawn detection generates better results when the camera captures front view of the driver as compared to the side view. Hence, our proposed model has been trained and tested on only the front view videos. There were 29 videos of 16 males and 13 females, with and without glasses/sunglasses, from different ethnicities taken in sunny conditions through a camera placed on the dashboard of the car. A researcher was asked to manually annotate frames extracted from videos from dataset into three categories—normal, talking, and yawning.

3.2 Results and Discussion

From the 29 videos obtained from the database, our training set consists of 22 videos from which a number of temporal window samples having F -dimensional features were extracted. For the value of F equal to 32, the number of 65-dimensional training samples was 41,902 having 13,426 normal class samples, 24,082 talking class samples, and the rest, i.e., 4394 yawn class samples. The remaining seven videos were used to evaluate the trained model. For the same value of F , the testing data consists of a total of 18,846 samples comprising 5966 normal class samples, 10,597 talking class samples, and 1807 samples from the yawn class. The features were extracted using dlib's facial landmark detector in Python environment using OpenCV deep learning-based face detector. The training and testing of different classifiers were done in Python environment on an Intel i7-8700 K processor system with 16 GB RAM.

In the first set of experiments, we dealt with the task of mouth state categorization into three categories: normal, talking, and yawning (Task-1). We experimented with three different values of F and evaluated the performance of different classifiers. For support vector machine classifier, we used grid-search using tenfold cross-validation to obtain the optimal values of the hyperparameters. From the grid-search experiment, we found that the optimal classifier to be used is ‘linear’ and the corresponding value of the margin parameter ‘ C ’ is 1000. For training and testing the SVM classifier, we used the open-source Scikit-learn machine learning library [23]. Using these

parameters, we trained the classifier again on the full training data and then tested the trained classifier model on the testing data which is completely independent of the training samples. Again, for the extreme learning machine classifier, we performed tenfold cross-validation using different number of hidden neurons and reported the performance of the optimal model tested on the testing data. The library used for performing training and testing of extreme learning machine classifier has been made available by the authors of work presented in [24]. Finally, for DNN training we used deep learning framework called Keras using TensorFlow as backend. We used Adam optimizer with a learning rate of 1e-5 and a batch size of 64. Parameters of different classifiers used were kept uniform for different values of F .

The performance of different classifiers trained using 65-dimensional features obtained from temporal window of size 65 corresponding to the value of F equal to 32 is shown in Table 1. Here, the task is to classify the mouth state of the person into three different states, viz. normal, talking, and yawning. From the table, we could find that the performance of the support vector machine classifier is relatively better compared to the other two classifiers. For value of F equal to 16, i.e., using 33-dimensional features, the performance of the classifiers has been reported in Table 2. Here, again we can find that the SVM classifier performed better than the ELM classifier and DNN. Finally, the performance of the classifiers trained using 17-dimensional feature obtained for value of F equal to 8 has been listed in Table 3. With 17-dimensionl features, again the performance of different classifiers remained unchanged and SVM again won the race.

Table 1 Performance of classifiers evaluated using temporal window of length 65 (Task-1)

Method	Testing accuracy	Avg. precision	Avg. recall	Avg. F1 score
Support vector machine (SVM)	94.51	94.00	93.33	93.67
Extreme learning machine (ELM)	91.94	92.67	88.33	90.33
Deep neural networks (DNN)	93.25	93.00	91.00	92.00

Table 2 Performance of classifiers evaluated using temporal window of length 33 (Task-1)

Method	Testing accuracy	Avg. precision	Avg. recall	Avg. F1 score
Support vector machine (SVM)	96.97	96.33	97.00	97.00
Extreme learning machine (ELM)	94.32	94.67	93.00	93.67
Deep neural networks (DNN)	95.26	95.33	94.66	95.00

Table 3 Performance of classifiers evaluated using temporal window of length 17 (Task-1)

Method	Testing accuracy	Avg. precision	Avg. recall	Avg. F1 score
Support vector machine (SVM)	97.04	96.33	97.00	96.67
Extreme learning machine (ELM)	95.57	96.00	94.67	95.00
Deep neural networks (DNN)	95.85	96.00	94.67	95.67

From the tables listed above, we can make a conclusive remark that the optimal value of F which is necessary to categorize mouth states into three different categories, viz. normal, talking, and yawning is 8, i.e., a temporal window of 17 frames is required to differentiate mouth states with high accuracy.

In another set of experiments, we dealt with the task of mouth state categorization into two categories: normal and yawning (Task-2). The training and testing data for the experiments were generated from the previously used data with just a minor modification. In this case, we merged the data from the normal and talking class and treated the combined data as the data of the normal class. Thus, these experiments consist of training a binary classifier corresponding to SVM, ELM, and DNN for three different values of F , i.e., 32, 16, and 8. The performance of different classifiers trained using 65-dimensional feature vector corresponding to the value of F equal to 32 has been shown in Table 4. As expected, in this case also the SVM classifier performed relatively well compared to the other counterparts. The values of different classifier evaluation metrics for the value of F equal to 16 are shown in Table 5. Here,

Table 4 Performance of classifiers evaluated using temporal window of length 65 (Task-2)

Method	Testing accuracy	Avg. precision	Avg. recall	Avg. F1 score
Support vector machine (SVM)	98.51	96.00	95.00	95.50
Extreme learning machine (ELM)	97.88	95.00	92.50	94.00
Deep neural networks (DNN)	97.99	95.50	93.00	94.00

Table 5 Performance of classifiers evaluated using temporal window of length 33 (Task-2)

Method	Testing accuracy	Avg. precision	Avg. recall	Avg. F1 score
Support vector machine (SVM)	99.25	97.50	98.50	98.00
Extreme learning machine (ELM)	98.86	97.50	96.00	96.50
Deep neural networks (DNN)	98.97	97.00	97.00	97.00

Table 6 Performance of classifiers evaluated using temporal window of length 17 (Task-2)

Method	Testing accuracy	Avg. precision	Avg. recall	Avg. F1 score
Support vector machine (SVM)	99.25	97.50	98.50	98.00
Extreme learning machine (ELM)	99.19	98.50	97.00	98.00
Deep neural networks (DNN)	99.16	98.00	96.50	97.50

again the performance of SVM classifier is superior to the ELM and DNN classifiers.

Finally, our last experiment consists of training and testing of the ELM, SVM, and DNN classifiers using 17-dimensional features obtained using a temporal window of length 17 corresponding to the value of F equal to 8. The experimental results have been listed in Table 6. From the table, we can easily find that in this case, all the classifiers performed well and there is not much difference in the value of different metrics.

Similar to the task of classification of the mouth states of the person into three different categories, the value of F remained unchanged also in the task of classification of the mouth states of a person into two different categories. Therefore, we can easily conclude that the optimal value of F required for detection of different mouth states is 8 which corresponds to a temporal window of size 17 consisting of 17-dimensional MAR features.

4 Conclusion

In this paper, a method for yawn detection has been proposed which uses facial landmarks of the mouth to calculate the degree of openness of the mouth using a proposed technique called mouth aspect ratio (MAR). To identify the state of mouth, experiments were done using temporal window of different sizes and the optimal performance is obtained using a temporal window of length equal to 17. We evaluated the performance of three different classifiers, viz. the ELM, SVM, and DNN for two tasks related to classification of the driver's mouth state. The first task requires classification of the mouth state of the driver into normal, talking, and yawning, while the second task was associated with categorization of the driver's mouth state into only normal and yawning. A number of experiments were performed using a manually annotated YawDD dataset using different evaluation metrics like classification accuracy, precision, recall, and F1 score to evaluate the performance of classifiers and found that the SVM classifier outperformed the other two classifiers with a substantially large margin in case of task one and a lower margin in case of task two. Our proposed method outperforms current state-of-the-art techniques presented in [7] and [12] which give an accuracy of about 95%. Finally, our works

affirm the idea that yawning detection should be computed over a sequential video rather than a single frame.

References

1. National Center for Statistics and Analysis.: Fatal motor vehicle crashes: Overview. (Traffic Safety Facts Research Note. Report No. DOT HS 812 456). National Highway Traffic Safety Administration, Washington, DC (Oct 2017)
2. Ji, Q., Zhu, Z., Lan, P.: Real-time nonintrusive monitoring and prediction of driver fatigue. *IEEE Trans. Veh. Technol.* **53**(4), 1052–1068 (2004)
3. Wang, T., Shi, P.: Yawning detection for determining driver drowsiness. In: Proceedings of 2005 IEEE International Workshop on VLSI Design and Video Technology, 2005, pp. 373–376 (May 2005)
4. Rongben, W., Lie, G., Bingliang, T., Lisheng, J., Monitoring mouth movement for driver fatigue or distraction with one camera. In: Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems (Oct 2004)
5. Lu, Y., Wang, Z.: Detecting driver yawning in successive images. In: 2007 1st International Conference on Bioinformatics and Biomedical Engineering, pp. 581–583 (July 2007)
6. Fan, X., Yin, B.C., Sun, Y.F.: Yawning detection for monitoring driver fatigue. In: 2007 International Conference on Machine Learning and Cybernetics, Vol. 2, pp. 664–668. IEEE (Aug 2007)
7. Medeiros, R.S., Scharcanski, J., Wong, A.: Multi-scale stochastic color texture models for skin region segmentation and gesture detection. In: 2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), pp. 1–4. IEEE (July 2013)
8. Li, L., Chen, Y., Li, Z.: Yawning detection for monitoring driver fatigue based on two cameras. In: 2009 12th International IEEE Conference on Intelligent Transportation Systems, pp. 1–6. IEEE (Oct 2009)
9. Bouvier, C., Benoit, A., Caplier, A., Coulon, P.Y.: Open or closed mouth state detection: static supervised classification based on log-polar signature. In: International Conference on Advanced Concepts for Intelligent Vision Systems, pp. 1093–1102. Springer, Berlin (Oct 2008)
10. Wei, B., Lu, X., Zhang, C., Wu, X.: Efficient detection of eye blinking and yawn based on facial video utilizing IPPG technique. In: 2016 International Forum on Mechanical, Control and Automation (IFMCA 2016). Atlantis Press (Mar 2017)
11. Alioua, N., Amine, A., Rziza, M.: Driver's fatigue detection based on yawning extraction. *Int. J. Veh. Technol.* (2014)
12. Omidyeganeh, M., Shirmohammadi, S., Abtahi, S., Khurshid, A., Farhan, M., Scharcanski, J., Hariri, B., Laroche, D., Martel, L.: Yawning detection using embedded smart cameras. *IEEE Trans. Instrum. Meas.* **65**(3), 570–582 (2016)
13. Jie, Z., Mahmoud, M., Stafford-Fraser, Q., Robinson, P., Dias, E., Skrypchuk, L.: Analysis of yawning behaviour in spontaneous expressions of drowsy drivers. In: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), pp. 571–576. IEEE (May 2018)
14. Zhang, W., Su, J.: Driver yawning detection based on long short-term memory networks. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–5. IEEE (Nov 2017)
15. King, D.E.: Dlib-ml: a machine learning toolkit. *J. Mach. Learn. Res.* **10**, 1755–1758 (2009)
16. Kazemi, V., Sullivan, J.: One millisecond face alignment with an ensemble of regression trees. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1867–1874 (2014)
17. Cech, J., Soukupova, T.: Real-time eye blink detection using facial landmarks. 21st Comput. Vis. Winter Work (2016)

18. Huang, G.B.: What are extreme learning machines? Filling the gap between Frank Rosenblatt's dream and John von Neumann's puzzle. *Cognitive. Comput.* **7**(3), 263–278 (2015)
19. Vashisth, S., Saurav, S.: Histogram of oriented gradients based reduced feature for traffic sign recognition. In: 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 2206–2212. IEEE (Sep 2018)
20. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). Preprint at [arXiv:1511.07289](https://arxiv.org/abs/1511.07289) (2015)
21. Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification (2003)
22. Abtahi, S., Omidyeganeh, M., Shirmohammadi, S., Hariri, B.: YawDD: a yawning detection dataset. In: Proceedings of the 5th ACM Multimedia Systems Conference, pp. 24–28. ACM (Mar 2014)
23. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
24. Akusok, A., Björk, K.M., Miche, Y., Lendasse, A.: High-performance extreme learning machines: a complete toolbox for big data applications. *IEEE Access* **3**, 1011–1025 (2015)

FIoT: A QoS-Aware Fog-IoT Framework to Minimize Latency in IoT Applications via Fog Offloading



K. S. Arikumar  and V. Natarajan

Abstract The impact of the Internet of Things (IoT) in our daily life is rising in recent days. The automated IoT devices possess some advantages like low communication costs, minimizes human effort, and efficient resource allocation. However, reducing the latency in the data transmission between IoT devices and the storage data centers like the cloud is still a challenge. To diminish the latency for IoT applications, we propose a Fog-IoT collaborative framework (FIoT) with an offloading strategy. We further developed an analytical model for evaluating the performance of FIoT. The analytical results prove that our FIoT framework minimizes the service delays and thereby enhancing the Quality of Service (QoS) in IoT applications.

Keywords Internet of Things · Fog computing · Latency · Quality of Service · IoT applications · Offloading strategy

1 Introduction

Recently, the Internet of Things (IoT) is seeking greater attention among researchers and public people due to its various advantages [1]. IoT automates our daily life by enabling communication between things like electronic devices, home appliances, streetlights, and other things. The traditional IoT devices do not have the capability of storing the information being collected, and thus, IoT relies on another computing infrastructure like a cloud platform for storing and computation [2].

Traditionally, the cloud data centers act as the key enabler for IoT applications, which makes storage simpler and computes optimal results with its high processing capacity [3]. However, the cloud data centers for IoT applications suffer from various

K. S. Arikumar ()
St. Joseph's Institute of Technology, Chennai, India
e-mail: ksarikumar@gmail.com

V. Natarajan
Anna University, MIT Campus, Chennai, India
e-mail: natraj@mitindia.edu

limitations such as higher latency, high traffic, and lack of global mobility due to its remote location. Further, these limitations affect the delay-sensitive applications such as smart health care and fully automated smart transportation systems. Hence, an alternate data center, which is very closer to the IoT devices, is required to overcome these limitations. Recently, Cisco proposes fog computing for storing and processing the data closer to where it is generated [4].

Fog computing consists of fog nodes or fog data centers, which can be located anywhere closer to the data generating nodes. The fog nodes are an intermediate layer between IoT applications and cloud. Since the fog nodes are located very close to the IoT devices, the location-aware applications can be computed with low latency and high efficiency [5].

In this paper, we propose a Fog-IoT framework called FIoT, which minimizes latency via fog offloading. Our FIoT framework consists of three major layers. The lower layer is the IoT layer, which includes various IoT devices, IoT applications, and end users. The higher layer is the cloud layer, which includes various data centers composed of numerous high-performance rack servers. The intermediate layer between the IoT layer and cloud layer is the fog layer, which includes fog nodes.

The basic workflow of the FIoT framework is as follows. In the IoT layer, the IoT devices locally process the data generated by the sensors and actuators, and then it transmits the data either to fog nodes or to the cloud storage for further processing. In the fog layer, the fog nodes store and process the data received from the IoT layer. Here, the fog nodes can transmit the processed data to other fog nodes or the cloud data centers. In the cloud layer, the cloud data centers perform the complex computations over the received data and stores the computed data permanently. We aim to minimize the latency in IoT applications through our offloading strategy in fog nodes.

2 Related Works

In this section, we deliver a summary of the other existing IoT frameworks. We survey both the cloud-based frameworks as well as the IoT-based frameworks. We initialize with the cloud-based framework, a cloud platform called Azure has been utilized in IoT-Azure [6] to tackle the massive IoT data transmissions. The framework is completely data oriented. In [7], the cloud computing technology is exploited for automating and controlling the IoT networks. The local clouds are utilized for storing and processing the IoT data. However, the cloud-based framework always lags in the factor called latency. Hence, the fog-based frameworks started to emerge.

One of the fog-based IoT frameworks in [8] provides a general summary of design principles for IoT architecture concerning adaptability, performance, reliability, and scalability. To increase efficiency and revenues in an unlicensed spectrum, two channel access schemes and two spectrum allocation schemes are taken into account. In [9], the authors proposed a holistic architecture of the IoT health

ecosystem for discussing IoT's application in medicine and health care. For this, a transition has to make from clinic centric treatment to patient-centric health care. This transition requires a multi-layer architecture such as fog and cloud computing layers.

Another fog-based architecture in [10] deals with the outburst of IoT traffic by considering new computational, network, and storage resources. To amortize this problem, this demo provides an insight into FITOR, which is the proposed orchestration system for IoT applications. In [11], an architecture called EBA is proposed, which minimizes the energy consumption and the delay of the network in balancing the energy among all the fog nodes. The optimal transmission power and transmission rate of terminal nodes are calculated by using the best transmission power and transmission rate.

Recently, fog computing is considered as an important research direction in healthcare IoT systems. In [12], the authors represent the systematic review of fog computing in the healthcare IoT field and analyze it. The authors in [13] research about the challenges in fog and establish the current developing improvements. A taxonomy of fog computing is described based on the key features and challenges identified.

3 Fog Offloading Strategy

In this section, we present an offloading strategy for the fog nodes in the FIoT framework. In FIoT, the fog nodes collaboratively work together to accomplish the IoT application requests and tend to minimize the latency. The fog nodes work based on their available load. If a fog node has a low load, then the request from the IoT application will be processed, else that request will be offloaded to other fog nodes. If all the fog nodes are overloaded, then the request will be transferred to the cloud. In the following sections, we discuss the above-mentioned offloading strategy and fog-cloud collaborations in detail.

3.1 Communication Approaches

In our FIoT framework, we assume two approaches for communication among fog nodes. One is a centralized communication approach; another is a decentralized communication approach.

In the centralized communication approach, a centralized fog node is located for each fog domain, which will control the communication among fog nodes in that particular fog domain. Each fog node in the domain F will transmit its average waiting time to the centralized fog node. The average waiting time (W) is the summation of the queuing delay of each fog node and the average processing delay for the requests in the queue. Once the centralized fog node in domain F receives the average waiting

time (W), it broadcasts the time W of each fog nodes to all the nodes in the domain F . Once the fog nodes receive W from the centralized fog node, it stores them in their resource allocation table, which can be updated by both fog nodes and the centralized fog node. The resource allocation table is used to choose the best fog node in a domain F for offloading the IoT requests.

Another way of communication among the fog nodes is decentralized communication. The fog nodes in a particular domain communicate with each other by transmitting its current information to their adjacent fog nodes. In decentralized communication, the centralized fog node will be unavailable; instead, they communicate among themselves employing a dedicated protocol. Here, each fog node in the domain maintains the resource allocation table by entering the average waiting time of the neighboring fog nodes. Like centralized communication, a fog node with least average waiting time and round-trip delay will be chosen as the best node to offload.

3.2 Offloading Decisions

In our FIoT architecture, based on some important factors, the offloading decisions are made. The factors include the response time of a fog node, computation cost for a single workload, processing capability of a particular fog node, and the current queuing status of the fog node. However, the value of these factors differs for individual workloads. Therefore, we categorize the workloads into high and low-workload.

FIoT, we consider only the high-workload and low-workload. An example of low-workload is the requests sent by temperature sensors to the processing technology to calculate the temperature of a closed space, whereas the requests sent by the video capturing sensor node are the example for high-workload. Let the average computation time of the low-workload be c_i at the fog node i , and C_j at the cloud data center j . Similarly, we assume the computation time of the high-workload as \bar{c}_i at the fog node i , and \bar{C}_j at the cloud data center j .

FIoT follows some crucial constraints to decide to offload the incoming IoT requests. Thus, either a fog node may process an incoming request from an IoT application, or it may be offloaded to an alternative fog node based on the offloading decision. The detailed procedure for the offloading decision is shown in Fig. 1. As shown in Fig. 1, once the fog node accepts a service request from IoT application, it estimates the average waiting time of the pending jobs in the queue of each fog node. A threshold value th_i for the fog node i is computed at each instance of incoming IoT traffic requests. After estimating the average waiting time, its relationship with the threshold value will be validated. If the average waiting time is less than the threshold th_i at the fog node i , the task will be assigned to the fog node i . Hence that particular IoT request will be handled by the fog node i , and simultaneously the average waiting time will be updated in the resource allocation table. Suppose if the

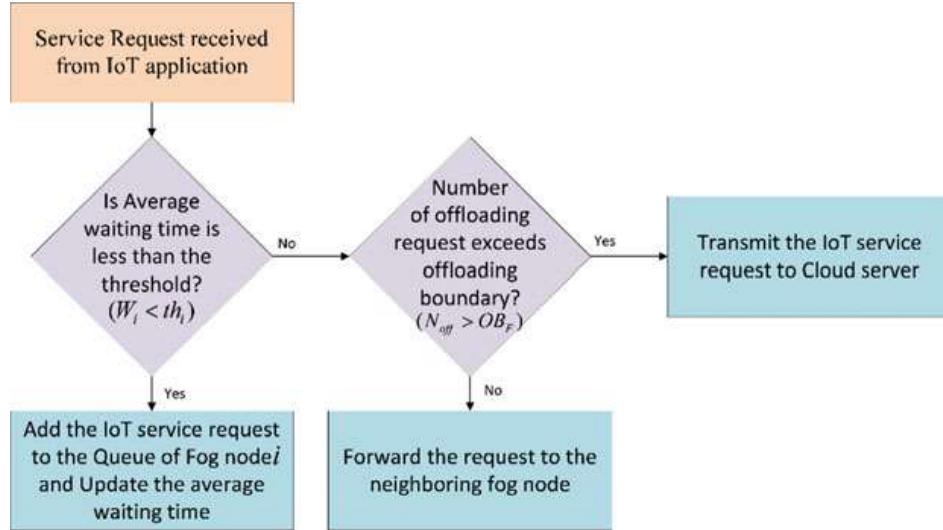


Fig. 1 Data flow of FIoT offloading strategy

average waiting time exceeds the threshold th_i , the fog node i will offload the request to the neighboring fog nodes or the cloud data center.

In addition to the dynamic threshold value computation, we estimate another threshold value called offloading boundary, which can be denoted by the term OB_F for the fog domain F . Therefore, when the average waiting time exceeds the threshold th_i , the FIoT checks whether the total number offloaded requests (N_{off}) exceeds the offloading boundary OB_F . If the resultant value is exceeded, the IoT request will be forwarded to the cloud; else, it will be transferred to the nearby fog node. Therefore, FIoT offloads the incoming request based on the available workload of the fog node.

3.3 Best Fog Node Selection

The average waiting time of each fog node gets updated frequently in the resource allocation table either in a centralized or decentralized mode of communication. Though the average waiting time is calculated for each IoT request, the round-trip delay is estimated after the fog node starts processing the requests. When an IoT request is offloaded from a fog node i to a best fog node j , the round-trip delay of the fog node i will be approximately half of the round-trip delay to reach the best fog node j . However, the IoT request has to wait until the average waiting time of the best fog node j overs. If the IoT request has not been handed over to any fog node queue, then the request should be offloaded again.

4 Analytical Model of FIoT Framework to Evaluate the Latency

In this section, we introduce an analytical model to formulate the low and heavy-workload IoT requests to the FIoT framework for calculating the latency in responding to those request. In our FIoT framework, the IoT service requests are transmitted to the fog nodes; if the fog nodes could not process the IoT requests, it will be transmitted to the cloud data centers. Thus, the latency in processing the IoT request depends on the probability on which the IoT request will be processed (p_l), the other processing delays in the framework. Thus, the latency L_l in processing the IoT request sent from the IoT device l is given in Eq. (1),

$$\begin{aligned} L_l = & (p_l^I \cdot \text{Avg}_l) + p_l^F \cdot (P_{li}^{IF} + T_{li}^{IF} + d_{li}) \\ & + p_l^C \cdot (P_{lm}^{IC} + T_{lm}^{IC} + \overline{\text{CD}}_m + P_{ml}^{CI} + T_{ml}^{CI}) \end{aligned} \quad (1)$$

where p_l^I is the probability of IoT request sent from the IoT device l is processed locally, p_l^F is the probability of processing the request at the fog node, and p_l^C is the probability of processing the request at the cloud data center. However, the sum of these probabilities would yield 1, such that, $p_l^F + p_l^I + p_l^C = 1$. The other processing delays in FIoT can be in three cases. The first one is when the IoT application request is processed locally, the second case is when it is processed in a fog node, and the third case is when it is processed in a cloud data center.

If the IoT device itself processed the request, then the average processing delay of the IoT device (Avg_l) is alone to be considered.

When the fog nodes handled the requests, the propagation delay when the service request is transferred to the fog node i from IoT device l , the total transmission delays (T_{li}^{IF}) when transmitting the request from IoT to fog node, and the total delay (d_{li}) that occurs in the processed response transmission from fog to IoT device are considered.

Similarly, when the cloud data centers handled the IoT requests, the processing delay (P_{lm}^{IC}) at cloud data center m , the total transmission delays (T_{lm}^{IC}) when transmitting the request from IoT device to cloud data center, the sum of queueing delay and the processing delay for computing the IoT request at the cloud data center ($\overline{\text{CD}}_m$), and the delays occur in transmitting the response from the cloud data center to the IoT device (P_{ml}^{CI} and T_{ml}^{CI}) are considered.

Our main objective is to minimize the latency in processing the IoT application requests. Hence by using our analytical model, we formulate the minimization of latency for the IoT devices available in the domain F . It can be given by Eq. (2),

$$\min \frac{1}{|N_{\text{IoT}}^F|} \sum_{l=1}^{N_{\text{IoT}}^F} L_l \quad (2)$$

where N_{IoT}^F is the total number of IoT devices available in the domain F , N_{Fog}^G denotes the total number of fog nodes in the domain G , and N_{Cloud}^S denotes the total number of cloud data centers in the domain S .

5 Results and Discussions

In this section, we present the evaluation results of our FIoT framework and the latency comparison with other mechanisms. We tabulate the simulation parameters in Table 1.

To show the performance variations both in the presence and absence of the fog nodes, we compare the performance of the FIoT framework with a fog-based framework called EBA [11] and a cloud-based framework called IoT-Azure [6]. The framework EBA has fog nodes, whereas the framework IoT-Azure does not have fog nodes; instead, it has cloud data centers for processing the IoT requests.

Finally, we compare the latency of the frameworks in processing the IoT requests irrespective of its type. Figure 2 shows the latency in the frameworks for processing and reverting. From the figure, it is clear that the FIoT framework has the lowest latency. This is due to the fog nodes and its offloading strategy. Meanwhile, the IoT-Azure framework possesses very high latency because it does not have fog nodes as well as offloading strategy. Thus, the FIoT framework is 82.3% better than the IoT-Azure framework and 54.8% better than the EBA framework.

Table 1 Simulation parameters for evaluating the latency of FIoT framework

Simulation parameter	Value
Number of IoT devices	600
Number of fog nodes	28
Number of cloud data centers	8
Average node degree of the fog node	3
Communication technology for low-workload requests	IEEE 802.15.4, NB-IoT
Link bandwidth for low-workload requests	250 kbps
Communication technology for high-workload requests	IEEE 802.11 a/g
Link bandwidth for high-workload requests	54 Mbps
Link bandwidth for fog nodes	100 Mbps
Link bandwidth between fog nodes and cloud data center	10 Gbps

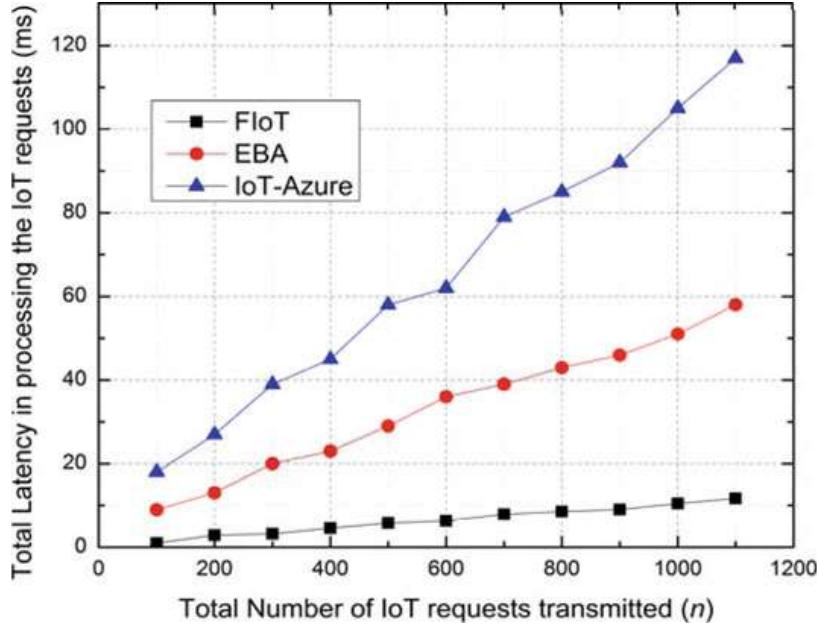


Fig. 2 Comparison of total latency observed in the IoT frameworks

6 Conclusion

In this paper, we introduce a framework called the FIoT framework, which includes a fog layer in addition to the other traditional IoT frameworks. Thus, the structure of our framework is a three-tier framework of IoT-Fog-Cloud. The main purpose of introducing the fog layer in the IoT framework is to minimize the delays in processing the IoT requests. Further, we propose an offloading strategy, which offloads both the low-workload and high-workload requests in an optimal way. In contrast to the traditional frameworks, the fog nodes in our FIoT framework have the capability of communicating with each other and with the cloud to offload the requests. Thus, our proposed FIoT framework minimizes the latency in processing the IoT requests. We further developed an analytical model to formulate the offloading strategy and to mathematically illustrate the latency. The simulation results also prove the low-latent performance of our FIoT framework. In the future, the other metrics, such as the arrival rate of the IoT requests, can also be considered to make the analytical model of the offloading strategy more optimal.

References

1. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M.: Internet of things: a survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutor.* **17**(4), 2347–2376 (2015)
2. Darwish, A., Hassanien, A.E., Elhoseny, M., Sangaiah, A.K., Muhammad, K.: The impact of the hybrid platform of internet of things and cloud computing on healthcare systems: opportunities, challenges, and open problems. *J. Ambient Intell. Humaniz. Comput.* **10**(10), 4151–4166 (2016)
3. Mor, N., Zhang, B., Kolb, J., Chan, D.S., Goyal, N., Sun, N., Lutz, K., Allman, E., Wawrzynek, J., Lee, E.A. Kubiatowicz, J.: Toward a global data infrastructure. *IEEE Internet Comput.* **20**(3), 54–62 (2016)
4. Cisco.: Fog computing and the internet of things: extend the cloud to where the things are. In: *Cisco, Technical Report* (2015)
5. Ali, M., Riaz, N., Ashraf, M.I., Qaisar, S., Naeem, M.: Joint cloudlet selection and latency minimization in fog networks. *IEEE Trans. Industr. Inf.* **14**(9), 4055–4063 (2018)
6. Liu, Y., Akram Hassan, K., Karlsson, M., Pang, Z., Gong, S.: A data-centric internet of things framework based on Azure cloud. *IEEE Access* **7**, 53839–53858 (2019)
7. Delsing, J., Eliasson, J., van Deventer, J., Derhamy, H., Varga, P.: Enabling IoT automation using local clouds. In: 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), pp. 502–507. Reston, VA (2016)
8. Yang, N., Zhang, H., Long, K., Jiang, C., Yang, Y.: Spectrum management scheme in fog IoT networks. *IEEE Commun. Mag.* **56**(10), 101–107 (2018)
9. Farahani, B., Firouzi, F., Chang, V., Badaroglu, M., Constant, N., Mankodiya, K.: Towards fog-driven IoT eHealth: promises and challenges of IoT in medicine and healthcare. *Futur. Gener. Comput. Syst.* **78**(2), 659–676 (2018)
10. Donassolo, B., Fajjari, I., Legrand, A., Mertikopoulos, P.: Demo: fog based framework for IoT service orchestration. In: 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), pp. 1–2. Las Vegas, NV, USA (2019)
11. Abkenar, F.S., Jamalipour, A.: EBA: energy balancing algorithm for fog-IoT networks. *IEEE Internet Things J.* **6**(4), 6843–6849 (2019)
12. Mutlag, A.A., Ghani, M.K., Arunkumar, N.A., Mohammed, M.A., Mohd, O.: Enabling technologies for fog computing in healthcare IoT systems. *Futur. Gener. Comput. Syst.* **90**, 62–78 (2019)
13. Mahmud, R., Kotagiri, R., Buyya, R.: Fog computing: a taxonomy, survey and future directions. In: *Internet of Everything*, pp. 103–130. Springer, Singapore (2018)

A Study on Implementation of Text Analytics over Legal Domain



Dipanjan Saha, Riya Sil, and Abhishek Roy

Abstract With the evolution of time, problem, and expectation of human beings, advancement of science and technology has facilitated scientific analysis of bulk dataset to generate desired output. This approach of bulk data analysis may be specifically implemented using Machine Learning and Data Analytics, which are the sub-domains of Artificial Intelligence (AI). The application of this cutting-edge technology can improve the efficiency of multivariate service sectors having societal significance (like legal system, education system, public transportation, rural healthcare management, etc), which directly or indirectly affect the well-being and productivity of an individual and the society as a whole. For example, India being a developing nation suffers due to insufficient number of judges, advocates, infrastructure, etc, and for which people have to wait long time to cherish their desired justice. In this paper, authors have proposed Machine Learning and Text Analytics-based legal support system to assist judges and advocates for faster delivery of justice to citizen.

Keywords Machine Learning · Text Analytics · Sentence vector

1 Introduction

Bulk amount of information [20], words [7], text [15, 16], and scripts [21] can be studied easily using Data Analytics [2, 3, 9, 10, 14, 22] to find correlations and patterns over the dataset available in digital market. In this technology-based digital market, business houses are using softwares to improve efficiency of their

D. Saha · R. Sil (✉) · A. Roy

Department of Computer Science & Engineering, Adamas University, Kolkata 700126, India

e-mail: riyasil1802@gmail.com

D. Saha

e-mail: sahadipanjan6@gmail.com

A. Roy

e-mail: dr.aroy@yahoo.com

products and reduce expenditure so as to increase their profit percentage. If this advanced technology can be used over multivariate service sectors having societal significance, it can generate an efficient mechanism for prompt delivery of services, which cannot be achieved using its conventional counterpart. As a result, society will have overall development in various service sectors like judiciary, education, public transportation, agro-based micro industry, etc, which directly or indirectly affects the well-being of populace (i.e., end user). Currently, there are wide scope available for this type of technological applications. For example, in case of *legal system*, due to lack of sufficient number of judges [5, 6], advocates, and infrastructure, people have to wait for long time to get their desired justice. As *legal system* have to handle huge amount of legal dataset generated from the proceedings, judgements, precedence, legal briefs, etc, application of Machine Learning [1, 8] and Data Analytics can help judges and advocates to analyze the situation correctly and take prompt decision to deliver justice to its end user. An advocate can use Data Analytics-based software related to a specific type of particular case (i.e., criminal, civil, etc), which will reduce manual work load to a significant level. Even judge can also use it for critical analysis of parameters and actions of accused and convicted person pertaining to any specific case, which will assist and expedite delivery of judgement. Considering the gravity of situation, in this paper, authors have focused in application of Data Analytics over *legal system* to propose text analytics [4, 12, 13, 17]-based legal support system for faster delivery of justice to its end user.

Section 2 discuss the fundamental concepts of Text Analytics (TA) using Machine Learning and its significance in Legal Domain. Section 3 explains our proposed Data Analytics-based legal system through phase by phase manner, whose conclusion is drawn in Sect. 4 of this paper.

2 Basics of Text Analytics

2.1 Machine Learning

Machine Learning (ML) is known to be the sub-field of Artificial Intelligence (AI). The main purpose of Machine Learning is to build models which will help people to develop their customized operational models depending on their input parameters. Several type of problems can be addressed using this concept, which can be broadly categorized as *Classification Problems* and *Regression Problems*. In Machine Learning, the models *train* themselves in order to give the appropriate output to its best possible level. Depending upon the ways and techniques of learning, it may be classified as *Supervised Learning*, *Unsupervised Learning*, *Semi-supervised Learning*, and *Reinforcement Learning*.

In Supervised Learning [18, 19], the referred system is being provided with sample inputs along with their labels as well as with their outputs. The main motive of this

type of learning algorithms is to *learn* or *train* by comparing the actual outputs with the *taught* outputs to find out the errors for further enhancement of model.

In Unsupervised Learning [11], the dataset is not labelled. Hence, the system is expected to find commonalities between its input data. We may correlate Unsupervised Learning to find out hidden patterns of input data, but at same time, it also has a goal of learning the features on its own.

Semi-supervised learning is the type of Machine Learning, which uses unlabelled data for the purpose of training. Scientists have found out that when both labelled and unlabelled data are used simultaneously, then it leads to considerable amount of improvement in accuracy of the algorithm.

Reinforcement Learning is an area of Machine Learning, where a suitable action is taken in order to maximize the reward obtained in a particular situation. It differs from Supervised Learning in a way that, in case of Supervised Learning, whole learning or training process is dependent on existing labelled dataset. However, in case of Reinforcement Learning, no such labelled dataset is available for the purpose of training. As a result, system has to decide its execution in absence of dataset, which generates output from the previous experiences.

2.2 *Text Analytics*

Text Analytics, a sub-domain of Natural Language Processing (NLP) is an area which deals with texts and their analysis. Researchers belonging to the computer science community uses it to extract useful and important information from raw texts. High-quality information can be extracted to recognize common patterns among various texts and studying the same. It may be also used for structuring input texts, deriving textual patterns within the structures and then finally resulting in evaluation of the output through proper interpretation. There are several applications of *Text Analytics*, among which *Text Classification* through manual approach is applied in our proposed legal support system.

The goal of Text Classification system is to help user to discover important information and make available or actionable to support decision-making strategy. The Text Classification system requires following features:

1. It requires text documents to work upon.
2. It contains a tree that describes topics of an organization which claims to be the most relevant.
3. Sample documents are presented to the model that can identify the type of contents which is to be assigned to each category.

There are *three* ways of text classification, namely *Manual Approach*, *Statistical Approach*, and *Rules-based Approach*.

Manual Approach proceeds by forming *Bag-Of-Words* model, where all words from all input documents are combined through sentence tokenization. After tokeniz-

ing, all the words from each of those documents are collected and kept in a separate document which is also known as *Mega-Document* or *Bag-Of-Words*.

The Statistical Approach is based on the manual identification or tagging of training dataset that belongs to same topic. This procedure uses a text classification classifier, namely Bayesian, LSA, etc. It looks at those terms whose frequency are considered to be the key element of document.

In *Rule-based* approach, the text are classified into organized group by applying a set of linguistic rules that indicates the system to recognize related categories with the utilization of semantically related elements of a text based on its content. The rules fall under pattern and prediction category.

Although there were various approaches for text classification, we have chosen the manual approach, i.e., the *Bag-Of-Words* model. This is because we did not have any ready-made softcopies of the legal documents with us. We had only hardcopies of them, from which we have manually converted them into softcopies. So, in this paper, we have used the manual approach, but we will not limit ourselves with this approach only. Working in both the approaches, i.e., the *Statistical Approach* and *Rules-based Approach*, may be considered as the *future scope* of our work.

2.3 Applications of Text Analytics

Text Analytics software provides tools, servers, data mining, and extraction tools for converting unstructured data into a structured data, which will serve as meaningful dataset for analytical purpose. The output of this algorithm evolves in the form of extracted entities, forms, and relationships. There are wide window open for application of Text Analytics, like *Sentiment Analysis*, *Summarization*, *Text Mining*, *Text Categorization*, *Text Clustering*, *Document Retrieval*, etc. We have applied this approach in our proposed legal support system, which is discussed further in Sect. 3.

3 Proposed Model

In this section, Fig. 1 shows our proposed Text Analytics-based legal support system, which are described through phase by phase approach.

3.1 Phase 1

In this section, Fig. 2 shows collection of *Legal Data Facts* (i.e., prosecution witness) from various trial courts and open-source data repository in hard copy version.

The *Output* of Phase 1 generates soft copy version of legal dataset (obtained through scanning of documents), which acts as input for next phase.

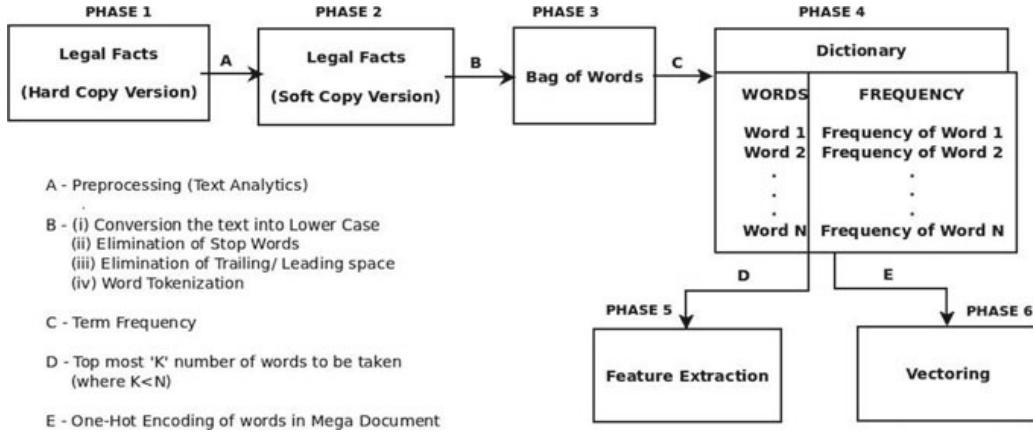
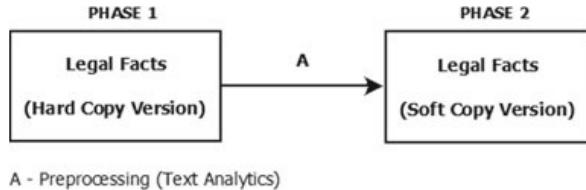


Fig. 1 Block diagram of our proposed Text Analytics-based legal support system

Fig. 2 Block diagram of transition from Phase 1 to Phase 2



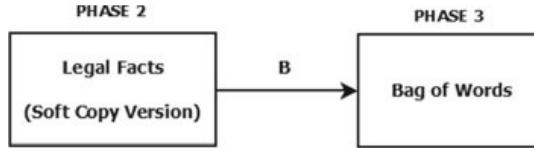
3.2 Phase 2

In this section, Fig. 3 focuses on *Data Cleansing* process of text files obtained through output of Phase 1. Data Cleansing or Data Cleaning is the process of removal of corrupted or inaccurate data from multiple sources such as record set, table, or database. It also refers to incomplete and inaccurate parts of data that are irrelevant for text. Figure 4 shows the program snippet written in Python programming language for cleansing of data, which are described below:

- Step 1: All sentences in input document are converted into lowercase characters.
- Step 2: All the *Stopwords* are removed from input documents. In computing, *Stopwords* are defined as English words which are generally filtered out from the input dataset before processing of natural language text (data).
- Step 3: At this stage, Word Tokenization is carried out. All the input sentences were splitted into words followed by removal of leading and trailing spaces.
- Step 4: Finally, *Bag-Of-Words* model is created after necessary cleansing process. The tokenized words obtained previously, are saved for further operation.

3.3 Phase 3

Bag-Of-Words generated as output of Phase 2, consists of all the tokenized words of sentences of input documents irrespective of their position in the sentences. As



B - (i) Conversion the text into Lower Case
(ii) Elimination of Stop Words
(iii) Elimination of Trailing/ Leading space
(iv) Word Tokenization

Fig. 3 Block diagram of transition from Phase 2 to Phase 3

```
#forming the wordlist
for sen in sentenceList:
    arr1 = sen.split(" ")
    for j in arr1:
        if not j.startswith("\\") and j!="":
            if j.lower() in words.words() and j.lower() not in stop_words and len(j.strip(" "))!=1:
                wordList.append(str(j))
```

Fig. 4 Program snippet for transition from Phase 2 to Phase 3

Fig. 5 Block diagram of transition from Phase 3 to Phase 4

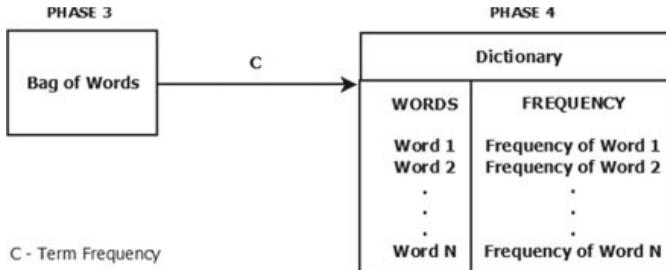


Fig. 6 Program snippet for transition from Phase 3 to Phase 4

```
wordfreq = {}
for token in corpus_new:
    if token not in wordfreq.keys():
        wordfreq[token] = 1
    else:
        wordfreq[token] = wordfreq[token]+1
```

shown in Fig. 5, we have calculated *Term Frequency* of each term or word present within Bag-Of-Words. *Term Frequency* may be defined as the count of each distinct or unique word from the Bag-Of-Word. Figure 6 shows our program snippet written in Python programming language for generation of this *Term Frequency*.

After getting *Term Frequencies* of all distinct words of Bag-Of-Word, we have built one dictionary containing *key-value* pairs. All distinct words of bag form the keys of that dictionary. And the values comprise of all the frequencies of those words. The dictionary which is thus formed is also known as the *Mega-Document*.

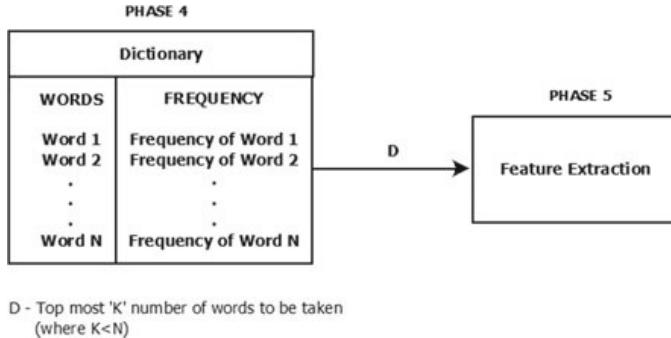


Fig. 7 Block diagram of transition from Phase 4 to Phase 5

```
#finding the most frequently occurred words from the corpus
most_freq = heapq.nlargest(100, wordfreq, key=wordfreq.get)
```

Fig. 8 Program Snippet for finding 100 most frequent words

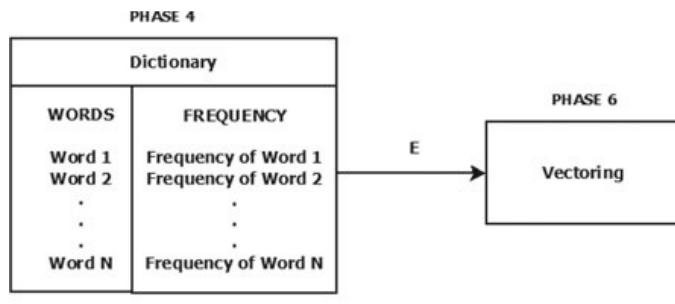
3.4 Phase 4

As shown in Fig. 7, the main objective of this phase is to extract topmost words from dictionary, which is also known as *Mega-Document*. User can use this Mega-Document for analysis of texts. In our proposed system, we have taken value from user and sorted dictionary in descending order based on the elements of value field and extracted topmost k words from the dictionary. Figure 8 focuses over our implementation approach to achieve this objective.

3.5 Phase 5

Text Vectorization done in our proposed system is shown through Fig. 9. It is the process of representing a particular English sentence with their corresponding vectors consisting of only zeros (0s) and ones (1s).

Fig. 9 Block diagram of generating sentence vectors from dictionary



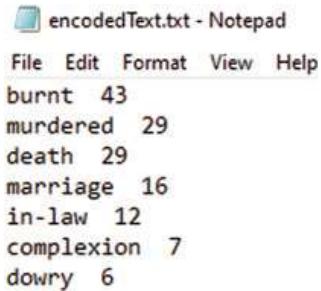
E - One-Hot Encoding of words in Mega Document

Fig. 10 Program snippet to form sentence vector

```
#Finding the SENTENCE VECTORS of each of the sentences in the corpus
sentence_vectors = []
for i in range(len(totalList)):
    arr = totalList[i].split(".")
    for sentences in arr:
        sentence_tokens = nltk.word_tokenize(sentences)
        sent_vec = []
        for token in most_freq:
            if token in sentence_tokens:
                sent_vec.append(1)
            else:
                sent_vec.append(0)
        sentence_vectors.append(sent_vec)

sentence_vectors = np.asarray(sentence_vectors)
```

Fig. 11 Snapshot of text file containing the most frequent words



The screenshot shows a Notepad window with the title 'encodedText.txt - Notepad'. The menu bar includes File, Edit, Format, View, and Help. The content of the window is a list of words and their corresponding counts:

Word	Count
burnt	43
murdered	29
death	29
marriage	16
in-law	12
complexion	7
dowry	6

In case of value zero (0), it denotes absence of that particular word in the *Mega-Document*. In case of value one (1), it denotes the presence of particular work in the English sentence of input document and Mega-Dictionary. Figure 10 shows our program snippet written in Python programming language to achieve this objective.

3.6 Sample Outputs

The sample outputs obtained through execution of our program (written in Python language) are mentioned below:

Figure 11 shows the text file generated as the output of our program, which contains the most frequent words from our sample inputs.

Figure 12 shows the list of most frequent words generated as output of our program.

Figure 13 shows the sentence vector generated as output of our program.

4 Conclusion

In this work, we have tried to create a *Bag-Of-Words* model belonging to several text documents under Legal Domain. After creating the bag, we have also found out k number of frequent words from the *Bag-Of-Words*. In addition to this, we have also transformed all the sentences of documents into their respective vectors, which is

```
C:\Users\Lenovo\Documents\Python Programs>python textAnalytics.py
Most frequent words are:-
['murdered', 'husband', 'burnt', 'murdered', 'complexion', 'dowry', 'kill', 'husband', 'in-law', 'crying', 'amount', 'residing', 'torture', 'dowry', 'treatment', 'murdered', 'injured', 'death', 'murdered', 'marriage', 'matrimonial', 'murdered', 'residing', 'burnt', 'treatment', 'complexion', 'paternal', 'death', 'dowry', 'paternal', 'dowry', 'husband', 'death', 'in-law', 'matrimonial', 'died', 'hospital', 'accused']
```

Fig. 12 List of most frequent words

```
C:\Users\Lenovo\Documents\Python Programs>python posTags.py
Sentence Vectors:
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 1 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

Fig. 13 Sample of generated sentence vectors

known as sentence vectorization. However, there are certain technical constraints of Bag-Of-Words model, like the vocabulary should be carefully designed to manage its size. Even in terms of space and time complexity, the sparse representation of mega-document is comparatively tough to design than the dense documents. We also know that context and meaning of a word adds a lot of vital information to the document. Hence, proper modelling is essential for these words to represent the difference between those words and their synonyms. In this paper, we are yet to achieve proximity analysis for all the sentences from a reference point. As a future scope of work, we are also yet to predict the conviction of an accused person with the application of our proposed model.

References

1. Bredeche, N., Shi, Z., Zucker, J.-D.: Perceptual learning and abstraction in machine learning: an application to autonomous robotics. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **36**(2), 172–181 (2006). <https://doi.org/10.1109/tsmcc.2006.871139>
2. Feng, M., Zheng, J., Ren, J., Hussain, A., Li, X., Xi, Y., Liu, Q.: Big data analytics and mining for effective visualization and trends forecasting of crime data. *IEEE Access* **7**, 1–9 (2019). <https://doi.org/10.1109/access.2019.2930410>

3. Flesch, B., Vatrapu, R., Mukkamala, R.R., Hussain, A.: Social set visualizer: a set theoretical approach to big social data analytics of real-world events. In: 2015 IEEE International Conference on Big Data (Big Data), pp. 2418–2427. IEEE, USA (2015). <https://doi.org/10.1109/bigdata.2015.7364036>
4. Gordon, T.F., Walton, D.: Legal reasoning with argumentation schemes. In: 12th International Conference on Artificial Intelligence and Law—ICAIL 09, pp. 137–146. ACM, Spain (2009). <https://doi.org/10.1145/1568234.1568250>
5. India has 19 judges per 10 lakh people: data. <https://www.thehindubusinessline.com/news/india-has-19-judges-per-10-lakh-people-data/article25030009.ece>. Accessed 31 Oct 2019
6. India has only 19 judges per 10 lakh people: Law Ministry. <https://www.jagranjosh.com/current-affairs/india-has-only-19-judges-per-10-lakh-people-law-ministry-1537854436-1>. Accessed 31 Oct 2019
7. Kar, R., Saha, S., Bera, S.K., Kavallieratou, E., Bhateja, V., Sarkar, R.: Novel approaches towards slope and slant correction for tri-script handwritten word images. Imaging Sci. J. **67**(3), 159–170 (2019). <https://doi.org/10.1080/13682199.2019.1574368>
8. Kohestani, A., Abdar, M., Khosravi, A., Nahavandi, S., Kohestani, M.: Integration of ensemble and evolutionary machine learning algorithms for monitoring diver behavior using physiological signals. IEEE Access **7**, 98971–98992 (2019). <https://doi.org/10.1109/access.2019.2926444>
9. Kumar, S., Singh, M.: Big data analytics for healthcare industry: impact, applications, and tools. Big Data Min. Anal. **2**(1), 48–57 (2019). <https://doi.org/10.26599/bdma.2018.9020031>
10. Li, D., Wang, Y., Wu, S., Qi, J., Wang, T.: An visual analytics approach to explore criminal patterns based on multidimensional data. In: 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 5563–5566. IEEE, USA (2017). <https://doi.org/10.1109/igarss.2017.8128264>
11. Li, Q., Zhao, J., Zhu, X.: An unsupervised learning algorithm for intelligent image analysis. In: 2006 9th International Conference on Control, Automation, Robotics and Vision, pp. 1–5. IEEE, Singapore (2006). <https://doi.org/10.1109/icarv.2006.345232>
12. Liu, S., Yin, J., Wang, X., Cui, W., Cao, K., Pei, J.: Online visual analytics of text streams. IEEE Trans. Vis. Comput. Graph. **22**(11), 2451–2466 (2016). <https://doi.org/10.1109/tvcg.2015.2509990>
13. Luo, D., Yang, J., Krstajic, M., Ribarsky, W., Keim, D.A.: EventRiver: visually exploring text collections with temporal references. IEEE Trans. Vis. Comput. Graph. **18**(1), 93–105 (2012). <https://doi.org/10.1109/tvcg.2010.225>
14. Malik, A., Maciejewski, R., Towers, S., McCullough, S., Ebert, D.S.: Proactive spatiotemporal resource allocation and predictive visual analytics for community policing and law enforcement. IEEE Trans. Vis. Comput. Graph. **20**(12), 1863–1872 (2014). <https://doi.org/10.1109/tvcg.2014.2346926>
15. Murty, M.R., Murthy, J.V.R., Prasad Reddy, P.V.G.D., Satapathy, S.C.: A survey of cross-domain text categorization techniques. In: 2012 1st International Conference on Recent Advances in Information Technology (RAIT), pp. 499–504. IEEE, India (2012). <https://doi.org/10.1109/RAIT.2012.6194629>
16. Murty, M.R., Murthy, J.V.R., Prasad Reddy, P.V.G.D.: Dimensionality reduction text data clustering with prediction of optimal number of clusters. Int. J. Appl. Res. Inf. Technol. (IJARITAC) **2**(2), 41–49 (2011). <https://doi.org/10.5958/j.0975-8070.2.2.010>
17. Park, D., Kim, S., Lee, J., Choo, J., Diakopoulos, N., Elmquist, N.: ConceptVector: text visual analytics via interactive lexicon building using word embedding. IEEE Trans. Visual Comput. Graph. **24**(1), 361–370 (2018). <https://doi.org/10.1109/tvcg.2017.2744478>
18. Rissland, E.L.: Artificial intelligence and law: stepping stones to a model of legal reasoning. Yale Law J. **99**(8), 1957–1981 (1990)
19. Saravanan, R., Sujatha, P.: A state of art techniques on machine learning algorithms: a perspective of supervised learning approaches in data classification. In: 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 945–949. IEEE, India (2018). <https://doi.org/10.1109/iccons.2018.8663155>

20. Satapathy, S.C., Tavares, J.M.R.S., Bhateja, V., Mohanty, J.R. (eds.): Information and decision sciences. In: Proceedings of the 6th International Conference on FICTA (2017). ISBN: 978-981-10-7562-9. <https://doi.org/10.1007/978-981-10-7563-6>
21. Sing, P.K., Sarkar, R., Bhateja, V., Nasipuri, M.: A comprehensive handwritten Indic script recognition system: a tree-based approach. *J. Ambient Intell. Hum. Comput.* 1–18 (2018). ISSN: 1868-5137. <https://doi.org/10.1007/s12652-018-1052-4>
22. Turkay, C., Kaya, E., Balcisoy, S., Hauser, H.: Designing progressive and interactive analytics processes for high-dimensional data analysis. *IEEE Trans. Vis. Comput. Graph.* **23**(1), 131–140 (2016). <https://doi.org/10.1109/tvcg.2016.2598470>

Advanced Key Management System (AKMS) for Security in Public Clouds



J. L. Amarnath, Pritam G. Shah, and H. Chandramouli

Abstract Cloud computing has become an emerging model of information technology industry as it can be accessed anywhere in the world on a pay-per-use basis. However, one major problem it is facing in today's challenging world is the security issues. The various existing key generation and management systems are suffering with the security issues. Along with the security feature, the data sharing is also more important issue to be studied. Considering these two parameters, it is necessary to work on the secure data sharing in public clouds. For that purpose, we have proposed an advanced key management system (AKMS) which can perform identity-based encryption. Also, the proposed AKMS type encryption provides high security, scalability, and flexibility in the public clouds.

Keywords Cloud computing · Security · Encryption · Public clouds and key management system

1 Introduction

Key generation and management of key distribution play an important role in cloud data storage system. In this type of systems, basically three parties are involved during the data retrieval and data storage. First is user, second one is third-party auditor (TPA), and third one is cloud server. The storage of data over cloud provides user authentication mode in three different segments. One from user to TPA and

J. L. Amarnath (✉)
VTU University, Machhe Belagavi, Bengaluru, India
e-mail: amar.rv2010@gmail.com

P. G. Shah
University of Canberra, Canberra, Australia
e-mail: wsnpgs@gmail.com

H. Chandramouli
CSE, East Point College of Engineering and Technology, Bangalore, India
e-mail: hcmcool123@gmail.com

second one is TPA to server and third one is cloud server to user. In overall scenario, key distribution plays an important role. Cloud data storage and access of data faced a big security issue in concern of security and validation of user authentication. For the authentication of user used various cloud security model. All cloud security model used cryptography technique for the generation of key for access of data and retrieval of data. The generation of key handled by server usually take more time for the generation of key and more time for accessing of data. Now key generation and handling of key in cloud computing is big issue. Nowadays, various authors used various key distribution and key authentication techniques such as AES, DSS, RSA, ECA, and many more technique for the grouping of key [1]. The process of key generation takes more time and invites man-in-the-middle attack on the time of retrieval of data. Now the minimization of time in key generation policy will be a big issue for the system. Among the different access control mechanisms of the cloud systems, the controlling of read and write access operation is the most important task. It is the task related to the data confidentiality and the data integrity. Another important task is to ensure the stored data available for always, and the server is solely responsible to ensure this task. To address the above task, it is necessary to provide the control mechanism with the three levels of access permission. It can be achieved using the cryptography methods. Cryptography methods can perform these operations locally on the client so that it can increase the security levels. Data confidentiality and integrity can be achieved using the symmetric and asymmetric encryption methods, respectively. Here, the owner has created the keys so he can have all three access permissions. In the distribution process, the owner can grant different access permissions to other users. This kind of mechanism is well suited for the construct of secure access control system (SACS).

Generally, the cloud storage data frequently undergo for updating by its users, including insertion, deletion, modification, etc. [2, 3]. Also, ensuring the storage correctness under dynamic data update is very important. The deployment of cloud computing (CC) is powered by data centers running in a simultaneous, cooperated, and distributed manner. Individual user data is redundantly stored in multiple physical locations to reduce the data integrity threats. Storage correctness is assured using the distributed protocols. Hence, it is very important to achieve robust and secure cloud storage.

A cloud user stores the data through the cloud service provider in the cloud servers. These servers are working simultaneously, in a cooperated and distributed model. Data redundancy will be used for the error correcting methods in case of any server crash. The users interact with the server through the cloud service provider to access their data. To accomplish this task, the users perform block level operation on the data like block updating, deletion, insertion, etc. [4].

The rest of the paper as in Sect. 2 discusses about the literature review. Section 3 describes the advanced key management system. In Sect. 4, we discuss the experimental results, and finally, conclusion and future work are described in Sect. 5.

2 Literature Review

This section discusses the security of cloud storage process over the Internet. Various authors used various cryptography techniques and some techniques are discussed here. Yu et al. [5] have accomplished this objective by consolidating the methods of characteristic-based encryption, intermediary and sluggish re-encryptions. They have talked about the plan which has remarkable properties of clients to have the advantage of confidentiality and their secret key responsibility. Broadly they have demonstrated that the mechanism is exceptionally efficient and prove that it is the most secure mechanism compared to the existing security models. This paper goes for fine-grained information get to control in distributed computing. One test in this setting is to accomplish fine-grandness, information confidentiality, and adaptability all the while, which is not given by current work. They talked about a plan to accomplish this objective by misusing key-policy attribute-based encryption (KP-ABE) and particularly joining it with systems of intermediary and sluggish re-encryption.

Chow et al. [6] have expressed the worry in utilizing distributed storage that the touchy information ought to be confidential to the outside servers. They concentrated on the different elements given by cryptographic unknown verification and encrypted instruments. Also instantiates their outline with verifier-neighborhood revocable gathering mark and personality-based communicate encryption with steady size figure writings and private keys. The idea is to outfit the communicate encryption with the dynamic figure content refresh highlights and gives formal security ensure against versatile picked figure content decoding and refresh assault.

Xia et al. [7] have proposed a protected, efficient, and dynamic inquiry plan that is an extraordinary watchword adjusted twofold tree as the file. Also they talked about the “Voracious Depth-first Search” calculations to acquire preferred efficiency over direct pursuit. Trial comes about to show the efficiency of their plan. The information proprietor is an in-charge for the creation of refreshing data and to send them onto the server. Also, he has to store the decoded file tree and data, as they are crucial in the recalculation of IDF values. Finally, these kind of dynamic proprietors are not well suited for the distributed systems. Really, there exist different secure difficulties in the multi-client plot. Firstly, every client normally keeps the same secure key for trapdoor era in a symmetric SE conspires. In these situations, the denial of client is an enormous test. In the events of expectation to renounce a client, they have to remake the record and circulate the updated keys to the approved clients. Furthermore, symmetric SE plots for the most part expect that every one of the information client is dependent. It is not functional and an exploitative information client will inform several security related threats.

Yang et al. [8], in this paper, first plan an evaluating structure for distributed storage frameworks and talked about an efficient and protection saving reviewing convention. They talked about an efficient and characteristically secure element examining convention. It secures the information protection against the evaluator by joining the cryptography strategy with the bi-linearity property of bilinear paring, instead of utilizing the veil method. Along these lines, their multi-cloud bunch reviewing

convention does not require any extra coordinator. Their group inspecting convention can likewise bolster the bunch evaluating for numerous proprietors. Besides, their examining plan acquires less correspondence cost and less calculation cost of the evaluator by moving the figuring heaps of inspecting from the inspector to the server, which extraordinarily enhances the reviewing execution and can be connected to huge scale distributed storage frameworks.

Wei et al. [9] in this paper, the itemized examination is offered to acquire an ideal testing size to limit the cost of the system. They have constructed a down to earth secure-mindful distributed system trial condition, or SecHDFS, as a proving ground to execute SecCloud. Advance test comes about and has shown the viability and efficiency of the SecCloud. They have defined the ideas of un-cheatable cloud calculation and protection deceiving disheartening and examined SecCloud to accomplish the security objectives. To enhance the efficiency, distinctive clients' solicitations can be simultaneously dealt with through the group verification. By the broad security examination, execution and recreation in their SecHDFS demonstrates that convention is successful, also efficient in accomplishing a safe distributed system.

Selvakumar et al. [10], in this paper, talk about the parceling technique for the information stockpiling which is to avoid the neighborhood duplicate at client by utilizing apportioning method. This strategy ensures high distributed storage uprightness, improved mistake limitation and a proof of getting out of hand server. They examined a productive information stockpiling security in cloud benefit. The dividing of information empowers putting away of the information in simple and viable way. It likewise gives route for adaptability and there is less cost in information stockpiling. The space and time are additionally successfully decreased amid capacity. The basic idea of dynamic operation involves encoding and translating operation to secure the information while putting away in the cloud. Additionally, the remote information verification identifies the dangers and making trouble server as and when putting away the information into cloud guaranteeing secured information.

Rajathi et al. [11] clock stockpiling administration maintains a strategic distance in terms of programming cost, staff support. Also it provides smooth in execution, low capacity cost and versatility. It has been studied on the different distributed storage system strategies and their points of interest, downsides. Also it discusses the difficulties in the execution of secure cloud information stockpiling. Distributed computing is a rising mechanism in the worldview, permits clients to share assets and data from a pool of conveyed processing as an administration over Internet. Apart from the benefits of the cloud, it is necessary to provide security and protection of the information stored in it. Distributed storage is a great deal more gainful and invaluable compared to the conventional mechanisms in terms of adaptability, cost, conveyability, and usefulness prerequisites. It has been discussed about the secure stockpiling strategies in the CC environment. Initially, the different stockpiling procedures for the information security have been talked about in detail and at last displayed a relative examination on capacity procedures that incorporates the talked about approach, preferences, and confinements of those capacity systems.

Chris et al. [12] consider the issue of efficiently describing the uprightness of information provided by the un-trusted servers. In the provable information ownership show, the client preprocesses the information and then sends to an un-confided in server. Next the client requests the server demonstrating that to put away the information has not been messed. They demonstrate that customers of an un-trusted CVS server even those putting away none of the formed assets locally can question the server to demonstrate ownership of the archive utilizing only a little division of the data transfer capacity required to download the whole vault. “Evidence size and time per submit” allude to a proof sent by the server to demonstrate that a solitary confer was performed effectively, speaking to the ordinary utilize case. These submit evidences are little and quick to process, rendering them reasonable despite the fact that they are required for each confer. Their tests demonstrate that their DPPD plan is efficient and reasonable for use in disseminated applications.

Cong et al. [13] have implemented an auditable cloud information system. They first present a system design for adequately portraying, creating, and assessing secure information stockpiling issues. They then propose an arrangement of deliberately and cryptographically attractive properties for open examining administrations of tried and true cloud information stockpiling security to end up distinctly a reality. Through top to bottom investigation, some current information stockpiling security building squares are inspected. The upsides and downsides of their handy ramifications with regard to distributed computing are abridged. Additionally difficult issues for open evaluating administrations that should be centered around are examined as well. They trust security in distributed computing, a region loaded with difficulties and of central significance is still in its outset now however will pull in tremendous measures of research exertion for a long time to come.

Kan et al. [14] have discussed another get to control structure for multi-specialist frameworks in distributed storage and talked about an efficient and secure multi-expert get to control plot. They have planned an efficient multi-specialist ciphertext policy-ABE conspire that does not require a worldwide expert and can bolster any LSSS get to structure. At that point, they demonstrated that their multi-specialist ciphertext policy-ABE plan is provably secure in the irregular prophet display. Also, it has been described about another strategy to tackle the trait denial issue in multi-specialist ciphertext policy-ABE frameworks. The examination and reproduction is used to demonstrate that the control plan is versatile and efficient.

Yang et al. [15], in this paper, researched the reviewing issue for information stockpiling in distributed computing and examined an arrangement of prerequisites of outlining the third-party auditing conventions. They likewise portrayed and broke down the current reviewing strategies in the writing. At long last, they talked about some difficult issues in the plan of proficient reviewing conventions for information stockpiling in distributed computing.

3 System Architecture

The proposed advanced key management system (AKMS) supports secure data sharing by an access control mechanism for all keys and metadata. It also manages the public, private keys and user accounts. Figure 1 shows the public cloud storage architecture. It has the following configurations.

- *Cloud configuration:* It is a text configuration file which provides the information of available dispersed public cloud storages and their details and also provides the threshold number of minimum dispersed data required to reconstruct the data.
- *Secure cloud configuration:* It is a java JKS configuration file which gives the details of tokens used by the clouds to call cloud storage's API. It has a passphrase-type protection system.

i. Advanced Key Management System

The two main parts of the advanced key management system (AKMS) are the cloud key management client (CKMC) and the cloud key management server (CKMS). Figure 2 shows the advanced key management system (AKMS). Client deals with the cloud applications. Also, it serves as the cloud service model, including software, infrastructure as a service (IaaS). Server uses cloud key management interoperability protocol (CKMIP) to interact with the clients. For this purpose, it uses the combination of symmetric key management system (SKMS) and public key infrastructure (PKI). It is achieved using the symmetric and asymmetric key management protocols. CKMIP protocol can be used by any cloud cryptographic client to address/resolve the critical need issues of any comprehensive key management protocol.

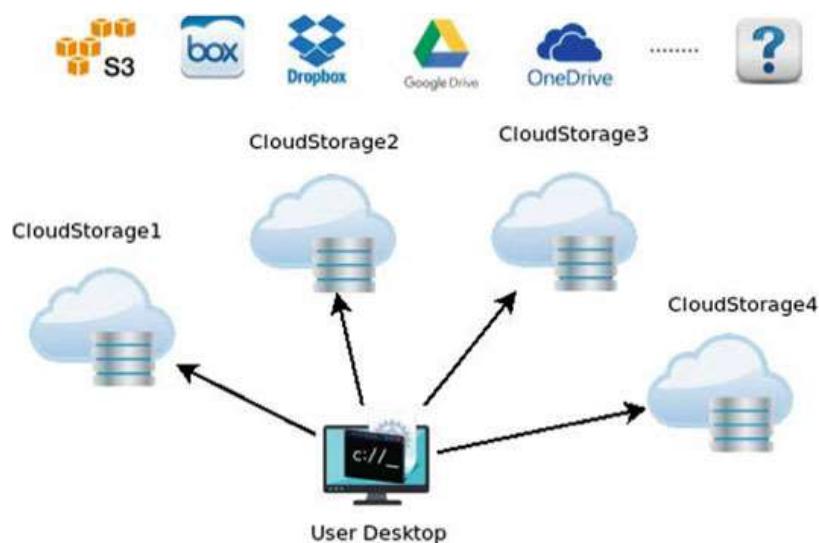


Fig. 1 Public cloud storage

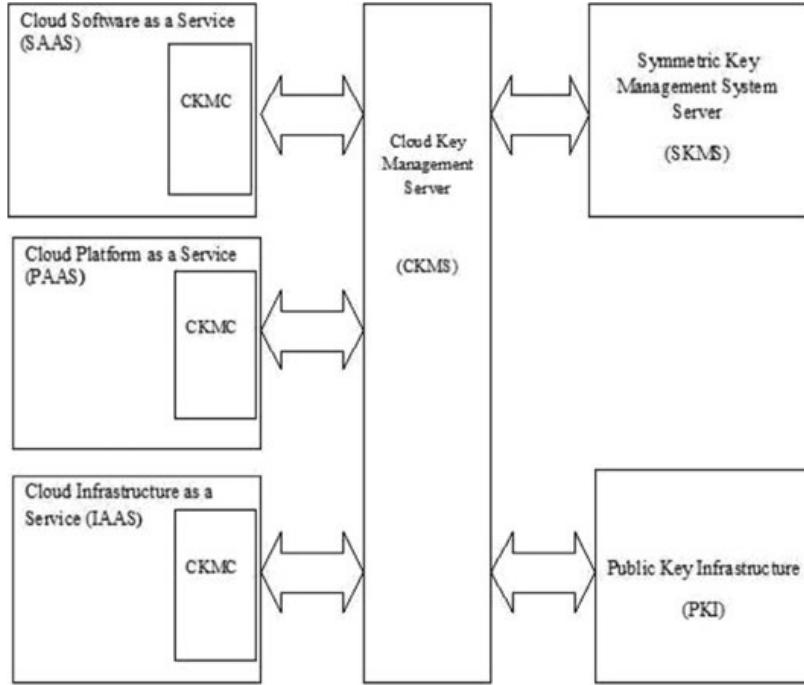


Fig. 2 Advanced key management system

It is an effective key management for the encryption, certificate-based device authentication and digital signatures. This system uses elliptic-curve Diffie–Hellman (ECDH) technique for the key exchange process.

(a) Elliptic curve Encryption/Decryption

Suppose user X wants to send a message Pm to user Y then he randomly selects a positive integer k and a private key d_X .

Then the public key of X is generated by $P_X = d_X G$ and the cipher text Cm is produced with consisting of pair of points.

$$Cm = (kG Pm) + kP_Y \quad (1)$$

Here G is the base point of an elliptic curve. Similarly, the public key of Y is given by $P_Y = d_Y G$ and the cipher text Cm which is sent from X . The decryption at the Y will be obtained using Eq. (2), where the original message (Pm) is decoded accurately.

$$Pm + kP_Y - d_Y(kG) = Pm + k(d_Y G) - d_Y(kG) = Pm \quad (2)$$

(b) Elliptic-curve Diffie–Hellman (ECDH) key exchange

In general, bulk amount of data encryption/decryption is achieved using the symmetric key cryptosystems as they can compute faster than other public key cryptosystems. A secret key between two users for each session can be generated using the ECDH key exchange method.

Suppose that users X and Y want to agree upon a secret key, which will be used for secret key cryptography. User X generates a private key d_X and a public key $P_X = d_X G$. Here, G is the generator of the elliptic curve. Then user X sends key P_X to Y . Similarly, user Y generates private key d_Y and a public key $P_Y = d_Y G$. Then user Y sends P_Y to X .

After receiving the X 's message, Y computes,

$$d_Y(P_X) = d_X d_Y G \quad (3)$$

After receiving the Y 's message, X computes,

$$d_X(P_Y) = d_X d_Y G \quad (4)$$

Now, both the users can use the key $d_X d_Y G$. This gives the point on the given elliptic curve as a common secret key. The user first logs into the cloud and authenticates himself. Then the authenticated user uses the ECDH technique for the key exchange process.

4 Results

The user interface of the implemented system is shown in Fig. 3. Every user is provided with sign-in option which is managed by the AKMS.

Fig. 3 User interface with sign-in option and browser

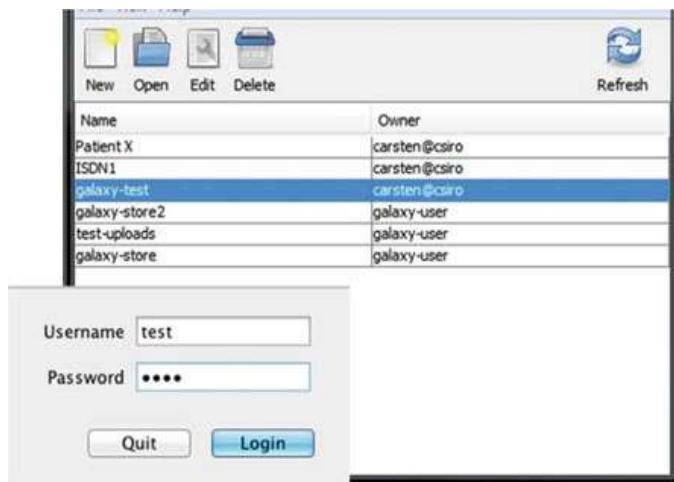
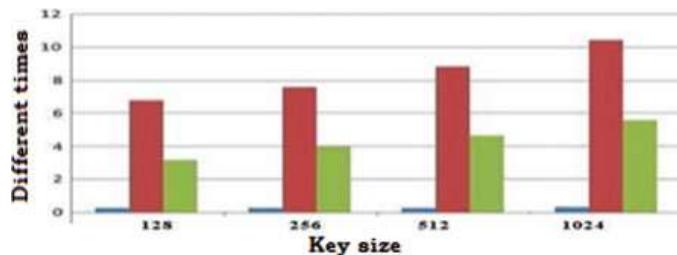


Table 1 Key size versus different times

Key size	Key generation time (ms)	Encryption time (ms)	Decryption time (ms)
128bits	0.250	6.764	3.162
256bits	0.253	7.572	3.975
512bits	0.266	8.829	4.612
1024bits	0.285	10.415	5.576

Fig. 4 Key size versus different times



The implemented advanced key management system generates the key within less time of 0.250 ms for a key size of 128 bits. The corresponding encryption and decryption times are also noted as shown in Table 1.

The experimentation has been carried out with the different key sizes like 128, 256, 512, 1024 bits. Figure 4 shows the pictorial representation of the different times (like key generation time, encryption time, and decryption time) with respect to the key size.

5 Conclusion

The proposed key management has considered various parameters for the effective key generation and management which are the most important parameters for providing the desired security in the public clouds. The highest security levels of any public/private clouds improve the trustworthiness of that cloud among the users. As the key size (in bits) increases, the different times like key generation, encryption and decryption times also increases. Also, increase in the key size leads to the complexity of key generation process. But in this system, the ECC and ECDH algorithms provide the same level of security as of other public key cryptosystems with less key size. This system improves the average access time for the cloud users. Also, this system improves the scalability and flexibility in public clouds. Finally, the AKMS provides the strong encryption with advanced key management, and it is an important mechanism for cloud computing system to protect data.

References

1. Sanket, S., Nitte, Sanjay, H.A., Deepika, K.M.; Rangavittala: An encryption, compression and key (ECK) management based data security framework for infrastructure as a service in cloud. In: 2015 IEEE International Advance Computing Conference (IACC) 12–13 June 2015, Bangalore, India, pp. 210–216. ISBN: 978-1-4799-8047-5
2. Subashini, S., Kavitha, V.: A survey on security issues in service delivery models of cloud computing. *J Netw Comput Appl* 01–011 (2011)
3. Yang, K., Jia, X., Ren, K.: Attribute-based fine-grained access control with efficient revocation in cloud storage systems. In: ACM, pp. 0523–528 (2013)
4. Li, M., Yu, S., Zheng, Y., Ren, K., Lou, K.: Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. IEEE, pp. 01–014 (2013)
5. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving secure, scalable, and fine-grained data access control in cloud computing. IEEE, pp. 01–09 (2010)
6. Chow, S.S.M., Chu, C.-K., Huang, X., Zhou, J., Deng, R.H.: Dynamic secure cloud storage with provenance. Springer, pp. 0442–0464 (2011)
7. Xia, Z., Wang, X., Sun, X., Wang, Q.: A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE, pp. 01–013 (2015)
8. Yang, K., Jia, X.: An efficient and secure dynamic auditing protocol for data storage in cloud computing. IEEE, pp. 01–011 (2012)
9. Wei, L., Zhu, H., Cao, Z., Dong, X., Jia, W., Chen, Y., Vasilakos, A.V: Security and privacy for storage and computation in cloud computing. *Inf. Sci.* 0371–0386 (2014)
10. Selvakumar, C., Rathanam, G.J., Sumalatha, M.R.: PDDS—Improving cloud data storage security using data partitioning technique. IEEE, pp. 07–011 (2012)
11. Rajathi, A., Saravanan, N.: A survey on secure storage in cloud computing. *Indian J. Sci. Technol.* 4396–4401 (2013)
12. Erway, C.C., Küçü, A., Papamanthou, C., Tamassia, R.: Dynamic provable data possession. ACM, pp. 0213–0222 (2010)
13. Wang, C., Ren, K.: Toward publicly auditable secure cloud data storage services. IEEE, 019–024 (2010)
14. Yang, K., Jia, X.: Attributed-based access control for multi-authority systems in cloud storage. IEEE. pp. 536–545 (2012)
15. Yang, K., Jia, X.: Data storage auditing service in cloud computing: challenges, methods and opportunities. *World Wide Web* 0409–0428 (2012)

Machine Learning and Feature Selection Based Ransomware Detection Using Hexacodes



Bheemidi Vikram Reddy, Gutha Jaya Krishna, Vadlamani Ravi,
and Dipankar Dasgupta

Abstract Ransomware attacks increased within the past few years resulting huge financial losses to various businesses across the globe. To overcome the ransomware attacks, executables (or binary files) are converted back to assembly-level language or source code for further examination. In this work, we propose a novel ransomware detection method based on just hexacodes and without opcodes, which is clear departure from earlier studies. We first extracted the hexadecimal codes from the ransomware and then employed machine learning (ML) techniques and a few feature selection methods. Here, we leverage the dump and parser to decode binaries for extracting hexacodes. Apart from ransomware, files and benign executables are also used for training the classifiers. We conclude that out of the several ML techniques and the feature selection methods employed, random forest together with information gain-based feature selection obtained the highest accuracy of 88.39% in tenfold cross-validation setup. We also performed a statistical significance test to corroborate our results statistically. One significant observation is that random forest with only 30 features from information gain gave an improvement of 1% in accuracy, over the best model with all features. This architecture can be utilized as an early detection system.

B. V. Reddy · G. J. Krishna · V. Ravi (✉)

Center of Excellence in Analytics, Institute for Development and Research in Banking Technology, Hyderabad 500057, India
e-mail: rav_padma@yahoo.com

B. V. Reddy
e-mail: vreddy27193@gmail.com

G. J. Krishna
e-mail: krishna.gutha@gmail.com

G. J. Krishna
School of Computer and Information Sciences,
University of Hyderabad, Hyderabad 500046, India

D. Dasgupta
Department of Computer Science, The University of Memphis, Memphis, TN, USA
e-mail: ddasgupt@memphis.edu

Keywords Hexadecimal codes · Classification · Machine learning · Malware · Multi-class problem · Ransomware

1 Introduction

Cyber attacks with ransomware are increasing in various fields, where the financial risk to banks and financial firms is inherent. Moreover, ransomware attacks can happen in a number of ways, for instance, by making a user to click a Web link that is certainly malicious in an email or by any other alternative methods. Ransomware may most likely use functions that are cryptographic to lock the machine or use a remote control channel and digital modes of money payment mechanisms [24].

Extensively, ransomware is grouped into two classes: Crypto ransomware and Locker ransomware. Crypto ransomware encrypts the records, making the documents cryptic to use. Removing the ransomware or utilizing a disk that is difficult to get affected which will probably not solve the issue. The affected individual is approached to pay, explicitly by bitcoin in order to decrypt the data. Bitcoin is generally used, because of its privacy, as it is hard to track the attacker. Paying the cryptocurrency will not ensure that the attacker provides the decryption key to recover the affected system. Some of the Crypto ransomware are: SamSam [19], CryptoLocker [12], CryptoWall [18], Locky [13]. Locker ransomware attacks by locking the victim's machine. It may not use, like Crypto ransomware, an encryption scheme. Here, the attacker may request for ransom after locking the machine. The affected individual may physically move the drive to an unaffected PC and acquire the locked files. Some of the Locker ransomware are: Winlock [16] and CTB-locker [17].

In accordance with the Cisco 2017 Annual Cybersecurity Report [14], ransomware attacks continue to increase yearly. Cybersecurity ventures [11] predicts an expansion in cybercrime worldwide, costing to over \$6 trillion by the year 2021. Ransomware is anticipated to have an unquestionably significant effect to this cybercrime by year 2021.

The intuition for using hexacodes as opposed to using opcodes is follows: as the machine architecture changes, the opcodes also change. Therefore, choosing opcodes instead of hexacodes creates ransomware sample vectors of variable length or a lot of missing values. This makes the data mining task hard. If hexacodes are chosen instead of opcodes, the size of the ransomware sample vectors remains constant with no missing values. So, each hexacode frequency vector represents a unique signature for each of the considered file.

This paper is organized with a motivation and literature review in Sects. 2 and 3. Classifications algorithms and feature selection methods employed for the ransomware detection is presented in Sect. 4. Proposed approach is described in Sect. 5. Description of the data used for the current work is given in Sect. 6. Results are presented in Sect. 7, and finally, we conclude in Sect. 8.

2 Motivation

The motivation for the work is as follows:

- To propose a novel method for detecting ransomware only from hexacodes using machine learning, which earlier was not attempted.
- To find out discriminating hexacodes for detecting ransomware using feature selection methods.
- To extract data-driven rules from the hexacode-based ransomware data.

3 Literature Review

A few works have now been done to assess and recognize ransomware. In [21], a resistance mechanism which is absolutely practical is developed. The device investigates the I/O demands and ensuring safeguarding the master record table into the NTFS. This approach gives an option to distinguish and introduce a broader breadth for quantifying critical ransomware attacks. In [28], a device-based approach which is also an adaptive for classifying ransomware is proposed. In [10], an answer for monitoring network traffic information and extract the important features is presented. These features are utilized for classifying ransomware. In [22], a unique framework called UNVEIL to recognize ransomware is presented. It makes use of a counterfeit user environment to caution, when ransomware interacts with the user environment. Encryption and locking activities of ransomware are difficult to conceal by which they pronounce that their methodology can distinguish ransomware by following changes to the victim's file system. Following ransomware end-to-end by [20] demonstrates the ransomware conduct from the contamination to the ransom in bitcoins. They followed the ransomware transactions which frequently involve money. Data mining methods are utilized in [31] to get rules, to find and recognize ransomware families utilizing both static and dynamic methodology. In [32], authors proposed a ransomware discovery methods by just taking API calls into consideration. The ransomware tests are kept running in a sandbox to get the API call data. In [2], network traffic examination was performed for windows ransomware. In [8], introduced a strategy for finding ransomware in virtual servers. The memory that is acquired from crime scene investigation is broke down to make meta-features. In contrast to different works, [24] attempts to investigate and build database from binaries for applying diverse ML models. We also got the research motivation, for using hexacodes, from the malware challenge report of [1].

4 Classification Algorithms and Feature Selection Methods

4.1 Classifiers Employed

Decision Tree (DT) [6] is an ML technique used for classification. A DT generate rules with following parameters like splitting criteria as gini or entropy, best or random splitting, maximum features, minimum samples per split. The rules generated thus indeed represent multiple classes.

Multinomial Naive Bayes (NB) [26] figures the conditional a-posterior probabilities of a class variable given explanatory variables utilizing the Bayes rule. Multinomial Naive Bayes [23] classifier is a Naive Bayes classifier which uses a multinomial distribution for the features.

Multinomial Logistic Regression For logistic regression (LR), please refer [4, 23]. LR is an ML classification technique utilized for binary classification. It can be used for the multiple classes, by employing softmax [3] rather than sigmoid.

Probabilistic Neural Network (PNN) [30] is a ML technique utilized for classification and also for regression. PNN has four layers which are input layer, a pattern layer, summation layer, and an output layer. From the samples of numeric features, PNN combines them into different classes depending on the distribution of the samples.

Extreme Gradient Boosting (XGBoost) uses a tree-based boosting methodology. XGBoost is an off shoot of gradient boosting trees. XGBoost proposed in [7] talks about computational overhead, overfitting criteria, sparsity, and also handling enormous data. The algorithmic usage of XGBoost has a regularization function to work around overfitting.

Random Forest (RF) [5] is a ML technique which can perform both classification and regression. It combines several weak trees learners to come up with a strong learner based on voting. It can likewise be utilized for multi-class problems.

Multilayer Perceptron (MLP) is a class of feed-forward neural networks [27]. A MLP comprises of at least three layers: an input layer, hidden layer/s, and an output layer. With the exception to the input layer, every node uses a nonlinear activation function. MLP uses a back propagation learning procedure to learn nonlinearly separable functions.

K-Nearest Neighbor KNN classifier [9] yields a class membership value for each sample. A sample is grouped by a majority vote of its neighbors, with the sample being assigned to the class that is most common among its k closest neighbors (where k is a positive whole number).

4.2 Features Selection Methods Employed

Information Gain (IG) is computed for each feature considering the output feature [25]. It generates values ranging from 0 (no information) to 1 (most information). Those features that contribute more information will have higher value of information gain, and these can be chosen accordingly by ranking them in a decreasing order.

Gain Ratio (GR). Information Gain is biased toward picking features with large number of values as root [29]. Therefore, in order to decrease this bias, additional measure called gain ratio (GR) is proposed by considering split information, i.e., the quantity of branches that would result before making the split.

5 Proposed Approach

“objdump” is a command line program for showing various information about the files on Unix/Linux environments. For example, it can be utilized as a disassembler to see an executable in assembly-level language along with the corresponding hexacodes.

The proposed methodology is as follows as shown in Fig. 1:

1. Collect the executables of the genuine and ransomware files.
2. Preprocessing procedure:
 - (a) Preprocess the executable to extract hexacodes using the Linux command called “objdump.” The sample of the resultant hexacodes can be seen in Fig. 2.
 - (b) Generate a matrix where the rows represent the files and columns represent the 256 hexacodes, file size, and the corresponding class label. The cells corresponding to hexacodes represent the frequency of occurrence of the hexacode in the file.
 - (c) Impute the missing values with mean value of that particular feature and perform Min/Max normalization.
3. Perform IG or GR based feature selection.
4. Perform tenfold cross validation (10-FCV).
5. Apply the machine learning classifiers mentioned in Sect. 4.1 on the data described in Sect. 6 individually.
6. Perform pairwise statistical t -test on the best performing models, with the features selected by IG and GR.

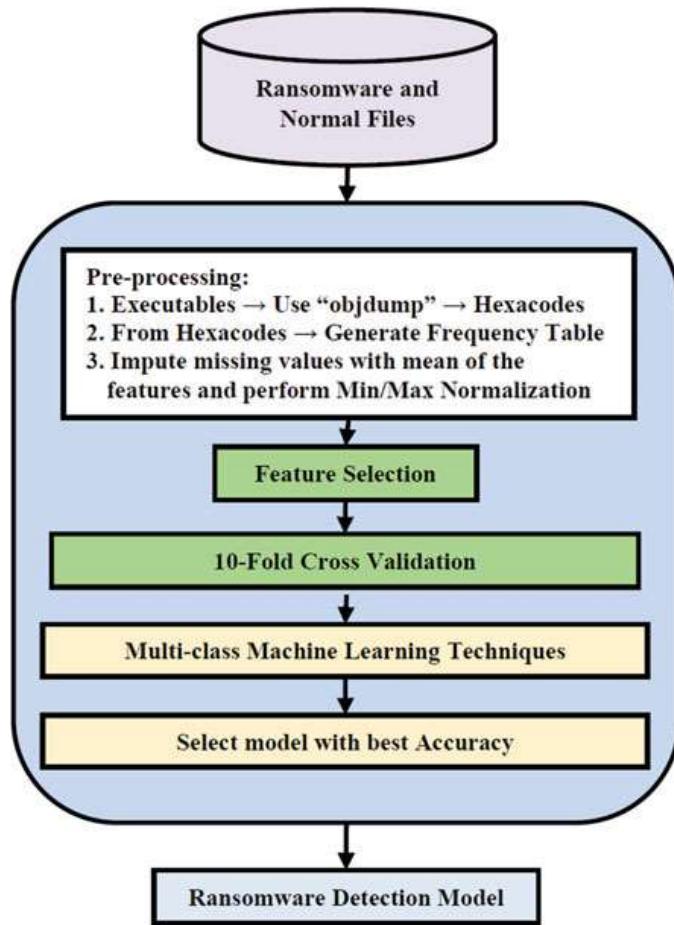


Fig. 1 Proposed approach

```

5 Disassembly of section .text:
6
7 00401000 <.text>:
8 401000:   12 b0 df aa 3e 67      adc  0x673eaadf(%eax),%dh
9 401006:   56                      push %esi
10 401007:  e1 9c                  loop %eax
11 401009:  48
12 40100a:  02 e4
13 40100c:  23 00
14 40100e:  c4 05 61 a0 df e5    and  (%eax),%eax
15 401014:  0c e7
16 401016:  8b dc
17 401018:  be 33 bc dd 13      les  0xe5dfa061,%eax
18 40101d:  79 f0
                                         or   $0xe7,%al
                                         mov  %esp,%ebx
                                         mov  $0x13ddbc33,%esi
                                         jns  0x40100f
  
```

Fig. 2 Hexacodes from Objdump

6 Ransomware Data Description

The data, taken from [24], contains ransomware binaries along with genuine files. We only used 275 of the total 302 binaries, as some could not be labelled with MalwareBytes [15] were ignored. These 275 samples consist of 162 normal executables and 113 ransomware files with total 257 features. 257 features represent hexacodes from ‘00’ to ‘ff’ (i.e., 256 features) and a feature which represents size of the executable. The 275 samples are labelled using MalwareBytes [15] with the December 2018 virus definitions. The class label distribution is given in Table 1.

Table 1 Data description

File type	Count
Normal	162
filelocker	31
teslacrypt	29
locky	20
filelock	5
cryptolocker	4
cryptowall	4
torrentlocker	3
zbot	3
injector	2
filecryptor	2
zerolocker	2
agent.qaz	1
fakems	1
crypt.rpe	1
andromeda	1
kovter.generic	1
xorist	1
malpack.pk	1
agent.gen	1

Table 2 Modified data description

File type	Count
Normal	162
others	33
filelocker	31
teslacrypt	29
locky	20

Some of the class labels count is below 10. 10-FCV cannot be performed on below ten samples. Therefore, we merged them into “others” class for the reason to perform 10-FCV. The modified class label distribution is given in Table 2.

7 Results and Discussion

Model building and model testing were performed on an Intel (R) Core (TM) i7-6700 processor with 32GB RAM. The multi-class models for the ransomware detection problem are built in Python 3. Our emphasis is not on improving the training time.

Accuracy of the models is considered as the performance measure. We see that 30 features chosen from IG based feature selection with RF as the ML technique yielded the best outcomes, and this is demonstrated statistically by a pairwise *t*-test. Results of the study are presented in Tables 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, and 14.

The top 30 features obtained from IG are: **ff, 18, 8b, 5e, c7, 0a, 30, 48, fe, 89, 2, 5, 69, 6a, 7, 0c, 0f, f8, 6, 63, 68, Size, 5c, e9, 95, b8, 85, 53, 44, e4**. One must note that when all 257 features were used, the classification accuracy was 87.17%. However,

Table 3 Results of all features data

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	82.28	0.0704	1.585	0.130
KNN	86.60	0.0714	0.182	0.857
DT	82.65	0.0992	1.191	0.249
NB	64.09	0.0647	7.802	0.000
RF	87.17	0.0675	–	–
XGBoost	86.50	0.0664	0.222	0.826
MLP	84.06	0.071	1.003	0.329
PNN	84.67	0.14	0.508	0.617

Table 4 Results of top 30 features (IG)

MODEL	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	76.08	0.08334	3.92	0.001
KNN	85.124	0.05826	1.29	0.213
DT	82.43	0.10366	1.6	0.127
NB	76.66	0.10337	3.17	0.005
RF	88.34	0.05319	–	–
XGBoost	87.24	0.0758	0.37	0.715
MLP	71.54	0.173	2.93	0.0089
PNN	84.0	0.02	2.41	0.026

Table 5 Results of top 50 features (IG)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	84.217	0.111	0.951	0.354
KNN	87.37	0.0855	0.269	0.790
DT	83.07	0.109	1.225	0.236
NB	67.66	0.1257	4.345	0.0003
RF	88.39	0.08343	—	—
XGBoost	87.54	0.0789	0.37	0.715
MLP	81.41	0.1156	2.93	0.008
PNN	85.78	0.02	2.41	0.026

Table 6 Results of top 100 features (IG)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	84.08	0.0517	1.266	0.221
KNN	85.39	0.0721	0.682	0.503
DT	78.58	0.0724	2.826	0.011
NB	65.52	0.0977	5.811	0.000
RF	87.54	0.0694	—	—
XGBoost	86.14	0.0591	0.486	0.632
MLP	75.74	0.089	3.308	0.0039
PNN	83.62	0.06	1.354	0.1924

Table 7 Results of top 150 features (IG)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	83.01	0.0477	1.2909	0.213
KNN	85.08	0.07019	0.5175	0.611
DT	82.62	0.0662	1.276	0.218
NB	64.81	0.07102	6.484	0.000
RF	86.83	0.0805	—	—
XGBoost	85.75	0.0639	0.332	0.743
MLP	77.90	0.057	2.862	0.0103
PNN	83.71	0.03	1.146	0.266

Table 8 Results of top 200 features (IG)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	83.31	0.049	1.506	0.149
KNN	86.23	0.069	0.3205	0.752
DT	82.14	0.085	1.493	0.152
NB	64.10	0.07094	7.589	0.000
RF	87.19	0.065	—	—
XGBoost	86.20	0.0622	0.346	0.733
MLP	80.45	0.099	1.799	0.088
PNN	83.44	0.13	0.816	0.424

Table 9 Results of top 30 features (GR)

MODEL	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	73.43	0.0588	4.238	0.0004
KNN	86.16	0.0746	—	—
DT	81.44	0.0378	1.787	0.090
NB	63.33	0.0561	7.737	0.000
RF	85.75	0.0713	0.126	0.901
XGBoost	84.27	0.0718	0.578	0.570
MLP	71.49	0.0888	4.001	0.0008
PNN	85.43	0.02	0.298	0.769

Table 10 Results of top 50 features (GR)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	75.29	0.0514	4.16	0.0005
KNN	84.72	0.0726	0.548	0.590
DT	80.12	0.0856	1.834	0.083
NB	65.20	0.1059	5.35	0.00004
RF	86.43	0.0673	—	—
XGBoost	85.35	0.0613	0.376	0.711
MLP	73.91	0.0633	4.284	0.0004
PNN	85.50	0.01	0.433	0.670

Table 11 Results of top 100 features (GR)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	82.25	0.048	1.805	0.087
KNN	83.98	0.0696	0.958	0.350
DT	82.28	0.0666	1.559	0.136
NB	67.78	0.1067	4.831	0.0001
RF	86.09	0.0708	0.256	0.8008
XGBoost	86.87	0.065	–	–
MLP	73.62	0.0823	3.995	0.0008
PNN	84.44	0.04	1.006	0.327

Table 12 Results of top 150 features (GR)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	84.08	0.0676	1.022	0.32
KNN	85.05	0.0777	0.659	0.518
DT	80.79	0.0799	1.911	0.072
NB	64.74	0.0832	6.52	0.000
RF	86.10	0.0703	0.361	0.722
XGBoost	87.24	0.0706	–	–
MLP	85.33	0.0679	0.617	0.544
PNN	84.42	0.05	1.032	0.315

Table 13 Results of top 200 features (GR)

Model	Average accuracy	Std. dev.	<i>t</i> -Value	<i>p</i> -Value
LR	83.34	0.0657	1.198	0.246
KNN	86.56	0.0679	0.095	0.925
DT	82.22	0.0487	1.796	0.089
NB	64.43	0.06158	7.899	0.000
RF	86.05	0.0714	0.259	0.798
XGBoost	86.84	0.06528	–	–
MLP	82.19	0.0672	1.571	0.133
PNN	83.73	0.11	0.77	0.451

Table 14 *t*-Test results

Models, feature selection methods, and number of features	<i>t</i> -Value	<i>p</i> -Value
RF_IG_50 versus KNN_GR_30	0.628	0.537
RF_IG_50 versus RF_GR_50	0.576	0.571
RF_IG_50 versus XGB_GR_100	0.453	0.655
RF_IG_50 versus XGB_GR_150	0.332	0.743
RF_IG_50 versus XGB_GR_200	0.460	0.650
RF_IG_50 versus ALL	0.360	0.722
RF_IG_50 versus RF_IG_30	0.015	0.987
RF_IG_50 versus RF_IG_100	0.245	0.808
RF_IG_50 versus RF_IG_150	0.425	0.675
RF_IG_50 versus RF_IG_200	0.358	0.723

the best model with 30, 50, 100, 150, 200 features from IG gave 88.34%, 88.39%, 87.54%, 86.83%, 87.19% respectively and best model with 30, 50, 100, 150, 200 features from GR gave 86.16%, 86.43%, 86.87%, 87.24%, 86.84%, respectively. Therefore, from the above results, one significant observation is that RF with only 30 features from IG gave an improvement of 1% in accuracy, over the best model with all features.

Apart from the models employed above, the data-driven rules generated by DT can act as early-warning system for the detection of ransomware. The rules from the best performing DT model across all features selection methods, i.e., IG with 50 attributes is presented in Fig. 3. The rules contain inequalities in the antecedent, i.e., the left side part of the inequality represents the hexacode and the right side represents the frequency value, except SizeH, which represents size of the executable. The last part of the rule is the consequent which is the class label.

We performed a statistical significance test, namely *t*-test across models and also across feature selection methods. The pairwise *t*-test is performed on the average accuracy over 10 folds at 1% significance level and 18 degrees of freedom. If the *p*-value is under 0.01, then the null hypothesis is dismissed thereby stating that the pair of models are different statistically. Else, if the *p*-value is more than 0.01, at this point, the null hypothesis is not rejected, and the pair of models compared are the same statistically. *t*-Test results portray that, with a significance value of ‘T’ as true, when the *p*-value is below 0.01, and significance value of ‘F’ as false, when the *p*-value is higher than 0.01. From Table 14, we see that all the best compared models across feature selection methods are statistically the same. Therefore, from Table 14, we conclude to choose Random Forest with 30 features selected from information gain as the best combination, because the *p*-value is highest for the pair and also with a minimal number of features.

```

1. If fdH<=249.5 and 2H<=2843.0 and 6dH<=39.0 and 0H<=5215.5 and 68H<=45.0 then Normal
2. If fdH<=249.5 and 2H<=2843.0 and 6dH<=39.0 and 0H<=5215.5 and 68H>45.0 and 95H<=8.5 then others
3. If fdH<=249.5 and 2H<=2843.0 and 6dH<=39.0 and 0H<=5215.5 and 68H>45.0 and 95H>8.5 and 6dH<=27.5 then filelocker
4. If fdH<=249.5 and 2H<=2843.0 and 6dH<=39.0 and 0H<=5215.5 and 68H>45.0 and 95H>8.5 and 6dH>27.5 then others
5. If fdH<=249.5 and 2H<=2843.0 and 6dH<=39.0 and 0H>5215.5 then Normal
6. If fdH<=249.5 and 2H<=2843.0 and 6dH>39.0 and 44H<=250.5 then filelocker
7. If fdH<=249.5 and 2H<=2843.0 and 6dH>39.0 and 44H>250.5 and c7H<=222.5 then Normal
8. If fdH<=249.5 and 2H<=2843.0 and 6dH>39.0 and 44H>250.5 and c7H>222.5 and 37H<=170.5 and 2eH<=183.5 then filelocker
9. If fdH<=249.5 and 2H<=2843.0 and 6dH>39.0 and 44H>250.5 and c7H>222.5 and 37H<=170.5 and 2eH>183.5 then teslacrypt
10. If fdH<=249.5 and 2H<=2843.0 and 6dH>39.0 and 44H>250.5 and c7H>222.5 and 37H>170.5 then filelocker
11. If fdH<=249.5 and 2H>2843.0 then Normal
12. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 8aH<=250.5 and 30H<=258.0 then others
13. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 8aH<=250.5 and 30H>258.0 and 8bH<=3133.0 then filelocker
14. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 8aH<=250.5 and 30H>258.0 and 8bH>3133.0 then others
15. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 8aH>250.5 and 40H<=1537.5 then Normal
16. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 8aH>250.5 and 40H>1537.5 then locky
17. If fdH>249.5 and 15H<=3497.5 and 18H>392.5 and 2H<=543.0 and e0H<=820.0 then teslacrypt
18. If fdH>249.5 and 15H<=3497.5 and 18H>392.5 and 2H<=543.0 and e0H>820.0 then filelocker
19. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 2H>543.0 and 89H<=1878.0 and c9H<=1221.5 and fdH<=826.0 and 50H<=614.5 then filelocker
20. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 2H>543.0 and 89H<=1878.0 and c9H<=1221.5 and fdH<=826.0 and 50H>614.5 then teslacrypt
21. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 2H>543.0 and 89H<=1878.0 and c9H<=1221.5 and fdH>826.0 then filelocker
22. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 2H>543.0 and 89H<=1878.0 and c9H>1221.5 then others
23. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 2H>543.0 and 89H>1878.0 and 69H<=1324.0 then others
24. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 2H>543.0 and 89H>1878.0 and 69H>1324.0 and 95H<=2116.5 then filelocker
25. If fdH>249.5 and 15H<=3497.5 and 18H<=392.5 and 2H>543.0 and 89H>1878.0 and 69H>1324.0 and 95H>2116.5 then others
26. If fdH>249.5 and 15H>3497.5 and 89H<=2321.5 and 7H<=1562.5 then filelocker
27. If fdH>249.5 and 15H>3497.5 and 89H<=2321.5 and 7H>1562.5 then teslacrypt
28. If fdH>249.5 and 15H>3497.5 and 89H>2321.5 and SizeH<=6.03 and 2eH<=1760.5 and 8aH<=53.0 and 40H<=2273.0 then filelocker
29. If fdH>249.5 and 15H>3497.5 and 89H>2321.5 and SizeH<=6.03 and 2eH<=1760.5 and 8aH<=53.0 and 40H>2273.0 then locky
30. If fdH>249.5 and 15H>3497.5 and 89H>2321.5 and SizeH<=6.03 and 2eH>1760.5 and 8aH>53.0 then locky
31. If fdH>249.5 and 15H>3497.5 and 89H>2321.5 and SizeH<=6.03 and 2eH>1760.5 and fdH<=1587.5 then teslacrypt
32. If fdH>249.5 and 15H>3497.5 and 89H>2321.5 and SizeH<=6.03 and 2eH>1760.5 and fdH>1587.5 then filelocker
33. If fdH>249.5 and 15H>3497.5 and 89H>2321.5 and SizeH>6.03 and 2H<=2592.5 then teslacrypt
34. If fdH>249.5 and 15H>3497.5 and 89H>2321.5 and SizeH>6.03 and 2H>2592.5 then filelocker

```

Fig. 3 Rules obtained from DT

8 Conclusions and Future Directions

In this study, we proposed a novel ransomware detection method based on just hexacodes and without opcodes and invoking machine learning classifiers. The use of hexacodes alone is a significant departure from the extant research. We employed two feature selection methods, namely information gain (IG) and gain ratio (GR), and concluded that IG with RF is statistically significant. We also presented 34 ‘if-then’ rules obtained by DT, which could be used as an early-warning expert system. One significant observation is that, RF with only 30 features from IG, gave an improvement of 1% in accuracy, over the best model with all features. In future, we can employ wrapper-based feature selection methods on the hexacode features and also deep learning methods as classifiers. Further, instead of the frequency, we could consider TF-IDF values of the hexacodes while forming the document-term matrix. Moreover, we can also perform class association rule mining.

References

1. Agarwal, P., Bansal, K.: Malware Classification Challenge. Technical report
2. Alhwai, O.M., Baldwin, J., Dehghanianha, A.: Leveraging machine learning techniques for windows ransomware network traffic detection. In: Dehghanianha, A., Conti, M., Dargahi, T. (eds.) Advances in Information Security, vol. 70, pp. 93–106. Springer, New York (2018)
3. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)

4. Bohning, D.: Multinomial logistic regression algorithm. *Ann. Inst. Stat. Math.* **44**(1), 197–200 (1992)
5. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
6. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: *Classification and Regression Trees*. Wadsworth International Group, Monterey, CA (1984)
7. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining—KDD '16*, San Francisco, CA, USA, pp. 785–794. ACM Press (2016)
8. Cohen, A., Nissim, N.: Trusted detection of ransomware in a private cloud using machine learning methods leveraging meta-features from volatile memory. *Expert Syst. Appl.* **102**(C), 158–178 (2018)
9. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* **13**(1), 21–27 (1967)
10. Cusack, G., Michel, O., Keller, E.: Machine learning-based detection of ransomware using SDN. In: *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks and Network Function Virtualization (SDN-NFVSec 2018)*, Co-located with CODASPY 2018, Tempe, AZ, USA, pp. 1–6. Association for Computing Machinery, Inc. (2018)
11. <https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/> : Cybercrime Damages \$6 Trillion by 2021
12. <https://www.avast.com/c-cryptolocker> : CryptoLocker Ransomware
13. <https://www.avast.com/c-locky> : Locky Ransomware
14. https://www.cisco.com/c/m/en_au/products/security/offers/annual-cybersecurity-report-2017.html : Cisco 2017 Annual Cybersecurity Report
15. <https://www.malwarebytes.com/business/>: Malwarebytes
16. <https://www.microsoft.com/en-us/wdsi/threats/malware-encyclopedia-description?Name=Ransom:Win32/LockScreen.AO> : WinLock Ransomware
17. <https://www.secalliance.com/blog/ransomware-ctb-locker/> : CTB-Locker Ransomware
18. <https://www.secureworks.com/research/cryptowall-ransomware> : Cryptowall Ransomware
19. <https://www.us-cert.gov/ncas/alerts/AA18-337A> : SamSam Ransomware
20. Huang, D.Y., Aliapoulios, M.M., Li, V.G., Invernizzi, L., Bursztein, E., McRoberts, K., Levin, J., Levchenko, K., Snoeren, A.C., McCoy, D.: Tracking ransomware end-to-end. In: *IEEE Symposium on Security and Privacy*, San Francisco, CA, USA, pp. 618–631. Institute of Electrical and Electronics Engineers Inc. (2018)
21. Kharraz, A., Robertson, W., Balzarotti, D., Bilge, L., Kirda, E.: Cutting the gordian knot: a look under the hood of ransomware attacks. In: Almgren, M., Gulisano, V., Maggi, F. (eds.) *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Milan, vol. 9148, pp. 3–24. Springer, Italy (2015)
22. Kirda, E.: UNVEIL: a large-scale, automated approach to detecting ransomware (keynote). In: *IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, Klagenfurt, Austria, pp. 1–1. Institute of Electrical and Electronics Engineers (IEEE) (2017)
23. Krishnapuram, B., Carin, L., Figueiredo, M., Hartemink, A.: Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(6), 957–968 (2005)
24. Poudyal, S., Subedi, K.P., Dasgupta, D.: A framework for analyzing ransomware using machine learning. In: *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*, Bangalore, India, pp. 1692–1699. Institute of Electrical and Electronics Engineers Inc. (2019)
25. Quinlan, J.R.: Induction of decision trees. *Mach. Learn.* **1**(1), 81–106 (1986)
26. Rish, I.: An empirical study of the naive Bayes classifier. In: *IJCAI—Workshop on Empirical Methods in Artificial Intelligence*, Seattle, Washington, USA, pp. 41–46 (2001)
27. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. *Nature* **323**(6088), 533–536 (1986)

28. Sgandurra, D., Muñoz-González, L., Mohsen, R., Lupu, E.C.: Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection (2016)
29. Shih, Y.S.: A note on split selection bias in classification trees. *Comput. Stat. Data Anal.* **45**(3), 457–466 (2004)
30. Specht, D.F.: Probabilistic neural networks. *Neural Netw.* **3**(1), 109–118 (1990)
31. Subedi, K.P., Budhathoki, D.R., Dasgupta, D.: Forensic analysis of ransomware families using static and dynamic analysis. In: IEEE Symposium on Security and Privacy Workshops, SPW 2018, San Francisco, CA, USA, pp. 180–185. Institute of Electrical and Electronics Engineers Inc. (2018)
32. Takeuchi, Y., Sakai, K., Fukumoto, S.: Detecting ransomware using support vector machines. In: ACM International Conference Proceeding Series, Eugene, OR, USA. Association for Computing Machinery (2018). Article No. 1

Hypertension Risk Prediction Using Deep Neural Network



M. J. Sivambigai and E. Murugavalli

Abstract Predictions about the future are done by learning from historical data. In medical care, forecast analysis will empower the best results to be made, enabling for care to be individualized to each person. Hypertension is a prevalent state that can lead to severe problems if untreated. This paper presents a risk prediction model for hypertension using deep neural network. Immediately users get notified about their risk status and doctors also receive alerts about their patient's status. These alert notifications are sent to the respective people via email. The results of the prediction model show that the deep neural network and decision tree both achieved an accuracy of 95.74% and 92.63%, respectively. From the comparison of the two models, deep neural network model was chosen for the risk prediction of hypertension.

Keywords Risk prediction · Deep neural network · Decision tree · Hypertension

1 Introduction

Hypertension (HT) is a worldwide health problem which is caused by high blood pressure (BP) in the blood vessels [1]. Fatigue, severe headache, disorientation, chest pain, shortness of breath, and irregular heartbeat are the symptoms caused by high BP [1]. About 7.5 million or 12.8% deaths globally happen because of raising BP [2]. The pervasiveness of hypertension in developing countries is almost low, ranging from 10 to 15% in Asia, but it is steadily increasing and thriving at a faster rate. It is likely to estimate that about 1.56 billion adults will have hypertension in year 2025 [3]. For the last three decades, in India, the number of people with hypertension is increased by 30 times amidst urban inhabitants and by 10 times amid rural Indians [4].

M. J. Sivambigai · E. Murugavalli
Thiagarajar College of Engineering, Madurai, Tamil Nadu 625015, India
e-mail: sivambigaimj1996@gmail.com

E. Murugavalli
e-mail: murugavalli@tce.edu

Due to the progress made in computer technology, artificial intelligence (AI) data mining techniques have been broadly used in the medical field which provide better prediction accuracy than traditional statistical methods [5–7]. At recent times, a branch of machine learning (ML) based on deep learning (DL) approaches, such as deep neural network (DNN), has attained magnificent and produces greater outcomes across various AI applications [8]. Most advanced AI applications are powered by DL models. DL is a common name for neural network approaches, which is based on picking up patterns from raw data and usually hold more than one hidden layer [9]. These approaches are nowadays the state of the art in the areas of computer vision, natural language processing, and speech recognition. Advantages of DNN against traditional ML methods incorporate that they need a less domain understanding of the problem they try to solve and are easier to scale due to increase in accuracy is generally realized either by increasing the training data or the capacity of the network.

Nowadays, many devices, for instance, Internet of things (IoT) sensors, are used in health care for sensing. Hence, huge amount of data is generated from these devices. Conventional decision-making method is of no use when the dataset is large and a difference has to be brought in for decision making. Gaining intuitions from huge data is hard for doctors which fundamentally affects the accuracy of the result. Hence, predicting the disease existence will be a great clinical important for the doctors. Most powerful data-driven algorithms likely increase the ability of prevention and improve the outcomes of patient through primary detection and treatment. Hence, the following are the contributions of this paper.

- To predict the hypertension risk using deep neural network based on the user's health parameters, occupational, and personal details
- To make alerts regarding the risk prediction and send it to patient and doctor's mail ids.

This paper is organized into different sections. In Sect. 2, we summarize the researches related to prediction of hypertension. Section 3 presents in detail about the proposed work. Section 4 provides the results and their inferences. Section 5 concludes the paper with the future scope.

2 Related Work

Literatures in the past revealed that researchers have utilized different algorithms for the classification of hypertensive patients and also for the prediction of hypertension in patients.

In 2016, LaFreniere et al. [10] proposed an artificial neural network (ANN) model to predict hypertension in the Canadian Primary Care Sentinel Surveillance Network (CPCSSN) data where 82% accuracy is achieved. In 2016, Quachtran et al. [11] analyzed the advantage of convolutional neural network (CNN) to withdraw attributes automatically from waveform morphology that are associated with intracranial hypertension. While testing, the model predicted the high intracranial pressure

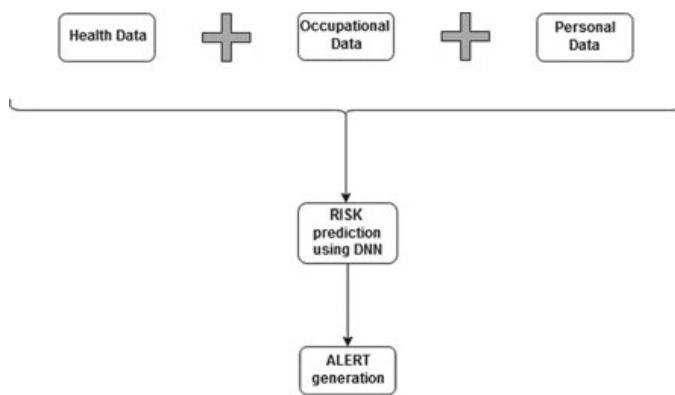
from dataset with $92.05 \pm 2.25\%$ accuracy. In 2018, Chai et al. [12], the model consists of decision tree algorithm and a Bayesian classifier to predict whether the person will be affected by the disease or not and to give proper medications. In 2018, Sood et al. [1], the data are collected in real time using Internet of things (IoT) sensors and are processed at the fog layer where ANN is used for the prediction and alerts are generated continuously to the user's mobile. In 2019, Wu et al. [13], the learning vector quantization (LVQ)-neural network algorithm and the Fisher–support vector machine (SVM) coupling algorithm are used to evaluate the risk of hypertension in steel workers, whereas LVQ neural network model is more accurate and 90.50% accuracy is achieved.

Most of the researches used neural networks. We aim at using deep learning models to predict the risk of hypertension and generating alerts about the prediction made in user's email, which allows doctors to determine the patterns available in data better and enhance the essential performance measures.

3 Methodology

The functional flow diagram of our work is shown in Fig. 1. The course of our work can be split into three different parts. They are data part, prediction part, and finally the alert generation part. The data part consists of various data needed in order to predict the risk of hypertension. BP, cholesterol, blood glucose, and heart rate are the parameters that have to be kept in control, because they are responsible for the risk of hypertension in patients. The model is perfectly modeled and utilized to make risk prediction of hypertension. After the prognosis, notifications are made regarding the outcome obtained from the model. And those messages will be sent to the user's mail id. It is vital to know their risk status which aids in extending their lifespan. This acts as an alarm and allows users to take preventive medications on time so that they can avoid fatal situations.

Fig. 1 Functional flow diagram



3.1 Dataset

The first part is data collection, which is mandatory. It acts as the essence to train the model. The dataset is obtained from [14]. Our model completely depends on these data for prediction.

Health data: Health data mainly consist of BPs, cholesterol, blood glucose, and heart rate. To manage the risk of disease, individuals have to keep a check on their health parameters.

BP stages have been known to be positive and unceasingly correlated to the risk of the disease. Typically, BP is articulated with two numbers. The stages of HT are shown in Table 1. Systolic BP (SBP) is the upper number which denotes to the pressure in the blood vessel when the heart beats. Diastolic BP (DBP) is the lower number which signifies to the pressure in the middle of heartbeats [15].

Cholesterol is conveyed in blood by high-density lipoprotein (HDL) together with low-density lipoprotein (LDL). The different class of total cholesterol is shown in Table 2. A person's total cholesterol is made up of one-fifth triglyceride along with both HDL and LDL cholesterol [16]. The plaque which results from cholesterol gets deposited on the walls of blood vessel. The narrowing raises the pressure of blood.

Blood glucose (BG) at times mentioned as blood sugar, which exactly denotes the quantity of glucose in blood. BG level of a being is pretentious by various causes. Diabetes can cause damage to arteries by hardening them, which leads to the cause of high BP. Table 3 provides the BG chart.

Heart rate (HR) is the number of heartbeats per minute. HR is related to raise in BP, in terms of increasing the risk of hypertension. The normal heart rate for adults is about 60–100 bpm.

Table 1 Classification of stages of HT

Stages of HT	SBP (mmHg)	DBP (mmHg)
Normal	<120	<80
Prehypertension	120–139	80–89
Hypertension stage 1	140–159	90–99
Hypertension stage 2	≥160	≥100
Hypertensive emergency	≥180	≥110

Table 2 Total cholesterol count

Classes	Total cholesterol count (mg/dL)
Normal	<200
Marginal high	200–239
Very high	≤240

Table 3 Blood glucose chart

Level (mg/dl)	Fasting	Post-meal	2–3 h after meal
Normal	80–100	170–200	120–140
Reduced glucose	101–125	190–230	140–160
Diabetic	>126	220–300	>200

These are some of the important attributes which have impact on hypertension, whereas other features including body mass index (BMI), information related to presence of stroke, heart disease, hypertension, diabetes, and BP medications are also taken into consideration for better results. Hence, these attributes are considered to be the primary data used for the hypertension prediction whereas the remaining data too have impact on hypertension and therefore they are considered as secondary sources.

Occupational data: Work pressure is also a cause for the disease. Hence, work type is considered as an attribute in the dataset. The diverse work types fall into the category of private, public, self-employed, and never worked. The prevailing models do not take this as an attribute. Here, it was considered in a more general form of occupational types.

Personal data: The age, gender, marital status, residence type, and smoking status all come under the user's personal data. These data too have an impact on the disease. For both male and female, the BP goes up with age. The BP of one completed 50 years would rise more quickly than the BP of those between 20 and 39 years [17]. Experiments show that smoking has a noteworthy result of the BP [18, 19].

3.2 *Prediction Algorithms*

Deep neural network: Neural network (NN) is interconnected by a number of neurons, motivated by data computing in the human brain. DNN is distinguished from the more conventional single hidden layer NN by their complexity, that is, the number of hidden layers through which data are passed through various steps of pattern recognition. Untrained NN are like infants. They are produced unaware of the world, and it is merely through disclosure of the world, going through it, they slowly attain knowledge. Training a NN with a relevant dataset, its ignorance is diminished. The NN knowledge with respect to the world is seized by its weights. The output layer will make a decision which is often wrong about the input, because they have not learned enough yet. These values are then squashed by applying activation functions. There are many activation functions available. Some are linear function, sigmoid function, tanh function, rectified linear unit (ReLU) function, and softmax function.

$$\text{ReLU}(\text{val}) = \max(0, \text{val}) \quad (1)$$

$$\text{Sigmoid (val)} = \frac{1}{1 + e^{-\text{val}}} \quad (2)$$

Equation 1 is used as the activation function in the hidden layers whereas Eq. 2 is used in the output layer since it is binary, i.e., either 1 or 0. The term val refers to the output. It results in same val if it is positive or else it is 0 in case of ReLU, whereas in case of Sigmoid, it lies in the range of 0–1. Hence, the output val will be 1 if it is greater than 0.5 or else it will be 0.

After the prediction is completed, the difference between the true value and predicted value is calculated which is termed as an error or loss. The errors are reduced by adjusting the weights using gradient descent optimization and backpropagation which computes the gradient of the cost function [20, 21].

$$W = W - \mu \frac{\partial \text{loss}}{\partial W} \quad (3)$$

where W is the weight and μ is the learning rate in Eq. 3. Choosing a learning rate is vital because if we choose it to be extremely small, then the DNN will learn very slowly, and on the other hand, if we choose it to be very large, then we would not be capable of reaching the least possible value.

Decision tree: Decision tree (DT) is one of the most common ML algorithms that frequently imitate the human-level intelligence and simple to comprehend the data and make some good understandings. A DT is a tree where each node signifies an attribute, each connection denotes a decision, and each leaf represents an output. The DT is a distribution free or nonparametric technique, which does not rely upon likelihood distribution expectations. DTs can manage high-dimensional info with greater precision.

First calculate the entropy for the entire dataset which is represented as S using Eq. 4,

$$\text{Entropy} = \frac{-p}{p+n} \log_2 \left[\frac{p}{p+n} \right] \frac{-n}{p+n} \log_2 \left[\frac{n}{p+n} \right] \quad (4)$$

Then for each and every attribute or feature, compute entropy for all categorical values. Take average information entropy for the current attribute using Eq. 5 where A stands for individual attribute or feature in the dataset.

$$I(\text{attribute}) = \sum \frac{p_i + n_i}{p+n} \text{Entropy}(A) \quad (5)$$

Calculate gain for the current attribute using Eq. 6. Then pick the highest gain attribute. Repeat until we get the desired tree.

$$\text{Gain} = \text{Entropy}(S) - I(\text{attribute}) \quad (6)$$

3.3 Alert Notification

The alert notification is sent to user's email every time whenever prediction is done. The alerts are regarding the predicted risk level of hypertension. It is also sent to the doctor's email to notify about the current state of the user who needs special care. For users, this is done in order to make the individual conscious about the disease and to lessen the worse effects. The communication is quick, effective and seeks the instant attention.

4 Results and Discussion

In this section the results are presented in four parts. They are DNN algorithm, DT algorithm, comparison of both algorithms, and alert generation part. All the executions of this work were carried out in Python and the datasets were maintained in Excel 2013.

4.1 Deep Neural Network Algorithm

Our DNN model has three hidden layers interconnected in the middle of input and output layers, with their sizes 8, 8, and 4. The number of hidden layers and the number of neurons were decided on trial and error basis. ReLU and sigmoid are used as the activation functions in the hidden and output layers, respectively, since the output is of binary type, i.e., either 1 or 0 representing whether the user has the risk of hypertension or not.

After training the model, it is evaluated on the same data. Preferably the loss has to be 0 and the precision has to be 1. But it is not achieved most of the times due to little amount of error. So our main aim was to train the model with minimum loss and with maximum accuracy.

Here, validation is done by splitting the entire dataset into two, i.e., 80% for training and remaining 20% for validation purpose. Then the model achieved an accuracy of 95.74% as shown in Fig. 2. From this figure, we infer that our curve shows a sign of good fit. As epoch increases, DNN learns and recognizes the pattern well and hence accuracy also keeps increasing.

4.2 Decision Tree Algorithm

Splitting of dataset into training and validation is the best approach in order to learn about the model performance. The same 80 and 20% of split is used, respectively.

Fig. 2 Accuracy plot of DNN algorithm

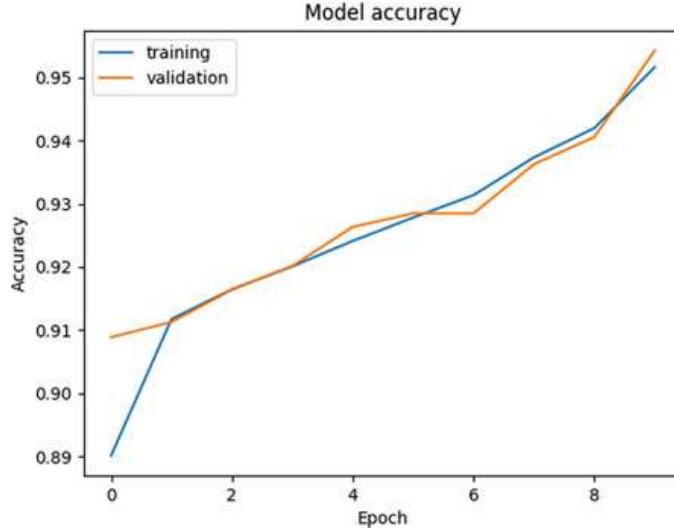


Fig. 3 Accuracy plot of DT algorithm

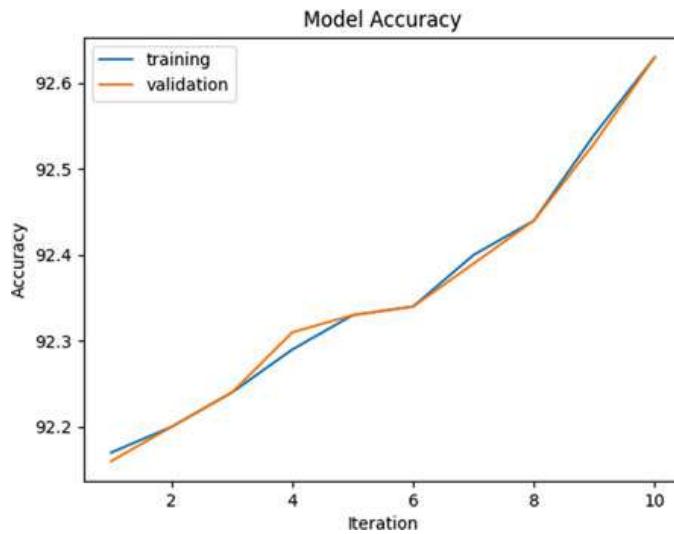


Figure 3 shows that the DT has achieved an accuracy of 92.63%. It is plotted over accuracy versus iterations. During each iteration, the difference of accuracy was only in point. Thus, the whole number 92 was achieved every time.

4.3 Comparison of DNN and DT

This part brings out the contrast of ML and DL algorithms. These two algorithms can be compared because both can handle nonlinear relationships among features and have connections with the attributes. Hence, DNN is compared with DT.

From the comparison shown in Fig. 4, we get to know that the DL algorithm outperforms the ML algorithm with a 3.11% difference between them. This difference

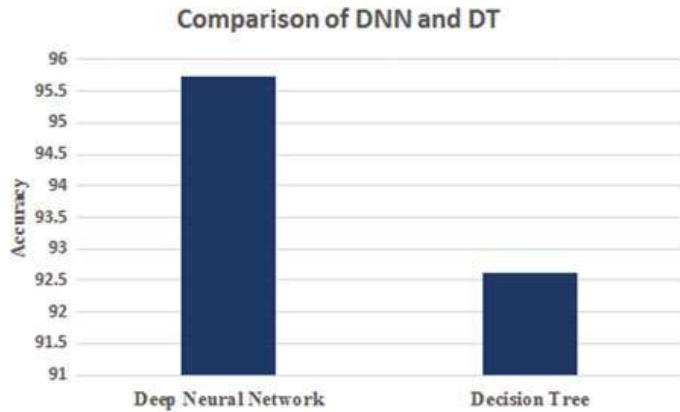


Fig. 4 Comparison of DNN and DT

could result due to the fit of the models. The model fit comes from our dataset. Thus, it shows that our dataset is good. The best of two is identified by accuracy since it reflects that it has perfectly fitted with our dataset. Thus, DNN is used in the model for risk prediction.

Even though DT is not the best in our case, it has certain advantages over DNN such as they are easily understood and hence refer to them as white box whereas NN are called as black box, then ML algorithm can perform well with small-sized dataset too.

4.4 Alert Generation

Python has a Simple Mail Transfer Protocol (SMTP) client session entity needed to send mail to other legal mail ids on the network. Initially, create an account with the use of SMTP objects to establish connection. Thereafter, it enters into security mode for encryption purpose. Feed the valid mail username and password to make login details. Pass the mail ids to whom ever need to get notified.

It can be seen in Fig. 5 that for the above-mentioned attributes in the image DNN has predicted the risk as 0, i.e., no risk. Therefore, the program gets terminated.

For this different set of attributes in Fig. 6, the prediction is 1, i.e., risk. Hence, it sends alert notification to user and doctor's mail ids to make aware about this. After sending alerts to all respective people, the program gets terminated.

Figures 7 and 8 show the notification received via mail for user and doctor. First image in both figures shows the unread inbox message, and the second image in the figures shows the full message after it is read.

```

hypertension_predictor.py
venv
nn.py
CNNtry.py
DBN.py
Yamingham.csv
hypertension_data.csv
hypertension_predictor

Using TensorFlow backend.
2019-10-16 20:48:03.544822: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 F
2019-10-16 20:48:03.569999: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 1696075000 Hz
2019-10-16 20:48:03.570639: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x51bbc90 executing computations on platform Host. Devices:
2019-10-16 20:48:03.570682: I tensorflow/compiler/xla/service/service.cc:175] StreamExecutor device (0): Host, Default Version
[[0]]
Process finished with exit code 0

```

Fig. 5 Predicting as no risk

```

hypertension_predictor.py
venv
nn.py
CNNtry.py
DBN.py
framingham.csv
hypertension_predictor

2019-10-16 20:24:29.559390: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AV
2019-10-16 20:24:29.718117: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 1696075000 Hz
2019-10-16 20:24:29.718912: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0xabbcc400 executing computations on platform Host. Devices:
2019-10-16 20:24:29.718912: I tensorflow/compiler/xla/service/service.cc:175] StreamExecutor device (0): Host, Default Version
[[1]]
Its HT!!! Sending mail to necessary people
Sent message to patient successfully
Sent message to doctor successfully
Process finished with exit code 0

```

Fig. 6 Predicting as risk



Fig. 7 Alert received by the user via mail



Fig. 8 Alert received by the doctor via mail

5 Conclusion

Hypertension is a typical reason of unexplained disease with higher arterial pressure. Hence, a risk prediction model for hypertension is significant. The prediction was done using two different algorithms, and the algorithm which achieved greater accuracy among them was used for the model. Alerts generated and sent via email make patients and doctors to prevent the risk of hypertension at the initial stage. This work can be extended to predict other such diseases along with the addition of real-time data.

References

1. Sood, S.K., Maharajan, I.: IoT-fog-based healthcare framework to identify and control hypertension attack. *IEEE Internet Things J.* **6**(2) (2019)
2. Mendis, S.: Global status report on non-communicable diseases 2010. Tech. Rep., World Health Organisation (2010)
3. Tabrizi, J.S., Sadeghi-Bazargani, H., Farahbaksh, M., Nikniaz, L., Nikniaz, Z.: Prevalence and associated factors of prehypertension and hypertension in Iranian population: the lifestyle promotion project (LPP). *PLoS ONE* **11**(10), Article ID e0165264 (2016)
4. Gupta, R.: Rethinking diseases of affluence: coronary heart disease in developing countries. *South Asian J. Prev. Cardiol.* **10**, 65–78 (2006)
5. Wu, J., Zhang, L., Yin, S., Wang, H., Wang, G., Yuan, J.: Differential diagnosis model of hypocellular myelodysplastic syndrome and aplastic anemia based on the medical big data platform. *Complexity* **2018**, Article no. 4824350 (2018)
6. Lui, Y.W., et al.: Classification algorithms using multiple MRI features in mild traumatic brain injury. *Neurology* **83**(14), 1235–1240 (2014)
7. Tseng, W.-T., Chiang, W.-F., Liu, S.-Y., Roan, J., Lin, C.-N.: The application of data mining techniques to oral cancer prognosis. *J. Med. Syst.* **39**(5), 1–7 (2015)
8. Hung, C.-Y., Chen, W.-C., Lai, P.-T., Lin, C.-H., Lee, C.-C.: Comparing deep neural network and other machine learning algorithms for stroke prediction in a large-scale population-based electronic medical claims database. In: 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 3110–3113 (2017)
9. Singh, D., et al.: Human activity recognition using recurrent neural networks. In: Machine Learning and Knowledge Extraction. Lecture Notes in Computer Science, vol. 10410, pp. 267–274 (2017)
10. LaFreniere, D., Zulkernine, F., Barber, D., Martin, K.: Using machine learning to predict hypertension from a clinical dataset. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI) (2016)
11. Quachtran, B., Hamilton, R., Scalzo, F.: Detection of intracranial hypertension using deep learning. In: 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 2491–2496 (2016)
12. Chai, S., Wu, L.Y., Chang, S.-T., Lin, C.-J., Liu, Y.-T.: Establish a Predictive Model of Hypertension Complications, vol. 2, pp. 515–520 (2018)
13. Wu, J.-H., Wei, W., Zhang, L., Wang, J., Robertas, D., Jing, L., Wang, H.-D., Wang, G.-L., Zhang, X., Yuan, J.-X., Wozniak, M.: Risk assessment of hypertension in steel workers based on LVQ and Fisher-SVM deep excavation. *IEEE Access* **7**, 23109–23119 (2019)
14. Available at <https://www.kaggle.com/asaumya/patient-data-train-and-test-set> and <https://www.kaggle.com/navink25/framingham>
15. American Heart Association: Understanding Blood Pressure Readings (2011)

16. American Heart Association: Good vs. Bad Cholesterol (2009)
17. Sparrow, D., Garvey, A.J., Rosner, Jr., B., Thomas, H.E.: Factors in predicting blood pressure change. *J. Circ.* **65**, 789–794 (1982)
18. American Heart Association: Prevention and Treatment: Tobacco and Blood Pressure (2014)
19. Omvik, P.: How smoking affects blood pressure. *Blood Press.* **5**, 71–77 (1996)
20. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. *Nature* **323**(6088), 533 (1986)
21. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning, vol. 1. MIT Press, Cambridge (2016)

Machine Learning Approach for Student Academic Performance Prediction



Sachin Rai, K. Aditya Shastry, Surendra Pratap, Saurav Kishore, Priyanka Mishra, and H. A. Sanjay

Abstract The assessment in outcome-based learning is very vital. A significant approach toward measuring the student performance is required. Nowadays, the amounts of data that are stored in educational database are increasing rapidly. At any point of time, the institutions face various problems. One is the poor performance of the student. Second one is the exit of a student due to the complexity of curriculum, financial problems, psychological problems, lack of support, etc. To handle these issues, machine learning can be utilized. Considerable work has been done to measure the student's performance by using different methodologies and modern technologies. In this work, the machine learning (ML) algorithms have been implemented on the student dataset of a university to predict the student's performance. Based on the analysis of the result, it has been concluded that accuracy of the random forest (RF) classifier is more than the other classification method such as support vector machine (SVM).

Keywords Prediction · Machine learning · Preprocessing · Random forest · Student performance

S. Rai · K. A. Shastry (✉) · S. Pratap · S. Kishore · P. Mishra · H. A. Sanjay
Department of Information Science and Engineering, Nitte Meenakshi Institute of Technology,
Bengaluru, India
e-mail: adityashastry.k@nmit.ac.in

S. Rai
e-mail: sachinrai1122@gmail.com

S. Pratap
e-mail: surendraps007@gmail.com

S. Kishore
e-mail: souravsk2019@gmail.com

P. Mishra
e-mail: primisha2012@gmail.com

H. A. Sanjay
e-mail: sanjay.ha@nmit.ac.in

1 Introduction

Higher education has gained significance during the earlier few years. The objectives and scope of the higher educational institutes are revised periodically due to several factors such as private participation, faculty, among others. Furthermore, different students have varying grasping capabilities and obtain different outcomes. Also, it is difficult to acquire and analyze the different educational needs of students [1].

Recently, ML methods are being utilized to forecast the student's academic performance particularly in his/her first year at a university. The key objective of machine learning is to discover hidden and significant associations within the information having diverse characteristics. Various techniques of machine learning like decision trees are found to be effective for forecasting the performance of students based on the data stored in the university's database. The forecast of the academic student's performance utilizing machine learning is developed with the objective of automating the student results prediction process [2].

The objective of this work is to split the students into three different classes which are good, average, and poor classes by analyzing the student dataset. To measure the performance of the student, we need to find the main attributes which influence the student's performance. We are using the machine learning to identify the level of understanding and provide learning to the student as per their requirements. Lately, the universities are growing day by day in numbers. Hence, providing quality education and improving the success rate of students are equally important. The biggest challenges for today's education system are to provide the quality education and get the higher success rate for the student's overall performance.

The structuring of the paper is as follows. Section 2 describes the background work in the field of forecasting the academic performance of student using machine learning. Section 3 gives the details of the proposed work. Section 4 gives the details about the trials that were carried out and results obtained along with the comparison of results. This is followed by the conclusion.

2 Related Work

In this section, previous works on predicting the students' performance utilizing ML methods are discussed.

Ahmed et al. [1] predicted the final grade of students using decision tree. The student behavior was analyzed using educational data mining. Post preprocessing, the authors applied different methods like association, clustering, and classification to extract knowledge describing the student's behavior.

In [2], the adaptive neuro-fuzzy inference system (ANFIS) was utilized to predict the students' performance which would improve their academic success. In [3], Acharya and Sinha applied machine learning algorithms such as decision trees for the prediction of students' results.

The slow learners were identified and displayed by Kaur et al. [4] by utilizing classification-based algorithms based on predictive modeling. The student demographics linked with their success were found by Gurlur et al. [5] utilizing decision trees. The work by Vandamme et al. [6] performed early prediction of student's performance using the ML methods like neural networks, linear discriminant analysis, and decision trees.

Tair and Halees [7] applied the educational data mining to improve performance of graduate students having low grades. Educational data mining was utilized for extracting knowledge from the graduate student's data that was collected between the years 1993 and 2007 from the college of Science and Technology.

The dropout possibility was predicted by Bharadwaj et al. [8] utilizing the decision tree classifiers. The aim was to reduce the dropout frequency among graduates and increase student retention. In [9], the authors extracted knowledge of student's performance from result sheets using decision trees. This aided in identification of dropouts and allowed the tutor to counsel the students.

Cortez and Silva [10] applied four classification techniques, namely decision trees (DT), RF, artificial neural networks (ANN), and SVM to improve the student's performance and help to achieve the goal by extracting the discovery of knowledge from the end semester marks.

3 Proposed Work

This section illustrates the proposed method used for student academic performance prediction using machine learning approach. Figure 1 depicts this process.

The detailed steps in the proposed approach are described below:

1. Student dataset collection: The student academic dataset is collected from the UCI student performance dataset [11]. The data consists of 8000 data 831 samples with 22 attributes for each student. The datasets published by UCI have more records, attributes and classes when compared to other public datasets. Some of the attributes used were credit hours, study field, study hours, success rate, among others. The original dataset consisted of 21 input attributes corresponding to gender, caste, SSLC marks, PUC marks, internal assessment marks, end sem, year back status, marital status, town or village, admission type, family income, family size, father qualification (FQ), mother qualification (MQ), father occupation (FO), mother occupation (MO), friend size, SH, school type, medium of education, time travel (TT). The output attribute was the class to be predicted.
2. Preprocessing: Data preprocessing is utilized as an initial data processing step. It transforms the information into a format which is suitable for the ML algorithms. The preprocessing is done on the data collected from UCI having multivariate attributes. The preprocessing involved the following steps:
 - Importing the libraries: We imported pandas, label encoder, train_test_split, random forest classifier, accuracy_score. The label encoder holds the label for

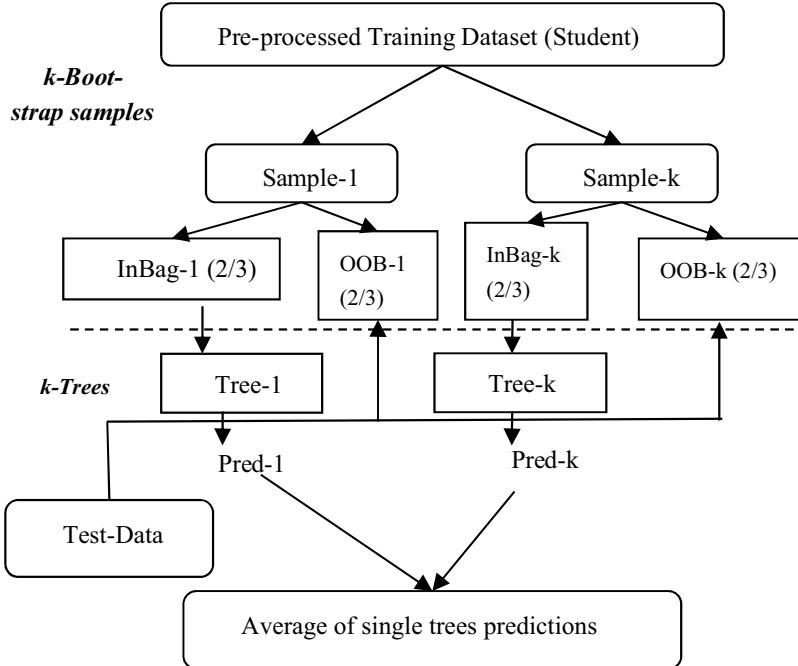


Fig. 1 ML approach for prediction of the academic performance of student

each class. It encodes categorical attributes by utilizing a one-hot or ordinal encoding scheme. The `train_test_split()` function splits the data into a training set and a test set. We provided 30% of data to be used as a test set. The parameter `random_state` was used to provide the random seed so that repeatable results are produced. The RF classifier was made up of DT classifiers that were applied on different subsets of data. The predictive accuracy is improved, and over-fitting is controlled by averaging the outcomes of different classifiers. The `accuracy_score` represents the number of matches between the actual and predicted values [12].

- Handling of missing values: Sometimes, a dataset consists of “holes” in it which are known as missing values. Certain statistical methods like regression analysis fail to work on dataset having missing values. Hence, these observations must be removed or substituted using appropriate statistical procedures to produce meaningful results. In python, pandas can detect the missing values; pandas will recognize both empty cells and “NA” types as missing values. We have used `dataset.isnull().any().sum()` function for detecting the missing values [13].
- Splitting data: In ML, the data is separated into test and training sets. The training set comprises of records with known labels. The ML model is trained on the training dataset so that it acquires learning ability. The model’s prediction ability is then tested on the test data with unknown labels. The `train_test_split()` function was employed to perform the partition. In this

work, 70–30% splitting was used, where 70% comprised the training data and the remainder of 30% formed the test dataset.

3. Feature selection (FS): A subgroup of pertinent features (variables, predictors) is identified by feature selection techniques to facilitate the effective construction of ML models. It is also called as variable subset selection/attribute selection/variable selection. The FS techniques are utilized for four reasons mentioned below [14]:

- To simplify models so that they are easily interpretable by the researchers/users.
- To reduce the training time required for building the models.
- To prevent the dimensionality curse.
- To reduce over-fitting by enhanced generalization.

Many datasets comprise of dependent or redundant features. It is crucial that we identify the features that make an actual difference to the work at hand. The technique of random forest falls under the group of embedded methods in which feature selection methods are built in. Embedded methods are highly accurate, generalize better and are interpretable. RF algorithms are supervised learning algorithms that can be utilized for both regression and classification. They are easy to use and flexible [15].

The following steps highlight how feature selection is performed utilizing RF classifier.

- i. Create a RF classifier.
- ii. Obtain the scores (Gini importance) for each attribute. The importance scores of all the attributes should add up to 100%.
- iii. Selector object is constructed that uses the RF classifier to recognize the important features having an importance greater than the specified threshold value of 0.15.
- iv. The test and training datasets are then transformed to hold only the important features that were selected.

The selected features are displayed in Table 1.

Initially, the dataset is comprised of 22 features. After feature selection, 14 features were selected using the feature importance concept of random forest. The 14 attributes that were selected by random forest were: gender, caste, SSLC percentage (10%), PUC percentage (12%), internal assessment, end semester, back status, town or village, admission category, FQ, MQ, SH, school type, medium of education and TT as depicted in Table 1.

4. Train RF classifier: The RF classifier represents an ensemble learning technique that works by constructing multiple decision trees during training time and outputs classes for classifications or real values for regression. The predicted values outputted by individual trees are then averaged. The RF classifiers overcome the over-fitting of training data that happens when individual decision trees are utilized.

Table 1 Selected features

Attribute	Description	Values
GE	Gender/sex	Female, Male
CST	Caste	ST, MOBC, General, SC, OBC
TNP	Class 10th percentage	Best (≥ 80), Very Good ($\geq 60 \&& < 80$), Good ($\geq 45 \&& < 60$), Pass ($\geq 30 \&& < 45$), Fail (< 30)
TWP	Class 12th percentage	Best (≥ 80), Very Good ($\geq 60 \&& < 80$), Good ($\geq 45 \&& < 60$), Pass ($\geq 30 \&& < 45$), Fail (< 30)
IAP	Internal assessment average	Best (≥ 80), Very Good ($\geq 60 \&& < 80$), Good ($\geq 45 \&& < 60$), Pass ($\geq 30 \&& < 45$), Fail (< 30)
ESP	End semester percentage	Best (≥ 80), Very Good ($\geq 60 \&& < 80$), Good ($\geq 45 \&& < 60$), Pass ($\geq 30 \&& < 45$), Fail (< 30)
ARR	Whether the student has back or not	Yes, No
AS	Admission category	Free, Paid
FMI	Family month income	Very High, Medium, High, Above, Low
FQ	Father qualification	Illiterate, Under Class 10, 12, Degree, Post Graduate
MQ	Mother qualification	Illiterate, Under Class 10, 12, Degree, Post Graduate
SH	Study hour	Good, Average, Poor
TT	Time travel	Large, Small, Average
ATD	Class attendance percentage	Good, Average, Poor

The major benefit of RF classifiers is that they can be employed for both regression and classification problems that constitute most ML systems. In this work, RF is used for predicting the student's grade. In our work, several hyper-parameters of RF such as predictive power, maximum-features, minimum-leaf-nodes, speed and out-of-bag (oob) sampling score were varied. The DTs were varied between 10 and 100 in order to improve the forecasting power of the RF classifier. Generally, the performance of RF classifier improves with the increase in the number of DTs. However, it can also slow down computation and after a certain level, the performance starts to decrease. In our work, for 30 trees better performance was achieved.

The second hyper-parameter that was varied was the "max_features". It represents the maximum number of attributes that a RF classifier considers while partitioning a node. The "min_sample_leaf" was the third hyper-parameter that was varied. It determines the least number of leaf nodes required for partitioning an internal node.

The model speed can be increased by increasing the n_jobs parameter. The parameter tells the engine how many processors it is allowed to use. If it has a value of 1, it can only use one processor. A value of "-1" means that there is no limit. The

Table 2 Confusion matrix

Evaluation criteria	SVM classifier	RFC classifier
Correctly classified instances	22	15
Prediction accuracy (%)	79	94

“random_state” makes the model’s output replicable. That is, the model generates same results for same hyper-parameters on the same training data.

The “oob_score” method or “oob_sampling” method is a RF cross validation technique that we utilized in our work. In this method, about 1/3rd of data was utilized to estimate the model’s performance without considering it for training. The test samples are called as out of bag samples. It differs from leave-one-out method in that it suffers a reduced amount of overhead in computation.

4 Experimental Results

This section describes the experimental results that were obtained along with the comparison of methods. The implementation was performed using Python and PyCharm on Windows 10 platform. We have compared the RF classifier with the SVM model. Table 2 below shows the confusion matrix.

From Table 2, we deduce that the RF classifier achieved considerable improvement of 15% over the support vector machine. The two algorithms provide different accuracy level, i.e., each of them interprets the relevance of attribute in a diverse manner. There are evaluation criteria based on classification algorithm used.

5 Conclusion

This work for predicting the student performance using ML techniques is developed with the intention to automate the student results prediction process. Our work uses the ML algorithm called the RF classification algorithm. The RF algorithm has been utilized in many real-world scenarios because of their ability to solve problems with diverse ease of use and the model-free property.

The outcome of this work shows that we can predict the student performance to various classes. We have concluded that RF algorithm generates the best prediction result. The results are achieved by applying RF algorithm. The result indicates that RF algorithm produces the best prediction as compared to SVM algorithm. The proposed methodology can be adopted to help the teacher as well as the student to enhance the quality of learning. The model will allow the lecturers to take early actions to help and assist the poor and average category students to improve their results.

In future, this work can be extended by using a user interface and larger dataset and extracting a greater number of features to provide higher level of accuracy. A mobile application can also be built for the same.

References

1. Ahmed, A.B.E.D., Elaraby, I.S.: Data mining: a prediction for student's performance using classification method. *World J. Comput. Appl. Technol.* **2**(2), 43–47 (2014)
2. Altaher, A., BaRukab, O.: Prediction of student's academic performance based on adaptive neuro-fuzzy inference. *Int. J. Comput. Sci. Netw. Secur.* **17**(1), 165–169 (2014)
3. Acharya, A., Sinha, D.: Early prediction of student performance using machine learning techniques. *Int. J. Comput. Appl.* **107**(1), 37–43 (2014)
4. Kaur, P., Singh, M., Josan, G.S.: Classification and prediction-based data mining algorithms to predict slow learners in education sector. *Procedia Comput. Sci.* **57**, 500–508 (2015)
5. Guruler, H., Istanbullu, A., Karahan, M.: A new student performance analyzing system using knowledge discovery in higher educational databases. *Comput. Educ.* **55**(1), 247–254 (2010)
6. Vandamme, J.P., Meskens, N., Superby, J.F.: Predicting academic performance by data mining methods. *Educ. Econ.* **15**(4), 405–419 (2007)
7. Abuteir, M., El-Halees, A.: Mining educational data to improve students' performance: a case study. *Int. J. Inf. Commun. Technol. Res.* **2**, 140–146 (2012)
8. Baradwaj, B.K., Pal, S.: Mining educational data to analyze students performance. *Int. J. Adv. Comput. Sci. Appl.* **2**(6), 63–69 (2011)
9. Priya, K.S., Kumar, A.V.S.: Improving the student's performance using educational data mining. *Int. J. Adv. Netw. Appl.* **4**(4), 1680–1685 (2013)
10. Cortez, P., Silva, A.: Using data mining to predict secondary school student performance. In: Brito, A., Teixeira, J. (eds.) 5th Future Business Technology Conference (FUBUTEC 2008), pp. 5–12 (2008)
11. UCI repository. <https://archive.ics.uci.edu/ml/datasets/student+performance>. Last accessed 2019/3/24
12. VanderPlas, J.: Python data science handbook essential tools for working with data. O'Reilly Media, pp. 541 (2016)
13. Banu, A.K.S., Ganesh, S.H.: A study of feature selection approaches for classification. In: International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), pp. 1–4. Coimbatore (2015)
14. Jaiswal, J.K., Samikannu, R.: Application of random forest algorithm on feature subset selection and classification and regression. In: World Congress on Computing and Communication Technologies (WCCCT), pp. 65–68. Tiruchirappalli (2017)
15. Paul, A., Mukherjee, D.P., Das, P., Gangopadhyay, A., Chintha, A.R., Kundu, S.: Improved random forest for classification. *IEEE Trans. Image Process.* **27**(8), 4012–4024 (2018)

Detection of A Stable Age in Children for Face Recognition Application



R. Sumithra, N. Vinay Kumar, and D. S. Guru

Abstract In this work, a novel approach for finding an accurate age of a child to be adapted for an automatic face recognition application is proposed. Our proposed approach initially uses Viola–Jones method to automatically locate a key point on facial components namely nose, eyes and mouth. We extract the intensity value in the region across these key points, find the row mean intensity value in the region of each component and sorted. Then, the corresponding index value of the sorted elements are preserved and represented as a Feature Vector (FV). By using the FV of each component, we find the correlation coefficient between two consecutive ages of a child. Then, the First Order Difference (FOD) of correlation coefficient of two consecutive ages is computed. The Cumulative Histogram (CH) of FOD is estimated and average CH of all components is determined. If the difference between average CH of two consecutive ages is negligibly small, then we considered that age as stable age for face recognition application. The proposed model has been validated on a reasonably sized dataset, which is created during this work. It consists of totally 421 longitudinal face images of 32 children each having 12–15 years face images, single sample per year. Kappa coefficient is used to measure the goodness of the model and results with 50% accuracy.

Keywords Longitudinal face images · Stable face recognition · Facial components · Viola–Jones algorithm

R. Sumithra (✉) · N. Vinay Kumar · D. S. Guru

Department of Studies in Computer Science, University of Mysore, Manasagangotri, Mysuru 570009, India

e-mail: sumithraram55@gmail.com

N. Vinay Kumar

e-mail: vinaykumar.natraj@gmail.com

D. S. Guru

e-mail: dsg@compsci.uni-mysore.ac.in

1 Introduction

Biometrics is the science of establishing the identity of a person based on physical or behavioral attributes such as fingerprint, face, iris, signature and voice [1]. Among these modalities, face has been a choice of consideration because human face images are useful not only for person recognition but also for revealing other attributes such as gender, age, ethnicity and emotional state of a person. The success of a face recognition system largely depends on the stability of a face. Any highly sophisticated and efficient face recognition system fails if the parameters of a face change regularly [2].

Among several issues, aging variation is now receiving a greater attention by the research community as it is an irreversible process and with the progress of age, the appearance of human faces change remarkably [2]. It refers to changes in a biometric trait over a time span, which can potentially impact the accuracy of a biometric system. Aging with respect to facial recognition system includes variation in shape, size and the texture of the face. As the face changes over time, the ability to recognize a person becomes more challenging. These temporal changes will cause performance degradation.

Sparse Representation Coding (SRC) [3] and face recognition based on deep learning [4] are some of the most notable advances in the area of facial aging recognition for the last decade. Some of the standard aging face models are anthropometric model [5], active appearance model [6], aging pattern subspace [7], and aging manifold [8]. Some of the typical approaches for facial aging identification are Orthogonal Locality Preserving Projection (OLPP) [9], Conformal Embedding Analysis (CEA) [10], Locally Adjusted Robust Regression (LARR) [11], and Synchronized Submanifold Embedding (SSE) [12]. Rowden et al. [13] experimented on a longitudinal study on automatic face recognition and concluded that proper statistical analysis is very essential for studying face recognition performance over long period of time. Though, there are only few attempts toward the development of facial aging recognition are found, but all of them concentrate only on adults. Facial aging needs to be studied at a continuous interval of time beginning with a new born baby to effectively study the varying parameters of a human face, their stability and the age group at which they get stabilized. Rowden et al. [14] created longitudinal face image of newborns, infants and toddlers in the period of one year with different interval of time stating that face recognition technology is not yet ready to reliable recognition of very young children at the age of about 3 years. In facial aging recognition system, one of the biggest issues is the availability of vast amount of data required to fully understand the human face and its maturation process. Some of the publically available dataset for the applications of accurate face recognition system, includes the face images of children are FG-NET database, which contains 82 subjects in total, and only 6 (out of which 3 male and 3 female) have face images at ages younger than 16 years old. Face Tracker database has only one image per child, Cross Age Celebrity Database (CACD) does not contain any subjects younger than 10 years

old, and In the Wild Child Celebrity (ITWCC) database have average age of the first image of each subject is 10 years.

From the literature, it is understood that, any face recognition model demands stability of a face for its consistent performance. There are few works in adult face recognition through aging with relatively good result, but we find a very few works on child face recognition through aging. Due to the lack of availability of a child face dataset and the drastic changes in a growth rate of a child face, child stable age identification through face images is still untouched area in face recognition system. In other words, the question, what should be the minimum age of a child for any face recognition system to work reliably and consistently? Has not been answered by any method available in the literature. To this consequence, we address the problem on predict an age of a child at which its facial biometric becomes stable so that it could be recommended for any biometric application.

In this work, we attempt on predicting an age of a child at which its facial biometric becomes stable so that it could be recommended for any biometric applications. We utilized the Viola–Jones algorithm to automatically detect the key points of facial components such as eyes, nose, and mouth. Further, the regions around each detected face components are extracted to study the changes in the intensity distribution across the different ages of a child. Subsequently, the row mean intensity distributions are sorted and its corresponding abscissa indexes values are preserved and represented in a Feature Vector (FV). The FV of each component is utilized to find correlation coefficient between two consecutive ages of a child. Correlation coefficient is used to measure the relationship between two FV. First Order Difference (FOD) is computed using the correlation coefficient of two different ages of a child. Then, the stable age is detected through difference of two consecutive ages. If the difference between two consecutive ages is negligibly small, then that age is considered as a stable age. To measure the goodness of our proposed model we created our own dataset of face images of 32 children starting from the age of 1–15 years, with single sample per year.

The following are the contributions in this work:

- Creation of our own longitudinal dataset of face images of children of age from 1 to 15 years.
- Exploring the appearance based features for identification of a stable age of children.
- Identification of an approximate age of a child's face available for biometric application.

The organization of the paper is as follows: Sect. 2 presents the details of proposed model. Experimental results and analysis are presented in Sect. 3. Finally, conclusion and future enhancement are given in Sect. 4.

2 Proposed Model

The proposed model has stages such as preprocessing, feature extraction, representation and decision making. The general architecture of the proposed model is shown in Fig. 1.

2.1 Preprocessing

Let us consider a face image of size $n \times n$ which is scanned photographs, hence some manual adjustments are essential. The face part image shown in Fig. 2a is cropped by using paint tool and by physically aligning the images through gridlines, we rotated the image to alter the skewness present in an image with the goal that two eyes are aligned to be horizontally straight and resized to $n' \times n'$ (where $n > n'$) as shown in Fig. 2b.

2.2 Feature Extraction

Generally, any face image is distinguished by its unique identity through its facial components such as eyes, nose and mouth. These facial components play a major

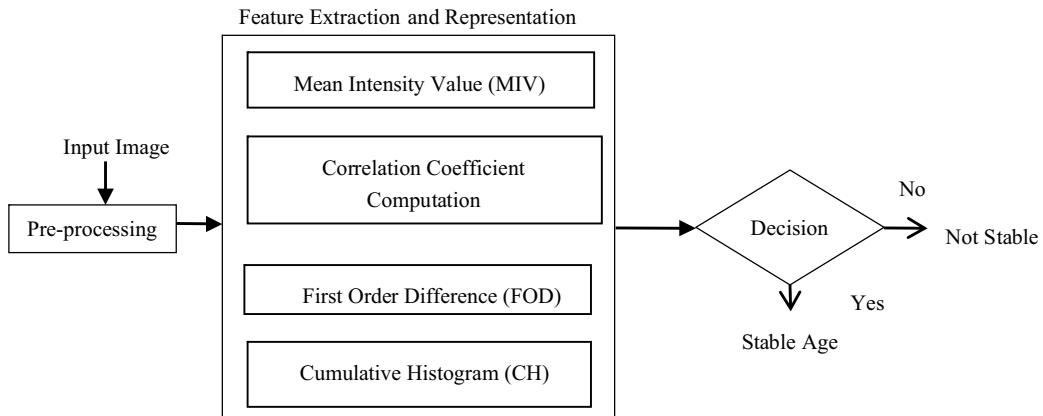


Fig. 1 General architecture of the proposed model

Fig. 2 Illustration of preprocessing **a** sample scanned child image; **b** preprocessed image



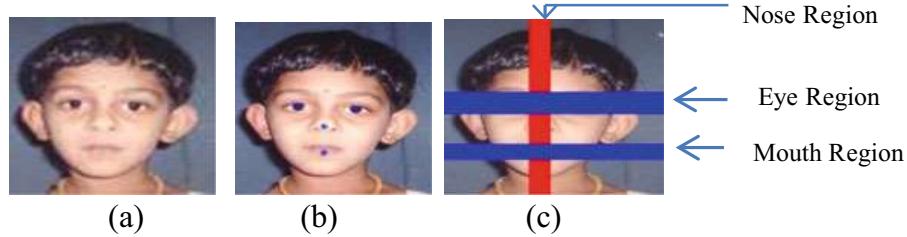


Fig. 3 Illustration of feature extraction **a** preprocessed image; **b** detected facial key points; **c** facial components region are extracted

role in discrimination from one age to the other age based on the appearance of the face, while comparing with other components over the face.

2.2.1 Key Point Detection

Firstly, we estimate the variations in each facial component such as eyes, nose and mouth; we determine key points on each facial component by using Viola and Jones [15] algorithm. The algorithm automatically places a bounding box (rectangular window) to the region of each component. Then, the centroid of the bounding box to plot the key points are shown in Fig. 3b. Further, to study the intensity variations across the ages associated with eyes and mouth component, we created a rectangular window on the center of the key point from one end to other end of an image varying from 1 to L (where L is an abscissa index) and height $+c$ and $-c$ (where c is a constant value) from the key points as shown in Fig. 3c. Same procedure is applied to extract nose component region with rectangular window of width size $+c$ and $-c$ from the key point and height from top to bottom of an image varying from 1 to L (where L is an ordinate index) corresponding to an age of a child. Same procedure is applied on images of all the ages of each child.

2.2.2 Mean Intensity Values (MIV)

The Mean Intensity Value (MIV) is computed associated with each facial components viz., eye (horizontal), mouth (horizontal), and nose (vertical) derived by Eq. (1).

$$\text{MIV}_h^l = \{\text{MIV}_h^1, \text{MIV}_h^2, \text{MIV}_h^3, \dots, \text{MIV}_h^L\} \quad (1)$$

$$\text{Where, } \text{MIV}_h = \frac{1}{2C+1} \sum_{i=1}^{2C+1} X_i$$

L = Level of the abscissa index; X_i = intensity value of a pixel; $2C + 1$ = height and width w.r.t the vertical region (eye and mouth region) and horizontal region (nose region) and C is a constant; $h = 1, 2, \dots, M$ (M = Maximum age of a child); $l = 1, 2, 3, \dots, L$.

We analyze this intensity distribution by sorting the mean intensity values and preserving their corresponding indices in a vector called Feature Vector (FV), for all individual face components across the different ages of a child. Similarly, for mouth and nose components FV extracted using the same technique for preserving each facial component.

2.2.3 Correlation Coefficient Computation

Further, we studied the correlation among the Feature Vector (FV) of different ages corresponding to each facial component of a child. Thus, resulting with correlation coefficient matrix of size $M \times M$. The correlation between two feature vectors is given by:

$$\text{CM}(\text{FV}_p, \text{FV}_q) = \frac{\text{Covar}(\text{FV}_p, \text{FV}_q)}{\sqrt{\text{Var}(\text{FV}_p)} * \sqrt{\text{Var}(\text{FV}_q)}} \quad (2)$$

$$\text{Where, } \text{Covar}(\text{FV}_p, \text{FV}_q) = \frac{1}{L} \sum_{\forall i=1}^L (((\text{FV}_p)_i - \overline{\text{FV}}_p) * ((\text{FV}_q)_i - \overline{\text{FV}}_q)) \quad (3)$$

$$\text{Var}(\text{FV}_p) = \frac{1}{L} \sum_{\forall i=1}^L ((\text{FV}_p)_i - \overline{\text{FV}}_p)^2 \quad (4)$$

$$\text{Var}(\text{FV}_q) = \frac{1}{L} \sum_{\forall i=1}^L ((\text{FV}_q)_i - \overline{\text{FV}}_q)^2 \quad (5)$$

$$\overline{\text{FV}}_p = \frac{1}{L} \sum_{i=1}^L (\text{FV}_p)_i; \quad \overline{\text{FV}}_q = \frac{1}{L} \sum_{i=1}^L (\text{FV}_q)_i \quad (6)$$

where CM = Correlation coefficient Matrix; $p = 1, 2, \dots, M$ and $q = 1, 2, \dots, M$ and M is the maximum age of a child. L is the total number of abscissa index.

The correlation coefficient value between two consecutive ages of a child is selected with the help of correlation coefficient matrix using Eq. (2). Further, the absolute difference between two correlation coefficients is resulted with a First Order Difference.

2.2.4 First Order Difference (FOD) and Cumulative Histogram (CH)

The First Order Difference (FOD) helps to find a higher average response to changing in intensity [16]. In this work, FOD is used to analyze the changes in the growth rate between each age of a child (Eq. 7) more efficiently. Further, we compute the Cumulative Graph of FOD corresponding to each component to detect the stable

age of a child. The Cumulative Graph (CG) is defined as the sum of FOD of two consecutive ages of a child and is given in Eq. (8).

$$\text{FOD} = (|\text{Correlation Coefficient}_x - \text{Correlation Coefficient}_{x+1}|) \quad (7)$$

$$\begin{aligned} \text{CG}_1 &= \text{FOD}_1 + \text{FOD}_2; \text{CG}_2 = \text{FOD}_2 + \text{CG}_1; \text{CG}_3 \\ &= \text{FOD}_3 + \text{CG}_2; \dots \text{CG}_n = \text{FOD}_n + \text{CG}_{n-1}; \end{aligned} \quad (8)$$

where FOD is First Order Difference, CG is Cumulative Graph and n is the maximum age of a child.

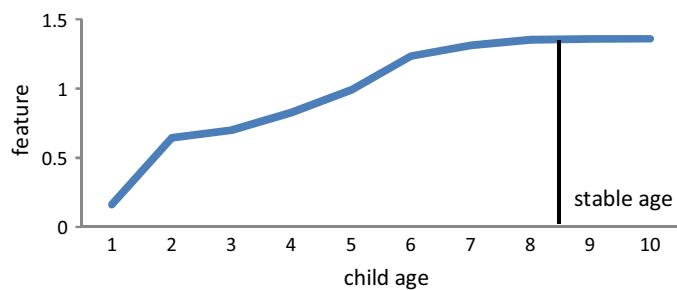
2.3 Decision Making

We computed average Cumulative Graph (CG) from each component such as CG of nose, CG of mouth and CG of eye using Eq. (9), to identify the stable age of a child. That is, the difference between average CG_x and average CG_{x+1} (where x is the age of a child) must be negligibly small (or equal to zero), then x is said to be a stable age of a child used for biometric application. Figure 4 shows the illustration of detection of stable age of a child for face recognition application using function Stable Age Detector (SAD).

$$\text{Average CG} = \frac{(\text{CG})_{\text{NOSE}} + (\text{CG})_{\text{MOUTH}} + (\text{CG})_{\text{EYE}}}{3} \quad (9)$$

$$\text{SAD}(\text{child}) = \begin{cases} \text{Stable age if } (|\text{Average CG}_x - \text{Average CG}_{x+1}| \approx 0) \\ \text{otherwise Not stable } < \varepsilon \end{cases}$$

Fig. 4 Cumulative graph of first order difference



3 Experimental Results and Analysis

The proposed model has been validated on a reasonably sized dataset created during this work. In this section, we present details on creation of the dataset and analysis of the experimental results.

3.1 Longitudinal Face Database

Dataset plays a major role in face recognition, the creation of a suitable dataset is crucial for any machine learning application [17] as the dataset is a primary requirement for designing and testing the system. Creating such a dataset is also a key contribution to this domain of research. In this work, we created our own dataset consisting of 421 scanned images of 32 children longitudinal face images from the age of 1–15 years photographs in the regular interval of an image per year. The photographs of children were collected from all possible sources with the only constraint that the child should be facing the camera and with neutral expression wherever possible. The different events or sources such as birthday, trip, school ID, school annual day etc., were considered during dataset collection. Collecting photos of a child from age 1 year till 15 years at a regular interval of one year is a denting task. So, we decided to collect minimum age from 1 to 12 years and maximum age from 1 to 16 year’s children photographs. We collected both hard copies and soft copies of photos. Totally, 421 images of 32 children out of which 14 are male and 18 are female. Table 1 gives the details of on this dataset. In Fig. 5 some samples images collection are shown.

We scanned the children photographs with 1200 dpi resolution by adjusting area dimensions: width and height, and cropped face part only. We use HP laser Jet professional M1136 MFP scanner machine for scanning. Both hard copies and soft copies of photographs were accepted. The most of the images were collected by scanning photographs of children found in personal collections. So the quality of the images depends on the skills of the photographer, the quality of image equipment used, the quality of photographic paper and printing and the condition of photographs. Therefore, in the dataset there is a considerable variability in resolution, quality, illumination, viewpoint and expression.

3.2 Experimental Setup

We have conducted experimentation on 421 longitudinal face images of 32 children each with 12–15 samples, single sample per year. During experimentation, we cropped, rotated and resized the images to 1000×1000 . The algorithm exactly points to the two eyes center, nose center and mouth center automatically. For illustration, we have given same example images with plotted key points in Fig. 6.

Table 1 Datasets description

Child number	Gender	No. of images	Age interval	Child number	Gender	No. of images	Age interval
child1	Male	20	[1–20]	child 17	Female	12	[1–12]
child 2	Female	17	[1–17]	child 18	Female	12	[1–12]
child 3	Female	17	[1–17]	child 19	Female	12	[1–12]
child 4	Female	16	[1–16]	child 20	Female	12	[1–12]
child 5	Female	16	[1–16]	child 21	Female	12	[1–12]
child 6	Male	16	[1–16]	child 22	Male	12	[1–12]
child 7	Female	15	[1–15]	child 23	Male	12	[1–12]
child 8	Female	15	[1–15]	child 24	Male	12	[1–12]
child 9	Female	15	[1–15]	child 25	Male	12	[3–12]
child 10	Male	15	[1–15]	child 26	Male	12	[3–12]
child 11	Female	14	[1–14]	child 27	Male	12	[1–12]
child 12	Female	14	[1–14]	child 28	Female	12	[1–12]
child 13	Female	13	[1–13]	child 29	Female	12	[1–12]
child 14	Male	13	[1–13]	child 30	Male	12	[1–12]
child 15	Male	13	[1–13]	child 31	Male	12	[1–12]
child 16	Female	12	[1–12]	child 32	Male	12	[1–12]

**Fig. 5** Sample face images of children collected during the course of this research work**Fig. 6** Results of plotted key points of each component using Viola–Jones algorithm

As explained in Sect. 2.2, we extracted the intensity value in the rectangular region across the key points where height is 101 such that +50 and –50 from the center of the rectangular box, and width is 1000, the size of rectangular box is 1000×101 (height \times width). Figure 7 shows the horizontally extracted intensity values of eye region from one end to other end of a face image, similarly extracted for mouth region, and the vertically extraction of intensity value of nose region from top to bottom of a face image respectively. Then using Eq. (1), the mean intensity value of each component is computed which is of size of 1000×1 , as explained in Sect. 2.2.



Fig. 7 Illustration of mean intensity value across eye region, mouth region and nose region

We sorted the mean intensity values corresponding to intensity level $L = 1000$ for each facial component. Then, we preserve the corresponding indexing value of mean intensity value of each facial component in a Feature Vector (FV). The correlation coefficient is calculated from the FV of different ages corresponding to three facial components of a child, using Eq. (2). Correlation coefficient is represented in the matrix of size 15×15 (where, 15 is the maximum age of a child) individually for three facial components. Then, the absolute difference between two correlation coefficients gives FOD, with matrix of the size of 14×14 , which is used to analyze the changes in the growth rate between each age of a child more efficiently.

The cumulative Graph (CH) of FOD of each component of two consecutive ages is calculated and its results are shown in Fig. 8. Figure 9 shows the plots of mean values of all components for 2 children.

The difference between averages CG of successively age equal to zero, then the associated age is said to be a stable age of a child used for biometric application.

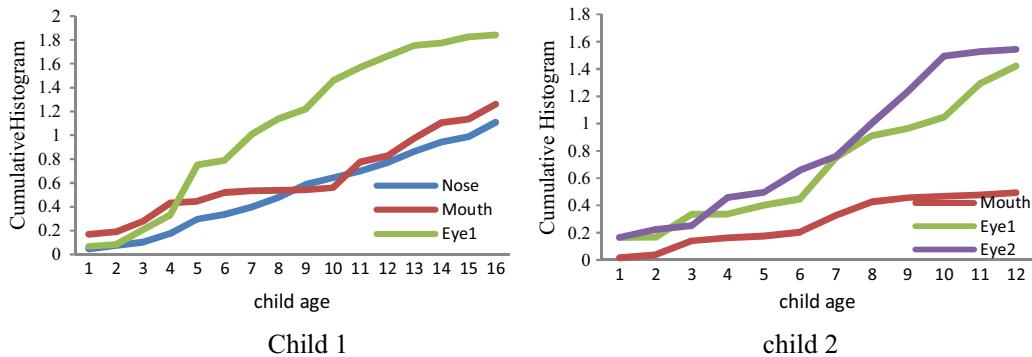


Fig. 8 Results from cumulative graph of three children with different facial components

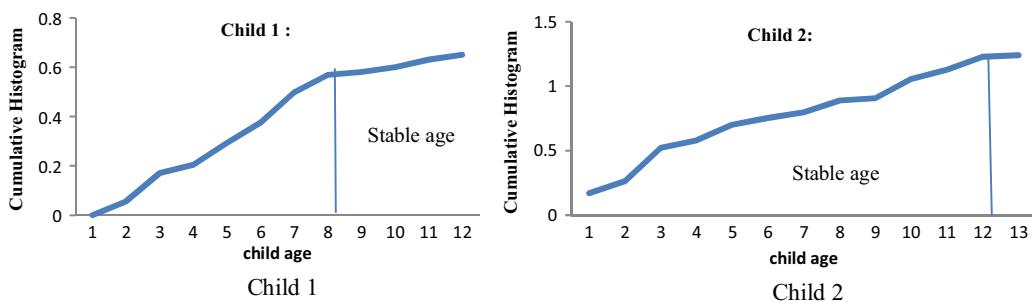


Fig. 9 Average values of all facial components of CG

3.3 Evaluation

With the help of 10 human experts, we created a Ground Truth (GT) for 10 children to measure the efficiency of the proposed model. During GT creation, we requested 10 people whose are above 25 years old, intellectual intelligence and enough matured for identifying the age of a child to estimate the high rate of similarity between two consecutive ages so, we called them as human experts. We provided all 10 children photographs from age 1–15 years, to human experts for analysis.

Then, to find the validity and the goodness of the model, we used kappa coefficient measures [18]. The technique of measuring the disagreement between GT and the obtained system results is called kappa coefficient shown in Eq. (10).

$$\text{Kappa coefficient } (K) = \frac{P_o - P_e}{1 - P_e} \quad (10)$$

$$P_o = \sum_{i=1}^n p_{ii} \quad (11)$$

$$P_e = \sum_{i=1}^n p_{i.} \cdot p_{.i} \quad (12)$$

where P_o is the relative observed agreement among the observer and we calculated by the sum of the principle diagonal elements of each observer using Eq. (11). P_e is the hypothetical probability of change in agreement between the observers and we calculated using Eq. (12) the sum of product $P_{i.}$ and $P_{.i}$, where $P_{i.}$ is the sum of all rows of the observer i and $P_{.i}$ is the sum of all column of the observer i , which is illustrated in Table 2.

Table 2 Performance of the proposed model in comparison with ground truth

Child	Proposed model (stable age)	Ground truth (stable age)
Child 1	8	8
Child 2	9	8
Child 3	9	9
Child 4	8	8
Child 5	9	9
Child 6	6	8
Child 7	10	10
Child 8	10	8
Child 9	8	9
Child 10	8	8

From Eqs. (12) and (13), we get P_o is 6, P_e is 32.28 and Kappa coefficient (K) is 0.85, which is obtained from Eq. (11). The results of Kappa coefficient the disagreement between each observer is 0.85 (85%) so, the agreement is 0.15 (15%). We calculated the accuracy of 10 children from the obtained results and GT, from the 10 human expert's observation. We find the most frequently occurring stable age of every child from all observers. For five children stable ages are exactly match with the system result out of 10 (50% accuracy).

4 Conclusion and Future Enhancement

In this work, we had attempted on the child stable age identification for face biometric application. Toward this work, we used Viola–Jones algorithm to automatically extract a key points from the facial components and extracted the intensity value from the entire region of the key points. The mean intensity values of each component are sorted and their corresponding indices are represented in Feature Vector (FV) then the correlation coefficient is calculated. The cumulative histogram of the first order derivative from the correlation coefficient matrix is minimized, when age increases sequentially. Kappa coefficient is used to measure the goodness of the model and results with 50% accuracy.

Though our created dataset is quit less in number, for applying the learning models, but with the help of data augmentation it is possible. In future, we suppose to test our dataset for standard state-of-the-art methods for both feature extraction like LBP, SURF, HOG etc., and learning models like SVM, LDA, Neural Network, etc.

Acknowledgements We are grateful to the children who participated in these studies and to their parents and teacher. We thank Mrs. Suneetha madam the teachers of Gopalswami Shishuvihara, Mysuru, for their cooperation in collecting the data. The work by Sumithra R was financially supported by Rajiv Gandhi National Fellowship (RGNF) under University Grant Commission, Government of India, and India.

Undoubtedly, developments in the platforms for the collection and subsequent analysis of data have some clear benefits to children, relating to their protection, their safety and their participation within the global community.

References

1. Jain, A.K., Nandakumar, K., Ross, A.: 50 years of biometric research: accomplishments, challenges, and opportunities. *Pattern Recognit. Lett.* (2016) <https://doi.org/10.1016/j.patrec.2015.12.013>
2. Jain, A.K., Ross, A., Nandakumar, K.: *Introduction to Biometrics*. Springer, Berlin (2011)
3. Wright, J., Yang, A.G., Sastry, S., Ma, Y.: Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**, 361–369
4. Sun, Y., Wang, X., Tang, X.: Deep learning face representation from predicting 10,000 classes. In Proceedings of the IEEE Conference on Computer Vision and PR, pp.1891–1898

5. Sohail, A.S., Bhattacharya, P.: Detection of facial feature points using anthropometric face model. Concordia University, 2-9525435-1 © SITIS, pp. 656–664 (2006)
6. Edwards, G.J., Cootes, T.F., Taylor, C.J.: Face recognition using active appearance models. In: Image Analysis Unit, Department of Medical Biophysics, University of Manchester, Manchester M13 9PT, U.K 1997
7. Geng, X., Zhou, Z.H., Smith, M.K.: Automatic age estimation based on facial aging patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(12), 2234–2240 (2007)
8. Alex, P., Moghaddam, B., Starner, T.: View-based and modular eigen spaces for face recognition. In: M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 245, IEEE Conference on Computer Vision & Pattern Recognition (1994)
9. He, X., Cai, D.: Orthogonal locality preserving indexing. ACM 1-59593-034-5/05/0008. SIGIR'05. Salvador, Brazil, 15–19 Aug (2005)
10. Fu, Y., Liu, M., Huang, T.S.: Conformal embedding analysis with local graph modeling on the unit hyper sphere. In: IEEE Conference CVPR'07, Workshop on Component Analysis (2007)
11. Guo, G., Fu, Y., Charles, R.D., Thomas, S.H.: Image-based human age estimation by manifold learning and locally adjusted robust regression. *IEEE Trans. Image Process.* **17**(7) (2008)
12. Yan, S., Wang, H., Fu, Y., Yan, J., Tang, X., Huang, T.S.: Synchronized sub manifold embedding for person-independent pose estimation and beyond. *IEEE Trans. Image Process.* **18**(1) (2009)
13. Best-Rowden, L., Hoole, Y., Jain, A.: Automatic face recognition of newborns, infants, and toddlers: a longitudinal evaluation. In: 2016 International Conference of the Biometrics Special Interest Group (BIOSIG). IEEE (2016)
14. Best-Rowden, L., Jain A.K.: Longitudinal study of automatic face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* (2017)
15. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, pp. 511–518 (2001)
16. Gonzalez, R.C., Woods, R.E.: Digital image processing, 3rd edn, pp. 104–105. Publishing House of Electronics Industry, Beijing China (2010)
17. Quionero-Candela, J., et al. Dataset Shift in Machine Learning. MIT Press, Cambridge, MH (2009)
18. Cohen, J.: Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychol. Bull.* **70**(4), 213 (1968)

An Automatic Predictive Model for Sorting of Artificially and Naturally Ripened Mangoes



Anitha Raghavendra, D. S. Guru, and Mahesh K. Rao

Abstract There is a fact that mangoes are being ripened using artificial ripening agents like—ethrel (ethenol) and calcium carbide and a well-known carcinogenic; consuming such mangoes might cause cancer. In this paper, a model for classification of artificially and naturally ripened mangoes based on their images using machine learning techniques is proposed. An extensive experimentation has been conducted on the dataset consists of 2440 images based on two varieties of mangoes locally termed as badami and raspuri, which were naturally and artificially ripened. Artificially ripened was further classified based on the artificial ripening agent—calcium carbide or ethrel solution. The model is based on the extraction of best color feature, best texture feature and also their fusion. To express the efficacy of this model, the best color and texture features are chosen based on the highest match of results which were obtained from five well-known classifiers like LDA, Naïve Bayesian, kNN, SVM and PNN. The obtained accuracy was between 69.89 and 85.72% in all categories by choosing the best color features. Similarly, best texture features gave an accuracy between 75 and 79%. It has also been observed that accuracy decreases as the fusion of color and texture features is chosen for classification.

Keywords Badami · Raspuri · Calcium carbide · Ethrel (ethenol) · Fusion of color/texture

A. Raghavendra (✉)

Maharaja Research Foundation, Maharaja Institute of Technology, Mysore, India
e-mail: anithbg@gmail.com

D. S. Guru

Department of Studies in Computer Science, University of Mysore, Manasagangotri, Mysore 570006, India
e-mail: dsg@compsci.uni-mysore.ac.in

M. K. Rao

Maharaja Institute of Technology, Mysore, India
e-mail: maheshkrao_ece@mitmysore.in

1 Introduction

With increasing urbanization, the needs of the societies are changing and have a bearing on overall health status of the population. This has resulted in people leaning toward more nutritional choices such as fruits. Mango is one such fruit contains high level of fiber, vitamin C and pectin which makes it a perfect fruit which helps in controlling high-level cholesterol along with fighting cancer. Mango fruits are antioxidant capsules for properties such as fisetin, astragalin, gallic acid and quercetin. Another benefit of consuming mangoes is that treats pores and gives a glow to your skin; and it cleanses skin from deep inside our body. India is one of the largest mango producers and exporters worldwide. About 57% of worldwide production of mangoes happens in India. Automation of mango classification is considered very essential for wider acceptance of our agricultural produce in the world. To evaluate the quality of fruits, three non-destructive methods can be used such as hyperspectral imaging, near infrared spectroscopy and machine vision [1]. The machine vision applications for the inspection of vegetables and fruits have increased in recent years [2]. Cubero et al. reviewed latest evolution in the application of machine vision technology to the investigation of the external and internal quality of fruits and vegetables [3]. In the literature, there is a plenty of work related to grading and sorting of mangoes. Mango fruit sorting has been done on the basis of their shape, size, maturity level and external defects. Shape is a primary quality factor to be examined by consumer while purchasing mango fruit. Mangoes have been sorted based on the shape features (region and contour) using Fourier transform [4] and have been sorted based on its shape using Hue moments and graded based on defective surface area [5]. Even sorting of mango has been done based on its size estimation; the parameters considered as size metrics are area, diameter, perimeter, standard deviation radius signature, etc. [6]. An optimal threshold was used in segmentation; boundary tracing was used to determine the perimeter [7]. Defect is the second most quality parameter to be taken into consideration by consumer while purchasing mangoes. Mango defects have been identified using wavelet texture features [8], Gabor wavelet features [9] and discrete cosine transform (DCT) features [10]. Mango quality has been graded based on the fruit surface using back-propagation neural network [11]. Mango fruit skin damage has been detected using texture features like color texture moments and fast discrete curvelet transforms [12]. Ripeness evaluation of mango fruit plays an important role during packaging to decide based on shipping time. However, color has useful information which helps to improve the efficiency of ripeness prediction of fruit. A few works have been done on maturity/ripeness classification of mango using different sensors [13] and based on color space statistical feature extraction techniques [14, 15]. An electronic vision-based system for grading and sorting of mangoes based on their size and maturity level, Gaussian mixture model (GMM) and fuzzy logic technique are employed to evaluate the parameters of the individual classes [16]. Along with maturity level, size and surface defect identification on mangoes have been attempted in the same work. So to solve the multicharacteristic problem, multiattribute decision-making theory was adopted [17]. In this brief

survey on mango fruit, it is understood that mango quality has been assessed based on their size, shape, defects and also their maturity or ripeness. Despite this tremendous work on mangoes, a crucial problem still remains unaddressed that is artificially and naturally ripening of mangoes which has major health implications. Fruit ripening is a natural process, however, these days to make the fruits available in abundance and to make it attractive for consumers; fruits are being ripened and/or processed with artificial chemical agents. Artificial ripening is a health hazard as these carcinogens finally make their way into the food chain. Calcium carbide is also used as a ripening agent in some countries such as India, Bangladesh, Nepal, Pakistan even though it is banned. The international and national laws regulations prohibit artificial fruit ripening [18]; hence, there is a decline in export market for Indian mangoes. Curbing and controlling chemical fruit ripening is a complex socioeconomic concern especially in growing countries. It requires combined involvement and commitment of government agencies, fruit-sellers, scientists, farmers, policy-makers and consumers. Food technologist and doctors have cautioned that calcium carbide is dangerous and is unsuitable for consumption in any quantity as it is a known carcinogenic compound. Calcium carbide also has traces of phosphorus hydride and arsenic which have carcinogenic properties [19]. Consumption of artificially treated fruits can affect liver, small intestine, eyes, lungs, along with other neurotoxic effects. On ingesting, it causes vomiting, nausea, headache and burns in the gastrointestinal tract. This neurotoxic chemical causes sleepiness, dizziness, mental confusion, loss of memory, seizures and cerebral edema [20]. As a result of consuming these artificially ripened fruits, health hazards have been rapidly increasing. The humidity in the fruit helps in producing heat, and thus chemical ripens the fruit faster. Fruits which are ripened with calcium carbide are less tasty and too soft. Ethrel solution, also named as ethenol, is another ripening agent which affects human health; this chemical agent is often observed better than calcium carbide because it is not carcinogenic. The fruits which are ripened with ethrel have more admissible color than naturally ripened fruits and have longer shelf-life than fruits which are ripened with calcium carbide [19]. Ethrel is decomposed into ethylene bi-phosphate ion and chloride ion in aqueous solution. Effect of ethrel spray on the ripening behavior of mangoes has been discussed [21].

Identifying chemically treated mangoes may be possible for fruit-sellers, but it is very difficult for a layman due to lack of knowledge as well as identification traits. Hence, this study is aimed at finding a method which can help the common man identify artificially ripened mangoes using image processing techniques. Related works have been discussed below to understand various aspects of artificial ripening of fruits in general. The presence of chemical ripening agent is usually detected by the experts based on the fruit skin which may be in the form of color or texture. Chemically ripened mangoes also have uneven wrinkles on the skin [22]. Bananas which are ripened naturally seem to be dark yellow in color, having small brown/black spots (here and there) on the skin and also stalks being black [23]. Bananas ripened with calcium carbide are lemon yellow in color, they are clear yellow color without any black spots, and their stalks are green. Those ripened with calcium carbide are soft; however, they have good peel/skin color but are poor in flavor. They also have a

shorter shelf-life [24]. Nazibul Islam et al. have analyzed ripening agents and change in nutritional parameters on artificially ripened banana samples. Ravindran et al. [19] have done a comprehensive review on artificial fruit ripening and their types. An artificially ripened fruit has a yellow outer skin, but tissues inside remain raw and green. Developing a system for classification of naturally and artificially ripened mangoes is a difficult task, because of considerable similarities between these two classes as well as due to large intra-class variations shown in Fig. 2. In a real environment, consumers can use their android phone camera to capture mango images and use the developed model being proposed here to know the similarities and variations between two classes. Developing a model which satisfies criteria, e.g., resolution variation from device to device; and the distance variation between the camera and the mango while capturing the image, makes classification even more difficult. In addition, background also makes the problem more difficult as the mango has to be segmented automatically and varying background creates additional challenges. Therefore, classification of naturally and artificially ripened mangoes is quite challenging to achieve the best result. Overall, there is no comprehensive model to detect the artificially ripened mangoes to naturally ripen exists, and this research aims at finding one such model. This paper is organized as follows: Section 2 discusses proposed model which describes the method adopted in our work. Experimentation and results are presented in Sect. 3, and finally, Sect. 4 concludes the paper and discusses directions for future work.

2 Proposed Model

The flow diagram of our model is as shown in Fig. 1. Given a collection of N varieties of mangoes and in each variety considering C number of classes, where each class refers to type of treatment for ripening of mangoes. Out of these classes, one refers to naturally ripened mangoes, while remaining classes are considered to be artificially ripened mangoes. To classify naturally and with one class of chemically treated mangoes, a model is proposed here using the best color feature, best texture feature and their fusion. Let $[S_1, S_2, S_3, \dots, S_n]$ be a set of S samples of a mango class say $C_j; j = 1, 2, \dots, N$ (N denotes number of individuals).

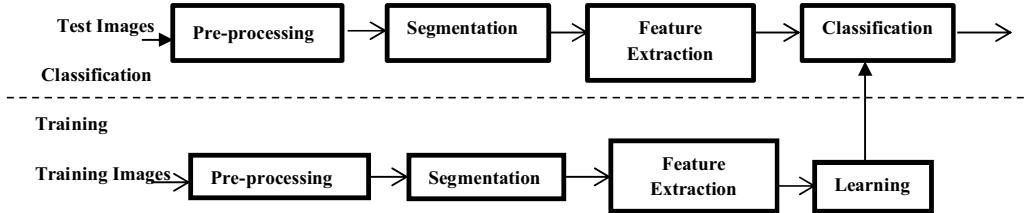


Fig. 1 Flow diagram of our proposed model

For the purpose of feature extraction, let there be k number of color spaces. In each color spaces, there are three color space channels, let I_{ij} be the intensity values in two dimensional space and μ_{sk} ; $k = 1, 2, 3 \dots m$ be the mean of the individual color space channels obtained for all S samples of the class C_j , i.e.,

$$\mu_{sk} = 1/mn \left(\sum_{i=1}^n \sum_{j=1}^m I_{ij} \right), \quad 1 \leq i \leq n \quad \text{and} \quad 1 \leq j \leq m \quad (1)$$

Similarly, let σ_{sk} ; $k = 1, 2, 3 \dots m$ be the standard deviation of the individual k th color space obtained from all the s samples of the class C_j

$$\sigma_{sk} = \left[1/mn \left(\sum_{i=1}^n \sum_{j=1}^m I_{ij} - \mu_{sk} \right)^2 \right]^{1/2} \quad (2)$$

Let there be $F_i = [d_{i1}, d_{i2} \dots d_{im}]$ the set of features of mango sample S_i of the class C_j , where d are the values of mean and standard deviation of samples S_i for k color spaces. As the aim is to calculate accuracy of classifier for various combination of features, accuracy is calculated for P classifiers using individual k color space statistical features. The experimentation is carried out for T trials by changing the training and validation samples. Samples are randomly selected for T trials. Then, the average of results obtained for T trials is calculated to measure the randomness in the samples.

Further in a matrix, each column lists a group of P classifiers output with their corresponding k number of color spaces. Similarly, accuracy is calculated for L combinations of color space features, referred to in Eq. 3.

$${}^k C_L = k!/(k - L)!L! \quad (3)$$

To select the best features, the matrix is built with P classifiers in row, Q individual color space features and their combinations in column, which are derived from Eq. 3. In order to select the best color/texture features which are capable of discriminating natural and artificial ripened mangoes, highest accuracy of a classifier for a particular set of features (column wise) is chosen instead of choosing highest accuracy obtained with respect to particular classifier for all the features (row wise) which is shown in Eq. 5.

Let x_{ij} be the accuracy obtained using P_i classifiers where $i = 1, 2, \dots, n$ and Q_j be the individual color space and their combinations where $j = 1, 2, \dots, m$

$$X = (x_{ij}) \quad 1 \leq i \leq n, \quad 1 \leq j \leq m \quad (4)$$

$$D_j = \max(x_{ij}) \quad 1 \leq i \leq n, \quad (5)$$

where D_j refers to maximum value among all rows in the column j .

The first step toward making the ultimate choice of best features begins with the selection of a classifier that is based on the highest accuracy obtainable in each of the columns. The results of the histogram need to be taken into consideration next. The highest value thus obtained by using the P classifier levels ranging from $[0, G]$ is defined as the discrete function given in Eq. (6).

$$h(r_n) = v_n \quad (6)$$

where r_n is the nth classifier in the interval $[0, G]$ and v_n is the number of maximum accuracies having a classifier result r_n ; v_n is the number of the highest results that can be obtained at nth classifier whose value is r_n . A single classifier can be selected by using the histogram having the highest number of results.

Let Z_j be the number of features, where $j = 1, 2, \dots, m$ which are really dominating for obtained classifier from the histogram shown in Eq. (6). With this, we propose to choose the features based on their highest accuracy values and choosing one set of features on the basis of below Eq. (7).

$$Z = \max(Z_1, Z_2, \dots, Z_m) \quad (7)$$

Now Z can be treated as a best color/texture features. Further fusion of best color and best texture is performed using P classifiers.

3 Experimentation and Results

In this section, the details of the dataset and experiments conducted during the course of research along with the result are presented.

3.1 Dataset Preparation

Since there was no existing database on chemically and naturally treated mangoes, a database for experimental purposes was created. Mango images are captured with five different resolution cameras, viz. 5MP, 8MP, 12MP, 14MP and 20MP by covering all the views of the mango. The reason for choosing different resolution camera is, once the model is built, this model can be used further to develop an android application, so the camera resolution must be independent of the device where different people use different resolution camera. The images have been taken with the three distinct distances which are 12, 14 and 16 cm between the camera and the mango to make the device independent for any distances. The dataset consists of two varieties of mangoes, namely badami (Alphonso) and raspuri. Each variety constituted into three classes based on method of ripening, one class of mangoes were ripened naturally

and remaining two classes were artificially ripened, with calcium carbide and ethrel. Naturally ripened mangoes were ripened with paddy straw bed for a week. We had collected ten mangoes in each class and captured images of all the samples. According to the samples described above, 1320 images were expected in each variety for experimentation, but only 1270 images under badami and 1170 images under raspuri were obtained, since some mangoes were decayed over the period of ripening process. Hence, only good mango images which are 1270 in badami and 1170 in raspuri variety have been considered for experimentation. Developing a model to classify naturally and chemically ripened mangoes in each variety is very challenging due to a less interclass variation and more intra-class variation; to achieve this, only ripened mango images have been captured.

3.2 Preprocessing and Segmentation

The first step in mango classification is to preprocess the mango images. Image normalization is required, as we have captured images from various distances; mango region of interest appears different. To avoid such ambiguity for further processing, image normalization is done using region properties available as libraries in MATLAB. Image normalization is based on connected component and bounding box. It calculates centroids for connected components and forms a bounding box in the image using region properties. Hence, mango region looks common in all the images. Next step is to extract the mango image from its background using segmentation. Segmentation divides an image into component parts. The level to which this segmentation is being carried depends on the problem being solved. In general, segmentation is the most difficult part in image processing, and it plays a very important role in further classification process. Mango region in images is surrounded by white background and also shadow. Ashok et al. [25] have done comparison of K-means clustering and C-means algorithm segmentation technique with their histograms and obtained good results in K-means on apple images. Hence, we decided to use the modified K-means clustering segmentation technique where $K = 2$ for the mango images shown in Fig. 2.

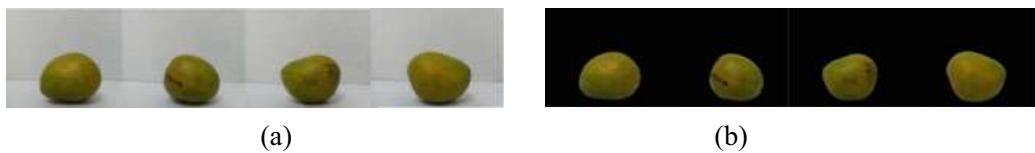


Fig. 2 **a** Mango images before segmentation and **b** after segmentation using K-means clustering

3.3 Experimental Results

For the purpose of assessing the proposed method, accuracy has been adopted. During ripening process of mango, the change in the texture and skin color is more predominant in both artificial and natural ripened mangoes, thus making it difficult to detect those which are artificially ripened to the naked eye. Chemically ripened mangoes also have uneven wrinkles on the skin [23]. As texture and color features are the only properties dominating in the ripened mangoes irrespective of the ripening agent, best color and best texture features have been identified to classify artificial and natural ripened mangoes.

In order to quantify the color present in a mango region, mean and standard deviation from segmented mango regions over individual channels of five different color spaces: RGB(**F1**), CMY(**F2**), YCbCr(**F3**), HSV(**F4**) and CIE L*a*b(**F5**) are extracted. Along with these color features, derived RGB color features RG, GB and BR are also extracted under RGB color space. In order to evaluate the texture present in a mango region, a set of texture features based on the gray-level co-matrix (GLCM-Haralick), local binary pattern (LBP) and Gabor was employed. The problem of chemically and naturally ripened mango classification is large and complex, and it makes sense to first try an extensive method to see what performance can be achieved. Furthermore, five different classifiers which are either statistical classifiers or neural network-based classifiers such as linear discriminant analysis (LDA), Naïve Bayesian (NB), k-nearest neighbor (kNN), support vector machine (SVM) and probabilistic neural network (PNN) are considered in this work. Once features are extracted, a classifier can be trained to classify a test sample as a member of one of the known classes. Experimentation has been conducted on databases of three classes in each variety with varied training samples randomly of percentage of 60. During experimentation, 100 trials are conducted for every set of training and testing samples randomly. Classification accuracies are calculated for the individual and combinational color spaces using Eq. (8).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (8)$$

Accuracy is calculated based on True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). Being interested in understanding suitable features for this problem, a matrix was built in which each row indicates classifiers and each column in turn indicates individual and combinational color spaces. Experimentation has been performed for 31 times based on Eq. (3). Each time best classifier result has been taken into consideration out of five classifiers that give the highest result for a particular set of features (column wise). Under all the classes, the above is depicted in Fig. 3a-d. From Fig. 4, x-axis indicates classifiers and y-axis indicates number of best results obtained out of 31 times for a particular classifiers.

Now selecting only one classifier based on the above histogram shows highest value obtained. Further features are considered which are really dominating for a

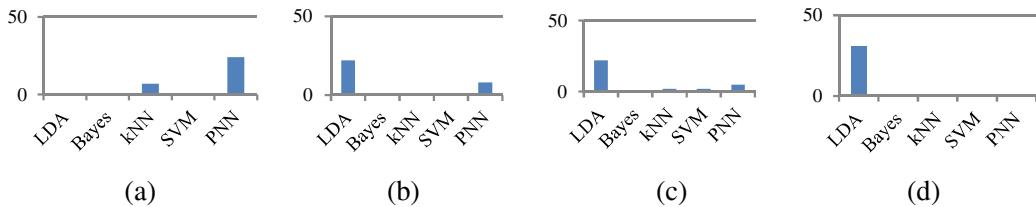


Fig. 3 Frequency of occurrence versus classifiers: **a** badami-natural and calcium carbide; **b** badami-natural and ethrel solution; **c** raspuri-natural and calcium carbide; **d** raspuri-natural and ethrel solution

particular classifier obtained from the histogram. With this, we propose to choose the highest value features based on their accuracy values which are shown in Fig. 4.

From Fig. 4, combination of color spaces features appears to be the best color features for classification of natural and artificially treated mangoes under each category. Results are shown in Table 1.

Similarly, same model has been considered for choosing the best of texture in all categories. In texture feature extraction, GLCM (Haralick-H), local binary pattern-L (LBP) and Gabor (G) with individual and their combinational features are extracted, and four classifiers are taken for classification.

Based on the histogram from Fig. 5, LDA is chosen to be the best classifier for extraction of texture features. Using LDA classifier, texture features and their combinations results for all categories are shown in Fig. 6.

In Fig. 6, texture features are represented in a short form such as H for GLCM-Haralick, G for Gabor and L for LBP features. From Fig. 6, combination of texture features appears to be the best of texture features for classification of natural and artificially treated mangoes under each category shown in Table 2.

Table 2 shows that GLCM and Gabor features are suitable for classification. Accuracy for fusion of best color and best texture features is also calculated in each category which is shown in Table 3a-d.

Fusion of color and texture features cannot be calculated under badami variety for classification of natural and calcium carbide-treated mangoes, since best color and best texture features are obtained with two different classifiers which are shown in Table 3a.

When fusion is done in other categories, accuracy decreases, because some of the features in best color and best texture may not contribute together to obtain high accuracy. Perhaps only some of the features in best color and texture contribute together to gain good result. Hence, best features are needed to be identified even in best color and best texture features to obtain highest accuracy.

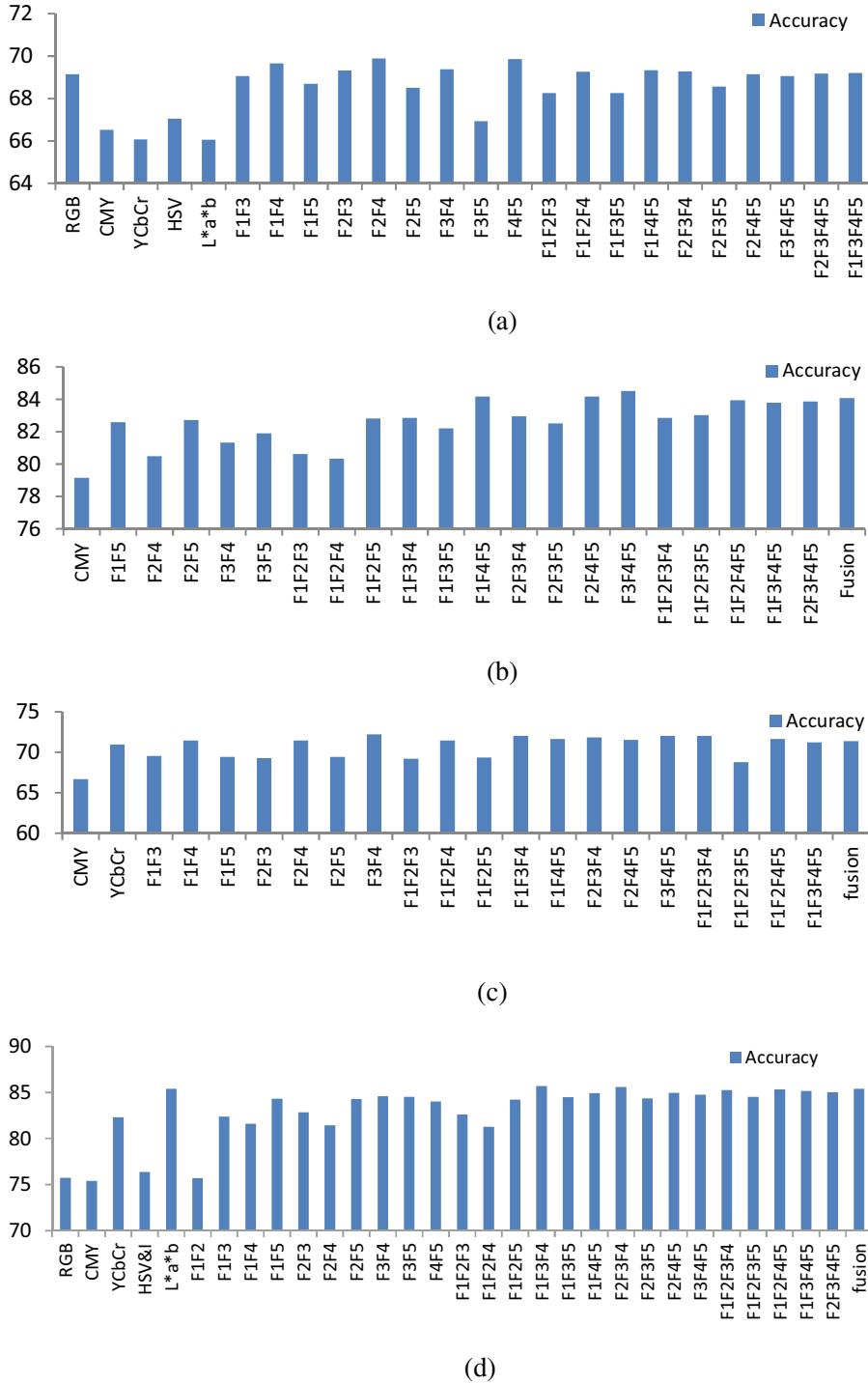
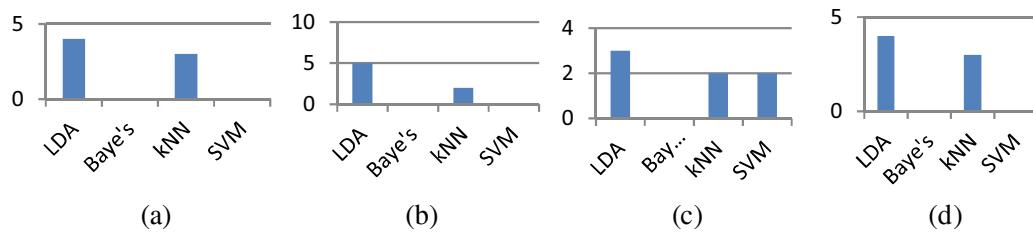
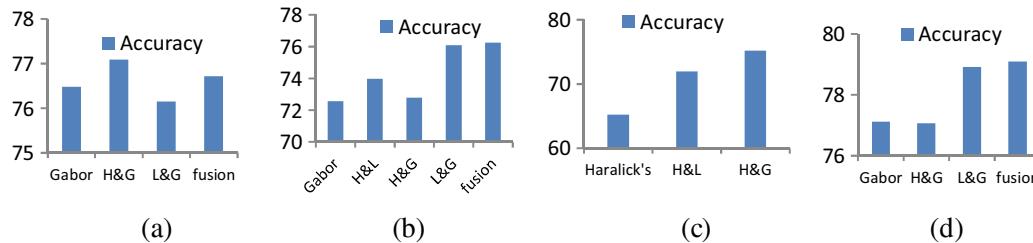


Fig. 4 Accuracy versus features: **a** badami variety classification of natural and calcium carbide-treated mangoes; **b** badami variety classification of natural and ethrel-treated mangoes; **c** rasipuri variety classification of natural and calcium carbide-treated mangoes; **d** rasipuri variety classification of natural and ethrel-treated mangoes

Table 1 Results for different category of classes for color features

Category	Best color features	Accuracy (%)
Badami-natural and calcium carbide-treated mangoes	Combination of CMY and HSV	69.89
Badami-natural and ethrel solution-treated mangoes	Combination of YCbCr, HSV and L*a*b	84.53
Raspuri-natural and Calcium carbide-treated mangoes	Combination of YCbCr, HSV	72.20
Raspuri-natural and ethrel solution-treated mangoes	Combination of RGB, YCbCr, HSV	85.72

**Fig. 5** Frequency of occurrence versus classifiers. **a** Badami-natural and calcium carbide; **b** badami-natural and ethrel solution; **c** raspuri-natural and calcium carbide; **d** raspuri-natural and ethrel solution**Fig. 6** Accuracy versus texture features and their combinations. **a** Badami-natural and calcium carbide; **b** badami-natural and ethrel solution; **c** raspuri-natural and calcium carbide; **d** raspuri-natural and ethrel solution**Table 2** Results for different category of classes for texture features

Category	Best of texture features	Accuracy (%)
Badami-natural and calcium carbide-treated mangoes	Combination of Haralick(GLCM) and Gabor	77.09
Badami-natural and ethrel solution-treated mangoes	Fusion of LBP, Haralick(GLCM) and Gabor	76.24
Raspuri-natural and calcium carbide-treated mangoes	Combination of Haralick(GLCM) and Gabor	75.19
Raspuri-natural and ethrel solution-treated mangoes	Fusion of LBP, Haralick(GLCM) and Gabor	79.09

Table 3 (a) Badami (N&C),
 (b) badami (N&E), (c) raspuri
 (N&C), (d) raspuri (N&E)

(a)			
Classifiers	Color	Texture	
PNN	69.89%	–	
LDA	–	77.09%	
(b)			
Classifiers	Color	Texture	Fusion
LDA (%)	84.53	76.09	68.55
(c)			
Classifiers	Color	Texture	Fusion
LDA (%)	72.2	75.19	65.73
(d)			
Classifiers	Color	Texture	Fusion
LDA (%)	85.72	79.09	74.19

4 Conclusion

In this proposed work, two varieties of mangoes were chosen such as badami and raspuri. In each variety of mangoes, three classes have been considered based on their type of ripening treatment, namely natural, calcium carbide and ethrel solution. To express the effectiveness of the proposed model, an extensive experimentation has been conducted on our own dataset of 2440 images. From three classes of mangoes, all types of color space features with individual and their combination were extracted. Then, the classification is performed with statistical and neural network-based classifiers. We obtained accuracy between 69.89 and 85.72% in all categories by choosing the best color features. Similarly, best texture features gave an accuracy between 75 and 79%. We have observed that accuracy decreases when fusion of color and texture features are done. The performance is also found to be reasonably good in comparison with human experts. The accuracy of the system is quite considerably less, due to a less interclass variation and more intra-class variation. The strength of this model is that it works for any resolution camera images between 8MP and 20MP and also distance from 12 and 16 cm. Further, this model can be used to develop as an android application for consumers to use it at the market such that creating awareness to fruit-sellers for not applying any artificial ripening agent during fruit ripening process.

Acknowledgements Maharaja Research Foundation, Mysore, supports this research work. We appreciate the support of MRF.

References

1. Islam, M.N., Imtiaz, M.Y., Alam, S.S., Nowshad, F., Shadman S.A., Khan, M.S.: Artificial ripening on banana (*Musa Spp.*) samples: analyzing ripening agents and change in nutritional parameters. *Cogent Food Agric.* **4**(1) Article: 1477232
2. Blasco, J., Aleixos, N., Molto, E.: Machine vision system for automatic quality grading of fruit. *Biosyst. Eng. Sci. Direct.* **85**(4), 415–423 (2003)
3. Cubero, S., Aleixos, N., Molto, E., Gomes-Sanchis, J., Blasco, J.: Advances in machine vision applications for automatic inspection and quality evaluation of fruits and vegetables. *Food Bioprocess Technol-Springer* **4**(4), 487–504 (2011)
4. Khoje, S., Bodhe, S.: Performance comparison of fourier transform and its derivatives as shape descriptors for mango grading. *Int. J. Comput. Appl.* **53**(3), 0975–8887 (2012)
5. Pauly, L., Sankar, D.: A new method for sorting and grading of mangoes based on computer vision system. In: IEEE International Advance Computing Conference (2015). <https://doi.org/10.1109/iadcc.2015.7154891>
6. Khoje, S.A., Bodhe, S.: Comparative performance evaluation of size metrics and classifiers in computer vision based automatic mango grading. *Int. J. Comput. Appl.* **61**(9), 0975–8887 (2013)
7. Tomas, U., Ganiron Jr.: Size properties of mangoes using image analysis. *Int. J. Bio-Sci. Bio-Technol.* **6**(2), 31–42 (2014)
8. Khoje S.A., Bodhe, S., Adsul, A.P.: Identification of artificial neural network configuration for quality sorting of mango (*Mangifera Indica L.*) using wavelet texture features. In: Elsevier-International Conference on Information and Mathematical Sciences. ISBN-9789351071624
9. Musale, S.S., Patil, P.M.: Identification of defective mangoes using gabor wavelets: a non-destructive technique based on texture analysis. *Int. J. Agric. Innov. Res.* **2**(6), 992–996 (2014)
10. Khoje, S.A., Bodheand, S., Adsul, A.P.: Automated skin defect identification system for fruit grading based on discrete curvelet transform. *Int. J. Eng. Technol.* **5**(4), 3251–3256
11. Nanaa, K., Rizon, M., Nordin, Rahman, A., Zaliha, Y.I.A., Aziz, A.: detecting mango fruits by using randomized hough transform and backpropagation neural network. In: IEEE International Conference on Information Visualization (2014). 10.1109/IV.2014.54
12. Khoje, S., Bodhe, S.: Comparative performance evaluation of fast discrete curvelet transform and color texture moments as texture features for fruit skin damage detection. *Springer J. Food Sci. Technol.* (2015). <https://doi.org/10.1007/s13197-015-1794-3>
13. Zakaria, A., Shakaff, A.Y.M., Masnan, M.J., Saad, F.S.A., Adom, A.H., Ahmad, M.N., Jaafar M.N., Abdullah, A.H., Kamarudin, L.M.: Improved maturity and ripeness classifications of magnifera indicav, harumanis mangoes through sensor fusion of an electronic nose and acoustic sensor. *Sensors* **12**(5), 6023–6048 (2012)
14. Salunkhel, R.P., Patil, A.A.: Image processing for mango ripening stage detection: RGB and HSV method. In: IEEE International Conference on Image Information Processing (2015). <https://doi.org/10.1109/iciip.2015.7414796>
15. Nagle, M., Romano, G., Intani, K., Spreer, W., Mahayothee, B., Sardsud, V., Joachim Müller, J.: A novel optical approach to monitor color changes of color peel for machine vision applications. In: International Conference of Agricultural Engineering CIGR-AgEng (2012)
16. Nandi, C.S., Tudu, B., Koley, C.: Machine vision based techniques for automatic mango fruit sorting and grading based on maturity level and size. In: Springer International Publishing-Sensing Technology: Current Status and Future Trends. Smart Sensors, Measurement and Instrumentation vol. 8, (2014). https://doi.org/10.1007/978-3-319-02315-1_2
17. Nandi, C.S., Tudu, B., Koley, C.: Computer vision based mango fruit grading system. In: International conference on Innovative Engineering Technologies. <http://dx.doi.org/10.15242/IIE.E1214004>
18. Mursalat, M., Rony, A.F., Hasnat, A., Rahman, M.S., Islam M.N., Khan, M.S.: A critical analysis of artificial fruit ripening: scientific, legislative and socio-economic aspects. *Chem. Eng. Sci. Mag.* **4**(1), (2013). ISSN 2220-3389

19. Ravindran, A., Anitha, R., Ravindran, A.: A review on non-destructive techniques for evaluating quality of fruits. *Int. J. Eng. Res. Technol.* **4**(09), (2015)
20. Live Chennai.com. <http://www.livechennai.com/healthnews.asp?newsid=10973>
21. Gurjar, P.S., Verma, A.K., Dikshit, A., Shukla, D.K.: Effect of ethrel spray on the ripening behavior of mango (*Mangifera indica L.*) variety 'Dashehari'. *J. Appl. Nat. Sci.* **9**(3), 1619–1623
22. <https://www.mid-day.com/articles/cac2-may-cause-cancer-blindness-seizures/15346168>
23. Nutritional Talk. <https://nwg-works.blogspot.in/2013/04/how-to-identify-banana-ripened-using.html>
24. Inter Institutional Inclusive Innovations Centre. www.i4c.co.in/idea/getIdeaProfile/idea_id/2969
25. Ashok, V., Vinod, D.S.: Using K-means cluster and fuzzy c means for defect segmentation in fruits. *Int. J. Comput. Eng. Technol.* **5**(9), 11–19 (2014)

A Comparative Analysis of Different Classifiers to Propose a Genetically Optimized Neural Network



Ankita Tiwari and Bhawana Sahu

Abstract Breast cancer is the most serious cancer found in female and sometimes in male. The only way of prevention is early detection. By using machine learning techniques we can detect symptoms at an early stage. As observed, different ML techniques for breast cancer has been implemented already to identify the type of tumor. In this paper we have done a comparative study of different existing classifiers like SVC, DTC, KNN, RFC and MLP to find the classifier which performs best for our dataset. Further, we have adopted the linear regression approach to optimize the feature selection. The obtained results will be given as input to the neural network utilized in genetic algorithm. Since machine learning techniques have the problem of solving the redundant and irrelevant features in the dataset, this leads to undesired result. We have tried to overcome this drawback by optimizing feature selection method to reduce this shortcoming. The optimized dataset will be used by Genetic algorithm to produce the fittest attributes being the most correlated ones (according to our problem). This will help to categorize the cancer being benign or malignant. The dataset used for current study in our demonstration is Wisconsin Breast Cancer Dataset (WBCD) from UCI Machine learning repository. The obtained results indicate that MLP classifier provides better accuracy for diagnosing breast cancer and further genetically optimized neural network architecture has been proposed.

Keywords Wisconsin breast cancer dataset · Feature selection · Diagnosis of cancer · MLP

A. Tiwari (✉) · B. Sahu

School of Computer Science and Engineering, VIT University, Vellore, Tamil Nadu, India
e-mail: ankita.tiwari2019@vitstudent.ac.in

B. Sahu

e-mail: bhawana.sahu2019@vitstudent.ac.in

1 Introduction

The early diagnosis of breast cancer can exponentially increase the chances of successful treatment of the ailment. According to the survey by the world health organization two decades back, 2% were in 20–30 years of age group, 7% were in 30–40 and 60% were above 50 years of age (out of every 100 breast cancer patients) and coming back to the present scenario the estimate has grown to 28% in 40–50 years of age, 16% in 30–40, 4% in 20–30 years of age group. This means almost 48% patients are below 50 years of age which is a very devastating condition [1]. To overcome this condition, the need of the hour is to find some early detection method for breast cancer as it can decrease the death rate. Also it can help doctors to predict more efficient result to carry such cases. The objective of this paper is to classify tumors into benign (non-cancerous) or malignant (cancerous) using features selected from WBCD.

Multilayered Perceptron (MLP) and k-Nearest Neighbor algorithm performs better than the other methods for cancer detection [2]. Although k-NN performs good but we have emphasized more on neural network to carry our prediction, as once the neural network is trained on one task its parameters are used as a good initializer for another task and this transfer learning cannot be achieved by k-NN [3].

We have compared [4] and analyze different classifiers such as Support Vector, Random Forest Classifier, K-Neighbor Classifier, Decision Tree classifier and Multi-layer perceptron. Their performances will be accessed using different evaluation parameters and identify the best classifier which will be used in future for evaluation in genetic algorithm upon the dataset optimized using linear regression [5]. The findings of this paper will be used for future research to opt for the best performing baseline technique which can provide the optimal prediction for future comparison.

The paper constitutes as follows: Sect. 2 illustrates the overview of the related work. Section 3 describes the methodology. Section 4 presents the results and Sect. 5 concludes and future scope of this study.

2 Literature Review

The idea of neural networks has been derived from the biological neurons (brain cells); i.e., they have their own communication channels known as axons and dendrites. A Neural Network has millions of processing units that forms an interconnected network to process a large amount of information based on the response or input given by the experiments. It has several hidden layers present in between where the bias function and weight is applied (Fig. 1).

Nemissi et al. [6] in his paper has discussed about Extreme Learning Machine Algorithm and has worked on WBCD for examination. The classification system opted better generalization performance when reduced the no of hidden nodes, compared to the conventional techniques.

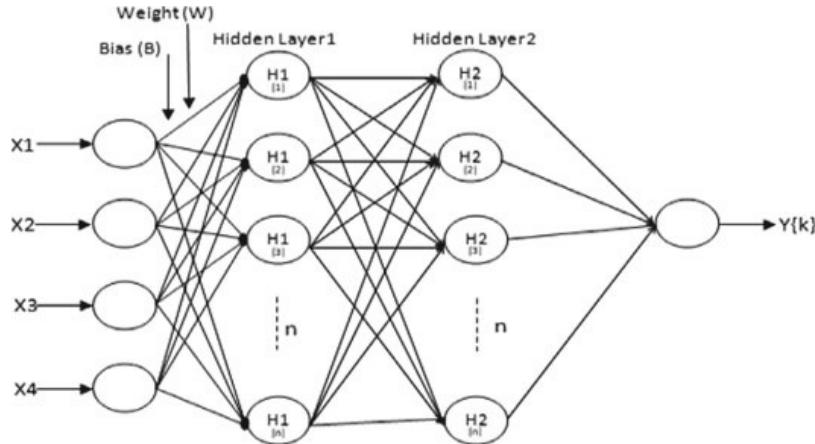


Fig. 1 Multilayer perceptron (Neural N/W) architecture

Sethi in [7] performed the comparison analysis between evolutionary algorithms and ML algorithms are performed using WBCD dataset. The results obtained by her in the four classifiers Genetic artificial neural network, CPSO, C4.5 and K-Nearest Neighbor are used in WBCD. Among all, GANN gives the best with 95%.

Lavanya and Usha Rani [8] used CART algorithm. Breast Cancer, WBC (Original) and WBCD (diagnostics) with different types of attribute from UCI repository is taken by her. The results analyzed by her are SVM AttributeEval-73.03%-breast cancer, Principal Components Attributes Eval-96.99%-For WBC (Original), Symmetric Uncert Attribute SetEval-94.72% WBC (diagnostic).

Xue et al. [9] performed the PSO for feature selection. Fourteen datasets is used from the repository of UCI machine learning. The experiment results in the initial-isolation strategy can affect the features selected and computational time.

Belciug and Gorunescu [10] investigated Genetic Algorithm is used for hybrid neural network. WPBC; LRBC; WRBC; MLP; GA; RBF; PNN; PCNN datasets is used for this approach the results obtained are the approach for selection of optimal weight for Neural N/W using a Genetic algorithms which has the natural ability to optimize model parameters in some intelligent way. It has used different datasets without decreasing the accuracy. Another approach for and the classical BP learning is an algorithm based on the network-error in the mutation operator.

Alic et al. [11] applied Logistic Regression algorithm, Support Vector Machine (SVM) algorithm, Rotation Forest, Decision Tree algorithm, Bayesian Network, RF algorithm, MLP and Radial-Basis-Function Networks (RBFN). GA feature selection based on genetic algorithm to find the optimal attribute is used with data mining techniques for classification is applied. The result requires using genetic algorithm and random forest gave accuracy as high as 99.48%.

3 Methodology

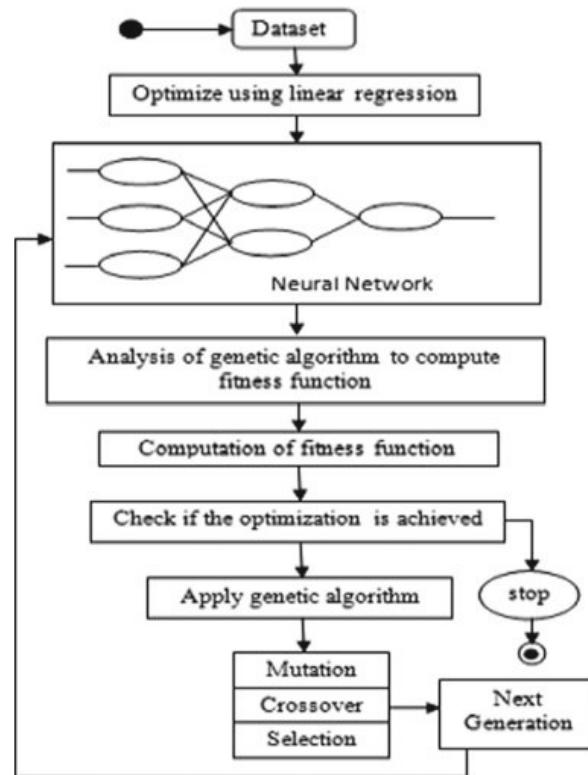
We have studied about the Machine Learning techniques used for classification of breast cancer. We have provided the description on various algorithms and evaluated based on the performance measures like accuracy, sensitivity, specificity, complexity, the size of the decision tree, computation time and so on.

The Activity diagram using unified modeling has been presented below for reference which provides an overview of the steps we have followed for execution (Fig. 2).

3.1 Dataset

The dataset used in this paper has been taken from WBCD (<https://archive.ics.uci.edu/ml>: UCI Repository) [12] which consists of total 569 instances (357-Benign and 232-Malignant) taken from FNA of affected human breast tissue.

Fig. 2 Methodology



3.2 *Classifiers Used*

We have used five Machine Learning classifiers to analyze the accuracy. Based on the results we can head toward the next step to optimize the process genetically. Various classifiers used are:

3.2.1 *Support Vector Machine*

It is supervised and majorly used for regression and classification problems. All the data items are plotted as a point in an n-dimensional space (n is the number of features) and every coordinate has a particular value. The values of all coordinates are the values of features respectively. Later classification is done based on the hyperplane found which differentiates the two classes. It is capable of performing linear as well as non-linear classification and regression (sometimes) then outlier detection is used for building a classifier. The hyperplane is taken in such a way that it is closest possible to the data points from each class and these closest points are called as support vectors. When we apply SVC [13], the main aim is to select the accurate input features subset very precisely so as to retrieve optimal parameters.

3.2.2 *Random Forest Classifier*

Random Forest classifier is a quick learner and provides efficient results for huge data. It is popularly used machine learning technique as it saliently provides two features in data mining, that are: high accuracy in forecasting and provides new information on variables which are very important aspects for classification methods. It is one of the most accurate algorithms which is capable of handling thousands of attributes without any feature selection. It provides the estimation of the important attributes. It is a highly efficient algorithm for estimating the missing data and also maintains the accuracy in estimation.

3.2.3 *K-Nearest Neighbor Classifier*

K-Nearest Neighbor (k-NN) algorithm is a supervised ML technique used particularly for classification and regression problems. It makes no assumptions on the fundamentally distributed data. K-NN has a huge utilization in the field of pattern recognition and predictions. It works on the principle of object classification methods in which objects are classified into several groups. The basis for classification is the values of variables which are self-explanatory. There is no particular model associated, errors are computed arithmetically and one simple solution is provided which classifies a new object referenced to the results known.

3.2.4 Decision Tree Classifier

The main characteristic of decision tree is that it acquires inferences from instances based on general laws and learns from it. It can gather a particular fact and provide some particular conclusions which are very specific. According to the branching series which perform various logical and arithmetic tests is a predictive model. The approach used by decision tree classifier is to break one stage complex classification into several smaller less complex test cases. While the decision tree in breaking down the tests into smaller and smaller pieces at the same time, linking of the other instances is also progressing simultaneously.

3.2.5 Multilayered Perceptron

A multilayer perceptron contains several layers of node which are interconnected, where each node is named as a perceptron and resembles multiple linear regression technique. The multiple linear regression produces signals which is fed by perceptron to the activation function (which can be non-linear). The input pattern is mapped by the input layer and output layer maps output signals [14]. There are several layers hidden between the input and output layer which optimizes the input weights (values) until the neural network's error margin is least. According to the hypothesis the hidden layers exhibit unique features in the input data which has the ability to predict the outputs.

After examining all the above mentioned classifiers we have observed that the best accuracy result is obtained by the multilayer perceptron and k-neighbor classifier with the accuracy of 93.97% and 93.85% respectively.

3.3 Neural Network and Genetic Algorithm

Multilayered perceptron [15] consists of the following parameters: an input denoted by ' p ', some ' n ' number of layers which are hidden, a layer ' q ' which provides the output and a set of biases and weights which are fed as intermediate in each layer, denoted by ' b ' and ' w ' respectively. An activation function is chosen which maps the output in the desirable format; here we have used sigmoid function denoted by ' σ '.

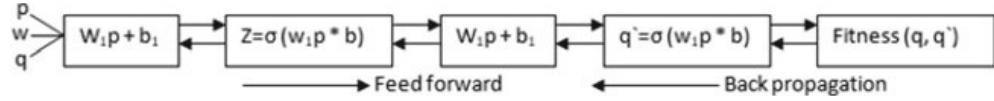
3.3.1 Training Neural Network

The output q for 2-layer MLP is:

$$q = \sigma(w_2\sigma(w_1p + b_1) + b_2) \quad (1)$$

The bias (b) and the weight (W) are the only variables that affect the output q and determine the strength of network. Optimization of the weights and biases from the given input is known as training MLP [16].

The process of training for each iteration comprises of calculating the predicted output q (feed forward) and updating the weights and bias (back-propagation).



3.3.2 Forward-Feed

Forward-feed is just simple calculus which gives the result (output) for the MLP as:

$$q = \sigma(w_2(w_1 p + b_1) + b_2 A) \quad (2)$$

3.3.3 Fitness Function

The fitness function used will be sum-of squares error which is the sum-of differences between obtained value which is predicted and actual value. The difference found is squared and the absolute value of the difference is calculated. We want to retrieve the most optimal set of biases and weights that minimizes the function while training our neural network.

$$\text{sum-of-square error} = \sum_{i=1}^n (q - q')^2 \quad (3)$$

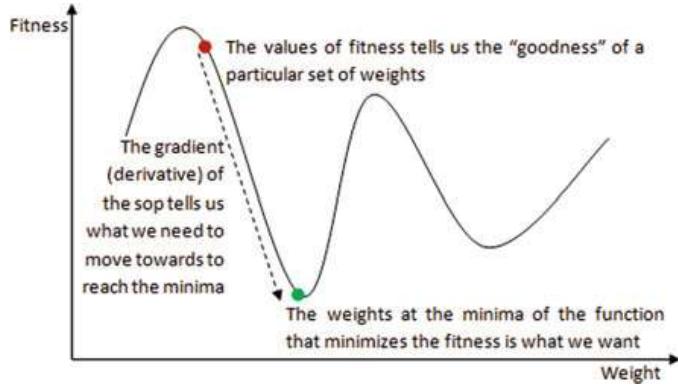
3.3.4 Back-Propagation

After calculating the error of our prediction, we will feed the error calculated back and update to the MLP. To find the value that we will use to adjust the biases and weights, we will differentiate the (sum-of-squares) error function in terms of the biases and weights (Fig. 3).

The result obtained after differentiation will be used to update the weights and biased by increasing/ decreasing in the next iteration. This process is known as Gradient Descent.

$$\text{Loss}(q, q') = \sum_{k=1}^n (q - q')^2$$

Fig. 3 Gradient descent curve



$$\frac{\partial \text{Loss}(q - q')}{\partial W} = \frac{\partial \text{Loss}(q - q')}{\partial q'} * \frac{\partial q}{\partial z} * \frac{\partial z}{\partial W}$$

where, $z = Wp + b$

$$2(q - q') * \text{derivative of sigmoid function} * p = 2(q - q') * z(1 - z) * p \quad (4)$$

3.3.5 Linear Regression

To optimize the feature selection for using as input in multilayer perceptron, we have used linear regression to model the relationship between two attributes [17]. By using the linear regression concept we have calculated the attributes with maximum correlation value. These correlated attributes will have maximum probability of being the factor responsible for causing tumor. And hence, we can deduce the nature of the tumor being benign or malignant. We can understand the linear regression as it calculates the regression coefficients A_0 and A_1 in the equation.

$$y_j = A_0 + A_1 x_j + \epsilon \quad (5)$$

To compute the confidence interval and hypothesis tests, the assumption is that, the errors are normally distributed with mean zero and variance σ^2 and are independent.

$$\text{variance}(\sigma^2) = \frac{1}{n} \sum x_{i^2} - x'^2 \quad (6)$$

Taking a sample of n-observations on x and y , the method of least squares calculates A_0 and A_1 as well as various other parameters that describes the accuracy for the estimates and the fitness extent of a linear line to the data. The calculated line should fit the data exactly, a value should be added between the actual and fitted value signifying the discrepancy between them. Then the equation becomes:

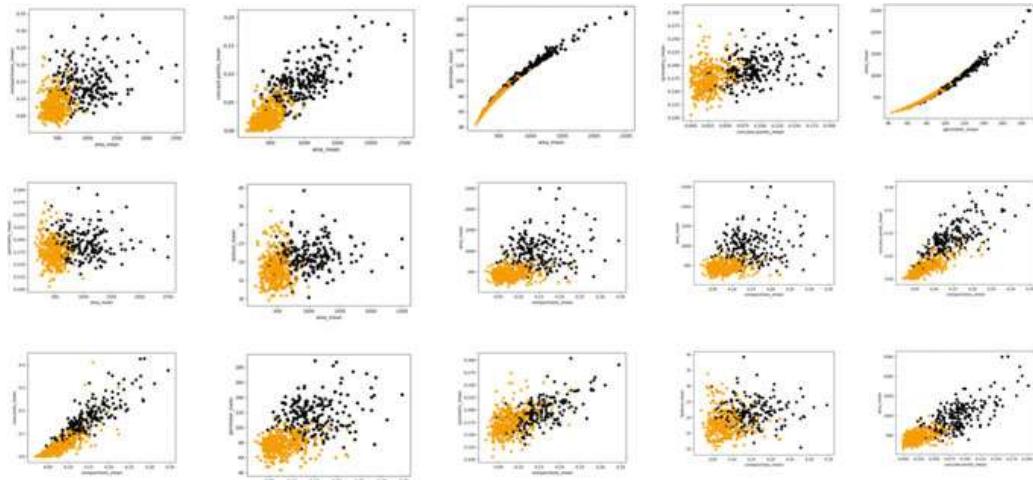


Fig. 4 Correlated feature plot

$$y_j = b_0 + b_1 x_j + e_j \quad (7)$$

j : Row or observation number which gives 0; b_1 : gives 1; b_2 : The data discrepancy calculated between actual-data valued y_j and the fitted value calculated by equation of linear regression (\bar{y}_j)

Some of the graphs plotted using the above mentioned concept of linear regression for feature selection for our methodology has been provided below for reference (Fig. 4).

3.3.6 Genetic Algorithm

It mimics the process of evolution, based on Darwin's theory and is primarily based on the concept of natural selection and natural genetics. Only the fittest will survive, procreate and successive generations will become better compared to the previous generations [18]. It maintains the balance between efficiency and efficacy which is necessary for survival in many different environments. [19] It belongs to the evolutionary algorithms which extract intelligence in random search provided, with ancient data to mold the direction of the search in such a way that better performance is achieved in solution space and these high quality solutions are used for optimization.

3.3.7 Logistic Regression

In genetic algorithm, we will use logistic regression as the activation function. Since we want to relate predicted values to occurrences (probabilities), hence we will use the sigmoid-curve-function. The Sigmoid-curve-function will match the actual value into a binary value. DNA segment can be depicted as a string of binary digits. Hence

to obtain desired output from the neural network designed we will use sigmoid function. It is used to map predictions to probabilities and can be deduced as: $S(z)$ gives output between 0 and 1, z gives input to a function

$$s(z) = \frac{1}{1 + e^{-z}} \quad (8)$$

4 Result

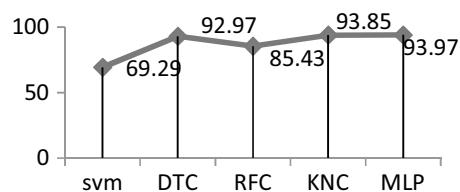
The results obtained from various classifiers such as Support Vector classifier, Decision tree classifier, Random Forest classifier, k-Nearest Neighbor classifier and Multi-layered perceptron (Neural Network). To evaluate the performance, we have used Breast Cancer Dataset from Wisconsin hospital (WBCD). The used database consists of 585 records in which 16 records were missing. So these missing records are excluded. The remaining dataset consists of 569 records, from which 232 samples from the complete dataset are malignant and 357 are benign. In order to evaluate the generalized performance, we have divided the dataset in the ratio 80:20, of which 80% data has been trained and tested, and the rest of the 20% data has been tested only. This procedure is repeated ten times and observations have been noted which can be observed by the table mentioned (Table 1).

The average performance given by support vector classifier is 69.29%, decision tree classifier 92.97%, random forest tree 85.43%, k-Nearest Neighbor 93.85% and Multilayer perceptron 93.97% (Fig. 5).

Table 1 Performance analysis table

SVC	69.29	69.29	69.29	69.29	69.29	69.29	69.29	69.29	69.29	69.29
DTC	91.22	93.85	94.75	92.98	92.98	91.22	93.85	91.22	92.1	92.98
RFC	93.85	95.61	94.73	94.73	97.36	93.85	93.85	94.73	93.85	95.61
KNC	93.85	93.85	93.85	93.85	93.85	93.85	93.85	93.85	93.85	93.85
MLP	93.98	94.47	93.98	94.85	94.98	93.35	93.1	94.47	94.22	93.35

Fig. 5 Graph showing accuracy



5 Conclusion and Future Scope

In this paper we have implemented five classifiers like SVM, RFC, KNN, DTC and MLP to find the accuracy of dataset for the selection of the best classifier which will be used to optimize the genetic algorithm. From the study, it is proved that Multilayered perceptron outperforms other classifiers. Therefore for future study we will adopt multilayer perceptron to optimize the genetic algorithm's result. As the input attributes have been optimized in our current work, this can be taken to process further as input datasets of the genetic algorithm.

References

1. Dubey, A.K., Gupta, U., Jain, S.: Breast cancer statistics and prediction methodology: a systematic review and analysis. *Asian Pac. J. Cancer Prev.* **16**, 4237–4245 (2015)
2. Abdel-zaher, A.M., Eldeib, A.M.: Breast cancer classification using deep belief networks. *Expert Syst. Appl.* **46**, 139–144 (2016)
3. Khuriwal, N.: Breast cancer diagnosis using adaptive voting ensemble machine learning algorithm. In: 2018 IEEMA Engineer Infinite Conference (eTechNxT), pp.1–5 (2018)
4. Bayrak, E.A., Kirci, P.: Comparison of machine learning methods for breast cancer diagnosis. pp. 4–6 (2019)
5. Chaurasia, V., Pal, S.: Data mining techniques : to predict and resolve breast cancer survivability about breast cancer
6. Nemissi, M., Salah, H., Seridi, H.: Breast cancer diagnosis using an enhanced extreme learning machine based-neural network. In: 2018 2018 International Conference on Signal, Image, Vision and their Applications (SIVA), pp. 1–4 (1945)
7. Sethi, A.: Analogizing of evolutionary and machine learning algorithms for prognosis of breast cancer. In: 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), pp. 252–255 (2018). <https://doi.org/10.1109/icrito.2018.8748502>
8. Lavanya, D., Rani, Dr.K.U.: Analysis of feature selection with classification: breast cancer datasets. *Indian J Comput Sci Eng (IJCSE)* **2**, 756–763 (2011)
9. Xue, B., Zhang, M., Browne, W.N.: Particle swarm optimisation for feature selection in classification: novel initialisation and updating mechanisms. *Appl. Soft Comput. J.* **18**, 261–276 (2014)
10. Belciug, S., Gorunescu, F.: A hybrid neural network/genetic algorithm applied to breast cancer detection and recurrence. *Expert Syst.* **1**(00), 1–12 (2012)
11. Alic, E.: Breast cancer diagnosis using GA feature selection and Rotation. (2015). <https://doi.org/10.1007/s00521-015-2103-9>
12. breast-cancer-wisconsin-data_data
13. Polat, K., Güne, S.: Breast cancer diagnosis using least square support vector machine. *Digit. Signal Proc.* **17**, 694–701 (2007)
14. Rakhlina, A., Shvets, A., Iglovikov, V., Kalinin, A.A.: Deep convolutional neural networks for breast cancer histology image analysis
15. Ma, M., Zhang, L.: Optimizing a fuzzy neural network with A hierarchical genetic algorithm. pp.19–22 (2007)
16. Jafarpisheh, N., Nafisi, N., Teshnehlab, M.: Breast cancer relapse prognosis by classic and modern structures of machine learning algorithms. pp. 120–122 (2018)
17. Zheng, B., Yoon, S.W., Lam, S.S.: Expert systems with applications breast cancer diagnosis based on feature extraction using a hybrid of K-means and support vector machine algorithms. *Expert Syst. Appl.* (2013). <https://doi.org/10.1016/j.eswa.2013.08.044>

18. Chauhan, P., Swami, A.: Breast cancer prediction using genetic algorithm based ensemble approach. In: 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1–8 (2018)
19. Lozano, F., Koltchinskii, V.: Self bounding genetic algorithms for machine learning. (2005)

Vehicle Direction-Based B-MFR Routing Protocol for VANET



Parimala Garneputdi and D. Venkatesulu

Abstract Vehicular ad hoc network (VANET) is a kind of self-configured network for connecting vehicles and roadside units (RSUs). The backbone of the network and the RSUs are connected together, to access the Internet by each vehicle in order to access network applications and various services by users or drivers. One of the important application of VANET is to improve the public safety by providing the necessary information to the drivers. The trending characteristics of VANET like large network size, dynamic topology, and high mobility, caused the development of efficient routing protocols. Most forward within radius (MFR) is a position-based routing protocol selects a node from the list of nodes which are in its coverage area as a next forwarding node. In MFR, the hop count becomes increased with the increase of network size. Border node-based most forward within radius (BMFR) is also a position-based routing for reducing the limitation of MFR by selecting border nodes as forwarding node to establish a path from source to destination. Frequent link damage happens in BMFR because of its next forwarding node selection principle, i.e., choosing of border nodes. This paper introduces a new position-based routing protocol named vehicle direction-based BMFR (VDBMFR) for establishing a route between two end devices. This protocol uses the principle of BMFR by adding a new feature—vehicle direction. End-to-end delay of VDBMFR is less compared with the end-to-end delay of MFR and BMFR. This protocol is simulated using NS-2 simulator and analyzes the result of VDBMFR, BMFR, and MFR in terms of end-to-end delay. The end-to-end delay of B-MFR and MFR is significantly higher than VDBMFR.

Keywords MANET · VANET · ITS · RSUs · Routing protocols · MFR · BMFR · VD-BMFR

P. Garneputdi (✉) · D. Venkatesulu
VFSTR University, Vadlamudi, AP, India
e-mail: garneputdi.parimala@gmail.com

D. Venkatesulu
e-mail: hodcse@vignan.ac.in

1 Introduction

Group of wireless nodes connected together to form a network without any fixed infrastructure is called ad hoc network. MANET is a subclass of ad hoc network. VANET is a subclass of MANET. It uses several methods for intelligent transport system (ITS).

VANET became more difficult and challenging research domain to provide the interaction of messages among V2V and V2I [1] because of frequent link disconnections due to their high unpredictable dynamic topology.

VANET works with a special application called intelligent transport system (ITS) for providing road safety and public safety with the increasing no. of vehicles on road day to day [2, 3]. Millions of people are dying in accidents because of the high speed of the vehicles, unknown road condition. VANET provides a good facility of sharing such information in order to guide the drivers to reach their destination safely. This can be done with the help of routing. Routing is a mechanism used for exchanging the information between source and destination [4, 5]. Routing in VANET became a challenging task for providing safer and well-organized transportation in future [6].

The general architecture of VANET is shown in Fig. 1, having vehicle, roadside units (RSUs) with Internet facility. In VANET, one vehicle is connected with other vehicle and RSUs through a special device called on-board unit (OBU) [7] which is with an Internet facility to exchange the information. The communication among the vehicles will be done using a mechanism called routing by generating a path from source to destination. Routing is a process of communicating one vehicle with other vehicle to exchange the information. The communication architecture of VANET is shown in Fig. 2.

High mobility of the node leads to dynamic changes in network topology and frequent path failure, which reduces the performance of routing protocol and increases the overhead to the network by generating a new path all the times dynamically [8].

Based on the process of finding the path from S to D in VANET [9], routing is categorized as the following types: position-based routing, cluster-based routing, geocast-based routing, topology-based routing, and broadcast routing. This paper gives the description of position-based routing protocols.

Fig. 1 General architecture of VANET

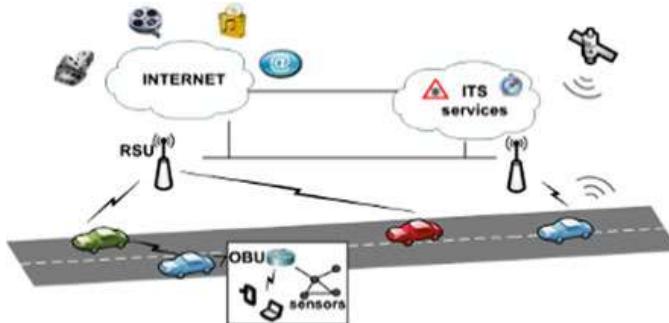
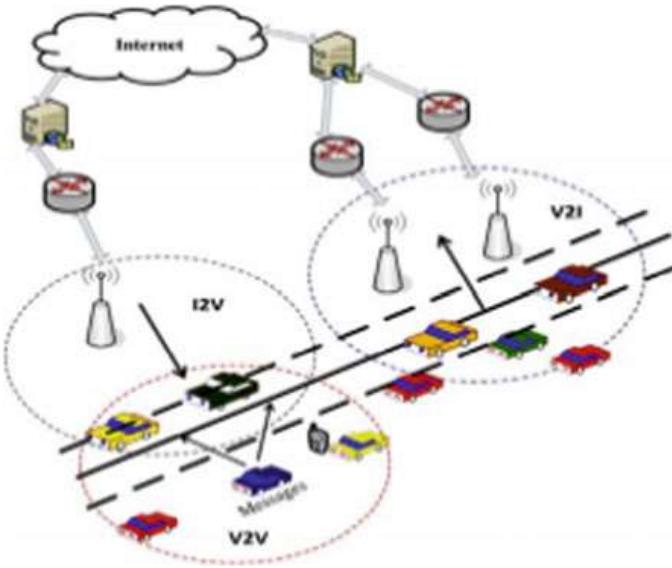


Fig. 2 Communication architecture of VANET



Position-based routing is performed by finding the geographic position [10] of the vehicle and sharing among the on-road vehicles [11–17]. Information forwarding in position-based routing is performed through direct communication or through intermediate nodes. Because of the ad hoc nature of the nodes, VANET does not have any fixed infrastructure [18] and causes frequent path failure.

In this paper, a new routing protocol called VD-BMFR was developed to connect one vehicle with other vehicle to exchange the information. End-to-End delay of VDBMFR is compared with the MFR and BMFR protocols. MFR and BMFR position-based routing protocols are explained in Sects. 2 and 3, and VDBMFR is described in Sect. 4.

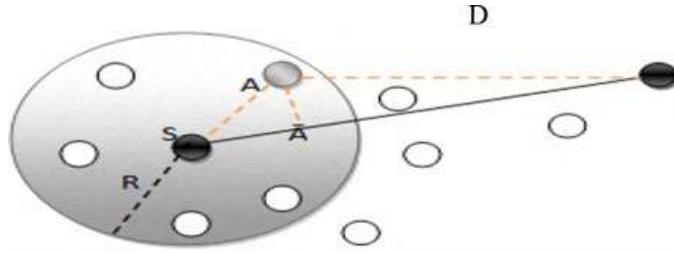
2 Most Forward Within Radius (MFR) Routing Protocol

MFR is a routing protocol to generate a (route) communication link between source and the destination by taking the geographic position of the vehicle into consideration. MFR calculates the progression value of all the neighbors of a source to destination and selects one among those with highest progression as a next forwarding node for vehicle to forward the packet to a distant.

MFR sends a packet to a node which is nearest (node with highest progression) to destination node in order to decrease the hops number. In Fig. 3, S denotes source, D denotes destination nodes, R (radius of circle) which denotes the highest communication range of the source node S . It has five neighbors in that communication range. It chooses node A as succeeding hop to transmit packet to the destination. The projection A' of A on line SD is nearest to destination D .

The location of neighbor node is identified by every vehicle by sending a beacon message having their own identity (ID), the exact position, speed, and time of vehicle

Fig. 3 MFR forwarding method



in network. On the other side, a beacon message from neighbor node the received node provides the position details of the neighbor. The source node checks the position information of its neighbor nodes that are closer to destination will choose as a next hop.

3 Border Node Based—Most Forward within Radius (B-MFR)

B-MFR routing nodes with one hop distance are classified into two types of nodes—interior nodes and border nodes. The distance of the one hop neighbor is equal to coverage range of central node which is considered as the border node [8, 18], and the remaining node which is at distance less than the communication range of the source is treated as the interior node (Fig. 4).

In B-MFR projection value of all the border nodes of a central node is calculated and selects one among those which is with maximum projection value as the forwarding node (next hop) in avoiding the usage of interior nodes as the forwarding node for further transmissions.

In Fig. 5, the node S selects the A as next node (forwarding node) because it is considered as the highest transmission range with highest projection on SD (source

Fig. 4 Border node architecture

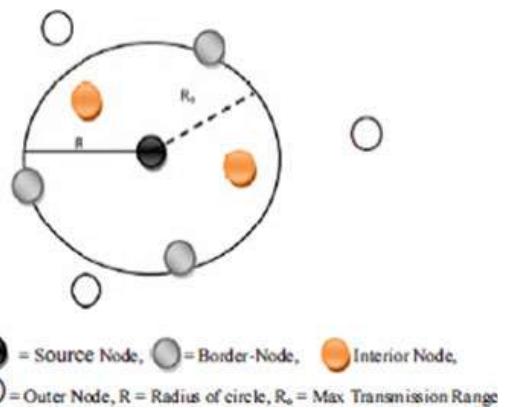
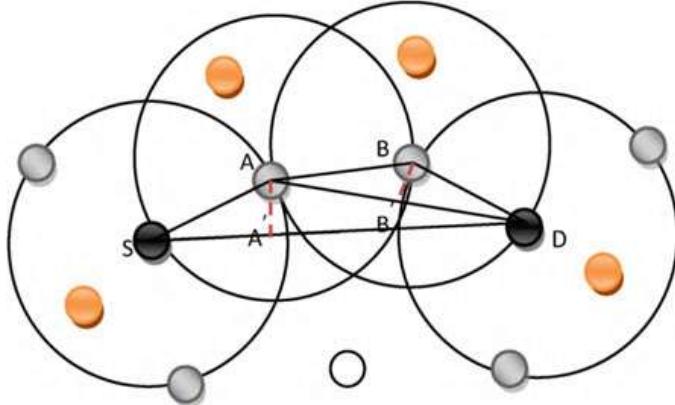


Fig. 5 B-MFR forwarding method



and destination). Node A follows the same process in order to find the next transmitting node. In this process, node B is identified as next hop which is at border of A to forward the packets from S to D .

4 Vehicle Direction-Based Border Node Most Forward within Radius (VD-BMFR)

MFR selects an interior node for finding the optimal path from source to destination. In MFR, hop count will be increased because of usage of interior nodes. This limitation can be overcome by developing a next protocol called BMFR, which selects border node as next hop (forwarding node) instead of interior node to construct the route from source to destination. Choosing the border node in BMFR causes frequent link breaks because the node is at the border because border node may move out frequently from the communication range of the node because of its high mobility. This becomes the limitation of BMFR protocol. Direction of the intermediate node is considered by both MFR and BMFR protocols in finding the path from source and destination. VDBMFR is a routing protocol, which overcomes the limitations of MFR and BMFR protocol. VD-BMFR protocol works on the mentioned concepts:

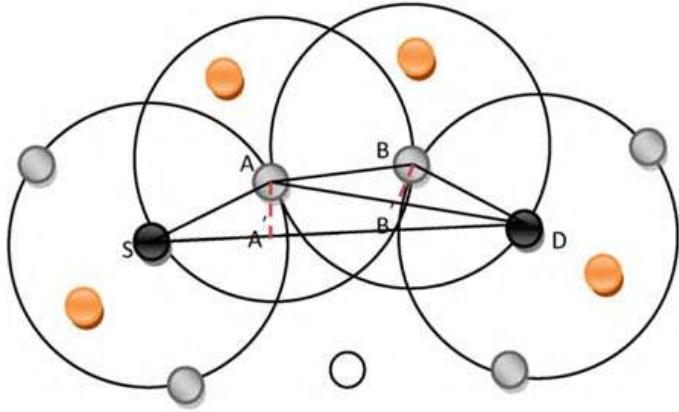
- use of border node to transmit the data.
- send a default message to neighbors.
- each vehicle should be with GPS digital map facility.
- the direction of packet delivery must be toward the destination
- message-oriented communication among nodes will occur.

Routing Strategy In order to identify the direction of the moving vehicle [19], each and every vehicle in network continuously broadcasts an INIT message (see Table 1), which includes the vehicle speed, destination address, vehicle's ID, current

Table 1 INIT message

Vehicles INIT message

Vehicle ID	Position of the vehicle	Driving direction	Vehicle speed	Destination address	Current time
------------	-------------------------	-------------------	---------------	---------------------	--------------

Fig. 6 VD-BMFR forwarding method

position, moving direction, and time of sending the INIT message to find single-hop neighbors. Each vehicle in network maintains two tables: data list table and neighbor table. After receiving INIT message, the message content is stored by each vehicle merge the sender's data to neighbor table. Data list checks the packet id whenever the data packet is received by the vehicle. Each packet in list is ordered in FIFO manner or else vehicle looses the packet (Fig. 6).

Algorithm

Notations:

PFN: present transmitting node

GNN: group of neighbors of present transmitting node

SCN: group of chosen candidate nodes

SNN: choose the preceding node

Ro: maximum range of communication

Pseudo Code:

- 1) PFN = SN // SN is actual source node
 - 2) Identify whether the destination is present in coverage range: So Stop
 - 3) Calculate Euclidian distance to each and every node starting from source node in GNN PFN
 - 4) For all $N \in NN$, $I \leftarrow 1$ to n
 - { If(direction of N_j is towards destination node)
 - { if (distance of N_j from PFN == R_o)
 - { SCN = SCN U N_j ;
 - For $I \leftarrow 5$
 - { get the position of the border node;
 - find the distance from border node to destination node;
 - if the distance is in descending order then
 - select the node as forwarding node;
- 5) In SCN find the projection of each and every node at X-axis
- 6) Choose the next preceding node NN includes maximum projection (X-value) as SNN
- 7) CFN = SNN // source is selected as next preceding node
- 8) Repeat 1- 6
- 9) Stop

5 Results

Performance Evaluation: In this section, the performance of the VDBR is compared with MFR and BMFR algorithms with respect to the latency and packet delivery ratio through simulation at transmission range of 250 m and a transmission rate of 2 Mbps.

In each run, five sender and receiver pair are selected randomly at a constant bit rate of 512-bytes in a UDP packet generation application. No. of vehicles considered for simulation is 20–40 at the duration period of 500–1000 s accomplished 10 runs. Table 2 shows list of methods considered for simulations.

Figure 7 shows a network topology with 30 vehicles, where vehicle 2 has a packet and destination is vehicle 12.

End-to-End Delay The average delay between source and destination for successful packet delivery is measured and plotted in the following figure. From Fig. 8, the one-way delay for VD-BMFR is probably less than MFR and B-MFR when no. of

Table 2 Simulation parameters

Parameters list	Esteem values
Simulator of network	NS2
Simulator of mobility	VanetMobiSim
Area of simulation	1600 m × 1600 m
CBR rate	512 bytes/s
802.11 rate	2 Mbps
Transmission range	250 m
Simulation runs	10
Average speed	50 km/h
Time of simulation	500–1000 s
Number of vehicles	30–50

Fig. 7 A snapshot of the VD-BMFR network topology

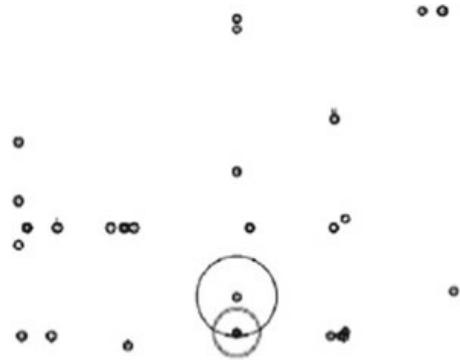
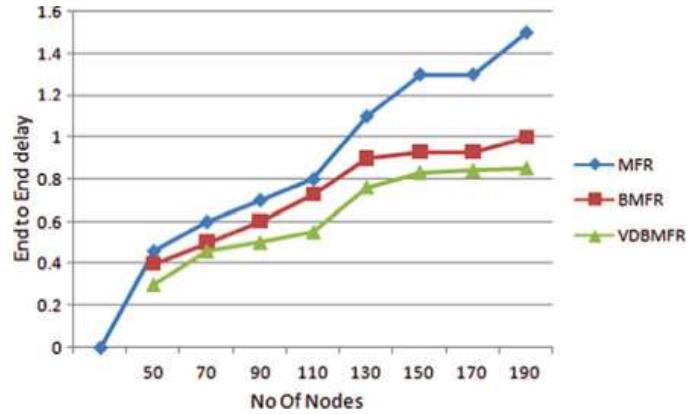


Fig. 8 One-way delay



vehicles becomes more. VD-BMFR chooses border nodes which are at the direction toward the destination to reduce the hops in a route. So, the end-to-end delay [20] from start point to end point is reduced and then no. of nodes will be increased (Fig. 8).

6 Conclusion

VDBMFR selects the border node vehicle which moves in the direction toward the destination as following hop for transmitting data. Each point in the VDBMFR sends an INIT message to its neighbor to acquire the routing information to make a routing decision in generating route from source to destination. The result shows that VDBMFR improves the packet delivery ratio and reduces the end-to-end delay.

References

1. Rana, S., Rana, S., Purohit KC.: A review of various routing protocols in VANET. *Int. J. Comput. Appl.* **96**(18), 0975–8887. (2014)
2. Garnepudi, P., Venkatesulu, D.: A comparative study on pothole detection techniques for vehicles in an intelligent transportation system. *Pak. J. Biotechnol.* **14**(II), 344–347 (2017)
3. Lasya Priya, B.V.N.S., Garnepudi, P., Venkatesulu, D.: VANET: metropolitan scenario for traffic handling and congestion avoidance. *Pak. J. Biotechnol.* **14**(II), 277–280 (2017)
4. Zorkany, M., Abdel Kader,N., Chen, K.: VANET routing protocol for V2V implementation: a suitable solution for developing countries. *Journal* **4**(1)
5. Dutta, R., Thalore, R.: A review of various routing protocols in vANET. *Int. J. Adv. Eng. Res. Sci. (IJAERS)* **4**(4), 2456–1908 (2017). ISSN: 2349-6495. <https://dx.doi.org/10.22161/ijaers.4.4.34>
6. WHO.: Status Report on Road Safety, World Health Organization (WHO), Geneva, Switzerland (2013)
7. Abbas, M.K., Marwan, A.-A., Taharim, N.F.: Vehicle’s Direction Determination Protocol via VANET. *Int. Rev. Autom. Control.* **9**(3), 161–166 (2016). <https://doi.org/10.15866/ireaco.v9i3.9262>
8. Kumar, R., Dave, M.: A review of various VANET data dissemination protocols. *Int. J. U-E-Ser. Sci Technol.* **5**(3) (2012)
9. Siva Nageswara Rao, S., Parimala, G., LakshmiNadh, K.: An efficient location detection mechanism for VANETS using smart phone device. *Int. J. Pure Appl. Math.* **118**(14), 215–219 (2018) ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version). <http://www.ijpam.eu>
10. Nageswara Rao, S.S., Parimala, G., Lakshmi Nadh K., An efficient location detection mechanism for VANETS using smart phone device. *Int. J. Pure Appl. Math.* **118**(14), 215–219 (2018)
11. Taleb, T., Sakhaei, E., Jamalipour, A., Hashimoto, K., Kato, N., Nemoto, Y.: A stable routing protocol to support ITS services in VANET networks. *IEEE Trans. Veh. Technol.* **56**(6), 3337–3347 (2007)
12. Kaiwartya, O., Kumar, S.: Geocast routing: recent advances and future challenges in vehicular adhoc networks. In: Proceedings of Signal Processing and Integrated Networks (SPIN ‘14), pp. 291–296 (2014)
13. Kaiwartya, O., Kumar, S.: Cache agent based geocasting (CAG) in VANETs. *Int. J. Inf. Commun. Technol.* in Press
14. Kaiwartya, O., Kumar, S., Kasana, R.: Traffic light based time stable geocast (T-TSG) routing for urban VANETs. In: Proceedings of the 6th International Conference on Contemporary Computing (IC3 ‘13), pp. 113–117 (2013)
15. Kaiwartya, O., Kumar, S.: GeoPSO: geocasting through particle swarm optimization in vehicular adhoc networks. In: Proceedings of Information Systems and Design of Communication (ISDOC ‘14), ACM (May 2014)
16. Blum, J.J., Eskandarian, A., Huffman, L.J.: Challenges of intervehicle Ad Hoc networks. *IEEE Trans. Intell. Transp. Syst.* **5**(4), 347–351 (2004)

17. Lochert, C., Mauve, M., Füßler, H., Hartenstein, H.: Geographic routing in city scenarios, ACM SIGMOBILE Mobile Computing and Communications Review (MC2R) **9**(1), 69–72
18. Alasmary, W., Zhuang, W.: Mobility impact in IEEE 802.11p infrastructure less vehicular networks. *Ad Hoc Netw.* **10**(2), 222–230 (2012)
19. Kumaresan, G., Adiline Macriga, T., Veenadhari, S.: Vehicle direction based routing for VANET. *Middle East J. Sci. Res.* **24**(12), 3888–3893 (2016). ISSN: 1990-9233 © IDOSI Publications, 2016 <https://doi.org/10.5829/idosi.mejsr.2016.3888.3893>
20. Garnepudi, P., Nageswararao, Lakshminadh, K.: DDSRC: Algorithm for improving QOS in VANET. *Int. J. Recent. Technol. Eng.* (IJRTE), Vol 7(6S5) (2019)

Glacier Surface Flow Velocity of Hunza Basin, Karakoram Using Satellite Optical Data



S. Sivarajani, M. Geetha Priya, and D. Krishnaveni

Abstract High degree of uncertainty about Karakoram glaciers is being reported in literature due to complicated topography and weather. Karakoram glaciers receive their precipitation primarily by westerlies. Glaciers are the real indicators for understanding the impacts of climate change due to global warming. Remote sensing technologies are widely used to study glacier dynamics including estimation of glacier surface flow velocity. The present study focuses on understanding glacier flow dynamics of Hunza basin, Karakoram which consists of surging glaciers. Landsat-8 Operational Land Imager (OLI) Panchromatic data of 15 m resolution have been used to estimate surface flow velocity of glaciers using sub-pixel normalized cross-correlation technique using an open-source software package COSI-CORR(Co-registration of Optically Sensed Images and Correlation) for the Hydrological year 2017–2018. Processed results indicate that surface flow velocity of maximum 120 and 80 m/year has been obtained at basin scale and for Braldu glacier, respectively. The proposed study shows that surface flow velocity is highly dynamic in nature varying from one glacier to another and within the glacier.

Keywords Braldu glacier · Hunza basin · Velocity · COSI-CORR · Karakoram

S. Sivarajani · M. Geetha Priya (✉)
CIIRC, Jyothi Institute of Technology, Bengaluru 560082, India
e-mail: geetha.sri82@gmail.com

S. Sivarajani
e-mail: sivasumathi.95@gmail.com

D. Krishnaveni
Department of Electronics and Communication Engineering, Jyothi Institute of Technology,
Bengaluru 560082, India
e-mail: mailkveni73@gmail.com

1 Introduction

Glacier amendment is widely acknowledged as an indicator for ever-changing climate with diverse studies documenting rapid ice mass changes in mountain regions throughout the globe. Glaciers are the vital supply of water for most of the rivers originating from Himalayas and Karakoram which plays an essential part during the summer season to manage water resources [1]. Karakoram glaciers are unique as they are placed in tropics, at high altitude regions, preponderantly natural depression kind with many glaciers covered with debris. Due to a lack of continuous field observations, rugged topography, and quite a complex climate, the glaciers in the Karakoram are least explored [2]. Literature reports that there are no substantial vicissitudes in terms of Karakoram glacier areas indicating retreat other than advancing/surging in the past decade. To measure and understand various glacier parameters on an oversized scale, remote sensing techniques are ideal as they cover glaciers that are not easily accessible. Within the recent past, the optical satellite data has been explored extensively to measure glacier areas in numerous mountain ranges around the world and several other digital ice mass inventories have been developed [3]. Parameters such as mass balance, surface flow rate, glacier ice thickness, area, meltwater, length, and bed topography are essential to understand the glacier dynamics and other parameters are being investigated by scientific communities either by field measurements or remote sensing methods or combining the both. Climate change impact on glaciers can be linked to glacier properties such as velocity and thickness [4]. Velocity of the glacier and snow cover mapping was very important for several aspects in hydrology [5] and glaciology, as patterns of ice flow play a vital role within the reaction of glacier systems to temperature changes. Ice flow velocity act as a sensitive indicator of glacier state. Glacier mass budget is well facilitated by surface flow velocities [6]. Generally, glacier velocity is derived using feature tracking, SAR interferometry, cross-correlation and sub-pixel correlation techniques. In the proposed study, velocity has been derived with sub-pixel normalized cross-correlation technique for the Hunza basin using remotely sensed optical data.

2 Study Area

Hunza basin ($35^{\circ} 54' 59''$ N and $74^{\circ} 21' 47''$ E) of Karakoram (Fig. 1) is one of the chief sub-basins of the upper Indus basin. Total area of Hunza basin is around $17,880 \text{ km}^2$ with 7159.069 km^2 of glaciated region. The surface elevation ranges from 1147 to 7850 m above mean sea level. Most of the glaciers in Hunza basin hold dominant aspects in north face [7]. Hunza watersheds are treated as representatives of the western Karakoram region [8]. Importance of the basin comes from the fact that many of the glaciers in this basin are surging in nature as reported in literature with 2547 total glaciers in Hunza basin. Spatial-temporal changes have also been identified using deep learning methods for shispar glacier [9]. Recent Study also suggests that

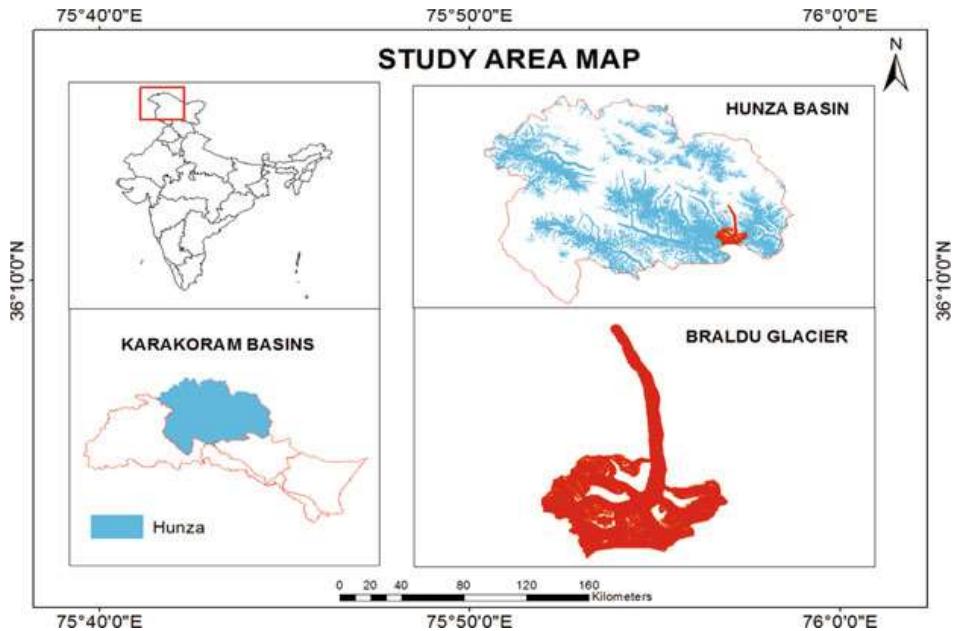


Fig. 1 Study area map

there may be probability of glacier lake outburst flood (GLOF) particularly due to shispar glacier surge in this basin [10]. Based on the past analysis there are various causes for flood in the river basins [11]. Braldu glacier ($36^{\circ} 13' 60''$ N $75^{\circ} 52' 0''$ E) of 176 km^2 is one of the benchmark glaciers with Randolph Glacier Inventory Identification number RGI 60-14.04411 in Hunza basin that comes under surge type. Literature reported accumulation area ratio (AAR) of Braldu glacier is 0.21 [12] with elevation ranging from 3972 to 6528 m above sea level. The topography of Hunza basin (Fig. 2) and Braldu glacier (Fig. 3) of Karakoram are shown below.

3 Data Used

In the present study, Landsat-8 Panchromatic data (HY 2017–2018) of 15 m spatial resolution along with Advanced Spaceborne Thermal Emission and Reflection radiometer (ASTER) Global Digital Elevation Model (GDEM) with spatial resolution of 30 m have been used to estimate surface flow velocity of the glaciers. An updated version of glacier dataset of RGI-Version 6 for glacier outlines was used for glaciated region. Table 1 provides details of the datasets used for the study.

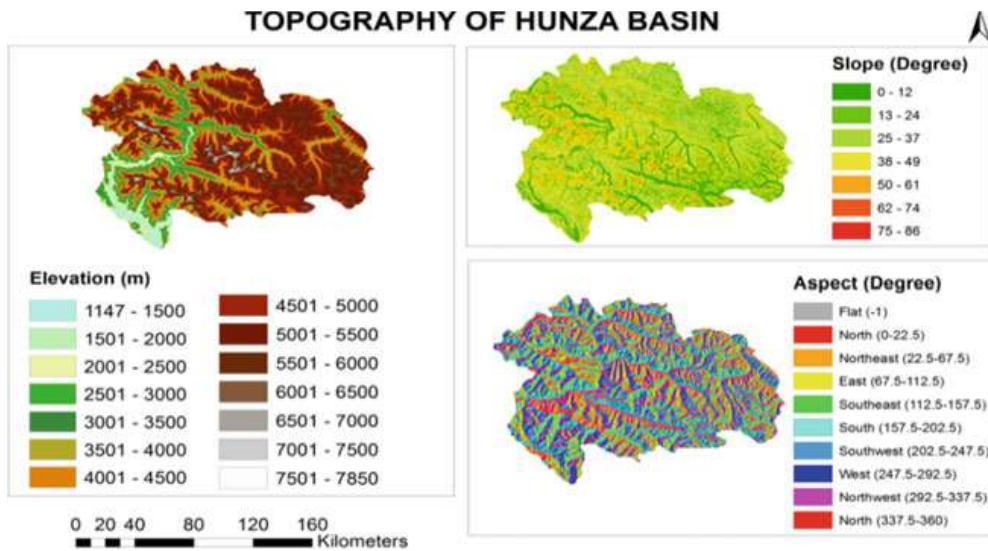


Fig. 2 Topography of Hunza basin

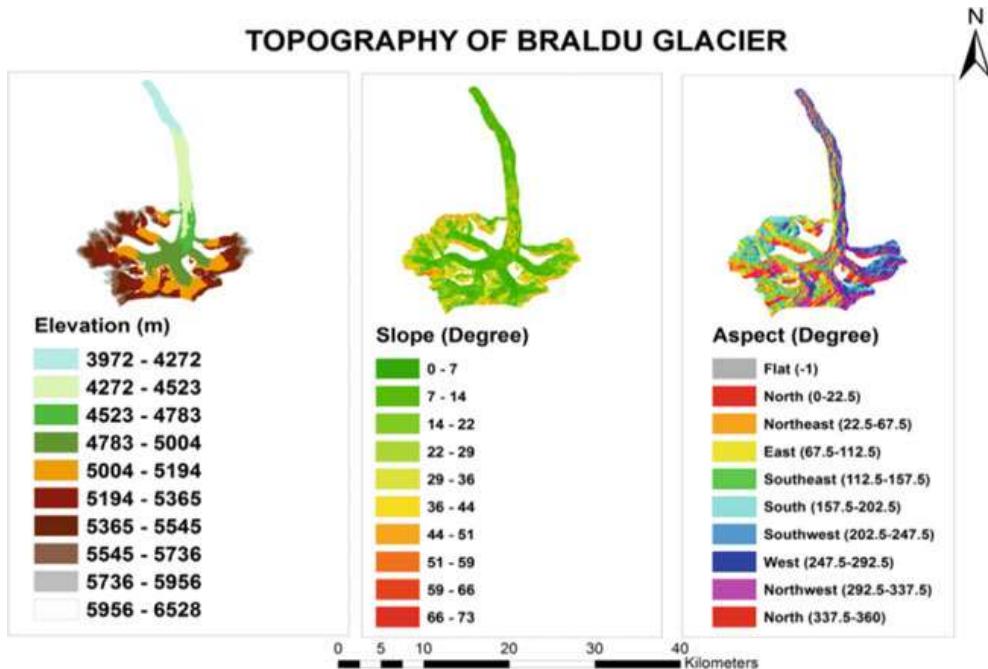


Fig. 3 Topography of Braldu glacier

Table 1 Description of datasets used for the study

Data	Path/row	Date of acquisition	Band number	Resolution (m)
Landsat-8	149/35	November 05, 2017	8	15
		November 24, 2018		
Landsat-8	150/35	November 12, 2017 November 15, 2018	8	15

4 Methodology

The overall methodology employed in the present study is shown in (Fig. 4). For precise flow velocity along with its flow direction at sub-pixel level, COSI-Corr software package developed by Leprince et al. was used [13]. The velocity vectors in the glacier flow direction can also be derived using this technique. This COSI-Corr module is freely available and can be accessed from <http://www.tectonics.caltech.edu>. It involves the application of sub-pixel normalized cross-correlation technique on two multispectral images of different time periods. Frequency correlator was used to correlate two selected time-series images after ortho-rectification process. A 32×32 pixel sliding window with a step size of two pixels was considered for normalized cross-correlation. As a result of the correlation technique, three output components are generated namely EW (East-West), NS (North-South) and SNR (Signal to Noise Ratio). Due to the step size two and spatial resolution (15 m), results at 30 m spatial resolution are generated. In order to avoid uncertainties due to noise, pixels corresponding to SNR less than 0.9 are discarded. Resultant displacement (R) is calculated using (1) Euclidean distance formula by taking square root of the sum of squares of EW and NS displacement.

$$R = \sqrt{NS^2 + EW^2} \quad (1)$$

The annual surface flow velocity (U_s) is estimated using displacement data and acquisition interval between two satellite images (2). For characterizing dynamics of the glacier, surface flow velocity and ice thickness are the major parameters [14].

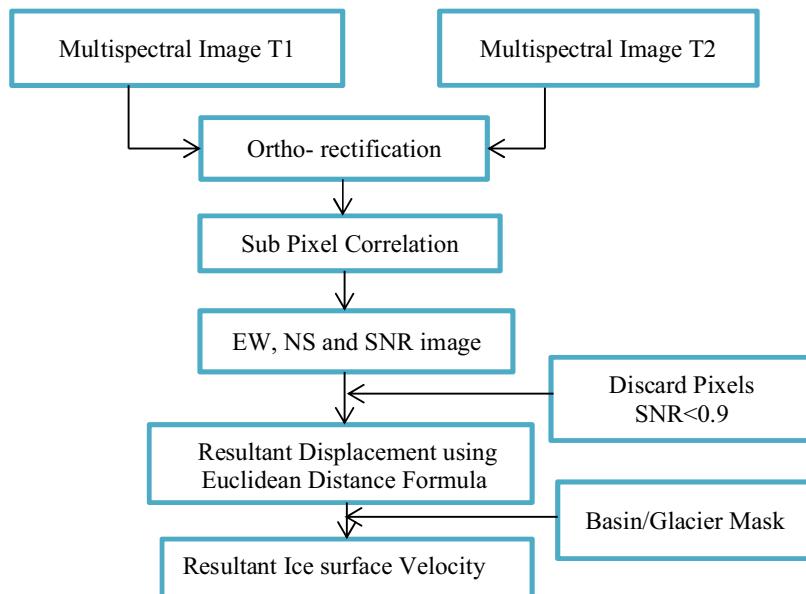


Fig. 4 Flowchart for methodology adopted

$$U_s = \frac{R \times 365}{\text{Number of days between two used images}} \quad (2)$$

5 Results

Using normalized cross-correlation technique discussed under the methodology section, the annual ice surface velocity for Hunza basin including the benchmark glacier has been obtained for HY 2017–2018 using Landsat-8 PAN data of 15 m spatial resolution. Ice surface flow velocity map has been derived for Hunza basin including Braldu glacier on an annual scale (Fig. 5). From the derived velocity results (Fig. 5), the following inferences can be made:

- Surface flow velocity in the glaciated region of the Hunza basin ranges to a maximum of 120 m/year with an average velocity of around 23 m/year for HY 2017–2018. Maximum surface velocity around 90–120 m/year has been observed in few glaciers of the basin along the central flow line close to the snout as well as in close proximity to the tributary glaciers joining the main glacier.
- For Braldu glacier, surface flow velocity ranges to a maximum of 80 m/year with an average velocity of 18 m/year for HY 2017–2018. Surface flow velocity ranges between 10 and 20 m/year near the snout of Braldu glacier. Middle ablation region of Braldu glacier exhibits surface flow velocity of 20–60 m/year due to slope. Braldu glacier shows maximum surface velocity values at only some areas of upper accumulation regions due to thicker ice/crevasses/surging. Large variations in velocity observed mainly due to the complex glacier topography particularly

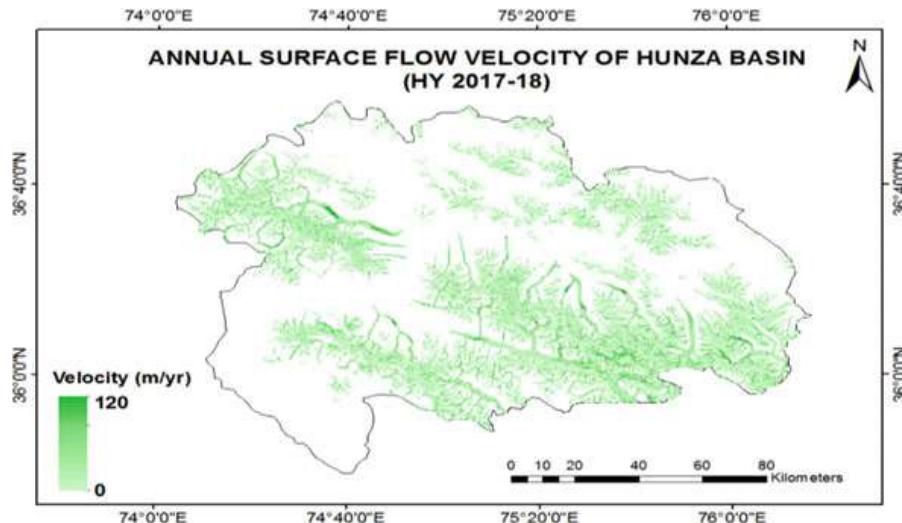


Fig. 5 Annual surface flow velocity map—Hunza basin

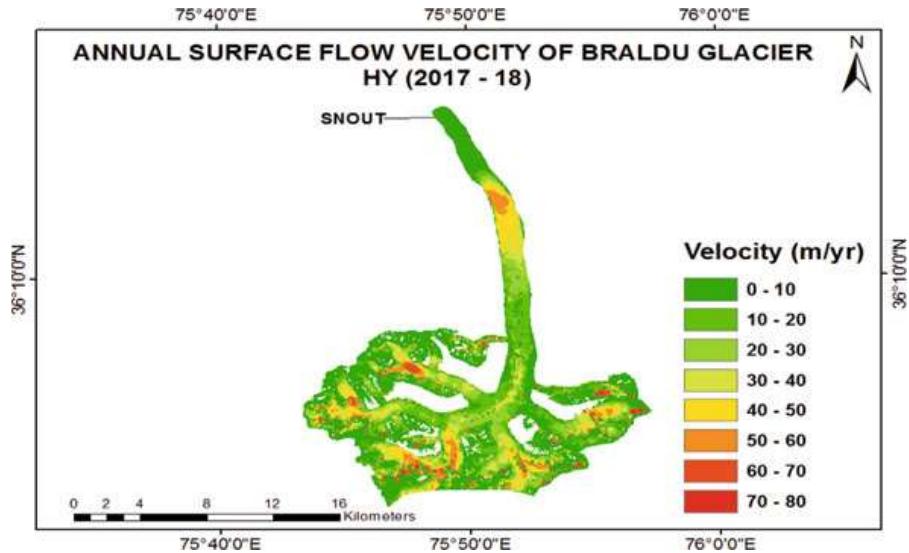


Fig. 6 Annual surface flow velocity map—Braldu glacier

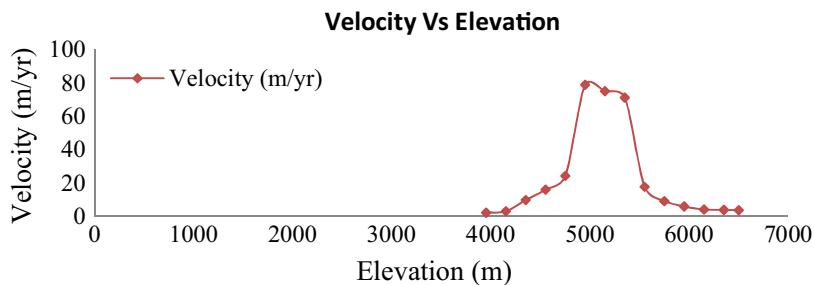


Fig. 7 Spatial variability of Braldu glacier

in terms of slopes/crevasses. Mainly in the Karakoram, the increase in velocity towards the central flow line of the glacier exists due to predominant basal sliding.

Due to the dynamics of the tributary glaciers, the surface flow is not stable on the main trunk of the glacier. For most of the glacier (more than 90%) velocity remains below 40 m/year (Fig. 6). Velocity versus Elevation (Fig. 7) and Distance versus Elevation profiles of Braldu glacier (Fig. 8) show the spatial variability of surface flow indicating that the velocity increases with elevation and reduces at higher elevations where the accumulation region experiences less flow due to thicker ice covered with snow.

6 Conclusion

Annual surface flow velocity for Hunza basin and one of the benchmark glaciers namely, Braldu glacier has been estimated for HY 2017–2018 using sub-pixel

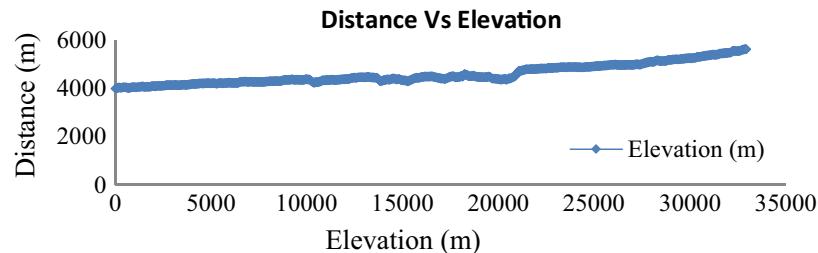


Fig. 8 Distance versus Elevation—Braldu glacier

normalized cross-correlation technique with COSI-CORR open-source package available. Studies using optical data have limitations due to the unavailability of cloud-free data. In this study, the surface flow velocity for the glaciated region of Hunza basin ranges to a maximum of 120 m/year with an average velocity around 23 m/year. For Braldu glacier, surface flow velocity ranges to a maximum of 80 m/year with an average velocity of 18 m/year. Glacier dynamics are largely influenced by surface flow velocity, hence glacier velocity related studies are important to assess the health of glaciers.

Acknowledgements This research work is being financially supported by SPLICE-DST (DST/CCP/NHC/156/2018(G)) under NMSHE Network Programme on Climate Change and Himalayan Cryosphere (DST/CCP/NHC/156/2018(G)). The authors gratefully acknowledge the support and cooperation given by Dr. Krishna Venkatesh, Director, CIIRC, Jyothi Institute of Technology, Bengaluru.

References

1. Hock, R., Rees, G., Williams, M.W., Ramirez, E.: Contribution from glaciers and snow cover to runoff from mountains in different climates. *Hydrol. Process.* **20**, 2089–2090 (2006)
2. Lal, P., Vaka, D.S.: Glacier velocity mapping using Sentinel-2 and Landsat 8 OLI materials & methods references (2018)
3. Kaab, A., Winsvold, S.H., Altena, B., Nuth, C., Nagler, T., Wuite, J.: Glacier remote sensing using sentinel-2. part I: Radiometric and geometric performance, and application to ice velocity. *Remote Sens.* **8** (2016)
4. Minora, U., Bocchiola, D., D’Agata, C., Maragno, D., Mayer, C., Lambrecht, A., Vuillermoz, E., Senese, A., Compostella, C., Smiraglia, C., Diolaiuti, G.A.: Glacier area stability in the Central Karakoram National Park (Pakistan) in 2001–2010: The “Karakoram Anomaly” in the spotlight (2016)
5. Sattar, A., Goswami, A., Kulkarni, A.V., Das, P.: Glacier-surface velocity derived ice volume and retreat assessment in the Dhauliganga basin, central Himalaya—a remote sensing and modeling based approach. *Front. Earth Sci.* **7**, 1–15 (2019)
6. Nagajothi, V., Priya, M.G., Sharma, P.: Snow cover estimation of western himalayas using sentinel-2 high spatial resolution data, *Indian. J. Ecol.* **46**, 88–93 (2019)
7. Bhushan, S., Syed, T.H., Arendt, A.A., Kulkarni, A.V., Sinha, D.: Assessing controls on mass budget and surface velocity variations of glaciers in Western Himalaya. *Sci. Rep.* **8**, 1–11 (2018)
8. Qureshi, M.A., Yi, C., Xu, X., Li, Y.: Glacier status during the period 1973–2014 in the Hunza Basin, Western Karakoram. *Quat. Int.* **444**, 125–136 (2017)

9. Mukhopadhyay, B., Khan, A., Gautam, R.: Rising and falling river flows: contrasting signals of climate change and glacier mass balance from the eastern and western Karakoram. *Hydrol. Sci. J.* **60**, 2062–2085 (2015)
10. Hussain, A.: A brief communication of shispar glacier surge in 2018, Hunza river basin, Pakistan. *Int. J. Adv. Geosci.* **7**, 124 (2019)
11. Gomathi, M., Geetha Priya, M., Krishnaveni, D.: Supervised classification for flood extent identification using sentinel-1 radar data. In: Proceedings of 39th Asian Conference Remote Sensing Enabling Prosper. ACRS 2018, vol. 5, pp. 3277–3284 (2018)
12. Hewitt, K.: Glacier Change, Concentration, and Elevation Effects in the Karakoram Himalaya, Upper Indus Basin. *Mt. Res. Dev.* **31**, 188–200 (2011)
13. Jawak, S.D., Kumar, S., Luis, A.J., Bartanwala, M., Tummala, S., Pandey, A.C.: Evaluation of geospatial tools for generating accurate glacier velocity maps from optical remote sensing data. In: Proceedings, vol. 2, p. 341 (2018)
14. Gantayat, P., Kulkarni, A.V., Srinivasan, J.: Estimation of ice thickness using surface velocities and slope: case study at Gangotri Glacier, India. *J. Glaciol.* **60**, 277–282 (2014)

Diabetic Retinopathy Detection Using Transfer Learning and Deep Learning



Akhilesh Kumar Gangwar and Vadlamani Ravi 

Abstract Diabetic retinopathy is one of the major causes of blindness in the population aged 20–65. In this paper, we address the problem of automatic diabetic retinopathy detection and proposed a novel deep learning hybrid to solve the problem. We use transfer learning on pre-trained Inception-ResNet-v2 and added a custom block of CNN layers on top of Inception-ResNet-v2 for building the hybrid model. We evaluated the performance of the proposed model on Messidor-1 diabetic retinopathy dataset and APTOS 2019 blindness detection (Kaggle dataset). Our model performed better than other published results. We achieved a test accuracy of 72.33% and 82.18% on Messidor-1 and APTOS dataset, respectively.

Keywords Diabetic retinopathy · Image classification · Deep learning · Inception ResNet-v2

1 Introduction

Diabetic retinopathy is one of the major causes of the rise in global blindness. As per records, there are as many as 415 million diabetic patients worldwide. To prevent blindness, diabetics should be screened every year. A common practice for detecting diabetic eye disease is to examine the fundus image and assess the severity of

A. K. Gangwar · V. Ravi ()

Center of Excellence in Analytics, Institute for Development and Research in Banking Technology, Castle Hills Road 1, Masab Tank, Hyderabad 500057, India
e-mail: rav_padma@yahoo.com

A. K. Gangwar
e-mail: gangwar.akhilesh1993@gmail.com

A. K. Gangwar
School of Computer and Information Sciences,
University of Hyderabad, Hyderabad 500046, India

the disease. The severity depends on the type of retinopathy (e.g., microaneurysms, hemorrhages, hard exudates, etc.), which are indicators of eyeball bleeding and exudation.

Despite having nearly 127,000 ophthalmologists in India, nearly 45% of patients suffer from blindness before being diagnosed . This should not happen because diabetic retinopathy is completely preventable. To improve this situation, the Google research team worked closely with EyePACS in the USA and three eye hospitals in India [1], Aravind Eye Hospital, Sankara Nethralaya, and Narayana Nethralaya. Therefore, building better deep learning-enabled software for automatic diabetic retinopathy detection becomes necessary.

There is an increasing progress in collaboration between a diabetes care physician and an ophthalmologist. There are a few opportunities for diabetic patients to undergo fundus examinations in areas with few doctors. Further, there is a shortage of ophthalmologists who can diagnose and treat diabetic retinopathy. However, the level of dispensary ophthalmologic monitoring of patients with diabetes today remains unsatisfactory even in economically highly developed countries. For example, in the USA, about a third of diabetics have never undergone an ophthalmologic examination, and according to other data, only half have been examined by an eye doctor over the past year. The quality of the ophthalmological examination also does not always meet the requirements. Serious shortcomings were noticed in the professional training of ophthalmologists, who are often poorly informed about the possibilities of modern methods of treatment of diabetic retinopathy. Thus, according to a survey of 1655 general ophthalmologists of the State of New York, many of them did not consider it necessary to refer patients with proliferative diabetic retinopathy to laser treatment.

There is a problem of manually diagnosing DR from retinal images because it is a time-consuming and number of specialists are less. So to solve this problem, it is good if we build automatic diabetic retinopathy detection system that can work without the help of a doctor. To this end, several researchers have applied deep convolutional neural network (CNN) [2] to diagnose diabetic retinopathy. However, proposed models employed very deep CNN models, GoogleNet [3], and VGGNet [4]. These works achieved good accuracy but great improvement is still possible. The present work therefore focused on applying end-to-end, accurate CNN-based hybrid model that utilizes the power of transfer learning.

The major contributions of the present paper are as follows:

1. We proposed computationally efficient CNN Inception-ResNet-v2 hybrid. We used Inception-Resnet network for transfer learning. Then, we added an ensemble-based inception block on top of the Inception-ResNet.
2. One benchmark dataset, namely Messidor-1, and a new dataset, APTOS 2019, are analyzed to validate the effectiveness of the proposed Inception ResNet-v2 hybrid model.
3. Our proposed model achieved better accuracy compared to the state of the art. We compared the accuracy score of our model with accuracy score of GoogleNet [34] on Messidor dataset [5] and APTOS 2019 blindness detection (Kaggle dataset) [6] .

2 Literature Survey

Numerous works were reported in literature with respect to automatic diabetic retinopathy detection. End-to-end automatic diabetic retinopathy detection can be categorized broadly into two categories based on techniques used. First one using traditional machine learning algorithms in which we have to use external feature extraction techniques using image processing and second one is deep learning-based diabetic retinopathy detection where automatic feature extraction takes place.

2.1 Handcrafted Feature Extraction-Based Retinopathy Detection Using Traditional Machine Learning Algorithms

Nijalingappa and Sandeep [7] proposed diabetic retinopathy detection using KNN algorithm. First, they extracted features using image processing and then applied these features to KNN. They achieved 87% accuracy on local dataset. Seoud et al. [8] proposed new feature extraction method, namely dynamic feature shape using image processing. They used random forest and decision tree for classification. Their method yielded high sensitivity score. Singalavania et al. [9] trained multilayer perceptron using extracted features from diabetic retinopathy image. They first extracted the features using region growing segmentation algorithm after which they trained the MLP. Their proposed framework achieved sensitivity and specificity of 74.8% and 82.7%, respectively on 336 eye images. Kahai et al. [10] proposed an automatic diabetic retinopathy detection system. They also extracted features using image processing to detect the diabetic retinopathy through Bayesian criteria. Their method successfully yielded high sensitivity and good specificity.

2.2 Automatic Feature Extraction-Based Retinopathy Detection Using Deep Learning Algorithms

Nowadays, researchers are using deep learning for building end-to-end automatic diabetic retinopathy detection system. Due to the availability of large amount of data and transfer learning concepts, deep learning-based system has extra edge over traditional machine learning system. There are several research works in literature that deal with deep learning-based retinopathy detection. Some of the important works are reviewed below.

Pratt et al. [11] proposed convolutional neural network-based diabetic retinopathy detection system. They have used 13 CNN layers. They applied data augmentation techniques for handling data imbalance problem. They trained on 80,000 images of train data and reported 95% sensitivity on 5000 images of test data of EyePACS DR

dataset. Rufaida et al. [12] used residual neural network to classify retinal image. They used the EyePACS DR dataset for building end-to-end framework. Evaluation metric to check the performance of model is quadratic weighted kappa (QWK) [13]. They reported 0.5104 kappa score. Zhang et al. [14] proposed ResNet for diabetic retinopathy images classification. They added layers to intermediate hidden layers of ResNet. By adding these custom layers, they improved the performance of the model. They trained model on EyePACS DR dataset. They achieved quadratic weighted kappa score 0.73. Alban and Gilligan [15] used GoogleNet and AlexNet to classify the retina images. First, they performed preprocessing of retina image and then passed it to GoogleNet and AlexNet. They used EyePACS DR dataset for training and testing the model. They reported 0.68 and 0.78 AUC score using AlexNet and GoogleNet, respectively. Lam et al. [16] used transfer learning for diabetic retinopathy detection. They used GoogleNet and AlexNet models that are pre-trained on ImageNet dataset. They have built the model using Messidor-1 diabetic retinopathy dataset. GoogleNet has achieved a maximum accuracy 66.03% on Messidor-1 dataset.

3 Proposed Model

The whole pipeline of our proposed method is depicted in Fig. 1. It consists of four stages: image preprocessing, data augmentation, Inception-ResNet transfer learning, and prediction.

First, we input the color retina image and preprocessed it using blurring, bounding box and cropping as depicted in Fig. 2. Then, we resized it a particular size that is suitable for deep learning models (299×299 RGB scale because it is default input size of Inception-ResNet-v2). After that, we applied Inception-Resnet and hybridized it with an additional layer for fine-tuning the model on augmented train data set. Finally, we will use the prediction score of softmax layer to predict the class.

3.1 Data Prepossessing

We used Messidor-1 dataset [5] and APTOS 2019 blindness detection (Kaggle dataset) [6]. They have images of different pixel sizes. We want to have images of same size for CNN because CNN cannot take input of different dimensions. So first we have to preprocess the image and convert it to a suitable size. We have performed three preprocessing steps, namely blurring, getting coordinates of bounding box, cropping, and resizing as depicted in Fig. 2.

Bounding box helps to decide image portion and removes extra background part. It is obtained by the following methods.

Fig. 1 Pipeline overview of diabetic retinopathy detection

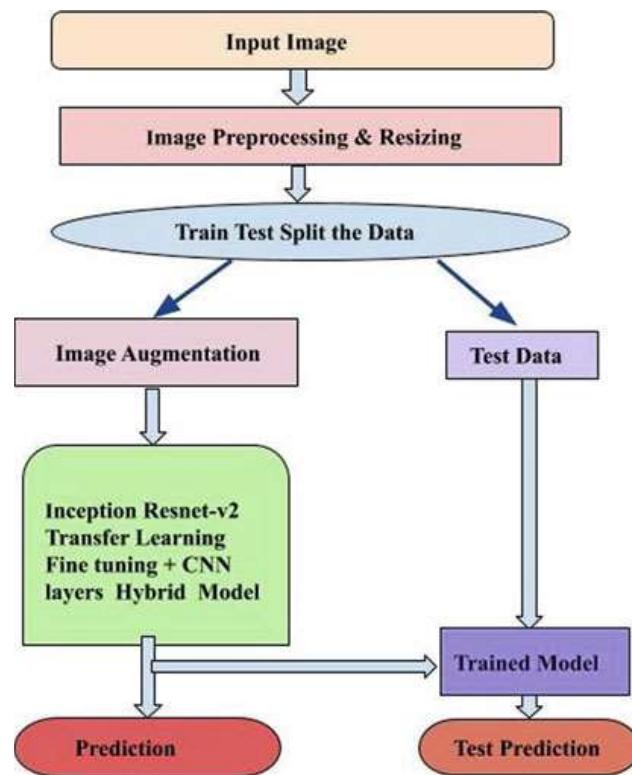
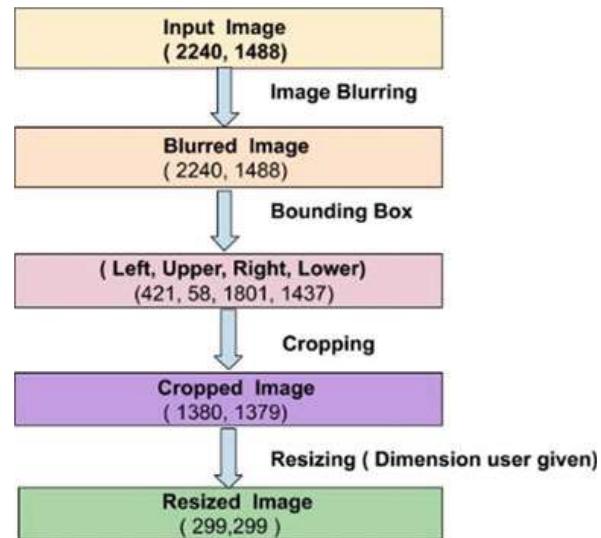


Fig. 2 Data preprocessing steps of diabetic retinopathy image



Algorithm 1 Calculation to get the coordinates of the boundary box of the image

```

1: First calculate the height and width of image
2: Left = maximum(width-(width-height)//2,0)
3: Upper = 0
4: Right = Minimum(width - (width - height)//2, width)
5: Lower = height
Return (Left, Upper, Right, Lower)

```

3.2 Data Augmentation

An invariant attribute of convolutional neural network means, it can classify an object even if it is placed in a different place. CNN is invariant to translation, size, viewpoint, illumination (or all of above) [17]. This is essentially a prerequisite for data enhancement. In a real-life scenario, we might have a collection of data shots in a limited scene. Future data may have different views angle so data augmentation is necessary to build robust model.

Common techniques used for data enhancement are rotating the image, cropping the image, image color enhancement, changing the image size, and enhancing the image noise (usually use Gaussian noise, salt pepper noise).

Retinopathy dataset is an unbalanced dataset and it has less number of images. There is a high chance of overfitting if data is less. So we will enhance our data through data augmentation techniques. We have used the horizontal flipping, rotation, zoom, width shift, height shift, and vertical flipping techniques for data enhancement (augmentation).

3.3 Novel Hybrid Architecture of Inception-ResNet-v2 and Custom CNN Layers

We used Inception-ResNet-v2 [18] for transfer learning. We have taken pre-trained Inception-ResNet-v2 model that is trained on ImageNet dataset and we removed last layers of that model. We added a few more layers on top of that. In Fig. 3, we depicted the architecture of proposed hybrid deep learning architecture of Inception-ResNet-v2 and custom CNN. Thus, the proposed hybrid model has mainly two building blocks. First one is the pre-trained inception net and the second one is the custom convolutional block followed by fully connected layers.

In the custom block, we built a naive custom inception block that is inspired by Google Inception block. In this block, we used four different convolutional layers with different filters followed by batch normalization and dropout for each layer. After that, we concatenated output of these layers. Then, we used global max-pooling operation for converting 3-dimensional output to 1-dimensional output. After that, we used fully connected layer followed by softmax.

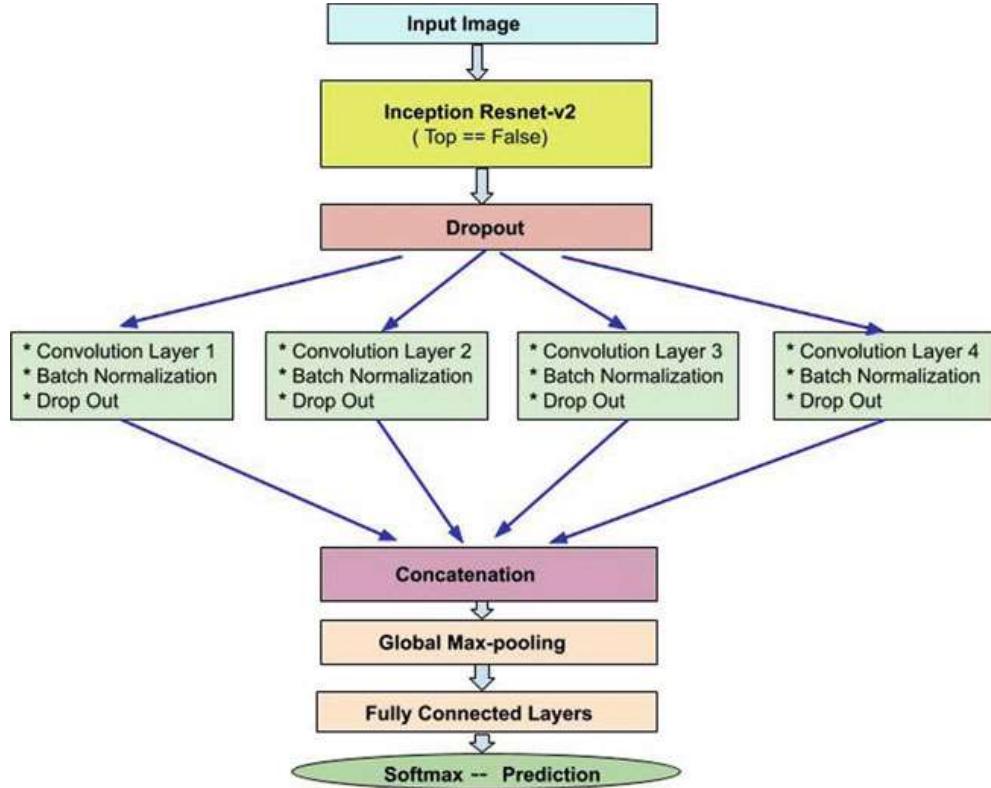


Fig. 3 Proposed hybrid architecture of Inception-ResNet-v2 and custom CNN layers

We preprocessed the image and fed the input to Inception-ResNet-v2 that is pre-trained on ImageNet data set. We removed the last layers of the Inception-ResNet-v2 and added the custom block of CNN followed by global max-pooling, fully connected layers and softmax. In the custom block, we used four different convolutional layer with different filter sizes so that there is no need to tune the filter that means if we need any different filter for new data, then there is no need to change because we are using ensemble of four different filter size here. Thereafter, the output of four parallel convolutional layers is concatenated and passed to the next layer. We adopted softmax output for class label prediction.

3.4 Experimental Setting

According to the typical workflow depicted in Fig. 1, we applied preprocessing first. Then, we converted the image of different sizes to 299×299 RGB scale as it is default input size for Inception-ResNet-v2 model. We performed data augmentation using ImageDataGenerator library [19] of Keras framework. We have used 3×3 , 5×5 , 7×7 , and 9×9 filter size for custom inception block. We trained model using Adam optimizer [20] with learning rate of 0.0001, dropout rate 0.2, batch size of 30, and the number of epochs 200.

Fig. 4 Retina image of different grades of Messidor-1 dataset



Table 1 Messidor-1 datasets images distribution

Grade	Description	Number of images
Retinopathy Grade 0	$(\mu A = 0) \text{ AND } (H = 0)$	546
Retinopathy Grade 1	$(0 < \mu A \leq 5) \text{ AND } (H = 0)$	153
Retinopathy Grade 2	$((5 < \mu A < 15) \text{ OR } (0 < H < 5)) \text{ AND } (\text{NV} = 0)$	247
Retinopathy Grade 3	$(\mu A \geq 15) \text{ OR } (H \geq 5) \text{ OR } (\text{NV} = 1)$	254

μA : number of microaneurysms

H : number of hemorrhages

$\text{NV} = 1$: neovascularization

$\text{NV} = 0$: no neovascularization

Table 2 Train-test data distribution according to retinopathy grades of Messidor-1 dataset

Grade	Train data	Test data
Retinopathy Grade 0	409	137
Retinopathy Grade 1	95	38
Retinopathy Grade 2	185	62
Retinopathy Grade 3	191	63

4 Data Set Description and Evaluation Metrics

We used Messidor-1 dataset [5] and APTOS 2019 blindness detection (Kaggle dataset) [6] to train the model for end-to-end framework for diabetic retinopathy detection. Messidor-1 dataset has 1200 eye fundus color numerical images divided into four classes (Fig. 4; Table 1). These images taken from different cameras. Images have different pixel sizes like 2304×1536 , 1440×960 or 2240×1488 pixels. APTOS 2019 blindness detection (Kaggle dataset) has 3662 images divided into 5 classes. We have used the training data of Kaggle competition because it has labels.

We created a training dataset and a test dataset using hold-out split method in the 75:25 ratio. Distribution of images in training and test datasets is depicted in Tables 2 and 3.

Table 3 Train-test data distribution according to retinopathy grades of APTOS 2019 blindness detection (Kaggle dataset) [6]

Grade	Train data	Test data
Retinopathy Grade 0	1534	271
Retinopathy Grade 1	314	56
Retinopathy Grade 2	849	150
Retinopathy Grade 3	164	29
Retinopathy Grade 4	251	44

5 Result and Discussion

We used accuracy score to measure the effectiveness of the proposed model. The results are evaluated on 300 test images of messidor-1 dataset and 550 images of APTOS 2019 blindness detection (Kaggle dataset). Lam et al. [16] also proposed GoogleNet for diabetic retinopathy detection using Messidor-1 data. We cannot directly compare our results with their results [16] despite the fact that they too used hold-out method of testing because their training and test datasets created by them are neither publicly available nor mentioned in their paper. Nevertheless, we obtained significantly higher accuracy compared to them. The test data results of both datasets are presented in Tables 4 and 5.

Our model outperformed the Lam et al. [16] methods and other old methods on Messidor-1 dataset because we exploited the advantages of inception block and residual connections. Inception-ResNet has advantages of inception net as well as residual connections that is why increasing the depth of the network does not decrease the performance. Due to inception block, there is no need to decide the filter size because we have taken concatenation of outputs of most important filter sizes. Due to inception block and residual block, Inception-ResNet-v2 is better feature extractor. On top of the Inception-ResNet-v2, we proposed other CNN block that is inspired

Table 4 Test data accuracy on Messidor-1 dataset

Model	Accuracy score
GoogleNet (transfer learning)	66.03 %
Hybrid Inception ResNet-v2 (transfer learning)	72.33 %

Table 5 Test data accuracy on APTOS 2019 blindness detection (Kaggle dataset)

Model	Accuracy score
Hybrid Inception ResNet-v2 (transfer learning)	82.18 %

by inception block. We have used four CNN layers of different filter sizes and concatenated the output of these CNN layers. Due to this ensembling, the custom block become better than normal CNN layers.

6 Conclusion and Future Directions

In this paper, we proposed hybrid Inception-ResNet-v2 deep learning architecture for diabetic retinopathy detection problem. In the proposed hybrid, we invoked the pre-trained Inception-ResNet-v2 and on top of that we added a custom CNN block. We fine-tuned the whole network. We evaluated the effectiveness of the proposed model on Messidor-1 dataset and ATPOS Kaggle dataset. We achieved better accuracy compared to an existing method on Messidor-1 dataset. We obtained 6.3% more accuracy compared to the GoogleNet.

In future, we would like to use generative adversarial network (GAN) [21] for data oversampling instead of just augmentation. If we generate new retinopathy images artificially, then it will be very helpful to balance the data and its distribution. We would also like to investigate ensembling of different transfer learning model to improve the performance.

References

1. Eyepacs Google Rethink. <http://www.eyepacs.com/blog/eyepacs-helps-google-rethink-the-future-of-eye-health>
2. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
3. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
4. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)
5. Messidor Dataset. <http://www.adcis.net/en/third-party/messidor/>
6. APTOS Dataset. <https://www.kaggle.com/c/aptos2019-blindness-detection/data>
7. Nijalingappa, P., Sandeep, B.: Machine learning approach for the identification of diabetes retinopathy and its stages. In: 2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), pp. 653–658 (2015). <https://doi.org/10.1109/ICATCCCT.2015.7456965>
8. Seoud, L., Hurtut, T., Chelbi, J., Cheriet, F., Langlois, J.M.P.: Red lesion detection using dynamic shape features for diabetic retinopathy screening. IEEE Trans. Med. Imaging **35**(4), 1116–1126 (2016). <https://doi.org/10.1109/TMI.2015.2509785>
9. Singalavani, A., Supokavej, J., Bamroongsuk, P., Sinthanayothin, C., Phoojaruenchanachai, S., Kongbunkiat, V.: Feasibility study on computer-aided screening for diabetic retinopathy. Jpn. J. Ophthalmol. **50**(4), 361–366 (2006). <https://doi.org/10.1007/s10384-005-0328-3>
10. Kahai, P., Namuduri, K.R., Thompson, H.: A decision support framework for automated screening of diabetic retinopathy. Int. J. Biomed. Imaging **2006**, 1–8 (2006). <https://doi.org/10.1155/ijbi/2006/45806>

11. Pratt, H., Coenen, F., Broadbent, D.M., Harding, S.P., Zheng, Y.: Convolutional neural networks for diabetic retinopathy. *Procedia Comput. Sci.* **90**, 200–205 (2016). <https://doi.org/10.1016/j.procs.2016.07.014>
12. Rufaida, S.i., Fanany, M.I.: Residual convolutional neural network for diabetic retinopathy. In: 2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS), pp. 367–374 (2017). <https://doi.org/10.1109/ICACSIS.2017.8355060>
13. Cohen, J.: Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychol. Bull.* **70**(4), 213–220 (1968). <https://doi.org/10.1037/h0026256>
14. Zhang, D., Bu, W., Wu, X.: Diabetic retinopathy classification using deeply supervised ResNet. In: 2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), pp. 1–6 (2017). <https://doi.org/10.1109/UIC-ATC.2017.8397469>
15. Alban, M., Gilligan, T.: Automated detection of diabetic retinopathy using fluorescein angiography photographs. Report of Stanford Education (2016). http://cs231n.stanford.edu/reports/2016/pdfs/309_Report.pdf
16. Lam, C., Yi, D., Guo, M., Lindsey, T.: Automated detection of diabetic retinopathy using deep learning. *AMIA Summits Transl. Sci. Proc.* **2017**, 147 (2018)
17. Iesmantas, T., Alzbutas, R.: Convolutional capsule network for classification of breast cancer histology images. In: International Conference Image Analysis and Recognition, pp. 853–860. Springer (2018)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
19. ImageDataGenerator keras. <https://keras.io/preprocessing/image/>
20. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
21. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)

Prediction of Admissions and Jobs in Technical Courses with Respect to Demographic Location Using Multi-linear Regression Model



Anjali Mishra, Aishwary Kumar, Shakti Mishra, and H. A. Sanjay

Abstract With more employment opportunities, we have seen tremendous growth in number of institutions which offer professional courses across globe. One side, there are very well-established institutes where infrastructure, faculty profile and teaching facilities have established a niche, while on the other side, new institutes, continuously looking for recognition from government bodies on the basis of infrastructure only. Studies published on various online and offline surveys show that with increased number of institutes offering professional courses like engineering, the employment potential has come down. This paper describes a multiple linear regression-based model which can predict employment potential of different courses according to a specific region. The model includes the job employment potential as a contributing parameter to approve government recognition to such institutes. The proposed work can help in balancing the number of institutes at a location which are more promising in providing jobs to engineering graduates. Our work predicts employment potential rate of each course. The number of admissions to an institute and number of students getting an employment in campus have been considered as two parameters. These values are used to provide the employment potential for each course. If the predicted value of employment potential of the requested course lies within the range of the value of employment potential at that specific location, then course can be considered for approval. We have observed that our proposed work predicts the employment potential with an accuracy of around 92%.

A. Mishra
HSBC Technologies, Pune, India
e-mail: anjali.mishra21feb@gmail.com

A. Kumar (✉)
CodeParva Technologies, Bengaluru, India
e-mail: aishwary08@gmail.com

S. Mishra · H. A. Sanjay
Nitte Meenakshi Institute of Technology, Yelahanka, Bengaluru, India
e-mail: prof.shaktimishra@gmail.com

H. A. Sanjay
e-mail: sanjay.ha@nmit.ac.in

Keywords Machine learning · Multiple linear regression model · Prediction

1 Introduction

In developing countries, the number of engineering graduates passing each year does not comply with the number of graduates getting employed. For instance, in China a study proclaims that a record breaking 8 million students had graduated from Chinese universities in 2017 [1]. This figure crosses twice the number of students graduating in USA. Despite the explosive increase in number of students, unemployment rate has remained quite stable over the years [2]. The proposed work aims at determining future admissions and jobs at a demographic location for different courses by using multiple linear regression technique of machine learning. Using this prediction, government bodies can get an idea of potential courses which are most promising for providing jobs in an area. In this paper, we have provided insight of machine learning followed by multi-linear regression algorithm chosen for prediction of jobs.

2 Literature Survey

2.1 *Machine Learning Techniques*

Machine learning models help to determine or predict the result for similar data. Machine learning algorithms are suitable in condition where theoretical knowledge of the problem is not available but there are relevant information and observations from the past. In the case of machine learning, a model is developed using observations and analytical results [3]. These algorithms give accurate results and adapt themselves to the changing dataset itself. There are two types of machine learning approaches: supervised learning and unsupervised learning [4]. In supervised learning, input and output data is known, the system maps these values and finds a relationship between them. While in the case of unsupervised learning, only the input data is known and system has to find the output data by analyzing the structure. Supervised learning has two categories: classification and regression. In classification, output is a binary value, while regression gives a continuous range of output [5]. Unsupervised learning has only one category which is clustering. Various machine learning techniques are available for data mining such as decision trees, Bayesian Network, K-Nearest Neighbor and Support Vector Machine [6]. Machine learning assumes that the accuracy can be increased by apprehending the training data with more observations. But with the increase in data various challenges emerge [5].

There are a number of challenges associated with machine learning such as those enlisted below:

- For large dataset, we often reach a situation where data of one category dominates the data repository which leads to class imbalance.
- As the number of dimensions increase, the processing performance of algorithm is affected. This situation is called the curse of dimensionality. In case of large datasets, non-linearity exists between variables which makes it difficult to correlate features.

Regression is the technique used to predict or learn numeric features. There are various types of regression such as locally weighted regression, rule-based regression, projection pursuit regression, instance-based regression, multiple linear regression and recursive regression [7]. Parametric regression is the regression technique which imparts a linear relationship between dependent (y) and independent (x) variables. The linear relationship between these models can be given by Eq. (1):

$$y = \beta_0 + \beta_1 * xi_1 + \beta_2 * xi_2 + \beta_3 * xi_3 + \dots + \beta_p * xi_p + \varepsilon \quad (1)$$

where x 's represent the feature values, y represents the output, β represents the coefficients and ε represents the error induced during prediction.

Different machine learning algorithms have changing performance rates on different datasets [8]. It depends on the dataset which regression technique would be more suitable for the problem. A study carried out [8] suggests that based on the type of data, a ranking list is created for different regression approaches. This can be used to identify the correct fit for the problem proposed.

2.2 Similar Work

In the year 2016, Wang and Shi [9] used a predictive model to predict the admission lines of College Entrance Examination based on machine learning. In this paper, authors predicted the admission line of college entrance examination based on the machine learning algorithms: Adaboost and Random Forest. The paper indicated that the Adaboost gave better result with more accurate values.

In 2016, Giri et al. [10] proposed a model to predict the placement using K-Nearest Neighbors Classifiers. The outcome of paper was a model imparting binary result value of yes or no stating whether student was placed or not.

Application of improved Multiple Linear Regression Method in Oilfield Output Forecasting by L. Guo and X. Deng provides an insight on explaining the feature significance in regression [11]. Authors found out that the prediction accuracy mainly depends on the linear relationship of the output variables and input variables. They used oilfield output forecasting to show how features can have an impact on the output variable.

After examining and analyzing these papers, we infer that the papers mentioned above for prediction of admission line and placement considered an individual student

and not on an institution level. Models proposed for these predictions used classification techniques which give the output as a fixed range of values not a continuous range.

3 Preliminaries

3.1 Multiple Linear Regression

Multiple linear regression technique involves two kind of variables: dependent and independent. Independent variables are the variables whose values affect the dependent or the target variable and dependent variable is the output. When the number of independent variables is more than two, then multiple linear regression comes in picture. The multiple linear regression model is given by Eq. (2).

$$Y = X\beta + \varepsilon \quad (2)$$

where Y denotes the target matrix with values, i.e., the set of output values from the historic data. It is given by a matrix as shown below:

$$[y_1 \ y_2 \ \dots \ y_n] \quad (3)$$

X denotes the set of dependent variables or the input features.

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{nk} \end{bmatrix} \quad (4)$$

An extra column of 1s is used with an assumption to minimize the error rate induced. β is the set of coefficients which determines the factor by which independent variables affect the final output, i.e., the dependent or target variable.

$$[\beta_1 \ \beta_2 \ \dots \ \beta_n] \quad (5)$$

ε represents the vector of error which is induced with each set of values and it is as shown below

$$[\varepsilon_1 \ \varepsilon_2 \ \dots \ \varepsilon_n] \quad (6)$$

Considering the above values, we can rewrite the equation for multiple linear regression as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix} * \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \quad (7)$$

To find out the value of coefficient vector such that the error value is minimized, we have used least square method, where we try to reduce the error involved.

$$\varepsilon' \varepsilon = \begin{bmatrix} \varepsilon_1 & \varepsilon_1 & \cdots & \varepsilon_n \end{bmatrix} * \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} = \sum_{i=1}^N \varepsilon_i^2 \quad (8)$$

Least square method considers square of error so that every possible error is covered without a chance of positive and negative errors canceling out each other

$$\min_b \varepsilon' \varepsilon = (y - X\beta)'(y - X\beta) \quad (9)$$

$$\min_b \varepsilon' \varepsilon = y'y - 2\beta'X'y + \beta'X'X\beta \quad (10)$$

To find the value of beta such that error is minimized, we partially differentiate the above equation with respect to beta and equate it to zero. Thus, we get

$$\frac{d\varepsilon' \varepsilon}{d\beta} = -2X'y + 2X'X\beta = 0 \quad (11)$$

$$X'X\beta = X'y \quad (12)$$

$$\beta = (\mathbf{X}\mathbf{X}')^{-1}\mathbf{X}'\mathbf{y} \quad (13)$$

Once, we get the value of beta vector, we can find the value of target variable easily.

4 Methodology

We propose a multiple linear regression-based analysis which predicts the number of admission and jobs. By correlating this data, employment potential of different courses is determined. For the prediction of admissions and jobs, different factors reflecting the quality of education are used. These factors are listed in Table 1.

Table 1 Table for features and symbols used to represent them

S. No.	Features for admission prediction	Features for placement prediction	Symbols
1	College rank	Number of companies visiting for recruitment	γ
2	Cutoff rank	Number of partner companies	δ
3	Placement percentage	Quality of placement training	ε
4	Accreditation	Average salary package	ζ

We can rewrite the multiple linear regression as shown in Sect. 2 for our proposed model. The rewritten equation will be as follows:

$$y = \alpha + \beta_0 * \gamma + \beta_1 * \delta + \beta_2 * \varepsilon + \beta_3 * \zeta \quad (14)$$

where α = intercept, y = target class (no. of admissions/placements).

For admission prediction,

$$\begin{aligned} y &= f(\alpha, \beta, \gamma, \delta, \varepsilon, \zeta) \\ \text{such that } \beta &\in \{\beta_0, \beta_1, \beta_2, \beta_3\} \\ \beta_0 &\propto \gamma, \beta_1 \propto \delta, \beta_2 \propto \varepsilon, \beta_3 \propto \zeta \end{aligned} \quad (15)$$

For placement prediction,

$$\begin{aligned} y &= f(\alpha, \gamma, \delta, \varepsilon, \zeta) \\ \beta &\in \{\beta_0, \beta_1, \beta_2, \beta_3\} \\ \beta_0 &\propto \gamma, \beta_1 \propto \delta, \beta_2 \propto \varepsilon, \beta_3 \propto \zeta \end{aligned} \quad (16)$$

4.1 Workflow

Initially, the user fills in the details with accordance to the factors mentioned in Table 1. Once, the client submits the information, a request is sent to the server. This server has a python script running in background with the implementation of multiple linear regression. Script fetches the historical data from the database and trains the model with the fetched data. This is used to predict admissions and jobs. By correlating admissions and jobs, employment potential for different courses at a location is determined. Result is rendered on the screen to the user as shown in Fig. 1.

Figure 2 explains the flowchart of the model. Initially, the python script fetches the data from the database on the basis of location of the institute for which the approval is required. The fetched data is split into features and the target values. Multiple linear regression takes data as numerical value, but the information in Table 1 is

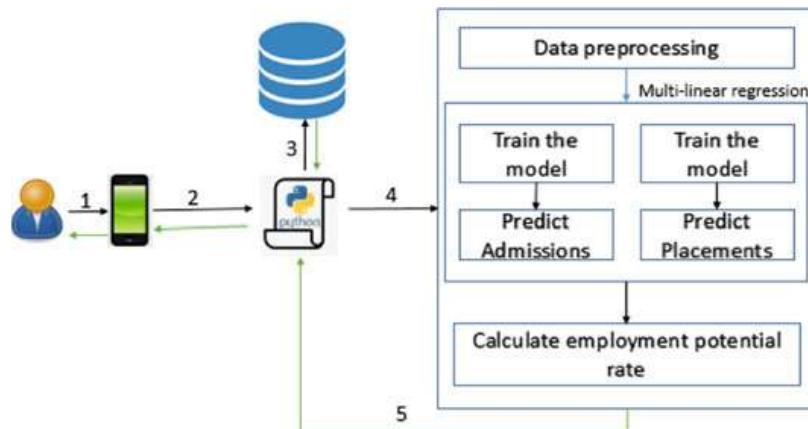


Fig. 1 Workflow of the proposed model



Fig. 2 Flow of the python script

not numerical. Thus, data preprocessing is required to handle this categorical data. After the preprocessing phase, multiple linear regression is trained for admission and placement data, respectively. These values are predicted and employment potential for each course is calculated.

5 Implementation and Result Analysis

The proposed model is used for the prediction of admissions and jobs at three different locations in India. For each location, three colleges are considered. 90% of the collected historical data is used for training the multiple linear regression model, while 10% of the data is kept for validation purpose. This testing data is used to check the accuracy of the result by comparing the actual value and the predicted value.

Figure 3 depicts a graph with predicted and actual value for admission, placement and employment potential for three different course for college A, B and C. Blue, gray and navy-blue lines indicate the actual value of admission, placements and employment potential. These values are from the testing data for which the model was not trained. Orange, yellow and green lines represent the predicted values.

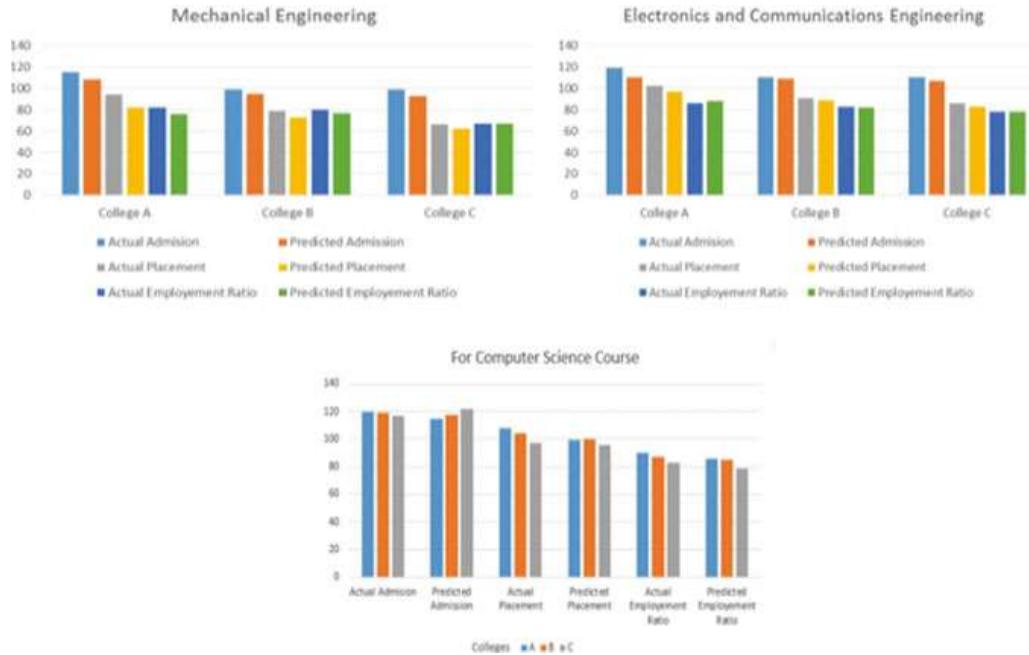


Fig. 3 Resultant graph for actual and predicted values for three different courses

6 Conclusion

Job employment potential for a specific location holds a key position in inferring the employment ratio of our nation. So, the problem of plethora of unnecessary branches and institutes and the production of unskilled and unemployed graduates emerging out every year from these courses needs to be avoided. Our work correlates admissions and placements and finds employment potential of each branch for a location. This employment potential is used to provide a priority list including each branch for a specific demographic location.

Acknowledgements We extend our sincere gratitude to AICTE for considering the project problem statement and our approach worthy for Smart India Hackathon 2018, conducted by Government of India. We are thankful to officials of AICTE and Ministry of Gujarat for providing their valuable inputs to make the project prominent.

References

1. <https://www.weforum.org/agenda/2017/04/higher-education-in-china-has-boomed-in-the-last-decade>. Accessed 04 May 2018
2. <https://www.indiatoday.in/education-today/featurephilia/story/engineering-employment-problems-329022-2016-07-13>. Accessed 04 May 2018

3. Ahuja, S., Angra, S.: Machine learning and its applications: a review. In: 2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC) (2017). <https://doi.org/10.1109/ICBDACI.2017.8070809>
4. <https://www.mathworks.com/discovery/machine-learning.html>. Accessed 04 May 2018
5. Alexandra, L., Katarina, G., Hany, F.E., Miriam, A.M.C.: Machine Learning with Big Data: Challenges and Approaches, pp. 7776–7797. IEEE Access (2017)
6. Sharma, S., Agrawal, J., Agarwal, S., Sharma, S.: Machine learning techniques for data mining: a survey. In: IEEE Conference on Computational Intelligence and Computing Research (2013). <https://doi.org/10.1109/ICCIC.2013.6724149>
7. Uysal, I., Güvenir, H.A.: An overview of regression techniques for knowledge discovery. *Knowl. Eng. Rev.* **14**(4), 319–340 (1999)
8. Doan, T., Kalita, J.: Selecting machine learning algorithms using regression models. In: IEEE International Conference on Data Mining Workshop (ICDMW) (2015). <https://doi.org/10.1109/icdmw.2015.43>
9. Wang, Z., Shi, Y.: Prediction of the admission lines of college entrance examination based on machine learning. In: 2nd IEEE International Conference on Computer and Communications (ICCC) (2016). <https://doi.org/10.1109/compcomm.2016.7924718>
10. Giri, A., Vignesh, M., Bhagavath, V., Pruthvi, B., Dubey, N.: A placement prediction system using K-nearest neighbors classifier. In: Second International Conference on Cognitive Computing and Information Processing (CCIP) (2016). <https://doi.org/10.1109/ccip.2016.7802883>
11. Guo, L., Deng, X.: Application of improved multiple linear regression method in oilfield output forecasting. In: International Conference on Information Management, Innovation Management and Industrial Engineering (2009). <https://doi.org/10.1109/iciii.2009.39>

Content-Based Image Retrieval Using Statistical Color Occurrence Feature on Multiresolution Dataset



Debanjan Pathak , U. S. N. Raju , Sukhdev Singh, G. Naveen, and K. Anil

Abstract In modern life, the increasing use of different image-taking devices made image acquisition no longer a difficult task. To access a huge quantity of images having different resolutions stored in the dataset, the images must be kept in an organized manner. Content-Based Image Retrieval (CBIR) is an application of image retrieval problem, that is searching for a digital image from image dataset. Then term “content” in the context refers to some features that can be derived from the image itself. Color features are one of the important content of image which plays a vital role in image retrieval. Existing color features concentrate the only occurrence of pixel values or the correlation between pixel values. This paper proposed a new color feature which combines information about color shade percentage, color pixel occurrence percentage, pixel having maximum and minimum occurrence altogether. At the same time, proposed feature vector has a significantly reduced length which reduce computational cost of the retrieval system. This new feature is applied to one computer-generated image dataset (NITW-7500) and it's translated and multiresolution version (using bilinear interpolation) and one standard natural image dataset (Corel-1K). Performance is improved in all variations of computer-generated images.

Keywords CBIR · Multiresolution · Statistical occurrence · Image interpolation

D. Pathak · U. S. N. Raju · S. Singh

National Institute of Technology Warangal, Warangal, Telangana 506004, India

e-mail: debanjan21pathak@gmail.com

U. S. N. Raju

e-mail: usnraju@nitw.ac.in

S. Singh

e-mail: singhsukhdev1415@gmail.com

G. Naveen · K. Anil

Rajiv Gandhi University of Knowledge Technologies, Basar, Telangana 504107, India

e-mail: naveengenupuri108@gmail.com

K. Anil

e-mail: anilchathurvedi@gmail.com

1 Introduction

1.1 CBIR

CBIR is one of the applications of computer vision, which can be described as finding images in a large dataset. Jose Ramos et al. used CBIR as an application to Interstitial Lung Diseases [1], where CBIR is used as searching for retrieving earlier reports which are related to the patient. A new CBIR method is proposed by Jiann-Jone Chen et al., where it is applicable for the segmentation of images by using Morphological Reconstructions in different scales [2].

1.2 Multiresolution

Different resolution of an image can be achieved using different interpolation techniques such as Nearest Neighbour interpolation method [3], bilinear interpolation method [4], Bi-Cubic interpolation method [5], Bi-Quantic interpolation method [6].

1.3 Features

Features of an image can be of four types: point, texture, shape, and color features. SIFT [7], SURF [8] can be used as point features. Texture features can be of two types: Global Texture Features and Local texture features. Some well known Global Textures are First-Order Statistics [9], GLCM Statistical Parameters [10]. Some well-known Global Textures are Local Binary Pattern [11], Uniform Local Binary Pattern [11], Diagonally Symmetric Pattern [12]. Some of the existing shape features are histogram of oriented gradient (HoG) [13], Fourier Descriptor [14], Wavelet Fourier Descriptor [15], Convex Hull [16]. Color histogram, color autocorrelation are some well-known color features.

1.4 Color Histogram

A color histogram of an image can be defined as the distribution of the color composition of an image. A color histogram of an image can be represented by a vector, $h = \{h[0], h[1], h[2], \dots, h[i], \dots, h[n]\}$ where i is the color bin of a histogram and $h[i]$ represents the counting of pixel numbers in i th color bins in that image, and n is the total bins used in color histogram. Swain et al. [17] first proposed a method to use color histogram in CBIR, which gives a color histogram of length $(256 * 256 * 256)$,

known as pixel-level color histogram. For reducing the number of bins, color quantization can be applied. For further reduction of histogram length, Murala et al. [12] proposed one color histogram approach in which histogram is calculated for R, G, and B channels separately. The final color histogram is calculated by concatenating these three histograms. This variation of the histogram is known as a Global color histogram.

2 Main Contribution

The main contribution of this paper is proposing a new color feature which is calculated based on the following steps:

Step 1: Calculate ShadeBins, which is the Bin percentage by obtaining the number of color bins contributing at least one color including background (i.e., black), results in a vector of length 3. This component gives information about existing color bins in each channel. This component can be calculated using Eqs. (1)–(3)

$$\text{ShadeBins}(1, i) = 100 \times \frac{\text{Shades}(1, i)}{\sum_{k=1}^3 \text{Shades}(1, k)} \quad \forall i : 1 \leq i \leq 3 \quad (1)$$

$$\text{Shades}(1, i) = \sum_{j=1}^{256} f_1(\text{Hist}(i, j)) \quad \forall i : 1 \leq i \leq 3 \quad (2)$$

$$f_1(x) = \begin{cases} 0, & x = 0 \\ 1, & \text{else} \end{cases} \quad (3)$$

Hist is 3×1 matrix in which 1st, 2nd, and 3rd rows are histograms of red, green, and blue channels, respectively.

Step 2: Calculate the CountsPercentage by totaling the occurrences of all the non-background pixels in each channel, results in a vector of length 3. This component gives information about the most occurring color channel in the image. This component can be calculated using Eqs. (4)–(5)

$$\text{CountsPercentage}(1, i) = 100 \times \frac{\text{Counts}(1, i)}{\sum_{i=1}^3 \text{Counts}(1, k)} \quad \forall i : 1 \leq i \leq 3 \quad (4)$$

$$\text{Counts}(1, i) = \sum_{j=2}^{256} \text{Hist}(i, j) \quad \forall i : 1 \leq i \leq 3 \quad (5)$$

Step 3: Obtain the pixels with maximum frequency from each channel, results in a vector of 3, named Maxs. This component gives the most occurring pixel value in

each color channel. This component can be calculated using Eq. (6).

$$\begin{aligned} \text{Maxs}(1, i) = x, \exists \text{Hist}(i, x) \geq \text{Hist}(i, j), \forall x, \forall j, 1 \leq x, j \leq 256 \\ \text{and } x \neq j, \forall i : 1 \leq i \leq 3 \end{aligned} \quad (6)$$

Step 4: Obtain the pixels with minimum frequency from each channel, results in a vector of 3, named as Mins. This component gives the least occurring pixel value in each color channel. This component can be calculated using Eq. (7).

$$\begin{aligned} \text{Mins}(1, i) = x, \exists \text{Hist}(i, x) \leq \text{Hist}(i, j), \forall x, \forall j, 1 \leq x, j \leq 256 \\ \text{and } x \neq j, \forall i : 1 \leq i \leq 3 \end{aligned} \quad (7)$$

Step 5: Concatenate all the components calculated from Step 1 to Step 4, which is the final feature vector of length 12.

3 Methodology

Entire retrieval process of the proposed method is shown in Fig. 1 and the corresponding algorithm is given below:

Algorithm

1. Take an RGB image as input image.
2. Calculate BinPercentage, CountPercentage, Maxs, and Mins based on Eqs. (1)–(7).

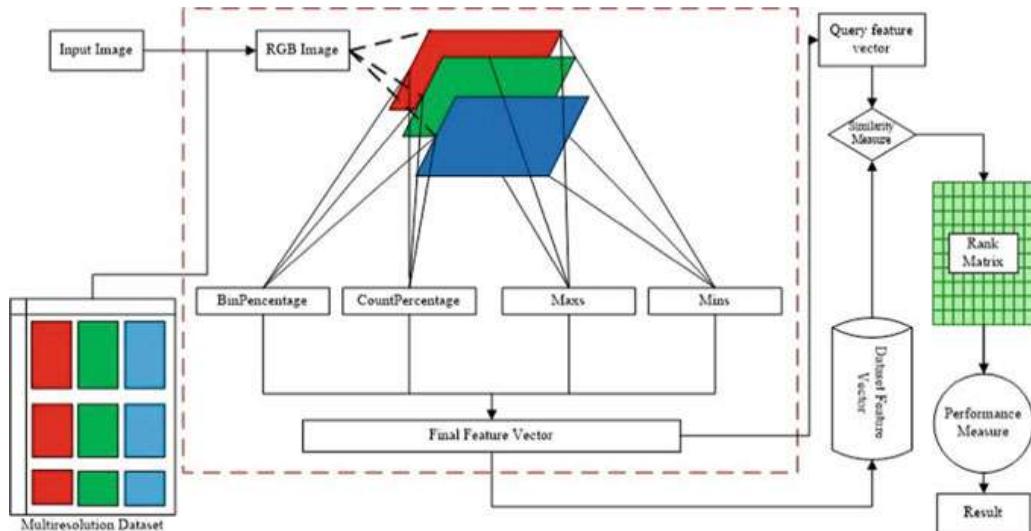


Fig. 1 Block diagram of the proposed system

3. Final feature vector is the concatenation of all four components.
4. Feature vector is calculated for both input image and all images of the dataset.
5. Rank matrix is constructed based on the sorted similarity measure applied between feature vectors obtained from input image and all images in the dataset.
6. Performance measures are calculated based on the Rank matrix computed in step 6.

4 Experimental Results and Discussions

For evaluating the proposed method performance, four computer-generated datasets of color images have been used among them two are single resolution and another two are multiresolution image dataset. The results of the proposed method are compared with two of the existing methods shown in Table 1. The results of the proposed method are compared with the existing methods, with the basic of six performance measures: Precision, Recall, ANMRR, TMRE, and F-measure [18].

4.1 NITW-7500 Dataset

The NITW-7500 which is the first dataset used to measure the performance of the proposed method, consisting of six classes having 1250 images of dimension 256×256 each result into a total of 7500 images in this dataset. This dataset is classified on six classes based on six different color Red (1–1250), Green (1251–2500), Blue (2501–3750), Yellow (3751–5000), Magenta (5001–6250), and Cyan (6251–7500). Each class of the dataset contains any total of 250 different shades of the same color having a range of 1 to 250. In each shade, we have five different shapes (triangle, horizontal rectangle, vertical rectangle, square, and diamond). In experiments, different number of images for each query image (125, 250, ..., 1250) are retrieved separately and its performance is evaluated. APR, ARR, and F-measure are shown from Figs. 2, 3, and 4. The APR, ARR, F-measure, ANMRR, and TMRE values for the NITW-7500 dataset are shown in Table 2.

Table 1 Different methods used for evaluating the performance

S. No.	1	2	3
Method	Global color histogram	Pixel-level color histogram	Proposed method
Feature vector length	$(3 * 256) = 768$	$(8 * 8 * 8) = 512$	$(3 + 3 + 3 + 3) = 12$

Fig. 2 Precision graph for NITW-7500

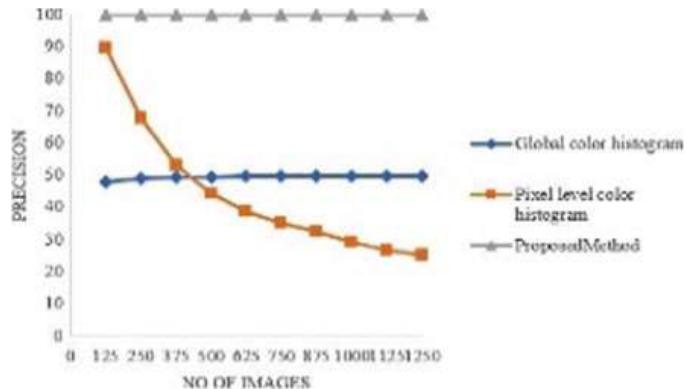


Fig. 3 Recall graph for NITW-7500

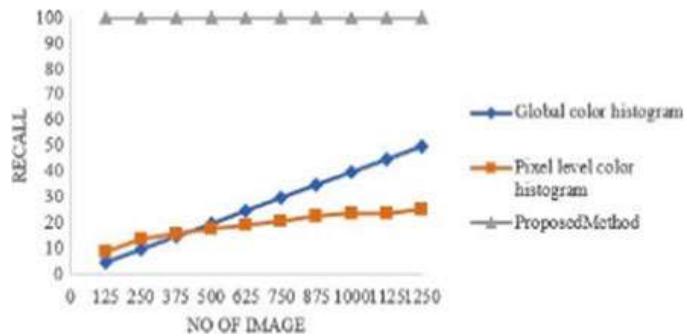
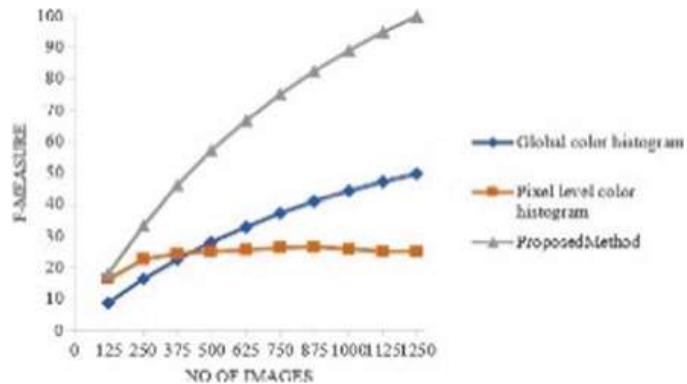


Fig. 4 F-Measure graph for NITW-7500



4.2 NITW-7500_MR Dataset

The multiresolution version of NITW-7500(NITW-7500_MR) is created to explore the result of the proposed method applied to the images having different dimensions. To create the multiresolution version of NITW-7500 dataset each class is divided into five different resolutions using bilinear interpolation. In each class, first 250 images are of size 256×256 , next 250 images are of size 128×128 , next 250 images are of size 64×64 , next 250 images are of size 32×32 and last 250 images are of size 16×16 . Some simple images are shown in Fig. 5. APR, ARR, and F-measure graphs are shown in Figs. 6, 7, and 8. The performance measures are given in Table 2.

Table 2 Performance measures for NITW-7500 and NITW-7500_Shifted_MR database

S. No.	Methods name	NITW_7500				NITW_7500_MR					
		APR	ARR	F-measure	ANMRR	MRE	APR	ARR	F-measure	ANMRR	MRE
1	Global color histogram	48.00	49.80	32.85	0.49	3184.80	44.70	18.80	13.52	0.73	6995.55
2	Pixel-level color histogram	89.66	25.15	24.38	0.64	7488.42	85.37	38.26	34.37	0.45	6239.44
3	Proposed method	100.00	100.00	66.24	0.00	1250.00	100.00	100.00	66.24	0.00	1250.00

Fig. 5 NITW-7500_MR samples (five different resolution per category)

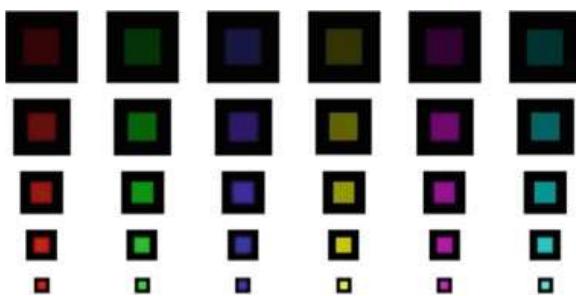


Fig. 6 Precision graph for NITW-7500_MR

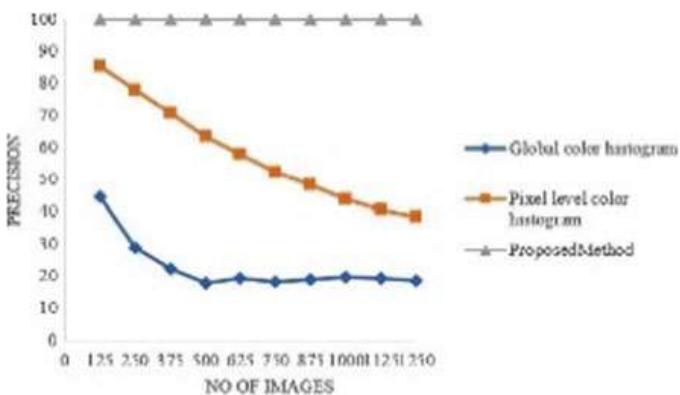


Fig. 7 Recall graph for NITW-7500_MR

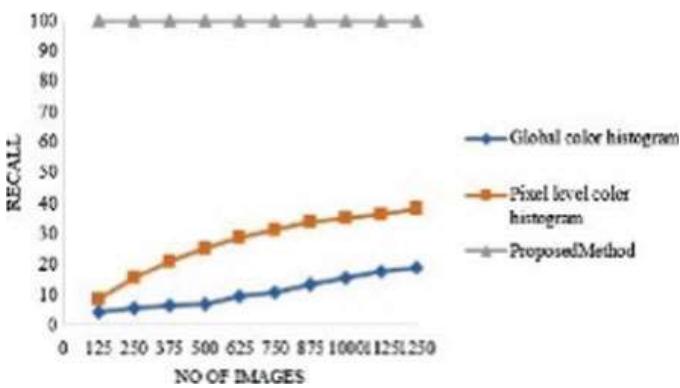
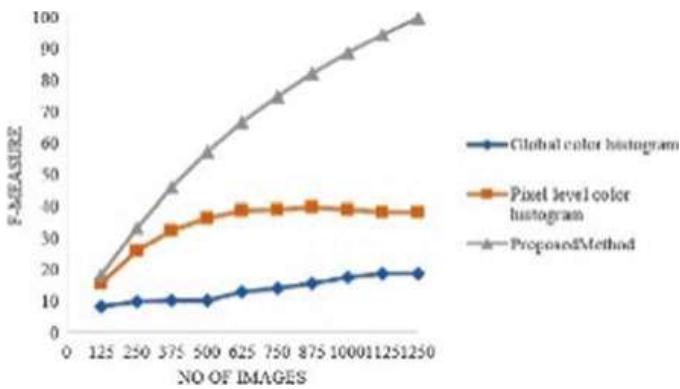


Fig. 8 F-measure graph for NITW-7500_MR



4.3 NITW-7500_Translated Dataset

To show the location independent property of the proposed method we create NITW-7500_Translated dataset where the number of images and dimensions of all the images are the same as NITW_7500 dataset but location of each shape is different with respect to NITW-7500 image dataset. APR, ARR, and F-measure graphs are shown in Figs. 9, 10, and 11. The performance measures are given in Table 3.

Fig. 9 Precision graph for NITW-7500_Translated

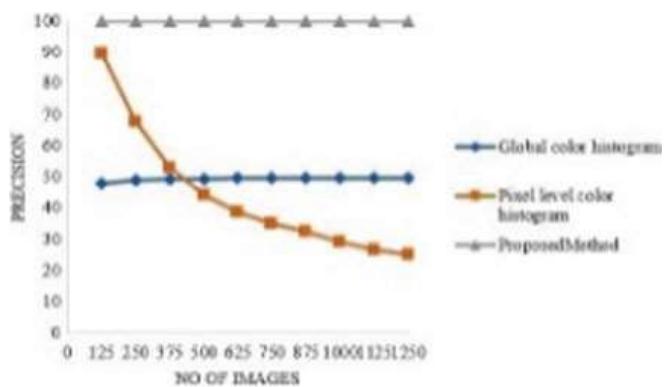


Fig. 10 Recall graph for NITW-7500_Translated

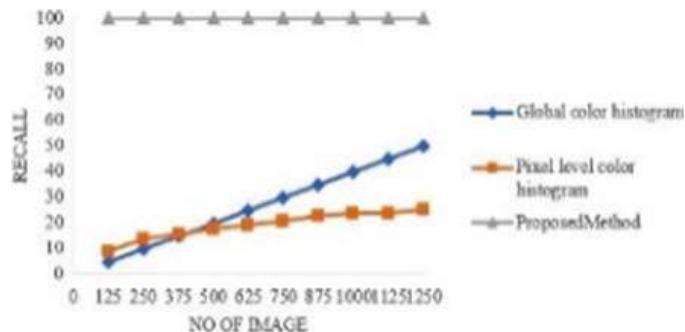


Fig. 11 F-Measure graph for NITW-7500_Translated

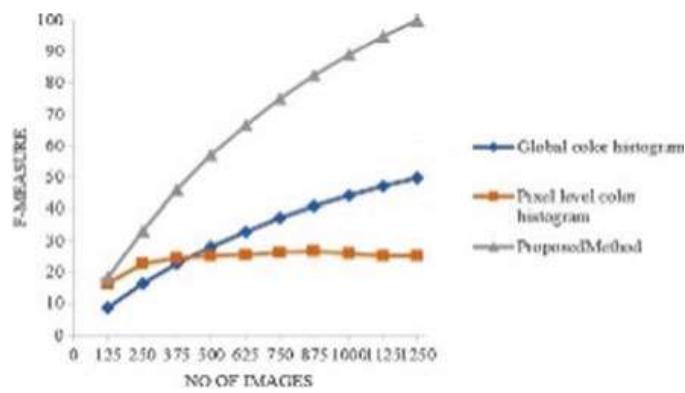


Table 3 Performance measures for NITW-7500_Translated and NITW-7500_Translated_MR database

S. No.	Methods name	NITW_7500_Translated				NITW_7500_Translated_MR					
		APR	ARR	F-measure	ANMRR	MRE	APR	ARR	F-measure	ANMRR	MRE
1	Global color histogram	48.00	49.80	32.85	0.49	3184.80	37.79	18.31	12.96	0.74	7012.82
2	Pixel-level color histogram	89.66	25.15	24.38	0.64	7488.42	85.69	38.26	34.37	0.45	6239.44
3	Proposed method	100.00	100.00	66.24	0.00	1250.00	100.00	100.00	66.24	0.00	1250.00

4.4 NITW-7500_Translated_MR Dataset

To show both location and dimension independent property of proposed method NITW-7500_Translated_MR dataset is created. To create this dataset same multiresolution approach is applied on NITW-7500_Translated_MR dataset. APR, ARR, and F-measure graphs are shown in Figs. 12, 13, and 14. The performance measures are given in Table 3.

Fig. 12 Precision graph for NITW-7500_Translated_MR

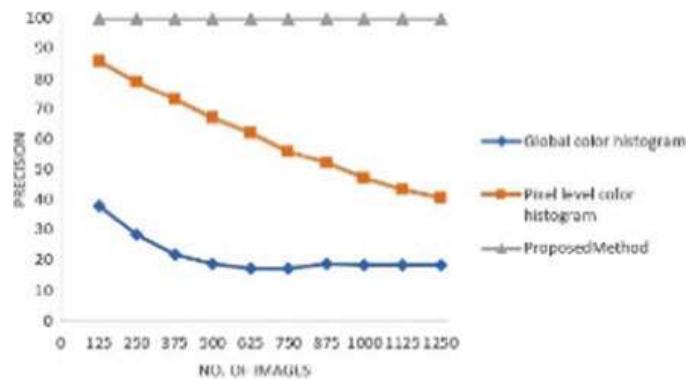


Fig. 13 Recall graph for NITW-7500_Translated_MR

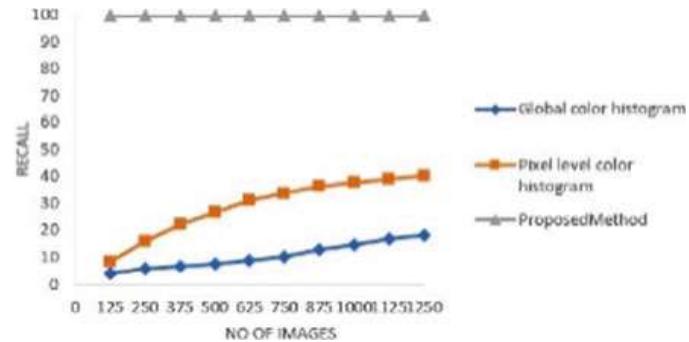


Fig. 14 F-Measure graph for NITW-7500_Translated_MR

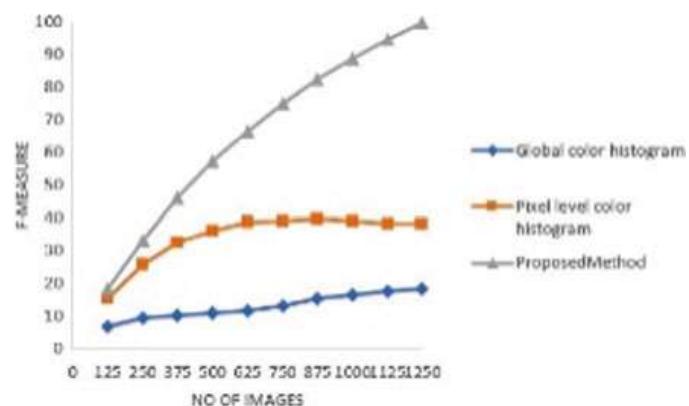


Table 4 Performance measures for NITW-7500_Shifted_MR database

S. No.	Methods name	Corel-1 K				
		APR	ARR	F-measure	ANMRR	MRE
1	Global color histogram	69.56	40.92	31.87	0.49	791.88
2	Pixel-level color histogram	71.62	37.65	30.72	0.52	825.78
3	Proposed method	33.19	20.29	15.12	0.72	968.05

4.5 Corel-1K Dataset

To measure the efficiency of our proposed method one standard natural image dataset Corel-1K is used, which contains a total of 1000 images distributed among 10 classes each having 100 images. The performance measures are given in Table 4.

5 Conclusion

In this paper, we proposed a novel color feature that combines Bin Shade information, count pixel information, minimum, and maximum pixel occurrence information. Proposed method is applied on two single resolution and two multiresolution datasets. Performance measures: APR, ARR, F-measure, and ANMRR are compared with existing methods. It shows better performance in both multiresolution and single resolution datasets. At the same time, it reduces the length of the feature vector by 98.43% and 97.65% with respect to Global and Pixel-level color histogram, respectively, which gives a significant improvement in terms of computational cost.

Declaration We have taken permission from competent authorities to use the images/data as given in the paper. In case of any dispute in the future, we shall be wholly responsible.

References

1. Ramos, J., Kockelkorn, T.T., Ramos, I., Ramos, R., Grutters, J., Viergever, M.A., van Ginneken, B., Campilho, A.: Content-based image retrieval by metric learning from radiology reports: application to interstitial lung diseases. *IEEE J. Biomed. Health Inform.* **20**(1), 281–292 (2014)
2. Chen, J.J., Su, C.R., Grimson, W.E.L., Liu, J.L., Shiue, D.H.: Object segmentation of database images by dual multiscale morphological reconstructions and retrieval applications. *IEEE Trans. Image Process.* **21**(2), 828–843 (2011)
3. Rukundo, O., Maharaj, B.T.: Optimization of image interpolation based on nearest neighbour algorithm. In: 2014 International Conference on Computer Vision Theory and Applications (VISAPP), vol. 1, pp. 641–647 (2014)
4. Gribbon, K.T., Bailey, D.G.: A novel approach to real-time bilinear interpolation. In: Proceedings. DELTA 2004. Second IEEE International Workshop on Electronic Design, Test and Applications, pp. 126–131 (2004)

5. Gao, S., Gruev, V.: Bilinear and bicubic interpolation methods for division of focal plane polarimeters. *Opt. Express* **19**(27), 26161–26173 (2011)
6. Chen, F., Wong, P.J.: On periodic discrete spline interpolation: quintic and biquintic cases. *J. Comput. Appl. Math.* **255**, 282–296 (2014)
7. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **60**(2), 91–110 (2004)
8. Karami, E., Prasad, S., Shehata, M.: Image matching using SIFT, SURF, BRIEF and ORB: performance comparison for distorted images. *arXiv preprint arXiv:1710.02726* (2017)
9. Julesz, B.: A theory of preattentive texture discrimination based on first-order statistics of textons. *Biol. Cybern.* **41**(2), 131–138 (1981)
10. Haralick, R.M., Shanmugam, K., Dinstein, I.H.: Textural features for image classification. *IEEE Trans. Syst. Man Cybern.* (6), 610–621 (1973)
11. Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* (7), 971–987 (2002)
12. Heikkilä, M., Pietikäinen, M., Schmid, C.: Description of interest regions with center-symmetric local binary patterns. In: *Computer Vision, Graphics and Image Processing*, pp. 58–69. Springer, Berlin (2006)
13. Bhunia, A.K., Bhattacharyya, A., Banerjee, P., Roy, P.P., Murala, S.: A novel feature descriptor for image retrieval by combining modified color histogram and diagonally symmetric co-occurrence texture pattern. *Pattern Anal. Appl.* 1–21 (2019)
14. Hu, R., Barnard, M., Collomosse, J.: Gradient field descriptor for sketch based retrieval and localization. In: *2010 IEEE International Conference on Image Processing*, pp. 1025–1028. IEEE (2010)
15. Osowski, S.: Fourier and wavelet descriptors for shape recognition using neural networks—a comparative study. *Pattern Recogn.* **35**(9), 1949–1957 (2002)
16. Mathew, S.P., Balas, V.E., Zachariah, K.P.: A content-based image retrieval system based on convex hull geometry. *Acta Polytech. Hung.* **12**(1), 103–116 (2015)
17. Swain, M.J., Ballard, D.H.: Color indexing. *Int. J. Comput. Vis.* **7**(1), 11–32 (1991)
18. Kanaparthi, S.K., Raju, U.S.N., Shanmukhi, P., Aneesha, G.K., Rahman, M.E.U.: Image retrieval by integrating global correlation of color and intensity histograms with local texture features. *Multimedia Tools Appl.* 1–37 (2019)

EEDCHS-PSO: Energy-Efficient Dynamic Cluster Head Selection with Differential Evolution and Particle Swarm Optimization for Wireless Sensor Networks (WSNS)



T. Guhan, N. Revathy, K. Anuradha, and B. Sathyabama

Abstract The communication subsystem in WSNs is primarily responsible for energy consumption which becomes a consistent in the networks owing to the usage of non-rechargeable battery having a limited power supply. The technology of communicating through wireless mode is recommended in number of sensing applications as it is convenient to use affordable and reliable. Modern sensors are designed with compatibility to sense the factors of the environment and transfer them in wireless mode. The center which collects the information favor confined data that are being clustered from a set of sensors than gathering the data from individual sensors. In general, wireless sensor network (WSN) uses grouping algorithm for domestic and use in abroad for dynamic cluster head selection is considered to be a significant task. To resolve the issue of cluster head selection with greater coverage and balanced energy consumption during the formation of cluster, it is taken as an important aspect. In this formulated work, an efficient clustering algorithm is proposed for monitoring the environment called energy-efficient dynamic cluster head selection with particle swarm optimization (EEDCHS-PSO). The selection process of cluster heads (CHs) is carried depending on the calculation of ordinary transmission distance and lingering energy. It can be seen that the sensor nodes known as cluster head (CH) that performs the task to route the data from the cluster to the cluster head of other clusters or base stations. Proposed EEDCHS-DEBO shows better performance in energy competence, load balancing, and range of scale with low control overhead.

T. Guhan (✉)
Sri Ramakrishna Engineering College, Coimbatore, India
e-mail: justforutsk@gmail.com

N. Revathy · B. Sathyabama
Hindusthan College of Arts and Science, Coimbatore, India
e-mail: drnrevathy@gmail.com

B. Sathyabama
e-mail: msathyaimpossible@gmail.com

K. Anuradha
Karpagam College of Engineering, Coimbatore, India
e-mail: k_anur@yahoo.com

Keywords Energy-Efficient Grouping · Wireless Sensor Networks (WSNs) · Cluster Heads (CHs) · Particle Swarm Optimization (PSO) algorithm

1 Introduction

The sensing systems that work wireless is important in monitoring various activities that take place in distant places like dense forests [1], locations close to volcanoes [2], deep sea [3], etc. A large number of sensors is available in the market for the purpose of quantifying the atmospheric temperature, the moisture level in the soil, vibrations caused because of earthquakes, the enemy intrusion in the borders, etc. In recent days, the development in the field of sensor technology permits the measured data transmission by means of wireless communication to the region of interest [4]. The values are generated as electrical signals and are transmitted in remote areas in the air consuming wireless technology [5]. The networking process is done using the devices having many wireless sensor devices namely sensor nodes (or simply nodes) for a particular function called as wireless sensor network (WSN) [6].

The major objective in the WSN is the power resources limitation in sensor nodes. It is problematic to recharge the batteries of nodes. However, the main challenge in the WSNs is the restricted power resources of sensor nodes. It is not practical to recharge the nodes batteries or exchange them after full depletion of energy as in several cases the nodes are implemented in combative atmosphere. Hence, olden method of networks attained a great Quality of Service (QoS), the sensor network protocols should concentrate primarily on energy conservation to increase the life span of network [7]. Several problems in research have been discussed in this topic. The most challenging issue is the design of energy-efficient clustering and routing algorithms [8].

The clustering process in WSN contains the process of combination the sensor nodes into remote clusters effectively; every cluster contains one head namely Cluster Head (CH). The CHs gather the information from all the related members, group them and transfer them to the BS. Every node relates to one and only cluster and transmits only with CH. Hence, the selection process of choosing the CHs requires proper addressing so as to balance the consumption of energy of CHs; or they might die soon because of extra load for the process of data aggregation and data forwarding. Many algorithms that are cluster-based choose the CHs initially on random basis or setting the probability in order to form them as clusters [8]. In some cases, CHs can be fixed in a minimal area of network and some normal nodes will be separated, which might lead to dysfunction of network.

In this proposed work, an efficient algorithm called energy-efficient dynamic cluster head selection with particle swarm optimization (EEDCHS-PSO) algorithm is formulated for the purpose of monitoring the fields. Cluster heads (CHs) are chosen depending on transmission distance in an average and lingering energy. A parameter is proposed to rotate the role of cluster head between the nodes. The evaluation

of performance is performed using the new factor namely complete useful data percentage (CUDP), first node die (FND), simulation time, scalability, and load balancing.

2 Literature Review

Ye et al. [9] formulated an innovative clustering mechanism EECS meant for wireless sensor networks, which suits well for the purpose of gathering the data periodically. EECS method chooses the cluster heads with large amount of residual energy via local radio communication with good distribution of cluster heads. The results of simulation prove that EECS shows significance in LEACH, extending the lifetime of network over 35%. Yong and Pei [10] formulated a work on cluster structuring algorithm based on distance-energy, examining both the distance and residual energy of nodes are taken into account shown in dissertation, which develops the clustering procedure regarding the selection of cluster head and data transmission process. The recreation outputs show that the enhanced algorithm efficiently balancing the consumption of energy.

Dahnil et al. [11] formulated the features of the scheme and the cluster heads potentiality to modify their level of power to attain the optimal degree and sustain this value throughout the operations carried out in a network. The results of simulation prove that the formulated clustering algorithm sustains the degree, which is required for the conduct of inter-cluster activity of several rounds when co-related with the techniques like hybrid energy-efficient distributed clustering (HEED), energy-efficient clustering scheme (EECS), low-energy adaptive clustering hierarchy (LEACH), and energy-based LEACH.

Tarhani et al. [12] formulated an innovative distributed algorithm called scalable energy-efficient clustering hierarchy (SEECH), which chooses CHs and transmits individually depending on eligibility on nodes. When the results are compared between LEACH and TCAC protocols tends to provide a better output of SEECH on the lifetime of its network. In the evaluation process of SEECH scalability, the simulation is carried out fewer than three various network size cases. Yu et al. [13] formulated a new protocol for clustering, namely local energy consumption prediction-based clustering protocol (LECP-CP), the significance includes the algorithm of cluster head selection and tree construction algorithm called inter-cluster transmission routing tree construction algorithm. The optimization of the global energy consumption can be carried out by optimizing the consumption of the local energy and the consumption of energy between the nodes can be balanced better.

Yu et al. [14] formulated energy-aware clustering algorithm (EADC) and a cluster-based routing algorithm. EADC utilizes a competitive range to build uneven sized clusters. The results of simulation show that the protocol was able to balance the consumption of energy among the nodes and lift the lifetime of network. Lin et al. [15] formulated a protocol for clustering purpose namely fan-shaped clustering (FSC) to segment a huge network into fan-shaped clusters. The analysis of performance

shows that the formulated FSC algorithm can effectively save the energy better when compared to hybrid, energy-efficient, and distributed clustering in both the rate of packet collection and energy saving.

Jia et al. [16] formulated a dynamic method for cluster head selection process for WSN so as to resolve the issue of unreasonable cluster head selection that may take overlapping coverage and consumption of unbalanced energy in the communication of cluster. The lifetime of the network is maximized by 50%, greater than that of LEACH, and maximized by 30%, which is also greater than distribute energy-efficient clustering (DEEC) algorithm. Padmanaban and Muthukumarasamy [17] formulated an easy but efficient algorithm for clustering called energy-efficient structured clustering algorithm (EESCA) to monitor the environment. The selection of cluster heads (CHs) depends on the distance of average communication and lingering energy. Moreover, an innovative parameter namely cluster head to normal ratio (CTNR) is proposed in cluster head rotation among the nodes.

Singh et al. [18], due to the usage of sensor with non-rechargeable battery having limited power supply, power consumption is the major issue in designing WSNs. Various methods and routing schemes are available to reduce the energy consumption of WSNs so that network can perform the specific task without network failure. It can be concluded from the obtained results that with the proposed approach, a significant amount of energy can be saved by reducing the overheads in CH election process. Further, time series prediction-based data reduction scheme has minimized the energy consumption by reducing the number of data transmissions in intra-cluster and inter-cluster region. GM (1, 1) model serves to provide a reasonably good accuracy by reduction in significant number of data transmissions. The proposed approaches: A1, A2 and A3 have significantly improved the network lifetime as depicted by the computed values of FND and HND; thereby demonstrating improvements over LEACH.

Senkerik et al. [19], this work helps to provide a more in-depth insight into the inner dynamics of indices selection in DE. The focus is to experimentally investigate the influence of different types of unconventional non-random (chaotic) sequences to the performance of different classes of DE strategies. The research of randomization issues and insights into the inner dynamic of metaheuristic algorithms was many times addressed as essential and beneficial.

3 Proposed Methodology

Our main objective is to create a clustering algorithm by means of multi-hop routing which generates an energy-efficient network for collecting the data in excess of a stretched phase of time. Concurrently as the sensor hubs are energy constrain, the shrinking of the overall energy utilization of the computer arrangement is vital. Also, the capacity of task must be open in the midst of the nodes consistently toward guaranteeing the likelihood of the premature loss of a node owing to the overload. Subsequently, the algorithm supposed to encompass good acquiescence to reach

good scalability when at most hubs or groups are fitted additional to the system arrangement and/or the exploitation zone is bloated. At last, a more rapid reaction is more essential in addition to condense the impediment in operation.

3.1 Clustering Parameters

Productive clustering is established by utilizing the accompanying components.

ACD: The ACD is the warning of the reasonableness of the node to progress toward becoming CH in terms of the node's location centrality. The ACD of a node is determined by (1)

$$\text{ACD}_i = \frac{\sum_{i=1}^n D_i}{n} \quad (1)$$

where D_i is the distance to the i th node and n is the number of nodes in the cluster. In mode 1, ACD is the foremost criteria for pointing the CH. The node which has the minimum considered ACD acquires the CH role.

Lingering energy: The energy-rich nodes ought to be responsible of the CH role. Once minimum energy nodes are designated as CH, the packet losses are enormous. In mode 1, the lingering energy is used as an additional parameter. In mode 2, the CH selection is done only by making an allowance for the remaining energy of the nodes. The node which has high lingering energy compared with further nodes in the cluster for a particular round obtains the CH role.

Hence, necessary energy to transmit a k bit data to a distance d is

$$E_{TX}(k, d) = \begin{cases} kE_{elec} + k\epsilon_{fs}d^2 & \text{for } d \leq d_0 \\ kE_{elec} + k\epsilon_{mp}d^4 & \text{for } d > d_0 \end{cases} \quad (2)$$

The energy to take delivery of a k bit data is

$$E_{RX} = kE_{elec} \quad (3)$$

where d_0 can be calculated using (4)

$$d_0 = \sqrt{\frac{\epsilon_{fs}}{\epsilon_{mp}}} \quad (4)$$

3.2 Network Initialization

- Step 1: The nodes recognize their somatic position on their own.
- Step 2: A HELLO note is transmitted by BS to each and every node in the network. The distance of the nodes is identified from them to BS based on the RSSI.
- Step 3: The nodes segregate on their own into four different clusters based on the physical position and the distance to BS. The zones are fashioned in the shape of squares.
- Step 4: Every node in the group collects the information of the distances flanked by it and other nodes by sending control messages to other nodes. This information is indispensable during the activity.
- Step 5: The ACD is calculated by each node.
- Step 6: The nodes swap the ACD information among them.

3.3 CH Selection Process

CH is chosen in hybrid modes related to the criticality and node's lingering energies.

Particle swarm optimization (PSO) algorithm is a population-based search algorithm based on the simulation of the communal activities of birds within a flock. The preliminary intent of the particle swarm concept was to graphically conjure up the graceful and unpredictable choreography of a bird flock [20, 21], with the aspire of discovering patterns that preside over the ability of birds to fly simultaneously, and to swiftly change direction with a regrouping in an optimal construction. From this initial objective, the concept evolved into a simple and efficient optimization algorithm.

Changes to the position of particles within the search space are based on the social-psychological propensity of individuals to imitate the success of other individuals [22, 23]. The changes to a particle within the swarm are therefore influenced by the experience, or knowledge, of its neighbors. The search activities of a particle are hence affected by other particles within the swarm (PSO) is therefore a kind of symbiotic cooperative algorithm. The consequence of modeling this social behavior is that the search process is such that particles stochastically come back toward earlier successful regions in the search space.

Individuals in a particle swarm track a very simple behavior: to imitate the accomplishment of neighboring individuals and their own victory. The communal behavior that comes out from this simple behavior is that of discovering optimal regions of a high dimensional search space. A PSO algorithm maintains a swarm of particles, where each particle represents a potential solution. In comparison with evolutionary computation paradigms, a swarm is similar to a population, while a particle is similar to an individual. In simple terms, the particles are “flown” through a multidimensional search space, where the location of every particle is accustomed according to its own knowledge and that of its neighbors. Let $x_i(t)$ denotes the position of particle i in the search space at time step t ; unless otherwise stated, t denotes discrete time

steps. The position of the particle is changed by adding a velocity, $v_i(t)$, to the current position, i.e.,

$$x_i(t + 1) = x_i(t) + v_i(t + 1) \quad (5)$$

with $x_i(0) \sim U(x_{\min}, x_{\max})$. It is the velocity vector that drives the optimization process and reflects both the experiential knowledge of the particle and socially exchanged information from the particle's neighborhood. The experiential knowledge of a particle is generally referred to as the cognitive component, which is proportional to the distance of the particle from its own best position (referred to as the particle's personal best position) found since the first time step. The socially exchanged information is referred to as the social component of the velocity equation. Originally, two PSO algorithms have been developed which differ in the size of their neighborhoods. For the overall best PSO, or gbest PSO, the neighborhood for every particle is the whole swarm. For gbest PSO, the velocity of particle i is calculated as

$$\begin{aligned} v_{ij}(t + 1) = & v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] \\ & + c_2 r_{2j}(t)[\hat{y}_{ij}(t) - x_{ij}(t)] \end{aligned} \quad (6)$$

where $v_{ij}(t)$ is the velocity of particle in dimension $j = 1, \dots, n$ at time step t , $x_{ij}(t)$ is the location of particle in dimension j at time step t , c_1 and c_2 are positive acceleration constants used to level and scale the contribution of the cognitive and social components, respectively, and $r_{1j}(t), r_{2j}(t) \sim U(0, 1)$ are random values in the range $[0, 1]$, sampled from a unvarying allocation. These haphazard values initiate a stochastic building block to the algorithm.

The individual best arrangement is, y_i , associated with particle i is the best position the particle has been seen since initial step. Taking into consideration minimization problems, the private best spot at the further time step, $t + 1$, is deliberated as

$$y_i(t + 1) = \begin{cases} y_i(t) & \text{if } f(x_i(t + 1)) \geq f(y_i(t)) \\ x_i(t + 1) & \text{if } f(x_i(t + 1)) \leq f(y_i(t)) \end{cases} \quad (7)$$

where $f: \mathbb{R}^{n_x} \rightarrow R$ is the fitness function. As with EAs, the fitness function measures how accurate the consequent result is to the optimum, i.e., the fitness function quantifies the performance, or quality, of a particle (or solution). The global most excellent position, $\hat{y}_{ij}(t)$, at time pace t , is defined as

$$\hat{y}(t) \in \{y_0(t), \dots, y_{n_s}(t)\} | f(\hat{y}(t)) = \min\{f(y_0(t)), \dots, f(y_{n_s}(t))\} \quad (8)$$

where n_s is the total number of particles in the swarm. It is imperative to note that the description in equation indicates that $\hat{y}(t)$ is the preeminent location revealed by any of the particles so far—it is usually calculated as the best personal best position.

3.3.1 Steady-State Operation

- Step 1: The ordinary hubs in whole groups identifies the information from the background and direct the identified information to the equivalent CHs on TDMA slot assigned to them.
- Step 2: Subsequently, getting all the data from the nodes, the CH starts performing the data combination.
- Step 3: The CH confines and authorize for the intermediary CHs to direct the information toward BS. Thus, the CHs placed remotely from the BS (lower-level CHs) could broadcast the records to the BS via intermediate CHs (higher-level CHs) using various hop communication. The higher-level CHs drive the data nonstop to the BS.

3.3.2 CH Role Rotation Process

It consists of two key steps which are deliberated below,

CTNR: The CTNR is calculated by the proportion of energy consumed by the node when substituting as CH in mode 1 to the entire energy of the node. This factor should be chosen optimally to obtain the utmost epoch and nominal simulation time.

CH role rotation: The nodes that are nominated in mode 1, CHs keep hold of the role of CHs till $CTNR \leq 0.4$. The motive for choosing the value of CTNR is 0.4. Once a node misses its work of CH, it transmits the LOSE-CH communication to the node that clutches the succeeding minimum ACD. After getting this communication, the latest node broadcasts its CH-MSG to further nodes and remains its operation till $CTNR \leq 0.4$. This procedure is sustained at the maximum of every node in the cluster gain the role of CH at least one time.

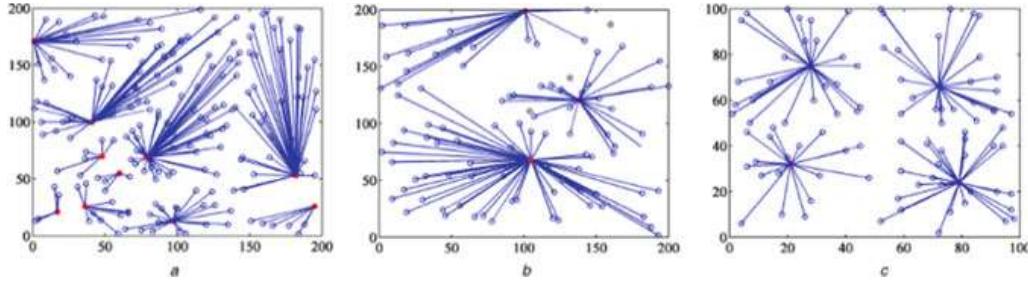
4 Simulation Results

The parameters of simulation are listed in Table 1. The simulations are carried out with the help of MATLAB installed on a standard PC (Intel® Core™ i5 processor and 8 GB random access memory). The formulated EEDCHS-PSO is assessed by the correlating it with existing methods LEACH, SEECH, and EESCA. To validate the check the scalability of the algorithms, three various cases are considered. Scene 1 has an area of $100 \text{ m} \times 100 \text{ m}$ and 100 nodes. Scene 2 has $200 \text{ m} \times 200 \text{ m}$ area and 100 nodes. Scene 3 has $200 \text{ m} \times 200 \text{ m}$ area with 200 nodes. In scene 2, the deployment area is doubled, and in scene 3 the area and the number of nodes is doubled.

As EEDCHS-PSO is a structured algorithm, the formation of cluster is better when correlated against LEACH and SEECH. This is depicted in Fig. 1. The clusters

Table 1 Simulation parameters

Parameter	Scene 1	Scene 2	Scene 3
N	100	100	200
Area of deployment	100×100	200×200	200×200
Location of BS	50, 100	100, 200	100, 200
$E_{\text{Initial}} (\text{J})$	0.5	0.5	2
Packet size, K		4000 bits	
E_{Tx}		50 Nj/bit	
d_0		87 m	
ε_{mp}		0.0013 pJ/bit/m ⁴	
ε_{fs}		10 Pj/bit/m ²	
E_{DA}		5 nJ/bit/message	

**Fig. 1** Load balancing in SEECH, EESCA, and EEDCHS-PSO. **a** LEACH for cluster formation in scene 3 in the fourth round, **b** SEECH for cluster formation in scene 2 in the first round, **c** EEDCHS-PSO for cluster formation in scene 1 in the 15th round

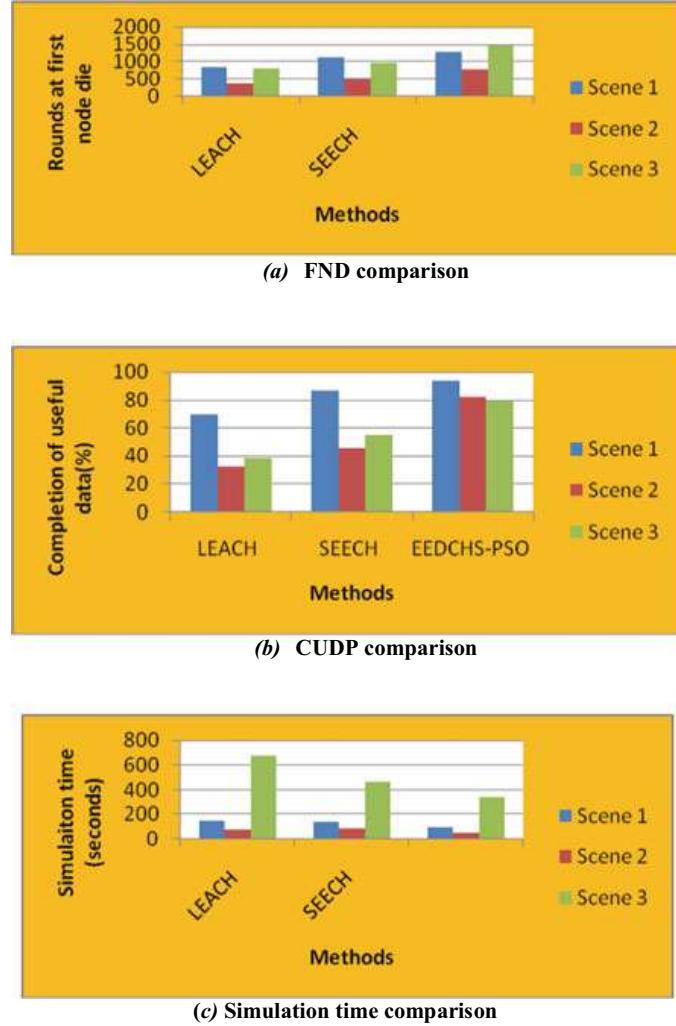
that are balanced form CHs which share even sized loads. Figure 1a, b depict the formation of cluster of LEACH and SEECH, and Fig. 1c depicts the formation of clusters in EEDCHS-PSO. The different instances are validated for the testing the formation of uniform cluster. In all the scenes, the distribution of the load is ununiform in LEACH and SEECH.

(ii) The CUDP can be measured using the equation as follows:

$$\text{CUDP} = (\text{FND}/\text{LND}) \times 100(8)$$

CUDP is also maximum in all the scenes when correlated against LEACH and SEECH algorithm. Figure 2b depicts the comparisons of CUDP. The CUDPs of the LEACH and SEECH are significantly minimized to 39.00 and 55.4% for scene 3 (see Fig. 2b). Figure 2a shows the FND is greater for the formulated EEDCHS-PSO when correlated with LEACH and SEECH in all the three scenes. The FNDs are 435, and 160 higher than LEACH and SEECH in scenes 1. Time comparison of the simulation

Fig. 2 FND, CUDP, and simulation time comparisons



is depicted in Fig. 2c shows EEDCHS-PSO carries out the operations quickly when correlated against LEACH and SEECH. In the case of EESCA, it has minimal time for simulation of 342.5 s, and the other techniques like LEACH and SEECH take of 678.2 and 468 s for scene 3(see Fig. 2c). It concludes that the formulated work has the simulation time is nearly half of the time consumed by further algorithms.

5 Conclusion and Future Work

A wireless sensor network (WSN) contains wireless sensor nodes and a sink node. Nodes are interconnected wirelessly to each another and to the sink. The production of energy-efficient WSNs is a must for the real-time applications which are located in remote area. In the recent research work, in every algorithm, the role of CH is rotated every round. Also, the data regarding other nodes is required in all

rounds. These processes form high control overhead which results in the wastage of high-energy. This motivated to conduct a research on energy-efficient dynamic cluster head selection with particle swarm optimization (EEDCHS-PSO) to improve the potential of WSN in the areas like scaling with overhead having the low control, energy efficiency, and load balancing. If the extra rounds are required, it chooses the cluster head automatically and minimizing the overhead controls and usage of energy. The automatic process chooses the CHs by PSO which leads in optimized clustering algorithm of adaptive energy. The formulated EEDCHS-PSO is very effective, avoiding the extreme restriction overhead, by the gyrating CH role between the nodes at correct interludes, and thereby minimizing the ingestion of energy remarkably. Future work of this is that, it will be applied to the existing system so as to implement on the applications that are real time. The cluster formation can be done as the future work, by means of certain adaptive clustering algorithms like fuzzy, density, and spatial clustering methods, to minimize the time of cluster formation.

References

1. Aslan, Y.E., Korpeoglu, I., Ulusoy, Ö.: A framework for use of wireless sensor networks in forest fire detection and monitoring. *Comput. Environ. Urban Syst.* **36**(6), 614–625 (2012)
2. Lara, R., Bentez, D., Caamaño, A., et al.: On real-time performance evaluation of volcano-monitoring systems with wireless sensor networks. *IEEE Sens. J.* **15**(6), 3514–3523 (2015)
3. Akyildiz, I.F., Pompili, D., Melodia, T.: Underwater acoustic sensor networks: research challenges. *Ad Hoc Netw.* **3**(3), 257–279 (2005)
4. Wang, F., Liu, J.: Networked wireless sensor data collection: issues, challenges, and approaches. *IEEE Commun. Surv. Tutor.* **13**(4), 673–687 (2011)
5. Dargie, W., Poellabauer, C.: *Fundamentals of Wireless Sensor Networks: Theory and Practice*. Wiley, Hoboken, USA (2010)
6. Zhao, F., Guibas, L.J.: *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann, Burlington, USA (2004)
7. Anastasi, G., Conti, M., di Francesco, M., Passarella, A.: Energy conservation in wireless sensor networks: a survey. *Ad Hoc Netw.* **7**(3), 537–568 (2009)
8. Abbasi, A.A., Younis, M.: A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.* **30**(14–15), 2826–2841 (2007)
9. Ye, M., Li, C., Chen, G., Wu, J.: EECS: an Energy Efficient Clustering Scheme in wireless sensor networks. In: Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference (IPCCC '05), pp. 535–540 (2005)
10. Yong, Z., Pei, Q.: An energy-efficient clustering routing algorithm based on distance and residual energy for wireless sensor networks. *Procedia Eng.* **29**, 1882–1888 (2012)
11. Dahnil, D.P., Singh, Y.P., Ho, C.K.: Topology-controlled adaptive clustering for uniformity and increased lifetime in wireless sensor networks. *IET Wirel. Sens. Syst.* **2**(4), 318–327 (2012)
12. Tarhani, M., Kavian, Y.S., Siavoshi, S.: SEECH: scalable energy efficient clustering hierarchy protocol in wireless sensor networks. *IEEE Sens. J.* **14**(11), 3944–3954 (2014)
13. Yu, J., Feng, L., Jia, L., et al.: A local energy consumption prediction-based clustering protocol for wireless sensor networks. *Sensors* **14**(12), 23017–23040 (2014)
14. Yu, J., Qi, Y., Wang, G., et al.: A cluster-based routing protocol for wireless sensor networks with nonuniform node distribution. *AEU-Int. J. Electron. Commun.* **66**(1), 54–61 (2012)
15. Lin, H., Wang, L., Kong, R.: Energy efficient clustering protocol for largescale sensor networks. *IEEE Sens. J.* **15**(12), 7150–7160 (2015)

16. Jia, D., Zhu, H., Zou, S., et al.: Dynamic cluster head selection method for wireless sensor network. *IEEE Sens. J.* **16**(8), 2746–2754 (2016)
17. Padmanaban, Y., Muthukumarasamy, M.: Energy-efficient clustering algorithm for structured wireless sensor networks. *IET Netw.* **7**(4), 265–272 (2018)
18. Singh, D.P., et al.: An efficient cluster-based routing protocol for WSNs using time series prediction-based data reduction scheme. *Int. J. Meas. Technol. Instrum. Eng. (IJMTIE)* **3**(3), 18–34 (2013)
19. Senkerik, R., et al.: Differential evolution and deterministic chaotic series: a detailed study. *Mendel* **24**(2) (2018)
20. Kennedy, J.: Particle swarm optimization. In: *Encyclopedia of Machine Learning*, pp. 760–766. Springer, Boston, MA (2011)
21. Poli, R., Kennedy, J., Blackwell, T.: Particle swarm optimization. *Swarm Intell.* **1**(1), 33–57 (2007)
22. Clerc, M.: *Particle Swarm Optimization*, vol. 93. Wiley (2010)
23. Du, K.L., Swamy, M.N.S.: Particle swarm optimization. In: *Search and Optimization by Metaheuristics*, pp. 153–173. Birkhäuser, Cham (2016)

An Unsupervised Searching Scheme over Encrypted Cloud Database



T. Janani and M. Brindha

Abstract In the progression of the imaging tools, like computerized cameras, cell phones, and medical imaging equipment the world has been seeing development in quantity, accessibility and significance of images. Significantly more storage space is required for the images over the text document. But in the storage outsourcing scenario, there is a potential hazard for clients that everyone may have the entrance to gain useful information from these image data. For the most part, the encryption mechanism is embraced to avoid such situations. Thus, it is a decent practice to encrypt the confidential information before outsourcing to the untrusted outsiders. Also, query processing for image retrieval is a huge task. Therefore, the proposed paper executes a methodology for secure image encryption using a chaotic system and machine learning based approach for searching similar images proficiently over encrypted image database without leaking any sensitive data.

Keywords Clustering · Feature extraction · Henon map · Chaotic encryption · Searching

1 Introduction

Distributed storage [1, 2] has become progressively pervasive in recent days. It gives a stage to a customer to store and furthermore share their information in an encrypted form. Image encryption techniques have been progressively used for preserving the confidentiality of the image. Xia [3], proposed a scheme to protect image data from an untrusted cloud server from ruining the information in Content-Based Image Retrieval (CBIR) outsourcing. Image encryption algorithms are generally divided

T. Janani · M. Brindha ()

Department of Computer Science and Engineering, National Institute of Technology
Tiruchirappalli, Tiruchirappalli, Tamil Nadu, India
e-mail: brindham@nitt.edu

T. Janani
e-mail: saijanani.308@gmail.com

into two groups concerning the methodology used to perform encryption: chaos-based method and non-chaos-based methods Zhang [4], proposed an image encryption mechanism with diffusion and confusion through a rotation matrix in its confusion phase. It uses a bit-level confusion, which modifies pixels as well as their values. Liu [5], stated that a bit-level permutation and high-dimension chaotic maps are utilized to encrypt colour images. It uses Chen's System for confusion and diffusion of the colour pixels. It uses the bit-level permutation to diffuse and confuse the colour image pixels simultaneously. Also, the author suggested an image encryption technique based on information entropy and a chaotic map. Liu [6] analyzed that chaotic cryptosystems have been deliberated widely because of its ergodicity, pseudo-randomness and sensitivity to initial conditions and control parameters, which are close to confusion and diffusion in cryptography. These properties make chaotic systems a potential choice for constructing cryptosystems. Ye [7], proposed a model for retrieving encrypted images in linear time based on the user's request. In order to get the encrypted images appropriately, the data owner uploads some additional information with the corresponding images. There are a variety of searching scheme for image database. According to Salahat et al. [8] survey, Feature detection, and description algorithms considered to be a retina for the computer vision system for identifying the exact images or objects. The author analyzed various feature extraction algorithms that are proposed so far. Particularly author report that the Maximally Stable Extremal Regions algorithm (MSER) and the Scale Invariant Feature Transform algorithms (SIFT) are the best of their type and selected for recent derivatives. Since SIFT has been proposed by Lowe [9] in the year 2004, a number of calculations are attempted to decrease the SIFT descriptor width. Choras [10] analyzed various image feature extraction techniques for CBIR and biometric systems. It provides a possible approach to utilize feature extraction on biometric systems. The method analyzed Hue, Saturation, Value (HSV) features in the field of biometric analysis like iris classification. Some researchers came with the idea of building an index like structure to make the searching efficient. Gudmundsson et al. [11] proposed a cluster-based pruning algorithm for large-scale image datasets. It includes index depth and cluster size also authors used extended cluster search for performance upgradation. Yang et al. [12] suggested two-stage clustering for image retrieval. Author utilized k-means clustering with the neighboring uniform histogram for hyperspectral remote sensing image classification. However, it could not determine the cluster number at the first stage.

The proposed paper utilizes Scale Invariant Feature Extraction (SIFT) method to extract the features like key points and descriptors from the image dataset and the respective index is constructed. To ensure the privacy of outsourced image, chaotic based image encryption is performed prior to the image outsourcing. Also, the proposed methodology uses cluster-based filter table for the fast retrieval of the query image. The proposed paper is organized as follows: Sect. 2 describes the proposed work, Sect. 3 discuss performance evaluation and comparison and Sect. 4 gives the conclusion.

2 Proposed System

To ensure the efficient privacy preserved query processing over encrypted image datasets, the proposed paper utilized chaotic based image encryption and searching is performed by clustering the encrypted datasets as extracting feature vectors from an image using SIFT. Figure 1 depicts the overall architecture of the suggested system. Due to the space constraints, data owners outsource their data to the cloud database. Data outsourcing will lead to the problem of data privacy and searching is more complex. To overcome this, the proposed framework uses chaotic based image encryption using henon map for encrypting the image before they are outsourced. Also, to make searching efficient and accurate, the framework utilizes SIFT-based feature extraction and k-means clustering for categorizing the image datasets.

The system works as follows:

1. Image owner I outsources their image data to the third-party server.
2. Cloud server C , performs SIFT for extracting key points and descriptors from the plain image and the image encryption is carried out based on henon chaotic map.
3. Cloud server C constructs an index structure by utilizing the extracted features.
4. For efficient searching, the server will also perform k-means clustering on index and computes clustered table which will reduce the searching time notably.
5. If query user Q post query image Q_i , cloud server will automatically compare the features of query and original image, and if matches, the corresponding image is sent to the user.
6. The user can decrypt the image after submitting their authentication details to the server.

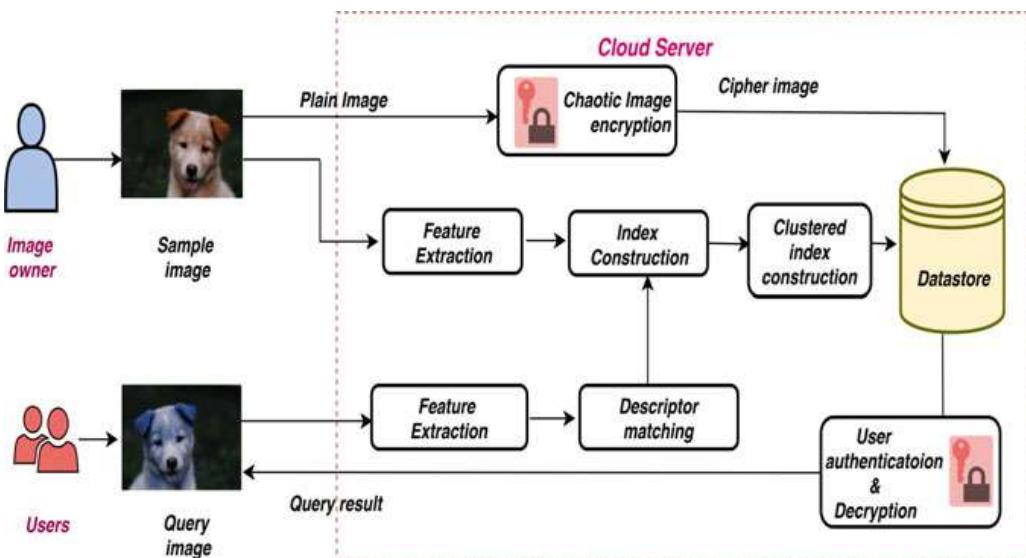
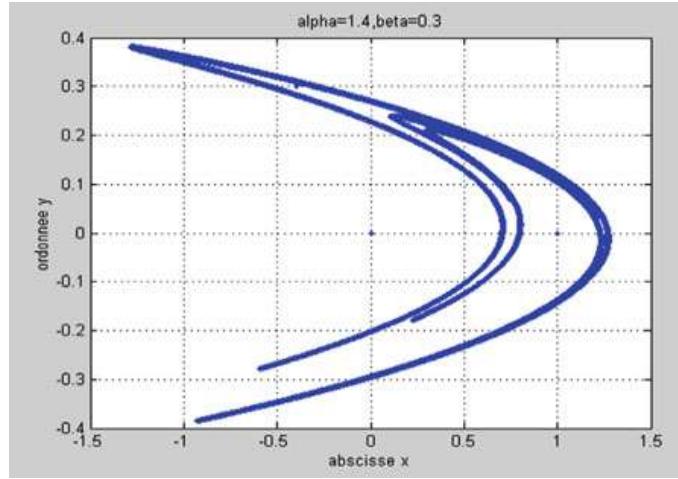


Fig. 1 Proposed architecture

Fig. 2 Henon map with $\alpha = 1.4$ and $\beta = 0.3$



2.1 Hénon-Based Image Encryption

A 2D invertible chaotic method called henon is utilized in the proposed framework. It was an extended version of the Poincare map for the Lorenz model. The numerical conditions of the henon map are mentioned as follows:

$$\begin{aligned} A_{(x+1)} &= 1 - \alpha A_x^2 + B_y \\ B_{y+1} &= \gamma x_n \end{aligned} \tag{1}$$

where α and β are the parameters which are analyzed for different numeric values and the chaotic nature is established for $\alpha = 1.4$ and $\beta = 0.3$. Figure 2 shows the structure of henon map.

To acquire the fine structure of the chaotic attractor, the system parameters should not be excessively huge or excessively small. The region of contraction will be extreme if β is too close to zero. Here α and β are chosen as a seed for the chaotic encryption.

2.2 Scale Invariant Feature Transform (SIFT)

The image dataset consists of numerous images, searching an image among these huge datasets is a time-consuming task. To perform efficient searching, the proposed system uses SIFT feature extraction method to extract the appropriate features from the image datasets and these features are stored in database. The steps involved in SIFT feature extraction are as follows.

- Step 1: Determine the approximate location and scale of feature points, i.e. Interesting points.
- Step 2: Refine their location and scale.
- Step 3: Establish orientations for each interesting point.
- Step 4: Identify corresponding descriptors for each interesting point.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (2)$$

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y,) \quad (3)$$

where G and L are the nearest intensity values.

2.3 K-Means Clustering-Based Search Scheme

The proposed system constructs the index table for the images according to their extracted feature vectors. To make the searching efficiency, the proposed model utilizes clustered based table to search the query image instead of searching the entire database. The clustered table is mapped with an index table, if matching found clustered table, the respective index is obtained to retrieve the images. The algorithm proposed for constructing clustered feature table is as follows:

Algorithm: K-means clustered feature vectors

Input: k -number of groups and $f v$ -feature vector table

Output: $C f v$ -clustered table.

- Step 1: choose k objects randomly from the feature table $f v$
- Step 2: repeat
 - 1.1 Compute similarity score between extracted features
 - 1.2 Recompute each object's centroids when it forms a new cluster
 - 1.3 Calculate mean vale of the similar features
 - 1.4 Update the cluster and calculate the mean value for the new feature vector
- Step 3: End

2.4 Image Decryption

When the user presented the query image to the cloud, cloud server will initially extract the feature descriptors from the query image and compares it with the stored features table. If the user wants first degree of result, i.e. the category of the query image, the cloud server will basically list the set of similar images by referring the

feature table. If any query user wants refined results, server will refer to the clustered table and list the relevant image. In both the scenario, decryption is performed after referring to the authentication of query user by submitting their credentials. This approach will maintain the secrecy of sensitive images over cloud database.

3 Results and Performance Evaluation

This paper proposes an efficient searching scheme using clustering method, also the searching time is improved by comparing the feature instead of whole datasets. The experiment was done on standard 101 object categories dataset which consists of 9146 images with 101 categories. Among 9146 images 7500 images are used for training and the remaining 1916 images are used for testing. This is roughly about the standard 80–20% for dataset splitting. Figures 3 and 4 show the sample image for the performance evaluation and the output of feature extraction stage, respectively. Once the features are extracted their pixel values are stored in a table. Here the paper employs 128 bits SIFT algorithm, hence 128×128 matrix for storing the interesting

Fig. 3 Sample image



Fig. 4 SIFT-feature vectors



points. Among the matrix values, index is constructed for every image. The output of the chaotic image is shown in Fig. 5.

To analyze the performance of cryptosystems, entropy calculation is made, and it is compared with the existing systems as shown in Table 1.

At intermediate stage, the method produces an output as the category of the query image with respect to datasets. The time complexity for retrieving a similar set of images from the encrypted database using cluster table is listed in Table 2.

Table 3 shows the overall result achieved on 101 object categories dataset over several feature values.

The proposed methods achieve better accuracy when choosing an appropriate number of features on similar image search also it exhibits minimal time for performing encryption and feature extraction.

Fig. 5 Henon transformed image-cipher image

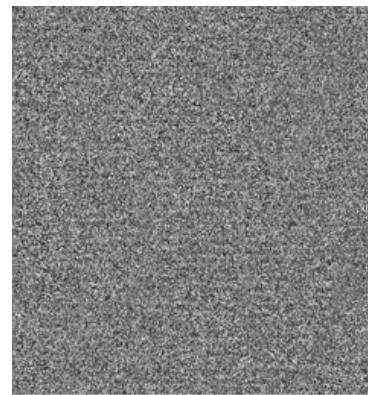


Table 1 Comparison of information entropy

S. No.	Scheme	Entropy
1	Proposed encryption	7.9877
2	Zhang et al. [4]	7.7924
3	Liu et al. [5]	7.7902
4	Xia [3]	7.8204

Table 2 Comparison of Time Complexity

S. No.	Method	Clustering time (s)	Query time (s)
1	Proposed searching scheme	20.2	5.6
2	Ye [7]	65	28.5
3	Choras [10]	54.6	32
4	Gudmundsson [11]	23.02	12.3

Table 3 Performance of proposed clustering-based searching scheme

Feature descriptors	TP (%)	TN (%)	FN (%)
15	40.03	29.22	30.75
30	64.01	17.23	18.76
60	78.07	8.75	13.18
80	88.38	3.39	8.23

4 Conclusion

To ensure the privacy of the outsourced image, the proposed paper uses chaotic based encryption and for efficient searching scheme. The paper utilizes clustered based filter table using k-means and SIFT. The performance evaluation shows that the encryption is fully randomized, and the searching is performed efficiently by identifying the interesting point sand descriptors. The existing system uses Paillier based image encryption which is infeasible for the huge image datasets. To enhance the searching efficiency for very large datasets with millions of images, this work can be extended by incorporating deep learning methods for classifications.

Acknowledgements This research work was funded by the Department of Science and Technology, India under the project “Design and Development of ICT-Enabled Cloud based mobile application for the self-promotion of products developed by Self Help Groups”.

References

1. Wu, J., Ping, L., Ge, X., Wang, Y., Fu, J.: Cloud storage as the infrastructure of cloud computing. In: International Conference on Intelligent Computing and Cognitive Informatics (ICICCI), Kuala Lumpur, Malaysia, 22–23 June 2010, pp. 380–383
2. Furukawa, J.: Request-based comparable encryption. In: Computer Security—ESORICS 2013—18th European Symposium on Research in Computer Security, Egham, UK, 9–13 Sept 2013. Proceedings, vol. 8134, pp. 129–146 (2013)
3. Xia, Z., Wang, X., Zhang, L., Qin, Z., Sun, X., Ren, K.: A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. *IEEE Trans. Inf. Forensics Secur.* **11**(11), 2594–2608 (2016)
4. Zhang, Y., Di, X.: An image encryption scheme based on rotation matrix bit-level permutation and block diffusion. *Commun. Nonlinear Sci. Numer. Simul.* **19**(1), 74–82 (2014)
5. Liu, H., Wang, X.: Color image encryption using spatial bit-level permutation and high-dimension chaotic system. *Opt. Commun.* **284**(16–17), 3895–3903 (2011)
6. Liu, H., Wang, X.: Color image encryption based on one-time keys and robust chaotic maps. *Comput. Math Appl.* **59**, 3320–3327 (2010)
7. Ye, J., Xu, Z., Ding, Y.: Image search scheme over encrypted database. *Future Gener. Comput. Syst.* (2018)
8. Salahat, E., Qasaimeh, M.: Recent advances in features extraction and description algorithms: a comprehensive survey. In: 2017 IEEE International Conference on Industrial Technology (ICIT), pp. 1059–1063. IEEE (2017)
9. Lowe, G.: SIFT—the scale invariant feature transform. *Int. J* **2**, 91–110 (2004)

10. Choras, R.S.: Image feature extraction techniques and their applications for CBIR and biometrics systems. *Int. J. Biol. Biomed. Eng.* **1**(1), 6–16 (2007)
11. Gudmundsson, G.P., Jonsson, B.P., Amsaleg, L.: A large-scale performance study of cluster-based high-dimensional indexing. In: Proceedings of the International Workshop on Very-Large-Scale Multimedia Corpus, Mining and Retrieval (pp. 31–36). ACM (2010)
12. Yang, W., Hou, K., Liu, B., Fanhua, Yu., Lin, L.: Two-stage clustering technique based on the neighboring union histogram for hyperspectral remote sensing images. *IEEE Access* **5**, 5640–5647 (2017)

Impact of Dimension Reduced Spectral Features on Open Set Domain Adaptation for Hyperspectral Image Classification



Krishnendu C. S., V. Sowmya, and K. P. Soman

Abstract Hyperspectral image classification has so many applications in the area of remote sensing. In recent years, deep learning has been accepted as a powerful tool for feature extraction and ensuring better classification accuracies. In this paper, model for HSI classification is created by implementing open set domain adaptation and generative adversarial networks (GAN). Open set domain adaptation is a type of domain adaptation where target has more classes which are not present in the source distribution. Huge dimension of hyperspectral image needs to be reduced for an efficient classification. In this work, we analysed the effect of dimensionality reduction for open set domain adaptation for hyperspectral image classification by using dynamic mode decomposition (DMD) technique. Experimental results show that 20% of the total available bands of Salinas and 30% of the bands of PaviaU dataset are the highest achievable reduction in feature dimension that results in almost same classification accuracy.

Keywords Image classification · Open set domain adaptation · GAN · Dimension reduction

1 Introduction

Spectral imaging is a technique where a full spectrum or a part of the spectral information is gathered from each location on the image plane and is processed. Image cubes are usually used to represent spectral images. Hyperspectral imaging has many applications in the field of agriculture, astronomy, chemical imaging, eye care, food

Krishnendu C. S. (✉) · V. Sowmya · K. P. Soman
Amrita School of Engineering, Centre for Computational
Engineering and Networking (CEN), Amrita Vishwa Vidyapeetham, Coimbatore,
TamilNadu, India
e-mail: krishnenduchaliyathu@gmail.com

V. Sowmya
e-mail: v_sowmya@cb.amrita.edu

processing, mineralogy, remote sensing and surveillance [1]. Hyperspectral images gather image information from hundreds of narrow spectral bands. Every pixel has continuous spectrum information and it is very useful in recognising the class to which it belongs to, with great precision and accuracy. Since hyperspectral data has complex characteristics, some traditional machine learning algorithms cannot make accurate classification. In recent years, deep learning became a powerful tool for image processing tasks.

In certain scenario, when the training and testing data belong to different domains, the classifier cannot classify properly and the accuracy is reduced [2]. To overcome this problem, there is an algorithm called domain adaptation. Domain adaptation is a technique which deals with situation where a model is trained on a source (training) domain is used in a different but related target (testing) distribution. Mainly there are two types of domain adaptation: closed set domain adaptation and open set domain adaptation. However, most image processing tasks describe closed set domain adaptation. Both training (source) and testing data (target) contain common classes in closed set domain adaptation. There are so many algorithms or models used for domain adaptation. Maximum mean discrepancy (MMD) is one of the techniques for domain adaptation, which measures the distance between the domains [3]. In this work, we used an open set domain adaptation approach, where only some classes or objects are shared among source and target [2]. By using open set domain adaptation, the main aim is to identify unknown samples in target as unknown and known target samples as known [4].

To classify the unknown target sample as unknown, we have not provided any labelled target samples during training [4]. Instead of providing externally, we used a technique called generative adversarial networks (GAN) [4]. GAN consists a generator and a classifier. Generator considers random images as input and it continuously challenge the classifier by imitating input data. Classifier or discriminator learns to determine whether a sample is from generator data or from original input data [5]. High classification accuracy can be achieved by continuously training between generator and classifier in the network.

Since hyperspectral imaging captures very fine spectral characteristics of the scene, it has high dimension and it results in highly complex computation [6]. The bands of HSI are correlated and the information provided by different bands are similar. One method to solve this problem is dimension reduction of the images. Dimension reduction is a process of projecting data into lower dimension feature space and this process helps to reduce the correlated bands. Ideally, the reduced version of the data should have a dimension in a manner conforming with the inherent dimensionality of the data. The inherent dimensionality of data is the least possible number of parameters needed to capture the important properties of the data [7].

Many dimension reduction techniques are being used in the area of machine learning such as principal component analysis (PCA) and singular value decomposition (SVD) [8]. Some techniques are incapable for large dimension data. Dynamic mode decomposition (DMD) is one of the newly incorporated technique for HSI classification [9]. In [9], they compared DMD with some standard techniques such as PCA, SVD and the results show that DMD is comparable with other techniques in reducing

redundancy of bands. In [10], they have used DMD on different Kernel-based HSI classification. In this paper, we analysed the effect of dynamic mode decomposition (DMD) for hyperspectral image classification along with open set domain adaptation. The performance of the model is analysed before and after dimension reduction in terms of classification accuracies.

The flow of the paper is as follows: Methodology is described in Sect. 2 and in Sect. 3 dataset description is given. In Sect. 4, experimental results and conclusion is provided in the last section.

2 Methodology

At the beginning, the model is trained with hyperspectral raw data and the performance of the model is evaluated. Figure 1 shows the flow of the proposed work.

In this work, we used a GAN model for open set domain adaptation for HSI classification [4]. Label dimension of source (training data) is considered as K , and for target class (testing data) as $K + 1$. Here, we have not provided any samples to train for unknown class [11]. Instead, we used GAN model to generate samples to train for unknown class. When input data is passed into the network, the generator generates images. Generator always challenge the classifier by increasing error rate. This generated images is then passed as input to the classifier. The purpose of classifier is to classify known class as known and unknown class as unknown. Classifier creates a margin between known and unknown classes. Output dimension will be $K + 1$ logits and softmax function converts these logits into class probabilities. Here, we train the classifier to give output as $P(y = K + 1/x_t) == t$, where y is the label, x_t is the target sample and t should be in between 0 and 1. Here, we took t as 0.5. If the probability value less than 0.5 the sample is identified as known and if t is greater than 0.5, the sample is identified as unknown.

Since hyperspectral data have huge dimension, time taken for computation is more. The main goal of this work is to analyse the impact of dimension reduction on

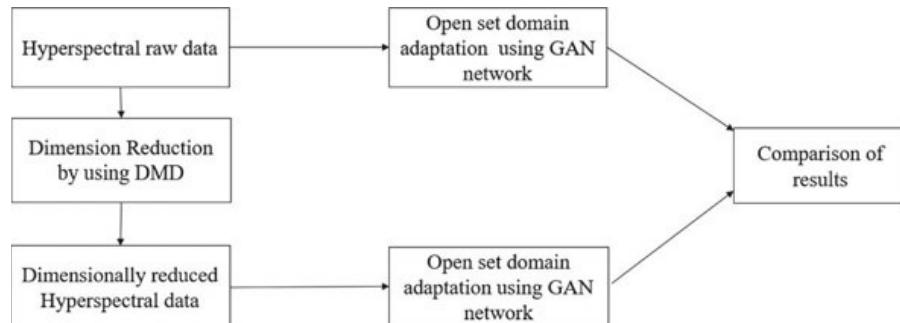


Fig. 1 Block diagram of proposed HSI classification using dimension reduced spectral features and open set domain adaptation

open set domain adaptation for hyperspectral image classification. We used DMD as dimension reduction technique and it is used for the analysis of nonlinear systems. This method is used to extract dynamic modes (eigenvalues and eigenvector representation) from a nonlinear system. First, we vectorised the data in $mn \times b$ format, where mn represents the number of pixels in each band and then appended the bands (b) as columns to create a 2D matrix. Then, the last column was replicated and added to the matrix. This obtained matrix is then separated into two matrix x_1 and x_2 . SVD of x_1 is computed and a low rank approximation matrix S is obtained [9]. By taking Q-R decomposition of eigenvector of S , we were able to compute the permutation matrix P which contains the order in which eigenvectors are arranged [12]. Bands are arranged in the matrix such a way that the top of the matrix contains most informative bands and least informative bands are at the bottom. We reduced the spectral features percentage wise and for each percentage of reduction of bands, the accuracy of the model is being noted. 50% of the features (bands) are reduced initially, and the feature dimension is reduced till highest possible reduction of the features is achieved without any data loss for feature extraction. Different values of learning rate were employed for each percentage of reduction of bands. The learning rate corresponds to the highest classification accuracy is used in the proposed work.

3 Dataset Description

This section provides the details of the dataset used for experimentation. Salinas is hyperspectral dataset collected over Salinas Valley, California using AVIRIS sensor [13]. The spatial resolution for this dataset is 3.7-metre pixels. The dimension of the data is $512 \times 217 \times 224$ and total number of classes is 16. Table 1 provides the class and sample details. The image includes bare soils, vegetables and vineyard fields. Out of total 16 classes, 6 classes are taken as source (Training data) and 8 classes of Salinas dataset are taken as target (testing data). The train test split is 80–20% only for 6 classes which are common for both source and target. And also 2 classes with around 5702 samples is used only for testing (target). These 5702 samples are obtained by combining two classes namely broccoli green weeds 2 and fallow of the original dataset.

PaviaU is one of the two scenes obtained using ROSIS sensor, from Pavia, Northern Italy [13] which are Pavia centre and Pavia university. Pavia university image comprises of 610×610 pixels and number of spectral bands are 103. The geometric resolution is 1.3 m and it contain total 9 classes. Table 2 shows the dataset description of PaviaU dataset with 5 classes as source, and PaviaU dataset with 8 classes as target.

For one of our experiment, 5 classes of PaviaU are common for both source (training) and target (testing). The train test split is 80–20% only for 5 classes which are common for both source and target. Three classes with around 23,278 samples is used only for testing (target). These 23,278 samples are obtained by combining three classes namely meadows, self-blocking bricks and shadows.

Table 1 Dataset description for Salinas dataset [13]

Class	Class labels	Source (Salinas with 6 class)	Target (Salinas with 8 classes)
0	Brocoli_green_weeds 1	1607	40
1	Corn_senesced_green weeds	2622	656
2	Lettuce_romaine_4wk	854	214
3	Lettuce_romaine_5wk	1542	385
4	Lettuce_romaine_6wk	733	183
5	Lettuce_romaine_7wk	856	214
6	Unknown class (Brocoli_green_weeds 2 and fallow)	0	5702
	Total samples	8214	7756

Table 2 Dataset description for PaviaU with 5 classes (source) and PaviaU with 8 classes (target) [13]

Class	Class labels	Source (PaviaU with 5 classes)	Target (PaviaU with 8 classes)
0	Asphalt	5305	1325
1	Gravel	1679	420
2	Trees	2451	613
3	Painted metal sheets	1076	269
4	Bare soil	4023	1006
5	Unknown class (meadows, self-blocking bricks, shadows)	0	23,278
	Total samples	14,534	26,911

4 Experiment Results

The proposed work is experimented on two datasets and evaluated through classification accuracies. The same architecture is used for dimensionally reduced data. The parameters which are used for raw dataset for obtaining good results are 128 batch size and 500 epochs. Learning rate of 0.0001 is used for both datasets and Adam optimisation method and cosine ramp down is used for training network [4]. Hardware configurations which are used for computation are Intel Core i5-8250U with clock speed of 1.6 GHz, driver version of GPU is NVIDIA-SMI 391.25. Cuda 10.0 is also used for computation purpose.

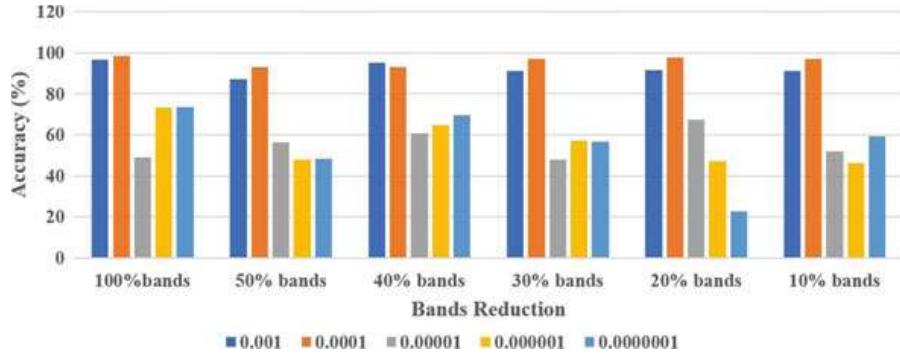


Fig. 2 Classification accuracy of the Salinas dataset for different learning rates

4.1 Dataset 1-Salinas

For Salinas dataset, Salinas with 6 classes is taken as source and target is Salinas with 8 classes. Figure 2 shows the classification accuracies obtained of Salinas dataset before and after dimension reduction for different learning rates. It is evident that the accuracy is almost same for 20% (44 bands) of bands as that of raw data with a learning rate of 0.0001. Figure also shows that the highest classification accuracy of 98.42% is achieved for the learning rate of 0.0001 for the original data consisting of 224 bands. Unknown class accuracy is 98.40% and known class accuracy is 98.00%. Hence, the learning rate is set to 0.0001. Likewise, for 44 bands (20% of the total bands), the highest classification accuracy of 97.66% is obtained for the learning rate of 0.0001. Unknown class accuracy is 98.45% and known class accuracy is 96.88%. So, it is fixed to 0.0001. For Salinas dataset, 20% of the bands is enough to obtain comparable accuracy. Table 3 shows the analysis of classification accuracies before and after dimension reduction and it is also showing the time taken for training the network.

Table 4 shows the classwise accuracy of Salinas dataset after reduction of 20% of bands (44 bands). It is evident that unknown class accuracy after reduction is still remains same as before reduction.

$$\text{Classwise Accuracy} = \frac{\text{No. of pixels correctly classified in each class}}{\text{Total no. of pixels in each class}} \times 100 \quad (1)$$

$$\text{Overall Accuracy} = \frac{\text{Total no. of pixels correctly classified}}{\text{Total no. of pixels}} \times 100 \quad (2)$$

Table 3 Classification results for Salinas dataset

	Without dimension reduction	With dimension reduction (20% of bands) using DMD
Overall accuracy (%)	98.42	97.66
Elapsed time (min)	98	48

Table 4 Classwise accuracies before and after dimension reduction

Class	Class labels	Before dimension reduction (224 bands)	After dimension reduction (44 bands)
0	Brocoli_green_weeds 1	99.75	100
1	Corn_senesced_green weeds	98.78	87.80
2	Lettuce_romaine_4wk	92.05	98.58
3	Lettuce_romaine_5wk	100	100
4	Lettuce_romaine_6wk	98.36	97.26
5	Lettuce_romaine_7wk	99.06	97.66
6	Unknown class (Bro- coli_green_weeds2 and fallow)	98.40	98.45

4.2 Dataset 2:PaviaU

As our second experiment, PaviaU dataset is used for training the network. Five classes are used as source and 8 classes are used as target. PaviaU dataset is considered to check the authenticity of effect of dimension reduction on OS domain adaptation for HSI. Figure 3 shows the classification accuracies achieved of PaviaU dataset before and after dimension reduction for different learning rates. It shows a comparative study of accuracy varying on different learning rates for PaviaU dataset. It is clear that accuracies are almost same for 30% (31 bands) of bands at learning rate of 0.0001.

Figure also shows that the highest classification accuracy of 83.10% is achieved for the learning rate of 0.0001 for the raw data consisting of 103 bands. Unknown class accuracy is 86.22% and known class accuracy is 60.42%. Hence, the learning rate is set to 0.0001. Likewise, for 31 bands (30% of the total bands), the highest classification accuracy of 82.3% is achieved for the learning rate of 0.0001. So, it is set to 0.0001. Unknown class accuracy is 85.74% and known class accuracy is 59.42%. For PaviaU dataset, 30% of the bands is enough to obtain comparable accuracy along with reduced computational time. Table 5 shows the analysis of classification accuracies before and after dimension reduction and it is also shows the time taken for training the network. It is obvious from the table that the computational time is also reduced after dimension reduction.

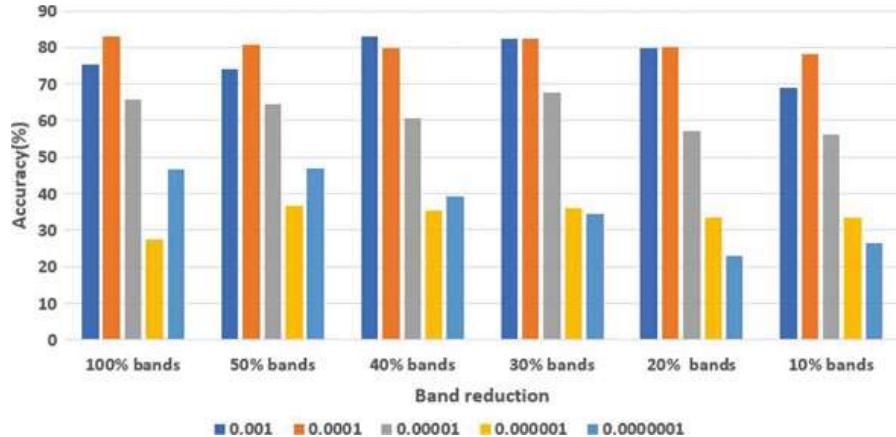


Fig. 3 Classification accuracy of the PaviaU dataset for different learning rates

Table 5 Classification results for PaviaU dataset

	Without dimension reduction	With dimension reduction (30% of bands) using DMD
Overall accuracy (%)	83.10	82.30
Elapsed time (min)	216.83	118.5

Table 6 Classwise accuracies before and after dimension reduction for PaviaU dataset

Class	Class labels	Before dimension reduction (103 bands) (%)	After dimension reduction (31 bands) (%)
0	Asphalt	99.62	95.92
1	Gravel	12.61	22.14
2	Trees	65.41	61.82
3	Painted metal sheets	99.62	99.62
4	Bare soil	24.85	17.59
5	Unknown class (meadows, self-blocking bricks, shadows)	86.22	85.74

Table 6 shows classwise accuracy of PaviaU dataset after reduction of 30% of bands (31 bands). It is evident from the table that unknown class accuracy after reduction is still remains same as before reduction. And also, we see that some of the known class accuracies are improved after dimension reduction.

5 Conclusion

In this work, we applied open set domain adaptation along with GAN for HSI classification and the performance is analysed before and after dimension reduction of hyperspectral images. Dynamic mode decomposition (DMD) is used as the dimension reduction technique and the results show that this approach is very advantageous in removing redundancy between bands. It performed well on open set domain adaptation for HSI classification. The main aim of the model is to classify the classes which were not present during training as unknown. From the experimental results for the two dataset, it is obvious that the network is capable to achieve comparable classification accuracy as that of raw data of HSI even after the reduction in feature dimension. Here, the computation time is also reduced after dimension reduction for both dataset. Also, the experimental results show that, 20% of the bands (44 bands) of the total available bands of Salinas dataset and 30% of the bands for PaviaU dataset are the highest possible reduction in feature dimension that results in almost same classification accuracy.

References

1. Lodha, S.P., Kamlapur, S.M.: Dimensionality reduction techniques for hyperspectral images. *Int. J. Appl. Innov. Eng. Manag. (IJAIEM)* **3**(10), 92–99 (2014)
2. Busto, P.P., Gall, J.: Open set domain adaptation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 754–763 (2017)
3. Gretton, A., Borgwardt, K., Rasch, M., Scholkopf, B., Smola, A.J.: A kernel method for the two-sample-problem. In: Advances in Neural Information Processing Systems, pp. 513–520 (2007)
4. Saito, K., Yamamoto, S., Ushiku, Y., Harada, T.: Open set domain adaptation by backpropagation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 153–168 (2018)
5. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
6. Koonsanit, K., Jaruskulchai, C., Eiumnoh, A.: Band selection for dimension reduction in hyper spectral image using integrated information gain and principal components analysis technique. *Int. J. Mach. Learn. Comput.* **2**(3), 248 (2012)
7. Fukunaga, K.: Introduction to Statistical Pattern Recognition. Elsevier, San Diego (2013)
8. Fong, M.: Dimension reduction on hyperspectral images. Univ. California, Los Angeles, CA (2007)
9. Megha, P., Sowmya, V., Soman, K.P.: Effect of dynamic mode decomposition-based dimension reduction technique on hyperspectral image classification. In: Computational Signal Processing and Analysis, pp. 89–99. Springer, New York (2018)
10. Charmisha, K.S., Sowmya, V., Soman, K.P.: Dimensionality reduction by dynamic mode decomposition for hyperspectral image classification using deep learning and kernel methods. In: Thampi, S.M., Marques, O., Krishnan, S., Li, K.-C., Ciouzo, D., Kolekar, M.H. (eds.) Advances in Signal Processing and Intelligent Recognition Systems, pp. 256–267. Springer, Singapore (2019)
11. Bendale, A., Boult, T.E.: Towards open set deep networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1563–1572 (2016)

12. Bharath Bhushan, D., Sowmya, V., Sabarimalai Manikandan, M., Soman, KP.: An effective pre-processing algorithm for detecting noisy spectral bands in hyperspectral imagery. In: 2011 International Symposium on Ocean Electronics, pp. 34–39. IEEE (2011)
13. U del Pais Vasco. <http://www.ehu.es/ccwintco/index.php/hyperspectral-remotesensing-scenes>. Accessed 25 Aug 2012

Fog-Based Video Surveillance System for Smart City Applications



B. V. Natesha and Ram Mohana Reddy Guddeti

Abstract With the rapid growth in the use of IoT devices in monitoring and surveillance environment, the amount of data generated by these devices is increased exponentially. There is a need for efficient computing architecture to push the intelligence and data processing close to the data source nodes. Fog computing will help us to process and analyze the video at the edge of the network and thus reduces the service latency and network congestion. In this paper, we develop fog computing infrastructure which uses the deep learning models to process the video feed generated by the surveillance cameras. The preliminary experimental results show that using different deep learning models (DNN and SNN) at the different levels of fog infrastructure helps to process the video and classify the vehicle in real time and thus service the delay-sensitive applications.

Keywords Fog computing · Edge computing · IoT · Deep learning models · Latency · Analytics · Classification · Video surveillance

1 Introduction

The development of IoT enabled many things/devices such as sensors, cameras, and other devices to connect to the Internet. Nowadays, cameras are deployed in many places to monitor and control the events. These camera devices will generate a massive amount of video data while tracking and thus require these videos to be streamed to the centralized cloud to process and analyze. But, streaming such

B. V. Natesha (✉) · R. M. R. Guddeti
Department of Information Technology,
National Institute of Technology Karnataka, Mangalore, Karnataka, India
e-mail: nateshbv18@gmail.com

R. M. R. Guddeti
e-mail: profgrmreddy@nitk.edu.in

extensive video data to the centralized cloud consumes a lot of bandwidth and also increases the network congestion. Thus, transferring this video data to the centralized cloud will increase the service time/latency, cloud resource consumption, and the service cost.

The applications such as smart video surveillance, smart parking, Industrial IoT applications (IIoT), and smart car are delay-sensitive and require service/response in real time so that it can reduce the cost and some critical failures in the smart industrial (Industry 4.0) environment. Thus, processing a video close to the edge devices (cameras) will reduce the amount of data to be transferred to the cloud and also enable to service the delay-sensitive applications in real time. Hence, pushing intelligence to the edge of the network using distributed fog computing architecture will reduce the latency, bandwidth requirement, and also the cloud resource usage.

Fog computing is defined as the distributed computing architecture which provides the storage, processing, and networking resources close to the data source devices. It acts as the intermediate computing infrastructure between the cloud and IoT. Hence, using fog nodes at the edge of the network will enable to process and analyze the videos to extract the most relevant information from the videos such as real-time object detection, anomaly detection, tracking objects/vehicle, provide parking information, and traffic control in real time.

Some of the advantages of using fog computing architecture are:

- Low latency/quick decision
- Handle heterogeneous data
- Edge intelligence and analytics
- Reduce cloud resource usage and
- Minimizes the service cost.

The network devices such as smart gateways [1], router, and micro data centers (MDCs) [2] are used as the fog nodes or fog/edge servers for processing and analyzing the videos. The cameras are interfaced with the fog nodes and deployed the deep learning models at the edge to analyze the videos to get the required information and transfer that information to the cloud for further processing or storage. Video analysis at the fog computing architecture requires computational resources to deploy and run the deep learning model to process and analyze the videos. Using the fog nodes with external hardware and fog server (MDCs at the edge) will provide the computational resources to deploy and run the deep learning models.

We developed a fog computing infrastructure which can be used for processing the videos and extract the relevant information to make decisions in real time. We develop the fog infrastructure by using the Raspberry Pi with Intel Neural Compute Stick (NCS) and the laptop as the fog node and server, respectively, at different levels. Then, deploy the deep learning models such as deep neural network (DNN) and shallow neural network (SNN) on these fog nodes for processing and analyzing the videos. In this paper, we present the preliminary work done for classifying the vehicles in the fog computing environment so that it will be useful in smart surveillance, smart parking, and the automated toll collection applications.

The rest of the paper is organized as follows: Sect. 2 discusses the related work, the proposed methodology is presented in Section 3, results and analysis are explained in Section 4, and finally, the paper is concluded in Section 5.

2 Related Work

Christos Stergiou et al. [3] considered cloud as the computing architecture for processing and analyzing the videos to make the decision and store the data. But, fog-based video analysis is not considered in real time. Sultana and Wahid [4] developed an event-based fog framework for crime detection and confirmation. The developed framework is used for security surveillance for crime detection by detecting the knife and gun objects in the video frames. Yang et al. [5] developed a Wi-Fi-enabled IoT platform for human occupancy detection based on the channel state information curve of the human presence. They used a cloud computing platform for activity recognition using machine learning techniques, which takes more time to provide the service.

Chen et al. [6] developed a dynamic video surveillance framework for object/vehicle tracking using drones and laptops as the fog computing nodes. But, they did not consider the vehicle classification. Diro et al. [7] developed deep learning-based cyber-attack detection for IoT using fog computing architecture. The experiments proved that the distributed fog computing-based attack detection is more scalable and effective than the centralized cloud architecture, but they did not consider the two-level fog architecture for processing the video. Li et al. [8] developed a deep learning-based model for the defect detection and degree of defect detection using fog computing for smart industrial applications, but they did not consider solving the classification problem in the fog computing environment.

Constant et al. [9] proposed a smart fog gateway-based end-to-end architecture for processing the wearable IoT devices data for different smart applications. Their model considered data conditioning and intelligent data filtering to decide what is to be sent to the centralized cloud for further processing and long-term storage. But, they did not consider processing the video in the fog computing environment. Jianyu Wang et al. [10] developed an architecture to adjust the processing capacity and route the data to the proper computing nodes in the architecture. They considered elastic resource provisioning for processing the videos in the developed architecture, but they did not consider deploying deep learning models in the fog nodes.

Some of the existing works consider the generic approach based on the pre-trained convolutional network for detecting the objects [11] and predicting the defect area through extracting the path features [12]. But none of the existing architecture considered processing the video at the two-level fog computing architecture so that it will extract the most relevant information at the lower level.

3 Proposed Methodology

3.1 System Architecture for Video Analysis

We considered the fog computing architecture for processing the videos at the network edge, such that we can push the intelligence to the edge to extract the relevant information, and it can be used to make the decision in real time. Thus, processing video at the network level will reduce the size of the video to be transferred to the centralized cloud. Using fog computing for processing the video will reduce the service latency, network bandwidth requirement, and the congestion overhead as well. In this work, we considered two processing approaches to process the video and extract the information, namely processing at the fog node or using fog server at the intermediate level between the cloud and IoT.

We used the deep neural network (DNN) and shallow neural network (SSN) model at the edge server and fog node, respectively, to analyze and extract the information from the incoming video feeds from the deployed cameras. The SSN model (tiny versions) is a low computational model that can be used at the fog node, which is computationally less powerful. These fog nodes can be used for processing the less complex tasks such that it can extract the information and transfer it into the fog server or cloud for further processing. The DNN is the more powerful neural network model which can be used for processing the more complex tasks and videos very fast as compared to the SSN and deploy these DNN model on the fog server.

In the developed fog architecture, we used the fog node at the lower level to perform the initial analysis and extract the information; thus, it can reduce the size of the data. In the lower level of the fog layer, the devices have the limited computational resources, and therefore, to enable them to process the video, we make use of the Intel NCS,¹ which provides the pre-trained neural network model to analyze the video to detect and classify the objects in real time. The local fog node-based computing architecture is shown in Fig. 1a. The other fog computing architecture, where a micro data center can be used as the fog server in between the cloud and IoT/camera devices, which runs the DNN model is shown in Fig. 1b. Based on the application requirement, we can make use of any of these architectures or combinations of this architecture to process and analyze the video in the fog computing infrastructure.

4 Experimental Setup and Results Analysis

4.1 Experimental Setup

We make use of our laptop as the fog server at the network edge and Raspberry Pi along with the Intel NCS as the local fog node for processing the videos at the fog

¹<https://software.intel.com/en-us/neural-compute-stick/>.

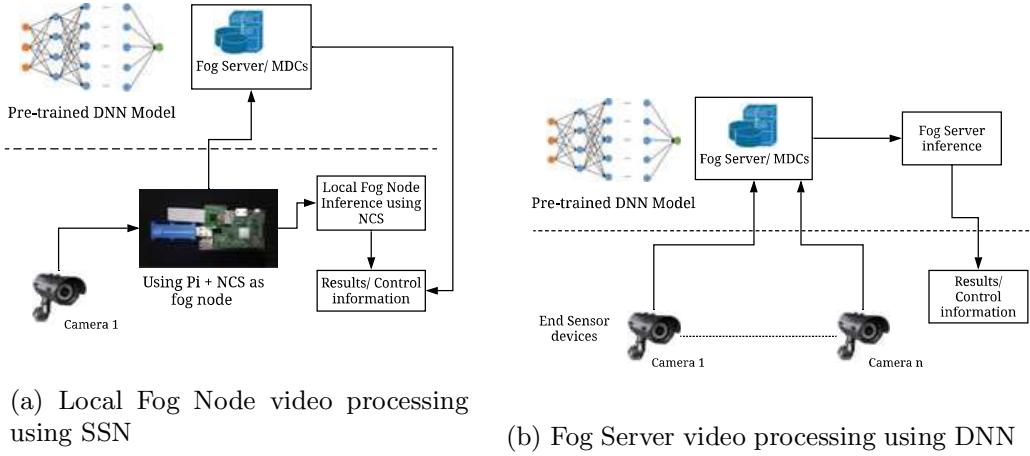


Fig. 1 Fog computing architectures

layer. We train and deploy the YOLOv3 as the DNN model at the fog server level to process the complex video at a faster rate, but we make use of the TinyYOLO at the fog node to process video in the less complex environment to process the video on the low computational resources. The system configuration used to carry out the experiments for the fog server is 1.8GHz Intel Core i5 processor with 8GB RAM, and the Raspberry Pi 3 model with camera acts as the fog node for processing the video locally using the Intel NCS environment. The experiment setup is shown in Fig. 2.

To carry out the experiment on the developed system architecture, we considered classifying the vehicles in the surveillance video feed. So that the classified vehicle information can be used for toll collection in smart toll gates based on the type of vehicles. Further, it can be used for the intelligent surveillance system in the campus environment to keep a record of the vehicle entered and also to notify the available parking locations in real time.

4.2 Dataset

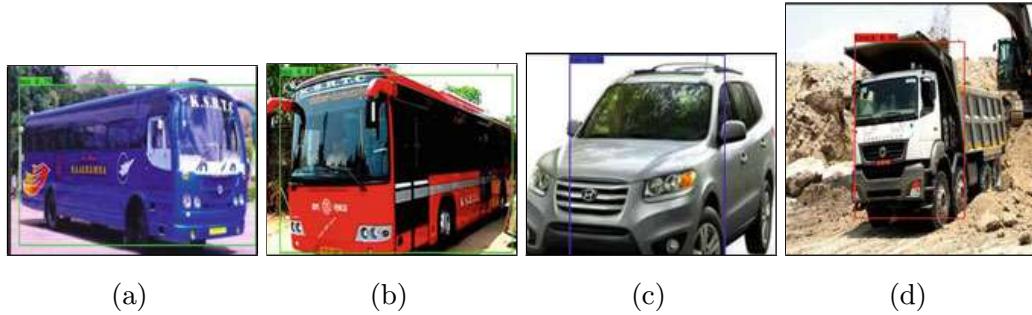
To perform vehicle classification, we prepared a dataset where a single vehicle is present in each frame. Then, we perform data augmentation to get a large number of images by performing some operations on the image frame, and then, manual annotation is done for the generated images. We used YOLOv3 [13] as the DNN model to run in the fog server node, which is a fast neural network model and requires some powerful computational devices. Used pre-trained weights of YOLOv3 to train with the prepared dataset and fine tune YOLOv3 model to classify the vehicles. Similarly, for SNN to run in the fog node, we used TinyYOLO, and it is trained and then converted to Caffe model to enable it to run in the Intel NCS environment for classifying the vehicles.

Fig. 2 Experimental setup

4.3 Results and Analysis

In this work, we presented the preliminary results obtained by processing the video at two different fog computing architectures, using fog node and the fog server at two levels. The results obtained for the above architecture are shown in Figures 3, 4 and 5. Using different fog computing architectures will help us to process the video in real time. The preliminary results show the classification of the vehicle in the fog computing architecture for the considered dataset.

Figure 3 shows the results obtained for the DNN model in the fog server environment, and Fig. 4 shows the results obtained for the local fog node processing at the lower level. Thus, using fog computing infrastructure between the cloud and IoT to process and analyze the videos at the network edge level reduces the size of the data to be transferred to cloud. Hence, it reduces the service latency and cloud resource consumption for servicing the IoT applications. The confusion matrix obtained for the developed DNN model to classify the vehicle is given in Fig. 5. We obtained an accuracy rate of 80% for the deployed models for classifying the vehicles in the fog computing environment.

**Fig. 3** Experimental results for fog server

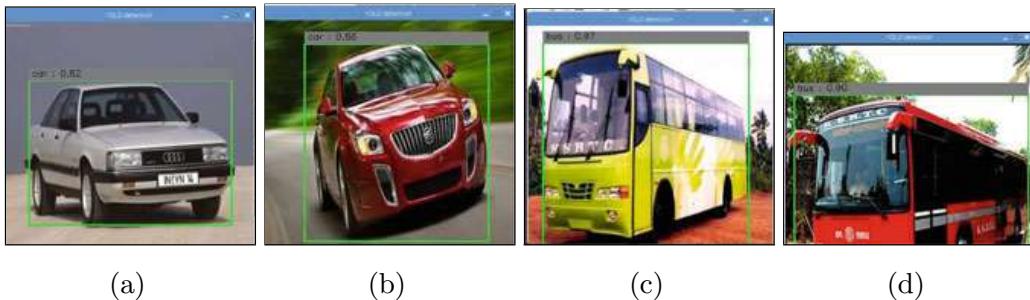
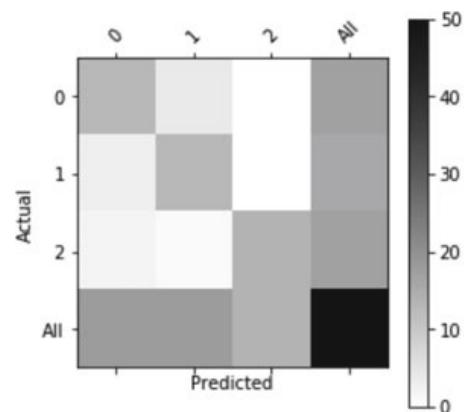


Fig. 4 Experimental results for local fog node with NCS

Fig. 5 Confusion matrix



5 Conclusion

In this paper, we propose a fog infrastructure for processing the video so that we can extract the relevant information at the fog level to reduce the size of the video to be transferred to the cloud for further processing. We considered deploying the deep neural network models in the two-level fog infrastructure, thus enabling the limited computational fog nodes to process and analyze the videos at the network level. We carried out an experiment considering the smart video surveillance application, where we process the video and classify the vehicles. The preliminary experimental results show how these fog nodes/servers at different levels of fog infrastructure will detect and classify the vehicles. Thus, using fog computing architecture for various surveillance applications is the best approach to classify the objects/vehicles in real time.

Our future work is to design and develop the offloading strategy to deploy the service requests in the two-level fog computing infrastructure and also using the developed architecture for different smart surveillance application environments.

Acknowledgements This work has been supported by the Visvesvaraya Ph.D. Scheme for Electronics and IT (Media Lab Asia), the Department of MeitY, Government of India. This work has been carried out at the Department of Information Technology, NITK Surathkal, Mangalore, India.

References

1. Aazam, M., Huh, E.N.: Fog computing and smart gateway based communication for cloud of things. In: 2014 International Conference on Future Internet of Things and Cloud, pp. 464–470. IEEE (2014)
2. Aazam, M., Huh, E.N.: Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT. In: 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, pp. 687–694. IEEE (2015)
3. Stergiou, C., Psannis, K.E., Kim, B.G., Gupta, B.: Secure integration of IoT and cloud computing. Future Gener. Comput. Syst. **78**, 964–975 (2018)
4. Sultana, T., Wahid, K.A.: IoT-Guard: event-driven fog-based video surveillance system for real-time security management. IEEE Access **7**, 134881–134894 (2019)
5. Yang, J., Zou, H., Jiang, H., Xie, L.: Device-free occupant activity sensing using wifi-enabled IoT devices for smart homes. IEEE Internet Things J. **5**(5), 3991–4002 (2018)
6. Chen, N., Chen, Y., You, Y., Ling, H., Liang, P., Zimmermann, R.: Dynamic urban surveillance video stream processing using fog computing. In: 2016 IEEE Second International Conference on Multimedia Big Data (BigMM), pp. 105–112. IEEE (2016)
7. Diro, A.A., Chilamkurti, N.: Distributed attack detection scheme using deep learning approach for Internet of Things. Future Gener. Comput. Syst. **82**, 761–768 (2018)
8. Li, L., Ota, K., Dong, M.: Deep learning for smart industry: efficient manufacture inspection system with fog computing. IEEE Trans. Ind. Inform. **14**(10), 4665–4673 (2018)
9. Constant, N., Borthakur, D., Abtahi, M., Dubey, H. and Mankodiya, K., 2017. Fog-assisted wiot: A smart fog gateway for end-to-end analytics in wearable internet of things. arXiv preprint [arXiv:1701.08680](https://arxiv.org/abs/1701.08680)
10. Wang, J., Pan, J., Esposito, F.: Elastic urban video surveillance system using edge computing. In: Proceedings of the Workshop on Smart Internet of Things, p. 7. ACM (2017)
11. Tuli, S., Basumatary, N., Buyya, R.: EdgeLens: deep learning based object detection in integrated IoT, fog and cloud computing environments. arXiv preprint [arXiv:1906.11056](https://arxiv.org/abs/1906.11056) (2019)
12. Park, J.K., An, W.H., Kang, D.J.: Convolutional neural network based surface inspection system for non-patterned welding defects. Int. J. Precis. Eng. Manuf. **20**(3), 363–374 (2019)
13. Redmon, J., Farhadi, A.: Yolov3: an incremental improvement. arXiv preprint [arXiv:1804.02767](https://arxiv.org/abs/1804.02767) (2018)

Performance Improvement of Deep Residual Skip Convolution Neural Network for Atrial Fibrillation Classification



Sanjana K., V. Sowmya, E. A. Gopalakrishnan, and K. P. Soman

Abstract Atrial fibrillation is a life-threatening cardiac disease which requires a long and tedious process of detection. So, the detection of atrial fibrillation has gained great importance. One of the most reliable ways to detect cardiac disease is through analysis of ECG signal. In this paper, we show that the performance of a deep residual skip convolution neural network-based approach for automatic detection of atrial fibrillation can be improved by hyperparameter tuning. For the present work, atrial fibrillation dataset from the 2017 PhysioNet/CinC Challenge is used. The proposed method obtained an overall accuracy of 96.08% and weighted average F1 score of 0.96, a recall of 0.96 and a precision of 0.96. The main advantage of the present work is the improved accuracy achieved using a lighter model which is trained for a lesser number of epochs.

Keywords Cardiac disease · Atrial fibrillation · Deep residual neural network

1 Introduction

Atrial fibrillation (AF) is a cardiac disease caused due to the malfunctioning of the electrical system in the heart [16]. It is recognized by irregular heartbeat rhythm. Particularly, the heart beats faster than the normal rate when a person is affected by the AF, which cause nausea and even may lead to death [4]. Even though there are

Sanjana K. (✉) · V. Sowmya · E. A. Gopalakrishnan · K. P. Soman
Amrita School of Engineering, Center for Computational Engineering and Networking,
Amrita Vishwa Vidyapeetham, Coimbatore, TamilNadu, India
e-mail: sanjanakala1996@gmail.com

V. Sowmya
e-mail: v_sowmya@cb.amrita.edu

E. A. Gopalakrishnan
e-mail: ea_gopalakrishnan@cb.amrita.edu

K. P. Soman
e-mail: kp_soman@amrita.edu

various studies conducted for the detection of AF, the process of AF detection is long and demanding because of its high risk to life [3]. The first step to treat AF is to detect AF as early as possible. ECG is the most reliable way to detect AF. One of the common techniques used to detect the AF is to look for the absence of P-wave in the ECG signal [12].

Various approaches are used for the detection of the AF but, in most cases, manually extracted features are required [16]. This can be efficiently replaced using the deep learning technique. The deep learning can learn the required features automatically and this helps in the efficient detection of disease. One of the famous deep learning algorithms called convolution neural network is used in various fields such as computer vision, bioinformatics, classification of images, signal classification, etc. [6].

Various studies have been conducted in the field of cardiac disease detection, particularly in the detection of atrial fibrillation. The 2017 PhysioNet CinC challenge was conducted to classify the atrial fibrillation (AF) data from normal and other noisy data. Many teams took part in the challenge in which four teams had shown equal and better score of 0.83 than other teams. Teijeiro et al. [17] used abductive interpretation to derive the handcrafted features and the sequence information from the ECG signal. The features are fed into the embedded recurrent neural network for the classification. Datta et al. [2] proposed a two-layer cascaded binary classifier for the classification of normal, AF and other rhythms. The real classification happens in the second layer of binary classifier before that the records are intermediately classified to normal and others, and AF and noisy in the second layer. Zabihi et al. [18] used a random forest classifier to classify 150 features chosen from 491 handcrafted features into normal, AF, and other rhythms. Hong et al. [8] combined the features extracted using the deep neural networks and extracted features from significant waves for the classification of normal, AF, noisy, and other signals. They used an ensemble classifier for the classification. Kropf et al. [11] proposed a method combining the machine learning and conventional signal analysis approach. They analyzed the different architectures and shown that a gradient boosted tree model is better than the random forest for AF classification. Rizwan et al. [14] used a machine learning algorithm to classify the feature extracted samples from the ECG signal. Together with the dimensionality reduction technique and sparse coding, they used an ensemble decision tree as the classifier.

Kamaleswaran et al. [10] proposed a 13 layer deep convolution neural network-based approach for the classification of the normal, AF, noisy, and other signals. The method also includes the effect of the influence of hyperparameters on model performance. Plesinger et al. [13] proposed an ensemble of convolution neural network and bagged tree for the classification of Holter ECG signal into four classes (Normal, AF, other arrhythmia and other noisy samples).

A deep residual skip convolution neural network (DRSCNN) was proposed by Kachuee et al. [9] for the classification of arrhythmia and used transfer learning to extract information learned by the model to classify the myocardial infarction. Gopika et al. [6] have shown performance improvement for the above-given method by retraining the network. Arrhythmia is a disease characterized by its irregular

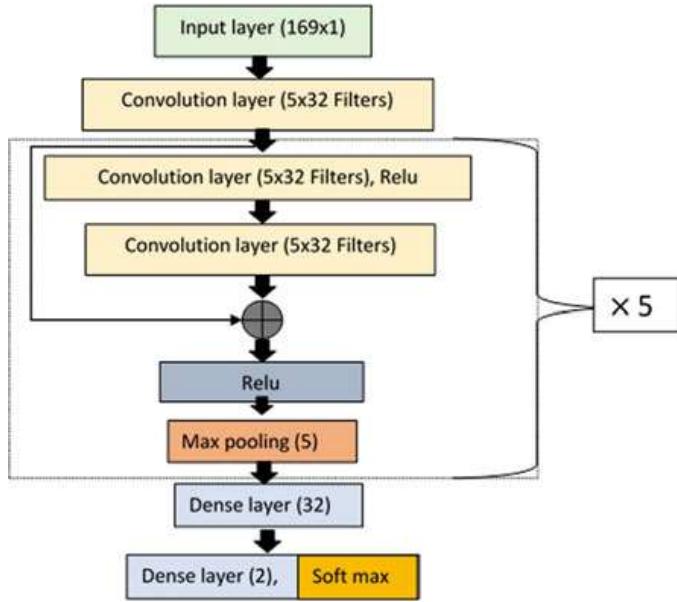
heartbeat rhythm. Atrial fibrillation is one of the types of arrhythmia. Through this knowledge, Gopika et al. [7] used the DRSCNN for the classification of AF data into normal and abnormal using the feature extracted samples extracted from the ECG signal proposed by Andreotti et al. [1]. However, it resulted in minimal performance in the case of AF classification. Gopika et al. [7] used other deep learning models such as long short term memory (LSTM), gated recurrent unit (GRU), recurrent neural network (RNN). All the above-mentioned architectures are trained for 1000 epochs and DRSCNN is trained for 75 epochs. In the existing work [7], all the other deep learning models have shown better performance when trained with a higher number of epochs but the DRSCNN model achieved only minimal performance even though it has comparatively less number of learnable parameter. Thus, the current work aims to improve the performance of the AF classification using DRSCNN by hyperparameter tuning. The present work is organized as follows: Sect. 2 gives the details about the dataset and the architecture of DRSCNN, Sect. 3 describes the methodology used, and experimental results are presented in Sect. 4. Section 5 gives a summary of the present work.

2 Dataset Description and Architecture Details

2.1 Dataset Description

Atrial fibrillation (AF) data is taken from AF classification 2017 PhysioNet/CinC Challenge [5]. The dataset contains 8528 raw ECG signals in which 5154 subjects are normal and 771 atrial fibrillation signals. To decrease the computational complexity and time consumption, the feature extracted vector is fed into the model as input. The feature vector is derived based on the method proposed by Andreotti et al. [1]. In this method, the feature vector is obtained by preprocessing the raw ECG signal using a Butterworth filter of order 10. The preprocessed signal is then segmented into ECG segments of 10 s with an overlap of 50%. The sampling frequency of the signal is 300 Hz. This segmented signal is then applied to the QRS detection algorithm. From these preprocessed samples, the feature is extracted using the heart rate variability metrics and the signal quality indices. The heart rate variability metrics contain time-domain features, frequency domain features, and nonlinear features. The signal quality indices include the height and width of different segments of the ECG signal. All the features together, the input vector is of dimension 169×1 . Out of 45,337 feature extracted samples considered for training, 39,369 are AF signals and 5968 are the normal case. 30% of the total samples were used for testing which is 19,430 feature extracted samples (16,873: Normal 2557: AF).

Fig. 1 Architecture details of the deep residual skip convolution neural network [9]



2.2 Deep Residual Skip Convolution Neural Network

Deep residual skip convolution neural network (DRSCNN) [9] has a convolution layer as the prime layer for the feature extraction. Preceding the convolution layer, there is an input layer which is of dimension 169×1 . As input is a signal, a 1D convolution layer with 32 filters of size 5 is used. Followed by this, the input layer and the convolution layer, the architecture contains 5 residual blocks. The residual block contains a convolution layer with ReLU (Rectifier linear unit) activation, skip connection, and a max-pooling layer. The connections between different layers are illustrated in Fig. 1. The skip connection is used to carry original information directly from the input layer to the final layer of the block, skipping in-between layers. Skip connection is made by adding the input from the previous layers to the preceding layers. The max-pooling layer of size 5 with stride 2 is used for dimensionality reduction. The output of the residual block is given to two fully connected dense layers. The first dense layer consists of 32 neurons. The final dense layer of 2 neurons with softmax activation function acts as the output layer. The number of neurons in the output dense layer depends upon the number of classes considered. The total number of trainable parameters is 54,914 for DRSCNN.

3 Methodology

The proposed method uses a deep residual skip convolution neural network for the classification of the atrial fibrillation and normal ECG samples. The input of the DRSCNN is a feature vector of size 169×1 . For training the CNN, we used Adam

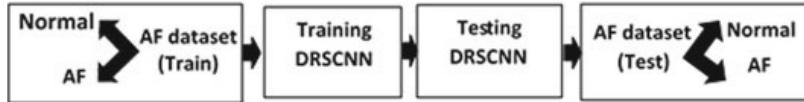


Fig. 2 Proposed methodology for performance improvement of DRSCNN for AF classification

optimizer and categorical cross-entropy as the loss function. The hyperparameters such as the number of epochs and learning rates are tuned in such a way that the model was able to learn the features perfectly. The proposed methodology is illustrated in Fig. 2. The model is trained with 70% of the AF dataset. The trained model is evaluated with 30% of the AF dataset. The metrics such as accuracy, F1 score, precision, and recall are used for the evaluation of the model.

4 Experimental Results

The number of epochs is fixed according to the performance of the model to classify AF. The model was trained for 300 epochs and with a learning rate of 0.001 (default). The deviation of training accuracy and loss of the model concerning the epoch are shown in Figs. 3 and 4.

From Figs. 3 and 4, it is evident that the training loss decreased from infinity to 0.07 till 105th epoch and the training accuracy increased from 0 to 97%. After 105th epoch, the training accuracy decreased for a few epochs and again there is an increase in accuracy till 227th epoch. Later, there is a steep decrease in accuracy. The training loss does not have any vigorous change after 105th epoch till 227th epoch. But after 227th epoch, there is a steep steady increase in loss. This is due to the overfitting of the model, i.e., the model is trained more so that it has started to generalize all the given data. Such cases affect the overall performance of the model. Thus, the model weight in 105th epoch which shown the best training accuracy of

Fig. 3 The variation in accuracy according to the increase in number of epochs for AF classification

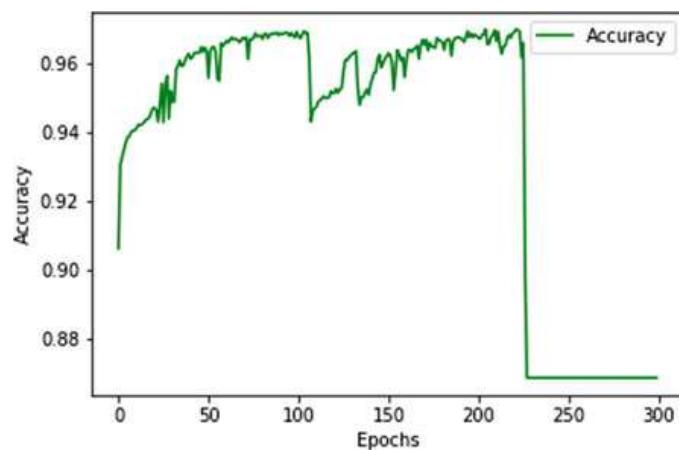


Fig. 4 The variation in loss according to the increase in number of epochs for AF classification

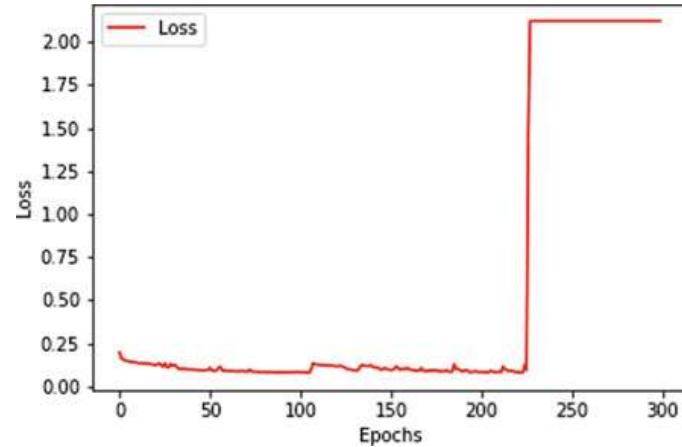


Table 1 Evaluation metric results obtained for the AF classification using DRSCNN and performance of previously existing method

Work	Accuracy (%)	F1 score	Recall	Precision
Proposed	96.08	0.96	0.96	0.96
Gopika et al. [7]	50.00	0.67	0.50	0.25

97% and loss of 0.07 was used for testing. The evaluation of the model efficacy is carried out using metrics such as accuracy and weighted average F1 score, precision, and recall. The evaluation metric acquired by the model together with the values acquired by the existing method is shown in Table 1. The proposed model acquired an accuracy of 96.08% and weighted average F1 score of 0.96 which is higher than other prevailing methods. The confusion matrix containing the true positive, false positive, true negative, and false negative is shown in Fig. 5. From the confusion matrix, it is clear that the model classified 98% of the normal data as normal (True negative) and 84% of the AF data as abnormal (True positive).

The receiver operating characteristics (ROC) and area under the curve (AUC) are other parameters that measure the efficiency of the model. In the ROC curve, if the graph goes more toward the upper left corner, it indicates better performance. AUC is the area covered by the curve and it measures the quality of performance. Higher the area, better the performance. The ROC curve and AUC are shown in Fig. 6. The AUC for the AF classification is 0.96. From the results, we understand that the DRSCNN

Fig. 5 Confusion matrix for AF classification obtained by hyperparameter tuning of DRSCNN model

		Predicted label	
		N	A
True label	N	16522	351
	A	411	2146

Fig. 6 ROC curve and AUC for AF classification obtained by hyperparameter tuning of DRSCNN model

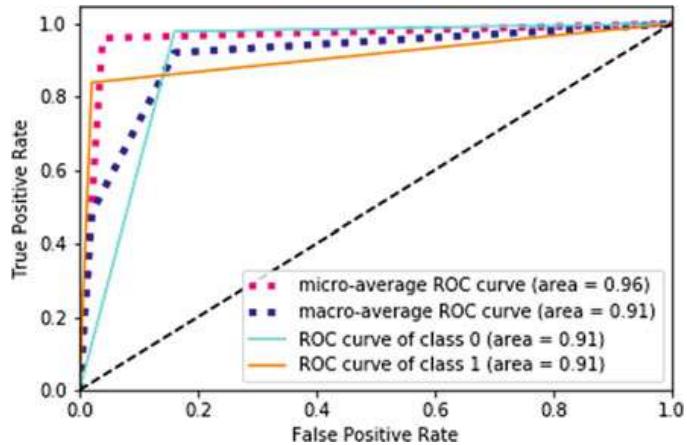


Table 2 Performance comparison for the proposed method and the method implemented in the literature for AF classification

Authors	F1_score normal	F1_score AF
Proposed	0.98	0.86
Teijeiro et al. [17]	0.92	0.86
Datta et al. [2]	0.92	0.82
Zabihi et al. [18]	0.91	0.84
Hong et al. [8]	0.92	0.85
Kamaleswaran et al. [10]	0.91	0.82
Plesinger et al. [13]	0.92	0.82
Kropf et al. [11]	0.91	0.84
Rizwan et al. [14]	0.91	0.78
Gopika et al. (GRU) [7]	0.96	0.87
Gopika et al. (DRSCNN) [7]	0.33	0.00

model classified the AF data with 96% accuracy. The performance comparison of DRSCNN with the existing approaches is shown in Table 2. From the table, it is evident that the performance of the proposed approach is 4% higher than the existing methods in the literature in the case of normal class classification and has shown equal performance in case of classification of abnormal class.

Unlike the recurrent neural networks used by Gopika et al. [7] which was trained for 1000 epochs, the present work attains the benchmark accuracy by training the DRSCNN network for just 105 epochs. The number of learnable parameters of DRSCNN is less than LSTM and the same in the case of GRU. Even though the number of learnable parameters for RNN is less than DRSCNN, the number of epochs used to train the DRSCNN is lesser than the number of epochs used to train the RNN model. Hence, the highlight of the present work is the attainment

of benchmark accuracy using the lighter model based on the number of learnable parameters and trained for less number of epochs. This was feasible due to the beat segmented features containing the heart rate variability in time and frequency domain existing in the literature [1].

5 Conclusion

Our model achieved a 46% increase in overall accuracy; 65% increase in F1 score for the classification of normal class; high performance in case of detection of abnormal class when compared to the existing DRSCNN model for AF classification. We also show that the proposed method shown a 2% increase in F1 score to classify normal data and an equal performance in case of abnormal classification when compared to existing methods in the literature. The main contribution of our work is that we achieved better performance with a lesser number of learnable parameters and less number of epochs when compared to existing deep learning models for AF classification. Thus, from this work, it is evident that although increase in the number of epochs increased the classification accuracy, the number of epochs required to train the deep learning architectures with feature extracted data is less than that exist in the literature for AF classification.

References

1. Andreotti, F., Carr, O., Pimentel, M.A.F., Mahdi, A., De Vos, M.: Comparing feature-based classifiers and convolutional neural networks to detect arrhythmia from short segments of ECG. *Comput. Cardiol. (CinC) Rennes* **2017**, 1–4 (2017)
2. Datta, S., et al.: Identifying normal, AF and other abnormal ECG rhythms using a cascaded binary classifier. *Comput. Cardiol. (CinC) Rennes* **2017**, 1–4 (2017)
3. Ganesan, A.N., et al.: Long-term outcomes of catheter ablation of atrial fibrillation: a systematic review and meta-analysis. *J. Am. Heart Assoc.* **2**(2), e004549 (2013)
4. Go, A.S., et al.: Prevalence of diagnosed atrial fibrillation in adults: national implications for rhythm management and stroke prevention: the Anticoagulation and Risk Factors in Atrial Fibrillation (ATRIA) Study. *JAMA* **285**(18), 2370–2375 (2001)
5. Goldberger, A.L., Amaral, L.A.N., Glass, L., Hausdorff, J.M., Ivanov, P.Ch., Mark, R.G., Mietus, J.E., Moody, G.B., Peng, C.-K., Stanley, H.E.: PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circ.* **101**(23), e215–e220 (2003)
6. Gopika, P., et al.: Performance improvement of residual skip convolutional neural network for myocardial disease classification. In: International Conference on Intelligent Computing and Communication Technologies. Springer, Singapore (2019)
7. Gopika, P., Sowmya, V., et al.: Transferable approach for cardiac disease classification using deep learning. *Deep Learn. Biomed. Health Inform. (BHI)* (2019, in press)
8. Hong, S., et al.: ENCASE: an ENsemble CLASsifiEr for ECG classification using expert features and deep neural networks. *Comput. Cardiol. (CinC) Rennes* **2017**, 1–4 (2017)

9. Kachuee, M., Fazeli, S., Sarrafzadeh, M.: ECG heartbeat classification: a deep transferable representation. In: 2018 IEEE International Conference on Healthcare Informatics (ICHI), New York, NY, pp. 443–444 (2018)
10. Kamaleswaran, R., Mahajan, R., Akbilgic, O.: A robust deep convolutional neural network for the classification of abnormal cardiac rhythm using single lead electrocardiograms of variable length. *Physiol. Meas.* **39**(3), 035006 (2018)
11. Kropf, M., et al.: Cardiac anomaly detection based on time and frequency domain features using tree-based classifiers. *Physiol. Meas.* **39**(11), 114001 (2018)
12. McManus, D.D., et al.: A novel application for the detection of an irregular pulse using an iPhone 4S in patients with atrial fibrillation. *Heart Rhythm* **10**(3), 315–319 (2013)
13. Plesinger, F., et al.: Parallel use of a convolutional neural network and bagged tree ensemble for the classification of Holter ECG. *Physiol. Meas.* **39**(9), 094002 (2018)
14. Rizwan, Muhammed, Whitaker, Bradley M., Anderson, David V.: AF detection from ECG recordings using feature selection, sparse coding, and ensemble learning. *Physiol. Meas.* **39**(12), 124007 (2018)
15. Shaffer, F., Ginsberg, J.P.: An overview of heart rate variability metrics and norms. *Front. Public Health* **5**, 258 (2017)
16. Sujadevi, V.G., Soman, K.P., Vinayakumar, R.: Real-time detection of atrial fibrillation from short time single lead ECG traces using recurrent neural networks. *Intelligent Systems Technologies and Applications*, pp. 212–221. Springer, Cham (2018)
17. Teijeiro, T., Garca, C.A., Castro, D., Flix, P.: Arrhythmia classification from the abductive interpretation of short single-lead ECG records. *Comput. Cardiol. (CinC) Rennes* **2017**, 1–4 (2017)
18. Zabihi, M., Rad, A.B., Katsaggelos, A.K., Kiranyaz, S., Narkilahti, S., Gabbouj, M.: Detection of atrial fibrillation in ECG hand-held devices using a random forest classifier. *Comput. Cardiol. (CinC) Rennes* **2017**, 1–4 (2017)

Detection and Classification of Faults in Photovoltaic System Using Random Forest Algorithm



C. Sowthily , S. Senthil Kumar , and M. Brindha 

Abstract Detection of faults in a photovoltaic system is a great challenge, for increasing the solar power generation and improving efficiency. Under low irradiance condition, the power generation gets reduced, at that time the line-line fault remains undetected. The array current, voltage and irradiance are measured and used for detecting and classifying the fault. This paper proposes a new fault classification algorithm based on supervised machine learning technique. The features are extracted for different test conditions under normal and partial shading conditions to get a sample dataset. The features in the database are analysed using Random forest classification algorithms. The classification accuracy of faults is evaluated using the confusion matrix. The experimental and simulated database results are collected from 100 W PV module with a 4×4 array configuration with a successful classification of faults with an accuracy of 99.98%.

Keywords PV array · Rotation forest · Classification algorithm

1 Introduction

Sources of renewable energy play a vital role in protecting us from global warming and greenhouse gas emissions. Solar energy is abundantly present, reliable, economical, less pollution and less-maintenance [1].

C. Sowthily · S. Senthil Kumar

Department of Electrical and Electronics Engineering, National Institute of Technology
Tiruchirappalli, Tiruchirappalli, Tamil Nadu, India
e-mail: sowthilyc@gmail.com

S. Senthil Kumar

e-mail: skumar@nitt.edu

M. Brindha 

Department of Computer Science and Engineering, National Institute of Technology
Tiruchirappalli, Tiruchirappalli, Tamil Nadu, India
e-mail: brindham@nitt.edu

The installation of PV system follows, the standards stated in the U.S. National Electric Code and the Over Current Protection Device, for detecting the short-circuit faults like line-line fault and line-ground faults [2]. The two variables, Gamma and array losses are calculated to differentiate the partial shading conditions and line-line faults [3]. The Kalman filter is used to detect the faults based on temperature, current, and voltage of PV arrays [4]. Analysis of solar power loss is used to detect the faults. Identifying and predicting different types of faults in the PV arrays are the main components to improve the efficiency, reliability and lifespan of the PV arrays [5]. The semi-supervised, supervised and unsupervised, are the types of machine learning methods used to classify the faults in PV modules. The supervised machine learning technique, the Support Vector Machine (SVM) has been commonly used for numerous fault identification problems [6]. In many applications, data collection is difficult and very expensive. In semi-supervised machine learning technique, the SVM can use the unlabelled data with the labelled dataset [7]. Using machine learning techniques, the proper functioning of solar arrays, like local outlier factors and statistic outlier detection rules are analysed [8]. Using the change in array voltage, energy and change in impedance the faults are detected using wavelet packets. [9, 10]. The Bayesian neural network and polynomial regression models [11] are used to identify the moisture on the solar arrays.

The proposed technique is used to classify and predict the faults in PV array. The fault detection and classification are done using the Random forest algorithm. To analyse the proposed method, line-line fault experiments were conducted in the laboratory to get the data samples. In the following sections, the Modelling of PV array is explained in Sect. 2, Sect. 3 gives the faults in Photovoltaic PV array, Sect. 4 provides the fault detection based on Random forest algorithm and Sect. 5 gives the experimental setup and the results.

2 System Description

The PV array consists of a PV module, Power conditioning unit integrated with the MPPT algorithm and protection devices like OCPD and blocking diodes (Fig. 1). The overcurrent protection device acts as a fuse. The blocking diodes are used to prevent the reverse flow of current back to the solar module.

The PV array with 4×4 array configuration connected with modules in series and strings in parallel to each other is contemplated in this paper, to achieve the expected degrees of maximum voltage and power. The datasheet specifications of a single PV module of a thin film are given in Table 1.

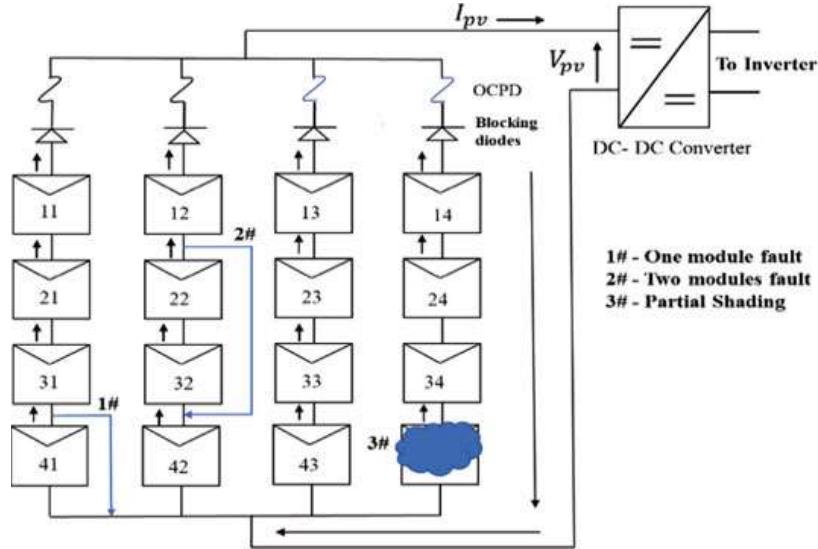


Fig. 1 Typical faults in PV array

Table 1 PV specifications

Parameters	Symbol	Value
Maximum power	P_m	100 W
Open-circuit voltage	V_{oc}	50.12 V
Short-circuit current	I_{sc}	3.4 A
Maximum power voltage	V_{mp}	36.6 V
Maximum power current	I_{mp}	2.9 A

2.1 Analysis of I-V Curves

I-V characteristics is a graphical representation of the solar PV module that gives the relation between voltage and current at existing conditions of temperature and irradiance. The current versus voltage traits for different irradiations at a constant temperature of 25 °C are illustrated in Fig. 2a and the current versus voltage characteristics for different temperatures 25, 35 and 45 °C at constant irradiation are shown in Fig. 2b.

3 Faults in Photovoltaic Array

Faults that occur in a PV system are categorised as open-circuit faults, line-line faults (two modules mismatched), line-ground faults (one module mismatched) and partial shading (temporary fault) as shown in Fig. 1. Line-line faults and partial shading are discussed in this paper. Using conventional protection devices, it is very difficult to detect the line-line faults. Short-circuit fault occurs with a greater number of PV

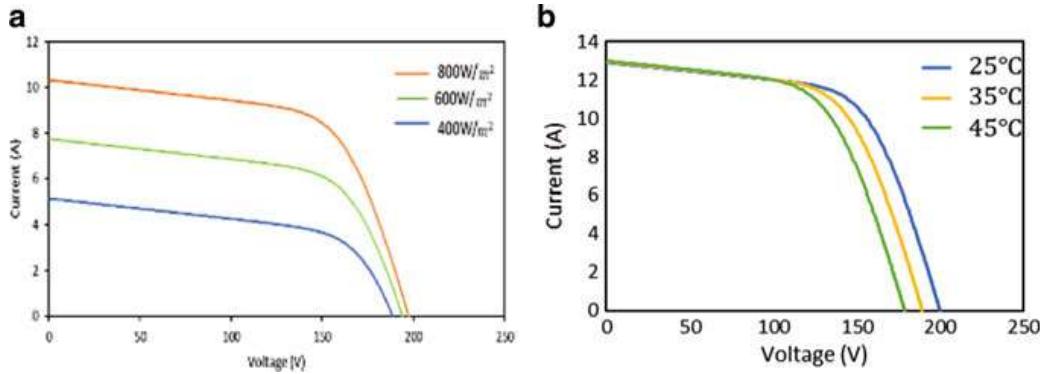


Fig. 2 **a** I-V characteristics for different irradiations of the solar PV module. **b** I-V characteristics of the PV module with different temperature ranges

modules mismatch and this reduces the power generation in the PV array. The failure of insulation cables can cause the line-line fault. The line-line faults are perpetual and partial shading is temporary.

Partial shading is a momentary fault which occurs for some time. When the PV cells are reverse biased, the diodes become active under partial shading condition. It creates a short-circuit in the shaded part when the bypass diodes get activated. It is difficult to differentiate the temporary fault and permanent faults in the PV system, so the classification techniques based on machine learning algorithms is proposed. These techniques are used to improve accuracy and performance.

3.1 One Module Mismatch

In this paper, the line-ground fault is analysed by short-circuiting the modules within the strings. Only 25% of mismatch losses are obtained in this one module mismatch fault. This condition is analysed using the I-V characteristics and P-V characteristics under fault and normal conditions as shown in Fig. 3a, b. The sampling rate for all recorded data is 10 s.

3.2 Two Modules Mismatched

When the fault occurs, the system current drops at the instant and then follows the low value of current. There is not much change in array voltage because the string gets open-circuited and the blocking diode is reverse biased. The two-module mismatch is simulated in MATLAB/Simulink for analysing the effects at STC conditions. The open-circuit faults also can be detected because when two modules mismatch automatically the string gets opened.

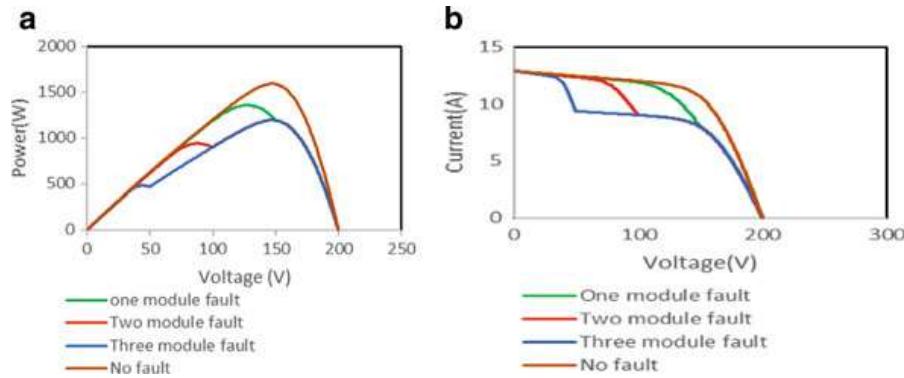


Fig. 3 a, b P-V and I-V characteristics curves line-line fault analysis

3.3 Partial Shading

Shading of the solar panels partially is faced by the rooftop solar plant due to shadows of buildings, trees and dust. Partial shading is classified into two categories based on shading area (a) static shading in which the shaded area of the PV modules is constant and (b) dynamic shading is caused when the shaded area varies with time.

When one of the modules in the PV system is partially shaded, it is observed that the least power loss can occur in a PV system. The partial shading of one module is done by using cardboard. To analyse the I-V and P-V characteristics of partial shading the one module in the designed PV system is shaded for three different cases PSC III-700 W/m², PSC II-400 W/m², PSC I-100 W/m² at Standard Test Conditions is shown in Fig. 4a, b.

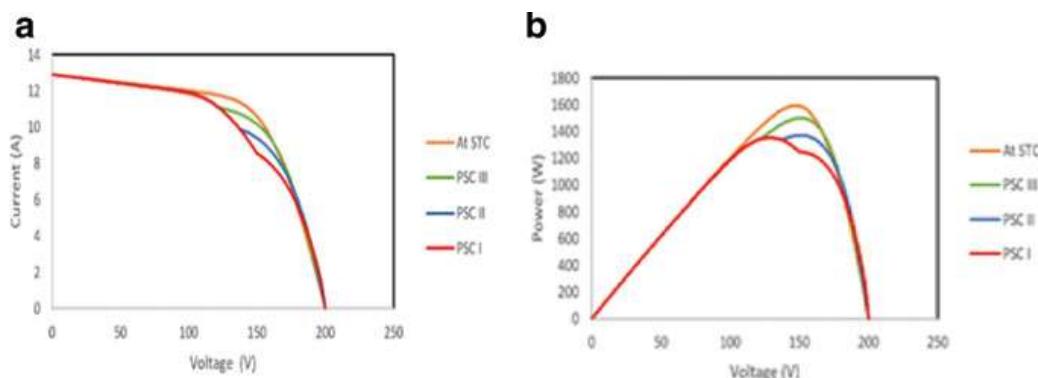


Fig. 4 a, b P-V and I-V characteristic curves of the PV system under partial shading condition (PSC) where one module is shaded at (1) At STC, (2) PSC III-700 W/m², (3) PSC II-400 W/m², and (4) PSC I-100 W/m²

4 Proposed System

4.1 Random Forest-Based Fault Detection Method

Random selection of attributes for each node is based on the decision tree in a random forest algorithm. At last, the classification results are done by voting method, so as to enhance the classification accuracy. Random forest algorithm is a type of supervised machine learning technique in which labelled fault samples are extracted from raw data received from simulations or experiments. To evaluate the performance model, the samples are labelled and divided into testing and training datasets. The testing dataset is used to test the performance of the model on unknown datasets and the training dataset is used to create the model.

The results of the PV array under fault and normal condition are obtained from the modelling of the PV array. The array voltage, current and irradiance values are collected from the data acquisition system. The experimental and simulated data samples are collected to process the algorithm. The datasets are preprocessed and feature extraction is done. The datasets are split into training and testing. The training and testing are done using the Random forest algorithm. The trained and tested models give the classification of faults in the PV system. The flowchart of the proposed method is shown in Fig. 5.

Fig. 5 Flowchart for the proposed method

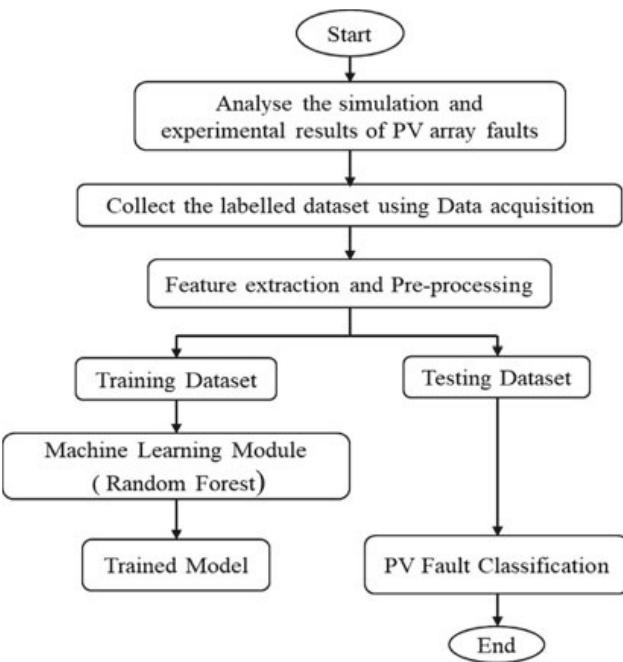




Fig. 6 Laboratory PV system

5 Experimental Results

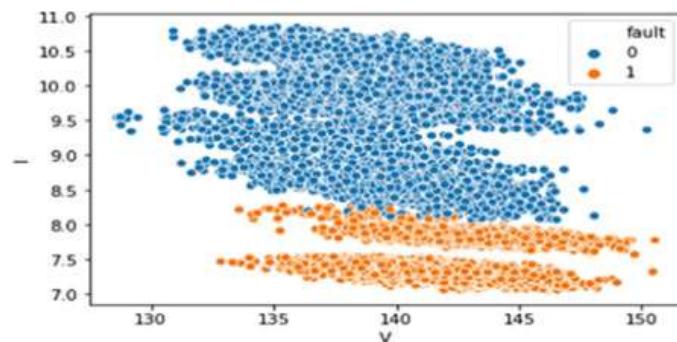
5.1 Laboratory Setup

A laboratory setup with 1.6 KW inverter connected to the grid in the PV system is used to validate the detection and classification of faults. In the PV array setup, the modules are connected in series and each string has four modules and four strings in parallel to each other. The monitoring system is composed of the PV Array, data acquisition card, sensing board, grid-connected inverter and a MATLAB software as shown in Fig. 6. Solar irradiance is measured using the reference module.

5.2 Result Analysis

The experiments were done on a cloudy day and readings were taken by giving different fault conditions. 44,500 experimental data samples for different faults are taken including the normal one. For training and testing the Random forest algorithm, 76% of total sample dataset is used for training and the remaining 24% data samples are taken for testing. The plot shown in Fig. 7 is the voltage versus current plot. This plot can differentiate the presence of a fault and the normal condition without fault.

Fig. 7 Voltage and current plot differentiating fault occurred or not



The experimental data samples are collected and labelled to analyse the line-line fault conditions. The plot shown in Fig. 8 is the voltage versus current and differentiates the line-ground, line-line fault and normal condition.

The plot shown in Fig. 9 gives the voltage versus current plot and differentiates the partial shading condition and normal condition. The readings are collected using the data acquisition and samples are labelled and plotted.

The proposed method of classification of faults in PV array using Random forest algorithm gives the highest accuracy when compared to that of the other classification models like Decision Tree, Naive Bayes and Logistic Regression. The comparison of accuracy in classification models is given in Table 2.

Fig. 8 Voltage versus current plot: one module, two-module mismatch and normal condition

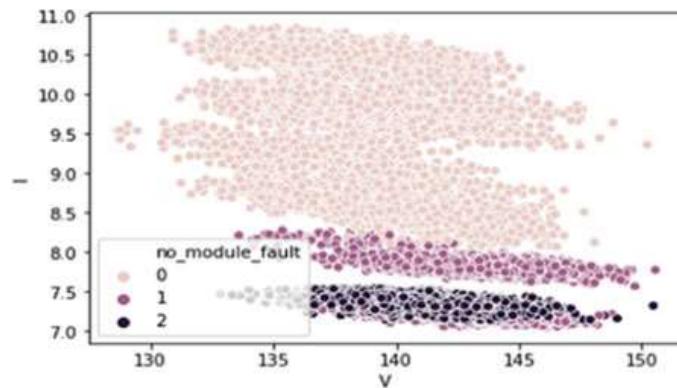


Fig. 9 Voltage versus current plot: Partial shading and normal condition

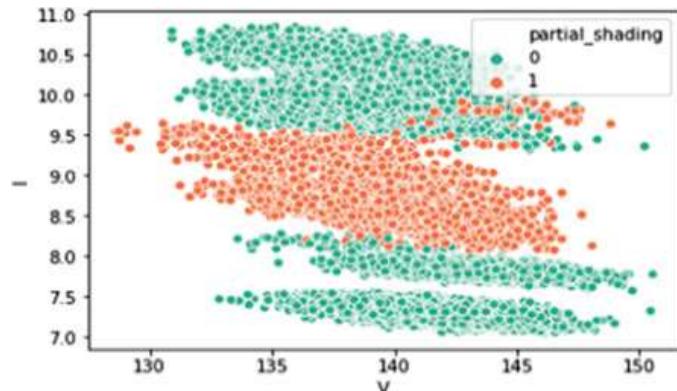


Table 2 Comparison of accuracy in classification models

Classification model	Accuracy (%)
Random forest	99.98
Decision tree	99.96
Naive Bayes	94.04
Logistic regression	91.04

6 Conclusion

This paper proposes a fault classification and detection of PV array faults based on random forest machine learning algorithm with good accuracy. Faults like line-ground fault, line-line fault (two modules mismatched) and partial shading are classified using the algorithm. PV current, voltage and irradiance are used as the features for detecting the faults. A large number of faults are created to verify the proposed work by experimentation and simulation to get the data samples. The proposed algorithm is compared with the accuracies of other classification models like Decision tree, Naive Bayes and Logistic Regression. The detection accuracy of the proposed testing data is 99.98%. The results obtained through both simulation and experiment achieve high accuracy and reliability.

Acknowledgements This research work was funded by the Department of Science and Technology, India under the project “Design and Development of ICT-Enabled Cloud-based mobile application for the self-promotion of products developed by Self Help Groups”.

References

1. U. S. National Electrical Code: Article 690—Solar Photovoltaic Systems (2011)
2. Zhao, Y., Lyons, Jr., R.: Line-Line fault analysis and protection in PV arrays. Tech Topics: Photovoltaic protection Note 2, Mersen Electrical Power, Issue 1 (2011)
3. Hariharan, R., Chakkarapani, M., Saravana Ilango, G., Nagamani, C.: A method to detect photovoltaic array faults and partial shading in PV systems. *IEEE J. Photovoltaics* **6**(5), 1278–1285 (2016)
4. Kang, B.K., Kim, S.T., Bae, S.H., Park, J.W.: Diagnosis of output power lowering in a PV array by using the Kalman-filter algorithm. *IEEE Trans. Energy Convers.* **27**(4), 885–894 (2012)
5. Abdulmawjood, K., Refaat, S.S., Morsi, W.G.: Detection and prediction of faults in photovoltaic arrays: a review. In: IEEE 12th International Conference on Compatibility, Power Electronics and Power Engineering (2018)
6. Nan, S., Sun, L., Chen, B., Lin, Z., Toh, K.: Density-dependent quantized least squares support vector machine for large data sets. *IEEE Trans. Neural Netw. Learn. Syst.* **28**(1), 94–106 (2017)
7. Liu, Y., Xu, Z., Li, C.: Online semi-supervised support vector machine. *Inf. Sci.* **439–440**, 125–141 (2018)
8. Zhao, Y., Balboni, F., Arnaud, T., Mosesian, J., Ball, R., Lehman, B.: Fault experiments in a commercial-scale PV laboratory and fault detection using local outlier factor. In: IEEE 40th Photovoltaic Specialist Conference, pp. 3398–3403 (2014)
9. Massi Pavan, A., Mellit, A., De Pieri, D., Kalogirou, S.A.: A comparison between BNN and regression polynomial methods for the evaluation of the effect of soiling in large scale photovoltaic plants. *Appl. Energy* **108**, 392–401 (2013)
10. Kumar, B.P., Ilango, G.S., Reddy, M.J.B., Chilakapati, N.: Online fault detection and diagnosis in photovoltaic systems using wavelet packets. *IEEE J. Photovoltaics* **8**(1), 257–265 (2018)
11. Chen, L., Li, S., Wang, X.: Quickest fault detection in photovoltaic systems. *IEEE Trans. Smart Grid* **9**(3), 1835–1847 (2018)

Clustering Enhanced Encoder–Decoder Approach to Dimensionality Reduction and Encryption



B. R. Mukesh, Nara Madhumitha, N. Pai Aditya, Srinivas Vivek, and Anand Kumar M.

Abstract Dimensionality reduction refers to reducing the number of attributes that are being considered, by producing a set of principal variables. It can be divided into feature selection and feature extraction. Dimensionality reduction serves as one of the preliminary challenges in storage management and is useful for effective transmission over the Internet. In this paper, we propose a deep learning approach using encoder–decoder networks for effective (almost-lossless) compression and encryption. The neural network essentially encrypts data into an encoded format which can only be decrypted using the corresponding decoders. Clustering is essential to reduce the variation in the dataset to ensure overfit. Using clustering resulted in a net gain of 1% over the standard encoder architecture over three MNIST datasets. The compression ratio achieved was 24.6:1. The usage of image datasets is for visualization only and the proposed pipeline could be applied for textual and visual data as well.

Keywords Deep learning · Encryption · Autoencoders · Dimensionality reduction

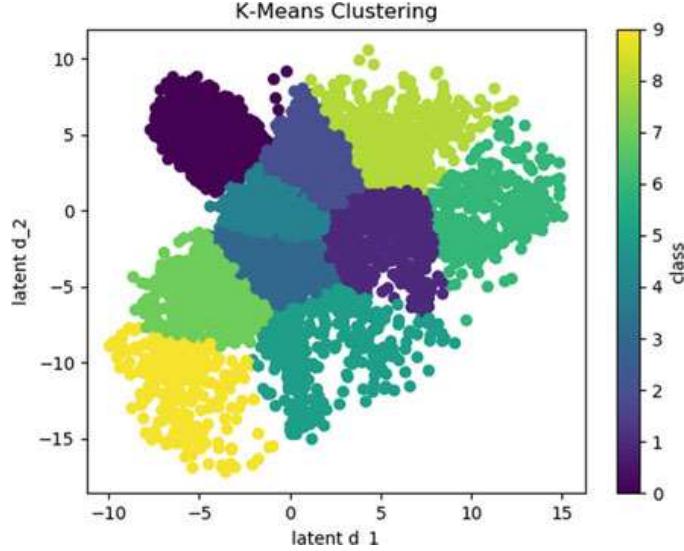
1 Introduction

With the advent of multiple video, image, and audio sharing platforms, we are now increasingly in the need for more storage. However, there is a limit to storage and memory is expensive. A well-known problem known as the curse of dimensionality arises with this. With the increase in the dimensionality of data, there is an exponential increase in the volume of data generated. Hence, the concept of data encoding is of huge interest. To process the high dimensional data generated, a projection to a lower dimensional space is of great help.

B. R. Mukesh (✉) · N. Madhumitha · N. P. Aditya · S. Vivek · Anand Kumar M.
Department of Information Technology,
National Institute of Technology Karnataka, Mangalore, India
e-mail: mukeshbr.br309@gmail.com

Anand Kumar M.
e-mail: m_anandkumar@nitk.edu.in

Fig. 1 K-Means clustering of MNIST dataset



To solve the problem of data storage, popular dimensionality reduction methods have been proposed in the recent years. Singular value decomposition methods such as principal component analysis (PCA) have been used for solving problems associated with high dimensionality [1, 2]. However, the compression ratio is still pretty low, and the data stored is vulnerable to attacks. This might be because PCA only recognizes linear relationships in data (Fig. 1).

In this paper, we propose a clustering boosted autoencoder architecture in order to overfit data to ensure safety and decryption of data. Overfit is necessary to ensure the model does not generalize on other data leading to weakened encryption. Clustering helps the model overfit by presenting it data which is very similar in structure. This results in a better local performance of the model on the dataset that needs to be compressed.

We demonstrate the effectiveness of our proposed system by experiments on three publicly image datasets. But due to the non-convolutional nature of the model, the same proposed pipeline can be used effectively on any high dimensional data such as weather datasets and videos.

2 Literature Review

Hu et al. [3] recognize the potential of autoencoders over other conventional compression methods like principal component analysis (PCA), locally linear embedding (LLE), stochastic neighbor embedding (SNE), etc. They propose the idea of a “folded” autoencoder in which the encoder and decoder models are no longer separate. The folded autoencoder has $(L-1)/2$ hidden layers, while the traditional autoencoder model has L hidden layers.

Sei-ichiro et al. [4] recognize the issue with traditional image compression techniques like JPEG which might introduce compression artifact problems. To overcome this problem, they use a highly dense autoencoder to extract the maximum features out of the image and use a Unet-like architecture to reduce any distortions caused during the compression.

Vincent et al. on deonising autoencoders [5] introduce said autoencoders. The paper recognizes that there are a lot of issues associated with using generative models, especially ones which have multiple layers, as they can develop a lot of intricate relations between attributes, some of which could be unnecessary. Previous works have already shown that using unsupervised learning methods to the data at an initial stage can help overcome this issue of the above-mentioned models.

Huang et al. [6] suggest a method for comparing and assessing the efficacy of dimensionality reduction methods for textual data visualization. The paper answers to the questions of the correct technique of compression which retains the relationships between text in full, the susceptibility of final results to the number of output dimensions and whether a reduction in dimensionality can be achieved by removing unnecessary words from the document vector while still preserving the majority of information.

Moravec et al. [7] compare the performance of several dimensional reduction strategies, including LSI, NMF, SDD, and FastMap. The statistical analysis was focused on ratings and tested on a selection of faces from the Olivetti Research Laboratory. The techniques were evaluated from both the visual impact, the quality of the “eigenfaces” produced and the retrieval results.

Kasun et al. [8] investigate dimension reducing frameworks like extreme learning machine AE (ELM-AE) and the sparse ELM-AE (SELM-AE). This paper recognizes problems in PCA (eigenvectors) and linear AE (unable to portray) data as pieces (e.g. eyes in a human image). The problem with NMF and nonlinear AE is their slow learning speed, and RP only portrays a subspace of the actual information. This paper proposes a dimension-reduction model which, to some degree, describes information as pieces, has a fast learning frequency, and learns the subspace between different scatter groups. It proves its efficiency over training time, discriminative capability, sparsity, and normalized mean square error.

Tan et al. [11] propose a neural network approach to minimize data dimensionality using input training where a single input pattern is not set but changed along with the network parameters to recreate the respective output pattern. By modifying the input a little, a trained network could replicate a dataset with limited distortion; the network inputs will provide the reduced information. A network with three layers which has been input trained will execute all tasks of a flue-layer auto-associative network, basically collecting nonlinear information correlations. In contrast, the simultaneous learning of the network and its inputs is proved to be more effective in dimensionality reduction than an auto-associative network.

Sakurada et al. [12] propose to use encoder–decoder networks with nonlinear dimensionality reduction to detect anomalies. The authors apply dimensionality reduction by using an encoder–decoder network, and compare it with PCA. This paper shows that autoencoders can realize anomalies which linear PCA cannot.

Zhang et al. [13] study semi-supervised dimensionality reduction. In their environment domain, information is of the form of pairwise constraints. It determines whether a pair of instances belong to the same class (must-link constraints) or to different classes (cannot-link constraints). They suggest an SSDR algorithm that stores the internal structure of unlabeled information as well as the must-link and non-link constraints defined in the labeled examples in the projected low-dimensional space. They show that the SSDR algorithm is efficient and has a closed-form solution.

Bi et al. [14] describe a technique for executing variable ranking and sorting using support vector machines (SVMs). The approach builds a sequence of sparse linear SVMs to create linear models. The models generalize well and employ a subset of nonzero weighted variables generated by the linear models to build a final nonlinear model. The approach uses the fact that a linear SVM (no kernels) with L1-norm regularization necessarily executes parameter selection as part of reducing the capacity of the SVM model. The distribution of linear model weights provides a mechanism for ranking and interpreting the effects of variables.

Dash et al. [15] talk about the lack of work done on reducing unsupervised data which do not have any class information. PCA is usually used but it creates new features. With the new features, it is difficult to come up with an idea of the original data. Therefore, the authors focus on the problem of determining and choosing the important original features for unsupervised data. Their method is based on the observation that excluding an unimportant feature from the feature set will not modify the data underneath. They propose an entropy measure for ranking features, and experiments were described to prove that their solution finds the important features.

3 Methodology

To facilitate encryption, the model must not be able to generalize on non-training data. Hence, the goal of the pipeline is overfit on training data as well as learn the rich inherent feature representation of data in the source domain.

The proposed pipeline employs a clustering algorithm to partition the dataset into smaller and more manageable chunks. This is done to ensure that dataset split achieved has all members who are similar to each other in a metric of distance. The proposed denoising autoencoders described in Fig. 2 are then trained on each split of the dataset separately without sharing weights in order to ensure the model will not be able to generalize well on the testing dataset domain.

3.1 K-Means

The goal of the k-means clustering is to divide n observations into k clusters. Each observation belongs to one cluster, the cluster whose mean observation is the nearest, serving as a cluster model for the cluster it belongs to [9].

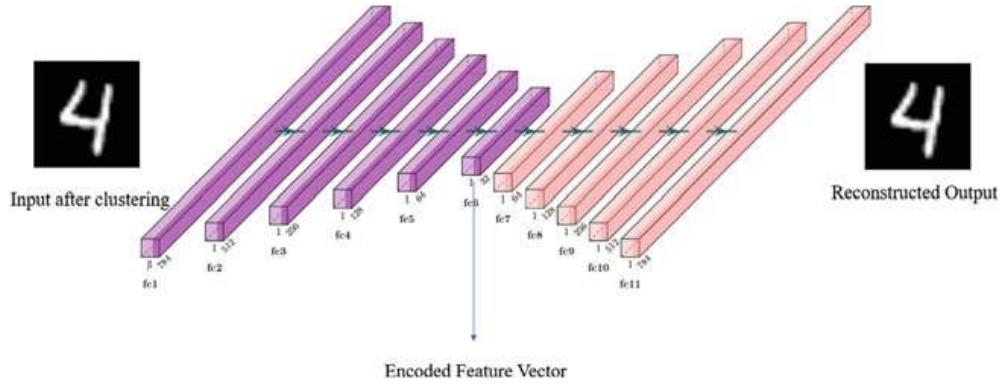


Fig. 2 Architecture of encoder–decoder models

For any given points (x_1, x_2, \dots, x_n) , where each point is a m -dimensional vector, k -means clustering will divide the n observations into k sets $S = \{S_1, S_2, \dots, S_k\}$. Let μ_i is the mean observation of points in S_i . The objective is to minimize the distance between points within each cluster that is described in Eq. 1

$$\mathbf{J} = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|^2 \quad (1)$$

To start with, k points are initialized randomly. These points are also referred to as means. Each point will be assigned to its closest mean (Euclidean L2 distance used in our implementation). The mean's coordinates are modified—the averages of the points categorized in that cluster in that iteration are used as new means. The process is repeated for a given number of iterations till the means do not have to be updated (certain threshold value is reached).

3.2 Autoencoders

Through the excellent representability achieved by neural networks, the autoencoder algorithm as standard dimensionality reduction methods have shown good results. Autoencoders are neural networks used for compression. An autoencoder, not unlike a regular neural network, also has input, hidden and output layers. It uses back-propagation for learning, and the inputs are equal to the target values. Autoencoders have risen in popularity as good dimensionality reduction tools. Their main purpose is to remap inputs to a smaller dimension, hence reducing the size of the original data. A decoder model is used to decompress the compressed data, and bring it into its original form. For video, audio, and series data, convolution layers can be used in the autoencoder model. While PCA performs one huge transformation of data, autoencoders work by gradually transforming data using multiple layers of the neural network. An autoencoder provides a representation of each layer as the output [10].

4 Experimental Validation and Training

The proposed pipeline was tested on three publicly available datasets. The overview of each dataset used is described in Table 1.

The proposed pipeline is designed to take single dimensional vector of varying dimensions as input. The model architecture was varied based on the size of input provided. The performance of the pipeline on a sample set from MNIST is shown in Fig. 3, highlighting the reconstruction potential of the proposed pipeline. The autoencoder uses rectified linear unit (ReLU) as the activation function for all hidden units except the last layer. ReLU is described in Eq. 2

$$R(z) = \begin{cases} z & , z > 0 \\ 0 & , z \leq 0 \end{cases} \quad (2)$$

The proposed pipeline was trained with loss function being binary crossentropy described in Eq. 3.

$$\text{BCE} = -(y \log(p) + (1 - y) \log(1 - p)) \quad (3)$$

using the Adam optimizer with the initial learning rate of 0.01 for 100 iterations per cluster while number of clusters were selected as 10. For training graphs, please refer Table 5. The model was trained on a NVIDIA DGX server having 1× Tesla V-100 cards with 32 GB of VRAM and compute capability of 7.0. The proposed model took 124.3 s per cluster averaging to around 1200 seconds for the entire dataset to be trained.

The standard performance measures used in dimensionality reduction are reconstruction error measured by mean absolute error. The equations used to compute these metrics are given in Table 2.

Table 1 Overview of datasets used for evaluation

Dataset	# Examples	Size of example
MNIST	60000	$28 \times 28 \times 1$
FASHION MNIST	50000	$28 \times 28 \times 1$
CIFAR 10	60000	$32 \times 32 \times 3$



Fig. 3 Reconstruction of images on MNIST dataset with correct decoder weights