Algorithms for Intelligent Systems
Series Editors: Jagdish Chand Bansal · Kusum Deep · Atulya K. Nagar

Prashant Johri
Jitendra Kumar Verma
Sudip Paul *Editors*

Applications of Machine Learning

Springer

Algorithms for Intelligent Systems

This book series publishes research on the analysis and development of algorithms for intelligent systems with their applications to various real world problems. It covers research related to autonomous agents, multi-agent systems, behavioral modeling, reinforcement learning, game theory, mechanism design, machine learning, meta-heuristic search, optimization, planning and scheduling, artificial neural networks, evolutionary computation, swarm intelligence and other algorithms for intelligent systems.

The book series includes recent advancements, modification and applications of the artificial neural networks, evolutionary computation, swarm intelligence, artificial immune systems, fuzzy system, autonomous and multi agent systems, machine learning and other intelligent systems related areas. The material will be beneficial for the graduate students, post-graduate students as well as the researchers who want a broader view of advances in algorithms for intelligent systems. The contents will also be useful to the researchers from other fields who have no knowledge of the power of intelligent systems, e.g. the researchers in the field of bioinformatics, biochemists, mechanical and chemical engineers, economists, musicians and medical practitioners.

The series publishes monographs, edited volumes, advanced textbooks and selected proceedings.

More information about this series at http://www.springer.com/series/16171

Prashant Johri · Jitendra Kumar Verma ·
Sudip Paul
Editors

Applications of Machine Learning

Springer

Editors
Prashant Johri
School of Computer Science
and Engineering
Galgotias University
Greater Noida, Uttar Pradesh, India

Sudip Paul
Department of Biomedical Engineering
North-Eastern Hill University
Shillong, Meghalaya, India

Jitendra Kumar Verma
Department of Computer Science
and Engineering, Amity School of
Engineering and Technology
Amity University Haryana
Gurugram (Manesar), Haryana, India

ISSN 2524-7565 ISSN 2524-7573 (electronic)
Algorithms for Intelligent Systems
ISBN 978-981-15-3356-3 ISBN 978-981-15-3357-0 (eBook)
https://doi.org/10.1007/978-981-15-3357-0

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

The main scope of this special issue is to bring together applications of machine learning in artificial intelligence (human language, heterogeneous and streaming data, unmanned systems, neural information processing, marketing and social sciences, bioinformatics, robotics, etc.) in order to give a wide landscape of techniques that can be successfully applied and also to show how such techniques should be adapted to each particular domain. Statistical learning theory is a framework for machine learning and deep learning drawing from the fields of statistics and functional analysis. Statistical learning theory deals with the problem of finding a predictive function based on data. Statistical learning theory has led to successful applications in fields such as computer vision, speech recognition, health care, business, marketing, bioinformatics, and baseball. From the perspective of statistical learning theory, supervised learning is best understood. Supervised learning involves learning from a training set of data and also from the previous experience. Every point in the training is an input–output pair, where the input maps to an output. The learning problem consists of inferring the function that maps between the input and the output, such that the learned function can be used to predict output from future input. Depending on the type of output, supervised learning problems are either problems of regression or problems of classification.

This book will target undergraduate graduate and postgraduate students, researchers, academicians, policy-makers, various government officials, academicians, technocrats, and industry research professionals who are currently working in the fields of academia research and research industry to improve the life span of general public.

Chapter 1: In the process of characterizing a given population by collecting samples, there are machine learning applications today that provide a wide range of possibilities regarding clustering and data mining topics. These possibilities consist of industrial and scientific application techniques that are adapted to each particular field for the successful achievement of results. As a fundamental element in statistical learning, to understand in a simple way the use of the t-Student statistical distribution, clarifying the concepts of sampling error and convergence criterion based on an iterative process for the calculation of the optimal number of samples.

With this reasoning and inference application of the t-Student distribution, this chapter is intended to find the convenience of a procedure that can be used to discard or not sampling protocols, serving as a starting point till more reliable data can be available.

Chapter 2: In the past few years, there has been an astounding rise in the number of mobile app analytical firms where the developers and innovation teams have been gauging their entire focus onto bringing in more advancement in the field of data analytics and thereby collaborate their own solutions to conduct competitive analysis or market research to understand customers. This can also help the firms in understanding various notable features like ranking, reviews, current price, and how these have key roles in building the monetization of apps under question. The main idea is to track mobile apps and user interaction within the apps. The baseline of this research is to get valuable insights into regarding the entire data extraction process being used for analyzing the app-centric metrics and predict user ratings on the basis of it.

Chapter 3: Data constitute the raw material of any kind of processing strategy and the core elements from the statistical learning viewpoint. The key aspect is the way in which a data processor or algorithm could capture the essence of the data meaning for discriminating its behavior based on its semantic. In this complexity, the data stream processing incorporates an additional concern because it implies that the data are processed as such they are, just using the available hardware resources at the moment in which they arrive. The real-time decision making tends to be a constant functionality in each data stream processing engine, be it through the incremental decision trees, cluster analysis, among others. In this chapter, a systematic literature mapping is carried forward with the aim of mapping strategies that allow discriminating the data meanings and use them for guiding the processing in the data stream context.

Chapter 4: Nowadays, object tracking is a challenge in the field of computer vision; many algorithms are put forward to overcome the issues such as noisy, colored background, and occlusion of object and also illusion which is low and sudden changes in real-time videos; and trackers cannot perform well in real-time environment. An efficient algorithm is needed for effective tracing of an object because of variation in position under various conditions. Here, the MS algorithm is used due to its efficiency and performance. MS-tracking algorithm is used to obtain the position of an object by using a Kalman filter; therefore, the performance of tracking an object for different videos is evaluated, but it does not improve the target localization. The advanced version is developed for tracking and representing the object in different video sequences are MS and CBWH MS using Kalman filter; hence, this method is better than the traditional MS track, which results in the constant tracking of an object throughout the video. It does not get influenced by occlusions and less subjected to background cluttering.

Chapter 5: Humans are known as quick learners and have best ability to transfer acquired knowledge to others. This acquired knowledge can be utilized in solving different types of related tasks. Transfer learning and machine learning are conventionally designed to work independently and to solve specific tasks. Idea of

transfer learning changed its paradigm to utilize acquired knowledge in solving related tasks. Transfer learning is being used widely in industrial applications where the amount of labeled data is limited. This chapter applies transfer learning to the task of named-entity recognition and investigates the effectiveness of transfer learning in stacked bi-LSTM architecture with a fastest word embedding.

Chapter 6: Graphical representation is one of best ways of formal structure representing related data of various categories. Knowledge graphs are having several nodes showing entities and edge connecting theses nodes representing relation between them. Knowledge graph construction involves making logical relationship from unstructured data. In this process, many more similarity indexes to establish relation and to extract logical relationship are investigated. It is a great aid to someone who is not a domain expert as it provides a granular view of the domain. It provisions the context-specific domain knowledge that can be easily mined with minimal human effort.

Chapter 7: Recommendation-based systems help in prediction, performance, and rating of goods and services. This approach supports in making decision in terms of buying products like books, electronic devices, movies, music, etc. by combining the suggestion and rating given by previous users. Recommendation system uses analytic technique in calculating the response and willingness of a user whether he is interested to buy or not. This chapter briefly describes such types of systems based on review, recommendation, suggestion, and rating.

Chapter 8: This chapter gives a glimpse of the use of singular-value decomposition and fuzzy C-means algorithms for Arabic text classification. Al-Jazeera Arabic news and CNN Arabic news datasets are used to measure the effectiveness of this approach in classifying Arabic texts. The experimental results are compared with four supervised classification methods that have been used in previous work on the same datasets we used in this research. These include support vector machine, Naïve Bayes, decision tree, and polynomial networks. The results proved the effectiveness of the proposed approach compared to recent works in Arabic text classification.

Chapter 9: Echo state network (ESN) is a class of neuromorphic computing approach called reservoir computing consisting of a large number of randomly interconnected neurons. Only the reservoir-to-output read-out mappings are variable and are modified during the process of training. ESN functions as a densely interconnected recurrent neural network and is suitable for temporal prediction tasks, but with significantly reduced training complexity. In this chapter, we propose an equalizer based on ESN and evaluate its performance over nonlinear dispersive channels.

Chapter 10: Melody extraction plays a significant role in the field of music information retrieval (MIR). Over the past decade, it has emerged as one of the active research problems in MIR domain. Nowadays, the music providers have to facilitate searching of music based on their contents or recommend music based on user's interest having identical contents. Melody extraction is mandatory to fulfill these user-interest-driven searching and recommendation. The primary objective of melody extraction is to achieve a frequency series that corresponds to the pitch

of the dominant melody in an audio sample. Numerous approaches have been introduced for melody extraction, mainly from polyphonic music. Melody extraction approaches can be classified into three categories on the basis of the key concepts used for the algorithmic design, namely salience-based, source separation-based, and data-driven approaches. Salience-based approaches have used the salience function of pitch candidate, which is nothing but a pitch salience based on time–frequency representation. Source separation-based approaches distinguish the melody source from the polyphonic music. Data-driven approaches are new and recent development for melody extraction, which basically classifies the peak of the pitches of polyphonic music. In this chapter, these approaches are discussed broadly along with the inherent challenges that remain to be solved. In addition, the commonly used datasets and the performance measures that are used to evaluate the performance of melody extraction approaches are also discussed. The MIR applications and the music analysis, where melody extraction plays a vital role, are briefly explained.

Chapter 11: The entropy generation principle of thermodynamics and statistical methodology has been used in the present work for the evaluation of performance parameters of gas turbine-steam turbine (GT-ST)-based combined power generation system and assessment of major losses in components. The mathematical model is carried out for the quantitative analysis of exergy estimation and efficiency of parts in accurate manner. The statistical analysis is also agreed for examining the variations in performance of plant at different operating conditions.

Chapter 13: In this chapter, a method of an improved training pattern in back-propagation neural networks using Holt-Winters' seasonal method and gradient boosting model (NHGB) is given in detail. It removes the errors that cause disabilities in the hidden layers of BPNN and further improves the predictive performance. It increases the weights and decays the error using Holt-Winters' seasonal method and gradient boosting model, which reduces longer convergence time. The NHGB method is compared with other existing methods against average initial error, root mean square error, accuracy, sensitivity, and specificity metrics.

Chapter 14: Machine learning (ML) is increasingly being used in the healthcare domain for time-series predictions. However, the development of multiheaded ML architectures for multivariate time-series predictions has been less explored in the literature concerning healthcare domain. Multiheaded neural network architectures work on the idea that each independent variable (input series) can be handled by a separate neural network model (head), and the output of each of these models (heads) can be combined before a prediction is made about a dependent variable. Here, its application and the utility of developing multiheaded ML architectures and their ensembles for prediction of patient-related expenditures in the healthcare domain have been described.

Chapter 15: This chapter presents framework which makes use of product and process metrics to predict faulty modules in agile software development environment. The model makes use of soft computing methods, like Mandani and Takagi–Sugeno-style fuzzy inference system, artificial neural network, and adaptive neuro-fuzzy inference system for building fault prediction models. For achieving

the goal of the study, several experiments are performed on the "Ant data set project version" of the "PROMISE" data repository. Additionally, to assess the outcomes of the proposed prediction model, evaluation criteria based upon receiver operating characteristics with the area under curve (AUC (ROC)) is applied. The application of the proposed model can help the developers not only in the design phase but also in testing and maintenance phases. It also reduces the time effort in code and review process.

Chapter 16: The seas and the oceans have existed since the origination of the earth and play a very significant role in balancing the ecosystem. Coincidentally, they are the primary and early indicators of global imbalances that maliciously creep in the nature. Sea surface temperature is the foremost parameter that is observed to have a direct or indirect relationship to such variations. There is a connection among the sea surface temperature with various bio-parameters that further affect not only the aquatic flora and fauna but the entire global biome, at times causing drastic changes.

Chapter 17: In this chapter, a regression model for estimation of the multimedia device performance has been established. The model takes into consideration peculiarities of the processing pipeline organized on the device. In this case, system works in the single-threaded mode, and therefore, all stages of processing are performed sequentially. We define all the stages for data processing within embedded system based on the system-on-chip circuit. According to the analysis of the process, we identified that the conventional unit of display area can be selected as independent variable in the model. The main peculiarity of the current investigation is that device processes multimedia data using its own resources without external assist from other devices.

Chapter 18: Time-series data generation is a standing problem in nearly every field, such as science, business, medicine, industry, or even entertainment. As a result, there is a growing demand for analyzing these data efficiently for gauging out useful information. The time-series data have intrinsic features like noise, multi-dimensional, and large volume. When we talk about data mining, it requires a wide spectrum searching for similar patterns, such as query by content, clustering, or classification. These data mining tasks can take great help from a good and robust time-series representations. It helps in the reduction of dimensions, noise adaptation, and also in achieving key aspect, effectiveness, and efficiency of data processing. This chapter aims to review the basic as well as recent approaches for representations along with dimensionality reduction for time-series data.

Chapter 19: In the real world, it is always the case that either the process or measurement noise is nonlinear. Extended Kalman filters are proposed for such scenarios, but they are only suboptimal solutions as they linearize the nonlinear data using methods like Taylor series expansion. In any case, in reality, one experiences an enormous number of situations where either the procedure or estimation model (or both) is nonlinear. Using recursive Gaussian inference, identifying noisy features and dynamically calculating the relative positions between them can be done. By applying more variations to the prior distribution calculations method, noisy sensor data can also be processed into small local maps and forming them into a

consistent global map. The Kalman filter-based or pure recursive Bayesian approach-based mapping algorithms are capable of extracting very few features from the sensor data which are not sufficient for effective simultaneous localization and mapping in noisy and multistoried environments.

Chapter 20: Extracting patterns from a complex real-life dataset, and drawing inferences, thereafter, is becoming part and parcel in various areas of research during the past two decades. Unsupervised machine learning is a type of self-organized learning, which helps to find previously unknown patterns in the real-life dataset without preexisting labels. However, analyzing, understating, and identifying the typical sequential patterns of events in complex event history data, and the ability to utilize this retrieved knowledge, create a significant impact on many aspects of individual life course. Introduction of the sequence analysis for analyzing the adulthood and family formation sequencing may give better understanding the evolution, features, and typologies of various complex life-course processes. Not only the ordering of sequences, grouping these existing sequential patterns into clusters is also challenging in life-course analysis. Modern tools of unsupervised machine learning are an appropriate choice to analyze the sequences of important life-course trajectories and in particular, when there is unobserved heterogeneity present in the data.

Chapter 21: A very simple approach of supervised learning for predicting a quantitative response is the regression model in linear form with Gaussian errors. Considering regression situations where the distribution of the stochastic component not only deviates from normality but also from symmetry and the mesocratic property. Introducing and developing quantile-based predictive models called herein as parametric regression quantile (PRQ) models with quantile-based symmetric and skewed as well as nonmesocratic additive errors.

Chapter 22: The "automatic text document summarization" (ATDS) is a multifaceted research field of machine learning, text analytics, data mining, linguistic, and natural language processing. This can be used in the fields of news summarization, e-governance, information retrieval, search engine optimization, etc. This chapter proposes a linear model for ATS, and this works to finding an optimal feature vector to decide the importance of different features. This is done by logistic regression and random forest.

Chapter 23: Gender classification is an emerging area of research for the accomplishment of efficient interaction between human and machine using speech files. Numerous ways have been proposed for the gender classification in the past. Speech recognition serves as a prime approach for the identification of the source. Other means for the gender classification include gait of person, lips' shape, facial recognition, and iris code. In this chapter, the gender has been classified for machine learning-based systems using speech files. These systems may be deployed for the critical investigation areas like crime scene. There are various challenges in this field of speech recognition like determining the multilingual segments added in the speech stream and the gender of the speaker. To resolve these problems and to identify the gender of the speaker, many different algorithms are used such as frequency estimation, hidden Markov models, Gaussian mixture models, pattern

matching algorithm, neural networks, matrix representation, vector quantization, and decision trees.

Chapter 24: Big data is a challenge for enterprise, and the prime challenge is how to accumulate big data? In the past and some organizations currently use the enterprise data warehouse to store big data. As technological advancement enterprise data warehouse is not suitable for data storage for current market demand, enterprise data warehouse works on the concept of schema-on-write architecture; to get data in the data warehouse, an ETL process is required. With this architecture, the organizations design a data model and prepare an analytic plan before loading data. But big data analytics want data storage who works on schema-on-read concept in which data are stored in raw format as data generated or there is no need to prepare an analytic plan before loading data. To fulfill market demand, a new data repository system evolved for big data storage who can store all kinds of data at one place known as data lake. Awareness of the data lake is to enhance an enterprise data warehouse environment. The data lake is the data-landing area for the raw data from many and always increases the number of data sources in the organization. Data from data lake can be distributed and transformed into the downstream system as they required.

Greater Noida, India — Prashant Johri
Greater Noida, India — Jitendra Kumar Verma
Shillong, India — Sudip Paul

Editorial Advisory Board

Technical Program Committee

Contents

About the Editors

Dr. Prashant Johri is a Professor at the School of Computing Science & Engineering, Galgotias University, Greater Noida, India. He received his MCA from Aligarh Muslim University and Ph.D. in Computer Science from Jiwaji University, Gwalior, India. He has also worked as a Professor and Director (MCA), Noida Institute of Engineering and Technology, (NIET). His research interests include big data, data analytics, data retrieval and predictive analytics, information security, privacy protection, big data open platforms, etc. He is actively publishing in these areas.

Dr. Jitendra Kumar Verma is Assistant Professor (Grade III) of Computer Science & Engineering at Amity School of Engineering & Technology, Amity University Haryana, Gurugram (Manesar), India. He received the degree of Ph.D. from Jawaharlal Nehru University (JNU), New Delhi, India in 2017, degree of M. Tech from JNU in 2013 and degree of B.Tech in Computer Science & Engineering from Kamla Nehru Institute of Technology (KNIT), Sultanpur, Uttar Pradesh, India in 2008. Dr. Verma is awardee of prestigious DAAD "A new Passage to India" Fellowship (2015-16) funded by Federal Ministry of Education and Research - BMBF, Germany and German Academic Exchange Service (DAAD). He worked at JULIUS-MAXIMILIAN UNIVERSITY OF WÜRZBURG, GERMANY (mother of 14 Nobel Laureate) as a Visiting Research Scholar. Dr. Verma is member of several technical societies e.g. IEEE, IEEE IAS, and ACM. Over his short career, he published several research papers in proceedings of various international conferences and peer-reviewed International Journals of repute. He also contributed numerous book chapters to the several books published with publishers of high international repute. Apart from scholarly contribution towards scientific community, he organized several Conferences/Workshops/Seminars at the national and international levels. He voluntarily served as reviewer for various International Journals, conferences, and workshops. He also served as Guest Editor and Editorial

Board Member of numerous international journals. His research interest includes cloud computing, Mobile cloud, Machine learning, AR & VR, Soft computing, Fuzzy systems, Health care, Pattern recognition, Bio-inspired phenomena, and advanced optimization model & computation.

Dr. Sudip Paul is an Assistant Professor at the Department of Biomedical Engineering, School of Technology, North-Eastern Hill University (NEHU), Shillong, India. He received his Ph.D. from the Indian Institute of Technology (Banaras Hindu University), Varanasi, with a specialization in Electrophysiology and Brain Signal Analysis. He was selected as a Postdoc Fellow in 2017–18 under the Biotechnology Overseas Associateship for scientists working in the Northeastern States of India, supported by the Department of Biotechnology, Government of India. Dr. Sudip has published more than 90 international journal and conference papers and has filed four patents. Recently, he completed three book projects and is currently serving as Editor for a further two. Dr. Sudip is a member of numerous societies and professional bodies, e.g. the APSN, ISN, IBRO, SNCI, SfN, and IEEE. He received First Prize in the Sushruta Innovation Award 2011, sponsored by the Department of Science and Technology, Government of India, and various other awards, including a World Federation of Neurology (WFN) Travelling Fellowship, Young Investigator Award, and IBRO and ISN Travel Awards. Dr. Sudip has also served as an editorial board member for a variety of international journals.

Chapter 1
Statistical Learning Process for the Reduction of Sample Collection Assuring a Desired Level of Confidence

Vicente González-Prida, Jesús Zamora, Adolfo Crespo, and Pedro Moreu

1 Introduction

Considering a process of characterizing a population when there is no possible to control the origin of the sample or to skew a part of them, accepting some others, it is usual to assess this characterization by averages. On the contrary, sometimes simplifications are applied without estimating the distortions introduced into the measurements. In general terms, a score is made with diverse values of the sampling error ("*e*") till the obtention of a reasonable number of samples *N*. In any case, the application of a protocol for characterizing the population should be made in a rigorous way. On other occasions, instead of defining the sampling error "*e*" as such, other convergence criteria are used as algorithms for calculating the number of samples, which seems to imply a different concept established in the previous definition. This clarification is contained in the following paragraphs.

In short, the quantitative assessment of uncertainty [1] (in this case, the characterization of an entire population based on a cluster of samples) implies in practice the following elements:

- An existing system that lies as the core of the analysis. It will be the population to characterize.
- A series of features on which there are certain interests or circumstances that prompted the uncertainty evaluation in the process of decision-making, problem-solving and/or planning.
- A variety of sources of uncertainty that affect the system (population) and that do not allow having sufficiently contrasted information on the characteristic that we are interested in defining the population.

V. González-Prida (✉) · J. Zamora
UNED, Madrid, Spain
e-mail: vgonzalezprida@us.es

V. González-Prida · A. Crespo · P. Moreu
University of Seville, Seville, Spain

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_1

For the above elements, probabilistic models are considered [2], seeking to inference from a cluster of samples, characteristics of the total population. Nevertheless, data mining and the latter reasoning and inference could always introduce uncertainty about possible inaccuracies in the models used for calculation [3].

This chapter is intended to provide a procedure that can serve as a starting point when there is unknown if the available data are reliable or representative for the entire population. The purpose here is to identify a number of samples, enough for the determination of a confidence level for the a.m. preliminary measure. Once studied the results of the preliminary characterization and taking into account the error of measurements, a further protocol can be applied considering the determined cluster of samples. Nevertheless, it is also suggested to explore the possibilities of using modeling techniques when there are no reliable previous data. An example of an application could be the assessment of organic matter composition by the sampling of a recycling plant.

Regarding the sources of uncertainty, they may affect the existing system by several means [1]. One example (among others) is through uncertain values of model inputs that may cause errors in the model, resulting in the formulation of an uncertain (or incomplete) structure model [3]. Regarding the outputs, they may be related to multiple needs or interests, as could be, for example, the economic optimization of certain processes. In either case, the decision-making process will depend significantly on the involved interests. In the current case, the uncertainty study is designed to obtain values of a specific characteristic with a determined confidence level, within a given precision error in the calculation. The percentages obtained for specific characteristics will be the ones that will condition the process of the subsequent decision-making [4].

2 Sampling Protocol Proposal

2.1 Mathematical Base

Usually, for the purposes of confidential estimations, it is assumed that random variables follow a normal distribution due to:

1. Many natural phenomena follow normal distribution, such as
   - The kinematics (velocity, energy, momentum) of gases in thermodynamic equilibrium follows a normal distribution (Maxwell–Boltzmann equations).
   - In signal processing, the impulse function (Dirac delta) is a particular case of normal distribution.
2. Many random phenomena that follow other distributions can be approached by a normal distribution with great precision. In general terms, and due to the central limit theorem, it is possible to assume that when the variable understudy is the

result of the sum of a sufficiently large number of effects that act as independently, this variable follows a normal distribution.

3. The "good behavior" of the normal distribution (in order to find confidence intervals easily for expectancy and variance) makes convenient, in the first instance, to consider that random variables are normal and, in any case, to verify whether or not they move away from that distribution.

Therefore, the confidence intervals to be studied would be those associated with the parameters of a normal random variable, which greatly simplifies the obtaining of confidence intervals [5]. For that purpose, a series of distribution based on the normal distribution can be introduced. The sampling protocol proposal will be used as the *t*-student distribution. This distribution can be defined as follows: Let $Z \sim N(0; 1)$ and $J \sim \chi^2$ be independent. The random variable $T_n = z/\sqrt{(J/n)}$ follows a *t*-Student distribution with parameter n and is denoted as $T_n \sim t_n$. For a random variable $T_n \sim t_n$, it will be denoted as t_n, the value that verifies that $P\left[Tn \leq t_{n,\alpha}\right] = 1 - \alpha$. Therefore, $P\left[Tn > t_{n,\alpha}\right] = \alpha$. The distribution values shall be calculated from the normal distribution values and the χ^2 values. Thus, they are presented in tables. Nevertheless, as far as the density function is symmetrical respecting its origin, one has that $P[Tn \leq -t] = 1 - P[Tn < t]$, hence it is possible to reduce the table dimension.

As commented, the statistical distribution to be used will be the *t*-Student distribution. It is convenient to clarify the concepts of sampling error and convergence criteria in the iterative process of calculating the number of samples [5]. In that sense, the parameter "*e*" in the expression for calculating the sample size and the convergence criterion for the iterative process are defined as follows:

$$e = X - \mu \tag{1}$$

where

- e: Maximal desired error
- X: Sample average
- μ: Population average.

Usually, it is recommended that the iterative process continues until the value of the obtained N in the next iteration differs from the previous iteration no more than a certain percentage. When this situation occurs, the process may be interrupted, taking as the final value thereof, the last value obtained for N. The significance of this approach is that by taking a value of N slightly greater than the one that could have been, in case of continuing the adjustment, the obtained confidence level would be raised slightly higher. The tables of *t*-Student used here correspond to those provided by the ATSM standard [6]. Therefore, the values shown in the first column (N) are increased in one unit respecting the original *t*-Student table [7]. This is due to the fact that, in the original table, this column indicates the "number of freedom degrees" (ν), which, in the case of estimating a single statistic (in our case, the populational average μ):

$$\nu = N - 1 \tag{2}$$

That is, the table to be used indicates directly the size of the sample by $N = \nu + 1$.

Similarly, confidence levels shown in ATSM table as 90% ($t_{0.90}$) and 95% ($t_{0.95}$) correspond to 95% ($t_{0.95}$) and 97.5% ($t_{0.975}$) values, respectively, from the original *t*-Student table [6]. The reason is because, in the original table, 95% ($t_{0.95}$) and 97.5% ($t_{0.975}$), respectively, represent the probability that $-\infty \leq t \leq 0.95$ (which may be understood that the probability of that "*t*" outside the range considered is 10% = 0.10) and $-\infty \leq t \leq 0.975$ (which may be understood that the probability of that "*t*" outside the range considered is 5% = 0.05), whereas the standard 90% ($t_{0.90}$) and 95% ($t_{0.95}$) represent probabilities that $-0.95 \leq t \leq 0.95$ (i.e., the probability that "*t*" outside the range considered is 10% = 0.10) and $-0.975 \leq t \leq 0.975$ (i.e., the probability that "*t*" outside the range considered is 5% = 0.05).

In other words, if *C* is the confidence level that corresponds to t_c, then

$$C_{\text{original table}} = C + (1 - C)/2 \tag{3}$$

For example, if a confidence level of 90% (=*C*) is desired, the value of *t*-Student should be searched in the original table in the column:

$$C_{\text{original table}} = 0.90 + (1-0.90)/2 = 0.95 \tag{4}$$

In the case that the desired level of confidence was $C = 95$, in the original table it would correspond to:

$$C_{\text{original table}} = 0.95 + (1-0.95)/2 = 0.975 \tag{5}$$

The main purpose of this protocol is to determine the mean values (average) of certain characteristics from a specific given population, with a desired level of confidence. Determining the sample size to be considered is a mean to get the a.m. purpose. As for the initial population data, needed for the beginning of the sample size calculation, it is assumed that the mean *X* and the standard deviation *s* are known through bibliographic studies, other preliminary studies, or by characteristic tables that can be provided by standards related to each corresponding case study [6].

In the order of the calculated sample size *N* allows to obtain a populational average with an error "*e*" relative to the initial starting value, and considering the established confidence level, it is necessary that the initial data come from a population similar to the one understudy. Otherwise, it cannot be assured that the confidence level results in the one established [2]. That is, the confidence level, in the latter case, would be completely unknown, so the procedure would be completely useless [3]. Therefore, when reliable initial data are available, it will be possible to apply the calculation procedures of sample size *N* usually established. However, when such data are not available, it is necessary to devise a new initial protocol.

Table 1 gives a summary of all levels of confidence.

Table 1 Level of confidence

	Level of confidence										
	10.00%	20.00%	30.00%	50.00%	70.00%	80.00%	90.00%	95.00%	98.00%	99.00%	99.90%
n \ α	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.02	0.01	0.001
1	0.1584	0.3249	0.5095	1.0000	1.9626	3.0777	6.3137	12.7062	31.8210	63.6559	636.5776
2	0.1421	0.2887	0.4447	0.8165	1.3862	1.8856	2.9200	4.3027	6.9645	9.9250	31.5998
3	0.1366	0.2767	0.4242	0.7649	1.2498	1.6377	2.3534	3.1824	4.5407	5.8408	12.9244
4	0.1338	0.2707	0.4142	0.7407	1.1896	1.5332	2.1318	2.7765	3.7469	4.6041	8.6101
5	0.1322	0.2672	0.4082	0.7267	1.1558	1.4759	2.0150	2.5706	3.3649	4.0321	6.8685
6	0.1311	0.2648	0.4043	0.7176	1.1342	1.4398	1.9432	2.4469	3.1427	3.7074	5.9587
7	0.1303	0.2632	0.4015	0.7111	1.1192	1.4149	1.8946	2.3646	2.9979	3.4995	5.4081
8	0.1297	0.2619	0.3995	0.7064	1.1081	1.3968	1.8595	2.3060	2.8965	3.3554	5.0414
9	0.1293	0.2610	0.3979	0.7027	1.0997	1.3830	1.8331	2.2622	2.8214	3.2498	4.7809
10	0.1289	0.2602	0.3966	0.6998	1.0931	1.3722	1.8125	2.2281	2.7638	3.1693	4.5868
11	0.1286	0.2596	0.3956	0.6974	1.0877	1.3634	1.7959	2.2010	2.7181	3.1058	4.4369
12	0.1283	0.2590	0.3947	0.6955	1.0832	1.3562	1.7823	2.1788	2.6810	3.0545	4.3178
13	0.1281	0.2586	0.3940	0.6938	1.0795	1.3502	1.7709	2.1604	2.6503	3.0123	4.2209
14	0.1280	0.2582	0.3933	0.6924	1.0763	1.3450	1.7613	2.1448	2.6245	2.9768	4.1403
15	0.1278	0.2579	0.3928	0.6912	1.0735	1.3406	1.7531	2.1315	2.6025	2.9467	4.0728
16	0.1277	0.2576	0.3923	0.6901	1.0711	1.3368	1.7459	2.1199	2.5835	2.9208	4.0149
17	0.1276	0.2573	0.3919	0.6892	1.0690	1.3334	1.7396	2.1098	2.5669	2.8982	3.9651
18	0.1274	0.2571	0.3915	0.6884	1.0672	1.3304	1.7341	2.1009	2.5524	2.8784	3.9217

(continued)

Table 1 (continued)

	Level of confidence										
	10.00%	20.00%	30.00%	50.00%	70.00%	80.00%	90.00%	95.00%	98.00%	99.00%	99.90%
n \ α	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.02	0.01	0.001
19	0.1274	0.2569	0.3912	0.6876	1.0655	1.3277	1.7291	2.0930	2.5395	2.8609	3.8833
20	0.1273	0.2567	0.3909	0.6870	1.0640	1.3253	1.7247	2.0860	2.5280	2.8453	3.8496
21	0.1272	0.2566	0.3906	0.6864	1.0627	1.3232	1.7207	2.0796	2.5176	2.8314	3.8193
22	0.1271	0.2564	0.3904	0.6858	1.0614	1.3212	1.7171	2.0739	2.5083	2.8188	3.7922
23	0.1271	0.2563	0.3902	0.6853	1.0603	1.3195	1.7139	2.0687	2.4999	2.8073	3.7676
24	0.1270	0.2562	0.3900	0.6848	1.0593	1.3178	1.7109	2.0639	2.4922	2.7970	3.7454
25	0.1269	0.2561	0.3898	0.6844	1.0584	1.3163	1.7081	2.0595	2.4851	2.7874	3.7251
26	0.1269	0.2560	0.3896	0.6840	1.0575	1.3150	1.7056	2.0555	2.4786	2.7787	3.7067
27	0.1268	0.2559	0.3894	0.6837	1.0567	1.3137	1.7033	2.0518	2.4727	2.7707	3.6895
28	0.1268	0.2558	0.3893	0.6834	1.0560	1.3125	1.7011	2.0484	2.4671	2.7633	3.6739
29	0.1268	0.2557	0.3892	0.6830	1.0553	1.3114	1.6991	2.0452	2.4620	2.7564	3.6595
30	0.1267	0.2556	0.3890	0.6828	1.0547	1.3104	1.6973	2.0423	2.4573	2.7500	3.6460
40	0.1265	0.2550	0.3881	0.6807	1.0500	1.3031	1.6839	2.0211	2.4233	2.7045	3.5510
80	0.1261	0.2542	0.3867	0.6776	1.0432	1.2922	1.6641	1.9901	2.3739	2.6387	3.4164
120	0.1259	0.2539	0.3862	0.6765	1.0409	1.2886	1.6576	1.9799	2.3578	2.6174	3.3734
∞	0.126	0.253	0.385	0.674	1.036	1.282	1.645	1.96	2.326	2.576	3.291

2.2 New Sampling Protocol

In order to solve the problem raised in the previous section, a new procedure is developed to obtain the number of samples that assures a desired level of confidence, or in order to know the new confidence level if a smaller number of samples are performed.

The steps to follow in this proposed basic procedure would be as follows:

- **Step 0**. Start with $N = 4$ samples
- **Step 1**. Perform the analysis, obtaining the average and standard deviation values.
- **Step 2**. Considering the desired sampling error (e), calculate the value "t" by the algebraic expression for this statistic:

$$t = ((X - \mu)/s)(N - 1)1/2 = (e \cdot X/s)(\mathrm{N}-1) \cdot (1/2) \tag{6}$$

- **Step 3**. In the original t-Student table, search the corresponding (or closest) confidence level to the value of t calculated in the previous step is searched, for $\nu = \mathrm{N} - 1 = 3$.
- **Step 4**. Due to the reasons given in the previous section, consider the following expression in order to determine the confidence level that corresponds to the experiment:

$$C = 1 + (C_{\text{Table}} - 1) \cdot 2 \tag{7}$$

  For example, if $C_{\text{Table}} = 0.6$ was obtained, then $C = 1 + (0.6 - 1) * 2 = 0.2$ (i.e., a 20% of confidence level).
- **Step 5**. If the confidence level obtained in the previous step was not enough (in the example shown, 20%), proceed to take a new sample. In this case, a fifth sample should be performed.
- **Continuation or termination**:
  - If the obtained level of confidence is enough, the process will be considered completed.
  - If the level of confidence was not sufficient and a new sample has been taken, it will return to step 1.

Overall, no problems are observed in the viability of this procedure, while the population does not vary. In addition, the above protocol could accelerate the calculation of the number of samples to be taken, combining it with other procedures which may provide average values that already approach a priori to the reality of the searched characteristic [5].

3 Study Cases

3.1 Simulation 1

In order to verify the validity of the new method, a simulation is carried out using data related to determine sampling of organic matter [8]. In particular, 26 values have been taken and are shown in Table 2.

Starting with the following formulas:

$$n = \left[(t \cdot s)/\left(e \cdot X'\right)\right]^2 \tag{8}$$

$$t = e \cdot X'/[s/(n(1/2))] \tag{9}$$

The 26 known values from the samples are going to be applied in the new calculation method, following the same order and considering a precision error of 6%. By this way, taking the values of organic matter in the first four samples, the average and the standard deviation are obtained as follows:

$$n = 4$$
$$X' = 45.36\%$$
$$s = 0.06641$$

With the above values and considering an error $e = 6\%$, it is obtained for the t-student:

$$t = 0.819682$$

With this and entering in the table, it implies the following level of confidence (see Table 3), that is, a confidence level of 0.575256 (or 57.526%).

Identically, we continue with a new iteration, that is, including a sixth sample in the calculation as follows:

$$n = 5$$
$$X' = 47.25\%$$
$$s = 0.07140$$

With the above values and considering an error $e = 6\%$, it is obtained for the t-student:

$$t = 0.887936$$

Table 2 Samples values

n	1	2	3	4	5	6	7	8	9	10	11	12	13
OM (%)	46.76	54.08	39.07	41.52	54.82	47.33	50.09	53.10	39.76	43.55	36.32	48.60	45.52
n	14	15	16	17	18	19	20	21	22	23	24	25	26
OM (%)	41.46	50.47	50.04	45.89	38.12	47.03	41.47	42.88	45.08	48.69	38.86	37.52	60.72

Table 3 Confidence level for $t = 0.819682$

	Level of Confidence							
α / n	10.00%	20.00%	30.00%	50.00%	70.00%	80.00%	90.00%	95.00%
	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05
1	0.1584	0.3249	0.5095	1.0000	1.9626	3.0777	6.3137	12.7062
2	0.1421	0.2887	0.4447	0.8165	1.3862	1.8856	2.9200	4.3027
3	0.1366	0.2767	0.4242	0.7649	1.2498	1.6377	2.3534	3.1824
4	0.1338	0.2707	0.4142	0.7407	1.1896	1.5332	2.1318	2.7765

Table 4 Confidence level for $t = 0.887936$

	Level of Confidence							
α / n	10.00%	20.00%	30.00%	50.00%	70.00%	80.00%	90.00%	95.00%
	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05
1	0.1584	0.3249	0.5095	1.0000	1.9626	3.0777	6.3137	12.7062
2	0.1421	0.2887	0.4447	0.8165	1.3862	1.8856	2.9200	4.3027
3	0.1366	0.2767	0.4242	0.7649	1.2498	1.6377	2.3534	3.1824
4	0.1338	0.2707	0.4142	0.7407	1.1896	1.5332	2.1318	2.7765
5	0.1322	0.2672	0.4082	0.7267	1 1558	1.4759	2.0150	2.5706

With this and entering in the table, it implies the following level of confidence (see Table 4), that is, a confidence level of 0.575256 (or 57.526%).

Identically, we continue with a new iteration. That is, including a seventh sample in the calculation as follows:

$$n = 6$$
$$X' = 47.27\%$$
$$s = 0.06386$$

With the above values and considering an error $e = 6\%$, it is obtained for the t-student:

$$t = 1.087797$$

With this and entering in the table, it implies the following level of confidence (see Table 5), that is, a confidence level of 0.673682 (or 67.368%).

Table 5 Confidence level for $t = 1.087797$

	Level of Confidence							
α / n	10.00%	20.00%	30.00%	50.00%	70.00%	80.00%	90.00%	95.00%
	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05
1	0.1584	0.3249	0.5095	1.0000	1.9626	3.0777	6.3137	12.7062
2	0.1421	0.2887	0.4447	0.8165	1.3862	1.8856	2.9200	4.3027
3	0.1366	0.2767	0.4242	0.7649	1.2498	1.6377	2.3534	3.1824
4	0.1338	0.2707	0.4142	0.7407	1.1896	1.5332	2.1318	2.7765
5	0.1322	0.2672	0.4082	0.7267	1.1558	1.4759	2.0150	2.5706
6	0.1311	0.2648	0.4043	0.7176	1.1342	1.4398	1.9432	2.4469

Continuing to applying the same procedure with the 26 samples, the following results are obtained. Table 6 gives a summary of all levels of confidence.

With this, it is observed that the desired level of confidence is achieved with 15 samples considering an error in the accuracy of 6%. Graphically, the previous results can be observed as follows (see Fig. 1).

3.2 Simulation 2

Like the previous case, the same study can be performed modifying the value of the precision error [9]. In this way, conforming the desired accuracy is higher, the obtained level of confidence is lower.

That is, in order to achieve a 90% of confidence level:

- For $e = 6\%$, it would be obtained with $n = 15$
- For $e = 5\%$, it would be obtained with $n = 19$
- For $e = 4\%$, it would be obtained with $n = 23$
- For $e = 3\%$, it would be obtained with $n > 26$.

Graphically, the previous results can be observed as follows (see Fig. 2).

According to the graphics, and as commented at the beginning of the chapter, a protocol based on *t*-Student statistical distribution can be implemented in order to provide good results as a criterion of convergence for calculating the number of samples. Since the objective here has been to determine a minimum number of characterizations that will reduce costs and efforts, it has been depicted a statistical learning process that adjusts the number of samples to the desired confidence level considering a specific measurement error.

Table 6 Summary of all confidence levels

n	G. freedom	Mean X_n (%)	S_n	t_n	Conf. level (%)
4	3.00	45.36	0.06641	0.819682	52.75
5	4.00	47.25	0.07140	0.887936	57.53
6	5.00	47.27	0.06386	1.087797	67.37
7	6.00	47.67	0.05927	1.276841	75.12
8	7.00	48.35	0.05813	1.411495	79.90
9	8.00	47.39	0.06145	1.388278	79.75
10	9.00	47.01	0.05920	1.506743	83.39
11	10.00	46.04	0.06475	1.414823	81.25
12	11.00	46.25	0.06218	1.545962	84.96
13	12.00	46.19	0.05957	1.677609	88.07
14	13.00	45.86	0.05862	1.756268	89.74
15	14.00	46.16	0.05773	1.858310	91.57
16	15.00	46.41	0.05660	1.967574	93.21
17	16.00	46.38	0.05482	2.092721	94.73
18	17.00	45.92	0.05664	2.063765	94.53
19	18.00	45.97	0.05510	2.182232	95.74
20	19.00	45.75	0.05457	2.249603	96.35
21	20.00	45.61	0.05355	2.341845	97.03
22	21.00	45.59	0.05228	2.454272	97.71
23	22.00	45.72	0.05148	2.555719	98.20
24	23.00	45.44	0.05226	2.555689	98.23
25	24.00	45.12	0.05356	2.527522	98.15
26	25.00	45.72	0.06074	2.302961	97.01

4 Future Research Lines and Conclusions

Another interesting exercise is to observe how the previous curves evolve when the order of the samples is taken randomly. The result obtained in all cases is similar, tending asymptotically to the same values of confidence level, as higher is the number of considered samples. This new procedure has the advantage that allows to stop the sample collection at the time in which a satisfactory level of confidence for the decision-maker is reached. When different fractions are analyzed in a population, it is possible that each one requires a different number of samples in order to achieve a certain level of confidence. Therefore, it does not seem as appropriate to obtain as many samples as required for the most unfavorable fraction. On the contrary, for

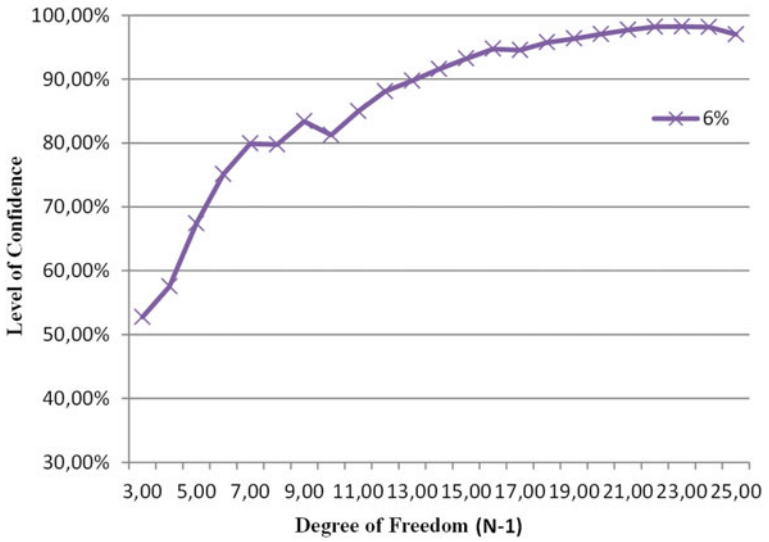


Fig. 1 Graphic summary of all confidence levels

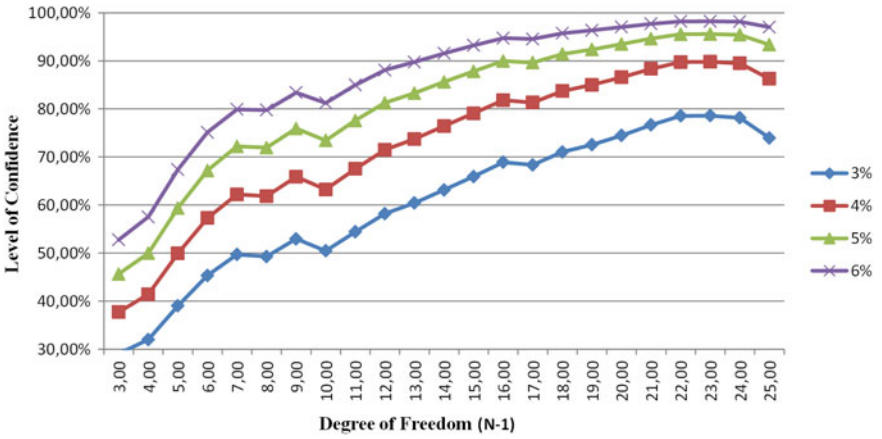


Fig. 2 Graphic summary of all confidence levels, changing the error

each fraction, the new procedure is totally independent. In other words, the number of samples needed for a fraction does not condition the number for any other.

In reference to the results obtained from simulations (where a random order of 26 possible samples has been considered), it is possible certain conclusions about the necessity or not of stratification [8]. The presented case considers simple random samples, so they must be chosen using a procedure that ensures decision. In such case, stratification would not be required [9]. Future research lines may explore the effect of emerging technologies on sampling collection, clustering, and problem-solving

related to industrial and scientific applications. In addition to this, the statistical reasoning may consider the rapidly advancing technologies thus the probabilistic models and methods may be inferenced to data mining in order to improve the decision-making under uncertainty and risks. Other interesting references regarding this topic are [10–13].

References

1. De Neufville R (2004) Uncertainty management for engineering systems planning and design, MIT Engineering Systems Monograph. http://esd.mit.edu/symposium/pdfs/monograph/uncertainty.pdf
2. Gonzalez-Prida V, Zamora J et al (2019) A risk indicator in asset management to optimize maintenance periods. In: WCEAM (World Congress on Engineering Asset Management), Stavanger, Norway, 24–26 Sep 2018
3. Helton JC, Oberkampf W (eds) (2004) Alternative representations of epistemic uncertainty. Spec Issue Reliab Eng Syst Saf 85(1–3)
4. de Rocquigny E, Devictor N, Tarantola S (2008) Uncertainty in industrial practice: a guide to quantitative uncertainty management. Wiley
5. Price C, Walker M (2019) Improving the accessibility of foundation statistics for undergraduate business and management students. Studies in Higher Education. Taylor and Francis Online. https://doi.org/10.1080/03075079.2019.1628204
6. ASTM D4687—95(2006) Standard guide for general planning of waste sampling ASTM. Test methods for evaluating solid waste, physical/chemical methods. SW-846. EPA. Publication. USEPA
7. Ramsey FP (2016) Truth and probability. In: Arló-Costa H, Hendricks V, van Benthem J (eds) Readings in formal epistemology. Springer Graduate Texts in Philosophy, vol 1. Springer, Cham
8. Crespo A, González-Prida V, Gómez J (eds) (2018) Advanced maintenance modelling for asset management. Techniques and methods for complex industrial systems. Springer International Publishing. ISBN 978-3-319-58045-6
9. Crespo Márquez A, Macchi M, Parlikad AJ (eds) (2019) Value based and intelligent asset. Mastering the asset management transformation in industrial plants and infrastructures. Springer International Publishing. ISBN 978-3-030-20703-8
10. Aven T (2003) Foundations of risk analysis. Wiley, Chichester
11. Helton JC, Cooke RM, McKay MD, Saltelli A (eds) (2006) Sensitivity analysis of model output: SAMO 2004. Spec Issue Reliab Eng Syst Saf 91(10–11)
12. Nilsen T, Aven T (2003) Models and model uncertainty in the context of risk analysis. Reliab Eng Syst Saf 79(309–317)
13. Gonzalez-Prida V, Zamora J (eds) (2019) Handbook of research on industrial advancement in scientific knowledge. IGI Global, Hershey, PA, pp 1–442. ISBN: 9781522571520

Chapter 2
Sentiment Analysis on Google Play Store Data Using Deep Learning

Swathi Venkatakrishnan, Abhishek Kaushik, and Jitendra Kumar Verma

1 Introduction

Mobile app stores such as Google and Apple have a wide range of applications to suffice the basic needs of customers in the digital platform. These days, with new renovated augmentations in technologies and ascent of opportunities, various development tool kits are readily available in the market for software developers. The developers are wary of interests of the customers and they also have an idea of the target audience with respect to the application in question, but unless they do get an appropriate feedback, there is little scope for improvement and credibility. This is what the developers yearn for. Customer feedback and ratings have always been one of the major metrics that can be used to review the performance and provide suitable recommendations to enhance the functionality provided by the app [1].

Application developers usually deal with large number of critical reviews which they have to filter out to suit the requirements of the appropriate stakeholders. The objectives of this research can be further explained as:

- Can the application-specific metrics, when combined as a whole, along with details and reviews, be used to determine the user rating?
- How can deep learning be adopted to combine different categorical, textual and numerical data to determine user app ratings?

S. Venkatakrishnan
School of Computing, Dublin Business School, Dublin, Ireland
e-mail: swathivenkat0191@gmail.com

A. Kaushik (✉)
ADAPT Centre, School of Computing, Dublin City University, Dublin, Ireland
e-mail: abhishek.kaushik2@mail.dcu.ie

J. K. Verma
Department of Computer Science and Engineering, Amity School of Engineering and Technology, Amity University Haryana, Gurugram (Manesar), Haryana, India
e-mail: jitendra.verma.in@ieee.org

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_2

- Can vectorization techniques affect the overall outcome of the classification?

We analyse the different variants of data together by combining text, numerical and categorical data. The sole purpose to take up this task was to put forth the various intricate involved in data extraction and pre-processing stage, and further study the performance of a deep learning model that would integrate the different forms of data to perform multi-class classification. This study describes both; source of the data and concatenation of deep learning models that use the functional layer of Keras for combining text vectors and numerical columns of data for performing classification [2].

2 Literature Review

Hengshu Zhu et al. [3] mainly emphasize on how the mobile app market is way more susceptible to fraud activities such as pushing up the apps in the popularity list. To counter this, they have explained a view of the current ranking fraud cases and proposed an approach to study the fraud detection system to determine which app ranks have spam markings. The two main features considered for training the model are ranking-based evidences and application-based ratings. Hypothetical statistical tests are performed on both; app rankings and rating variables; to study the varying trends of these features to determine the fraud occurrence. This study helped us to understand which mobile app features have more significance to study the varying trends in rankings.

Data mining is not just restricted to text or image processing but it involves a lot of other important factors such as multimedia mining or graph mining [4]. So this knowledge discovery process is elaborated in depth by the categorization process that performs on the same ideology as that of bag of words and the natural language processing (NLP) technique. Vijayarani et al. have given a detailed description of the extraction process, the stop words elimination techniques, stemming and comparison of methodologies adopted for removing stop words like using pre-compiled lists, Zipf's law, mutual information methods and term-based random sampling. They also showcase scenarios in which the stemming operations based on statistical analysis are performed. Here, it helped us in understanding the significance of stemming operation on text data to improve the overall performance of the model.

Abhishek and Sudhanshu [2, 5] have provided an extensive critique about various text mining techniques, tools and applications in this article. A wide-scale analysis was performed on several terminologies and techniques of data mining such as categorization, clustering, topic tracking and sentiment analysis to name a few. Kaur et al. [6] have investigated the sentiment analysis on Hinglish data using the semi-supervised learning using different forms of vectorizer in parametric and non-parametric model.

Zhang et al. [7] basically provide an overview of how deep learning and neural network can be used to showcase a comprehensive survey of the present scenario with respect to the applications used for sentiment analysis. It is stated how deep learning

uses multiple layers of processing models for main feature extraction. This study introduces various in-depth explanations about various deep learning architectures and how each of them, when used as an individual model or as a combination of various neural nets can thereby depict state of the art results for various text processing techniques.

Hinton and Salakhutdinov [8] have stated that deep learning methodology, in the long run, will be a widely used approach for text processing owing to its high tendency to study the data in a more refined way. If a data of high dimension is showcasing its constraints with respect to its varsity and shape; it can very well be converted to a low-dimensional form by training a multi-layer neural network. It also goes ahead to state that this auto-encoder network might be a more effective way than principal component analysis to learn the low-dimensional form of vectors.

Neural networks have always been a star performer by achieving remarkable results in sentence and document modelling [9]. When two popular and prevailing architectures, CNN and recurrent neural networks (RNN), are implemented for language processing tasks, we can observe that they adopt totally different methodologies for implementation. The end result indicates distinctly that the combined model outperforms both the individual models.

Isa et al. [10] implement a modified hybrid classification technique through the Naive Bayes approach and the support vector machine (SVM). This project focuses on the Bayes formula to vectorize a document rather than using the traditional classification technique. This technique comparatively uses less training time period than the L Squared methodology and has no constraints with respect to the type of data being used. On further evaluation, the classification accuracy increases to a legit scale compared to Naive Bayes systems.

Abhishek et al. [5] have proposed a system where one can automate rate the opinions in the form of unstructured data. Although it is a challenge in the present scenario, this paper presents an in-depth ideology on sentiment analysis methods and tools being used for the same. They have further elaborated the workflow of opinion mining where the techniques used for opinion mining involves opinion retrieval, classification and summarization.

Blessy et al. [11] have put forth different approaches of sentiment classification and the existing techniques with the framework. It states that the feature extraction stage deals with feature types specifically for opinion mining; in this context, feature selection phase opted for opinion classification, feature weighting strategy and finally reduction mechanisms for optimization purpose.

3 Data

3.1 Data Extraction

It is essential that the data source for research should be completely authentic. The play scraper authentic API package is available for users for extraction purposes. This package consists of major filters which can be used while fetching the data.

Table 1 Sample of app categories

Categories	Top free	Top trending
Weather	TOPFREE_WEATHER	TRENDING_WEATHER
Maps and navigation	TOPFREE_MAP_ AND_NAVIGATION	TRENDING_MAPS_ AND_NAVIGATION
Medical	TOPFREE_MEDICAL	TRENDING_MEDICAL
Photography	TOPFREE_PHOTOGRAPHY	TRENDING_PHOTOGRAPHY
Games	TOPFREE_GAMES	TRENDING_GAMES

Some of the user-friendly pointers used would be fetching data as per collection of sections such as trending, top ranked; top paid, top free, country or region-specific since it involves reviews from varied customers; developer-specific applications; to fetch details as per a developer's offered set of applications. Further, each of these sections is bifurcated into multiple categories. The whole list of packages and python files used is as listed out below:

1. Play scraper link[1]
2. Enhanced version of reviews module[2]
3. Built Python scripts.

For this particular research work, the data fetched mainly belonged to top trending and top free collection. Each collection had 58 categories. To mention a few as shown in Table 1. The data fetched through play scraper API (application programme interface) was in JavaScript Object Notation (JSON) format. A python script was used to convert the nested JSON structure to a CSV file format. This JSON form was converted to a CSV file form using a python script named *FetchJSONConvertToCSVDetails.py*.

3.1.1 Structure of Play Store

This third-party package is mainly used for extracting application details. However, this package does not have a module for extracting reviews for the apps. One enhanced version of the same module is available on GitHub for fetching reviews. This module works on this principle of fetching application reviews with respect to the application ID specified. They have basically created a new function that deals with reviews section and invokes the review URL. It fetches reviews and related details of the application ID specified.

Since the scope of this project involves dealing with multiple applications within each category that further falls under major collections; it would be tedious to invoke

[1] https://pypi.org/project/play-scraper/#details.

[2] https://github.com/danieliu/play-scraper/commit/3906d4f6814073cdc5c83b88635607dcc99ff3d5.

reviews of each application one-by-one. So, we have created a new script *MultipleAppsReviews.py* that can be forked for extracting reviews of all applications of the specified category name. It also involved tweaking the code in the python scraper main class *scraper.py* where we have augmented that part of the code where now, it accepts a list of apps instead of a single app name as an input and passes this entire list as one parameter in the POST request to the specified Uniform Resource Locator (URL). This is in turn retrieves the reviews of all the applications appended in the list.

3.1.2 Work Flow of Web Scraping

This play scraper is a python API. Figure 1 indicates the stage-by-stage functioning of each script used for getting the data from the play store. The detailed technical parameters and purpose of each script are given in the Github link provided in the references [12]. Further, the scraper.py file that was enhanced for fetching reviews of multiple applications falling under a category is also provided in Github [13].

3.2 App Details and Review Details Specifics

3.2.1 Details

- Fetch app details as per collections: TOPFREE and TRENDING.
- Every collection has 58 small categories.
- Each category has multiple applications that are sorted under it.
- Every application has a set of detailed metrics to be used for analytical processing.

Fig. 1 Entire workflow

- Columns for application details: application Id, category, content rating, current version, description, description HTML, developer, developer address, developer email, developer ID, developer URL, editor's choice, free, histogram, IAP, IAP range, icon, installs, interactive elements, price.

3.2.2 Reviews

- Since we have fetched application details with respect to collection and categories; it is advisable to fetch application reviews too on similar lines.
- List of columns fetched in the reviews case are as follows: author image, author name, current rating, review app ID, review body, review date, review ID, review permanent link, review title.

Since this research involved studying just application detail-specific and review-specific features, we have dropped out on all the other unnecessary columns to avoid clogging the idea.

1. Merging the app details within two collections:
   So, as per the design plan, there are two major collections: trending and top free. Within each of these collections, there are 58 CSV files containing application details of apps where each sheet represents a unique category within that collection. We combine all the 58 categories data sheets falling under one collection 'top free' with other 58 categories CSV sheets falling under the other collection 'trending'. A python script *CombineAllDetailCollections.py* was used to merge them.
2. Merging reviews within two collections:
   Similar to the details case, we merge the reviews under two different collections within one sheet. Both the merge functions are performed using Python scripts.

4 Dataset Description

While there were minor technical glitches while concatenating different types of data; the main focus was to maintain the overall quality of the data; after we combined the details and the reviews. As seen, the dataset seemed quite imbalanced with respect to the number of rows maintained in both these sheets. Each app had unequal number of reviews. To resolve this, we have considered just 20 rows of reviews for each app using newly created python script *ReviewHead.py*. To ensure that we have complete information of an app; all numerical and categorical columns in details are adjoined to the textual columns in reviews; stacked up horizontally against each other; a python script named *CombineDetailsToReviews.py* was created to combine the details of the apps present in the details sheet to that of the reviews sheet keeping application ID as the base identifier. When all 58 categories are present, it would be a little tedious to work through.

To make it simpler, sub-groupings of categories was added. Since there were a varied range of 58 categories involved, we segregated the categories into 14 broad unique categories as shown in Table 2.

The main broad categories are further label encoded to pass through the learning model.

- Improved performance of predictors
- Cost effective.

Table 2 lists apps having similar base functions under one major category.

Table 2 Number of sub-categories falling under each broad category

Broad categories	Sub-categories grouping
ART_AND_DESIGN	Art and design
AUTOSHOP	Auto and vehicles
PERFORMANCE_BOOST	Lifestyle/beauty/parenting/personalization/productivity/ tools/video players
KNOWLEDGE	Books and reference/business/education/libraries and demo
ENTERTAINMENT_LIST	Entertainment/music and audio/comics/events/photography/shopping
COMMUNICATION	Communication/dating/social
GAMES & FAMILY ACTVITIES	Game arcade/game casual/game card/game action/game adventure/game board/game card/game casino/game casual/game educational/game music/game puzzle/game racing/game role playing/game stimulation/game sports/game strategy/game trivia/game word/sports family pretend/family music video/family education/family create/family brain games/family action/family
HEALTHCARE	Medical/health and fitness
FINANCE	Finance
FOOD_AND_DRINK	Food and drink
HOUSING	House and home
MAPS_AND_TRAVEL	Maps and navigation/travel/travel and local
ENTERTAINMENT_LIST	Entertainment/music and audio/comics/events/photography/shopping
NEWS	News and magazines
WEATHER	Weather

5 Data Exploration

To study the impact of each of the weighted attributes such as installations, rating score in each of these categories, exploratory data analysis was performed. The distribution of the different major categories across the dataset is shown in Fig. 2.

Clearly, the value counts of GAMES category was the highest and the number of applications and categories falling under WEATHER was the least. The range of total average rating against each of the main categories and distribution of installs against every category is depicted using a scatter plot in Figs. 3 and 4.

6 Methods

The feature variables being selected for this research work were application ID, unique broad category, total average rating, number of installations, required android version, user review text, user rating, total number of reviews.

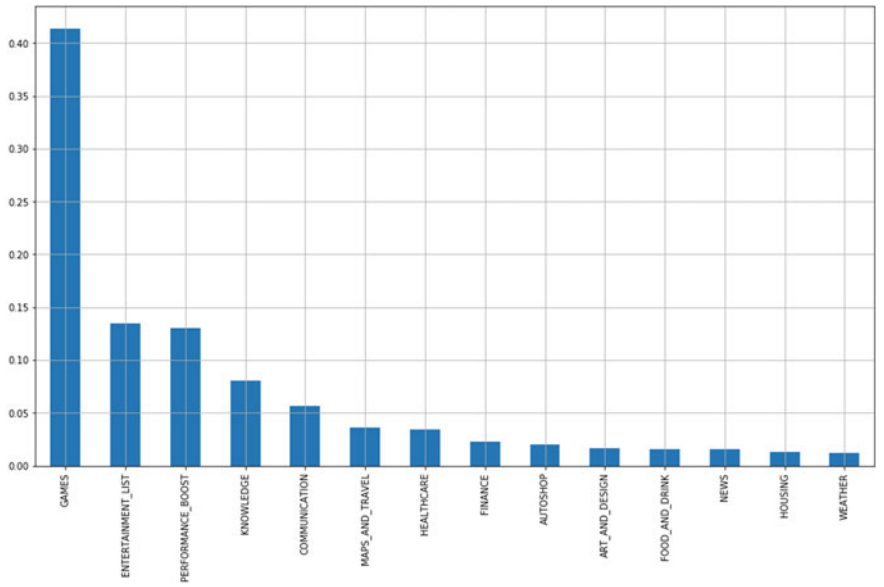


Fig. 2 Distribution of the different major categories

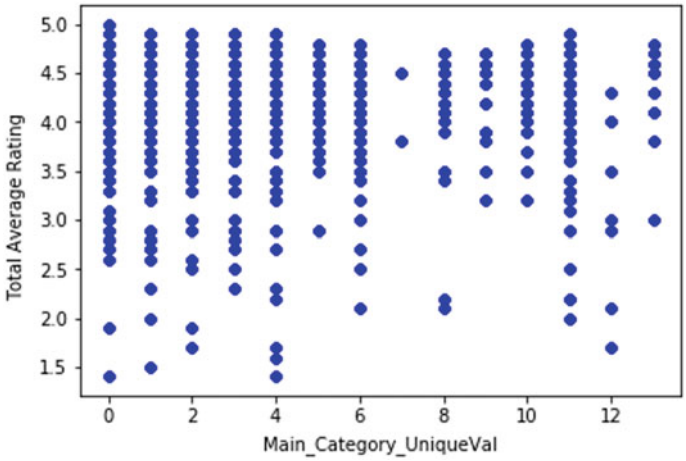
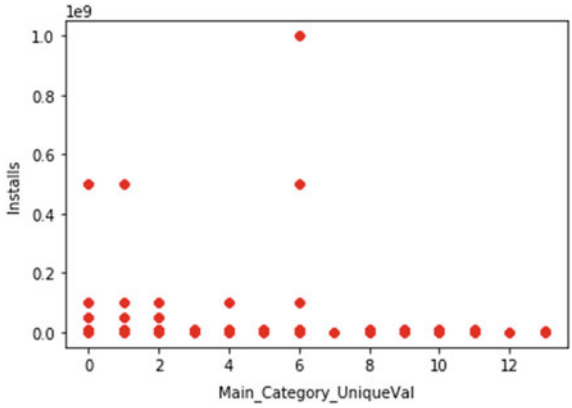


Fig. 3 Distribution of main categories versus average rating

Fig. 4 Distribution of main categories versus installs



6.1 Data Processing

Since we were dealing with raw extracted data, our primary goal was to clean it to avoid any discrepancy in our results.

6.1.1 Text Vectorization

The Scikit-learn library provides user-friendly libraries to use count vectorization and TF-IDF vectorizer on real text data. Before we could jump into the vectorization part, it was crucial to remove stop words, encoded punctuations, and lemmatize our content to maintain the quality decorum during processing.

Table 3 Results of some tested classifiers

ML classifier	CV accuracy (in %)	TF-IDF accuracy (in %)
Naive Bayes	67.29	70.13
XG Boost classifier	66.92	67.63
MLP classifier	67.58	65.68

1. **Count Vectorization**: This is one of the most basic ways that can be utilized to represent text data numerically. This concept is also called as one-hot encoding. The basic process that is being implemented here is we create vectors that have a dimensionality equal to the size of the vocabulary. At every instance of encountering a vocab word within a textual review; we increase the count by 1 and add 0 in place of null encounters.
2. **TF-IDF Vectorization**: The term simply means "term frequency—inverse document" (TF-IDF) frequency. This basically implements the vectorization by performing the TF-IDF transformation of the matrix of counts.

Both count vectorization (CV) and TF-IDF were applied on the same column to evaluate the performance of both the results in the case of text classification classifier model Table 3.

6.2 Numerical Data Columns

There were multiple numerical columns of different data types that were involved in this research work. Each value range was quite extensive, so it was essential to scale it down to a normalized range. List of columns that were normalized are as follows:

- Total Number of Reviews: Total number of reviews ranged from 10–15 to maximum range of 48160293. To ensure it stays in the same range as the other columns used in experiment, we use Min-Max Scalar libraries to reshape our column to lesser range.
- Installs: Total number of installations that have occurred for each application also has huge variations in the values. To bring it down to a normalized form, we divide the entire column by the maximum value in the column '1000000000'.
- Required Android Version: Required android version was one more column that had varying values which would have been time consuming for the model to interpret. This too was scaled down to a reasonable value using Min Max scalar.
- Average Rating: Total average rating basically ranges from 1 to 5. To bring it down to a value similar to other columns and in the [0,1] range, we divide the entire column using max value of the column.

6.2.1 Categorical Encoding

Since the output variable is varied range from 1 to 5; we need to label encode it to be used while processing the model. This is done using the to_categorical package provided by scikit-learn.

7 Modelling Results

Using our refined dataset [14], we have used deep learning classifiers to determine user ratings on the basis of all other application metrics. Functional layers of multi-layered perceptron classifier were used to build the model architecture. The entire design specification is as shown in Fig. 5. After performing web scraping and combining multiple sheets into one dataset, we divide the data frame into 30% test set and 70% train set. After the pre-processing stage, we divide our modelling stage into two parts:

1. For application details, we select the best performing classifier amongst MLP and DDMLP.
2. Similarly, we select the best performing model amongst MLP models in case of application reviews; the only difference being the vectorization method used for converting text reviews to the vector form.

The first sub-model was built specifically for text classification on the basis of user ratings falling within the range of 1–5. Second sub-model was solely developed to predict the user ratings from (1–5) with respect to the other numerical feature variables. The combined hybrid model was used to merge two-layer outputs of both the sub-models and process the entire data frame that was a combination of text vectors, numerical values and categorical features. This combined hybrid model recorded an optimal test accuracy of 70.45%.

This functional input layer gives the feasibility to first build layers of input models. Here, each layer is assigned a different name so it can distinctively identify each layer weights while training the model. The first input layer is the visible layer that takes in the set of inputs given to model for training. The number of feature variables added here was 5. So the shape of the value was fed as 5 in the input layer itself. The next three layers added were hidden layers constructed with ReLu activation functions. The last hidden layer was the Softmax functional layer that gives out the output of the app details classification. This output layer is named as output 1. The second input model has one input layer, two hidden layers and an output layer. There were 25, 5 and 6 neurons in each of those layers, respectively. The input layer and the intermediate layer neurons are activated by the ReLu function. The output layer is aligned with the Softmax function that produces a set of six-dimensional output vector where each row would signify the probability that the sample belonged to that class. This app reviews classification would directly indicate the user rating that the class would fall under. The output layer of these set of layers is called as output 2.

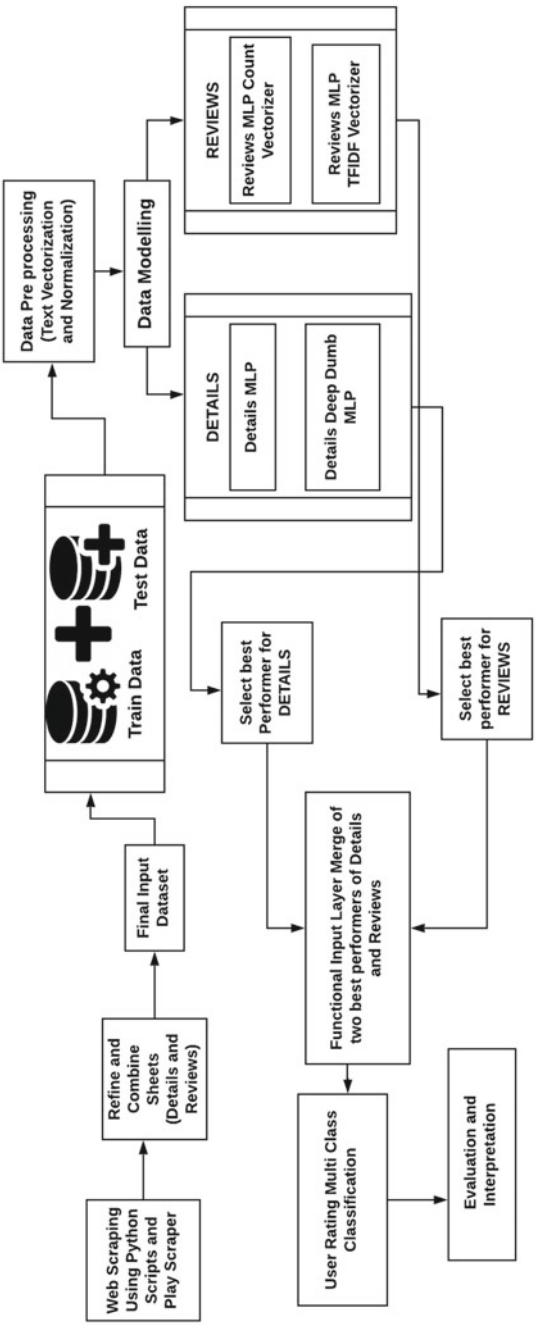


Fig. 5 Data modelling process flow

Keras provides few functions such as averaging the inputs and even concatenating the inputs to create a new input layer for the third set of layered network model to form a combination. The output 1 and output 2 are combined to form a merged content. This merged value is then provided as an input to the hidden dense layer activated by the ReLu function. The inputs consist of different sets of data, which would produce a six-dimensional vector of user ratings. This last layer is also embedded with Softmax function. The model is trained using the best probable optimizer rmsprop and sparse categorical cross-entropy loss function.

Since a number of models and their combinations were implemented, the variations of accuracy across different epochs and batch size were recorded which is shown in Table 4.

The ROC of the multi-class classification of MLP model was used to interpret the performance of the classifier. The curve shown in Fig. 6 shows the performance of the model embedded with text vectorized using TF-IDF concept. This is used

Table 4 Epochs and batch size for deep learning

Classifier	Epochs	Batch size	Train Acc	Test Acc
MLP-details	25	400	64.32	64.77
DDMLP-details	25	400	64.02	64.34
MLP-countVec	25	400	69.08	67.61
MLP-TF-IDF	25	400	88.73	66.56
Functional count	25	400	68.44	67.26
Functional-TF-IDF	25	400	85.04	68.62
MLP-details	12	450	64.22	64.45
DDMLP-details	12	450	63.42	63.4
MLP-countVec	12	450	68.18	68.16
MLP-TF-IDF	12	450	85.96	68.98
Functional count	12	450	67.63	67.83
Functional-TF-IDF	12	450	75.12	70.45
MLP-details	15	500	64.33	64.36
DDMLP-details	15	500	63.47	63.3
MLP-countVec	15	500	68.47	68.2
MLP-TF-IDF	15	500	87.77	67.86
Functional count	15	500	67.82	67.99
Functional-TF-IDF	15	500	75.92	69.94
MLP-details	13	480	63.96	63.99
DDMLP-details	13	480	63.45	63.44
MLP-countVec	13	480	68.24	68.17
MLP-TF-IDF	13	480	86.84	68.78
Functional count	13	480	67.99	68.07
Functional-TF-IDF	13	480	78.43	70.32

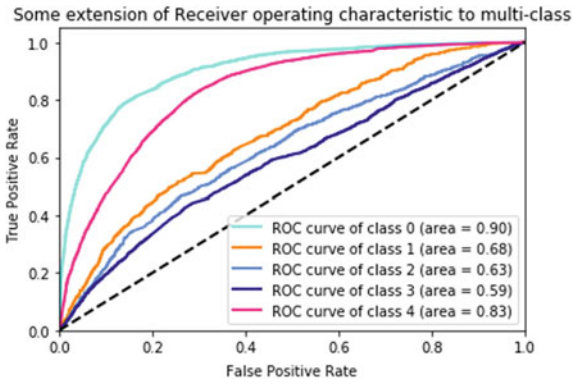


Fig. 6 ROC curve

to evaluate the quality of the output produced by the classifier. It aims in depicting the true positive rates on the *Y*-axis and the false positive rates are determined by the *X*-axis. The ROC works in a way to showcase that an ideal curve should indent more towards the true positive score of 1 and a false positive score close to 0. The projection should be such that the curve produced should have more area under them to determine the effectiveness of the compiler. The graph produced, indicates that class 0 and class 4 data, which means data classified under rating 1 and rating 5 were correctly classified as they projected a decent area score. Classes 2, 3 and 4 which correspond to ratings 2, 3 and 4, respectively could produce an improved area score by adding batch normalization and other pruning techniques to fetch a better result. This action is in pipeline for improving the overall score of the classifier.

8 Discussion

The main aim of this research process was to focus on the data refining process and combine different types of data; details and reviews of the mobile apps data, to determine the user rating by implementing deep learning model MLP. Although, there have been extensive research on the lines of text classification and implementation of deep learning as well as machine learning concepts for mobile apps data, not much light has been shed in the area of multiple inputs combination of variants of data using deep learning. TF-IDF does seem more appropriate and superior in some cases for text classification scenarios as it balances out the frequency of the entire text tensor under question using its inverse document frequency as opposed to count vectorizer which maintains the vector of frequency of a word occurrence in document or text corpus. It also depends on the content of the textual reviews under question, for differences in the performance scale of count vectorizer and TF-IDF. The design plan was to implement a combined approach using the best performing models of MLP for both details and reviews to attain maximum accuracy. From the table, for different epochs and batch sizes; it can be concluded that the combined functional model shows

better performance than as individual units. The MLP model gives a decent testing accuracy score of 70%, it can be improved by trying few combinations of neural nets. This, thereby, satisfies our initial research theory that this model performs well to determine the app user ratings, given various play store app metrics and user reviews. One major hurdle that was overcome was the ease with which the functional MLP neural model fits the entire data, combination of both the models, into its layers. Also, the combined model gave better results compared to individual models. Though there is still a lot of scope for improving the results, only a few variants of MLP network model was implemented. A lot of time was invested in the data extraction and data refining process. Owing to this time constraint, larger combinations of neural nets could not be implemented although it is in the pipeline. The scope for this research is huge. The various trends and key factors of mobile app stores can be exploited to unearth new findings and predictions.

9 Conclusion

Most of the prominent web-based applications could sustain their hold in top ranking charts for a long run mainly because the developers are well aware of the user requirements and user satisfaction criteria. App developers look forward to receiving credibility for their work in the form of rankings and reviews. The play store feature variables, irrespective of being largely unstructured and varied in its data format as well its content, can act as performance metrics if they can be pre-processed and well trained to determine the user-specific ratings. Although this research focused on the columns that we felt would be appropriate for analysis, we are keeping the topic open for other researchers too to experiment as per their needs. Although the DOI for the data [14] is given and the post-processing operations are irreversible, the original extracted files are provided in Github [12] for other further analysis. The future scope of the project includes testing and building the model on various other combinations of neural networks that can help in improving its performance.

References

1. Shah SR, Kaushik A (2019) Sentiment analysis on indian indigenous languages: a review on multilingual opinion mining
2. Kaushik A, Naithani S (2016) A comprehensive study of text mining approach. Int J Comput Sci Netw Secur (IJCSNS) 16(2):69
3. Zhu H, Xiong H, Ge Y, Chen E (2014) Discovery of ranking fraud for mobile apps. IEEE Trans Knowl Data Eng 27(1):74–87
4. Vijayarani S, Ilamathi MJ, Nithya M (2015) Preprocessing techniques for text mining-an overview. Int J Comput Sci Commun Netw 5(1):7–16
5. Kaushik A, Naithani S (2014) A study on sentiment analysis: methods and tools. Int J S Res (IJSR) 4:287–292

6. Kaur G, Kaushik A, Sharma S (2019) Cooking is creating emotion: a study on hinglish sentiments of youtube cookery channels using semi-supervised approach. Big Data Cogn Comput 3(3):37
7. Zhang L, Wang S, Liu B (2018) Deep learning for sentiment analysis: a survey. Wiley Interdiscip Rev Data Min Knowl Discov 8(4):e1253
8. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
9. Zhou C, Sun C, Liu Z, Lau F (2015) A c-lstm neural network for text classification. arXiv preprint arXiv:1511.08630
10. Isa D, Lee LH, Kallimani V, Rajkumar R (2008) Text document preprocessing with the bayes formula for classification using the support vector machine. IEEE Trans Knowl Data Eng 20(9):1264–1272
11. Selvam B, Abirami S (2013) A survey on opinion mining framework. Int J Adv Res Comput Commun Eng 2(9):3544–3549
12. Venkatakrishnan S, Kaushik A (2019) Extracted csv files. https://github.com/abhishekkshk68/Play-Store-Deeplearning-Analysis/tree/master/Extracted_CSV_Files
13. Venkatakrishnan S, Kaushik A (2019) Data refining scripts. https://github.com/abhishekkshk68/Play-Store-Deeplearning-Analysis (June 2019)
14. Kaushik A, Venkatakrishnan S (2019) Google play store data. https://doi.org/10.5281/zenodo.2844022

Chapter 3
Managing the Data Meaning in the Data Stream Processing: A Systematic Literature Mapping

Mario José Diván and María Laura Sánchez-Reynoso

1 Introduction

The data itself represents a captured fact or event which describes some kind of aspect related to the real world [1]. Even when the data could be persistent not always it is necessary. That is to say, when data collecting devices are providing data to make decisions in real time, always there will be newly updated data replacing the previous ones. The underlying idea of data-driven decision making is the using of data for supporting decisions avoiding the intuition. The information is data that simultaneously is trust, consistent, interest, and timely. Accordingly, it is possible to say that information is data, but data not necessarily are information [2, 3].

The information tends to be continuously required when managers need to make decisions. Updated information will imply lesser uncertainty in the decision environment than before. The real-time decision making jointly with the data stream engines constitute two faces on the same coin because when one is focused on collecting information itself, the second one is oriented to process it as quickly as possible [4, 5]. The data stream engines process the data on the fly and in the same way in which they arrive. However, the possibility of discriminating the meaning associated with each one will imply a better position at the moment in which a model (e.g., a classifier) needs to be trained or applied. Even more, it would be a key asset when some recommendation needs to be searched from repositories (such as the organizational memories), because of the expected accuracy using the data meaning possibly would be improved [6, 7].

While the big data environment is focused on batch processing, the data stream environment is oriented to the processing in real time at the instant in which the data

M. J. Diván (✉) · M. L. Sánchez-Reynoso
National University of La Pampa, C. Gil 353. 1st Floor, 6300 Santa Rosa, Argentina
e-mail: mjdivan@eco.unlpam.edu.ar

M. L. Sánchez-Reynoso
e-mail: mlsanchezreynoso@eco.unlpam.edu.ar

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_3

is received. That is to say, in the big data environment, it is possible to have a huge volume of data and it could be as big as possible but limited at the end, while in the data stream environment there is no limit oriented to the data volume coming from each data stream. The possibility of adequately using and applying the data meaning for guiding the data processing in the last one, it is bounded to the available processing resources in a given time (i.e., processor, memory, cache, etc.) [8].

The main contribution in this work is to carry out systematic mapping of the literature to identify and find the strategies responsible for modeling the data meaning and its application in the real-time data processing bounded to data stream environments. This is made through the use of the research method named systematic mapping studies (SMS) on the Scopus database, which is explored and analyzed following the filters that allow limiting the specific field and inclusion/exclusion criteria. Thus, data semantics, the effect in the data processing, and the impact in the online decision making are here dimensioned from the research point of view. Finally, the different strategies for managing the data meaning and its impact on the real-time data processing are outlined, dimensioned, compared, and classified.

This paper has five sections. Section 2 describes the underlying idea associated with the research work, while the systematic mapping studies methodology is introduced. Section 3 outlines the step-by-step protocol application related to the SMS research method. Section 4 summarizes the comparison related to each data-meaning strategy in relation to real-time decision making. A brief synthesis and classification of strategies are proposed. Finally, Sect. 5 presents reached conclusions and future work.

2 Research Method

Previous works have carried out systematic mapping on data semantic from the data interoperability point of view [9, 10], highlighting the necessity of the pragmatic interoperability. Another approach associated with big data, Internet of Thing, and streaming analytics was carried out by means of techniques related to surveys [11]. However, the data-meaning modeling in the context of the data stream requires real-time interpreting and processing, an aspect that is here included.

This research is based on the systematic mapping studies (SMS) in order to analyze the whole spectrum of works associated with data meaning in the data stream environment, focusing on real-time decision making. SMS is applied in terms established in [12–16] and following the recommended guidelines. This method is especially useful when there is little evidence, or alternatively, the research topic is potentially very broad. In another case, the systematic literature review (SLR) is able to be applied.

Basically, SMS is described through a protocol constituted by six stages, as follows: (1) ***aim***: It is necessary to define the need for carrying out the revision, (2) ***questions***: It defines the questions which guide the research, (3) ***search strategy***: Once the aim and research questions were established, the search strategy is defined based on such questions, (4) ***data extraction***: From the given results applying the

research questions and the search strategy, the data extraction and its associated steps are defined. The filters and criteria for extracting data are specified, (5) ***synthesis***: From extracted data, the synthesis and its associated process are specified, and (6) ***assurance***: It introduces the mechanism for reviewing the development of the protocol.

Next, this section is breaking down based on each stage of the protocol with the aim of formally defining each stage, instantiating the application parameters.

2.1 Aim and Research Questions

The aim is defined as the implementation of a systematic mapping of the literature with the aim of mapping strategies that allow discriminating the data meanings and use them for guiding the processing in the data stream context.

In this sense, and according to the chosen methodology, the research questions (RQs) which will guide the work are properly defined as follows:

- RQ1: What kind of alternatives exists for modeling data meaning?
- RQ2: What alternatives are feasible to be applied in real-time data processing?
- RQ3: What are the benefits and concerns related to the application of each alternative on data streaming?
- RQ4: Is there some articulation or orientation between alternatives for modeling the data meaning and recommender systems?
- RQ5: What types of publications focused on the modeling of data meaning in real-time data processing for supporting decision making?
- Each question has an associated motivation which is specified in Table 1.

As it is possible to appreciate in Table 1, the research questions go progressively specifying from the general perspective of the strategies oriented to model the data meaning, up to specifically to analyze the application of those ones in real-time environments articulated with the recommender systems.

2.2 Search Strategy

Once the research questions have been defined, the next step implies to define the search strategy based on the general aim, which is shown in Table 2. It indicates the major term in which the subject related to the research questions is expressed. In this case, the major term is focused on data semantics, considering "data meaning" as its synonym, while variations associated with the strategy and processing point of view are included.

The search strategy is performed on the Scopus database, focusing on the computer science area, while the type of documents was limited to conference papers, articles, and book chapters. In addition, the strategy considers documents written in English,

Table 1 Research questions and its motivations

Research questions	Motivation
RQ1: What kinds of alternatives exist for modeling data meaning?	It pretends to generate a whole map from the identification of all available alternatives for modeling the data meaning, independently the viewpoint of each one
RQ2: What alternatives are feasible to be applied in real-time data processing?	The idea is to focus on alternatives feasible to be applied in real-time environments. That is to say, those alternatives which allow using the data meaning for guiding the data processing at the time in which the data arrives
RQ3: What are the benefits and concerns related to the application of each alternative on data streaming?	The aim is to describe benefits/strengths, weaknesses, concerns, and trends associated with each alternative able to be applied to the real-time environment in a comparative way
RQ4: Is there some articulation or orientation between alternatives for modeling the data meaning and recommender systems?	The objective is to determine whether the data meaning strategy, which is able to work in real-time environments, is prepared in some way for working with recommender systems or not
RQ5: What types of publications focused on the data semantics exist?	The kind of publications allows knowing whether published results are partial or not, jointly with the level of maturity and evolution over time

Table 2 Main and alternative terms related to the search string

Major term	Alternative terms
Data semantics	"Data-meaning modeling" or "data meaning" or "data-meaning strategy" or "data-meaning processing"

without any time restriction, in which the keyword includes "data semantics," and some of the major or alternative terms are included in the abstract.

2.3 Filtering Results

From results obtained applying the general and alternative terms, this stage filters them using the inclusion and exclusion criteria, such as they are described in Table 3.

The duplicated items are removed before applicating the filtering. Thus, the filtering is applied for discarding or not each item in a selective way based on the abstract's reading. The non-discarded items become part of the definitive list.

Table 3 Inclusion and exclusion criteria related to the selection process

Inclusion criteria	1. The abstract must contain at least one of the terms (alternative or main) 2. The paper's contributions must be related to the data semantics
Abstract's exclusion criteria	1. The document does not focus on data semantics as core contribution, challenge or aspect 2. The article does not satisfy the search string
Full-text's exclusion criteria	1. When the paper language is not English 2. Synthesis of invited talks or keynotes 3. When the application core is not related to the data itself 4. Comparative analysis or surveys

Table 4 Data extraction point of view

RQ	Dimensions	Categories
RQ1	Semantic model	Representation, interpretation, method
RQ2	Data processing	Real-time, semantic-driven data processing, interpretation, decision making
RQ3	Application	Real-time, semantic-driven data processing, interpretation, decision making
RQ4	Use of the knowledge	Representation, interpretation, method, decision making, recommender systems
RQ5	Publications	Data semantics, real-time, semantic-driven data processing, recommender systems

2.4 Data Extraction Process

In this stage, each article is read for deciding whether it is retained or not. Once the list contains all the retained articles, classification is carried out based on the defined categories from the research questions. The categories represent the interest topics associated with the dimensions (i.e., points of view) as it is possible to appreciate in Table 4.

Each article is classified in one or more categories, and from those categories, it is able to gather papers working around the same topic contrasting different alternatives.

2.5 Synthesis Process

In this stage, retained and classified articles are analyzed for eliminating duplicates (from the conceptual point of view), discriminating alternatives, and selecting those which strictly satisfy the search chain jointly with the associated filtering criteria. In other words, this stage is expected that each article at least gives a partial answer to the research questions introduced in Table 1.

Finally, an assessment of the protocol application could be contrasted from the synthesis result itself and the grade of answering given to each research question.

3 Performing the Systematic Literature Mapping

Table 2 synthesizes the main and alternative terms used in the search string. Thus, the search string looks for expressions such as "data semantics," "data-meaning modeling," "data meaning," "data-meaning strategy," or "data-meaning processing" contained in the abstract of each article in Scopus. Those articles which contain some of the previous expressions will be retained as a part of the result (i.e., 650 documents).

Next, Sect. 2.2 defines the profile of the search strategy, indicating that the subject area is "computer science". The kinds of documents to be considered are "conference paper," "article" (journals), and "book chapter," focusing on those who contain "data semantics" as a keyword. All the kinds of documents must be written in the English language without any limitations related to the time. The filtered results are outlined in Fig. 1, which contains 151 records simultaneously satisfying the imposed conditions.

From the individual reading of each abstract associated with the filtered list (i.e., 151 records), just 61 records were retained considering the aim, major terms, and research questions. The rest of the papers (i.e., 89) was discarded because they were related to keynotes, invited talks, introduction letters, business policies, business processes, rules, study cases, surveys, among other aspects in which the data semantics

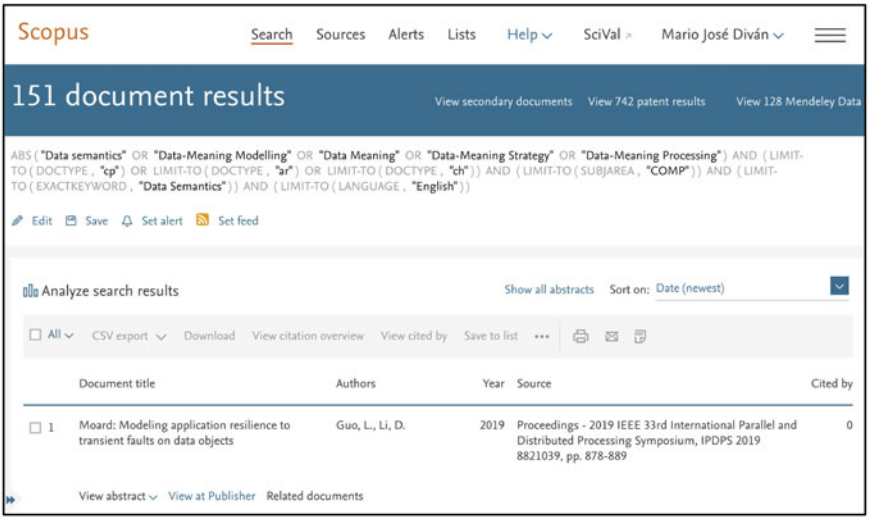


Fig. 1 Screenshot related to documents found on October 17, 2019 18:44 (Argentina)

indeed do not constitute a main focus or subject under research. Basically, the 61 records are results based on the inclusion and exclusion criteria defined in Table 3.

Table 5 shows the filtered results grouped by year and kind of publication, which is useful for identifying the time period associated with the topic. The topic started to be analyzed from 1996, and it continues at the date with different levels of intensity. In the time period [1996; 2019), thirty-three articles of journals (54.10%), one book chapter (1.64%), and twenty-seven conference papers (44.26%) were written. The most active year in the topic up to now was 2012 with seven articles, followed by 2011, 2010, and 2008 with six articles each one.

Considering the data extraction point of view introduced in Table 4, there are five non-exclusive defined dimensions: semantic model, data processing, application, use of knowledge, and publications. That is to say, given an item, this could be associated with more than one dimension because each one analyzes a different perspective of the subject. Thus, under the "semantic model" dimension there were 60 documents, while the "data processing," "application," "use of knowledge," and "publications" dimensions, there were 56, 56, 59, and 55 items, respectively.

4 Summary of Results: Data-Meaning Strategies and Real-Time Data Processing

In order to satisfy the research questions and motivations introduced in Table 1, just those items from the 61 that included at least one of the following key characteristics were retained: (i) *"data processing" dimension*: real-time processing, semantic-driven data processing, or decision making; (ii) *"application" dimension*: semantic-driven data processing or decision making; and (iii) *"use of knowledge" dimension*: decision-making or recommender systems. In this way, an item could have a lot of characteristics but each one must at least contain one of the mentioned for being fitted in terms of the research questions and to be included in the comparative analysis. Finally, applying the mentioned criteria for warranting the satisfaction of research questions, just 30 were considered as you can see in Table 6.

With the aim of making easy the comparative analysis, the table below discriminates for each highlighted item, the authors, aim, rate (i.e., level of answering to RQ), jointly with the individual contribution (when it is available) to the following aspects:

- *Real-time (RT)*: It indicates whether the work describes or not some aspect related to real-time data processing;
- *Semantic-driven data processing (SDDP)*: It refers to those semantic aspects which could be used for guiding the data processing in case they are present;
- *Decision making (DM)*: It synthesizes those aspects oriented to support the decision-making process in relation to the data processing itself;
- *Recommender system (RS)*: It introduces highlighted aspects oriented to give recommendations in front of some situations.

Table 5 Filtered results grouped by year and kind of publication

Year	BC	C	J	Accumulated	Year	BC	C	J	Accumulated
1996			1	1	2010	1	1	4	31
2002			2	3	2011		3	3	37
2003		2	2	7	2012		4	3	44
2004			3	10	2013		2	2	48
2005			1	11	2014		1	2	51
2006			2	13	2015		1	3	55
2007		2		15	2017		2		57
2008		4	2	21	2018		1		58
2009		2	2	25	2019		2	1	61
					Total	**1**	**27**	**33**	**61**

BC book chapter, *C* conference, *J* journal

Table 6 Items which contained at least one key characteristic able to answer the RQ

Year	Number of items	Accumulated	Year	Number of items	Accumulated
2003	1	1	2011	2	14
2004	1	2	2012	4	18
2006	1	3	2013	3	21
2007	1	4	2014	3	24
2008	2	6	2015	3	27
2009	3	9	2017	1	28
2010	**3**	**12**	**2019**	**2**	**30**

When an item fills an aspect (i.e., it has associated contributions), it will imply that the level of answer (i.e., rate) to the research questions will be higher than before. Thus, each time that an item has contributions related to an aspect, the rate will be increased in 0.25, being the minimum value 0.25 (because at least one is required to be considered in Table 6), while the maximum level is 1 when an item simultaneously satisfies the four defined aspects (i.e., it has simultaneously contributions for the four aspects). Table 7 outlines the items who reached a rate of 0.75 or upper, while the remaining are synthesized after the table in the proper order based on the rate.

The medium rate (i.e., 0.5) implies that each work simultaneously satisfies two aspects of four related to the research questions. The aspects' detected combinations were SDDP-RS, RT-SDDP, and SDDP-DM. Under the combination SDDP-RS, the contributions and works are (i) a methodology is proposed for describing graph-based data as a topos category. In terms of SDDP, it indicates graph-based data allow new ways for processing and analyzing relations between data. While in terms of RS, it identifies the topos associated with the concept definition and those involved in the definition of other concepts. This allows reasoning on concepts but also on their relationships in order to derive new concepts [22]; (ii) an architecture for discovering data semantics from human-readable documents without previous specification is introduced. In terms of SDDP, the automatic identification of files jointly with the content processing is based on the underlying ontology. While in terms of RS, the process contains an ontology of domain-specific concepts, which allow to answer queries from an abstraction layer that contains the extracted data [23]; and (iii) a query's recommending strategy is introduced based on ontologies for iteratively orienting users. In terms of SDDP, the queries and associated results are guided through a domain ontology, which tries to describe the searched and underlying concepts in order to improve accuracy. While in terms of RS, recommending's strategy pretends help users debugging the query, in those cases in which they do not receive results [24].

Under the combination SDDP-RT, the contributions and works are as follows: (i) A middleware platform oriented to the distribution of semantic data streams over HTTP called Ztreamy is introduced. In terms of RT, it focuses on semantically annotated data streams from the Web and the possibility associated with their distribution and processing. While in terms of SDDP, the data can be filtered based on the semantics

Table 7 Synthesis by item from the perspective of research questions limited to rates equal or upper than 0.75

Authors	Zhang et al. [17]	Rate: 1.0
Aim	Improve the semantic description in time series databases (TSDB) through ontologies and tagging	
RT	Authors proposed ontologies for modeling the semantics of the system, metric, and measurement unit thinking in the IoT environment as a data collecting strategy	
SDDP	The data processing is guided through the tags which are based on an ontological definition (e.g., the source sensor, the relationship between sensors and entities, etc.)	
DM	The queries from users are supported by the metadata based on the underlying ontology	
RS	They incorporate a reasoning engine in a tool suite named SE-TSDB which is able to derive new data streams based on the previous ones	
Authors	Volkovs et al. [18]	Rate: 0.75
Aim	A data-cleaning framework is introduced with the aim of repairing inconsistencies derived from the data structure but also from its semantic	
RT	No explicitly indicated. Authors refer to continuous data cleaning, but it is not really real-time data processing. They allude to a continuous and cyclic repairing process but that is not associated with partial processing or data stream processing	
SDDP	The data history is used for future analysis of data and its associated behavior	
DM	The classifier proposes the kind of likely repairing in order to solve semantic issues on data	
RS	It uses a logistic regression for modeling a classifier which allows recommending the kind of repairing to be made in order to solve inconsistencies in data	
Authors	Wang et al. [19]	Rate: 0.75
Aim	A distributed knowledge acquisition architecture is introduced in the context of the knowledge discovery in databases (KDD)	
RT	No explicitly indicated	

(continued)

Table 7 (continued)

Authors	Zhang et al. [17]	Rate: 1.0
SDDP	The semantic web technology is proposed for giving meaning to spread-out and heterogeneous data sites as a way of integrating them under the umbrella of the semantic grid	
DM	As an evolution of the original data warehouse concept, the authors claim that the semantic grid in this architecture will support the decision-making process in a consistent way because the data semantic is incorporated to deal with heterogeneity	
RS	The works indicate a component related to ontology reasoning and some way in which new knowledge could be derived from the previous ones. However, there is no precision about that	
Authors	Gu et al. [20, 21]	Rate: 0.75
Aim	A peer-to-peer network model supported by two semantic layers is introduced	
RT	A simulation describes the way in which a query could be routed and processed in a dynamical peer-to-peer network	
SDDP	The proposal incorporates two layers. The first one is focused on representing the context as grouped RDF documents (i.e., clusters), while the second one is oriented to represent a structured P2P network based on each semantic cluster	
DM	Semantic groups are used for guiding the query strategy and analyzing the network structure when data need to be reached	
RS	No explicitly indicated	

through provided services in the architecture, keeping the edge on the scalability [25]; (ii) it introduces an indexing method which considers the relative importance of data in data streams for answering queries. In terms of RT, it processes an unbounded data sequence for supporting the online query strategy. While in terms of SDDP, an Importance-aware Bloom Filter (IBF) is proposed to manage queries on data streams based on their data semantics [26]; (iii) a protocol based on data semantics from wireless sensor networks is introduced. In terms of RT, it is focused on decreasing the end-to-end delay and energy consumption in wireless sensor networks. While in terms of SDDP, the designed protocol delivers data based on its meaning and priority in wireless sensor networks. Important data is delivered before lesser important data [27]; (iv) a collaborative strategy oriented to foster the members' participation based on semantic is introduced. In terms of RT, in accordance with the level of member's

participation, each one could receive credits to improve its routing abilities in the members' network when some query needs to be answered. While in terms of SDDP, an ontology-based metadata is employed for describing the data semantics [28]; and (v) a real-time event detection strategy is introduced. In terms of RT, it collects data from heterogeneous sensors in real time. In addition, operations on the collected data (e.g., correlation analysis) can be carried out. While in terms of SDDP, the confidence functions available in the described platform are designed considering its data semantics [29].

Under the combination of SDDP-DM, just one paper was found. As a contribution, a data leak prevention model is introduced in order to classify data based on its semantics. In terms of SDDP, using the function term frequency—inverse document frequency (TF-IDF) jointly with statistical analysis, the classifier is trained for detecting the level of sensibility of each document for avoiding the data leak. While in terms of DM, the model will classify documents based on its semantic [30].

The lowest rate (i.e., 0.25, just one aspect receives contributions) is basically associated with two families of works which have just one common aspect in relation to the research questions: (i) Ontology and tagging family: It is specifically oriented to represent the knowledge through ontologies or tags with an underlying meaning; and (ii) relationships' family: It is focused on works which analyze the relationships through a graph, or alternatively, represent the meaning by mean of a graph. Works under both families do not incorporate in the treatment aspects related to the real-time data processing, support the decision making, and/or strategies linked to recommender systems.

Under the ontology and tagging family was grouped a set of works with the following contributions: (i) A tagging-based temporal conceptual model is introduced to aggregate the semantics related to a conceptual model [31]; (ii) a framework oriented to define a consistent semantic observational model is introduced, which uses an annotation language as a way to guide the data understanding and processing in a heterogeneous data sources' environment [32, 33]; (iii) an introduction on applicability and usability of the semantic decision table is outlined. It uses an ambient computing environment based on ontologies, exposing how semantics could be managed through semantic decision tables [34]; (iv) an anonymization method on categorical attributes is introduced [35, 36]; (v) a method for establishing correspondences between protein data sources through semantic relationships is explained [37]; and (vi) an approach which uses high-level data is used for capturing the data semantic. Taxonomies are used for modeling the data semantic in the high-level data. Next, algorithms based on the attributes' induction are employed for generalizing knowledge [38].

Under the relationships' family, a set of works with the following contributions was gathered: (i) An approach for correcting intra-column anomalies is introduced in big data [39]; (ii) algorithms for graph and subgraph similarity search queries are introduced, jointly with a performance analysis [40, 41]; (iii) it proposes a mapping schema able to be used on nominal variables along with data anonymization process [42]; (iv) an approach for detecting the underlying semantic related

to an arbitrary XML document is outlined [43]; (v) a strategy for generating synopses based on contained semantics in relational databases is shown [44]; (vi) a synthesis of algorithms oriented to distributed, autonomous, and heterogeneous data sources is introduced [45]; and (vii) a method for translating from unstructured data to information table based on ontological graphs is outlined [46].

5 Conclusions

The application of a systematic mapping study on data semantics was shown. The collecting method used "data semantics" as the main term, jointly with "*data-meaning modeling,*" "*data meaning,*" "*data-meaning strategy,*" or "*data-meaning processing*" as alternative terms. Thus, it was searched for those strategies responsible for modeling the data meaning and its application in the real-time data processing, bounded to data stream environments. In addition, aspects related to the real-time data processing jointly with recommender systems were considered. The initial results returned 650 documents, which were filtered considering the document type (i.e., only book chapter, article, and conferences), language (i.e., English), field (i.e., computer science), and focusing on "data semantics" in the keywords. Such filtering returned 151 documents, which were read for analyzing whether they really fit the terms, or they are only mentioned. Accordingly, 61 documents were obtained once the individual reading was ended. In order to satisfy the research questions and motivations introduced in Table 1, just those items from the 61 that included at least one of the following characteristics were retained (i) "*Data processing*": real-time processing, semantic-driven data processing, or decision making; (ii) "*application*": semantic-driven data processing or decision making; and (iii) "*use of knowledge*": decision making or recommender systems. Finally, 30 documents were retained and ready for deep analysis.

With the aim of analyzing different perspectives related to the research questions, four aspects were defined based on documents: real-time data processing (RT), semantic-driven data processing (SDDP), decision making (DM), and recommender systems (RS). For scoring the pertinence of each item in terms of research questions, a rate was introduced based on four aspects. The rate varied between 0.25 (i.e., the item falls in just one dimension) and 1 (i.e., the item has a content that simultaneously is related to four dimensions). Each time a document had part of its content related to a given aspect, the rate would be increased in 0.25. The lowest rate (0.25) grouped the works under two families called "ontology and tagging" and "relationships" with eight items each one focused on explaining the meaning through ontologies/tagging, or well, inferring semantics from the relationships, respectively.

The medium rate (i.e., 0.5) was characterized through the interaction of the following aspects: (i) SDDP-RS: three works which introduced the use of semantic-driven data processing jointly with recommender systems, (ii) SDDP-RT: five works oriented to the joint development of the semantic-driven data processing on real-time

context, and (iii) SDDP-DM: one work focused on the semantic-driven decision making along with the decision-making process.

Based on our criteria and study's results, in order to analyze this kind of subject under the indicated restrictions, the recommended options imply Zhang et al. [17], Volkovs et al. [18], Wang et al. [19], Gu et al. [20, 21] in the given order. The mentioned works simultaneously satisfy at least three aspects in consonance with research questions. This exposes the number of limited and specific contributions in the field, and being currently a challenge due to areas such as Internet of Thing brings itself a set of complexities and opportunities able to be exploited.

As future work, an overhead analysis related to data semantics on real-time systems using descriptions based on ontologies and tagging will be carry forward.

References

1. Sudhindra S, Ganesh LS, Arshinder K (2017) Knowledge transfer: an information theory perspective. Knowl Manag Res Pract 15(3):400–412
2. Divan MJ (2017) Data-driven decision making. In: 2017 international conference on Infocom technologies and unmanned systems (trends and future directions) (ICTUS), vol 2018, pp 50–56
3. Rejikumar G, Aswathy Asokan A, Sreedharan VR (2018) Impact of data-driven decision-making in Lean Six Sigma: an empirical analysis. Total Qual Manag Bus Excell 1–18
4. Silva BN, Diyan M, Han K (2019) Big data analytics. In: SpringerBriefs in computer science, 2019, pp 13–30
5. Kokate U, Deshpande A, Mahalle P, Patil P (2018) Data stream clustering techniques, applications, and models: comparative analysis and discussion. Big Data Cogn Comput 2(4):32
6. Piciu L, Damian A, Tapus N, Simion-Constantinescu A, Dumitrescu B (2018) Deep recommender engine based on efficient product embeddings neural pipeline. In: 2018 17th RoEduNet conference: networking in education and research (RoEduNet), pp 1–6
7. Chovanak T, Kassak O, Kompan M, Bielikova M (2018) Fast streaming behavioural pattern mining. New Gener Comput 36(4):365–391
8. Alharthi A, Krotov V, Bowman M (2017) Addressing barriers to big data. Bus Horiz 60(3):285–292
9. Neiva FW, David JMN, Braga R, Campos F (2016) Towards pragmatic interoperability to support collaboration: A systematic review and mapping of the literature. Inf Softw Technol 72:137–150
10. Pinto VA, Parreiras FS (2014) Enterprise linked data: a systematic mapping study. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp. 253–262
11. Mohammadi M, Al-Fuqaha A, Sorour S, Guizani M (2018) Deep learning for IoT big data and streaming analytics: a survey. IEEE Commun Surv Tutor 20(4):2923–2960
12. Runeson P, Höst M (2009) Guidelines for conducting and reporting case study research in software engineering. Empir Softw Eng
13. Verner JM, Sampson J, Tosic V, Abu Bakar NA, Kitchenham BA (2009) Guidelines for industrially-based multiple case studies in software engineering. In: Proceedings of the 2009 3rd international conference on research challenges in information science, RCIS, 2009
14. Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering version 2.3. Engineering
15. Kitchenham BA, Budgen D, Pearl Brereton O (2011) Using mapping studies as the basis for further research—a participant-observer case study. Inf Softw Technol

16. Petersen K, Vakkalanka S, Kuzniarz L (2015) Guidelines for conducting systematic mapping studies in software engineering: an update. In: Information and software technology, 2015
17. Zhang S, Zeng W, Yen I-L, Bastani FB (2019) Semantically enhanced time series databases in IoT-edge-cloud infrastructure. In: 2019 IEEE 19th international symposium on high assurance systems engineering (HASE), pp 25–32
18. Volkovs M, Chiang F, Szlichta J, Miller RJ (2014) Continuous data cleaning. In: 2014 IEEE 30th international conference on data engineering, pp 244–255
19. Wang H, Nie G, Fu K (2009) Distributed knowledge acquisition based on semantic grid. In: Proceedings—2009 Asia-Pacific conference on information processing, APCIP 2009
20. Gu T, Zhang D, Pung HK (2009) An ontology-based P2P network for semantic search. Int J Grid High Perform Comput 1(4):26–39
21. Gu T, Zhang D, Pung HK (2007) A two-tier semantic overlay network for P2P search. In: Proceedings of the international conference on parallel and distributed systems—ICPADS
22. Reformat MZ, Daniello G, Gaeta M (2018) Knowledge graphs, category theory and signatures. In: 2018 IEEE/WIC/ACM international conference on web intelligence (WI), pp 480–487
23. Balduccini M, Kushner S, Speck J (2015) Ontology-driven data semantics discovery for cybersecurity. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp 1–16
24. Ines F, Habib O (2012) An ontological approach for SQL query expansion. In: 2012 international conference on information technology and E-services, pp 1–6
25. Arias Fisteus J, Fernández García N, Sánchez Fernández L, Fuentes-Lorenzo D (2014) Ztreamy: a middleware for publishing semantic streams on the web. J Web Semant 25:16–23
26. Bhoraskar R, Gabale V, Kulkarni P, Kulkarni D (2013) Importance-aware Bloom Filter for managing set membership queries on streaming data. In: 2013 5th international conference on communication systems and networks, COMSNETS
27. Oh J, Kang KD, Kim JY, Gofman MI (2010) A cross-layer approach to reducing delay and energy consumption based on data importance in sensor networks. In: Handbook on sensor networks
28. Chen W, Wang CL, Lau FCM (2004) A collaborative and semantic data management framework for ubiquitous computing environment. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)
29. Li S, Lin Y, Son SH, Stankovic JA, Wei Y (2004) Event detection services using data service middleware in distributed sensor networks. Telecommun Syst 26(2–4):351–368
30. Alneyadi S, Sithirasenan E, Muthukkumarasamy V (2015) Detecting data semantic: a data leakage prevention approach. In: Proceedings—14th IEEE international conference on trust, security and privacy in computing and communications, TrustCom
31. Khatri V, Ram S, Snodgrass RT, Terenziani P (2014) Capturing telic/atelic temporal data semantics: generalizing conventional conceptual models. IEEE Trans Knowl Data Eng
32. Cao H, Bowers S, Schildhauer MP (2012) Database support for enabling data-discovery queries over semantically-annotated observational data. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp 198–228
33. Cao H, Bowers S, Schildhauer MP (2011) Approaches for semantically annotating and discovering scientific observational data. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp 526–541
34. Tang Y (2010) Towards using semantic decision tables for organizing data semantics. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)
35. Martínez S, Sánchez D, Valls A (2010) Ontology-based anonymization of categorical values. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp 243–254
36. Martínez S, Sánchez D, Valls A (2012) Towards k-anonymous non-numerical data via semantic resampling. In: Communications in computer and information science, pp 519–528

37. Sidhu AS, Dillon TS, Chang E (2006) Towards semantic interoperability of protein data sources. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)
38. Muyeba MK, Khan MS, Warnars S, Keane J (2011) A framework to mine high-level emerging patterns by attribute-oriented induction. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp 170–177
39. Zaidi H, Pollet Y, Boufarès F, Kraiem N (2015) Semantic of data dependencies to improve the data quality. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), pp 53–61
40. Zhao X, Xiao C, Lin X, Wang W, Ishikawa Y (2013) Efficient processing of graph similarity queries with edit distance constraints. VLDB J. 22(6):727–752
41. Zhao X, Xiao C, Lin X, Wang W (2012) Efficient graph similarity joins with edit distance constraints. In: 2012 IEEE 28th international conference on data engineering, pp 834–845
42. Domingo-Ferrer J, Sánchez D, Rufian-Torrell G (2013) Anonymization of nominal data based on semantic marginality. Inf Sci (Ny) 242:35–48
43. Shiu H, Fong J (2009) Reverse engineering from an XML document into an extended DTD graph. J Database Manag
44. Kantere V, Politou ME, Sellis T (2008) Conceptual synopses of semantics in social networks sharing structured data. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 5332 LNCS, No. PART 2, pp 1367–1384
45. Honavar V, Caragea D (2008) Towards semantics-enabled distributed infrastructure for knowledge acquisition. In: AAAI spring symposium—technical report, vol SS-08-05, pp 29–35
46. Pancerz K, Grochowalski P (2017) From unstructured data included in real-estate listings to information systems over ontological graphs. In: Proceedings of the international conference on information and digital technologies, IDT 2017

Chapter 4
Tracking an Object Using Traditional MS (Mean Shift) and CBWH MS (Mean Shift) Algorithm with Kalman Filter

Sandeep Kumar, Rohit Raja, and Archana Gandham

1 Introduction

Tracking can be referred to as a task in order to generate the trajectories of the objects which are moving and compute the motion of sequenced images. Numerous approaches are proposed for translating an object in a sequence of frames, MS is a common approach to perform the task for tracking an object. It is easy in implementing and robustly tracks the performance [1]. MS algorithm compares the target model with the current frame to obtain the region of an object which is selected. It is hard to deal with occlusion in the object and loss of an object in the frame [1]. So, MS procedure has to be improved by using a Kalman filter. Kalman filter estimates active systems state, though the exact form is not known. Other limitations of the MS approach are subjected to local minima where few features of the target are presented in the backdrop. BWH is implemented to reduce the backdrop interference which represents the target. But unfortunately the transformation formula is incorrect and BWH is similar to MS tracking with the usual representation of the target [1–4]. To achieve the improved target localization CBWH MS algorithm is implemented which is obtained by not changing the target candidate model but the target model in the frames. The advantage of CBWH is, in spite of having much information in backdrop CBWH can work robustly. An object position is tracked by using Kalman Filter and that position is observed by MS algorithm. Kalman filter composed with

S. Kumar · A. Gandham
Sreyas Institute of Engineering and Technology, Hyderabad, Telangana, India
e-mail: er.sandeepsahratia@gmail.com

A. Gandham
e-mail: archana.gandham121@gmail.com

R. Raja (✉)
IT Department, Guru Ghasidas Vishwavidyalaya (A Central University), Bilaspur, Chhattisgarh, India
e-mail: drrohitraja1982@gmail.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_4

a set of equations which are mathematical formulations that results as an effective computational work for estimating the particular state in several aspects in processing [5–8]. It consists of two groups: Time and measurement update equations, for projecting the progressive and ongoing state and error covariance measurement, are estimated and to extract the prior estimation of next step time updating equations are used, which gives optimal solutions [1, 9, 10]. Initially, an overview of the traditional MS algorithm is provided and further, the CBWH scheme is introduced in detail. Finally, it explains the fundamentals of KF using the formula and describes the algorithm by proposing the method of tracking an object using the CBWH algorithm and KF [4, 11–13].

2 Literature Survey

Wen and Cai presented an MS algorithm with Gaussian is studied and applied for tracking an object. A convergence theorem and proofs are provided [1]. From the experiment, an object is found even in the presence of occlusions in Fig. 1 [14–18].

Jeyakar and Babu presented MS algorithm has been proved as an efficient algorithm for tracking an object in a video sequence. The author proposed a robust tracking algorithm which overcomes the drawbacks presented in the color histogram-based tracking [9]. From Fig. 2, it is shown that multi-fragment representation of the target and candidate models is used to increase the robustness of object tracking [19–21].

Comaniciu et al. presented a new method is obtained toward representing the target and localization, the tracking of non-rigid objects is proposed visually by using a central component as shown in Fig. 3 [22].

Yang et al. presented a new algorithm is proposed for tracking an object through color as a feature in a complex environment. In order to find a location of an object, an iterative procedure is followed and to improve the MS algorithm a CBWH method is introduced to decrease the interference of the backdrop in target localization [23]. Based on the experiment, tracking of an object is not influenced by the changes in the scale and less subjected to clutter.

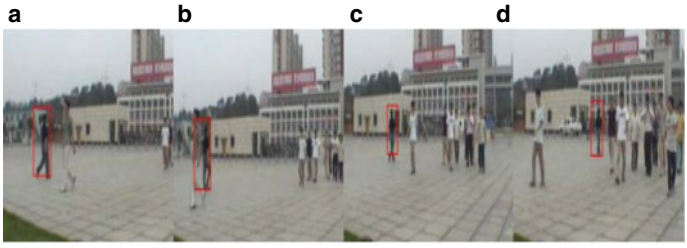


Fig. 1 Results are with the various frames **a** 20 frame, **b** 30 frame, **c** 70 frame, **d** 120 frame

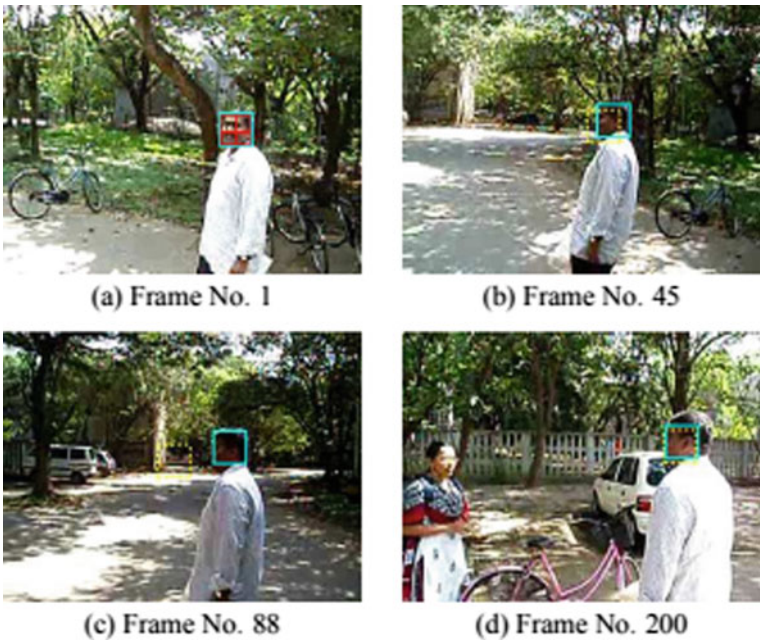


Fig. 2 Robustness to illumination change is shown in the above figure

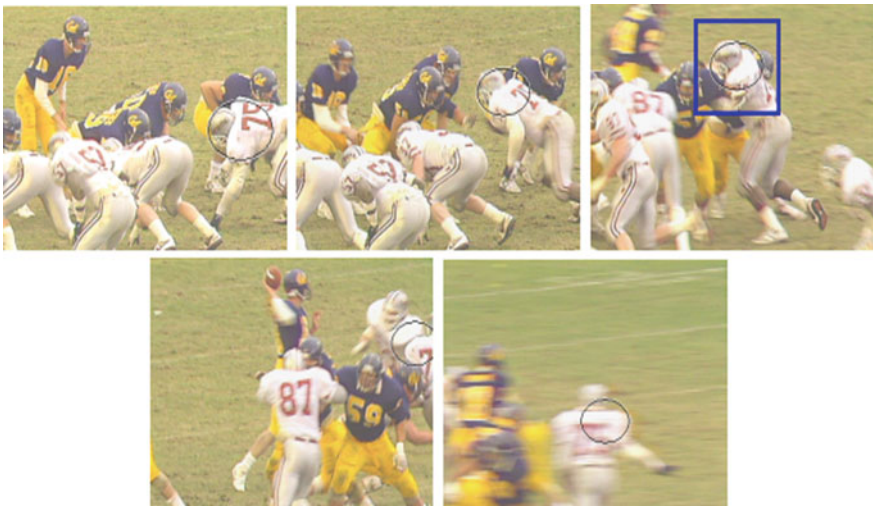


Fig. 3 Football sequenced video frames are used to extract the tracking results

Ning et al. presented an improved MS algorithm is introduced namely BWH which is put forwarded for reducing the interference of backdrop in the frame but it does not introduce any new improvement and it is similar to MS. Therefore, CBWH is proposed by changing target model only and refrains the changes in target candidate model [10]. The CBWH algorithm recognizes the target which is more reliable and accurate. It also achieves small errors and standard deviation than BWH MS algorithm.

3 Proposed Methodology

In order to have robust tracking, the mean-shift algorithm will be an effective approach for tracking the objects whose appearance is bounded by histograms. BWH- and CBWH-based MS tracking is implemented by decreasing the involvement of background in target localization. The following objectives are formulated for achieving the CBWH and BWH MS tracking algorithm.

- To study and analyze the conventional robust object tracking algorithms.
- To implement existing mean-shift tracking algorithm.
- To implement CBWH- and BWH-based MS tracking algorithms.

To compare the CBWH- and BWH-based MS tracking algorithms with existing mean-shift tracking algorithms. Tracking with BWH and CBWH MS algorithms. In this methodology, the BWH algorithm transforms both the candidate model and target model but does not actually decrease the interference of background features to amend localization of the target. CBWH algorithm is then introduced to overcome the issues by not changing the candidate model but changing the target model only. The basic goal of this CBWH algorithm is to achieve improved target localization by reducing the interference of background as shown in Fig. 4. MS algorithm, the Kernel-based deterministic procedure, is an iteration of MS tracking which converges the local maximum function of measurement with assumptions of behavior on the kernel. It is a less complicated algorithm; it gives a reliable and general solution to track an object and independent in representing the target.

Let $\{y_i^*\}_{i=1\ldots n}$ is denoted for the position which is normalized of a pixel which is present in the region of target, then it is a center in origin point. $\hat{q}$ is a target model which is corresponded to the region of target which is performed is described below:

$$\hat{q} = \{\hat{q}_v\}_{v=1\ldots m} \tag{1}$$

$$\hat{q}_v = C \sum_{(i=1)}^{n} k\left(\left\|y_i^*\right\|^2\right)\delta\left(f\left(y_i^*\right) - v\right) \tag{2}$$

Here, $\hat{q}_v$ is represented to the probability feature of vin target model $\hat{q}$, feature spaces are denoted as m, Kronecker delta function is denoted as (δ), $f\left(y_i^*\right)$ associates

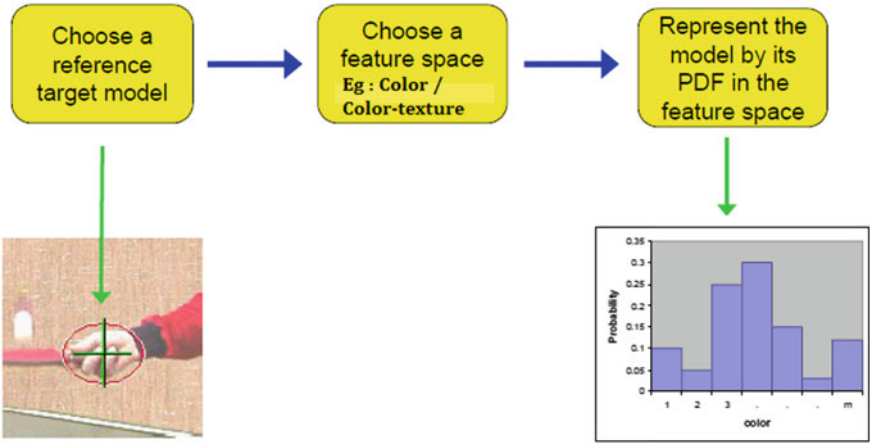


Fig. 4 Representing the target in MS algorithm

the pixel to the histogram bin, an isotropic kernel profile is $k(y)$ and C is constant which is a normalization function defined by

$$C = \frac{1}{\sum_{i=1}^{n} k\left(\left\|y_i^*\right\|^2\right)} \tag{3}$$

The representation of color feature space 'v' in the MS algorithm is clearly shown in the above picture of the target model. The candidate region corresponds to the target candidate model $\hat{p}_v(x)$ which is given by

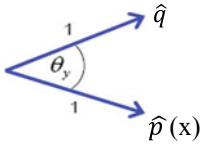
$$\hat{p}(x) = \left\{\hat{p}_v(x)\right\}_{v=1\ldots m} \tag{4}$$

$$\hat{p}_v(x) = C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{x - y_i}{h}\right\|^2\right)\delta(b(y_i) - v) \tag{5}$$

where

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k\left(\left\|\frac{y - X_i}{h}\right\|^2\right)} \tag{6}$$

where $\hat{p}_v(x)$ represented as the feature of probability v in the candidate model. The pixel positions are denoted as, $\hat{p}_v(x)$, $\left\{y_i^*\right\}_{i=1\ldots n}$ in the target candidate region which is centered at x, h is denoted as bandwidth and constant C_h is a normalization function. For calculating the target model and the candidate model, a Bhattacharyya coefficient is derived between the two normalized histograms $\hat{q}$ and $\hat{p}(x)$ as follows:



$$\rho\left[\hat{p}(x), \hat{q}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_v(x)\hat{q}_v} \tag{7}$$

Calculating the distance in between $\hat{p}(x)$ and $\hat{q}$ is defined as

$$d(x) = \sqrt{1 - \rho\left[\hat{p}(x), \hat{q}\right]} \tag{8}$$

3.1 Tracking the MS

The issue of tracking an algorithm is an offset computation form where it moves from the current location x to a new location x_1 based on iteration equation.

$$x_1 = \frac{\sum_{i=1}^{n_h} x_i w_i g\left(\left\| \frac{x - y_i}{h} \right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\| \frac{x - y_i}{h} \right\|^2\right)} \tag{9}$$

where

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_v}{\hat{p}_v(x_0)} \delta[b(y_i) - v]} \tag{10}$$

Choose the kernel within the Epanechnikov profile, (9) is decreased to

$$x_1 = \frac{\sum_{i=1}^{n_h} y_i w_i}{\sum_{i=1}^{n_h} w_i} \tag{11}$$

By using (Eq. 11), the algorithm tracks to finds the new frame which is similar to a region of an object. The target localization is shown briefly in Fig. 5.

Performance, the target localization accuracies for this algorithm are mentioned below and their diagrammatic representations are shown in Fig. 6.

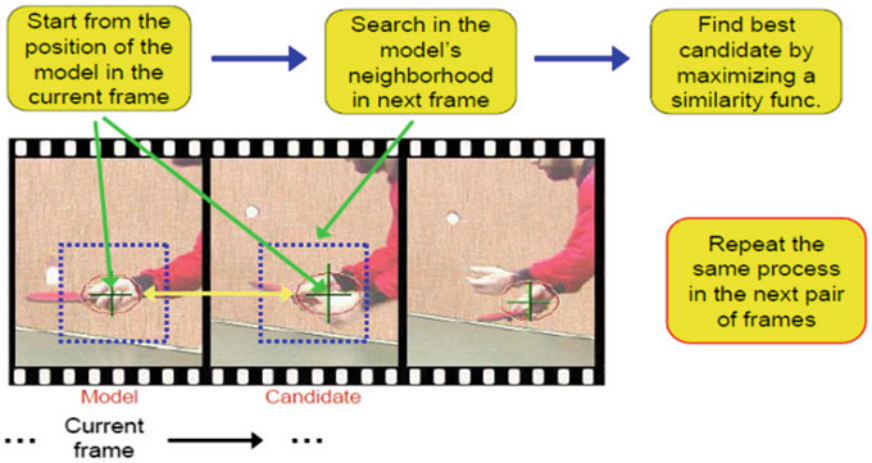


Fig. 5 Localization of target in MS tracking

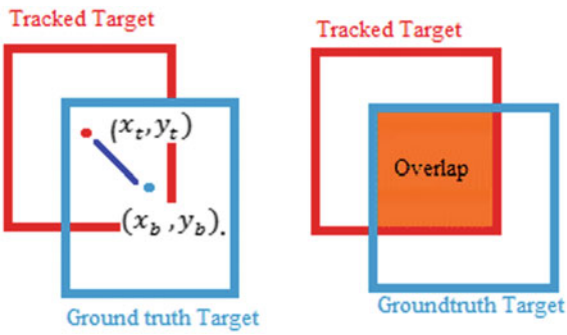


Fig. 6 Position of the tracker (red) and its associated ground truth bounding box (blue). Centroid distance is described by dark blue line. Overlap is represented by orange

3.2 Normalized Centroid Distance (NCD)

Normalized centroid distance (NCD) for a tracker centered at (x_t, y_t) and a ground truth bounding box with center (x_b, y_b). The NCD in terms of the height (h_b) and the width (w_b) of the bounding box is given as

$$\text{NCD} = \left(\frac{x_t - x_b}{w_b}\right)^2 + \left(\frac{y_t - y_b}{h_b}\right)^2 \tag{12}$$

Overlap, the useful measure of tracker accuracy is part of the ground truth bounding box is occupied in a given frame, which is referred to as overlap.

$$\text{Overlap} = \frac{\text{area}_{\text{Common}}}{\text{area}_{\text{bounding\_box}}} \tag{13}$$

3.3 BWH MS Tracking

BWH MS tracking, to decrease the interference in target localization which has salient background features, Comaniciu et al. proposed a model for representing the background features and discriminative features are selected from the target candidate region and the target region. In the background, the histogram is expressed as $\{\hat{o}_v\}_{v=1\ldots m}$ and surrounding area of target is calculated by the algorithm. The region present in the backdrop is considered as twice the size of the target. The value of non-zero is minimal and represented as $\{\hat{o}_v\}_{v=1\ldots m}$ is denoted by $\hat{o}^*$. To define the changes in between the representations of the candidate model and target model, a coefficient is used and given below.

$$v_u = \left\{ \frac{\min(\hat{o}^*)}{\hat{o}_u}, 1 \right\}_{u=1\ldots m} \tag{14}$$

This transformation will make the weights less of these features with low v_u, i.e., which are the features of background. Then, the new model of the target is defined as:

$$\hat{q}_v^{'} = C' u_v \sum_{i=1}^{n} k\left(\left\| y_i^* \right\|^2\right) \delta\left(b\left(y_i^*\right) - v\right) \tag{15}$$

where

$$C' = \frac{1}{\sum_{i=1}^{n} k\left(\|y_i^*\|^2\right) \sum_{u=1}^{m} u_v \delta\left(b\left(y_i^*\right) - v\right)} \tag{16}$$

The new target candidate model is:

$$\hat{p}_u^{'}(x) = C_h^{'} u_v \sum_{i=1}^{n_h} k\left(\left\| \frac{x - y_i}{h} \right\|^2\right) \delta(b(y_i) - v) \tag{17}$$

where

$$C_h^{'} = \frac{1}{\sum_{i=1}^{n_h} k\left(\left\| \frac{x-y_i}{h} \right\|^2\right) \sum_{u=1}^{m} u_v \delta(b(y_i) - v)} \tag{18}$$

The above algorithm is used for reducing the effect of features present in the target candidate model which is present in the target localization.

3.4 Similarities for Representing the BWH with Usual Representation

In the target candidate region, the weights of points determine the convergence by using the iteration formula in tracking an algorithm, only by reducing the weights in the features of backdrop will decrease the information in the backdrop in target localization relevantly. For evaluating the changes in weights of points y_i which is computed by the target candidate region in BWH, $w_i^{'}$ denotes the weight of a point computed in the target candidate region by the BWH.

$$w_i^{'} - \sum_{N=1}^{m} \sqrt{\frac{\hat{q}}{\hat{p}(y)}} \delta[b(y_i) - v] \tag{19}$$

Here, v' be the feature space in the bin index which corresponds to point y_i in the candidate region. We have $\delta\big(b(y_i) - v'\big) = 1$. So Eq. (16) can be simplified as

$$w_i^{'} = \sqrt{\hat{q}_{v'} / \hat{p}_{v'}}(y) \tag{20}$$

Substitute Eqs. (14) and (16) into Eq. (19), there is

$$w_i^{'} = \sqrt{\frac{c' u_v^{'} \sum_{j=1}^{n} k(IIyII)_j^2 \delta\left[b\left(y_j^*\right) - v'\right]}{c_h^{'} u_v^{'} \sum_{j=1}^{n_h} k\left(II\frac{x-y_j}{h}II\right)^2 \delta\left[b\left(y_j\right) - v'\right]}} \tag{21}$$

By substituting normal factors C and C_h and removing the common factor $v_{u'}$ from the numerator and into the above equation, we have

$$w_i^{'} = \sqrt{\frac{cc_h}{cc_h} \cdot \frac{c' \sum_{i=1}^{n} k(IIY_t^* II)^2 \delta\left[b(Y_t^*) - v'\right]}{c_h^{'} \sum_{i=1}^{n_h} k\left(II\frac{x-y_t}{h}II\right)^2 \delta[b(y_t) - v']}} = \sqrt{\frac{c'c_h}{cc_h^{'}}} \cdot \sqrt{\frac{\hat{q}_v^{'}}{\hat{p}_v^{'}}} = \sqrt{\frac{c'c_h}{cc_h^{i}}} w_i \tag{22}$$

where w_i calculated by Eq. (10) is the general representation of the target candidate model as the weight of the point and target model. Equation (21) suggests that $w_i^{'}$ is reciprocal to w_i. Moreover, by associating MS iteration Eq. (11) we have

$$y_1 = \frac{\sum_{i=1}^{n_h} y_i g_i w_i^{'}}{\sum_{i=1}^{n_h} g_i w_i^{'}} = \frac{\sum_{i=1}^{n_h} y_i g_i w_i \sqrt{\frac{c'c_h}{cc_h^{'}}}}{\sum_{i=1}^{n_h} w_i g_i \sqrt{\frac{c'c_h}{cc_h^{'}}}} = \frac{\sum_{i=1}^{n_h} y_i g_i w_i}{\sum_{i=1}^{n_h} w_i g_i} \tag{23}$$

Equation (23) is the MS iteration formula is uniform which is the scale transformation of weights is. Therefore, BWH actually does not strengthen MS tracking by

transferring the target model representation and also target candidate model where the results are almost the same as that, without using BWH (Fig. 7).

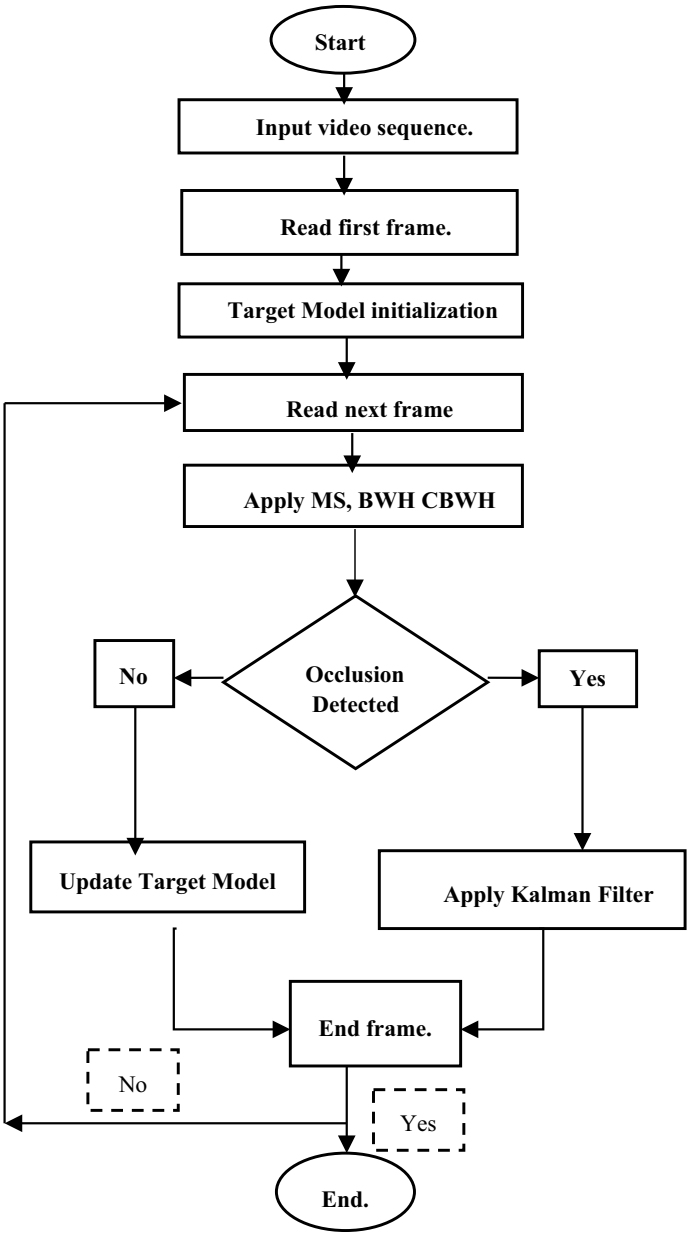


Fig. 7 Flowchart of proposed methodology

3.5 CBWH Algorithm

As it is discussed above the BWH algorithm does not improve any target localization. To achieve that truly a new formula is introduced known as CBWH as shown in Figs. 8 and 9. It is employed to change the only target but not the candidate model. By this, it can reduce the outstanding features of the backdrop in the target region.

New formula,

$$w_i^n = \sqrt{\frac{\hat{q}v}{\hat{p}v}}(x) \tag{24}$$

Therefore,

$$w_i^n = \sqrt{u_v}.w_1 \tag{25}$$

The CBWH MS tracking algorithm is:

Algorithm: CBWH

Input: $\hat{q}$ is the target model calculated and x_0 in previous fame as the location.

(1) A number of iterations are initialized $k \leftarrow 0$.

(2) Calculate the distribution of the target candidate model $\hat{p}$(y), in the current frame

(3) Weights are calculated by using$\{w_i^{"}\}_{i=1.....n_h}$.

(4) New x_1 location of the target candidate model is calculated.

(5) Let $k \leftarrow k+1, d \leftarrow \| x - x_0 \|, x_0 \leftarrow x_1$. Set the maximum iteration number 'N' and the threshold $'\varepsilon'$. (Here threshold is set to 0.1 and maximum iterations are 15)

If$d < \varepsilon$ or $k \geq N$. Calculate $\{o_{v'}\}_{v'=1.....m}$ and $\{u_v\}_{v'=1.....m}$ is obtained based on results of current frame tracking. Stop, go to Step 6 if it not tracked properly. Otherwise go to step 2 for next frame.

(6) The next frame is loaded as the current frame with the initial location x_0 and go to Step 1.

In this methodology, CBWH algorithm achieves improved target localization as this algorithm actually reduces the intrusion of background features in target representation. Whereas the BWH algorithm proved that its target representation is similar to the representation of the target in the framework MS.

Fig. 8 Algorithm of CBWH

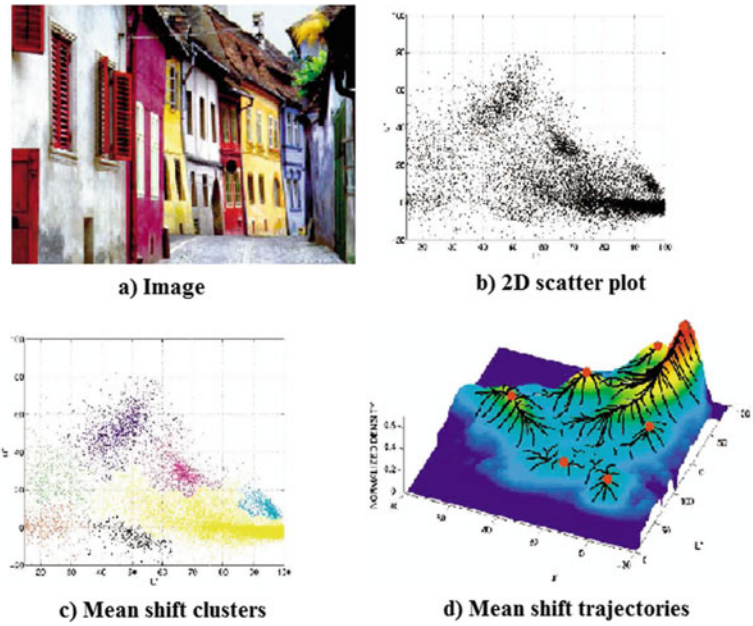


Fig. 9 Feature density example of MS clustering

3.6 Applications

It is a procedure of functional application to find the mode: In the density function, begin from the points of data and conduct mean-shift method in order to the find the stationary points. By holding the local maxima, compress or reduce the points. Area of the attraction of the model describes the set of locations which assembles to the same mode. The points which are available in the same area of attraction are correlated with the same cluster. Below figure shows feature density example of MS clustering.

4 Results

In the analysis of MS tracking, color feature space and the RGB color model feature are used in all the experiments and $16 \times 16 \times 16$ bins are quantized. The results of all the four video sequences are tracked, i.e., skating sequence, cube sequence, smiley sequence, and basketball sequence are shown in Figs. 10, 11, 12 and 13, Table 1 shows that the CBWH model which consists of greater accuracy in localization when compared to BWH model and the usual MS model. Because the CBWH model truly accomplishes the information of background in target localization with an

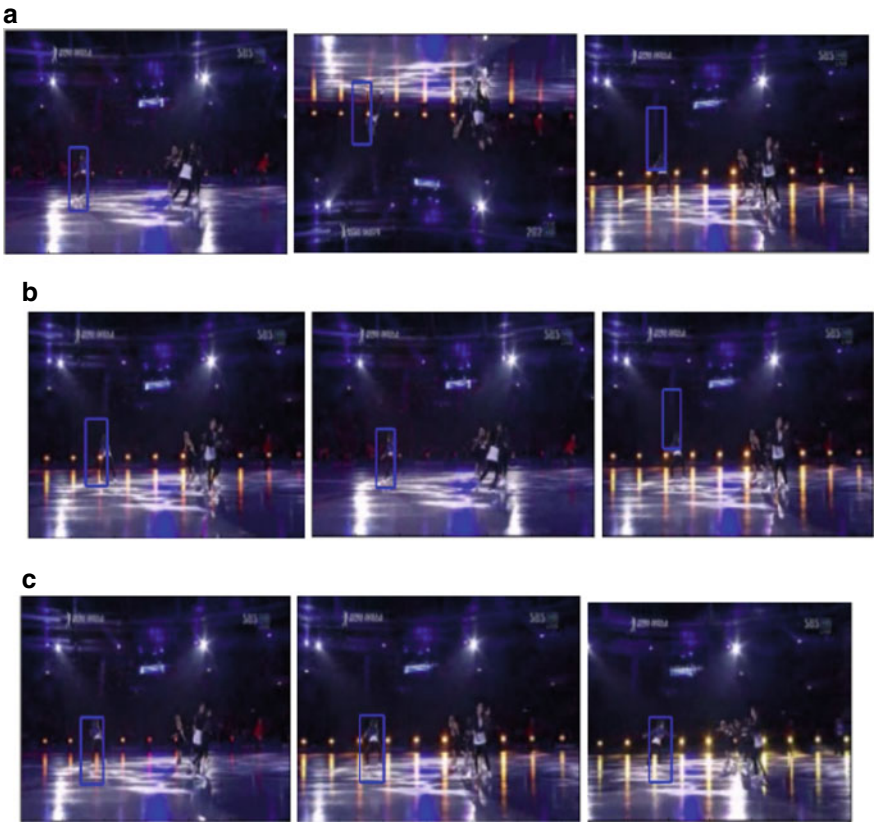


Fig. 10 Comparison of three methods using color feature space for skating sequence. Frames 10, 24, 40 are displayed. From (**c**), CBWH MS tracking method has improved tracking accuracy than BWH and usual MS tracking methods, **a** MS tracking, **b** BWH MS tracking, **c** CBWH MS tracking

error of low centroid distance and high overlap, Figures 14 and 15 show the plots of target localization accuracies. Figure 16 illustrates the numbers of iterations by plots in three methods for basketball sequence of video. Table 1 also shows the average number of iterations by three methods. An average number of iterations is less for CBWH when compared to BWH and the usual MS model which is observed through the images. The noticeable features are improved in the target model while the features of the background are muted in CBWH. By this, the MS algorithm can be located with the greater accuracy as a target. All experiments were done on MATLAB 7.12 version with Image Processing toolbox and Computer vision toolbox with the following specification of computer: Processor: Intel(R) Core(TM), Processor Speed: I3, Operating System: Windows 8, Hard Disk: 910 GB, RAM: 4 GB.

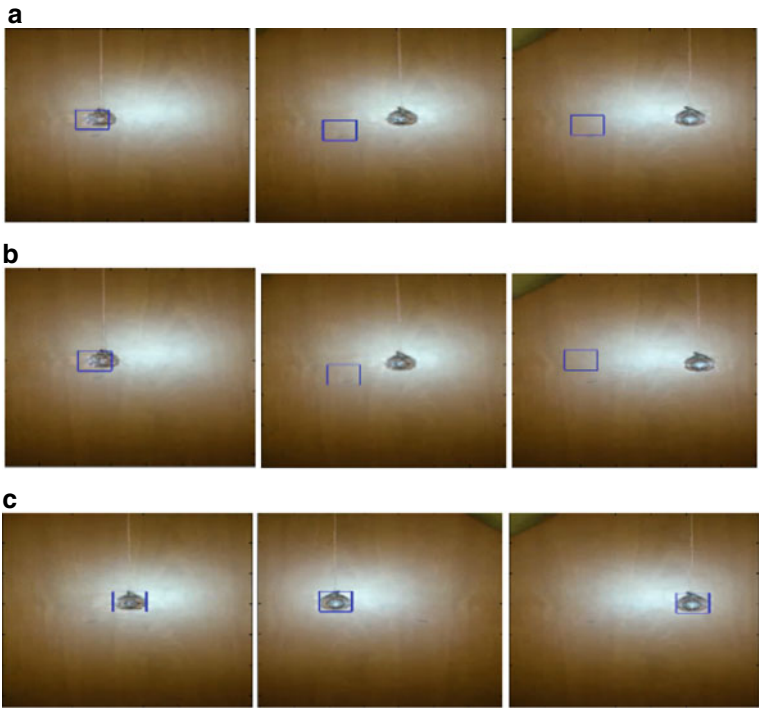


Fig. 11 Comparison of three methods using color feature space for cube sequence. Frames 19, 25, 36 are displayed. From (**c**), CBWH MS tracking method has improved tracking accuracy than BWH and usual MS tracking methods. **a** MS tracking, **b** BWH MS tracking, **c** CBWH MS tracking

5 Conclusion

In the computer vision field, the popular research subject or content is object tracking. It has applications which are necessary for the field of transportation intelligence, defense security, intelligent video surveillance, and robot navigation. There are many algorithms which are used in the literature on object detection and tracking. Histogram of color feature for mean shift is based on the object tracking algorithm which obtains a huge range of applications, due to its simplicity and good real-time performance. In this project, color feature space is analyzed based on tracking of mean shift, in target representation, localization of a target has to be improved and interference of background should be decreased, analyze the BWH and CBWH. The results of the experiments will confirm that CBWH with reduced MS iteration number and also provides 90.12% accuracy in tracking. The advantage of CBWH sensitivity has to be reduced in target initialization for tracking the mean shift, i.e., CBWH will strongly track even though the target is not initialized properly.

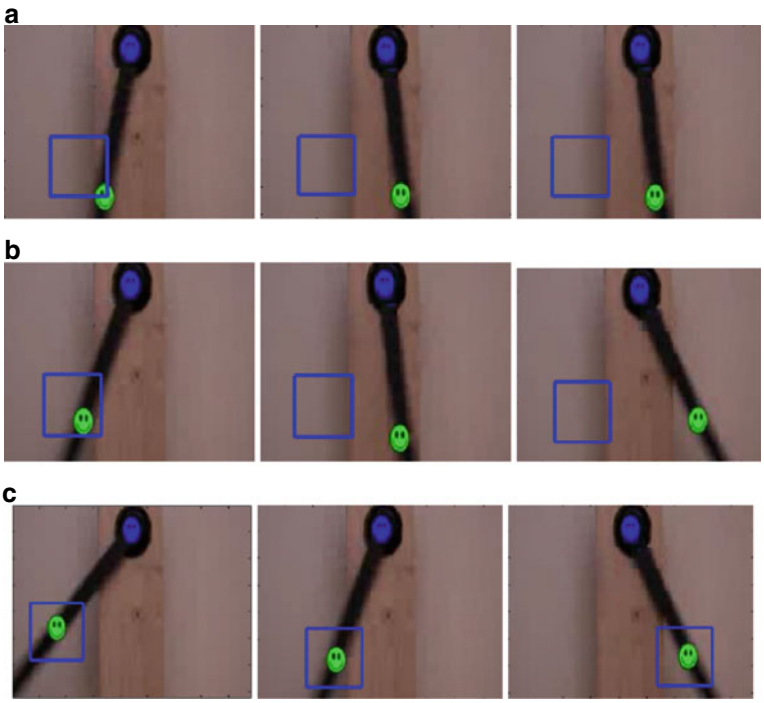


Fig. 12 Comparison of three methods using color feature space for the smiley sequence. Frames 15, 36, 55 are displayed. From (**c**), CBWH MS tracking method has improved tracking accuracy than BWH and usual MS tracking methods. **a** MS tracking, **b** BWH MS tracking, **c** CBWH MS tracking

Concerning the future research, the MS method can be further improved to track the objects using a fuzzy coding histogram to overcome the effects of quantization inherent in fixed bin histogram. At the initial stage of this algorithm, the fuzzy clustering is used on the initial detection region to get the cluster prototypes and regard it as a fuzzy codebook. During the tracking period, the candidate image region is also represented by a histogram which is constructed by the fuzzy memberships. In addition to this, a cumulative distribution function between the fuzzy coding histogram is used to construct a cross-bin metric and then use mean-shift iteration to realize the robust visual tracking. The concept of MS can also be extended for robustness by including tracking of multiple objects, occlusion detection, and scale changes to improve the tracking accuracy.

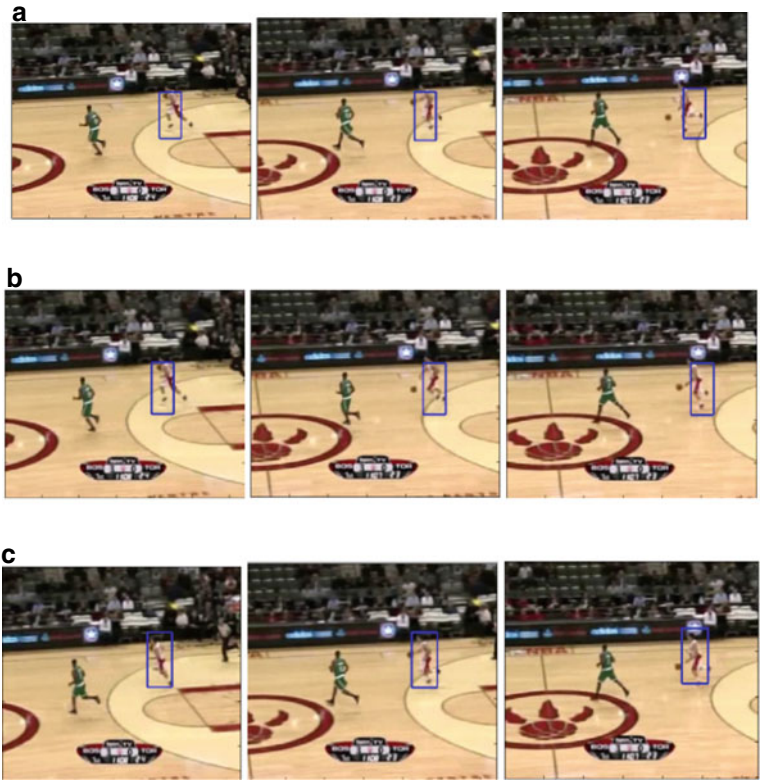


Fig. 13 Comparison of three methods using color feature space for basketball video sequence. Frames 26, 35, 45 are displayed. From (**c**), CBWH MS tracking method has improved tracking accuracy than BWH and usual MS tracking methods. **a** MS tracking, **b** BWH MS tracking, **c** CBWH MS tracking

Table 1 Comparison of the analysis of mean-shift-based algorithms

Videos	Algorithms	NCD	Overlap	Iterations
Skating	MS	2.5207	0.4593	3.01
	BWH	2.3975	0.4593	3.01
	CBWH	0.4169	0.8676	2.51
Cube	MS	2.018	0.5732	7.94
	BWH	2.018	0.5732	7.94
	CBWH	1.2942	0.7124	4.62
Basket ball	MS	0.89	0.8192	3.51
	BWH	0.89	0.8192	3.51
	CBWH	0.5543	0.8777	2.74
Smiley	MS	1.511	0.7273	3.57
	BWH	1.511	0.7273	3.57
	CBWH	0.1348	0.9818	2.86

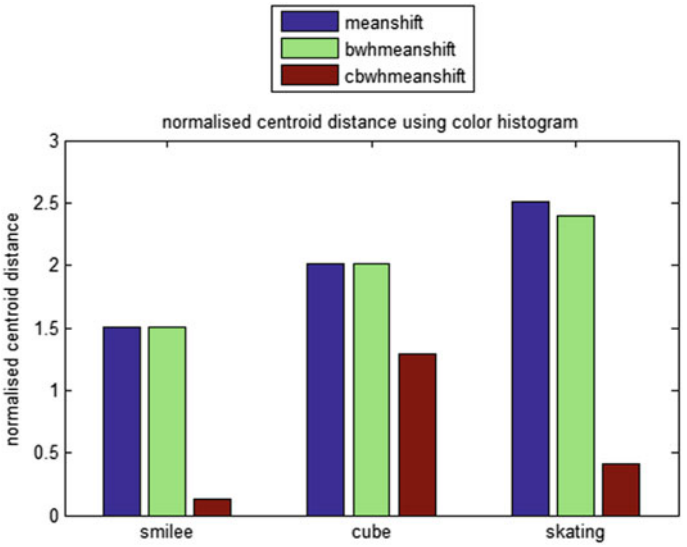


Fig. 14 Plot of normalized centroid distance error versus different video frames using color feature space

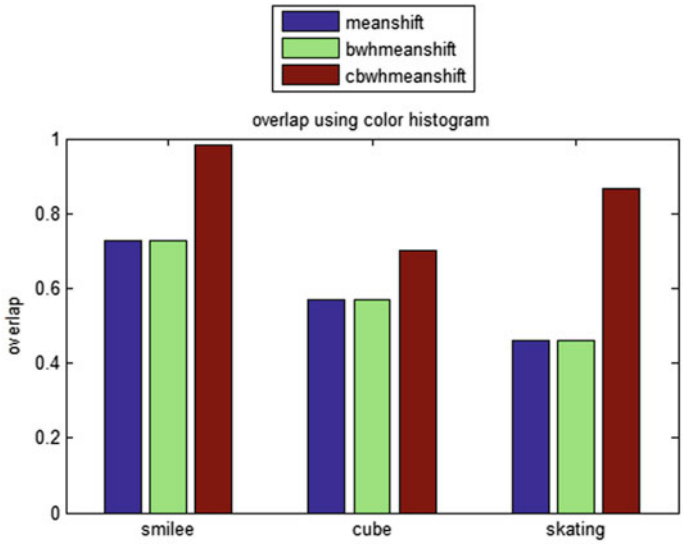


Fig. 15 Plot of overlap versus different video frames using color feature space

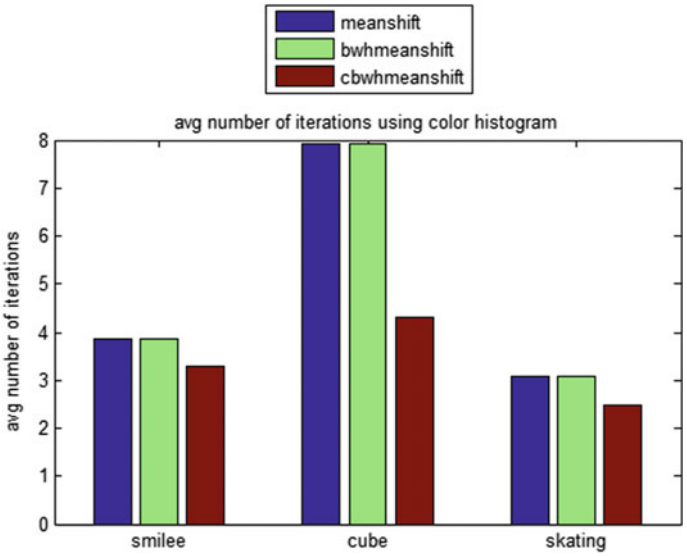


Fig. 16 Plot of average number of iterations versus different video sequences using color feature space

References

1. Wen Z-Q, Cai Z-X (2006) Mean shift algorithm and its applications in tracking of objects. Mach Learn Cybern, pp 13–16
2. Gmez NA (2009) A probabilistic integrated object recognition and tracking framework for video sequences. University at Rovira I Virgili, Ph.D. thesis, Espagne
3. Collins R, Yanxi Liu T, Leordeanu M (2010) Online selection of discriminative tracking features. IEEE Trans Pattern Anal Mach Intell 10:1631–1643
4. Ning J, Zhang L, Zhang D, Wu C (2011) Robust MS tracking with corrected background-weighted histogram to appear in IET computer vision
5. Hsu C-T, Yeh Y-J (2009) Online selection of tracking features using AdaBoost. IEEE Trans Circ Syst Video Technol 3:442–446
6. Nummiaro K, Koller-Meier E, Van Gool LJ (2002) Object tracking with an adaptive color-based particle filter. In Proceedings of the 24th DAGM symposium on pattern recognition, pp 353–360, London, UK. Springer-Verlag
7. Shah M, Saleemi I, Hartung L (2010) Scene understanding by statistical modeling of motion patterns. In: IEEE conference CVPR, pp 2069–2076
8. Ren X, Krainin M, Henry P et al (2011) Manipulator and object tracking for in-hand 3d object modeling. IJRR 30(11):1311–1327
9. Jeyakar J, Babu RV, Ramakrishnan KR (2008) Robust object tracking with background-weighted local kernels. Comput Vision Image Underst 112(3):296–309
10. Ning J, Zhang L, Wei Y, Zhang D (2009) Robust object tracking using joint color—texture histogram'. Patt Recogn Artif Intell 23(07):1245–1263
11. Comaniciu D, Meer P (2002) MS: a robust approach toward feature space analysis. PAMI 24(5):603–619
12. Grabner H, Matas J, Van Gool L et al (2010) Tracking the invisible: learning where the object might be. In: 2010 IEEE conference computer vision and pattern recognition (CVPR), pp 1285–1292

13. Kumar S, Singh S, Kumar J (2017) A study on face recognition techniques with age and gender classification. In: IEEE international conference on computing, communication and automation (ICCCA), pp 1001–1006
14. Comaniciu D, Ramesh V, Meer P (2003) Kernel-based object tracking. IEEE Trans Patt Anal Mach Intell 25(2):564–577
15. Sindhuja G, Renuka Devi SM (2015) A survey on detection and tracking of objects in a video sequence. Int J Eng Res Gen Sci 3(2):418–426
16. Ess A, Schindler K, Leibe B (2010) Object detection and tracking for autonomous navigation in dynamic environments. IJRR 29(14):1707–1725
17. cmp.felk.cvut.cz/~ vojirtom/dataset/,www.iai.unibonn.de/~kleind/tracking,clickdamage.com/…/cv_datasets.php
18. Ling H, Mei X (2009) Robust visual tracking using L_1 minimization. In: 2009 IEEE 12th international conference on computer vision, pp 1436–1443
19. Shi C, Zhou H, Yuan Y (2009) Object tracking using SIFT features and MS. Comput Vision Image Underst 113(3):345–352
20. Ning J, Zhang L, Zhang D et al (2009) Robust object tracking using the joint color-texture histogram. Int J Pattern Recogn Artif Intell 23(07):1245–1263
21. Kumar S, Singh S, Kumar J (2018) Live detection of face using machine learning with multi-feature method in wireless personal communication, pp 1–23. https://doi.org/10.1007/s11277-018-5913-0
22. Comaniciu D, Ramesh V, Meer P (2000) Kernel-based object tracking. Comput Soc Conf Comput Vis Patt Recogn 2(2):142–149
23. Yang Y et al (2013) Object tracking based on corrected background-weighted histogram mean shift and Kalman filter. Adv Mater Res 765–767:720–725

Chapter 5
Transfer Learning and Domain Adaptation for Named-Entity Recognition

Raghul Prakash and Rahul Kumar Dubey

1 Introduction

Transfer learning has shown significant strides in the domain of computer vision; however, its prevalence in the domain of natural language processing has only spiked recently due to novel deep learning architectures that have allowed for sequential data processing. If two or more different tasks have similar variations to be captured, and if they share similar low-level features such as edges and shapes in the case of computer vision [1] or semantic relationships in the case of natural languages, then transfer learning has been found to significantly improve the performance of the model in the target domain with very limited labelled data and training time. Natural language processing is a broad field, and this paper pertains to the field of named-entity recognition (NER).

Named-entity recognition is the foundation for many tasks related to information extraction. It is used to extract named entities in documents such as person, location and organization. There have been a lot of researches done in this field, but the study of the transfer of knowledge across different domains has not been researched upon adequately. The primary challenges in this field are the absence of labelled data in the target domain and the varying types of named-entity tags that are to be assigned. Hand annotating a text corpus with NER tags is highly cumbersome and requires a lot of understanding of a given domain. If sensitive or classified information is involved, then this task of providing labels for texts cannot be crowd sourced. This is a serious problem in NER.

R. Prakash · R. K. Dubey (✉)
Robert Bosch Engineering and Business Solutions Ltd., Bengaluru, Karnataka 560095, India
e-mail: RahulKumar.Dubey@in.bosch.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_5

Transfer learning can help solve this problem by allowing a model trained on ample labelled data in a given domain (source domain) to adapt to a different domain (target domain) even with sparse labelled data. There are different types of transfer learning methods, and this paper pertains to transductive transfer learning in different domains with similar tasks which is also called domain adaptation. If a model is trained extensively on a given domain (source domain) for a given task and the same task is assigned in another domain (target domain) with limited labelled dataset, then with a few shots of training on the labelled dataset in the target domain, the model can adapt to the variations in the target domain with its own characteristic probability distribution of words.

The architecture used to perform NER tagging in this research is a bi-directional stack LSTM neural network with attention. LSTM networks are a type of recurrent neural networks which capture long-term dependencies between the various parts of a sentence. They do so by solving the problem of vanishing gradients which is a major bottleneck in regular recurrent neural networks used bi-directionally because the model can capture patterns in the sequence not only from left to right but also from right to left. This makes it easier for the model to represent complex semantic relationships. Stacked bi-LSTM networks differ from traditional bi-LSTM networks, and in that they learn gradually the representations of the input. Increasing levels of abstractions can be achieved by using stacked bi-LSTMs, with higher layers learning domain-specific representations. Thus, their nature makes it more convenient for transferring the weights learned in the lower layers for the model in the target domain. Just as how transfer learning has become a common practice in image-related applications, with the use of models that support for increasing levels of abstraction which can increase the transfer of knowledge, it is hypothesized that domain-independent representations can be captured for NER in the lower layers of the model. Also, with images, since geometry is predominately hierarchical, there is a definitive correlation between the weights learned by layers and the various levels of hierarchy in images from lines to shapes to objects. However, with texts, such a definitive hierarchy is yet to be fully established in the field of NLP.

Attention models have been used in images in order to allow the neural network to focus on certain parts of the images and spend more time learning valuable representations from specific parts of the image, thus saving computational power. A similar approach is adopted in this paper in order to implement hierarchy in NER. The source dataset is the CoNLL-2003 dataset which is a newspaper corpus. The named entities for this domain are PERS, LOC, ORG, O and MISC. The target dataset is the CSAIL movie corpus. A survey of existing architectures is done, and the drawbacks of previous architectures like regular bi-LSTMS are investigated. A new model is proposed based on the stacked bi-LSTM with an attention model. The target datasets are the CSAIL movie corpus and the WikiGold corpus. The classes in the first corpus are ACTOR, CHARACTER, DIRECTOR, GENRE, PLOT, RATING, RATINGS_AVERAGE, REVIEW, SONG, TITLE, TRAILER, YEAR, O. The classes in the second corpus are PERS, LOC, ORG, MISC, O. Thus, the former domain is a novel class domain as it has different classes than the source domain and the later domain has the same tags as the source domain. The performance of

the model in both these target domains is used to evaluate the transferability of the model to the target domain. The performance of the model in the target domain with various splits of the fine-tuning and testing corpora is also investigated.

2 Related Work

Named-entity recognition is an established field and has been around since 1996, with the first NER model by Grishman and Sundheim. Most of the NER models were predominately rule-based and were not scalable and adaptable to the massive amounts of unstructured text data available these days. Around that time, the need for more sophisticated and robust NER models was emerging. NER models based on feature engineering and machine learning started gaining popularity. These architectures, however, were not able to adapt to different domains and required a lot more understanding of the application domain. However, this approach was soon to change as Collebert et al. laid the foundation for neural network-based NER systems that require minimal feature engineering. With the advent of more computational power, increasing depths of layers can be trained more easily, and this approach is known as Deep learning. This architecture has shown better $F1$ scores in comparison with popular shallow learning techniques like conditional random fields (CRF) when trained on a large amount of labelled data. Deep learning solves a task by learning from the data through increasing levels of abstraction, with lower layers learning general features and upper layers learning more domain-specific features. This approach of solving the problem layer by layer allows for fine-tuning much more effectively; thus, this approach promotes the transfer of knowledge across domains.

Existing work on transfer learning for NER include work done by Lee et al. for transfer learning in the bio-medical domain, Rodriguez et al. for transfer learning of novel classes across various popular corpora and domains like news, spoken query, medical, etc. These models use a bi-LSTM neural network as the foundation of the architecture and add various optimization layers like CRF, etc. They also learn the word embedding jointly along with the task and account for transfer in the word embedding layer as well.

3 Procedure

The first step to any natural language processing task is pre-processing the text corpus. The source domain NER dataset consists of newspaper articles that have been tagged to their corresponding classes. The labelled corpus starts with -DOCSTART- O and each line corresponds to a word, its POS tag, Chunk tag followed by its NER tag, respectively. Each sentence is separated by an empty line.

The corpus is converted to a DataFrame with columns corresponding to the sentence number of each word, the word itself and its NER tag. After that each sentence

based on the sentence number is extracted from the DataFrame and passed to the word embedding tool, fastText. FastText is a character-level embedding tool that converts words to vectors. It can handle unseen vocabulary and is thus beneficial to NER. This paper does not pertain to research on the various types of word embeddings; thus, a consistent character-level embedding is adopted based on surveying various popular embedding tools. This word embedding is pre-trained on both the source domain corpus (CoNLL-2003) and the target domain corpus. Thus, the word embedding layer will be neutral to both the source and target domain. The tags are then converted to one-hot encoding vectors. Research on the positive transfer of weights in the word embedding layer is well-established, and thus, instead of focusing on transfer in the embedding layer, the layer has been trained simultaneously on both the source and target corpora.

Next, a three-layer stacked bi-LSTM neural network is constructed in Keras. The model takes as input a sentence of length 50. If the actual sentence is less than 50, then it has been padded. Each of the words has been passed to the fastText embedding layer before it is passed to the network. The paddings are taken care of by converting them into a vector of zeros of size equal to the feature size which has been set to 100. Thus, the input is a sentence tensor of rank 2 and shape (50, 100).

Next, the output is handled. Each word is assigned a unique tag, and hence, the output tensor should also have 50 dimensions along the first axis. The tags are converted to one-hot encoding vectors of size equal to the number of tags. Thus, the shape of the output tensor is (50, n_tags). The corpora used in this research have tags to demarcate whether they form a part of a named entity in the beginning or inside of the entity if it is a multi-word entity. However, for this research, the prefixes (B- and I-) that allow for this demarcation in the tags have been dropped. This makes the number of classes lesser and makes it more easier for the model to learn in a resource-constrained setting.

Now, the model is trained on the source domain corpus. The training corpus contains 14,041 sentences. A validation split of 0.1 is made. The frequency of occurrence of each tag is shown in the table below.

The performance of the model in the source domain testing corpus is shown, and the entity-wise $F1$ score is also shown in the table below. The model is compared with existing top-performing models on the CoNLL-2003 corpus.

This shows that the model is fairly robust in the source domain.

Next, the weights of the model trained on the source domain are reused and the lower layer of the bi-LSTM is frozen. Freezing means that the model will not learn the weights through back-propagation. The model is compared with existing top-performing models on the CoNLL-2003 corpus.

This shows that the model is fairly robust in the source domain (Figs. 1 and 2).

Entity	Precision	Recall	F1-Score	Support
B-LOC	0.85	0.51	0.64	1656
B-MISC	0.62	0.12	0.21	701
B-ORG	0.74	0.42	0.54	1658
B-PER	0.50	0.76	0.61	1580
I-LOC	0.81	0.19	0.31	255
I-MISC	0.24	0.04	0.06	216
I-ORG	0.65	0.26	0.37	827
I-PER	0.54	0.88	0.67	1111
O	0.94	0.97	0.95	38147

Fig. 1 Entity-wise $F1$ score for source domain

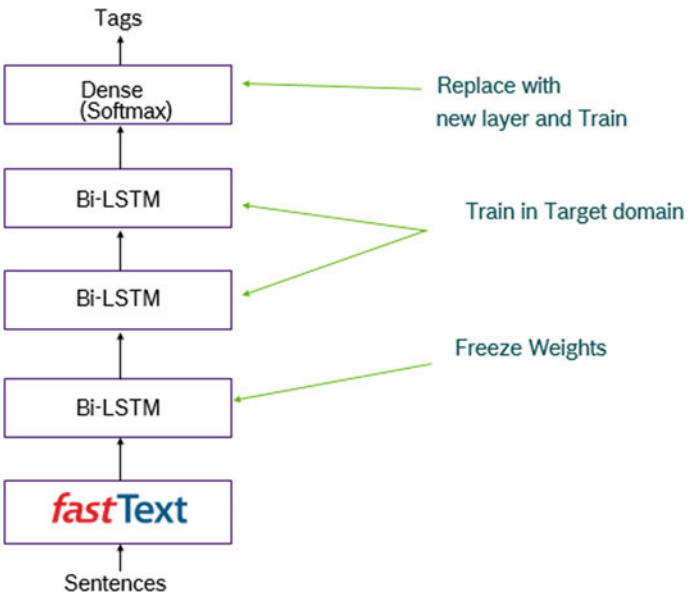


Fig. 2 Transfer learning model

4 Results and Analysis

The performance of the model in the target domain WikiGold is investigated. The approach to transfer learning used is supervised domain adaptation. After freezing the weights in the first layer of the bi-LSTM stack, the model is fine-tuned on the target domain with limited amount of labelled data. The amount of labelled data in the target domain training corpus ranges from 30 sentences to 500 sentences in the training corpus and 100 sentences in the testing corpus. The goal here is to analyze the correlation between the amount of labelled data in the target domain and the benefit of transfer to the target domain. A plot of the validation $F1$ score versus epoch is shown and is a quantitative measure of how much of the learned representations of the source domain are being reused while learning the representations in the target domain (Figs. 3, 4, 5 and 6).

#Train	Transfer (F1)	Without Transfer (F1)
500	0.879	0.870
300	0.887	0.858
100	0.882	0.862
50	0.896	0.845
30	0.892	0.805
15	0.880	0.727

Fig. 3 Table of Wiki domain transfer versus baseline

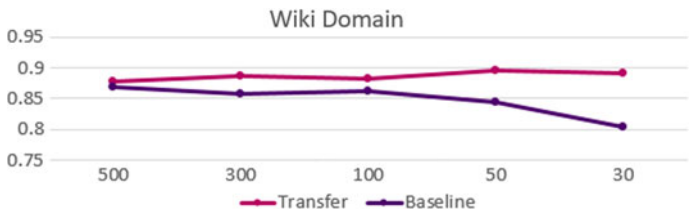


Fig. 4 Plot of Wiki domain transfer versus baseline

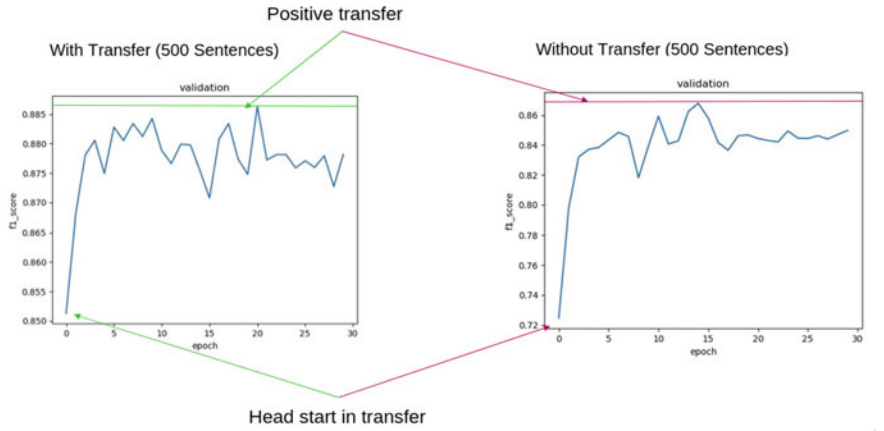


Fig. 5 $F1$ versus epoch (validation set)

Fig. 6 Table of Wiki domain transfer without O tag

#Train	Transfer (F1)	Without Transfer (F1)
100	0.555	0.491
50	0.615	0.428
30	0.610	0.302

It can be seen that the benefit of transfer is more when the O tags are disregarded from the $F1$ score. These can be attributed to the fact that O tags are easier to learn by the model.

The effect of the similarity index of the pair of domains to be transferred is also crucial in determining the extent of transfer.

Reference

1. Goodfellow I, Bengio Y, Courville A (2017) Deep learning. MIT Press, Cambridge, MA

Chapter 6
Knowledge Graph from Informal Text: Architecture, Components, Algorithms and Applications

Anmol Nayak, Vaibhav Kesri, Rahul K. Dubey, Sarathchandra Mandadi, Vijendran G. Venkoparao, Karthikeyan Ponnalagu, and Basavaraj S. Garadi

1 Introduction

A knowledge graph is designed to describe the entities and relationships of a domain. Domel [1] introduced the idea of a Web map to provide navigation support for hypertext browsers. Google leveraged a knowledge graph to improve the quality of search results by building semantic relationships in information extracted from multitude of heterogeneous sources [2]. The Google knowledge graph was based on data from Wikipedia, Freebase [3] as well as public databases. Microsoft developed the Bing search engine based on the Satori knowledge base, which can provide a variety of search services of Web, videos, images and maps. Pujara et al. [4] proposed knowledge graph identification based on ontology aware partitioning to obtain better results. Arnaout and Elbassuoni [5] proposed a general framework that extended both the search knowledge graph and triple-pattern queries for effective searching of RDF knowledge graphs. Fionda et al. [6] introduced the formalism regarding the Web of linked data. Ontology is important to help domain experts regulate and annotate knowledge in their fields. Ontology is a philosophical theory, and it defines a set of representational primitives to model domain knowledge or discourse in the context of computer and information sciences [7]. It is a specification for modeling concepts, an abstract model describing the domain and a formal definition of the concepts and their linkages. It includes class (concepts), slots (roles or properties) and facets (role restrictions). Ontology encapsulated with individual instances of classes comprises a knowledge base [8–23].

Further, in the recent decade, we are seeing an enormous growth in the volume of data being generated from a wide spectrum of areas. With an increased amount of devices being connected to the Internet, data growth is expected to grow dramatically in the coming years. As data generation and consumption increases, it is vital to plan

A. Nayak · V. Kesri · R. K. Dubey (✉) · S. Mandadi · V. G. Venkoparao · K. Ponnalagu · B. S. Garadi
Robert Bosch Engineering and Business Solutions Ltd., Bengaluru, India
e-mail: RahulKumar.Dubey@in.bosch.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_6

how the data should be stored to make data processing efficient. Based on how data is stored, it can be categorized as follows:

(a) Structured data: This type of data is typically stored in a relational database that follows a schema like MySQL and Oracle Database, which makes it easier to analyze.
(b) Semi-structured data: While this type of data is not stored in a relational database, it still follows some organization and structure such as XML format.
(c) Unstructured data: This kind of data does not follow any schema or structure and is often found in sources like word documents, PDF and Blogs. It is the hardest to analyze and requires Natural Language Processing (NLP) algorithms to process.

When data contains relationships between various entities of a domain, it is important that this be reflected in the way data is stored and processed to serve applications. Knowledge graph (KG) is a structured form to capture these relationships and entities from various structured and unstructured data sources. While KG has been gaining traction on open-source datasets that contain information found on the Web, there have not been significant efforts to build KG for applications in industries. Industrial applications can significantly benefit from having a KG representation of data. For example, a typical software application containing several modules, classes and functions with inter-dependencies can leverage a KG representation.

2 Knowledge Graph Development Pipeline and Components

The quality of KG depends on the nodes and relationships extracted from the data. KG development typically consists of two major phases: knowledge extraction and knowledge completion. Knowledge extraction consists of the tasks NER and RE. For example, consider the sentence: *Barack Obama was born in Honolulu*. In this sentence, the named entities can be classified as: {*Barack Obama*: Person}, {*Honolulu*: Location}, and the relation between the two entities is {*Barack Obama*, *born in*, *Honolulu*}. Figure 1 depicts the step-by-step knowledge graph development pipeline from structured/semi-structured (S.S) and unstructured (U.S) data. The details of each KG component are discussed in the next section.

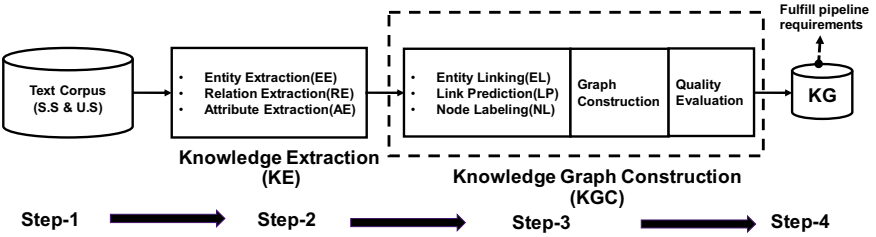


Fig. 1 Knowledge graph development pipeline

3 Knowledge Extraction

3.1 Entity Extraction

The first main task in the pipeline to build a KG is to extract the named entities from the data. The choice to select the appropriate algorithm for a NER task depends largely on the amount of labeled data available to train the models. Some of the popular algorithms to perform named entity recognition (NER) are explained below.

A. **Conditional Random Fields (CRFs)**

CRF is a discriminative undirected graphical model that learns the conditional probability distribution for the various classes of named entities in the data [9]. It treats the input data to be sequential and takes into consideration surrounding context while making a prediction for a given word. In comparison with Markov random fields which model the prior and likelihood, CRF computes the posterior probability between the class labels and inputs directly. The size of the context window determines the order of the CRF model. For example, if the previous word label is also considered while making the prediction of the current word, it will be a first-order CRF. If two previous words are considered, it will be a second-order CRF. The standard linear chain CRF model equations are as follows:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \exp\left\{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\} \tag{1}$$

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{t=1}^{T} \exp\left\{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\} \tag{2}$$

where $\mathbf{y}$: prediction labels of the sentence, $\mathbf{x}$: sequence of words in the sentence, T: total number of words in the sentence, K: total number of feature functions, y_t: label of word at position t, y_{t-1}: label of word at position $t-1$, x_t = word at time t, f_k: kth feature function, Z: normalizing function that sums the probabilities over all possible label assignments, θ_k: weight associated with the kth feature function

Each word is represented by n-dimensional feature vectors. These feature vectors can either be manually designed or derived from word embedding models. Feature functions play the most important role in the decision making of the CRF model. They take as input the feature vector of the word, the label of current word and the label of the previous word. For example, consider the feature function: $f(y_t, y_{t-1}, x_t) = 1$ if y_t = Location and y_{t-1}= Person; $f(y_t, y_{t-1}, x_t) = 0$ otherwise. In this case, a positive weight for this feature function will signify that an entity of type *Location* will typically follow an entity of type *Person*. A CRF model will consist of several such feature functions and learn weights for each feature function in the training phase. The parameters θ for the CRF model are learned using Maximum A Posteriori (MAP) estimates with the help of optimization algorithms like stochastic gradient descent [10, 11].

B. **Bidirectional Long Short-Term Memory (BiLSTM) Neural Networks**

Natural language text is sequential in nature and exploiting this characteristic was the main motivation behind recurrent neural networks (RNNs). RNNs are sequential neural network models that take into account words that have occurred until the point of the word for which the prediction has to be made. With this property, the prediction for a word in a sentence will be influenced by all the words that have occurred before it, thereby giving more contexts to the model. RNN models have performed well when the length of the sentences is short. In cases of longer length sentences, RNN's performance started to decrease significantly due to two major problems:

(i) Vanishing Gradient: This happens when the gradient of the loss function with respect to the weights in the initial layer becomes close to zero due to successive multiplications of the gradient values less than zero. This can happen when the activation function is a sigmoid activation function, which outputs a value between 0 and 1. As a result, the neural network weights in the beginning of the network stop learning which causes the output to become immune to words occurring in the beginning of the sentence.
(ii) Exploding Gradient: Contrary to the vanishing gradient problem, exploding gradient leads to very large updates in the weights in the neural network. This happens due to successive multiplications of values greater than 1, when using activation functions like ReLU. Exploding gradient can lead to an unstable neural network and even overflow in some cases.

To tackle these problems, LSTM [12] networks are used in NLP applications these days to capture long-term dependencies in a sentence. LSTM networks have gates that regulate the information going in and out of an LSTM cell. The components of an LSTM cell are as follows:

(a) Cell state: This is the memory held by each LSTM unit at every time step.
(b) Hidden state: This is a regulated amount of cell state passed onto future time steps.
(c) Input gate: It regulates the amount of new value computed in the LSTM cell being used for the cell state.

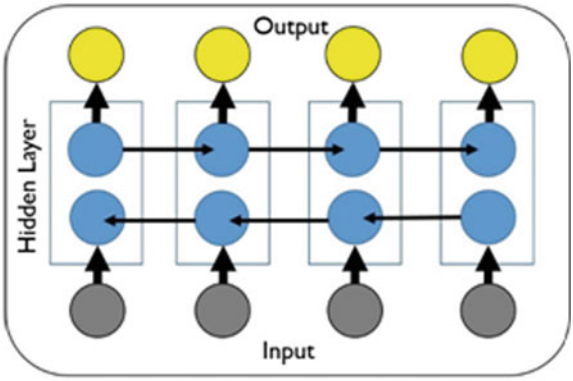


Fig. 2 BiLSTM model architecture

(d) Forget gate: It regulates how much of the previous cell state value is retained.
(e) Output gate: It regulates the amount of cell state that is allowed to be output from the LSTM cell.

The gates often use a sigmoid activation function as it gives an activation between 0 and 1. For example, if the forget gate value is 0, the entire value of the previous cell state is discarded. The LSTM neural network is able to remember and forget long-term dependencies. As a result, LSTM networks have been widely used to perform tasks like NER where it is important to know the sequence of words. While vanilla LSTM networks only look at words that have occurred before a target word, BiLSTM networks (Fig. 2) look for words occurring both before and after a target word. Having a bidirectional view significantly improves the performance of sequence models. BiLSTM model functions in the same way as a LSTM model except that it computes all the components for the forward pass as well as the backward pass. The hidden states from the forward and backward passes are concatenated and passed on for further computation.

3.2 Relation Extraction (RE) and Attribute Extraction (AE)

In most cases, knowledge is represented in the form of a relation triple in a KG. A relation triple is of the form <Subject, Relation, Object> or <Entity, Relation, Entity>. Many popular graph databases that follow the Resource Description Framework (RDF) store information in the form of triples. While the types of relations extracted from a corpus depend on the KG ontology, some of the popular RE algorithms in use today for extracting triples from natural language text are Stanford OpenIE [13] and ClausIE [14]. Further, attribute extraction (AE) is a special kind of relation extraction methodology where each attribute can be represented as a property (in labeled property graphs) or as a relation (in RDF graphs).

A. **Stanford OpenIE**

Stanford OpenIE focuses on handling long sentences by breaking them into short and coherent clauses, which are maximally shortened. This enables the algorithm to have a larger contextual awareness and gives meaningful relation triples to any application consuming it. It uses a multi-class logistic regression classifier that is trained on labeled relations of the Wikipedia corpus, which is then used to generate relations for an unseen corpus.

B. **ClausIE**

ClausIE extracts relations by separating detection of information from the representation in terms of extractions. It uses dependency parsing and a small set of domain-independent lexica. ClausIE needs no training data and is purely domain agnostic in its behavior. It detects clauses in the text and its type to generate meaningful relations.

4 Knowledge Graph Construction

KG construction (also called as completion) is an important phase to make the graph less noisy and more refined. During the graph construction phase, entities and relations might be referred using similar phrases. For example, *Vladimir Putin* and *vLaDImir PUTin* are the same entity and should be resolved into a single node in the KG. Graph completion consists of three important phases, namely entity resolution, link prediction and node labeling.

4.1 Entity Resolution

This task aims at finding nodes that are referring to the same entity in the KG and resolves them into a single node. Entity resolution is also referred to as de-duplication. There are several methods to perform entity resolution:

(i) String similarity: Using edit distance algorithms like Levenshtein distance, two node labels can be compared, and if the edit distance is below a chosen threshold, the two nodes can be merged.
(ii) Embedding similarity: To merge nodes based on semantic similarity, vector embeddings are used to compare two nodes. A vector embedding of each node label can be created and compared using measures like cosine similarity. Vector embeddings are learned using neural networks and can take into account semantic meanings of words. For example, if there are two nodes, namely *Russian President Putin* and *Vladimir Putin*, the two nodes will be merged based on their high semantic similarity.

(iii) Ontological rules: Based on a KG ontology, the category of the node as well as the relations it has with other nodes in the KG, entity resolution can be performed. For example, consider the following relations:
R1: President_of (*Russian President Putin, Russia*)
R2: President_of (*Vladimir Putin, Russia*)
Then, with an ontological rule like:
If $R1 = President\_of\ (A, B)$ *and* $R2 = President\_of\ (C, D)$ *and* $B = D$, *then* $A = C$.

With this rule, *Russian President Putin* and *Vladimir Putin* will be merged into a single node.

4.2 Link Prediction

Unstructured text can often contain missing information, which can lead to an incomplete KG. To tackle this problem, making inferences to predict relations between entities will make the KG more informative and complete. Link prediction is often done by following ontological rules. For example, consider the following relation:

R1: President_of (Russian President Putin, Russia)
Then, with an ontological rule like:
If R1 = President_of (A, B), then create R2 = lives_in (A, B) if R2 does not exist.
With this rule, a new relation lives_in will be created between Russian President Putin and Russia.
Although the above examples of entity resolution and link prediction are straight forward, both these tasks are usually performed via a probabilistic model known as Probabilistic Soft Logic (PSL) [15].

4.3 Node Labeling

Node labeling is the task of appending a new node to the KG by labeling its correct entity class and attributes. PSL is a widely used framework in KG completion tasks. PSL learns the probabilities of rules that will be used in the completion tasks. These rules can have logical operators, arithmetic operators or a combination of both. For example, a PSL rule can be represented as:

$$<\text{Weight}> : <\text{Rule}>\ 25.0 :\ \text{son\_of}\,(\text{A},\ \text{B})\ \&\ \text{son\_of}\,(\text{C},\ \text{B}) \rightarrow \text{brother\_of}\,(\text{A},\ \text{C})$$

Since a KG completion task depends on the soft logic probabilistic score calculated from a weighted sum of PSL rules, these weights decide the influence of a rule.

The weights are usually learned using optimization algorithms. In domain-specific applications based on the underlying ontology of the corpus, these rules form the criteria based on which nodes are merged and new relations are added to the KG.

5 Knowledge Graph Challenges in Sparse Corpus

Some of the notable efforts in KG development include the Google Knowledge Graph, Never Ending Language Learner (NELL) [16], Yet Another Great Ontology (YAGO) [17] and ConceptNet [18]. These have captured several million entities and relations from large-scale datasets like Wikipedia, WordNet, etc. The challenges of building a KG in an industrial setting are multifold: Data is largely unstructured, sparse and contains large amount of domain-specific lexica.

Due to the above challenges, off-the-shelf NLP algorithms perform significantly poor on industrial domain-specific data as compared to open-source datasets like Wikipedia. In the case of NER, while BiLSTM-based models perform extremely well when there is a large amount of labeled corpus, their performance is severely hampered in the case of less labeled and sparse corpus (Fig. 3). CRF-based models perform well even in the case of sparse corpus but require efficient feature engineering to be performed in domain-specific applications found in industries. The performance of the existing RE algorithms also drops when applied on a domain-specific corpus. For example, Stanford OpenIE uses a pre-trained classifier that is trained on labeled relations of the Wikipedia corpus. It uses this trained classifier as a source to generate relations for an unseen corpus. The problem arises when the domain-specific corpus do not follow the same semantic patterns as learned by the classifier. While ClausIE generates a large number of relation triples, they often miss the conditional phrases, which are extremely vital in a domain-specific application. Another challenge with

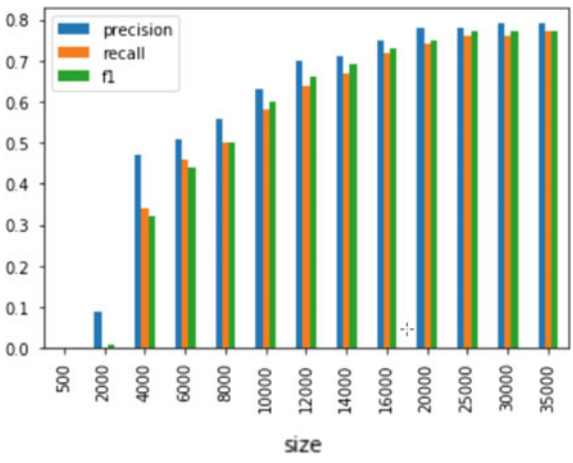


Fig. 3 BiLSTM performance with respect to training set size on Groningen Meaning Bank (GMB) corpus

the existing RE algorithms is that they do not take into consideration the underlying ontology of the corpus. As a result, large number of relations are missed or incorrectly formed when the RE algorithms are applied to an industrial corpus. To tackle the above challenges in NER and RE, we have developed improved versions of the CRF model and ClausIE model to handle domain-specific corpus.

5.1 CRF with Automatic Feature Engineering-Based NER Model

As discussed above, the existing CRF models have their limitations when performing a NER task on domain-specific corpus. To handle the sparsity of data, an efficient domain-specific feature engineering should be performed to capture the niche lexica. Manually performing feature engineering is a laborious task and often requires a domain expert to choose the relevant features. Another important task in CRF-based models is the order. For example, a first-order CRF model considers the labels and features of only the word at time step $t - 1$ in addition to the features of word at time step t. Choosing this order should be based on the n-gram distribution of the named entities found in the corpus. We made the following improvements to the existing CRF model for NER:

(i) Order of the CRF model is chosen based on the most frequently occurring n-gram (at word level) in the named entities. For example, if most of the named entities in the corpus are bi-grams (2 words), then a first-order CRF model will be adequate. However, if the entities are mostly tri-gram (3 words), then a second-order CRF model will perform better as the context window increases in making a prediction.

(ii) Automatic featuring engineering is done by extracting n-grams (at alphabet level) from named entities which occur in less than 'C' named entity classes. These n-grams are extracted from leading and trailing ends of the words. For example, if all the entities belonging to the class 'PERSON' begin with the n-gram phrase 'per_', then this n-gram feature will be useful in identifying PERSON entities.

The value of 'C' is decided based on the total number of entity classes. A small value of 'C' is recommended (<3), since the higher the value of 'C', the n-gram is a common occurrence across multiple entity classes and hence is not useful in distinguishing between entities of those classes. The recommended method to choosing value of 'n' is using heuristics depending on the taxonomy of entities in the domain. The larger the value of 'n', more will be the emphasis that will be given on the entire portion of the entity word.

5.2 AugmentedIE

The quality of relation triples is perhaps the most important component in determining the overall quality of a knowledge graph. As discussed earlier, industrial domain corpus poses its own unique challenges, which have to be handled to create a meaningful KG. We made the following improvements to the existing ClausIE model for improving RE:

(i) Grouping triples based on ontology: While the existing RE algorithms operate at a sentence level, it is important to consider the ontology of the domain while grouping triples. This leads to a fully connected graph and prevents dangling triples in the KG. Grouping triples is extremely important in an industrial application that consists of hierarchy between the various entity classes. For example, a complex system can have multiple modules, which in turn can have multiple functions, which in turn can have signals and variables. Adhering to such an ontology will help in creating new relations between entities.

(ii) Co-reference resolution (Coref): It is very common to see entities being referred in different ways across a domain document. For example, consider the following sentences: *Vladimir Putin is the president of Russia. He lives in Moscow.* In these sentences, *Vladimir Putin* and *He* are the same references. Before generating triples, *He* must be resolved as *Vladimir Putin*. Coref significantly improves the quality of triples generated from a corpus. A widely popular Coref algorithm that is used is Neuralcoref [19].

(iii) Handling conditional sentences: The existing RE algorithms do not perform well in the case of conditional sentences. Some of the problems of ClausIE on conditional sentences are:

- Triples miss the conditional phrase.
- The cause and effect components of the conditional sentence are not coupled.

For example, consider the following conditional sentence:

C is ON if A is ON and if B is OFF. ClausIE generated the following triples for it: *<C> <is> <ON if A is ON>*, *<C> <is> <ON if B is OFF>*, *<A> <is> <ON>*, *<B> <is> <OFF>*.

As we can see, the triples are noisy and incorrect. The ideal triple will be: *<C is ON> <if> <A is ON and B is OFF>*.

To tackle these challenges, we have developed a novel algorithm using a Constituency Parse Tree (CPT) to identify the cause and effect phrases from a conditional sentence. CPT breaks down each sentence into a set of clauses with the leaf nodes being Part of Speech (POS) tags for each word. Each causal condition is referred to as a Pre-condition, and each effect is referred to as Post-condition. In the above example, *A is ON* and *B is OFF* are Pre-conditions and *C is ON* is a Post-condition. The algorithms are described below:

CPT pathfinder algorithm steps for finding Pre-conditions:

Step 1: Apply Depth First Search (DFS) on the CPT to find a subtree with root node as *'SBAR'* or *'PP'*. Let this subtree be T_n found at depth n of the CPT.
Step 2: Apply Breadth First Search (BFS) for T_n and check if there exists a pair of child subtrees $ST_{x,n+1}$ and $ST_{x+1,n+1}$, where x and $x + 1$ are the index of child subtrees found at level $n + 1$, such that root node of $ST_{x,n+1}$ is *'IN'* or begins with *'WH'* and root of $ST_{x+1,n+1}$ is *'S'*.
Step 3: Check if there exists more than one child subtree $ST_{y,n+2}$ of $ST_{x+1,n+1}$, where y is in range(0, number of subtrees of $ST_{x+1,n+1}$ at level $n + 2$), such that the root node of $ST_{y,n+2}$ is *'S'*. For each such subtree, $ST_{y,n+2}$ perform Step 4. If none exist, perform Step 4 for $ST_{x+1,n+1}$.
Step 4: Apply DFS and create a Pre-condition string appending each leaf node until a subtree is found that satisfies Step 1 and Step 2.
Step 5: Step 1 DFS continues.

CPT pathfinder algorithm for finding Post-conditions:

Step 1: Apply BFS on the CPT to find pairs of subtrees $T'_{x',n'}$ and $T'_{x'+1,n'}$,with '*P*' in root node label where n' is the depth in the CPT and x' and $x' + 1$ are the BFS index, respectively.
Step 2: $T'_{x',n'}$ and $T'_{x'+1,n'}$ should not be subtrees of any Pre-condition string tree $ST_{x+1,n+1}$. The leaves of $T'_{x',n'}$ and $T'_{x'+1,n'}$ are appended in sequence to create a Post-condition string. Each such pair of subtrees creates a Post-condition string.
Step 3: If there exists an odd-numbered subtrees in Step 1, the leaves of this subtree create the Post-condition string.

Executing the above algorithms on the sentence:
C is ON if A is ON and if B is OFF, we get the following Pre-conditions and Post-conditions: Pre-conditions: (i) *A is ON* (ii) *B is OFF;* Post-conditions: (i) *C is ON*

As we can see, the algorithm successfully captures the cause and effect phrases in the conditional sentence. After we extract the Pre-conditions and Post-conditions, triples can be created as needed depending on the type of conjunction present in the sentence. In the above sentence, a triple can be extracted with the following template:
<POST-CONDITION1> <if> <PRE-CONDITION1 and PRE-CONDITION2>
<C is ON> <if> <A is ON and B is OFF>

6 Industrial Applications

KG can be greatly leveraged in industrial applications where there is a large amount of data containing numerous entities and relationships. Some of the general use cases that KG can serve are Question-Answering (QA) system, recommendation systems, fraud detection. Specifically in the SDLC, some of the applications where a KG can

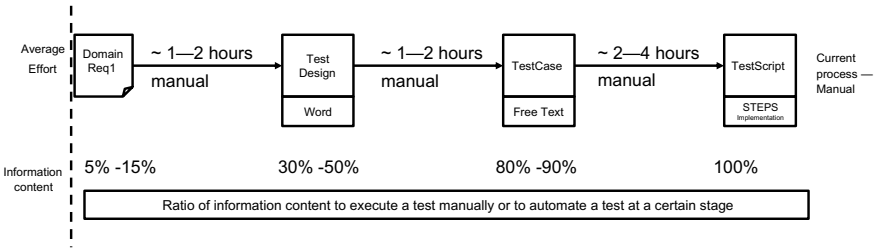


Fig. 4 Current test development challenges

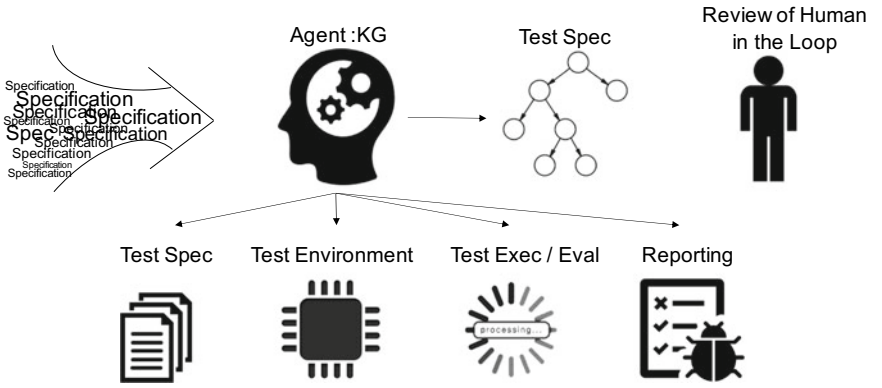


Fig. 5 KG-based Req2Test

be efficiently used are test case generation, requirement analysis, bug prediction and localization. Since every industrial application goes through a rigorous testing cycle, we have developed an automatic test case generation system called Requirement to Test Case (Req2Test) to overcome the existing test development challenges (Figs. 4 and 5) that leverages a KG to capture domain knowledge (Fig. 4).

The first step in generating test cases is to extract the test intent from the requirement statement. The test intent describes three fundamental components required to create a test case:

(i) Pre-conditions: These are all the conditions required to be satisfied before an Action or Post-condition of the requirement statement can be achieved.
(ii) Actions: These are all the actions that are performed once the Pre-conditions have been satisfied.
(iii) Post-conditions: These are all the conditions that are observed after the Pre-conditions and Actions have been satisfied.

For example, consider the requirement statement: *If SignalA is on and ButtonA is pressed, then SignalB is on*. The testable intent will be: Pre-conditions: *SignalA is on*, Actions: *ButtonA is pressed*, Post-conditions: *SignalB is on*.

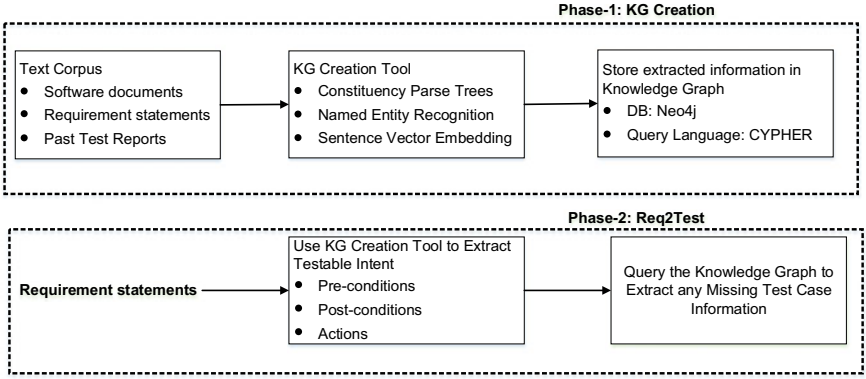


Fig. 6 KG pipeline for automatic generation of test cases

The Req2Test pipeline (Fig. 6) consists of two phases: Phase 1 deals with generating the KG from the text corpus available. Phase 2 deals with processing the requirement statements, extraction of test intent and querying the KG to generate the test case.

6.1 Phase 1—KG Creation

The information extraction tool consists of the CPT algorithm discussed above, HOCRF with automatic feature engineering-based NER model and Fasttext vector embedding [24]. The CPT algorithm is executed on the text corpus to represent the entire domain with respect to the ontology (Fig. 7). NER is performed to check if any of the Pre-conditions or Post-conditions are actually Actions. If so, these nodes are assigned to Action only entity. Sentence vector embedding is used to map text found in the requirement statements to the nodes found in the KG, as usually there is a variation in the descriptions found in the requirements and that of the KG. The KG (Fig. 8) is stored in a Neo4j graph database and queried using CYPHER language.

6.2 Phase 2—KG Based Req2Test Application

This phase uses the information extraction tool to extract the test intent from the requirement statements. Once the test intent is found, the corresponding nodes are queried in the KG. Since usually the requirements do not mention all the test case conditions, these missing conditions are grabbed from the KG.

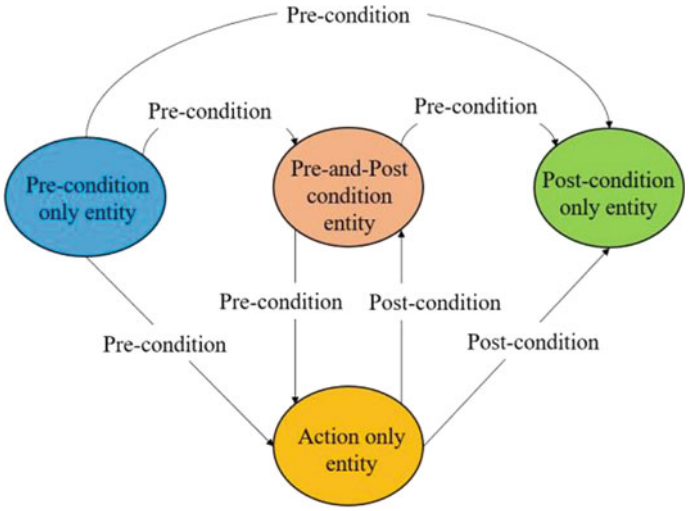


Fig. 7 KG ontology for Req2Test

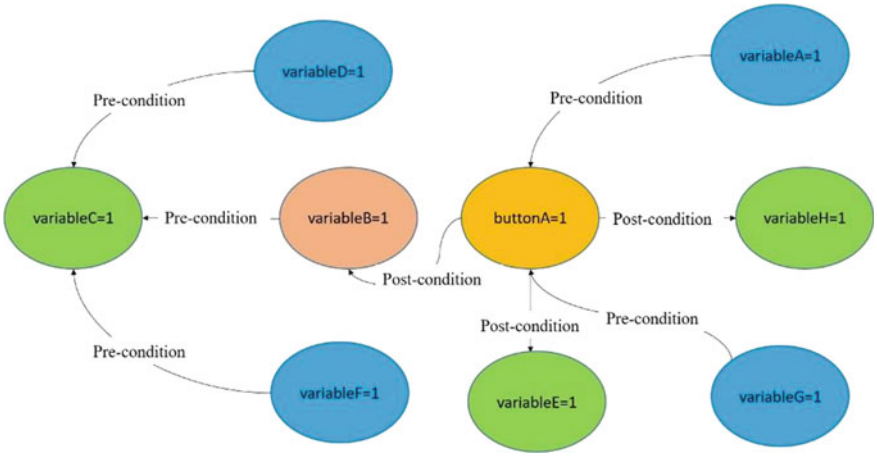


Fig. 8 Sample KG for Req2Test in Neo4j Database

7 Summary

Knowledge graph is a powerful tool that can be used to efficiently represent complex systems. While the existing KG has leveraged primarily structured data, new techniques need to be developed in NER and RE to handle semi-structured and unstructured data. The current NER and RE algorithms have challenges when applied to

industrial domain applications. We have proposed two algorithms, namely Higher-Order CRF with automatic feature engineering and AugmentedIE, that tackle challenges in off-the-shelf NER and RE models, respectively. There are several deep learning approaches which can be applied to potentially improve the performance of NER and RE models. For example, recent works have proposed leveraging transfer learning, where a pre-trained neural network is used as a starting point and fine-tuned with the less amount of labeled data in the target domain. Another recent advancement in NLP uses attention models to improve the performance of NER and RE models. Attention models can remember long-term dependencies in text and use that to make better prediction on tasks.

References

1. Domel P (1995) Web map: a graphical hypertext navigation tool. Comput Networks ISDN Syst 28(1–2):85–97
2. Singhal A (2019) Introducing the knowledge graph: things, not strings. https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html. Last accessed 19 Oct 2019
3. Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J (2008) Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on management of data. Vancouver, Canada, pp 1247–1250 (2008)
4. Pujara J, Miao H, Getoor L, Cohen WW (2013) Ontology-aware partitioning for knowledge graph identification. In: Proceedings of the 2013 workshop on automated knowledge base construction. ACM, San Francisco, CA, USA, pp 19–24 (2013)
5. Arnaout H, Elbassuoni S (2018) Effective searching of RDF knowledge graphs. J Web Semant 48:66–84
6. Fionda V, Gutierrez C, Pirrò G (2016) Building knowledge maps of Web graphs. Artif Intell 239:143–167
7. Liu L, Özsu MT (2009) Encyclopedia of database systems, vol 6. Springer, Boston, pp 1963–1965
8. Noy NF, McGuinness DL (2001) Ontology development 101: a guide to creating your first ontology. Knowledge Systems Laboratory, Stanford University
9. Lafferty JD, McCallum A, Pereira FCN (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th international conference on machine learning. Williamstown, MA, USA, pp 282–289 (2001)
10. Song D, Liu W, Zhou T, Tao D, Meyer DA (2015) Efficient robust conditional random fields. IEEE Trans Image Process 24(10):3124–3136
11. Vishwanathan SVN, Schraudolph NN, Schmidt MW, Murphy KP (2006) Accelerated training of conditional random fields with stochastic gradient methods. In: Proceedings of the 23rd international conference on machine learning, vol 6, pp 969–976
12. Zhou Q, Wu H (2018) NLP at IEST 2018: BiLSTM-Attention and LSTM-Attention via soft voting in emotion classification. In: Proceedings of the 9th workshop computational approaches to subjectivity, sentiment and social media analysis, pp 189–194
13. Angeli G, Premkumar MJ, Manning CD (2015) Leveraging linguistic structure for open domain information extraction. In: Proceedings of the 53rd annual meeting of the ACL and the 7th international joint conference on NLP, vol 1. Long Papers, pp 344–354
14. Corro LD, Gemulla R (2013) Clausie: clause-based open information extraction. In: Proceedings of the 22nd international conference on world wide web. ACM, New York, pp 355–366 (2013)

15. Kimmig A, Bach SH, Broecheler M, Huang B, Getoor L (2012) A short introduction to probabilistic soft logic. In: Proceedings of the NIPS workshop on probabilistic programming: foundations and applications, pp 1–4
16. Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka ER, Mitchell TM (2010) Toward an architecture for never-ending language learning. In: Proceedings of the twenty-fourth AAAI conference on artificial intelligence, pp 1306–1313
17. Suchanek FM, Kasneci G, Weikum G (2007) Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on world wide web. ACM, New York, NY, USA, pp 697–706
18. Speer R, Chin J, Havasi C (2017) ConceptNet 5.5: an open multilingual graph of general knowledge. In: Proceedings of the 31st AAAI conference on artificial intelligence (2017)
19. Lee K, He L, Lewis M, Zettlemoyer L (2017) End-to-end neural coreference resolution. In: Proceedings of the 2017 conference on empirical methods in NLP, pp 188–197
20. Lin ZQ, Xie B, Zou YZ, Zhao J, Li X, Wei J, Sun H, Yin G (2017) Intelligent development environment and software knowledge graph. J Comput Sci Tech 32(2)
21. Fröhlich P, Link J (2000) Automated test case generation from dynamic models. In: Proceedings of European conference on object-oriented programming. Springer, Berlin, pp 472–491 (2000)
22. Tahat LH, Vaysburg B, Korel B, Bader AJ (2001) Requirement-based automated black-box test generation. In: Proceedings of the 25th COMPSAC. IEEE, New York, pp 489–495 (2001)
23. Software & Systems Engineering Standards Committee of the IEEE Computer Society: IEEE Standard for software and system test documentation. IEEE, New York (2008)
24. Bojanowski P, Grave E, Joulin A, Mikolov T (2017) Enriching word vectors with subword information. Trans ACL 5:135–146

Chapter 7
Neighborhood-Based Collaborative Recommendations: An Introduction

Vijay Verma and Rajesh Kumar Aggarwal

1 Introduction

Collaborative filtering is the most popular and widely used technique in recommender systems. The detailed discussions on CFRSs are provided in various research articles published in the past; Table 1 lists these articles based on their nature/type along with the total citations received.

The key idea behind CFRSs is that the information regarding the past behaviors or interests of users (belonging to a particular system) are analyzed and exploited for predicting the items of interest for an active user of the system. CFRS techniques consider the following assumptions:

- In the past, if users had shown the same interests, then they will have similar interests in the future too.
- With respect to the time, user's preferences remain constant and consistent.

Additionally, CFRS provides three key advantages over content-based approaches. Firstly, these approaches are applicable for items for which the content is not available or burdensome since humans conclude the overall quality of items. Secondly, CFRS is based on the quality of items as judged by associates rather than the content of items, which may not be a good indicator of the quality. Finally, these approaches may suggest items with entirely different content as long as other similar users have shown interest in these items, i.e., items with lucky discovery.

V. Verma (✉) · R. K. Aggarwal
Computer Engineering Department, National Institute of Technology, Kurukshetra, Kurukshetra, Haryana 136119, India
e-mail: vermavijay1986@gmail.com

R. K. Aggarwal
e-mail: rka15969@gmail.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_7

Table 1 Research articles on CFRSs

References	Title	Type or nature	Year	Total citations[a]
[1]	A Survey of Collaborative Filtering Techniques	Survey	2009	2938
[2]	Collaborative Filtering Recommender Systems	Survey	2011	820
[3]	Collaborative Filtering beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges	Survey	2014	447
[4]	Collaborative Filtering Recommender Systems	Book chapter	2007	1750
[5]	A Comparative Study of Collaborative Filtering Algorithms	Comparative Study	2012	142
[6]	Comparing State-of-the-Art Collaborative Filtering Systems	Comparative Study	2007	198

[a]As reported from Google scholar till May 29, 2019

Article [7] suggests that collaborative recommender systems can be further categorized into two broad classes: memory-based and model-based approaches. Memory-based algorithms are basically heuristic in nature and provide recommendations using the entire collection of rating data available in the user-item matrix (UI-matrix) [8, 9]. Memory-based algorithms, also known as neighborhood-based collaborative filtering algorithms, were among the earliest approaches for collaborative filtering. Memory-based algorithms predict ratings for user-item pairs based on their neighborhoods. In contrast to neighborhood-based approaches, the model-based approaches build a model from rating data via learning [10–13]. The model-based approaches represent the user-item interactions with their latent characteristics and are trained using different methods from machine learning and data mining, such as decision trees, rule-based, Bayesian clustering, SVM, SVD, LSA, and LDA. Lastly, this model is used for predicting the preferences of users for new items.

2 Notations

In order to describe the mathematical formulations of the recommendation process, we need to introduce a few notations which are used throughout the chapter. Firstly, the notations for underlying objects (items and users) in an RS are described. The items are denoted by variables $i, j \in I$, and the users are denoted by variables $u, v \in U$, where I is the set of all items in the system, and U is the set of all users in the system. If more than two users or items are needed, then we will use the numeric subscripts, such as $i_1, i_2, \ldots, i_n$. The rating data recorded in the form of a user-item matrix (UI-matrix) of order $|U| \times |I|$, also called rating matrix and

Table 2 Notations used throughout the chapter

Symbol	Meaning
U	All the users of the system
I	All the items available in the system
R	All the ratings made by the users for the items
S	All possible preferences that can be expressed by the users such as $S = \{1, 2, 3, 4, 5\}$
r_{ui}	The rating value given by a user $u \in U$ to a particular item i
I_u	The set of items rated by a particular user u
U_i	The set of users who have rated a particular item i
I_{uv}	The set of items rated by both the users u and v, i.e., $(I_u \cap I_v)$
U_{ij}	The set of users who have rated both the items i and j, i.e., $(U_i \cap U_j)$

denoted by R. This rating matrix, R, is sparsely populated and may also be loosely treated as a set of ratings. An entry $r_{ui} \in R$ denotes the preference from user u for item i. Also, the set of all possible values for a rating is denoted by S (e.g., $S = \{1, 2, 3, 4, 5\}$ or $\{\text{like}, \text{dislike}\}$).

The following subsets are commonly used in the recommender system literature. U_i is the subset of users who have rated the item i; likewise, I_u represents a subset of items rated by user u. Finally, I_{uv} represents the set of items that are rated by two users, u and v, i.e., $I_u \cap I_v$. Similarly, U_{ij} denotes the set of users who have rated both items i and j. Table 2 summarizes these notations. The area of RSs is evolving continuously, and different authors/researchers have used endless notations for describing their work. To that end, research [14] tries to suggest a common way of providing notations with proper justification in the RS field for teaching and research purposes.

3 Neighborhood-Based Recommendations

Among all CFRSs, the neighborhood-based algorithms (e.g., k-nearest neighbors) are the traditional ways of providing recommendations [15]. These algorithms are very popular due to their simplicity in terms of implementation and efficiency in terms of performance. These algorithms are based on the popular precept of word-of-mouth, which states that people have confidence in the opinions of other like-minded people. The fundamental assumption behind these approaches is that similar users demonstrate similar interests, whereas similar items draw similar rating patterns [16]. In order to demonstrate this idea, consider the following example based on the ratings of Fig. 1.

Example 1 Suppose the system has to predict the interestingness of user Pintu for the movie "Darr" that he has not watched earlier. The system realizes that both Pintu and Chintu have similar preferences with respect to movies as both of them did not like

	Baazigar	Darr	Mohra	Krish	Dil Se
Chhotu	5	3		1	1
Pintu	2	?	3	4	5
Chintu	1	5	2	5	5
Motu		2			

Fig. 1 Example scenario explaining the principle of word-of-mouth

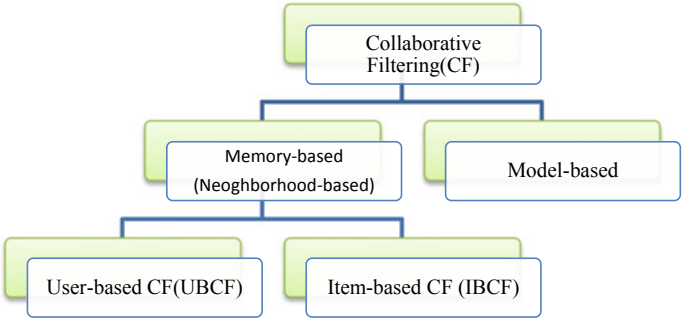


Fig. 2 General classification of collaborative filtering-based recommendations

"Baazigar" but admired "Krish"; therefore, the system may recommend the movie "Darr" to the user Pintu with high relevancy.

Basically, there are two types of neighborhood-based approaches: User-based collaborative filtering (UBCF) and item-based collaborative filtering (IBCF). Figure 2 depicts a broad classification of collaborative filtering-based recommender systems.

User-based Collaborative Filtering (UBCF): the main idea is to identify similar users (also called peer users or nearest neighbors) who displayed similar preferences to those of an active user in the past. Then, the ratings provided by these similar users are used to provide recommendations.
Item-based Collaborative Filtering (IBCF): In this case, for estimating the preference value for an item i by an active u, firstly, determine a set of items which are similar to item i, then, the ratings received by these similar items from the user u are utilized for recommendations.

One substantial difference between UBCF and IBCF algorithms is that in the former case, ratings of peer users are utilized, whereas in the latter case, active user's own ratings are used for prediction purposes. With respect to the UI-matrix, UBCF approaches define the similarities among rows (or users), whereas IBCF approaches define similarities among columns (or items), as shown in Fig. 3. However, both UBCF and IBCF provide different types of recommendations but are almost identical to each other with small differences.

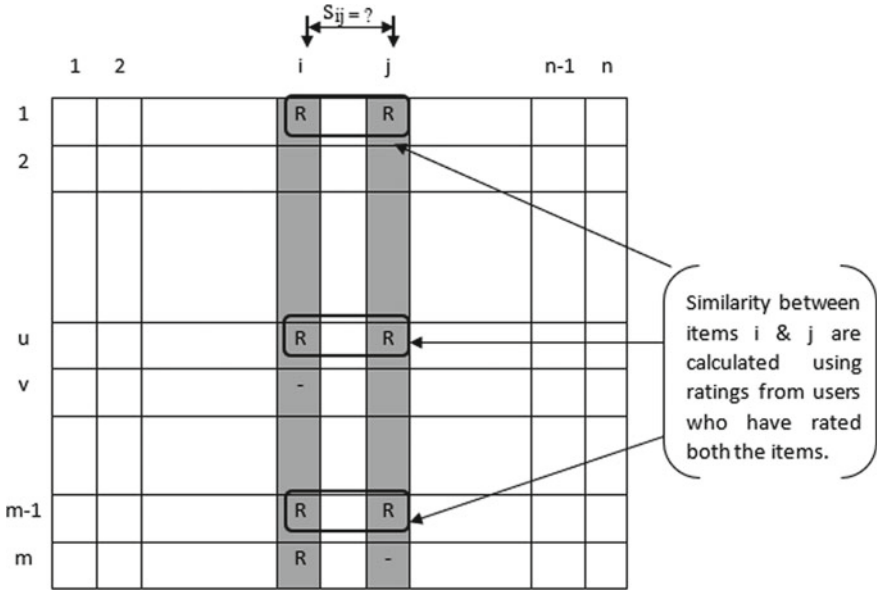


Fig. 3 Example scenario for calculating item-item similarity in IBCF

3.1 User-Based Recommendations

For an active user u for whom we want to predict the rating r_{ui} for new item i, a user-based method usually performs the following tasks [17]:

- Calculate the similarity between the target user and all other users.
  Research Issue: how this similarity can be computed?
- A subset of users, based on similarity weight, is retained as the neighborhood of the target user u and denoted by $N(u)$.
  Research Issue: how this neighborhood is selected?

The above two steps are used to improve the efficiency (in terms of speed) of user-based methods and may be performed off-line before the actual recommendation process. Furthermore, it is very difficult and impractical to consider every other item that the target user has not rated as a candidate for the recommendation; therefore, these steps consider only items that are familiar to users in the neighborhood.

- Select k closest users (from the neighborhood of the target user) who have rated item i (i.e., separately for each predicted item) such that $N_i(u) = k$. It is necessary since we have to consider only those neighbors who have rated item i, here, $N_i(u)$ denotes the neighbors of user u who have rated the item i. These k closest users are known as k-nearest neighbors (k-NN).
- Finally, the rating r_{ui} can be estimated using the ratings from users in the set $N_i(u)$.

The simplest way for estimating the value of r_{ui} is to average the ratings of $N_i(u)$.

$$\hat{r}_{ui} = \frac{1}{|N_i(u)|} \sum_{v \in N_i(u)} r_{vi} \tag{1}$$

But, Eq. (1) does not take into consideration the similarity weights of different neighbors while estimating the preference value $\hat{r}_{ui}$,; therefore, one must weight the rating of each user v with its similarity to the target user u.

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i(u)} w_{uv} * r_{vi}}{\sum_{v \in N_i(u)} |w_{uv}|} \tag{2}$$

Equation (2) takes only positive values of similarity in its denominator so that the predicted rating will remain within the range. Furthermore, Eq. (2) does not consider the individual's difference for quantifying the same level of admiration using different rating values for an item. This situation can be generally handled by using normalized ratings, i.e., a weighted average of normalized ratings is utilized by means of some function $h(r_{ui})$ in order to handle this issue. It is noteworthy to understand that predicted rating must be transformed into the initial rating scale; therefore, one has to perform the operation h^{-1} in Eq. (3).

$$\hat{r}_{ui} = h^{-1}\left(\frac{\sum_{v \in N_i(u)} w_{uv} * h(r_{vi})}{\sum_{v \in N_i(u)} |w_{uv}|}\right) \tag{3}$$

User-based classification

The approach, described in Sect. 3.1, estimates the rating values using the regression-based technique. However, one may also employ a classification-based approach for predicting the value of r_{ui} using considering neighbor's vote for the item i. Formally, for each rating $r \in S$, the vote v_{ir} given by the neighbors of u can be calculated as follows:

$$v_{ir} = \sum_{v \in N_i(u)} \delta(r_{vi} = r) w_{uv} \tag{4}$$

Here, $\delta(r_{vi} = r)$ is 1 if $r_{vi} = r$, and 0 otherwise. After computing the votes for each possible value of the rating, the estimated rating will be the one for which v is maximum.

$$\hat{r}_{ui} = \text{argmax}_{r \in S}\left(\sum_{v \in N_i(u)} \delta(r_{vi} = r) w_{uv}\right) \tag{5}$$

How to choose between regression and classification?
The selection of either neighborhood-based regression or classification usually depends upon the way of collecting or recording the users' preference data. Therefore, the system's rating scale determines which one is better suited for the application at hand. If ratings are continuous, then a regression-based approach is more suitable; however, if the rating scale consists of only a few discrete values, then a classification-based method might be fitting. Additionally, the normalization process usually results in ordinal ratings into continuous ratings, so the classification-based approaches also become challenging.

3.2 Item-Based Recommendations

In item-based recommendations, neighborhoods are defined in terms of items rather than users; therefore, similarities between items are calculated, and the user's own ratings are utilized for estimating ratings. The procedure is analogous to user-based recommendation; thus, the two methods experience an interdependent connection. Nevertheless, both approaches provide recommendations that have an ample amount of differences. Here, for a target item i for which we want to predict rating r_{ui} for a user u, an item-based method carries out the following tasks:

- Calculate the similarity between the target item and all other items.
- A subset of items, based on similarity weight, is retained as the neighborhood of the target item i and denoted by $N(i)$.
  Here, again, these two steps are necessary for improving the efficiency (in terms of speed) of item-based methods and may be performed off-line before the actual recommendation process.
- Select k closest items (from the neighborhood of the target item) which are rated by the user u (i.e., chosen for each user differently) such that $N_u(i) = k$. This is necessary as we have to consider only those items that are rated by the user u. Here, $N_u(i)$ denotes the neighbors of item i which are rated by the user u; these k closest items are known as k-nearest neighbors (k-NN).
- Finally, the rating r_{ui} can be estimated using the ratings from user u given to items in the set $N_u(i)$. Now, the value of r_{ui} may be estimated using methods analogous to the user-based methods.

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u(i)} w_{ij} r_{uj}}{\sum_{j \in N_u(i)} |w_{ij}|} \tag{6}$$

Here, again, the differences in the ratings received by an item can also be taken into consideration by using Eq. (7)

$$\hat{r}_{ui} = h^{-1}\left(\frac{\sum_{j \in N_u(i)} w_{ij} * h(r_{uj})}{\sum_{j \in N_u(i)} |w_{ij}|}\right) \tag{7}$$

3.3 User-Based Versus Item-Based Methods

RS designers should consider the following criteria when deciding between user-based and item-based methods.

Computational complexity
Neighborhood-based approaches consist of two phases: an off-line phase and an online phase. Computing the similarity values and neighborhood selection is accomplished during the off-line phase. Therefore, for each user (or item), an appropriate peer group is identified and stored based on the calculations of the off-line phase. In the online phase, these similarity values and peer groups are utilized for proving the recommendations. Let us assume that p is the maximum number of ratings given by a user, and obviously, p will be far less than total number of items (n) in the system, i.e., $p \ll n$; similarly, q is the maximum number of ratings received by an item, and clearly, q will be very less than the total number of users (m) in the system, i.e., $q \ll m$.

So, we can say that the similarity calculation will take the maximum running time of p for a couple of users. In order to determine the peers of a user, the total running time will be $O(m.p)$; thus, the overall running time for finding the peers of all users will be $O(m2.p)$, i.e., the running time of the off-line phase in a user-based method. Analogously, the running time of the off-line phase of an item-based method will be $O(n2.q)$. In an online phase, the estimation of the predicted rating will take $O(k)$ running time for both user-based and item-based methods; here, k is the neighborhood size of peers. Furthermore, such estimation of ratings can be computed for the maximum of all items; thus, the online running time will be $O(k.n)$ for both user-based and item-based approaches. Table 3 summarizes these computational complexities of neighborhood-based recommendations.

It is clear that the space and time complexity of neighborhood-based methods depend on the proportion of the number of users to the number of items. If the number of users is more than the number of items, then item-based methods require less space and time in comparison with user-based methods (during off-line phase) and vice versa.

Accuracy
Similar to computational complexity, the comparative accuracy between user-based and item-based approaches relies upon the ratio of the number of users to the number of items in the system, i.e., dataset at hand. In cases where the number of users is

Table 3 Space and time complexity of user-based and item-based methods

	Space	Time	
		Off-line	Online
UBCF	$O\ (\|U\|^2)$	$O\ (\|U\|^2\ .\ p)$	$O\ (\|U\|\ .\ k)$
IBCF	$O\ (\|I\|^2)$	$O\ (\|I\|^2\ .\ q)$	$O\ (\|I\|\ .\ k)$

more than the number of items (such as Amazon.com), then item-based approaches can result in more accurate recommendations [18, 19]. Similarly, when the number of items is more than the number of users (such as research paper recommender systems), then user-based methods provide better accuracy than item-based methods [20]. However, in most cases, item-based approaches often result in better recommendations in terms of accuracy since the user's own ratings are utilized for the recommendation process, whereas user-based methods extrapolate other users' ratings.

Stability
The comparative stability of user-based and item-based approaches relies upon the repetitiveness and quantity of alteration in the users and items of the system. If the number of items in the system is not changing with respect to the number of users in the system, then it is preferable to use item-based methods as the similarity between items can be computed and stored occasionally.

Explanation
It is always beneficial for both the system and its users if the system provides justification for the recommendations to the users. Justifiability will increase the trust and confidence of the users toward the recommender system. Item-based methods have the edge over the user-based approaches since the neighborhood items that are used in the prediction process can be shown to the users as a logical reason for the recommendations.

Serendipity
The concept of serendipity augments the notion of novelty by including a factor of surprise [20]. Since the item-based methods recommend items that similar to ones that are enjoyed by a user in the past; therefore, the recommendations are not different from the usual interest of that particular user. On the other hand, user-based methods recommend items that other similar users appreciated in the past. Thus, user-based approaches will provide more serendipitous recommendations in comparison with item-based approaches.

4 Neighborhood-Based Methods in Action

While implementing the neighborhood-based recommendations, one has to decide on many factors such as regression-based versus classification-based rating prediction, user-based versus item-based approach as these choices influence the overall quality of the recommendations. Additionally, there are other very crucial considerations that may have a significant impact on the recommendation process, e.g., rating normalization, computation of similarity values, neighborhood selection, etc. Here, we will discuss the most common variations for each of these components, along with their pros and cons.

4.1 Rating Normalization

In order to consider an individual's difference in quantifying the same level of admiration via different ratings, there are two popular ways for rating normalization: mean-centering and z-score.

Mean-centering
Here, the raw ratings are converted into positive or negative ratings, which directly provide the appreciation of the user for an item via the sign of the normalized rating. In a user-based method, raw rating r_{ui} is converted into mean-centered rating $h(r_{ui})$ by subtracting average user rating $\overline{r}_u$ from it as follows:

$$h(r_{ui}) = r_{ui} - \overline{r}_u \tag{8}$$

Therefore, the user-based prediction is obtained using Eq. (9)

$$\hat{r}_{ui} = \overline{r}_u + \frac{\sum_{v \in N_i(u)} w_{uv} * (r_{vi} - \overline{r}_v)}{\sum_{v \in N_i(u)} |w_{uv}|} \tag{9}$$

***Z*-score normalization**
This scheme also considers the spread of the ratings and is defined as follows:

$$h(r_{ui}) = \frac{r_{ui} - \overline{r}_u}{\sigma_u} \tag{10}$$

So, the rating prediction is computed using Eq. (11).

$$\hat{r}_{ui} = \overline{r}_u + \sigma_u \frac{\sum_{v \in N_i(u)} w_{uv} * (r_{vi} - \overline{r}_v)/\sigma_v}{\sum_{v \in N_i(u)} |w_{uv}|} \tag{11}$$

For item-based methods, analogous equations similar to Eqs. (9) and (11) can be written easily.

4.2 Similarity Computation

The similarity values are not only used for selecting the neighborhood but also for providing weight to these neighbors in the prediction process; therefore, similarity values portray a very crucial role in neighborhood-based recommendations. Numerous similarity measures have been proposed in the RS literature. The traditional similarity measures, such as the Pearson correlation coefficient (PCC), Cosine similarity, etc., are popularly used to determine the similarity between users or items.

$$\text{PCC}(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \overline{r}_u)(r_{vi} - \overline{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \overline{r}_u)^2 \sum_{i \in I_{uv}} (r_{vi} - \overline{r}_v)^2}} \tag{12}$$

$$\text{COS}(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} r_{vi}}{\sqrt{\sum_{i \in I_u} r_{ui}^2 \sum_{i \in I_v} r_{vi}^2}} \tag{13}$$

The rationale for the significance of similarity weight

Due to data sparsity, similarity values may be often calculated using a small number of common ratings, which may result in poor recommendations. Therefore, one should always consider the importance of similarity weights using some strategies. In particular, significance weighting [17, 21] calculates the effective similarity by acknowledging the number of commonly rated items in the following manner.

$$W'_{uv} = \frac{\min\{|I_{uv}, \gamma\}}{\gamma} \times w_{uv} \tag{14}$$

Here, γ is a threshold parameter which is determined empirically, and it has been shown that $\gamma \geq 25$ provides better predictive accuracy, while $\gamma = 50$ results in the best results [16, 17]. Yet, the parameter's optimal value is data-dependent. Another approach uses the concept of shrinkage for improving an inadequate predictor if it becomes valueless, and the strategy is rationalized through the Bayesian perspective [22].

Rationale for variance

While calculating the similarity between users via commonly rated items, it is irrelevant to consider items that are rated (either liked or disliked) by the entire population. For example, most users like a famous movie such as "Titanic"; therefore, the similarity between two users should not be calculated using this movie. In order to handle such biases, Breese et al. [7] used inverse user frequency analogous to the concept of inverse document frequency in the information retrieval domain. In this approach, each item is assigned a weight (λ_i), according to Eq. (15), and then, the similarity between users is calculated using frequency-weighted Pearson correlation (FWPC) as defined in Eq. (16) for better predictive accuracy.

$$\lambda_i = \log \frac{|U|}{|U_i|} \tag{15}$$

$$\text{FWPC}(u,v) = \frac{\sum_{i \in I_{uv}} \lambda_i (r_{ui} - \overline{r}_u)(r_{vi} - \overline{r_v})}{\sqrt{\sum_{i \in I_{uv}} \lambda_i (r_{ui} - \overline{r}_u)^2 \sum_{i \in I_{uv}} \lambda_i (r_{vi} - \overline{r}_v)^2}} \tag{16}$$

4.3 Variations in Selecting Peer Groups

The effectiveness of a recommender system is significantly influenced by both the neighborhood size and the criteria used for neighborhood selection. There are various ways in which one can select the peers of a target user or item. The neighborhood selection is made in two phases: off-line phase (also called pre-prediction filtering of neighbors) and online phase (during prediction).

Pre-prediction phase
It is irrelevant to store all non-zero similarity values between each pair of users or items, particularly in a very large-scale recommender system. Therefore, this phase causes neighborhood-based approaches within the realm of practicality. Usually, the following ways are utilized.

- Top-N: In this approach, only a list of top-N nearest neighbors is retained for each user or item. The value of N should be chosen cautiously in order to satisfy conflicting goals, such as memory requirement and coverage.
- Threshold-based: Here, only those neighbors have selected whose similarity weight values are higher than a given threshold. Further, the value of the threshold must be chosen very carefully and sometimes challenging to decide.

Neighbors during the predictions
After selecting the neighborhood of each user or item, the estimated value of rating for a user-item combination is predicted with k-nearest neighbors. The optimal value of k is normally dependent on data and calculated empirically. When the value of k is small (e.g., $k < 20$), then the recommendations are less accurate; thus, if we increase the value of k, then prediction accuracy will increase until a certain point (e.g., $k > 50$).

5 Rating Matrix

The user-item interactions are stored in the form of a matrix, called the rating matrix (denoted by R). The order of the rating matrix is $m \times n$ for a system with m users and n items. The type of interactions between users and items varies from one application to another but can be broadly categories into three categories: scalar, binary, and unary responses. Scalar responses may further be categorized into three types: continuous ratings, interval-based ratings, and ordinal ratings. It is noteworthy to understand that the design of a recommender system is influenced by the way of collecting user-item interactions. Additionally, there exist various ways in which these ratings can be obtained.

5.1 Continuous Ratings

Here, the preferences of users are specified on a continuous scale, which can demonstrate interest/disinterest for items at a fine-grain level. For example, Jester Joke RS enables users to provide rating values between -10 and $+10$ with any real value. The major problem with such continuous rating scales is that the users have to choose among infinite possible values, which are somehow cumbersome; therefore, such systems are very uncommon in a real-world scenario.

5.2 Interval-Scaled Ratings

In such ratings, a set of ordered integers are utilized for quantifying the interest of users. The size of the set may vary according to the application at hands such as five-point or seven-point or ten-point scale, albeit most of the systems use a five-point scale. The numerical integer values might be chosen wisely, for example, a five-point scale may be either $\{1, 2, 3, 4, 5\}$ or $\{-2, -1, 0, 1, 2\}$. The following assumptions are usually associated with such ratings.

- The separations between rating values are defined via designated numerical values.
- Usually, rating values are equidistant.

In real-world RSs, interval-based ratings are very popular; Fig. 4 illustrates two examples of five-point rating scales along with the semantic explanations used by different e-commerce portals such as (a) Amazon (b) Flipkart.

There exist systems which provide an unequal number of positive and negative preferences, therefore, results in an *unbalanced rating scale*. For example, Netflix

Fig. 4 Examples of five-point interval-based ratings with semantic explanation, **a** Amazon, **b** Flipkart

uses a five-point scale in which the last three rating values (i.e., 3, 4, and 5) are positive, whereas the first two ratings (1 and 2) are negative ratings.

5.3 Ordinal Ratings

These ratings are much analogous to the interval-based ratings and may be considered as a special case of interval-based ratings. Here, a set of ordered categorical values are used for modeling choices of users, hence called ordinal ratings (similar to the concept of ordinal attributes). For example, the set of ordinal values {very good, good, neutral, bad, very bad} may be used for achieving the desired goal. One important distinction from interval-based ratings is that the ordinal values may or may not be equidistant from each other. However, in real-world scenarios, the ordinal values may always be designated to equidistant values similar to interval-based ratings. In order to avoid any bias from rating design, in most cases, the numbers of positive and negative ratings are the same. If the number of ratings is odd, then there may be the "Neutral" option present, while in case of an even number of ratings, the neutral choice is not available. A rating system where neutral choice is not present is known as the *forced-choice method*.

5.4 Binary Ratings

Here, only two options are available for users to describe their interests; also, these two options are corresponding to +ve and −ve responses from users. This rating system is also a very special case of interval-based (or ordinal ratings) and does not permit users for neutral choice; hence, it is an example of the forced-choice method. For example, the Pandora music portal facilitates its users to either like or dislike a specific music item [23].

5.5 Unary Ratings

There are two different points of view for understanding the interpretation of the unary ratings. According to the first view, unary ratings are a very special case of ratings where a user can only specify a positive inclination toward items, but there is no way to specify negative inclination. For example, the like button of Facebook is the real-world example of a unary rating system [24]. Secondly, unary ratings may also be used to simulate the activities of users (such as browsing behaviors and purchase histories) within the system in the form of implicit feedback data [25–27]. In the case of implicit datasets, users do not specify the ratings explicitly, but the preferences are derived from their actions within the system. Sometimes,

	i1	i2	i3	i4	i5
u1	4			1	5
u2		2			3
u3			1	4	
u4	3		5		

(a)

	i1	i2	i3	i4	i5
u1	1			1	1
u2		1			1
u3			1	1	
u4	1		1		

(b)

Fig. 5 Example of **a** explicit ratings and corresponding, **b** implicit ratings

unary ratings are very helpful under certain cases and streamline the design and development of the recommender systems. Figure 5 illustrates an example of implicit ratings derived from the corresponding explicit ratings.

6 Characteristics of the Rating Matrix

Generally, real-world datasets satisfy the following key properties related to the distribution of ratings in the UI-matrix.

6.1 Sparsity

In real-world datasets, the user-item matrix is very sparse since users specify ratings usually for a small fraction of items only. The sparsity problem occurs in almost all real-world datasets and results in impediment for effective recommendations. The sparsity of a dataset is defined as the ratio of unspecified ratings to the total number of entries in the user-item matrix and calculated as follows.

$$\text{sparsity} = 1 - \frac{|R|}{|U| \times |I|} \tag{17}$$

Table 4 calculates the sparsity of the benchmark rating datasets from the MovieLens Web site, which are collected and made publically available by the GroupLens research lab [28]. Furthermore, it is clear from the real-world example of MovieLens datasets that the sparsity problem's volume is also increasing over time, as depicted in Fig. 6. The sparsity problem makes the process of finding a neighborhood very difficult due to very few numbers of common ratings between users or items. There

Table 4 Sparsity problem in MovieLens datasets

Dataset	Release date	Brief description	Sparsity level (%)
ML-100K	04/1998	943 users have provided 100,000 ratings on 1682 movies	$1 - \frac{100,000}{943*1682} = 93.696\%$
ML-1M	02/2003	6040 users have provided 1000,209 ratings on 3900 movies	$1 - \frac{1,000,209}{6040*3900} = 95.754\%$
ML-10M	01/2009	71,567 users have provided 10,000,054 ratings on 10,681 movies	$1 - \frac{10,000,054}{71,567*10,681} = 98.692\%$
ML-20M	04/2015	7,138,493 users have provided 20,000,263 ratings on 27,278 movies	$1 - \frac{20,000,263}{138,493*27,278} = 99.470\%$

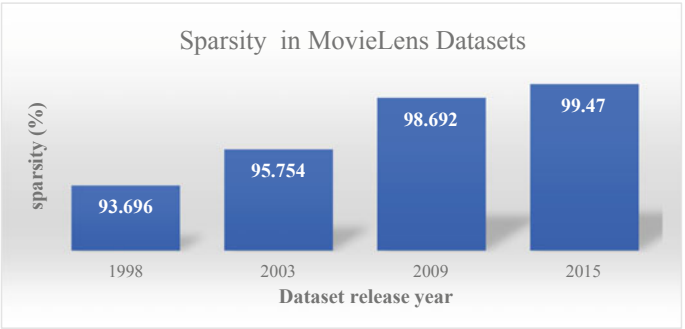


Fig. 6 Sparsity problem with respect to time for MovieLens datasets

are different approaches available in the literature for dealing with the sparsity problem. One simple solution is to utilize additional information (other than UI-matrix) about users such as age, gender, etc., in order to define the affinity between users. For example, research [29] presented an example of a hybrid approach for defining the neighborhood by utilizing the demographic information of users. Another common way to deal with the sparsity problem is to replace missing entries with default values [7, 30] such as user/item's average rating or central value of the rating scale. Furthermore, the content information may also be used to fill the missing values, and such techniques are considered to be more trustworthy.

6.2 The Long-Tail Property

In real-world datasets, ratings received by items usually show a specific characteristic, which is called long-tail property. This property demonstrates that the distribution of ratings among items is highly skewed, i.e., only a limited number of items are rated

frequently, and such items are called popular items. Furthermore, a huge number of items are rated rarely. We have evaluated this property for MovieLens dataset by plotting movie ids (on *X*-axis) against the total number of ratings received by movies (on the *Y*-axis); also, movie ids are arranged in decreasing order of frequency by which the movie is rated. Figure 7 illustrates this property for the MovieLens dataset. It is noteworthy to understand that the recommendation process has been affected by long-tail property in many situations, such as

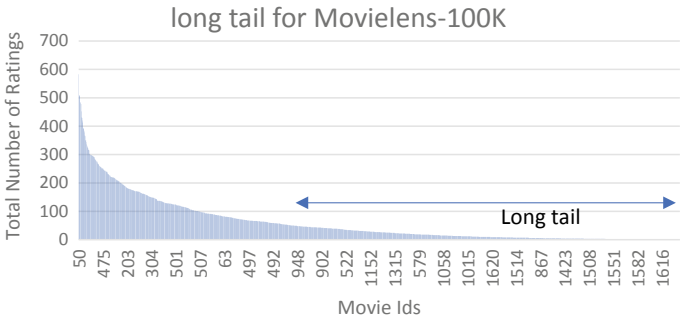
- RS service providers may target items that are present in long-tail due to a larger profit margin in comparison with the items that have high frequency and are competitive because of their popularity. Research [31] discusses the cases of a few e-commerce portals targeting items in the long-tail.
- Since items in the long-tail received very few ratings, therefore, it is hard to provide recommendations in the long-tail or predictions are misleading. In order to tackle such situations, the overall recommendation process must be modified for relevant recommendations [32, 33].

Table 5 provides a list of movies that have been rated highest for each dataset, and such data may be useful under specific situations.

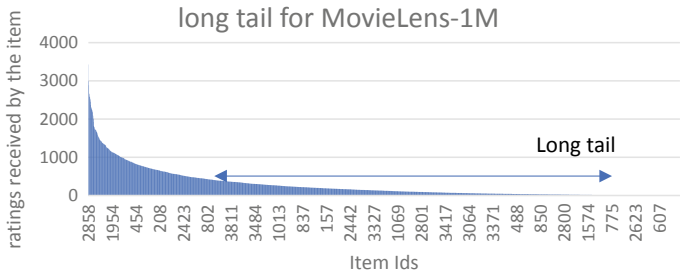
6.3 Cold-Start Problem

When new users or items are added to a recommender system, then there are no rating data available for these newly added users or items. In such cases, it becomes much more difficult for the system to provide recommendations for these newly added users or items by applying traditional collaborative filtering methods [34]. However, other recommendation approaches, such as content-based or knowledge-based, are more robust with respect to cold starts. This problem is linked with the sparsity problem in the UI-matrix [35]; further, there are three cases of cold-start problem: new item, new user, and new community [36].

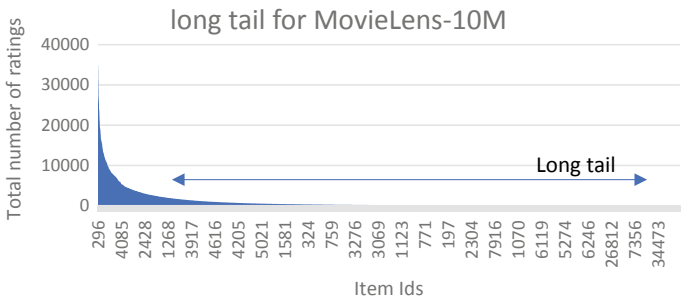
- New item: It appears when new items are added into the system and do not have initial ratings; therefore, it becomes very hard to recommend these newly added items [37].
- New users: It appears when new users enter into the system and have not rated any items; therefore, they cannot receive any personalized recommendation [38, 39].
- New community: It refers to the initial struggle that is required to activate the system for reliable recommendations, also known as recommendations on new items for new users [40].



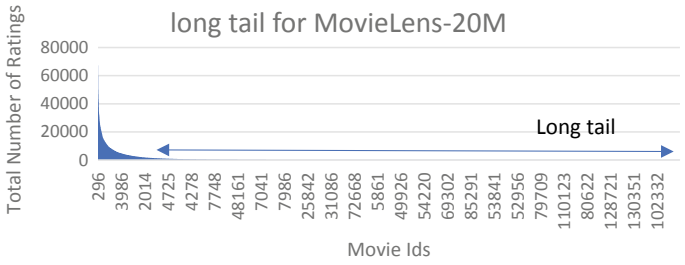
(a)



(b)



(c)



(d)

Fig. 7 Long-tail property for real-world datasets, **a** MovieLens-100K, **b** MovieLens-1M, **c** MovieLens-10M, **d** MovieLens-20M

Table 5 Highest-rated items for each dataset in MovieLens

Dataset	Movie Id	Frequency
MovieLens-100K	50	583
MovieLens-1M	2858	3428
MovieLens-10M	296	34,864
MovieLens-20M	296	67,310

References

1. Su X, Khoshgoftaar TM (2009) A survey of collaborative filtering techniques. Adv Artif Intell 2009(Section 3):1–19
2. Ekstrand MD (2011) Collaborative filtering recommender systems. Found Trends® Hum-Comput Interact 4(2):81–173
3. Shi Y, Larson M, Hanjalic A (2014) Collaborative filtering beyond the user-item matrix: a survey of the state of the art and future challenges. ACM Comput Surv 47(1):1–45
4. Ben Schafer J, Frankowski D, Herlocker J, Sen S (2007) Collaborative filtering recommender systems. In: Brusilovsky P, Kobsa A, Nejdl W (eds) The adaptive web: methods and strategies of web personalization. Springer, Berlin, Heidelberg, pp 291–324
5. Lee J, Sun M, Lebanon G (2012) A comparative study of collaborative filtering algorithms, pp 1–27
6. Candillier L, Meyer F, Boullé M (2007) Comparing state-of-the-art collaborative filtering systems. Mach Learn Data Min Pattern Recognit, pp 548–562
7. Breese JS, Heckerman D, Kadie C (1998) Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the 14th conference on uncertainty in artificial intelligence, vol 461, No 8, pp 43–52
8. Joaquin D, Naohiro I (1999) Memory-based weighted-majority prediction for recommender systems. Res Dev Inf Retr
9. Nakamura A, Abe N (1998) Collaborative filtering using weighted majority prediction algorithms. In: Proceedings of the fifteenth international conference on machine learning, pp 395–403
10. Getoor L, Sahami M (1999) Using probabilistic relational models for collaborative filtering. Work Web Usage Anal User Profiling
11. Ungar L, Foster D (1998) Clustering methods for collaborative filtering. In: AAAI workshop on recommendation systems, pp 114–129
12. Chen Y-H, George EI (1999) A Bayesian model for collaborative filtering. In: Proceedings of the 7th international workshop on artificial intelligence and statistics, No 1
13. Goldberg K, Roeder T, Gupta D, Perkins C (August 2000) Eigentaste, pp 1–11
14. Ekstrand MD, Konstan JA (2019) Recommender systems notation: proposed common notation for teaching and research
15. Goldberg D, Nichols D, Oki BM, Terry D (1992) Using collaborative filtering to weave an information tapestry. Commun ACM 35(12):61–70
16. Herlocker JON, Riedl J (2002) An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Inf Retr Boston, pp 287–310
17. Herlocker JL, Konstan JA, Borchers A, Riedl J (1999) An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval—SIGIR '99, pp 230–237
18. Pirotte A, Renders J-M, Saerens M et al (2007) Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Trans Knowl Data Eng 3:355–369
19. Last.fm, Play music, find songs, and discover artists. Available: https://www.last.fm/. Accessed 06 June 2019

20. Good N et al (1999) Combining collaborative filtering with personal agents for better recommendations. In: Proceedings of the sixteenth national conference on artificial intelligence and the eleventh innovative applications of artificial intelligence conference, pp 439–446
21. Ma H, King I, Lyu MR (2007) Effective missing data prediction for collaborative filtering. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval, pp 39–46
22. Bell R, Koren Y, Volinsky C (2007) Modeling relationships at multiple scales to improve accuracy of large recommender systems. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining, pp 95–104
23. Abdi H (2006) Abdi-KendallCorrelation2007-pretty, pp 1–7
24. Facebook. Available: https://www.facebook.com/. Accessed 18 June 2019
25. Hu Y, Koren Y, Volinsky C (208) Collaborative filtering for implicit feedback datasets. In: Proceedings of the 2008 eighth IEEE international conference on data mining, pp 263–272
26. Hsieh C-J, Natarajan N, Dhillon IS (2015) PU learning for matrix completion. In: Proceedings of the 32nd international conference on machine learning, vol 37, pp 2445–2453
27. Oard DW, Kim J et al (1998) Implicit feedback for recommender systems. In: Proceedings of the AAAI workshop on recommender systems, vol 83
28. MovieLens, GroupLens. Available: https://grouplens.org/datasets/movielens/. Accessed 22 Dec 2018
29. Pazzani MJ (2000) A framework for collaborative, content-based and demographic filtering. Framework 13:393–408
30. Deshpande M, Karypis G (2004) Item-based top-N recommendation algorithms. ACM Trans Inf Syst 22(1):143–177
31. Anderson C (2006) The long tail: why the future of business is selling less of more. Hachette Books
32. Park Y-J, Tuzhilin A (2008) The long tail of recommender systems and how to leverage it. In: Proceedings of the 2008 ACM conference on recommender systems, pp 11–18
33. Yin H, Cui B, Li J, Yao J, Chen C (2012) Challenging the long tail recommendation. Proc VLDB Endow 5(9):896–907
34. Schein AI, Popescul A, Ungar LH, Pennock DM (2002) Methods and metrics for cold-start recommendations. In: Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval, pp 253–260
35. Lika B, Kolomvatsos K, Hadjiefthymiades S (2014) Facing the cold start problem in recommender systems. Expert Syst Appl 41(4 Part 2):2065–2073
36. Bobadilla J, Ortega F, Hernando A, Bernal J (2012) A collaborative filtering approach to mitigate the new user cold-start problem. Knowl-Based Syst. 26:225–238
37. Park S-T, Chu W (2009) Pairwise preference regression for cold-start recommendation. In: Proceedings of the third ACM conference on recommender systems, pp 21–28
38. Rashid AM et al (2002) Getting to know you: learning new user preferences in recommender systems. In: International conference on intelligence, user interfaces, Proceedings of IUI, pp 127–134
39. Rashid AM, Karypis G, Riedl J (2008) Learning preferences of new users in recommender systems: an information-theoretic approach. SIGKDD Explor Newsl 10(2):90–100
40. Lam XN, Vu T, Le TD, Duong AD (2008) Addressing cold-start problem in recommendation systems. In: Proceedings of the 2nd international conference on ubiquitous information management and communication, pp 208–211

Chapter 8
Classification of Arabic Text Using Singular Value Decomposition and Fuzzy C-Means Algorithms

Ahmad T. Al-Taani and Sami H. Al-Sayadi

1 Introduction

Due to the exponentially increasing amount of available natural language electronic text over the past two decades, text classification is an important area in NLP and used in many applications, like data mining, spam e-mail filtering, information retrieval, online newspapers or Web site, opinion mining, automatic indexing, and recently used for detecting violence in social media. Text classification helps to classify huge amount of texts on the Web, which requires more time and cost classified by human experts.

There are many challenges faced researchers in Arabic text classification area; these include morphology complexity, no standard datasets for evaluation purposes, stemming, and widely used synonyms in the Arabic language [1].

Recently, researchers proposed many supervised approaches for Arabic text classification. Examples include support vector machine (SVM) [2–6], decision tree [2, 7] and k-nearest neighbor (KNN) [2, 5]. All authors used the vector space model for representation which was introduced by Salton et al. [8]. Vector space model has some limitations such as high dimension of the input space. This limitation should be tackled to prevent hardware limitation and execution time complexity, which lead to misclassification.

Fuzzy c-means algorithm (FCM) is a popular unsupervised machine learning algorithm. It was proposed by Roubens in 1978 [9]. FCM classifier uses the degree of the membership function (μ); it allows one document of a collection to belong to two or more clusters; this feature is one of the reasons that motivated us to use it in Arabic text classification because the root of the Arabic words has many possible meanings and may lead to many possible clustering. Thus, FCM solves the issues of Arabic document classification [10]. FCM was first used for document clustering in 2010 [11], but the large scale of datasets is still a challenge for the FCM algorithm.

A. T. Al-Taani (✉) · S. H. Al-Sayadi
Yarmouk University, Irbid, Jordan
e-mail: Ahmadta@yu.edu.jo

S. H. Al-Sayadi
e-mail: samialsayadi@ses.yu.edu.jo

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_8

To tackle the limitations of vector space model mentioned earlier and to increase the value of data, we applied the singular value decomposition (SVD) reduction technique. The proposed technique enhanced the performance of the classifier by reducing the high dimensionality, increasing the value of data, and increasing the execution speed for the FCM classifier. In the implementation, Gensim [12] and Sklearn tools [13] are used. Gensim is an NLP library implemented in Python, and Sklearn is a classification and dimensionality reduction tool.

Remaining parts of this paper are presented as follows: Sect. 2 presents the critical analysis of the previous work on Arabic text classification. A full description of the methodology is discussed in Sect. 3. Section 4 presents and discusses the experimental experiments as well as the comparison of the results with other approaches. Finally, conclusion directions for future work are presented in Sect. 5.

2 Literature Review

During the past twenty years, many approaches have been developed for Arabic text clustering and classification. The recent research in those areas is critically analyzed and assessed in this section.

Singh et al. [14] applied the traditional and the improved heuristic k-means algorithm for documents clustering as well as FCM algorithm. The authors evaluated their approach on some standard datasets. The results showed that the use of TF-IDF and stemming improved the clustering process.

Ludwig [15] proposed parallelization of FCM algorithm using MapReduce paradigm. The approach achieved competitive results in terms of accuracy compared to other clustering algorithms.

El-Ameen and Shaout [10] proposed the use of fuzzy representation of a single word to solve the uncertainty of the meaning of an Arabic word. No results are presented by the authors, but they draw a conclusion that fuzzy representation of Arabic documents can solve the ambiguities of words possible senses.

Harish et al. [16] introduced a method of interval-valued representation by using standard deviation and FCM. Results showed that the method obtained good classification results compared to the existing methods.

Deng et al. [17] improved FCM by editing the distance algorithm and reducing the high dimensionality of text vector and used it for classification. According to the authors, the improved approach gave more accurate clustering results in comparison with the traditional FCM algorithm.

Li et al. [18] suggested the use of neural network and SVD for text classification; they compared the impact of SVD on English text classification. Experimental results showed that combining neural network and SVD in classification gives better classification that traditional methods with vector space model.

Samat et al. [19] and Abidin et al. [20] used SVD for dimensionality reduction in unsupervised methods such as k-means for English and Malay text classification. The proposed approaches gave promising results as the authors reported.

Oliynyk et al. [21] combined FCM with SVD to classify single-neuron activity. The results showed high classification accuracy.

Hasan and Matsumoto [22] proposed the use of hierarchical clustering algorithm of k-means and logical structure between these clusters, after applying SVD to extract key features from the term weights in document-term matrix. They concluded that the results are better when conceptual similarity is used instead of term similarity.

Al-Anzi and AbuZeina [2] used SVD to extract textual features based on LSI and used well-known classifiers such as NB, KNN, random forest, SVM, TD, and cosine measure to classify the documents. Alqabas Newspaper corpus is used to evaluate the performance of their approach. The results proved that SVM performed (with the best accuracy of 82.5%) better than NB, KNN, random forest, SVM, TD, and cosine measure.

Elghannam [3] proposed the use of bigram alphabet and chi-square to construct feature terms and for feature selection, respectively. The authors reported that their approach obtained high accuracy compared to other Arabic text classification approaches.

Larabi and Alalyani [4] used the firefly algorithm for feature selection and SVM classification. The Open Source Arabic Corpus (OSAC) is used for evaluation. The results outperformed three previous approaches with a precision of 0.99.

Bahassine et al. [7] proposed the use of chi-square for feature selection to enhance the classification performance. Then SVM and decision tree classifiers are used for Arabic documents classification. A dataset of 5070 Arabic documents is used for evaluation. The authors reported that their approach obtained better results with an F-measure of 90.5 compared to three traditional feature selection methods.

Al-Salemi et al. [5] introduced a new multi-label Arabic news dataset named RTAnews. This dataset is used for Arabic text categorization using different classification algorithms such as SVM, KNN, and random Forest. The results showed the performance of each of these algorithms.

AbuZeina and Al-Anzi [23] suggested the use of Euclidean distance for Arabic text classification and used linear discrimination analysis for feature extraction while used document frequency for feature selection. The authors evaluated the proposed approach with SVM, KNN, NN, and NB classifiers. The results gave an accuracy of about 84% for both SVM and LDA.

Mesleh [6] conducted a comparison of different feature selection techniques using SVM classifier for the classification of Arabic articles. An Arabic corpus of 7842 documents is used to evaluate the robustness of the classifier. The author reported that the results of using Fallout FSS and chi-square metrics with SVM are better than other metrics for Arabic text classification.

Al-Anzi and AbuZeina [24] used Markov chain model to score Arabic documents for text classification purposes. To evaluate their approach, they used Arabic corpus which consists of 11,191 documents collected from Alqabas Newspaper. Reported results showed that the effectiveness of the classifier was better using Markov chain model compared when using the latent semantic indexing (LSI) method.

Harrag and Al-Qawasmah [25] combined artificial neural networks (ANN) with SVD for Arabic text classification. An artificial Arabic corpus collected from Hadith

channel Web site is used for the evaluation. Results showed that the use of ANN with SVD obtained better results than the ANN alone.

Chantar and Corne [26] used binary PSO algorithm for feature selection and KNN classifier for Arabic text classification. To validate the approach, three classifiers are used: J48, naive Bayes, and SVM. Alj-News5 dataset is used for evaluation. Preprocessing steps are performed on the dataset before feature selection and text classification; these include removing stop words, non-Arabic characters, digits, and diacritics. The authors reported that the PSO performed good for feature selection.

Ouatik and Alaoui [27] used deep learning for categorization of Arabic text. The deep learning approach obtained good performance in comparison with three algorithms, SVM, decision tree, and naive Bayes, applied on the CNN Arabic news dataset.

A summary of previous work reviewed is presented in Table 1.

After intensive investigations of the previous studies, we found SVM outperformed all competitive approaches in text classification area, and chi-square outperformed all competitive methods in feature selection. The existing researches used supervised methods for text classification with different feature selection methods. In this research, we proposed a new method for Arabic text classification based on unsupervised methods.

Table 1 A summary of key previous works in Arabic text classification

References	Dataset	Feature Selection	Method	Performance
[3]	Al Jazeera News, #Doc: 1500 Classes: 5	Chi-square	SVM-SMO	Precision: 0.94 Recall: 0.94 *F*-measure: 0.949
[4]	OSAC #Doc: 5843 Classes: 6	Firefly algorithm	SVM	*P*-measure: 0.99
[7]	CNN Arabic #Doc: 5070	Imp-Chi-square	SVM, DT	*F*-measure: 90% for SVM *F*-measure: 86% for DT
[5]	RTAnews Arabic #Doc: 23837 Classes: 40	Chi-square	SVM, KNN, RFBoost	*F*-measure: 0.73 for LP-SVM *F*-measure: 0.72 for RFBoost
[6]	Artificial #Doc: 7842 Classes: 10	Chi, lg, Gss, Ngl, Mi, Or, Gsss, Df, Pwr, Acc, Acc2, *F*1, Pr, Re, Fo, and Er	SVM	Precision: 95.54 with Chi Precision: 94.83 with Fo
[25]	HadithNews #Doc: 453 Classes: 14	SVD	NN	Accuracy: 52%

3 Methodology

The proposed approach consists of four phases: Text preprocessing is performed in phase one, feature extraction is done in phase 2, and then, features extracted in phase 2 are feed into phase 3 for classification. In phase 4, the results are evaluated using two standard datasets for Arabic text classification: CNN Arabic news and Al Jazeera news. Figure 1 shows the structure of the classification approach.

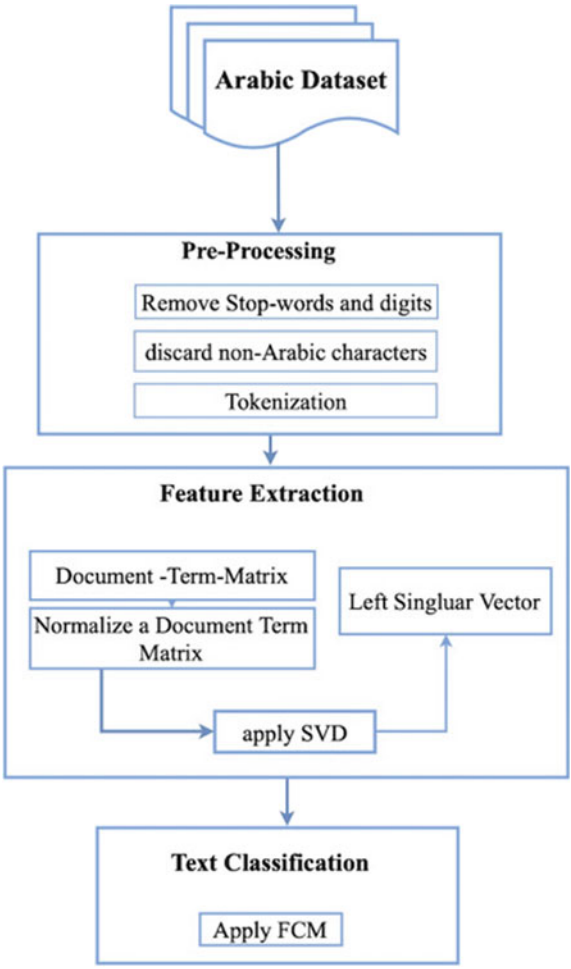


Fig. 1 Structure of the classification approach

3.1 Text Preprocessing

Reducing features in the dataset is a crucial step in order to reduce the complexity of text classification methods and to obtain high accurate classification results. Two preprocessing operations are performed on the two used datasets (CNN Arabic news and Al Jazeera news); these include: stop-words removal, elimination of non-Arabic characters, digits, and special characters, and tokenization. Stop words removed include pronouns, auxiliary verbs, preposition, determiners, articles, and conjunction. We extended and used the stop word list adopted by El-khair [28] which consists of 1337 stop words. We used the NLTK library in the tokenization step to convert documents into sequences of tokens instead of characters.

3.2 Feature Extraction

Feature extraction is a preprocessing step and plays important role is document classification. Feature extraction purpose is to convert the free-text sentences into a set of words and enriching their semantic meaning. This is done through the evaluation of their weights in different related domains. Three subtasks are involved in this stage: representing document-term matrix, normalizing document-term matrix, and applying singular value decomposition.

Representing Document-Term Matrix
Document-term matrix (f_{ij}) is a mathematical process to compute the frequency of terms that occur in a text. In this study, columns represent documents and rows refer to tokens in documents.

Normalizing Document-Term Matrix
Document-term matrix is normalized to local normalization factor based on simple logarithmic using Formula (1) and to global normalization factor based on entropy using Formula (2) [29].

$$\text{local}_{\text{factor}} = 1 + \log f_{ij} \tag{1}$$

$$\text{global}_{\text{factor}} = 1 + \sum_{j=1}^{n_{\text{doc}}} \text{Entropy} \tag{2}$$

Before applying entropy on global factor, we find the probability of terms in jth document. Let P_{ij} be the count of the words normalized to sum to one over the rows, Formula (3):

$$p_{ij} = \left(\frac{{f_i}^{\text{T}}}{\sum_{j=1}^{n_{doc}} f_{ij}} \right)^{\text{T}} \tag{3}$$

Subsequently, the log-entropy weighting is calculated as follows (Formula 4):

$$\text{Entropy} = \frac{p_{ij} \log p_i}{\log (n_{\text{doc}})} \tag{4}$$

From Formulas 1 and 2, we compute the normalized document-term matrix by:

$$\text{Normalize}_{\text{DTM}} = (\text{global}_{\text{factor}} * \text{local}_{\text{factor}}{}^{\text{T}})^{\text{T}} \tag{5}$$

Applying Singular Value Decomposition

Salton and Buckley [30] used vector space model (VSM) for feature extraction, but this representation has high-dimensional input space. This dimensionality increases execution time complexity and hardware limitations. In Arabic language and others, many words have the same meaning, and sometimes a word has a different meaning, so it is affected to measure the similarity between documents.

For text clustering, we used the SVD method to reduce time and to enhance classification performance. The goal of using SVD is to reduce and convert the high-dimensional space into low semantic dimensional space and to find the relationship between the terms and documents for Arabic text classification. SVD is a theorem which states that an $m \times n$ matrix A can be represented as a product of three matrices: left singular vector, $m \times k$ orthogonal matrix U, singular vector, $k \times k$ diagonal matrix S, and right singular vector, $k \times n$ orthogonal matrix V. This is represented in Formula (6) [31]:

$$A_{mn} = U_{mk} \times S_{kk} \times V_{kn}^{\text{T}} \tag{6}$$

where k is the rank of the matrix.

In this work, we used only the left singular vector (U_{mk}) matrix; it is represented as shown in Formula (7):

$$d^{\wedge} = N_{\text{DTM}}^{\text{T}} * U_{mk}. \tag{7}$$

where $N_{\text{DTM}}^{\text{T}}$ is the transpose of the normalized document-term matrix and $d^{\wedge}$ is the new document vector. This document is then used by the FCM classifier to classify documents. The process of applying the SVD is represented in Fig. 2.

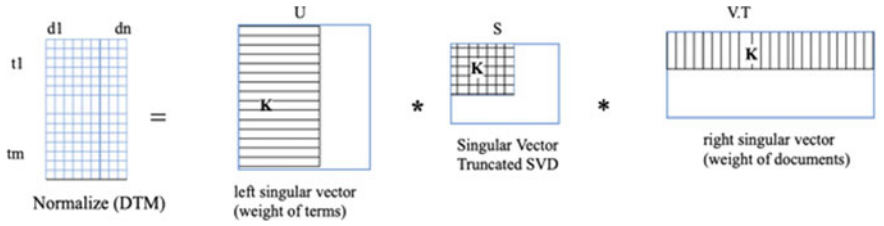


Fig. 2 SVD representation

3.3 Applying Fuzzy C-Means Classifier

FCM is an unsupervised learning classifier proposed by Roubens [9]. It is affected by feature weights; if the feature weights are high dimensional, then it performs poorly. Therefore, it is important to reduce high-dimensional space to ensure the good performance of the FCM algorithm. So, we used the SVD decomposition technique to reduce the high-dimensional space and to enhance the performance of the FCM algorithm. This classifier uses the degree of membership function (μ), i.e., the FCM allows one document of collections to belong to two or more clusters.

The coverage of the FCM may be either local or global minimum of the object function $J(D, U, C)$ which is represented as follows:

$$J(D, U, C) = \sum_{i=1}^{c} \sum_{j=1}^{N} {u_i}^m d^2 \left(C_i, D_j\right) \tag{8}$$

where ${u_{ij}}^m$ is the member function and $d^2(C_i, D_j)$ is the Euclidean distance, the distance between cluster centers C_i and the data point D_j.

4 Experimental Results

The Arabic text classification approach is evaluated on classifying Arabic texts using the CNN Arabic news and Al Jazeera Arabic news (Alj-News5) datasets. Detailed description of these datasets, the analysis and discussion of the result, and comparison of the proposed approach with the previous works are presented in this section.

4.1 Feature Reduction

We applied SVD for extracting the features for building the FCM classifier by selecting the left singular vector (U) only. To choose the topmost orthogonal matrix features, we set up k-ranking approximation as $k = 7$ for Alj-News5 dataset and $k = 8$ for CNN Arabic news dataset. This technique has proved to achieve high accuracy classification results compared to other reduction methods.

4.2 Arabic Datasets

We chose the CNN dataset and Alj-News5 dataset since they are used in several previous researches for Arabic text classification which enables us to compare our

results with these researches. The CNN Arabic news dataset consists of 5070 documents [32]. The documents are distributed among six classes: Business with 836 documents, Entertainments with 474 documents, Middle East news with 1462 documents, Science and Technology with 526 documents, Sports with 762 documents, and Worlds news with 1010 documents. Alj-News5 is an Arabic dataset which consists of 1500 documents [33]. The documents are divided into five classes; each class contains 300 documents: Sport, Art, Science, Economy, Politics.

4.3 Performance Evaluation Measures

Precision, recall, and F-measure are used to evaluate the performance of the proposed FCM classifier. These measures are computed as shown in Formulae (9), (10), and (11), respectively [32].

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{9}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{Falsely Negative}} \tag{10}$$

$$F\text{-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

Tables 2 and 3 show the detailed results on the two datasets. Results showed significant contributions on enhancing text classification by using just left singular vector of SVD for feature extraction and FCM classifier for text classification. Best performance achieved on Science class with a recall of 100% and an F-measure of 96%, while Politic class achieved a lowest recall (48%) and f-measure (65%) on Alj-News5 dataset (Table 3). On the other hand, using the CNN Arabic dataset, best performance achieved on Sport class with a recall of 100% and an F-measure of 90%, while Entertainment class achieved a lowest recall (10%) and F-measure (12%) (Table 2).

Table 2 Measures on CNN news dataset

Categories	Precision	Recall	F-measure
Business	0.92	0.76	0.832
Entertainment	0.15	0.1	0.12
Middle East	0.44	0.95	0.6014
SciTech	0.85	0.37	0.5155
Sport	0.81	1	0.8950
World	0.44	0.55	0.4888
Average	0.6016	0.6266	0.6118

Table 3 Measures on Alj-News5 dataset

Categories	Precision	Recall	*F*-measure
Art	**0.82**	0.86	0.839
Economy	0.48	**1**	0.649
Politics	**1**	0.48	0.649
Sciences	0.92	**1**	**0.958**
Sport	0.54	**1**	0.7012
Average	0.752	0.868	0.76

4.4 Comparison with Other Approaches

We compared the SVD-FCM proposed approach (unsupervised approach) with the previous supervised approaches using SVM, decision trees, naive Bayes [26, 27], and polynomial networks [34]. The comparison of the results and other approaches on the same datasets is presented next. Chantar and Corne [26] and Al-Tahrawi and Al-Khatib [34] results using Alj-News5 are summarized in Figs. 3, 4, and 5. After careful inspection of the results, it can be noticed that the proposed approach outperformed other approaches even though supervised approaches should perform better, while the proposed approach in an unsupervised approach. FCM is the proposed approach, DT is the decision tree, NB is naive Bayes, SVM is support vector machine, and PN is polynomial networks. These figures show that the proposed SVD-FCM approach achieved good results in comparison with the other approaches.

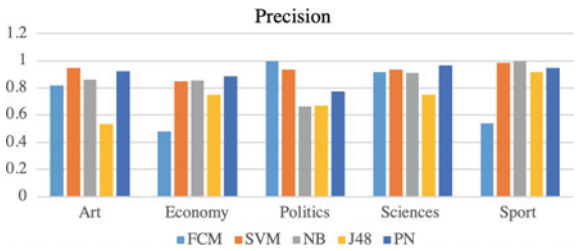


Fig. 3 FCMs precision versus others on Alj-News5 dataset

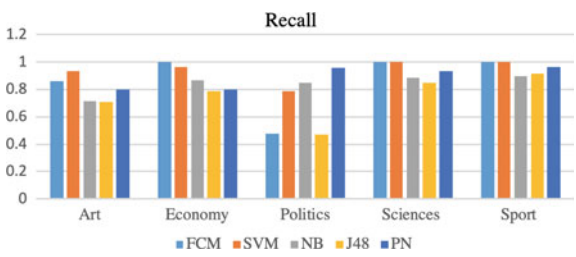


Fig. 4 FCMs recall versus others on Alj-News5 dataset

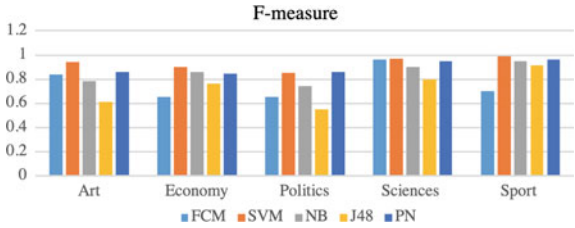


Fig. 5 FCMs *F*-measure versus others on Alj-News5 dataset

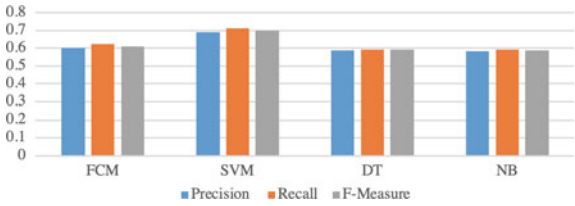


Fig. 6 FCMs precision, recall, and *F*-measure versus others on CNN dataset

Figure 6 presents the comparison of our results with the results of Ouatik and Alaoui [27] on the CNN Arabic news dataset.

In conclusion, the proposed approach outperformed the DT and NB algorithms on CNN Arabic news dataset; also, the proposed approach outperformed the decision tree DT and recall results with NB classifier on Alj-News5 dataset. SVM also outperformed all competitive approaches in text classification area. After the investigations of the proposed approach results, we can deduce that we added two basic contributions to the area of text classification. It achieved high-speed execution by using SVD-FCM classifier, and it achieved competitive results compared to supervised approaches on the same datasets.

5 Conclusions and Future Work

This paper proposed the use of SVD algorithm and FCM classifier for Arabic text classification. The SVD is used as reduction technique, and FCM is used for classification. The FCM approach has some limitations, which are high complexity time and high-dimensional space. To solve these limitations, we used SVD reduction technique to tackle the problem of high dimensionality of the space by converting it into semantic dimensional space, representing relationship between terms and documents. Since the FCM approach is based on degrees of membership function which is based on inverse-distance weight, and the root of the Arabic words have many possible meanings and may lead to many possible clustering, so the combination of SVD and FCM approaches produced high precision results in text classification area. The proposed approach has two main contributions. First, it reduced the high dimensionality which increased the value of data that helped to reach high-speed of execution by FCM with high performance of accuracy. Second, it reached competi-

tive results as an unsupervised method compared with results obtained by supervised approaches on the same datasets.

The proposed SVD-FCM approach achieved high performance accuracy compared with existing approaches evaluated on two datasets. The results proved that the use of SVD and FCM algorithms added a new contribution to the area of Arabic text classification. As a direction to future work, we suggest to apply this approach for detecting violence on social media.

References

1. Alwakid G, Osman T, Hughes-Roberts T (2017) Challenges in sentiment analysis for arabic social networks. Proc Comput Sci 117:89–100. https://doi.org/10.1016/j.procs.2017.10.097
2. Al-Anzi FS, AbuZeina D (2017) Toward an enhanced Arabic text classification using cosine similarity and latent semantic indexing. J King Saud Univ - Comput Inf Sci 29:189–195. https://doi.org/10.1016/j.jksuci.2016.04.001
3. Elghannam F (2019) Text representation and classification based on bi-gram alphabet. J King Saud Univ Comput Inf Sci. https://doi.org/10.1016/j.jksuci.2019.01.005
4. Larabi Marie-Sainte S, Alalyani N (2018) Firefly algorithm based feature selection for Arabic text classification. J King Saud Univ Comput Inf Sci. https://doi.org/10.1016/j.jksuci.2018.06.004
5. Al-Salemi B, Ayob M, Kendall G, Noah SAM (2019) Multi-label Arabic text categorization: a benchmark and baseline comparison of multi-label learning algorithms. Inf Process Manag 56:212–227. https://doi.org/10.1016/j.ipm.2018.09.008
6. Mesleh AMD (2011) Feature sub-set selection metrics for Arabic text classification. Pattern Recognit Lett 32:1922–1929. https://doi.org/10.1016/j.patrec.2011.07.010
7. Bahassine S, Madani A, Al-Sarem M, Kissi M (2018) Feature selection using an improved Chi-square for Arabic text classification. J King Saud Univ Comput Inf Sci. https://doi.org/10.1016/j.jksuci.2018.05.010
8. Salton G, Wong A, Yang C (1975) Information retrieval and language processing: a vector space model for automatic indexing. Commun ACM 18:613–620. https://doi.org/10.1145/361219.361220
9. Roubens M (1978) Pattern classification problems and fuzzy sets. Fuzzy Sets Syst 1:239–253. https://doi.org/10.1016/0165-0114(78)90016-7
10. El-Ameen A, Shaout A (2014) Fuzzy Arabic document classification. In: The international Arab conference on information technology, Nazwa University, Oman, Dec 2014
11. Win TT, Mon L (2010) Document clustering by fuzzy c-mean algorithm. In: Proceedings of the 2nd international conference on advanced computer control. ICACC 2010. vol 1, pp 239–242. https://doi.org/10.1109/ICACC.2010.5487022
12. Gensim. https://pypi.org/project/gensim. Last accessed 23 May 2019
13. Sklearn. https://pypi.org/project/scikit-learn. Last accessed 23 May 2019
14. Singh VK, Tiwari N, Garg S (2011) Document clustering using K-means, heuristic K-means and fuzzy C-means. In: Proceedings—2011 international conference on computational intelligence and communication systems, CICN 2011, pp 297–301. https://doi.org/10.1109/CICN.2011.62
15. Ludwig SA (2015) MapReduce-based fuzzy c-means clustering algorithm: implementation and scalability. Int J Mach Learn Cybern 6:923–934. https://doi.org/10.1007/s13042-015-0367-0
16. Harish BS, Prasad B, Udayasri B (2014) Classification of text documents using adaptive fuzzy C-means clustering. Adv Intell Syst Comput 235:205–214. https://doi.org/10.1007/978-3-319-01778-5_21
17. Deng J, Hu J, Chi H, Wu J (2010) An improved fuzzy clustering method for text mining. In: Proceeding of the 2nd international conference on networks security, wireless communications

and trusted computing, NSWCTC 2010, pp 65–69. https://doi.org/10.1109/NSWCTC.2010.23
18. Li CH, Park SC (2007) Neural network for text classification based on singular value decomposition. In: Proceedings of the 7th IEEE international conference on computer and information technology, pp 47–52. https://doi.org/10.1109/CIT.2007.4385055
19. Samat NA, Azrifah M, Murad A, Abdullah T, Atan R (2005) Malay documents clustering algorithm based on singular value decomposition. J Theor Appl Inf Technol, pp 180–186
20. Abidin TF, Yusuf B, Umran M (2010) Singular value decomposition for dimensionality reduction in unsupervised text learning problems. In: Proceedings of the 2nd international conference on education technology and computer (ICETC), pp 422–426
21. Oliynyk A, Bonifazzi C, Montani F, Fadiga L (2012) Automatic online spike sorting with singular value decomposition and fuzzy C-mean clustering. BMC Neurosci 13:96. https://doi.org/10.1186/1471-2202-13-96
22. Hasan M, Matsumoto Y (1999) Document clustering: before and after the singular value decomposition. Spec Interes Gr Nat Lang Process 4754
23. AbuZeina D, Al-Anzi FS (2018) Employing fisher discriminant analysis for Arabic text classification. Comput Electr Eng 66:474–486. https://doi.org/10.1016/j.compeleceng.2017.11.002
24. Al-Anzi FS, AbuZeina D (2018) Beyond vector space model for hierarchical Arabic text classification: a Markov chain approach. Inf Process Manag 54:105–115. https://doi.org/10.1016/j.ipm.2017.10.003
25. Harrag F, Al-Qawasmah E (2010) Improving Arabic text categorization using Neural Network with SVD. J Digit Inf Manag 8:233–239
26. Chantar HK, Corne DW (2011) Feature subset selection for Arabic document categorization using BPSO-KNN. In: Proceedings of the 2011 3rd World Congress on Nature & Biologically Inspired Computing, NaBIC 2011, pp 546–551. https://doi.org/10.1109/NaBIC.2011.6089647
27. Ouatik S, Alaoui E (2016) An efficient method based on deep learning approach for Arabic text categorization. In: International Arab conference on information technology, Morocco
28. El-khair IA (2006) Effects of stop words elimination for Arabic information retrieval : a comparative study. Int Inf 4:119–133
29. Pat M, Cho GE, Nelson S, Orum C, Janelle V, Mather L, Problem 4: term weighting schemes in information retrieval, 19
30. Salton G, Buckley C (1988) The types of Flatidae (Homoptera) in the Stockholm Museum described by Stl, Melichar, Jacobi and Walker. Insect Syst Evol 17:323–337. https://doi.org/10.1163/187631286X00251
31. Golub GH, Reinsch C (1970) Singular value decomposition and least squares solutions. Numer Math 14:403–420. https://doi.org/10.1007/BF02163027
32. Arabic-Corora -CNN Arabic dataset 2010. https://sourceforge.net/projects/ar-text-mining/files/Arabic-Corpora. Last accessed 1 Jan 2019
33. Arabic-Corora-Alj-News. https://filebox.vt.edu/users/dsaid/Alj-News.tar.gz. Last accessed 6 Mar 2013
34. Al-Tahrawi MM, Al-Khatib SN (2015) Arabic text classification using polynomial networks. J King Saud Univ - Comput Inf Sci 27:437–449. https://doi.org/10.1016/j.jksuci.2015.02.003

Chapter 9
Echo State Network Based Nonlinear Channel Equalization in Wireless Communication System

Saikat Majumder

1 Introduction

Signals transmitted over a communication channel are not only corrupted by noise but are also degraded by multipath interference. Multipath interference results in distortion called inter-symbol interference (ISI) which severely degrades the performance of communication system. In addition, the transmitted information may also suffer from nonlinear distortion introduced at various stages of transmitter and receiver equipments. Moreover, with steady increase in data rates in the last decade, severity of such impairments have become more significant. Addition of an adaptive equalizer enables reliable reception of transmitted symbols in nonlinear and ISI channels. Conventional methods of channel equalization utilize a transversal filter or lattice structure because of their simplicity in implementation. However, the performance of linear equalizers utilizing gradient-based algorithms is limited, specially in case of channels with deep spectral nulls and nonlinearity. For the recovery of symbols transmitted through such channels, complexity of the equalizer should be at least equivalent to the channel introducing the distortion. Research has shown that nonlinear equalizers significantly outperform linear equalizers in nonlinear and ISI channels.

In recent years, artificial neural networks (ANN)-based equalizers were introduced to provide effective solution in nonlinear channels. Inherent nonlinear architecture of ANN allows the receiver to have nonlinear decision boundaries which enable efficient classification of distorted digital signals [1]. In one of the earliest work on ANN-based equalizer, a multilayer perceptron (MLP) decision feedback equalizer was proposed and weighed against a LMS-based linear equalizer [2]. Chen et al. proposed channel equalizers consisting of MLP trained by back propagation algorithm in [3]. Superiority of MLP equalizers compared to established equalizers of the time were proven in [4, 5]. In order to reduce the computational complexity and training time of the neural network, alternate architecture in the form of functional link neural network (FLNN) was proposed by Pao [6]. Functional expansion in FLNN

S. Majumder (✉)
Department of Electronics and Communication, National Institute of Technology, Raipur, Raipur, Chhattisgarh, India
e-mail: smajumder.etc@nitrr.ac.in

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_9

is done using Gaussian or orthogonal polynomials like Legendre and Chebyshev. In [7], Patra et al. proposed Chebyshev polynomial-based FLNN for channel equalization. Work on FLNN-based equalizer was extended to Legendre neural network for equalization of QAM signal constellation in [8].

Radial basis function (RBF) equalizers offer a substitute framework to MLP and FLNN because of its close resemblance to Bayesian approach of equalization [5]. Many technique have been developed in literature applying RBF to channel equalization [9, 10]. These works show that RBF equalizer produce better results and has lesser computational complexity compared to maximum likelihood sequence estimator (MLSE) equalizer over time-varying fading channel.

Numerous nature-inspired optimization algorithm have been proposed in last decade and many of them were applied to the research on channel equalization. In many of these algorithms, nature-inspired algorithm replaces the conventional training algorithms like LMS, RLS or backpropagation. Nature-inspired algorithms use a swarm or population of possible solutions who combine their experience to explore the solution space and find the optimal solution [11]. Some of the recent works have applied particle swarm optimization [12, 13], directed search optimization [14], frog-leaping algorithm [15], moth-flame optimization [16] to the problem of training adaptive channel equalizer.

Besides ANN, recurrent neural networks (RNN) is a potent device which has both dynamic memory and flexible computational capabilities. In contrast to MLP, RNN exhibit dynamic behaviour due to presence of feedback connections. Studies show that RNNs are capable of superior estimation performance despite large training complexity [17]. However, training a RNN is inherently difficult and computationally slow. Because of the difficulties associated with training RNN, reservoir computing (RC) have recently attracted much attention. RC is considered a flavour of RNN, suited for temporal data processing tasks, which can store nonlinear transformations of the external input stimuli. ESN is one the popular forms of RC that provide an efficient computational model approximating nonlinear dynamical system. ESN does not have the training disadvantage of RNNs and can outperform Hessian-free trained RNNs [18].

In this chapter, we propose an ESN-based equalizer for a channel suffering from ISI and nonlinearity. Literature survey reveals that there has been little or no research on application of ESN for ISI channel equalization, other than some studies on equalization of nonlinear satellite channels [18, 19]. In addition, we compare the BER performance offered by some of the common reservoir types for the first time in literature.

Outline of remaining part of this chapter is as follows. In Sects. 2 and 3, we introduce ESN, its control parameters, training algorithm and its application in equalization of ISI and nonlinear channel. Section 4 describes some of the prevalent architecture of reservoirs and their design. We present the algorithm for ESN-based equalizer cum symbol detector for digital communication in Sect. 5 and evaluate its bit-error rate (BER) performance in Sect. 6. Sensitivity analysis of the parameters of ESN is performed and performance is compared to standard equalizer in literature. Finally, Sect. 7 is conclusion.

2 System Model and Equalization

The block diagram of an adaptive equalization scheme for signal transmitted through a nonlinear communication channel in shown in Fig. 1. Consider a data communication link which use binary phase shift keying (BPSK) modulation for transmission of random binary sequence $b(k)$. Each symbol in $b(k)$ is drawn randomly from the set $\{+1, -1\}$. This input $b(k)$ passes through a channel having impulse response $h(k)$ and nonlinearity f. The signal gets further contaminated by zero-mean, additive white Gaussian noise (AWGN) $w(k)$ at the receiver. The signal available at the input of equalizer is $u(k) = f(v(k)) + w(k)$, where $v(k) = \sum_{i=1}^{N_h} h(i)b(k-i)$ and $h(i)$, $i = 1, \ldots, N_h$, is channel impulse response. $h(i)$ is modelled as an finite impulse response (FIR) filter of length N_h and $b(k)$ is its input.

The receiver consists of an adaptive equalizer and a decision device. If the equalizer is designed to be an FIR filter, then its impulse response is $h_{\text{eq}}(j)$ for $j = 1, \ldots, N_{\text{eq}}$, where N_{eq} is length of the equalizer. The equalizer produces an estimate $\hat{y}(k)$, which should be as close as possible to desired output $y(k) = b(k-\tau)$. The delay parameter τ is usually taken to be $N_{\text{eq}}/2 + i_{\max} - 1$, where $i_{\max}$ is the index of maximum absolute value in $[h(1), \ldots, h(N_h)]$. In a nonlinear channel, an ideal equalizer cancels out the inter-symbol interference and channel nonlinearities by a suitable choice of weights $h_{\text{eq}}(j)$. Equalizer error at the output is computed as

$$e(k) = y(k) - \hat{y}(k) \tag{1}$$

Since, $e^2(k)$ is always positive and has better analytical properties, it is used as cost function instead of $e(k)$. The purpose of an adaptive algorithm is to recursively update the equalizer weights to reduce the $e^2(k)$ so that estimate $\hat{y}(k)$ approaches the desired signal $y(k)$.

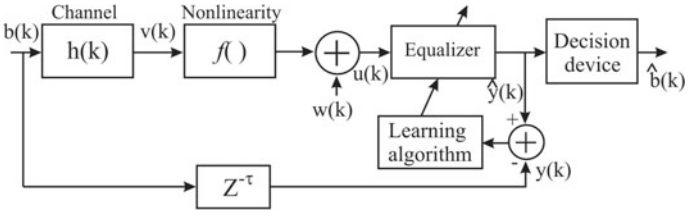


Fig. 1 Block diagram of a typical communication system employing an adaptive equalizer at the receiver

3 Echo State Network

ESN is an innovative new method of realizing the RNN. It is having low computational demands, without the disadvantages of RNN. The inner weights of the ESN are random values which are fixed at the time of initialization and only weights connected to the output node (readout weights) are trained using linear regression. Regression process selects the readout weights so as to minimize the difference between desired output and weighted sum of reservoir states. This is illustrated in Fig. 2a. ESN is initialized with random input matrix $\mathbf{W}^{\text{in}}$ and random reservoir $\mathbf{W}$. During the training process, only $\mathbf{W}^{\text{out}}$ is modified to get the desired output. It is the reservoir $\mathbf{W}$ from which ESN derives its power. Though the reservoir is initialized randomly, not all random initialization results in a good ESN. Finding good reservoir weights for a specific task requires a lot of repetitions with random initialization and the research on this is still in its nascent stage. A few methods exist in literature which try to find good reservoir for improving the performance of ESN using deterministic methods, but random initialization of $\mathbf{W}$ is most widely applied in ESN literature.

3.1 Architecture of ESN

Let us consider the architecture of a typical ESN, an example of which is shown in Fig. 2a. The ESN shown is a recurrent network with M inputs (constituting input layer), N internal processing elements (PE) (constituting the reservoir) and L output units. At time index k, signal array $\mathbf{u}(k) = [u_1(k), u_2(k), \ldots, u_M(k)]^{\text{T}}$ is applied to the input of the ESN causing the internal reservoir state to update to $\mathbf{x}(k) = [x_1(k), x_2(k), \ldots, x_N(k)]^{\text{T}}$. Output of the network at this point of time is, $\hat{\mathbf{y}}(k) = [\hat{y_1}(k), \hat{y_2}(k), \ldots, \hat{y_L}(k)]^{\text{T}}$ if there are L outputs or $\hat{y}(k)$ in case of single output network. The input weight matrix $\mathbf{W}^{\text{in}} = (w_{ij}^{\text{in}})$ is a $N \times M$ dimensional matrix connecting the input and the reservoir. There are N PEs in the reservoir which are connected with each other, with interconnection between them defined by the $N \times N$ matrix $\mathbf{W} = (w_{ij})$. Connections from PEs to the output nodes are defined by

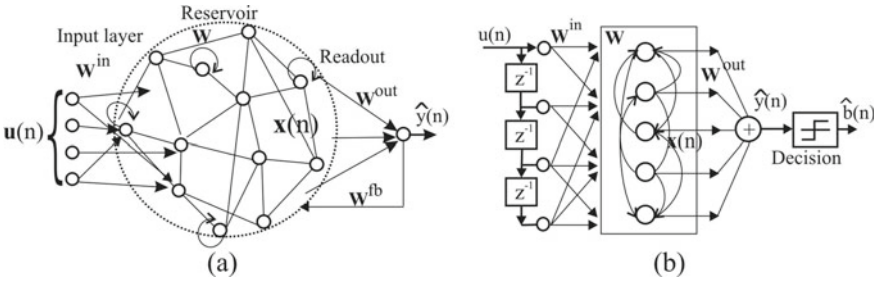


Fig. 2 **a** Architecture of ESN, **b** ESN-based equalizer

an $L \times N$ matrix $\mathbf{W}^{\text{out}} = (w_{ij}^{\text{out}})$ and an $N \times L$ matrix $\mathbf{W}^{\text{fb}} = (w_{ij}^{\text{fb}})$ for the feedback connection from output to internal PEs.

Figure 2b shows implementation of equalizer with ESN. It consists of a delay line which converts serial stream $u(k)$ into vector input $\mathbf{u}(k)$ at time instant n. Output of the ESN $\hat{y}(k)$ is applied to a decision device which converts continuous values into discrete symbols. BER is computed by comparing the estimated symbols $\hat{b}(k)$ with transmitted symbols $b(k)$.

3.2 Training

ESN is a supervised learning machine which can be made to output a desired signal $y(k)$ in response to input sequence $u(k)$. The reservoir in ESN emulates the function of a brain and consists of N^2 interconnections between N PEs. But unlike the brain, learning does not modify the interconnection weights w_{ij} of the reservoir. In the process of training, only the internal PE states $\mathbf{x}(k)$ are updated in response to inputs as given below

$$\mathbf{x}(k+1) = \mathbf{g}(\mathbf{W}^{\text{in}}\mathbf{u}(k+1) + \mathbf{W}\mathbf{x}(k) + \mathbf{W}^{\text{fb}}\mathbf{y}(k)), \tag{2}$$

where $\mathbf{g} = [g_1, g_2, \ldots, g_N]$ are activation function. Usually, g_i are hyperbolic tangent function (tanh) or linear function. Readout network weights-in the states $\mathbf{x}$ to produce the output according to

$$\mathbf{y}(k) = \mathbf{W}^{\text{out}}\mathbf{x}(k+1). \tag{3}$$

For simplification of understanding, we will consider ESN without feedback in this work. If $\mathbf{u}^{\text{tr}}(k)$ is the training input sequence, the state update equation is

$$\mathbf{x}(k+1) = \tanh(\mathbf{W}^{\text{in}}\mathbf{u}^{\text{tr}}(k+1) + \mathbf{W}\mathbf{x}(k)), \tag{4}$$

In Eq. (4), input vectors $\mathbf{u}^{\text{tr}}(k)$ are applied one-by-one to drive the dynamics of the reservoir. After the network is driven by a N_{tr} number of inputs, the state vectors $\mathbf{x}(k)$, for $k = 1, \ldots, N_{\text{tr}}$, are aggregated in a state matrix $\mathbf{S}$. The state vectors are ordered in such a way that each row consists of $\mathbf{x}(k)$ for a particular time index k. Therefore, kth row number of $\mathbf{S}$ indicates the time index k and column number $i \in [1, N]$ indicates state of ith PE. Usually, channel states $\mathbf{x}(k)$ recorded in the beginning of training are discarded since they are contaminated by initial transients.

Another popular architecture of ESN use a leaky-integrator type neuron. Compared to standard ESN, this design consists of an additional parameter called leaking rate α and its introduction enables ESN to learn slow dynamics like very slow sine wave [20]. The state update equations for leaky-integrator ESN are

$$\tilde{\mathbf{x}}(k+1) = \tanh(\mathbf{W}^{\text{in}}\mathbf{u}^{\text{tr}}(k+1) + \mathbf{W}\mathbf{x}(k)), \tag{5}$$

$$\mathbf{x}(k+1) = (1-\alpha)\mathbf{x}(k) + \alpha\tilde{\mathbf{x}}(k+1). \tag{6}$$

After collection of states in $\mathbf{S}$ for all the time steps (during training), readout matrix $\mathbf{W}^{\text{out}}$ is calculated by performing inverse operation.

$$\mathbf{W}^{\text{out}} = \mathbf{Y}^{\text{tr}}\mathbf{S}^{+}, \tag{7}$$

where $\mathbf{S}^{+}$ is pseudo-inverse of $\mathbf{S}$ and $\mathbf{Y}^{\text{tr}}$ is the array of all the target output samples $y^{\text{tr}}(k)$ for $k = 1, \ldots, N_{\text{tr}}$. Most universal and stable method to calculate (7) in this context is ridge regression:

$$\mathbf{W}^{\text{out}} = \mathbf{Y}^{\text{tr}}\mathbf{S}^{\text{T}}(\mathbf{S}\mathbf{S}^{\text{T}} + \beta\mathbf{I})^{-1}. \tag{8}$$

3.3 Testing

After applying training input $\mathbf{u}^{\text{tr}}$ to the reservoir and target output $\mathbf{Y}^{\text{tr}}$ for the computation of readout matrix, the ESN operates on test data $(\mathbf{u}(k), y(k))$ for $k = 1, \ldots, N_{\text{ts}}$. During testing, there is no need to washout initial states as the reservoir has already been initialized. The only difference with respect to training stage is that output $\hat{\mathbf{y}} = [\hat{y}(1), \hat{y}(2), \ldots, \hat{y}(N_{\text{ts}})]$ is computed using the $\mathbf{W}^{\text{out}}$ obtained earlier in (7). Finally, performance of the network is evaluated using mean square error (MSE) given below.

$$E(\hat{y}, y^{\text{ts}}) = \frac{1}{N_{\text{ts}}}\sum_{k=1}^{N_{\text{ts}}}(\hat{y}(k) - y(k))^2 \tag{9}$$

where, N_{ts} is the number of samples in testing sequence $y(k)$.

4 Reservoir Design Considerations

4.1 Reservoir

Reservoir matrix of ESN should satisfy certain algebraic properties called *echo state property*. Jaeger defined a network to have echo state property if after transient phase, the network state is independent of the effects of initial conditions and is driven completely by subsequent input signals [20]. One loose method of ascertaining echo state property is to keep spectral radius of $\mathbf{W}$ less than 1. Spectral radius is the largest of the eigenvalues of a matrix in terms of the magnitude.We next explain some of the common methods of constructing reservoirs in literature [21–23].

Random Reservoir: Random reservoir is just a $N \times N$ random matrix where matrix elements $w_{ij} \in [0, a]$ or $w_{ij} \in [-a, a]$ for some real number a. It is generated by initializing a random sparse matrix $\mathbf{W}'$ and its spectral radius $\mathcal{R}(\mathbf{W}')$ is calculated. Reservoir with desired spectral radius ρ is obtained by scaling $\mathbf{W}$ as

$$\mathbf{W} = \rho \frac{\mathbf{W}'}{\mathcal{R}(\mathbf{W}')} \tag{10}$$

The requirement of $\mathcal{R}(\mathbf{W}) < 1$ is a loose one and is merely used as initial reference point. Suitable value of ρ can be selected based on empirical results so as to minimize BER or MSE. A general guideline is to set ρ higher for tasks where generation of input signal involves long term memory [21]. Besides spectral radius, other important parameters are sparsity, size and underlying probability density function (PDF) of non-zero elements. Size of the reservoir is specified by N processing elements whose interconnection is defined by $N \times N$ matrix $\mathbf{W}$. In general, it is expected that larger reservoir will perform better compared to a smaller reservoir, provided due consideration has been given to regularization. Reservoir size may range from 10 to 10,000 depending on the application. A large reservoir is able to span a larger function space and can accommodate complex dynamical signals.

A sparse reservoir $\mathbf{W}$ consists of fewer non-zero elements compared to zeros in the matrix. In general, sparse reservoirs perform better and reduce computational cost of the network [20]. There are no analytical results for optimum value of sparsity and it need to be obtained through Monte Carlo simulations. Even though the matrix $\mathbf{W}$ is sparse, the PDF of non-zero elements have significant effect on the network. Uniform and Gaussian distributions are most commonly used and tend to give better performance. Width of the distribution does not matter because that is modified to achieve desired spectral radius [21].

Minimum Complexity Reservoirs: To simplify reservoir construction, three different topology templates were proposed in [23]. These ESN are similar in all aspects to random reservoir, except the construction of $\mathbf{W}^{\text{in}}$ and $\mathbf{W}$. In this case, all the non-zero elements of input matrix $\mathbf{W}^{\text{in}}$ are either $+c$ or $-c$ with equal probability, where c is a real number.

Weights in reservoir consists of only two different weights, p and q. Construction of the reservoir is greatly simplified because of this deterministic topology and is suitable for VLSI implementation. Three different topologies proposed are described next.

1. Delay line reservoir (DLR) consists of all the PEs connected in cascade. The reservoir matrix $\mathbf{W}$ is sparse with only lower subdiagonal having a random non-zero number. This is illustrated in Fig. 3a, where $w_{j+1,j} = p$ for $j = 1, \ldots, N-1$ and $w_{ij} = 0$ otherwise.
2. DLR with feedback (DLRB) network consists of PEs organized in line, but in addition, each PE is also connected to its previous PE. Non-zero elements of $\mathbf{W}$ are given as $w_{j+1,j} = p, j = 1, \ldots, N-1$ for lower subdiagonal and $w_{j,j+1} = q, j = 1, \ldots, N-1$ for upper subdiagonal. This is illustrated in Fig. 3b.

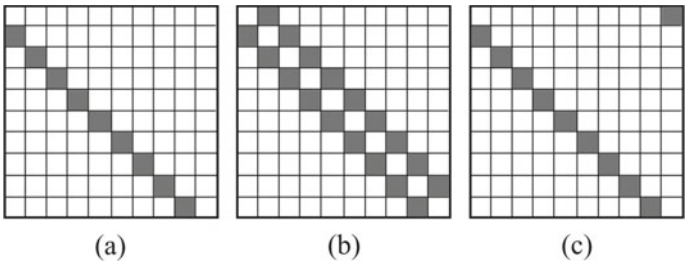


Fig. 3 Three different type of minimum complexity reservoir: **a** DLR, **b** DLRB and **c** SCR

3. Simple cycle reservoir (SCR) is one in which PE are connected to each other in such a way to form a loop. Non-zero elements are given as $w_{j+1,j} = p$ for $j = 1, \ldots, N-1$ and $w_{1,N} = p$.

Orthonormal Matrix Reservoir: It was postulated in [24] that random reservoir along with activation function in ESN acts as dynamical basis for the signal. Decomposition of signal into pertinent subcomponents in such a way to minimize redundancy between them will allow better utilization of reservoir resources. Utilization of orthonormal basis as weight matrix results in minimization of commonality between subcomponents and might provide improvement in results compared to a simple random weight matrix.

5 Channel Equalization Using ESN

Like other machine learning algorithms, training and subsequent testing of an ESN-based equalizer is required for it to perform the intended task. Training allows ESN to capture the underlying functional relationship using the training data and make correct inferences when the system is let out into the real world. For the purpose of training and testing, available dataset is divided into three parts:

1. First part serves to flush network of its initial states and transients.
2. Subsequent training inputs drive the network for actual learning of ESN output weights.
3. Finally, testing part is used for validating the newly trained network.

Training of ESN essentially consists of computing the output weights using stored collection of internal states and target output. The algorithm for ESN-based equalizer training and testing is summarized as follows:

1. Pass random binary signal $b(k)$ through communication channel with impulse response $h(k)$, nonlinearity f and noise $w(k)$ to produce training sequence $u(k)$.
2. Generate reservoir ($\mathbf{W}^{\text{in}}$, $\mathbf{W}$, α) ensuring echo state property. Once $\mathbf{W}^{\text{in}}$, $\mathbf{W}$ are generated at initialization, they are not changed during the training process.

3. Run the ESN-based equalizer with training data sequence $u(k)$ and subsequently save the reservoir states as $\mathbf{x}(k)$.
4. The states vectors of ESN for initial time steps $k = 1, \ldots, k_0$ are flushed out from $\mathbf{x}(k)$. For time $k > k_0$, collect network state $\mathbf{x}(k)$ in matrix $\mathbf{S}$.
5. Compute the readout weight matrix $\mathbf{W}^{\text{out}}$ using linear regression as given in (7).
6. Apply test input to the trained network and obtain the equalized/decoded symbols. Compare the decoded symbols with desired output to compute BER.

6 Simulation Results

We next present the simulation results which allowed us to explore the design choices in ESN. After that, simulation results for the proposed equalizer and symbol detector are given and compared with a standard equalizer in literature.

6.1 Effect of ESN Parameters

The system model adopted for simulation is shown in Fig. 1. A large number of simulation trials had been carried out to study the effect of parameters and channel conditions on BER performance. First set of simulations analyze the effect of reservoir parameters on the performance of the equalizer. Simulations were performed on two benchmark channels whose impulse response are given as [11]:

$$\text{Channel-1} : h_1 = [0.2602, \quad 0.9298, \quad 0.2602], \tag{11}$$

$$\text{Channel-2} : h_2 = [0.3040, \quad 0.9030, \quad 0.3040]. \tag{12}$$

Both the channels are associated with three different kinds of nonlinearity given by [25]

$$f_0(k) = v(k) \tag{13}$$

$$f_1(k) = \tanh(v(k)) \tag{14}$$

$$f_2(k) = v(k) + 0.2v^2(k) - 0.1v^3(k) \tag{15}$$

where, $v(k)$ is output of ISI channel with impulse response $h(k)$ as shown in Fig. 1. In (13), f_0 does not introduce any nonlinearity and is applied to study the behaviour of equalizer in the presence of ISI only. While nonlinearity f_1 can be observed in power amplifiers, f_2 is an arbitrary nonlinear function. Performance study of the equalizer and symbol detector was carried out using bit-error rate (BER) measure. During training, signal-to-noise ratio (SNR) at the input of the equalizer is 20 dB. Number of input samples applied to train the reservoir is 5000. For testing, the algorithm is run

for different values of SNR till some minimum numbers of errors are encountered. The ESN is of leaky-integrator type with $\alpha = 0.3$.

In the first set of experiments, we simulate the ESN-based equalizer and find out the minimum number of inputs required, so as to minimize BER. The BER performance for Channel-1 and Channel-2 as function of number of inputs is plotted in Fig. 4. The equalizer simulated consists of reservoir size $N = 50$ at an SNR of 15 dB. The reservoir is randomly generated with random weights uniformly distributed between $[-1, 1]$ with sparsity of 0.1 and spectral radius of 0.9. The input samples are applied to the reservoir in the form of delay-line with number of inputs varying from 2 to 10. It can be seen from Fig. 4 that BER of the equalizer improves with increase in the number of inputs. There is a sharp fall in BER for input length greater than 3, which is in fact the length of ISI channel filter. BER performance stagnates to a constant value when number of input is greater than 6 for both the channels. Hence, for all the subsequent experiments, number of input was to be $M = 6$.

In the next experiment, we compare the BER performance of equalizer for different reservoirs proposed in literature. These reservoirs differ with respect to the construction of the matrix $\mathbf{W}$ as discussed in earlier section. Simulations are performed with cascade of linear channels in (11)–(12) with nonlinearities given by (13)–(15). Table 1 lists and compares the BER performance for Channel-1 at an SNR of 15 dB. It was observed that for the linear channel (f_0), DLRB reservoir performs best with a BER of 2.27×10^{-5}. With nonlinearity f_1, minimum BER is obtained with DLR type reservoir. BLRB reservoir also exhibits best performance for the case of channel with nonlinearity f_2. Simulations were also performed for Channel-2 with the nonlinearities (13)–(15) and similar results in terms of performance were obtained.

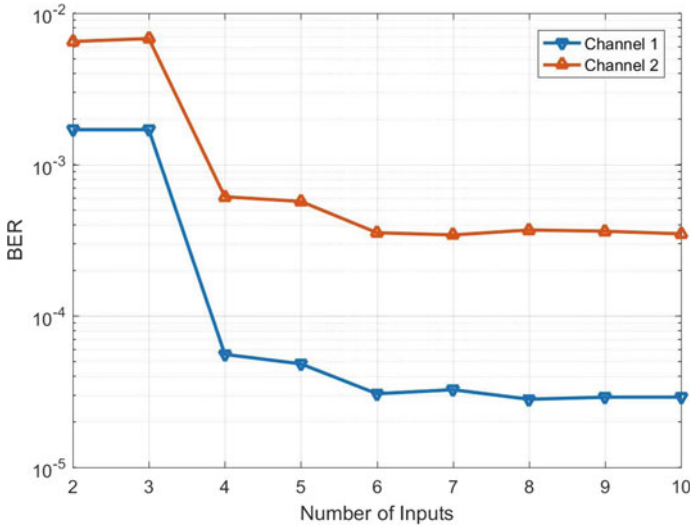


Fig. 4 BER of ESN-based equalizer as function of number of inputs

Table 1 Comparison of BER performance of different types of reservoirs at an SNR of 15 dB. Minimum value of BER for each channel combination is shown in bold

Reservoir	Channel-1			Channel-2		
	$f_0(\times 10^{-5})$	$f_1(\times 10^{-6})$	$f_2(\times 10^{-4})$	$f_0(\times 10^{-4})$	$f_1(\times 10^{-5})$	$f_2(\times 10^{-4})$
Random	2.70	1.88	2.00	3.85	2.70	13.0
Orth.	2.70	1.71	1.61	3.82	2.11	14.0
DLR	3.30	**1.36**	1.49	3.69	**1.77**	9.90
SCR	3.06	1.78	1.65	3.50	1.81	9.68
DLRB	**2.27**	1.95	**1.06**	**3.31**	2.00	**7.14**

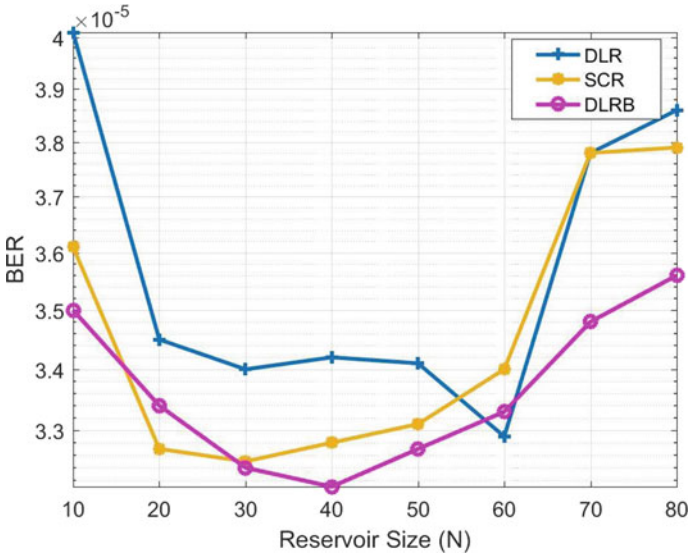


Fig. 5 BER performance plots as function of reservoir size for different types of reservoirs

Figure 5 displays the variation in BER of ESN equalizer with change in number of internal states N. The figure also compares the BER attained by DLR-, SCR- and DLRB- type reservoirs for different reservoir sizes. Because of the consistency in performance of these reservoirs in Channel-1 and Channel-2 and avoid repetition, this experiment is only performed for Channel-1. It can be observed from the figure that the equalizer performed best with reservoir size N of 30–50 for SCR and DLRB. On the other hand, DLR showed fairly uniform BER in the range of $N = 20$ to $N = 50$, with a sharp dip in BER at $N = 60$. With these observation, we can safely pick DLRB reservoir with size 40 for all the subsequent simulations. It should be noted that optimum reservoir size is dependent on the problem at hand is frequently obtained by empirical methods.

6.2 Equalizer Performance Comparison

The next set of results compare the performance of the ESN-based equalizer with LMS equalizer. LMS equalizer is a standard algorithm in wireless communication and is frequently used for comparing against other equalizers [18, 26].

In this experiment, the equalizers are compared for transmission over different severity of ISI distortion. Impulse response of channels with different amount of ISI is generated using the following relation [25, 26]

$$h(j) = \begin{cases} 0.5 + 0.5\cos(\frac{2\pi}{\Lambda}(j-2)), & j = 1, 2, 3 \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

where, parameter Λ controls the eigenvalue spread $\chi(R)$ and $R = E[\mathbf{u}(k)\mathbf{u}^{\mathrm{T}}(k)]$.

In Figs. 6, 7 and 8, BER is plotted for eigenvalue spread χ in the range of 10–50 and SNR of 15 dB. Figure 6 shows the plot for the case of transmission over ISI channel given by impulse response in (16) and without any nonlinearity (f_0). It can be observed that with increase in χ, BER also increases. For all values of χ, proposed ESN equalizer performs better compared to LMS equalizer, but the gap in BER performance also increases with increase in EVR. Similar conclusion can also be drawn from Fig. 7 for channel specified by (16) in cascade with nonlinearity f_1 and Fig. 8 for f_2.

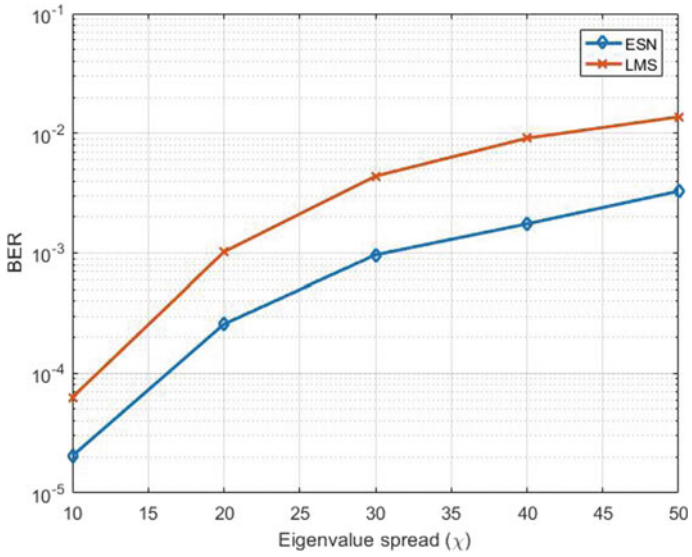


Fig. 6 BER as function of eigenvalue spread in linear ISI channel

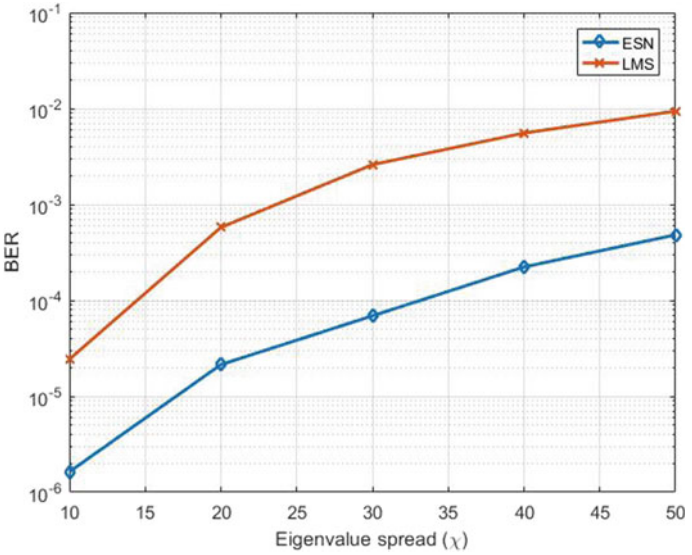


Fig. 7 BER as function of eigenvalue spread in ISI channel with nonlinearity f_1

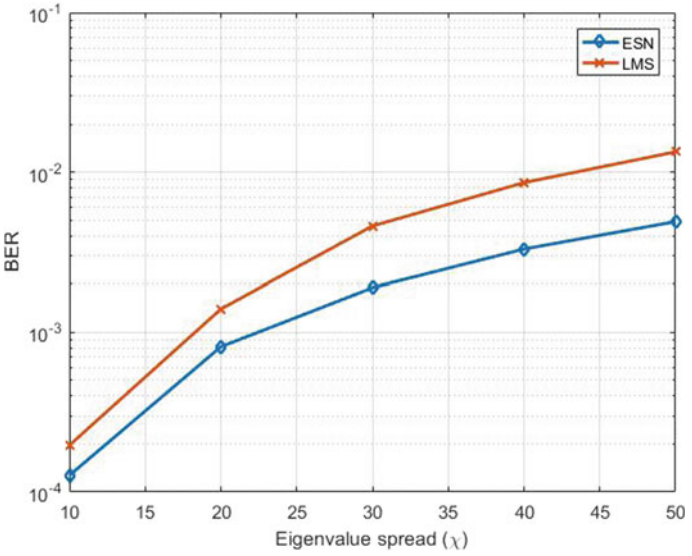


Fig. 8 BER as function of eigenvalue spread in ISI channel with nonlinearity f_2

7 Conclusion

In this article, we introduced a ESN-based equalizer for ISI and nonlinear channel, where it is trained as binary classifier for the purpose of symbol detection. Performance of this equalizer was studied for different length of input, reservoir size and type. Based on the results obtained, suitable parameter for ESN was selected and BER performance of proposed equalizer was compared to LMS equalizer. Simulations performed over different linear and nonlinear channels with different eigenvalue spread prove the superiority of proposed equalizer over LMS algorithm.

References

1. Bang SH, Sheu BJ (1996) A neural network for detection of signals in communication. IEEE Tran Circ Syst I: Fund Theory Appl 43(8):644–655
2. Siu S, Gibson GJ, Cowan CFN (1990) Decision feedback equalisation using neural network structures and performance comparison with standard architecture. IEE Proc I-Commun Speech Vis 137(4):221–225
3. Chen S, Gibson GJ, Cowan CFN, Grant PM (1990) Adaptive equalization of finite non-linear channels using multilayer perceptrons. Sig Process 20(2):107–119
4. Meyer M, Pfeiffer G (1993) Multilayer perceptron based decision feedback equalisers for channels with intersymbol interference. IEE Proc I (Commun Speech Vis) 140(6):420–424
5. Burse K, Yadav RN, Shrivastava SC (2010) Channel equalization using neural networks: a review. IEEE Trans Syst Man Cybern Part C (Appl Rev) 40(3):352–357
6. Pao Y (1989) Adaptive pattern recognition and neural networks. Addison-Wesley, Reading, MA
7. Patra JC, Poh WB, Chaudhari NS, Das A (2005) Nonlinear channel equalization with QAM signal using Chebyshev artificial neural network. In: Proceedings of international joint conference on neural networks, 2005, vol 5, pp 3214–3219 (2005)
8. Patra JC, Meher PK, Chakraborty G (2009) Nonlinear channel equalization for wireless communication systems using Legendre neural networks. Sig Process 89(11):2251–2262
9. Chen S, Mulgrew B, Grant PM (1993) A clustering technique for digital communications channel equalization using radial basis function networks. IEEE Trans Neural Networks 4(4):570–590
10. Jianping D, Sundararajan N, Saratchandran P (2002) Communication channel equalization using complex-valued minimal radial basis function neural networks. IEEE Trans Neural Networks 13(3):687–696
11. Nanda SJ, Jonwal N (2017) Robust nonlinear channel equalization using WNN trained by symbiotic organism search algorithm. Appl Soft Comput 57:197–209
12. Al-Awami AT, Zerguine A, Cheded L, Zidouri A, Saif W (2011) A new modified particle swarm optimization algorithm for adaptive equalization. Digit Signal Proc 21(2):195–207
13. Iqbal N, Zerguine A, Al-Dhahir N (2015) Decision feedback equalization using particle swarm optimization. Sig Process 108:1–12
14. Panda S, Mohapatra PK, Panigrahi SP (2015) A new training scheme for neural networks and application in non-linear channel equalization. Appl Soft Comput 27:47–52
15. Panda S, Sarangi A, Panigrahi SP (2014) A new training strategy for neural network using shuffled frog-leaping algorithm and application to channel equalization. AEU-Int J Electron Commun 68(11):1031–1036
16. Nanda SJ, Garg S (2019) Design of supervised and blind channel equalizer based on moth-flame optimization. J Inst Eng (India): Ser B 100(2):105–115

17. Kechriotis G, Zervas E, Manolakos ES (1994) Using recurrent neural networks for adaptive communication channel equalization. IEEE Trans Neural Networks 5(2):267–278
18. Mosleh S, Liu L, Sahin C, Zheng YR, Yi Y (2017) Brain-inspired wireless communications: where reservoir computing meets MIMO-OFDM. IEEE Trans Neural Networks Learn Syst 99:1–15
19. Bauduin M, Massar S, Horlin F (2016) Non-linear satellite channel equalization based on a low complexity echo state network. In: Annual conference on information science and systems (CISS). IEEE, New York, pp 99–104
20. Jaeger H (2001) The echo state approach to analysing and training recurrent neural networks-with an erratum note. German National Research Center for Information Technology GMD Technical Report, vol 148(34), p 13
21. Lukoeviius M (2012) A practical guide to applying echo state networks. In: Neural networks-tricks of the trade, Springer, Berlin, Heidelberg, pp 659–686
22. Ozturk MC, Xu D, Principe JC (2007) Analysis and design of echo state networks. Neural Comput 19(1):111–138
23. Rodan A, Tino P (2010) Minimum complexity echo state network. IEEE Trans Neural Networks 22(1):131–144
24. Millea A (2014) Explorations in echo state networks. Master's Thesis, University of Groningen
25. Patra JC, Pal RN, Baliarsingh R, Panda G (1999) Nonlinear channel equalization for QAM signal constellation using artificial neural networks. IEEE Trans Syst Man Cybern Part B (Cybern) 29(2):262–271
26. Haykin S (2005) Adaptive filter theory. Pearson Education, New Delhi

Chapter 10
Melody Extraction from Music: A Comprehensive Study

Ranjeet Kumar, Anupam Biswas, and Pinki Roy

1 Introduction

In recent years, music providers like Google Play, iTunes, JioSaavan and Gaana have been evolved overwhelmingly. The music industry is also restructured completely from disc era to digital era and todays situation where users can get access to millions of tracks on their phones or on cloud-based services. This massive collection of music requires some way to deal with searching and retrieve desired piece of track efficiently. Presently, the fundamental concern of music providers is to characterize this huge number of tracks on the basis of their components like beat, pitch, melody and so on [1]. Among these components, melody is predominantly used for characterization of music.

The term *melody* is a thought of musicological subject to the judgment of the human group of audience and we can hope to find diverse definitions of melody in diverse context. Some of the definitions for melody are: "mix of a pitch arrangement and a rhythm having an obviously characterized shape" [1], and "pitched sounds organized in melodic time as per given social conventions and constraints" [2]. Melody of music is the single or monophonic pitch gathering that an audience may copy at any point of time asked to hum or whistle a touch of polyphonic music, and that when an audience heard in contrast, they would just observe the *essence* of that particular music [3]. This concept is still accessible to a broad level of subjectivity, since various audience members might hum various parts in the wake of tuning in to a similar melody.

The melody extraction task includes automatically acquiring a sequence of frequency values of the predominant melodic line from polyphonic music signal Fig. 1.

R. Kumar (✉) · A. Biswas · P. Roy
National Institute of Technology Silchar, Silchar, Assam, India
e-mail: ranjeetjha8484@gmail.com

A. Biswas
e-mail: abanumail@gmail.com

P. Roy
e-mail: pinkiroy2405@gmail.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_10

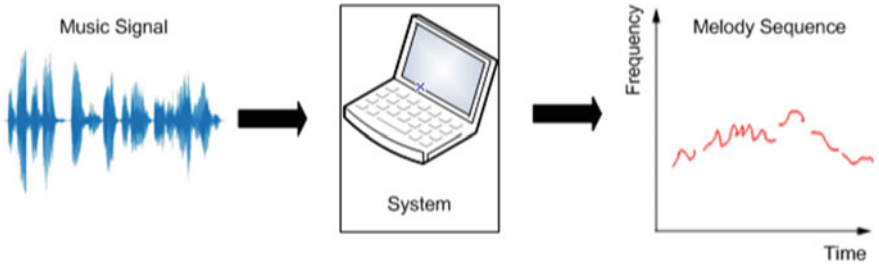


Fig. 1 Melody extraction from music signal

The term *polyphonic* means the music that can produce at least two notes at the same time, be it on various tools (e.g., bass, voice and guitar) or a solitary equipped instrument for playing several note on a single period like piano. A listener can reproduce the melodies without having any musical knowledge. However, this task becomes very difficult when we try to automate it, because of the superposition of two or more sounds produced by various instruments [4].

Automatic melody extraction is very popular task in MIR. Lots of approaches have been developed for melody extraction from polyphonic music [5]. These approaches are classified into three categories, namely (1) salience-based approach, (2) source separation-based approach and (3) data-driven approach. Recently, Deep Neural Network (DNN) has gained attention in MIR applications. DNN has the ability to estimate any function with linear weights and activation functions by providing enough number of data. DNN is a fully data-driven approach which performs well in sequence to sequence problems. Many application uses the melody extraction, including identification of a particular piece of music and searching a song by humming or singing of its melody, singer characterization and music transcription (description of notes being played in a music signal). It can also be used for creating music production tools. In this chapter, various melody extraction approaches, datasets, performance measures and applications are covered.

Rest of the chapter is organized as follows: Sect. 2 briefly explained the overview of techniques, Sect. 3 discussed the datasets used to extract melody and their details, Sect. 4 briefed about the performance measures of melody extraction techniques and their descriptions, Sect. 5 detailed about the application of melody extraction techniques, and Sect. 6 provided the challenges to improve the performance of melody extraction techniques and finally highlighted the concluding points and future perspectives.

2 Melody Extraction Techniques

As discussed in the introduction, melody extraction is a task to evaluate the sequence of f_0 (fundamental frequency) corresponds to the predominant melodic line from the music where more than two notes can sound simultaneously. Earlier melody

extraction approaches were designed for monophonic music. However, nowadays the focus has been shifted to melody extraction from polyphonic music to estimate f_0. In polyphonic music, melody extraction requires more attention to identify the part of music where actual melody resides and estimate pitch value on that part only. There are two types of sources from where the melody is extracted and built the datasets. (1) Waveform representation or audio and acoustic data and (2) Symbolic representation like scores and MIDI [1, 5]. The application where waveform representation used is very complex to implement in comparison with symbolic representation. For extraction of melody using waveform representation follows some steps like pitch evaluation, note segmentation and then melody estimation. All algorithms based on their fundamental strategy have been explored here to classify. Most methods fall within previously introduced two primary categories: salience-based approach and source separation-based approach [1, 4], and some techniques do not fit in either class: data-driven-based technique in which the energy spectrum is transferred straight into a machine learning technique which tries to divide the spectral frequency of melody.

2.1 Salience-Based Approaches

Most of the melody extraction approaches utilize salience function. A salience function is nothing but pitch salience, which is computed based on the time-frequency representation [6]. The pitch over time is calculated and then tracking rule is applied to estimate the melodic line without separation from rest of the music file [1]. The melody extraction process through salience-based approach is divided broadly into three steps as shown in Fig. 2.

Preprocessing: In this step, normally to enhance the frequency of the music part (where melody is expected), some filter is used. Band pass filter has been applied by Goto [7], while equal loudness filter which is perceptually motivated is used for preprocessing by Salamon and Gómez [1]. Source separation-based approach also been used before further processing for signal enhancement. However, Harmonic-

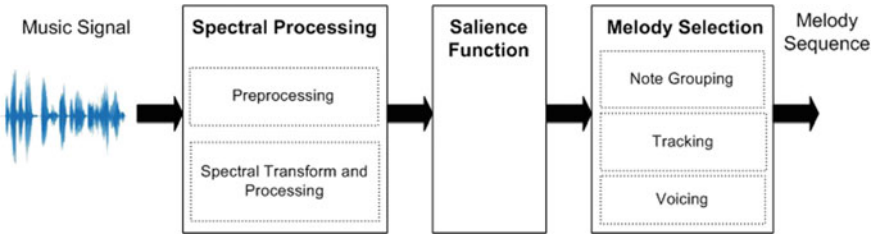


Fig. 2 Melody extraction using salience-based approach

Percussive Sound Separation (HPSS) technique has been used by the Hsu [8] to separate the accompaniment and melody from music signal.

Spectral transform and processing: The preprocessed signal is sliced into time frames. Next, on each of the frames, some transformation function has been implemented to obtain spectral representation. With a typical window length 40 to 100 ms (gives sufficient frequency resolution to differentiate diverse notes), Short-Time Fourier Transform (STFT) approach is applied as transform function. Multi-Resolution Fast Fourier Transform (MRFFT) is trying to overcome the limitation of time-frequency resolution. In this transforms, larger windows can be used at low frequency and lower windows' size at higher frequency [9]. Salamon and Gómez [1] showed that there are no significant difference between STFT and MRFFT used for melody estimation. After transformation is applied, most methodologies just utilize spectral maxims for further evaluation. Spectral magnitude normalization is applied to reduce the timbre on the analysis [5]. Other approaches apply Filter peaks based on magnitude to filter out peaks which do not contain harmonic information to detect the peaks. Salamon and Gómez [1] estimate more precise amplitude and frequency for every spectral peak from the phase spectrum by evaluating its frequency.

Salience function: Salience function lies in the multipitch representation which is core of salience-based approaches. It estimates the salience function of each pitch value where we anticipate to obtain the melody over time. To obtain the salience function, diverse techniques have been proposed. Most methodologies utilize some kind of harmonic summation, which determines the salience of the specific pitch as a weighted sum of the sufficiency of its harmonic frequencies [1, 10–12]. Expect Maximization (EM) to fit many tone models into the observed range [7]. The evaluated Maximum a Posteriori Probability (MAP) of tone model of which fundamental frequency f_0 relates to a specific pitch is viewed as the pitch's salience. An approach chooses a pitch esteem which is actually above a single octave or beneath the right pitch of the song referred as octave error, which is appeared when fundamental frequency f_0 of the actual pitched signal is exact multiple of f_0. Some approaches tried to reduce this error directly by inspecting sets of ghostly peaks which possibly have a place with a similar harmonic arrangement and constricting the consequence of their summation if there are numerous high amplitude spectral peaks whose frequencies lie between the pair being considered [13]. Spectral smoothness has applied for octave error reduction by Klapuri [14], in which amplitude of each peak in the salience function is recalculated in the wake of smoothing the phantom envelope of its relating harmonic frequencies. Peak representing to octave errors will have an unpredictable envelope (contrasted with a smoother envelope for genuine notes) and along these lines will be weakened by this procedure. Here, we note that for all intents and purposes all strategies decrease octave errors non-explicitly by punishing enormous bounces in pitch during the tracking phase of the approach.

Tracking: The rest is to determine the peaks (i.e., pitches) that make up the melody, given the peaks of the salience feature. Tracking is one of the most important phases of each approach and perhaps also the most diverse phase in which virtually every

algorithm utilizes a distinct strategy. Most methods try to monitor the melody straight from the peaks of salience, through some preliminary grouping phase in which peaks are grouped into fragments or trajectories and later from this only melody is chosen [1, 4, 11]. Usually this clustering is done by monitoring sequential pitches based on moment, peaks and salience. The final sequence of melody is achieved by a range of monitoring methods from given pitch contours. Dressler [13] and Goto [7] use heuristic-based tracking, while Yeh [15] and Ryynänen [10] use HMM. Salamon [1] and Paiva [4] come with a distinct approach—they try to delete all pitch counters (or notes) that do not belong to the melody rather than monitoring the melody.

Voicing: The voicing detection is a significant component of the melody extraction that is sometimes ignored. In this phase, algorithms detect the part of music where melody is present and when it is not. Usually, an algorithm's voicing detection step is implemented at the very end. Fixed or dynamic per frame salience-based threshold approach is usually used [4, 13, 16] but Ryynänen [10] comes with a distinct approach which includes a silence model in the algorithm's HMM monitoring portion (Table 1).

2.2 Source Separation-Based Approaches

The use of source separation algorithms is an alternative approach for salience-based melody extraction technique, to separate the melody source from the mixture of music. This strategy is the latest and has become popular following the progress made in studies on the separation of the audio source. Although there is a big bunch of research on the separation of a melody from the lead vocal source, these algorithms are typically assessed by measuring them based on a signal-to-noise relationship, and few are evaluated as our objective here in terms of the estimation of the melody frequency sequence.

Durrieu et al. [17] come with the idea of source separation-based approach. As an instantaneous amount of two contributions: the accompaniment and the lead, Durrieu models the energy spectrogram of the signal. Accompanying contribution as a sum of arbitrary sources and lead voice contribution is depicted by a source or filter model with separate spectral form. Smooth Gaussian-Scaled Mixture Model (SGSMM) and a Smooth Instantaneous Mixture Model (SIMM) are suggested as two distinct depictions of the source/filter model. The first depicts the lead voice (or instrument) as the immediate mix of all possible peaks while the other is more realistic that it enables only one source/filter pair to be active at any time, though computerized. In both scenarios, the model variables are calculated using a framework for maximization of expectations. Using a Viterbi algorithm, a smooth trajectory is acquired through model variables and obtained the final melody sequence which includes the fundamental frequency f0 of the source after model parameters are estimated. Voice detection occurs with the help of Wiener filters to split the melody signal based on model variables and to then calculate the power of that music signal in each frame for sequences in which the melody is present.

Table 1 Salience-based approaches

Algorithm	Preprocessing and spectral transform	Processing	Salience function	Tracking	Voicing
Arora and Behera [35]	STFT and log spectrum	Peak selection	Log specific IFT	Tracking of the harmonic cluster	Harmonic sum
Cancela [11]	Constant Q + high pass filter	Normalization of log power	Harmonocity map	Countour tracking and smoothing	Adaptive threshold
Dressler [13]	MRFFT and peak correction	Magnitude threshold	Spectral peak comparison	Streaming rules	Dynamic threshold
Goto [7]	Bandpass filter and multirate filterbank	IF-based peak selection	EM fit to tone model	Tracking agents	–
Jo et al. [12]	STFT	–	Harmonic summation	Stable candidate + rule-based selection	Implicit
Marolt [32]	STFT	Noisier harmonics	EM fit to tone model	Tracking agents	–
Paiva et al. [4]	Auditory model	Autocorrelation peaks	Summery correlogram	Multipitch trajectories and note detection	Salience valleys
Rao and Rao [16]	FFT with high resolution	Main lob matching magnitude	SMS and TWM	Dynamic programming	NHC threshold
Ryynänen and Klapuri [10]	STFT	Spectral whitening	Harmonic summation	HMM + global HMM	Silence model
Salamon and Gómez [1]	Eqal loudness filter and STFT	IF-based peak correction	Harmonic summation	Countour tracking	Salience distribution
Sutton [36]	Semitone + bandpass	NA	NA	Monophonic pitch tracker HMM	Confidence HMM
Hsu and Jang [8]	Harm/perc sound separation + MRFFT	Vocal partial discrimination	Sub-harmonic summation	Dynamic programming	Classification
Yeh et al. [15]	Harm/perc sound separation + MRFFT	Vocal partial discrimination	Sub-harmonic summation	HMM	–

Tachibana et al. [18] proposed a quite different strategy, which is based on the time variation of the melody compared to longer chord notes. Harmonic-Percussive Sound Separation (HPSS) approach is used for this purpose. The model was initially intended for harmonic separation from percussive components in a music mix by splitting channels which are smooth in time (harmonic content) and channels which are smooth in frequency (percussive content). The algorithm can be used to distinguish "continuous" sounds from "temporarily variable" (melody + percussive) sounds by altering the window duration used for assessment. The algorithm will run again once for the removal of percussive components in its initial form after the accompaniment is removed. The melody of the signal should be considerably

improved after these two passes. By applying dynamic programming, the sequence of melody is obtained directly from the improved signal spectrum by discovering the route to maximize the MAP of the frequency sequence, in which the probability of a spectrum-specific frequency is proportional to the weighted sum of multiples for energy and the possibilities of transformation are based on the distance between two sub-sequent frequency values. Detection of voicing is performed by setting a limit on the range between the two signals generated by the HPSS algorithm's second run (the melody signal and the percussive signal).

For comprehensiveness, a few singing speech source separation techniques are briefly discussed here. As already noted, although these techniques have not been assessed in aspects of melody extraction, they can either be used as a preprocessing phase comparable to the previously described approaches, by incorporating them with a tracking method for monophonic pitch that measures the melody fundamental frequency f_0 series from the separated speech signals. The fact that the accompaniment of music commonly has a recurring structure while there is various variety in tone is another approach to separate the lead voice. Huang et al. [19] take advantage of this assuming that the accompaniment's spectrograph can be modeled on a low rank matrix and a sparse matrix on the voice's spectrogram. Rafii and Pardo [20] proposed a distinct way to exploit repetition. First, they use autocorrelation applied to the spectrometric blend to determine the time of the accompaniment repeatedly. After calculating the median of successive repeat spectrograms, they get a spectrogram that only includes a repeating signal (accompanying). This spectrogram is used to distinguish the accompaniment from the voice by using a time-frequency mask.

2.3 Data-Driven Approaches

As we have already discussed, most of the algorithms are based on salience function and source separation from the music mixture, and data-driven-based approaches have been rarely explored. But in recent year, this type of approach turns as emerging area of research. This approach can be divided into following steps.

Preprocessing: In this step, some spectrogram is used to visualize the distribution of energy in time and frequency domain both of music signal. Most of the researchers used the spectrogram with hanning windows [21, 22] while Su [23] uses the high pass filter to allow high frequencies to get pass while cutting low frequencies in preprocessing step. Mainly, hamming window is used to reduce the effects of the leakage that occurs during spectral transform.

Spectral transform: Preprocessed signal is now ready to be chopped into frames. Each of theses frames goes through some transformation function to gain spectral representation. Most of the authors applied STFT because where the frequency elements of a signal differ over time, it supplies time-based frequency information [21]. Huang et al. [24] used Constant—Q Transform (CQT), as a time-frequency representation

for music recordings. CQT is essentially best suited and resulting representation is very least dimensional in comparison with STFT.

Activation function: Activation functions are very essential in order to study something very complex and non-linear complicated structure between inputs and responsive variable in Deep Neural Network (DNN) and to make sense of it. ReLU function is the mostly used activation function because it does not saturate and very quick to obtain [22, 23, 25]. Park et al. [21] and Fan et al. [26] applied sigmoid function instead of ReLU function.

Processing: Kum et al. [22] come up with the idea of multi-column deep neural networks for melody extraction. They used this model as classification-based approach and trained each of neural network to predict a pitch label. They combine the output of networks and post-processing it with hidden Markov model and inferred it as melody contour. Radenen [27] proposed a system based on combining two deep neural network for estimating the fundamental frequency of melody. Recently, Park and Yoo [21] come with the idea of long short-term memory recurrent neural network to extract the melody and simultaneously detecting the frame where melody is present. A convolution neural network based on patch has been introduced by Su [23] to extract vocal melody from polyphonic audio files. They have considered that a short pitch contour is sufficient to differentiate voiced melody from accompaniment and data augmentation is absent. Huang and Liu [24] combined the concept of harmonic structure and neural network and named as deep harmonic neural network while Lu and Su [28] proposed an algorithm based on deep convolutional neural networks with dilated convolution as semantic segmentation tool (Table 2).

Table 2 The data-driven-based approaches

Algorithm	Preprocessing	Spectral transform	Activation function	Processing	Approach
Dogac et al. [25]	Log-spectrogram with hamming windows	STFT + source filter nonnegative matrix factorization (SF-NMF)	ReLU	CNN- RNN	Classification based
Hyunsin et al. [21]	Spectrogram with hanning windowing	STFT	Sigmoid	LSTM- RNN + Harmonic sum loss	Classification based
Su [23]	High pass filter	STFT + combined frequency and periodicity (CFP)	ReLU	Patch based CNN	–

(continued)

Table 2 (continued)

Algorithm	Preprocessing	Spectral transform	Activation function	Processing	Approach
Kum et al. [22]	Spectrogram with hanning windowing	Augmentation pitch shifting	ReLU	MCDNN + HMM Viterbi smoothing	Classification based
Yuzhi et al. [24]	Spectrogram	Constant Q transform (CQT)	–	DHNN + beam search tracking	–
Fan et al. [26]	Magnitude spectrum	STFT	Sigmoid	DNN + root mean squared gradient RMSProp + dynamic programming	Source separation based

3 Datasets

As we have discussed the approaches to evaluate the melody of audio file. In this section, datasets which are commonly used for melody extraction has been discussed. There are many datasets collected by distinct research groups for melody extraction. Some of them are freely available for research in the field of MIR. Table 3 represents the datasets and their descriptions.

MIREX05: Most commonly used datasets in melody extraction field that are generated from MIDI files. There are various genres available like Rock, Pop, Jazz, classical piano and so on. This database is recorded on single channel, with 16-bit with 44,100 Hz sampling rate of 20–30 s segments [29].

Medley DB: This dataset consists of 196 total number of tracks recorded in WAV file with 44.1 kHz sampling rate in 16-bit. Medley DB is basically recorded by singer having genres like classical, rock, folk, jazz, etc. [30].

Table 3 Dataset collections and its description

Name	Sampling rate (in KHZ)	Pulse code modulation (in bit)	Genres	Duration (in seconds)
MIREX 2005	44.1	16	Rock, Pop, Jazz, Piano	20–30
Medley DB	44.1	16	Classical, Rock, Folk, Jazz	–
MIR-1K	16	16	–	4–13
ADC 2004	44.1	16	Daisy, Jazz, Opera, MIDI, Pop	20

MIR-1K: This is generated by group of 11 males and 8 females non-professional group of singers. The duration is between 4 to 13 seconds and the total length of the clip is 133 min. 1000 song clips were recorded at both right and left channels. This dataset is publicly available recorded with 16 kHz sampling rate [26].

ADC 2004: This dataset consists of four excerpts of each genres pop, jazz, daisy, opera and MIDI. It contains 20 audio clips recorded at sampling rate of 44.1 kHz and about 20 s duration with 16-bit pulse code modulation [22].

4 Performance Measures

In melody extraction approaches, two goals are accomplished: Firstly estimate at which interval melody is present and when it is not, and estimate pitch of the melody. For evaluating the performance of approaches for a given music file, we have to compare the ground truth with the algorithms outcome [1]. The actual series of fundamental frequency representing the melody of music file is containing in ground truth file having format as the format of outcome file. In this section, we have discussed some of the performance measures commonly used for melody extraction algorithms.

Let vector f and F represents the uni-dimensional estimated melody pitch frequency sequence and ground truth frequency sequence, respectively. v indicates the voicing indicator vector, whose ith element $v_i = 1$ when ith frame is estimated as voiced (i.e., particular frame where melody is present), with corresponding ground truth V. Unvoicing indicators represented by $\bar{v}_i = 1 - v_i$.

4.1 Voice Recall (VR)

The ratio of frames is labeled as voiced frame with the ground truth melodic frame, i.e., true labeled frames of melodic/ground truth melodic frame.

$$\mathrm{VR} = \frac{\sum_i v_i V_i}{\sum_i v_i} \tag{1}$$

4.2 Voicing False Alarm (VFA)

The ratio of mistakenly estimated frames as melodic frame with frames is labeled as non-melodic.

$$\mathrm{VFA} = \frac{\sum_i v_i \bar{v}_i}{\sum_i \bar{v}_i} \tag{2}$$

4.3 Raw Pitch Accuracy (RPA)

The portion of correct pitch of frames that are correctly detected as melodic and frames which are correctly pitch but estimated as unpitched with the frames that are ground truth melody frame.

$$\text{RPA} = \frac{\sum_i v_i \tau[\zeta(f_i) - \zeta(F_i)]}{\sum_i v_i} \tag{3}$$

where τ is described by the threshold feature and defined by:

$$\tau[a] = \begin{cases} 1 & \text{if } |a| < 50 \\ 0 & \text{if } |a| > 50 \end{cases} \tag{4}$$

And ζ maps a frequency value to a motivated axis, where each semitone is segmented into 100 cents. Frequency can be described over a reference frequency f_{ref} by a significant value number of cents.

$$\zeta(f) = \log_2\left(\frac{f}{f_{\text{ref}}}\right) \tag{5}$$

4.4 Raw Chroma Accuracy (RCA)

It is the same thing as the RPA except ignoring the octave error (i.e., both the ground truth and estimated frequency sequence are mapped on a single octave).

$$\text{RCA} = \frac{\sum_i v_i \tau[\langle\zeta(f_i) - \zeta(F_i)\rangle_{12}]}{\sum_i v_i} \tag{6}$$

where,

$$\langle a\rangle_{12} = a - 12\lfloor\frac{a}{12} + 0.5\rfloor \tag{7}$$

4.5 Overall Accuracy (OA)

Overall accuracy is nothing but the ratio of frames is correctly labeled with melody and pitch both. If L is total number of frame, then OA can be described as:

$$\text{OA} = \frac{1}{L}\sum_i V_i \tau[\zeta(f_i) - \zeta(F_i)] + \bar{V}_i \bar{v}_i \tag{8}$$

5 Melody Extraction Applications

The progress made during the last decade in algorithmic efficiency of melody extraction techniques now allows adequately excellent outcomes to build more complicated systems on them. In this section, we are discussing few of these systems, where melody extraction techniques play a crucial role.

Classification: Automatic music classification tries, through the automatic allocation of descriptive labels, to assist individual consumers and executives of big music distributors to organize their collections in these collections. Music genre (Rock, Pop, Folk, Jazz, etc.) is the most favorable labels for organizing the music. In [3], the writers introduce a model for genre classification on the basis of melodic characteristics that have been achieved using melody extraction. The writers show how these characteristics can help enhance classification precision with more frequently used timbral characteristics, like Mel-Frequency Cepstral Coefficients (MFCC).

Music retrieval: Music retrieval is one of the most highly beneficial systems for melody extraction, which is, by observing and comparing songs, helping consumers discover the songs, however, they are interested in or explore new musically. Here, we emphasize two different and related recruitment applications in this large application field: version identification (version ID) or also cover song ID and Query-By-Humming (QBH) applications. The tack of version ID is to automatically retrieve various releases of musical recordings supplied to the user by the system. To detection of possible violations of copyright on Web sites (e.g., YouTube), in order to automate the evaluation of how artists and musicians affect the composition of each other. Although the melody is one of the few unchanged aspects of music across various versions, various studies examined the usages of melody extraction in version ID, whether it is to try and transcribe it fully [31], using it as an intermediate representation for computing similarity [32] or merging it with another key characteristics (e.g., accompaniment, bass-line or harmony) [33].

QBH systems aim to assist the client in the circumstance where he recalls the song's melody but has no publisher data (e.g., artist, album or title). Applications based on QBH assist the client to collect this data by permitting them to hum or sing the melody. Although there was still no professional melody extraction scheme for QBH system, promising findings were presented in the study models [33].

Transcription: Although mid-level melody frequency representation is already extremely helpful but sometimes desirable to transcribe to western symbolic notation. Music transcript is a unique and appealing goal to help consumers understand music by generating automatic scores [10]. The broad variety of methods created for symbolic melodic resemblance and retrieval may be applied by acquiring a symbolic representation of the melody [34]. Transcription can be obtained by applying a melody extraction approach quantized its output to generate a musical note on time and pitch.

De-soloing: In this process, lead instruments are removed from a polyphonic music. In source separation techniques, melody extraction can be utilized by offering a melody's "score" as a first step toward de-soloing.

6 Challenges

High degree of polyphonic instrumental music: While most methods can handle instrumental music, many are specially adapted to vocal music due to the huge vocal music popularity and the distinctive voice of people which systems can take advantage of. If extraction of melody from instrumental music is handled adequately, then we can create systems that generalize to a wider spectrum of music materials. Compared to vocal melody extraction, this poses two difficulties: firstly, instrumental sound is not as limited as vocal music. Instruments have a broader variety of pits, can quickly generate changeable pitch patterns and can include big pitch jumps. Secondly, in timbre and on the pitch contour of the particular notes, the tool playing the melody can be nearer to other tools that make it more difficult to separate the melody from its accompaniment.

Irrespective of what the instrument is playing, the job is more difficult as we raise the number of instruments in the music mix. This makes it more difficult to properly identify particular pitching sources and to overlap spectral content. Even if the pitch attributes of distinct notes are properly differentiated, it is now extremely difficult to see which of them pertain to the melody.

Voicing detection: As we have discussed earlier, to achieve high overall accuracy there must be higher raw pitch accuracy and better voicing detection approach. Most of the techniques concentrate mainly on the former melody extraction aspect and less on the latter, some of them even do not involves a voicing detection step. At present, most efficient algorithms for voicing detection have an average false alarm rate. In [1], the researchers observe that the greatest possible increase in their algorithm's efficiency would be a reduction of false alarm voices.

7 Conclusion and Future Perspective

In this chapter, comprehensive study about the melody extraction approaches and their algorithmic design as well as potential systems, where melody extraction technique can be applied, has been briefly discussed. Melody can be defined as the estimation of predominant fundamental frequency f_0 from the melodic line of polyphonic music. The wide range of algorithms for melody extraction has been recognized. These approaches can be classified based on key concepts used in the algorithmic design, namely salience based, source separation based and data-driven approaches. The data-driven approaches perform better than the conventional salience-based

techniques and source separation-based approaches. Data-driven approach has potential to improve performance if training dataset is augmented by pitch shifting and modifying the pitch label accordingly. To evaluate the melody extraction algorithm, various performance measures have been discussed. It has been observed that melody extraction concept has been utilized in various applications such as classification, music retrieval, transcription and de-soloing. High degree of polyphonic instrumental music makes the melody extraction job more difficult as the raising of number of instruments in the music. Voicing detection algorithm can be utilized to improve the high overall accuracy.

Acknowledgements This research was funded under grant number: ECR/2018/000204 by the Science & Engineering Research Board (SERB).

References

1. Salamon J, Gómez E (2012) Melody extraction from polyphonic music signals using pitch contour characteristics. IEEE Trans Audio Speech Lang Process 20(6):1759–1770
2. Ringer AL (2013) Melody. grove music online 1(3):8. Oxford Music Online (last checked Jan. 2013). http://www.oxfordmusiconline.com/subscriber/article/grove/music/18357
3. Salamon J, Rocha B, Gómez E (2012) Musical genre classification using melody features extracted from polyphonic music signals. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, New York, pp 81–84
4. Paiva RP, Mendes T, Cardoso A (2006) Melody detection in polyphonic musical signals: exploiting perceptual rules, note salience, and melodic smoothness. Comput Music J 30(4):80–98
5. Salamon J, Gómez E, Ellis DP, Richard G (2014) Melody extraction from polyphonic music signals: approaches, applications, and challenges. IEEE Sign Process Mag 31(2):118–134
6. Poliner GE, Ellis DP, Ehmann AF, Gómez E, Streich S, Ong B (2007) Melody transcription from music audio: approaches and evaluation. IEEE Trans Audio Speech Lang Process 15(4):1247–1256
7. Goto M (2004) A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. Speech Commun 43(4):311–329
8. Hsu C-L, Jang J-SR (2010) Singing pitch extraction by voice vibrato/tremolo estimation and instrument partial deletion. In: ISMIR, pp 525–530
9. Disuanco J, Tan V, de Leon F (2015) Study of automatic melody extraction methods for Philippine indigenous music. In: 2015 IEEE international conference on control system, computing and engineering (ICCSCE). IEEE, New York, pp 464–469
10. Ryynänen MP, Klapuri AP (2008) Automatic transcription of melody, bass line, and chords in polyphonic music. Comput Music J 32(3):72–86
11. Cancela P (2008) Tracking melody in polyphonic audio. In: Proceedings of MIREX
12. Jo S, Joo S, Yoo CD (2010) Melody pitch estimation based on range estimation and candidate extraction using harmonic structure model. In: Eleventh annual conference of the international speech communication association
13. Dressler K (2011) An auditory streaming approach for melody extraction from polyphonic music. In: ISMIR, pp 19–24
14. Klapuri A (2004) Signal processing methods for the automatic transcription of music. Tampere University of Technology, Finland
15. Yeh T-C, Wu M-J, Jang J-SR, Chang W-L, Liao I-B (2012) A hybrid approach to singing pitch extraction based on trend estimation and hidden Markov models. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, New York, pp 457–460

16. Rao V, Rao P (2010) Vocal melody extraction in the presence of pitched accompaniment in polyphonic music. IEEE Trans Audio Speech Lang Process 18(8):2145–2154
17. Durrieu J-L, Richard G, David B, Févotte C (2010) Source/filter model for unsupervised main melody extraction from polyphonic audio signals. IEEE Trans Audio Speech Lang Process 18(3):564–575
18. Tachibana H, Ono T, Ono N, Sagayama S (2010) Melody line estimation in homophonic music audio signals based on temporal-variability of melodic source. In: 2010 IEEE international conference on acoustics, speech and signal processing. IEEE, New York, pp 425–428
19. Huang P-S, Chen SD, Smaragdis P, Hasegawa-Johnson M (2012) Singing-voice separation from monaural recordings using robust principal component analysis. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, New York, pp 57–60
20. Rafii Z, Pardo B (2012) Repeating pattern extraction technique (repet): a simple method for music/voice separation. IEEE Trans Audio Speech Lang Process 21(1):73–84
21. Park H, Yoo CD (2017) Melody extraction and detection through LSTM-RNN with harmonic sum loss. In: 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, New York, pp 2766–2770
22. Kum S, Oh C, Nam J (2016) Melody extraction on vocal segments using multi-column deep neural networks. In: ISMIR, pp 819–825
23. Su L (2018) Vocal melody extraction using patch-based CNN. In: 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, New York, pp 371–375
24. Huang Y, Liu G (2018) Melody extraction based on deep harmonic neural network. In: 2018 international conference on network infrastructure and digital content (IC-NIDC). IEEE, New York, pp 174–178
25. Basaran D, Essid S, Peeters G (2018) Main melody extraction with source-filter NMF and CRNN
26. Fan Z-C, Jang J-SR, Lu C-L (2016) Singing voice separation and pitch extraction from monaural polyphonic audio music via DNN and adaptive pitch tracking. In: 2016 IEEE second international conference on multimedia big data (BigMM). IEEE, New York, pp 178–185
27. Rigaud F, Radenen M (2016) Singing voice melody transcription using deep neural networks. In: ISMIR, pp 737–743
28. Lu WT, Su L et al (2018) Vocal melody extraction with semantic segmentation and audio-symbolic domain transfer learning. In: ISMIR, pp 521–528
29. MIREX (2007) Music information retrieval evaluation exchange (mirex)
30. Bittner RM, Salamon J, Tierney M, Mauch M, Cannam C, Bello JP (2014) Medleydb: a multitrack dataset for annotation-intensive MIR research. ISMIR 14:155–160
31. Tsai W-H, Yu H-M, Wang H-M, Horng J-T (2008) Using the similarity of main melodies to identify cover versions of popular songs for music document retrieval. J Inf Sci Eng 24(6)
32. Marolt M (2008) A mid-level representation for melody-based retrieval in audio collections. IEEE Trans Multimedia 10(8):1617–1625
33. Salamon J, Serra J, Gómez E (2013) Tonal representations for music retrieval: from version identification to query-by-humming. Int J Multimedia Inf Retrieval 2(1):45–58
34. Typke R (2007) Music retrieval based on melodic similarity. In: ASCI
35. Arora V, Behera L (2013) On-line melody extraction from polyphonic audio using harmonic cluster tracking. IEEE Trans Audio Speech Lang Process 21(3):520–530
36. Sutton C (2006) Transcription of vocal melodies in popular music. Master's thesis, Queen Mary, School of Electron. Eng. Comput. Sci., University of London, United Kingdom

Chapter 11
Comparative Analysis of Combined Gas Turbine–Steam Turbine Power Cycle Performance by Using Entropy Generation and Statistical Methodology

Kaushalendra Kumar Dubey and R. S. Mishra

1 Introduction

The precious quantity of heat is lost during the combining of two different power generation system which will affect the plant performance. The first law of thermodynamics defines the energetic criteria of system. It is incapable to give the information about thermal deficiency due to irreversibilities in process. The estimation of real performance of thermal system and quality of energy are possible by the exergy analysis. The entropy generation approach established the energy–exergy analysis for the different thermal utilities of thermal power plant, and statistical methodology helps to compare the performance parameters like power output, gas turbine thermal efficiency, fuel consumption and exergetic efficiencies at different combustible gases which are utilized in combustion process. The Taguchi optimization method is used for the parametric analysis in the present work for findings of possible overall efficiency, heat loss and gas turbine performance with the combination of operating temperature, pressure ratio and different fuel gases. The main objective of energy–exergy modeling is to identify losses in components and true performance of plant.

Several researches have been conducted on performance analysis of power plant and parametric optimization by using statistical analysis tool. The authors conducted exergy destruction analysis of gas turbine-based cogeneration thermal plant and observed the effect of compression ratio, steam pressure, turbine inlet temperature, refrigeration temperature, etc. Another research analyzed the exergy performance of ejector-absorption refrigeration system which is efficient by 8% in the application of

K. K. Dubey (✉)
School of Mechanical Engineering, Galgotias University, Greater Noida, Uttar Pradesh, India
e-mail: dubey.kaushalendra@gmail.com

R. S. Mishra
Department of Mechanical Engineering, Delhi Technological University, New Delhi, India

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_11

wastage-heat-combined power system with refrigeration effect [1–3]. Sapele, Nigeria, based low-power generation category steam power plant of 75 Mega Watt (MW) has computed maximum exergy destruction as 87.3% in boiler and plant exergy efficiency as 11.03%. The higher energy loss in boiler signifies the treatment of boiler utilities. The exergonomic concept for different power generation techniques is reviewed. Authors addressed the boiler and condenser component which have major exergy destruction in case of Rankine power cycle, whereas combustion chamber has major exergy destruction in case of gas turbine power plants [4–6]. Authors explain about exergy as the maximum rate of work, which is the theoretical limitation of system and undoubtedly shows that no real system can conserve exergy and can be recovered [7]. The exergetic performance of the re-powered Rankine cycle with gas turbine concluded that combustion chamber and boiler have more irreversibility in GT and ST plants, respectively. Researcher developed a computational model for the parametric investigation. Their analysis involved the effect of compression ratio of used combustion gas, turbine inlet temperature and boiler pressure on the energetic and exergetic performance [8]. The energy–exergy modeling is the performance in terms of energy–exergy analysis of several thermal systems like ammonia–sodium thiocyanate absorption system, organic Rankine cycle and combined cooling power generation plant (CCPP), it provides the opportunity to improve and identify the location of losses, losses mainly occur during the operation, and it is remarked as exergy destruction. The performance assessment of energy and exergy analysis of various configurations of the CCPP plant for waste heat recovery through HRSG and equivalent payback period is also estimated [9–11]. The physical system's simulation and quantitative modeling is mainly dependent on the mathematical models for accuracy and optimized design with multitude of parameters, furthermore, statistical mathematics implemented for error estimation, parametric comparison and optimization for both actual and model data examination. For modeling and analysis, generally traditional mathematical approaches have been incorporated, but complex problems can be solved by soft computing, fuzzy logic and neural networks approaches [12]. The another type of plant data analysis deals with statistical modeling techniques like regression model, least square method, maximum likelihood estimator (MLE) method, autoregression integrated moving average model (ARIMA), and multiple linear regression (MLR), artificial neural network (ANN), root mean square error, etc., for error identification, parametric comparison and complex problem solving [13–18]. The parametric analysis of geothermal power plant is estimated by using multiple linear regression method, and ANOVA-I and II. The another mathematical tools like ANN, Toguchi have been conducted for organic Rankine cycle performance analysis and compared with R-analysis tool for error estimation and parametric comparison in recent research work [19, 20].

2 Brief of Combined GT-ST Power Generation System

The proposed plant is integrated with gas turbine and steam turbine with HRSG system. The exhaust temperature from gas turbine has the potential to run the steam turbine plant through HRSG. The steam power plant thermal efficiency is fixed with maximum possible efficiency, 36%, because the gas turbine power cycle works as topping cycle, and the variation in GT plant will affect the overall performance of plant. The layout of proposed plant is shown in Fig. 1, and operational conditions and combustible gas properties have been listed in Tables 1 and 2.

The combined GT-ST plant is integrated with GT power generation and ST power generation. The following process is given below in Fig. 1 as per the mentioned plants schematic.

Process: 1-2-Isentropic compression of atmospheric air in GT plant.
Process: 2-3-Combustion of gas in combustion chamber-1 of GT plant.
Process: 3-4-Isentropic flue gas expansion in gas turbine of GT plant.
Process: 4-5-Combustion of flue gas in combustion chamber-2 of GT plant.
Process: 5-6-Stack flows out.

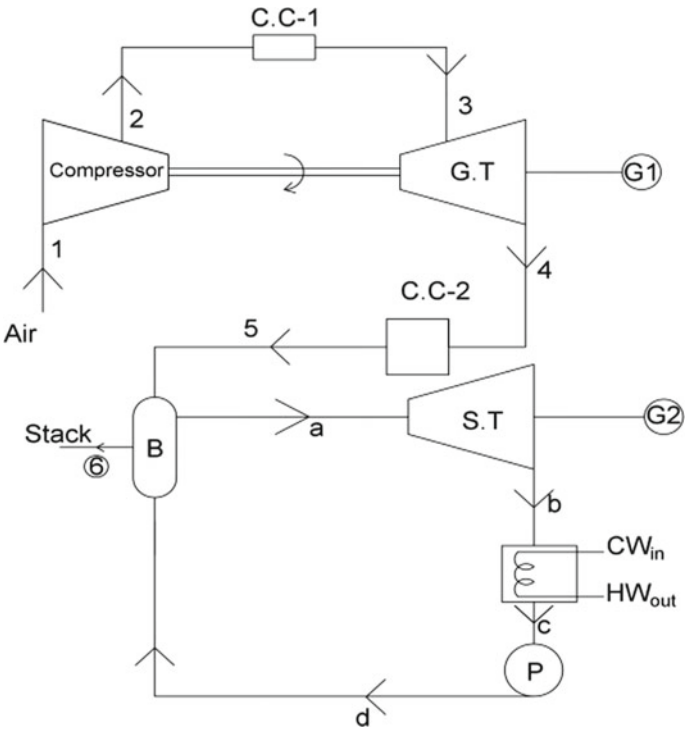


Fig. 1 Thermodynamic schematic of combined GT-ST plant

Table 1 Operating condition of GT-ST plant

Plant components/parameters	Unit and value	Plant components/parameters	Unit and value
Compressor inlet condition ($P1$, $T1$)	1 bar, 25 °C	Specific heat ratio of air (γ_{air})	1.4
Compressor pressure ratio ($P2/P1$)	8	Steam condition at inlet of steam turbine	40 bar, 425 °C
Gas turbine inlet gas temperature ($T3$)	900 °C	Condenser pressure	0.04 bar
Pressure drop in combustion chamber	3%	Feed water temperature at HRSG	170.4 °C
Compressor efficiency (η_c)-manufacturer efficiency	88%	ST efficiency (η_{ST})	82%
GT efficiency (η_{GT})—manufacturer efficiency	88%	Pressure drop of gas in the HRSG	5 kPa
Calorific value of liquid ethane as fuel (CV_f)	44.43 MJ/kg	Steam flow rate $= w_s$	105 TPH
Specific heat of air (Cp_{air})	1.006 kJ/kg K	GT outlet pressure ($P4$)	1.05 bar
Specific heat of gas (Cp_{gas})	1.148 kJ/kg K	Condenser inlet steam or ST outlet flow	105 TPH or 30 kg/s

Table 2 Properties of combustible gas for GT plant

S. No.	Name of fuel	Chemical formula	Heat value (MJ/kg)	Cp/Cv	γ	Combustion reaction (CaHb + c * O_2 – D * CO_2 + E * H_2O)	Stoichiometric combustion $A/F(A/F = C * O_{mol\ wt}/(23.2\% * Fuel_{mol\ wt}))$
1	Methane	CH_4	55.5	2.22/1.70	1.3	$CH_4 + 2O_2 - CO_2 + 2H_2O$	17.24
2	Ethane	C_2H_6	51.9	1.53/1.48	1.03	$C_2H_6 + 7/2O_2 - 2CO_2 + 3H_2O$	16.09
3	Propane	C_3H_8	50.4	1.67/1.48	1.12	$C_3H_8 + 5O_2 - 3CO_2 + 4H_2O$	15.6
4	Butane	C_4H_{10}	49.1	1.67/1.53	1.091	$C_4H_{10} + 13/2O_2 - 4CO_2 + 5H_2O$	15.45
5	Pentene	C_5H_{12}	48.6	1.66/1.52	1.092	$C_5H_{12} + 8O_2 - 5CO_2 + 6H_2O$	15.32

Process: 6-a-Steam generation in HRSG of ST plant by utilizing exhaust heat from GT plant.
Process: a-b-Isentropic steam expansion and steam turbine working performance of ST plant.
Process: b-c-Steam condensation and heat rejection of ST plant.
Process: c-d-Pumping of condensed water and makeup water and supplying it to HRSG of ST plant.

3 Plant Operation Condition

3.1 Thermodynamic Analysis and Statistical Modeling of Combined GT-ST Power Plant

The governing equation of analysis is based on energy–mass conservation and entropy generation principle of thermodynamics. The linear regression method is used in terms of Taguchi model for parametric optimization for the GT-ST plant performance. Mass and energy balance equations have been applied in all thermal utilities. In order to simplify the analysis, some fundamental assumptions and proposed analysis are adopted from PK Nag for combined GT-ST model [20, 21]:

1. The thermodynamic process is assumed as steady flow with consideration of control volume (CV) system.
2. The isentropic performance is considered for compressor, pumps and both GT and ST turbines.

Apply mass–energy and first law energy equation in all utilities of GT plant. The thermodynamic relation and equations for power plant components are as follows

Compressor
The work done by the compressor is the function of operating condition of air intake through compressor, and the outlet temperature can be expressed as follows:
$P1 = 1$ bar, $P2 = 8$ bar, $T1 = 25\ ^{\circ}\text{C} = 298$ K and $\eta_c = 0.88$

$$\frac{T2}{T1} = \left(\frac{P2}{P1}\right)^{\frac{(\gamma-1)}{(\gamma*\eta_c)}} \tag{1}$$

$$W_{\text{comp}} = m_{\text{air}} \times \text{Cp}_{\text{air}}(T2 - T1) \tag{1a}$$

Combustor
The pressure drop (ΔPcc) across the Cis as follows

$$P3/P2 = (1 - \Delta\text{Pcc}) \tag{2}$$

Assume flow rate of flue gas as 1 kg/s and that of fuel f kg/s

$$\text{The mass flow rate of air} = (1 - f)\text{kg/s} \tag{3}$$

Therefore,

$$f \times \text{CV}_f = m_f \times \text{Cp}_{\text{gas}}(T3 - T4) - (1 - f)\text{Cp}_{\text{air}}(T2 - T0) \tag{4}$$

$$\text{Air Fuel Ratio} = \frac{A}{F} = \frac{(1 - f)}{f} \tag{5}$$

Octane combustion reaction occurs as follows (it will vary for all mentioned fuel gases)

$$\text{C}_8\text{H}_{18} + 12.5\ \text{O}_2 = 8\text{CO}_2 + 9\text{H}_2\text{O} \tag{6}$$

For stoichiometric combustion A/F ratio $= \frac{(12.5 \times 32)}{(0.232 \times 114)} = 15.12$
($\text{C}_8\text{H}_{18} = 12 * 8 + 1 * 18 = 114$)

Gas Turbine

The GT outlet temperature is estimated by isentropic efficiency (η_{GT}), and the GT inlet temperature ($T3$) and gas turbine pressure ratio ($P3/P4$) are as follows:

$P4 = 1.05$ bar, $T3 = 900$ °C

$$\frac{T3}{T4} = \left(\frac{P3}{P4}\right)^{\frac{(\gamma-1)(\eta_{\text{GT}})}{(\gamma)}} \tag{7}$$

Heat Recovery Steam Generator (HRSG)

The flue gas temperature at inlet of HRSG is calculated by

Let the pinch-point temperature differences ($T5 - Tf$) be 30 °C

(pinch point at exit of GT and inlet of ST)

$Tf = (T\text{sat})_{40\ \text{bar}} = 250.4$ °C

$T5 = Tf + 30$ °C

(The enthalpy and entropy values taken from the steam properties table are as per the operating condition)

From mass and energy balance between GT and ST,

$$m_{\text{gas}} \times \text{Cp}_{\text{gas}}(T4 - T5) = m_{\text{steam}}\left(h_a - h_f\right) \tag{8}$$

And

$$\text{air flow rate entering the compressor} = m_{\text{air}} = (1 - f)m_{\text{gas}} \tag{9}$$

$$\text{Fuel mass flow rate } m_{\text{fuel}} = w_f = f \times m_{\text{gas}} \tag{9a}$$

Then, energy interaction between HRSG and stack flow, and temperature of stack flue gas are as follows

$$1.14(482 - T6) = 0.106(3272 - 721.1) \tag{9b}$$

Stack flow temperature $= T6$

Power Output of GT-ST Plant

The combined GT-ST plant power output is achieved by the net workdone by both gas turbine and steam turbine, and the total work of GT and ST plant is given by

$W_{\text{total}} = W_{\text{ST plant}} + W_{\text{GT plant}}$

But, the steam turbine workdone is a function of isentropic efficiency of steam turbine, which is given already, so thermal equation for steam turbine work in terms of enthalpy across inlet and outlet of ST is as follows

$$W_{\text{ST}} = w_s(h_a - h_{bs}) \times \eta_{\text{st}} \tag{10}$$

From enthalpy and entropy equation between steam turbine expansion process,

$$h = h_f + x h_{fg} \text{ and } s = s_f + x\, s_{fg} \tag{11}$$

After expansion of steam, steam quality also occurs in terms of dryness fraction, and entropy is also changed as follows

So,

$$s_{bs} = s_{f_b} + x_{bs} \times s_{fg_b} \tag{12}$$

And

$$\text{Enthalpy at condenser line} = h_{bs} = h_{f_b} + x_{bs} \times h_{fg_b} \tag{13}$$

Workdone by GT plant depends on the difference between gas turbine work and compressor work consumption, and the combined equation for W_{GT} and W_{COMP} is as follows

$$W_{\text{GT}}\text{plant} = W_{\text{GT}} - W_{\text{COMP}} \tag{14}$$

$$W_{\text{GT}}\text{plant} = m_{\text{gas}} \times \text{Cp}_{\text{gas}}(T3 - T4) - m_{\text{air}} \times \text{Cp}_{\text{air}}(T2 - T1) \tag{15}$$

When two plants are combined, there is always some heat loss. If heat rejected by GT plant as topping cycle is absorbed by ST plant as bottoming cycle, then lost heat is the coefficient in the exhaust stack

$$X_L = \left(\frac{w_g \times \mathrm{Cp}_g(T6 - T1)}{w_f \times \mathrm{CV}_f}\right) \tag{16}$$

For the overall thermal efficiency, the topping and bottoming cycle are evaluated by considering heat lost between topping and bottoming cycle, The overall efficiency of plant is as follows

$$\eta_{\text{overall plant}} = \eta_{\text{ST plant}} + \eta_{\text{GT plant}} - \eta_{\text{ST plant}} \times \eta_{\text{GT plant}} - \eta_{\text{ST plant}} \times X_L \tag{17}$$

The plant efficiency of GT and ST plant depends on workdone by both turbines and heat supplied through combustion chamber and HRSG, respectively, in GT and ST plant. The equations for efficiencies are as follows

$$\eta_{\text{STplant}} = (h_a - h_b)/(h_a - h_e) \tag{18}$$

Put all values of enthalpy from Eqs. 10–13

$$\eta_{\text{GTplant}} = (W_{\text{GT}})/\left(w_f \times \mathrm{CV}_f\right) \tag{19}$$

From Eq. 17 $\boldsymbol{\eta_{\text{overall}}}$

Exergy Analysis

$$\text{Assume exergy flux } \psi = (\Delta G_0/\Delta H_0) = 1.0401 + 0.1728\,(h/c) \tag{20}$$

where $\Delta H_0 = w_f \times (\mathrm{CV})_0$

And (h/c)—mass ratio of hydrogen to carbon in octane (C8H18) fuel.

From Eq. 20, exergy input $= \Delta G_0 = \psi \times \Delta H_0$

$$T_0 \Delta S_0 = \Delta G_0 - \Delta H_0 \tag{21}$$

Exergy destruction of components due to irreversibility in process, The all equations of irreversibility are in terms of TdS form, where temperature T is ambient temperature as T_0

Compressor

$$\text{Rate of energy dissipation in compressor}\left(I_{\text{comp}}\right) = w_a T_0 \left(\mathrm{Cp}_a \ln \frac{T2}{T1} - R_a \ln \frac{P2}{P1}\right) \tag{22}$$

But, $R_a = C_p\left(\frac{\gamma-1}{\gamma}\right)$.

Combustion Chamber

The irreversibility in CC is estimated by the energy balance between product, reactants, air and used fuel, respectively.

$$\text{Icc} = T_0\left[w_g\left\{\text{Cp}_g \ln\frac{T3}{T0} - \text{R}_g \ln\frac{P3}{P0}\right\} - \left\{w_g \text{Cp}_g \ln\frac{T2}{T0} - w_a R_a \ln\frac{P2}{P0}\right\} + \Delta S\right] \tag{23}$$

Gas Turbine

$$\text{Rate of energy lost or work lost in GT} = I_{\text{GT}} = w_g T_0\left[\text{Cp}_g \ln\frac{T4}{T3} - R_g \ln\frac{P4}{P3}\right] \tag{24}$$

HRSG

Rate of energy lost in heat recovery steam generator—I_{HRSG}

$$I_{\text{HRSG}} = T_0\left[w_s(s_a - s_e) + \text{Cp}_g \ln\frac{T6}{T4} - R_g \ln\frac{P6}{P4}\right] \tag{25}$$

Steam Turbine

Rate of energy or work lost in the steam turbine $= I_{\text{ST}}$

$$I_{\text{ST}} = T_0 w_s(s_b - s_a) \tag{26}$$

Exergy lost due to exhaust flue gas

$$I_{\text{EXHFLUEGAS}} = \int_{T0}^{T6}\left(1 - \frac{T0}{T}\right)dQ = w_g \times \text{Cp}_g\left[(T6 - T0) - T0 \ln\frac{T6}{T0}\right] \tag{27}$$

$$\text{Exergetic Efficiency} = \eta_{\text{EX}} = \frac{\text{Total Output}}{\text{Exergy Input}} \tag{28}$$

Linear Regression Analysis—Taguchi Method

The Taguchi-based linear regression model is used in this analysis for both overall and gas turbine. The five different gases have been used in combustion chamber at different pressure ratios and operating temperature ranges.

Taguchi model helps to identify the suitable combination of input factors for optimized result of plant operation.

The mathematical expression for expected value of performance parameters as follows which give the optimized result in all set of available factors and level of operating conditions.

$$A\overline{P} + B\overline{Q} + C\overline{R} - 2\overline{Y} \tag{29}$$

where A, B, C are response level values and P, Q, R are mean value of response. Y is average value of total runs of operating parameters.

In the present statistical model, five levels of three factors (pressure ratio, operating temperature and type of fuel gases) have been considered. And, overall efficiency, gas turbine efficiency, heat loss in GT plant, exergy of compressor, exergy of combustion chamber and exergy of gas turbine-like resultant parameters are investigated. All factors and its level are given in Table 3.

Above-mentioned factors and its level are analyzed by L25 model of Taguchi analysis.

Taguchi Model

Taguchi orthogonal array design

L25(5**3)

Factors: 3

Runs: 25

Columns of L25(5**6) Array 1 2 3

Table 3 Factors and levels of combined GT-ST power plant

Factors (input operational factor)	Level				
	1	2	3	4	5
A-pressure ratio in bar	6	7	8	9	10
B-operating temperature at inlet of GT in °C	500	600	700	800	900
C-fuel gases	Methane	Ethane	Propane	Butane	Propane

Taguchi Analysis

Response Table for Means

Level	A	B	C
1	39.80	32.60	**45.80**
2	42.00	38.00	39.60
3	41.00	42.80	40.60
4	41.60	45.60	40.40
5	**41.80**	**47.20**	39.80
Delta	2.20	14.60	6.20
Rank	3	1	2

Analysis of Variance for SN ratios

Source	DF	Seq SS	Adj SS	Adj MS	F	P
A	4	0.6840	0.6840	0.1710	0.38	0.816
B	4	35.6924	35.6924	8.9231	20.01	0.000
C	4	6.1002	6.1002	1.5251	3.42	0.044
Residual Error	12	5.3525	5.3525	0.4460		
Total	24	47.8290				

4 Results and Discussion

The observations and results of combined GT-ST plant are mentioned in tables and different figures. The general plant output and exergy of components are given in Tables 4 and 5, respectively. The present results consider ethane gas as a fuel gas in gas turbine plant with all the permissible conditions for plant working, which is already mentioned in Table 1.

The major highlights of combined GT-ST results in Table 4 show 38, 27 and 41.7 of ST plant efficiency, GT plant efficiency and combined GT-ST plant efficiency, respectively, with the 35.4% of heat loss during the GT and ST cycle combination. The 247 °C of stack flow temperature from HRSG has huge potential for cooling or refrigeration plan works as a generator heat source. Table 5 contains the results of actual useful energy of components of plant. The combustion chamber of GT plant has higher energy loss with 41.6%, whereas GT utilizes maximum energy of input.

Table 4 Observations of GT-ST plant performance

W_{STplant}	29 MW	$m_{\text{gas}}/m_{\text{air}}/m_{\text{fuel}}$	275.4 kg/s/271.3 kg/s/4.46 kg
$W_{\text{GT plant}}$	53 MW	Heat lost during combining of GT-ST cycle $= X_L$	35.4%
η_{STplant}	38%	Stack flow temp ($T6$)	247 °C
η_{GTplant}	27%	$\eta_{\text{overall plant}}$	41.7%

Table 5 Exergy balance of GT-ST power plant components

Compressor Ex_loss	6613 kW	Exhaust flue gases Ex_loss	17,760 kW = 8.3%
Combustion chamber Ex_loss	88,661 kW = 41.6%	Total I_{LOSS} or destruction = $\sum I$loss	132,945
Gas turbine Ex_loss	5646 kW = 2%	Exergy input (ΔG_0)	212,810 kW
HRSG Ex_loss	7583 kW = 3.5%	Total output	82,000 kW
Steam turbine Ex_loss	6412 kW = 3%	Exergetic efficiency = η_{EX}	**38.5%**

The overall efficiency of plant is estimated as 41.7%, but the actual plant performance is calculated as exergetic efficiency as 38.5%. This result is recommended for the plant operation as per the operating conditions for ethane gas.

4.1 Effect of Performance Parameters on Plant

4.1.1 Effect of Compression Ratio on Overall Efficiency with Different Gases

The compression ratio helps to enhance the gas turbine inlet temperature, and higher temperature of GT inlet gives higher thermal efficiency. Figure 2a–e are showing the effect of compression ratio in overall efficiency of combined GT-ST plant with five different gases burning in combustion chamber of GT plant. Methane gas has always higher efficiency with high CR and temperature values, whereas other gases have valuable output with all ranges of CR values. The performance is also influenced by the heat loss during the coupling of GT-ST cycle.

4.1.2 Effect of Heat Loss and Gas Turbine Efficiency in Overall Efficiency of Plant

The overall plant output is greatly affected by the heat loss. The maximum possible GT efficiency is achieved as 42% at 900 °C with methane gas firing as shown in Fig. 3a. The heat supply through combustion chamber is converted into GT workdone and GT cycle heat rejection further, but rejected heat of GT cycle is utilized through HRSG of steam power plant for ST work output. If heat loss is more, then overall performance will decrease as shown in Fig. 3b. At low temperature of gas turbine inlet temperature, heat loss is more, but at higher temperatures, rejected heat is completely utilized, and minimum heat loss enhances the overall performance which is justified in Eq. 17.

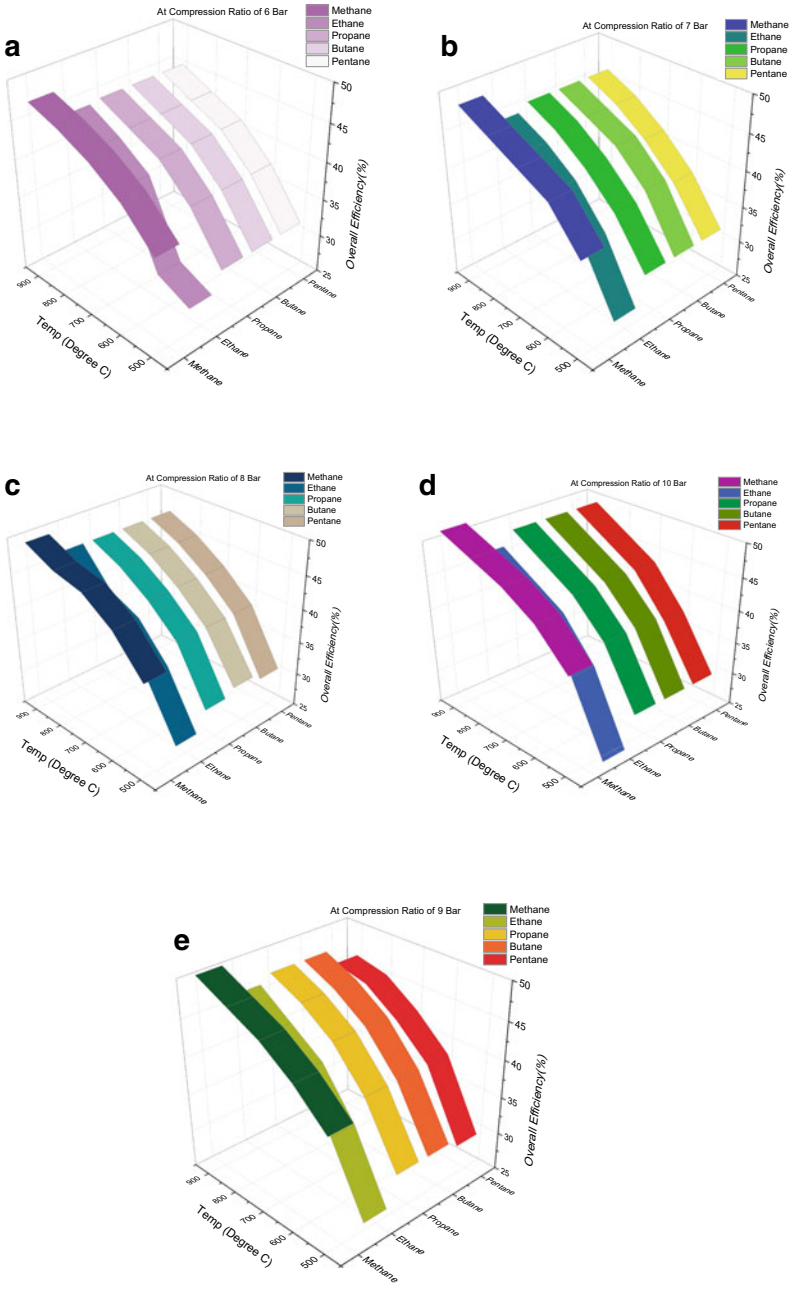


Fig. 2 **a** $\eta_{overall}$ at 6 bar of pressure ratio, **b** $\eta_{overall}$ at 7 bar of pressure ratio, **c** $\eta_{overall}$ at 8 bar of pressure ratio, **d** $\eta_{overall}$ at 9 bar of pressure ratio and **e** $\eta_{overall}$ at 10 bar of pressure ratio

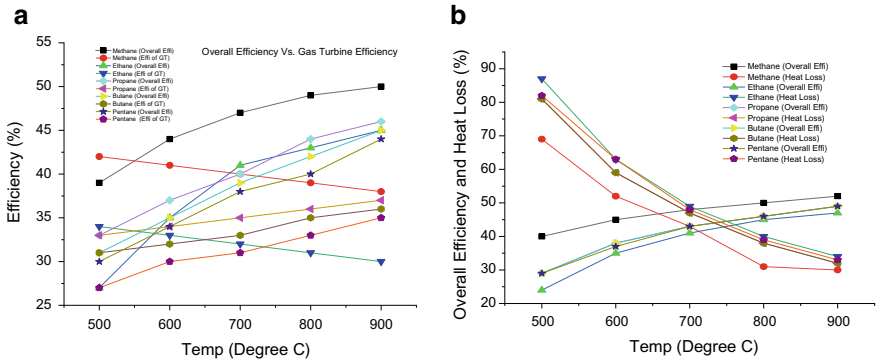


Fig. 3 **a** Variation of overall efficiency with gas turbine efficiency and **b** variation of overall efficiency with heat loss

4.1.3 Exergy Loss

The maximum loss in terms of exergy loss occurs in combustion chamber of GT plant as 41.6% in this analysis, and it is due to the incomplete combustion, unburnt fuel and heat loss to the surroundings. Exergy destruction of all components is shown in Fig. 4. Stack flow heat loss is also considered as major factor in exergy destruction at high temperature of stack flow. It should not be more than 250 °C.

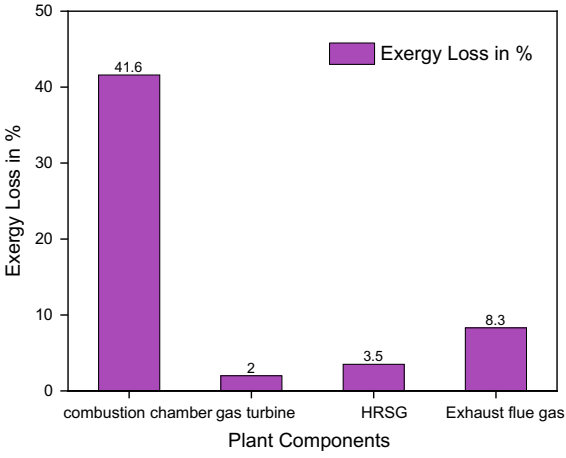


Fig. 4 Components exergy loss

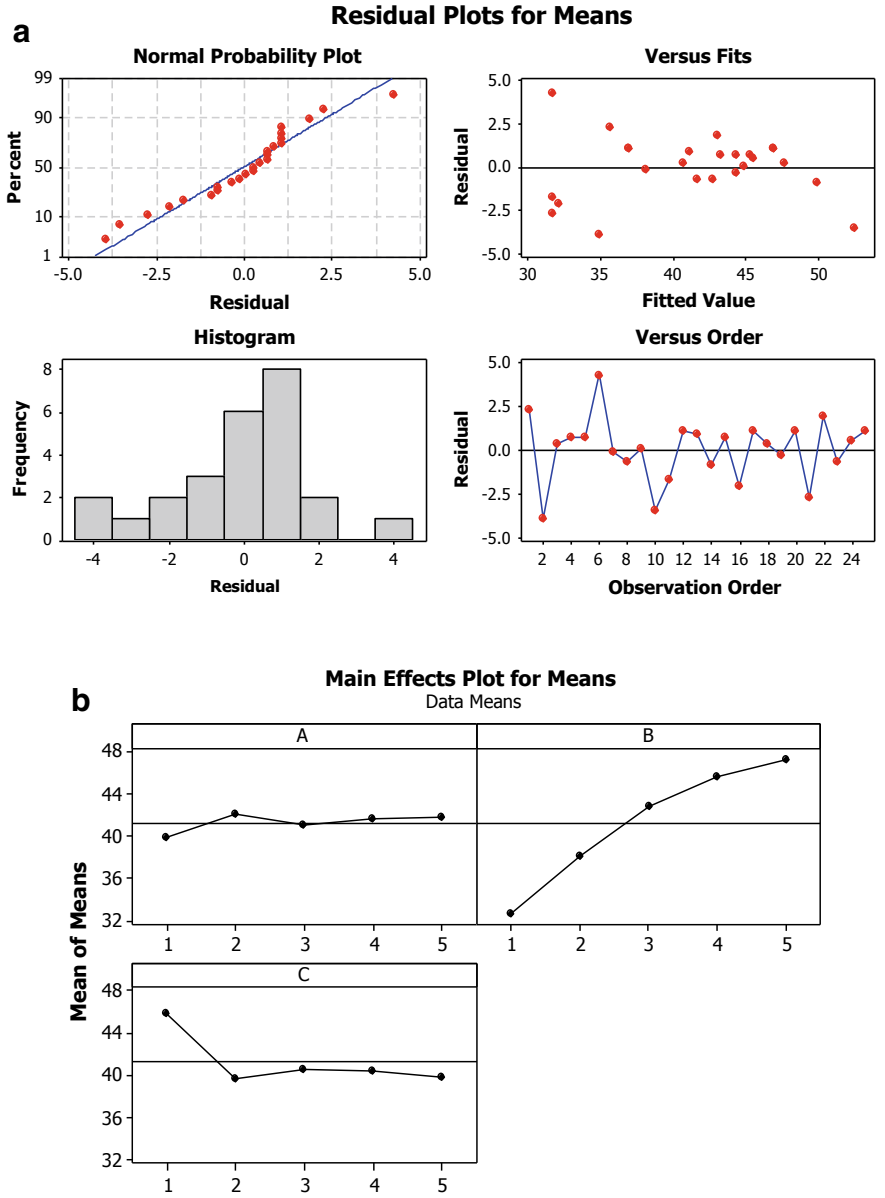


Fig. 5 **a** Linear regression plot of overall efficiency and **b** means value of level for overall efficiency

4.1.4 Optimized Result for Overall Efficiency and Heat Loss by Statistical Modeling

Figure 5a, b explains the suitable combination for performance of GT-ST plant which is achieved by the 5-5-1 combination of A, B, C factors, and it concludes that the 10 bar of compression ratio with 900 °C of gas turbine inlet temperature of maximum temperature of GT-ST plant gives the 47.6% of overall efficiency by using methane gas as a fuel in CC of gas turbine and at same combination estimates 30.6% of minimum heat loss. The residual plot in Fig. 5a also gives the perfect fit curve of linear regression model that means the factors' combination of 5-5-1 is suitable for this analysis under the different levels of pressure ratio, operating temperatures and gaseous fuel.

From Eq. 29—$A\overline{P} + B\overline{Q} + C\overline{R} - 2\overline{Y}$ (Y is average value all 25 values of efficiencies)
where P, Q and R are taken as 5-5-1, $A5 + B5 + C1 - 2(43.1)$
From response table of Taguchi analysis—$41.8 + 47.2 + 45.8 - 2(43.76)$
Optimized value of overall efficiency—47.23% (near to actual value of overall efficiency)
The actual plant overall efficiency estimated by thermodynamic approach is 47.6% and optimized value of same parameter is 47.23% by using linear regression method of Taguchi model.

5 Conclusions

Energetic criteria deal by the first law of thermodynamics, whereas exergetic criteria defined by the second law of thermodynamics, and it gives the real performance of thermal systems. The present analysis carried energy and exergy approach of thermodynamics with mathematical statistical modeling for optimized results with the different levels of performance factors. The highlights of this analysis have been mentioned below

1. The performance of combined GT-ST plant is influenced by gas turbine performance and as well as affected by heat loss. The temperature value of stack flow at the compression of 8 bar and 900 °C has sufficient potential for heat recovery and employment of trigeneration system further.
2. The assessment of exergy destruction is important for actual plant performance. The determination of exergy loss estimation of various components helps to identify the components for controlling heat loss or re-designing of identified parts. Combustion chamber and heat recovery steam generator have major exergy loss in GT and ST plant, respectively. Steam turbine exergy destruction is more than gas turbine destruction due to pressure reduction and steam condensation during the expansion of steam. It indicates that the temperature at the inlet of steam

turbine must be high as per materialistic temperature limit of turbine material. Superheated steam is required for the achievable turbine performance.

3. The three major input factors of this analysis in order of compression ratio, gas turbine inlet temperature and gaseous fuel type have been considered with the five levels of different operating conditions. The GTIT factor has significant impact on plant performance, whereas CR values have minimum influence. The CR value is most effective for methane and ethane gas fuel at high temperature of GTIT, and other gaseous fuel have no significant output in CR values, as shown in Fig.3.
4. The linear regression fit curve is also perfect for the present analysis, and the combination of all three factors are in order of 5-5-1 for optimized result as 10 bar of CR generates 900 °C with the methane gas burning that estimates 47.6% of overall efficiency and 30.1% of minimum heat loss.
5. The actual efficiency is achieved by the approach of energy–exergy analysis, and expected result is obtained by the Taguchi method of DOE. Both results are more close, so it justifies the present statistical model which is suitable for this multi-parametric analysis.

The both approaches of analysis give the comparative and optimized solution for plant operation at suitable operating condition. The present statistical model of DOE helps to compare the actual and expected values for plant performance study, machinery optimization, components identification for controlling heat loss, best combination of factor with the large range of input operating parameters.

References

1. Khaliq A, Kaushik SC (2004) Thermodynamic performance evaluation of combustion gas turbine cogeneration system with reheat. J Appl Thermal Eng 24:1785–1795
2. Khaliq A, Kaushik SC (2004) Second-law based thermodynamic analysis of brayton/rankine combined power cycle with reheat. Appl Energy 78(2):179–197
3. Khaliq A (2009) Exergy analysis of gas turbine trigeneration system for combined production of power heat and refrigeration. Int J Refrig 32(3):534–545. https://doi.org/10.1016/j.ijrefrig.2008.06.007
4. CO Osueke, AO Onokwai, AO Adeoye (2015) Energy and exergy analysis of 75 MW steam power plant in Sapele (Nigeria). Int J Innov Res Adv Eng 2(6): 169–179. ISSN-2349-2163
5. Kumar R (2017) A critical review on energy, exergy, exergonomics and economic (4-E) analysis of thermal power plants. Eng Sci Technol Int J: 283–292. ISSN-2215-0986
6. Chen LG, Feng HJ, Sun FR (2013) Exergy optimisation of irreversible closed Brayton cycle combined cooling, heating and power plant. J Energy Inst 86(2):97–106. https://doi.org/10.1179/1743967112z.00000000048
7. Ahmadi P, Rosen MA, Dincer I (2011) Greenhouse gas emission and exergo environmental analyses of a trigeneration energy system. Int J Greenhouse Gas Control 6:1540–1549
8. Tiwari AK, Hasan MM, Islam M (2012) Exergy analysis of combined cycle power plant: NTPC Dadri, India. Int J Thermodyn 16 (1):36–42. http://dx.doi.org/10.5541/ijot.443
9. Singh O, Kaushik SC (2013) Thermodynamic evaluation and optitimization of a Brayton-Rankine-Kalina combined triple power cycle. J Energy Convers Manage 71:32–42

10. Ibrahim TK, Mohammed MK, Awad OI, Abdalla AN, Basrawi F, Mohammed MN, Najafi G, Mamat R (2018) A comprehensive review on the exergy analysis of combined cycle power plants. Renew Sustain Energy Rev 90:835–850
11. Li J, Gao G, Kutlu C, Liu K, Pei G, Su Y (2019) A novel approach to thermal storage of direct steam generation solar power systems through two-step heat discharge. Appl Energy 236:81–100
12. Yang L, Entchev E (2014) Performance prediction of a hybrid microgeneration system using Adaptive Neuro-Fuzzy Inference System (ANFIS) technique. Appl Energy 134:197–203
13. Douglas CM, Elizabeth AP, Geoffrey GV (2011) Introduction to linear regression analysis, 5th edn. John Wiley and Sons
14. Scott LJ, Jeremy F (2014) Regression models for categorical dependent variables using Stata, 4th edn. Stata press, Texas
15. Martin WG, Maynard WS (2012) Predictive modeling of multivariable and multivariate data. J Am Stat Assoc: 646–653
16. Au ST, Ma GQ, Wang R (2011) Iterative multivariate regression model for correlated response prediction. Int Conf Cyber Enabled Distrib Comput Knowl Discov 5:55–59
17. McKelvey RD, Zavoina W (1975) A statistical model for the analysis of ordinal level dependent variables. J Math Sociol 4 (1)
18. Karadas M, Celik HM, Serpen U, Toksoy M (2015) Multiple regression analysis of performance parameters of a binary cycle geothermal power plant. Geothermics 54:68–75
19. Larsen U, Pierobon L, Wronski J, Haglind F (2014) Multiple regression models for the prediction of the maximum obtainable thermal efficiency of organic Rankin cycle. Energy 65(1):503–510
20. Tunckaya Y, Koklukaya E (2015) Comparative prediction analysis of 600 MWe coal-fired power plant production rate using statistical and neural based models. J Energy Inst 88(1):11–18
21. Nag PK (2008) Power plant engineering, 3rd edn. Tata McGraw Hill Publishing India Pvt Ltd. ISBN-13: 978-07-064815-9, 10:0-07-064815-8

Chapter 12
Data Mining—A Tool for Handling Huge Voluminous Data

Seema Maitrey and Yogesh Kumar Gupta

1 Introduction

Tremendous and exceedingly huge data is being accumulated recently almost in every field and growing continuously. The precious information is concealed in large databases. It is becoming very difficult and inefficient for researchers to analyze and retrieve knowledge from such huge tomb of data. Data is voluminous, so human intervention is not required, thus results in a rapid and economical way of exploring and analyzing data. Algorithms of data mining are comprised of techniques which existed few years back, i.e., at least 10 years [1]. Now, they are refined with matured, reliable and user-friendly tools in such a manner that they have consistently outperformed the previous methods. Data mining produced information and knowledge that got used in several areas, such as education, health care, finance, science, market analysis, intelligence agencies, internal revenue service, sports, Web education, credit scoring, engineering design and many more [2]. The significant use of data mining in these special areas affects our life in one way or other. It is improved due to the rise in information technology [3]. These fields are making the use of databases technology, parallel computing, distributed computing.

S. Maitrey (✉)
Department of Computer Science and Engineering, KIET Group of Institutions, Ghaziabad, India
e-mail: seema.maitrey@kiet.edu

Y. K. Gupta
Department of Computer Science, Banasthali Vidyapith, Vanasthali, India
e-mail: gyogesh@banasthali.in

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_12

1.1 Unavailability of Past Data

Many businesses and the organizations possess huge voluminous data but were not able to store such vast data in the past. Today, decision support system (DSS) is used in making decisions for the organizations, but a few years ago, the online transaction processing (OLTP) systems were key to business. Due to the resources issue in the 6 past, the usage of any service, e.g., disk space, processing cost, data retrieval, etc., has been costlier in comparison with the current time. Thus, it became a major reason for not storing historical data [3]. In the current time, the disk storage is not a problem and hence has become increasingly affordable. Almost every level of businesses treats their data as a corporate asset over money and uses it for competitive advantage. For instance, looking at the previous purchasing behavior of customers helps in identifying and predicting their upcoming buying manner. Thus, most of the businesses are extracting their archived patterns to evaluate strategies to explore their business growth potential [4]. Almost every organization is investing in setting up their own data warehouse so as to get more and more benefit from the precious information that is concealed in their data [5]. As soon as the data warehouse was set up and build, the data mining process come into focus [6].

1.1.1 New Business Challenges

Although there exist easily affordable systems to store and manage huge collection of data, yet the businesses encounter new challenges. For instance, the concern regarding software systems and hardware devices get sorted out efficiently through large data warehouses. Along with this, the concern was regarding the process required to convert data into information and finally to obtain the modest advantage. By the time, the data storage cost has become reasonable and affordable. But, the hardware configurations like the processor, memory devices, throughput and network remain constraints. The data collected from the past was so voluminous that many IT managers and business analysts do not know where to start [7]. They were unable to manage the massive amounts of data and failed to retrieve useful information from these data. In such critical situation, a process called as data mining came up with the solution. The upcoming process of data mining greatly relies on the technique of sampling. It is highly concentrated on pictorial representations for data exploration, statistical analysis and modeling, and evaluation of the outcome [8]. The database systems are required to become fast in evaluating query and efficient in retrieving information from the explosive collection of data. There accumulated 7 abundant data, but the available DBMS is very time-consuming, so it became essential building an effective database which can successfully work upon such huge collection of data [9]. To handle such situation where there is a mismatch in data collection and the processing power, it is required to make efficient use of the processing power of the CPU and exploit its ability of parallel processing [10]. We know that there have been proposed and done many efforts at parallelizing query evaluation, but there is

still opportunity to enhance existing techniques by merging this advance concept of parallelization [11, 12].

1.1.2 Outline of Chapter

The chapter is organized into five sections. Section 1.1 provides data mining introduction, unavailability and challenges of the past data. Section 1.2 focuses on the background of the earlier techniques and the preferred techniques to handle large databases. Section 1.3 demonstrates the overview of data mining. Section 1.4 gives patterns to accommodate different data. The key idea from data mining to big data is represented under Sect. 1.5.

1.2 Background

Fundamentally, data mining identify patterns and trends hidden in that information which helps to make the decision [13]. Data mining principles have been existing for many years, but gained popularity and acceptance when there was flood of data, i.e., when data resulted in a huge tomb of data or big data. This huge collection of data resulted in the sudden rise of more widespread data mining techniques. Due to the collection of the enormous set of data, the approval of simple and straightforward statistics is no longer adequate [14]. Thus, this required more complex data mining techniques. The precious information which found embedded in a vast group of data is extracted by data mining. It has become one of the remarkable areas of data mining to reveal such hidden information under voluminous datasets [15, 16]. It remains an issue for a long 8 time when required to take out appropriate information from large databases quickly. For this purpose, many techniques are evolved [17].

1.2.1 Earlier Technique for Mining Large Databases

There are several techniques found in data mining. They are clustering, classification, prediction, association, deviation and outlier analysis [18]. Among these techniques, clustering is taken into consideration for the research which would help in mining large databases [19], i.e., to be used in data exploration. The valuable information is hidden in large databases [20].

Preferred Technique to Handle Large Databases Data mining has evolved with seven operations to handle large databases. They are [21]:

- query and reporting, multidimensional analysis and statistical analysis, predictive modeling, link analysis, database segmentation and deviation detection [11].

Some efficient methods, such as fuzzy set theory, approximate reasoning and genetic algorithms, have been made into practical use to handle the very tedious task

of collecting, analyzing and extracting information from huge amount of datasets. These became very useful in decision making. Along with this, the knowledge discovery database (KDD) was also enhanced to retrieve information in logical and efficient manner from any format, such as graph, flow chart, audio, video and many others [20, 22].

1.2.2 Data Mining: A Combination of Several Fields of Study

Data mining, a multi-disciplinary field, can be shown in the figure [21, 23] (Fig. 1):

(i) Statistics: Deals with analyzing numerical data. Required tools are regression, clustering, correlation analysis and Bayesian network. A simple Bayesian network is given below [24] (Fig. 2).
(ii) Artificial Learning: The learning and predicting is based on past or earlier data that can be managed by using the methods—inductive concept learning, conceptual clustering and decision tree induction. Figure shows a decision tree which depicts weather forecasting, as [25] (Fig. 3).

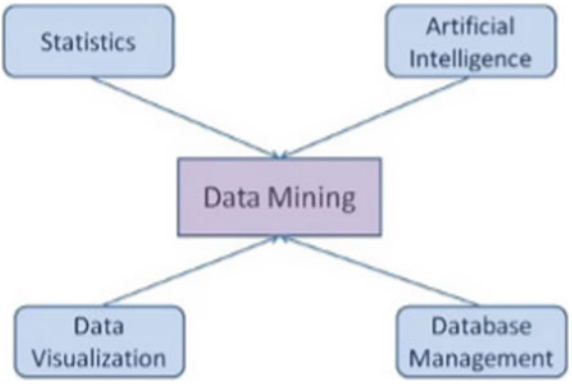


Fig. 1 Data mining: A combination of several fields of study

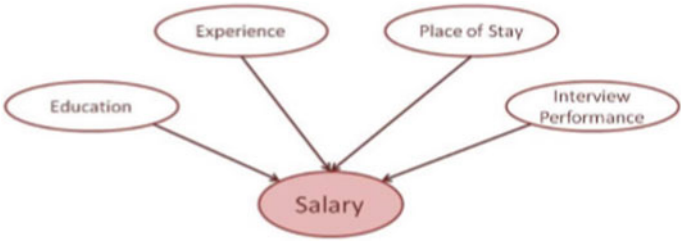


Fig. 2 Bayesian network with the variables shown in the nodes

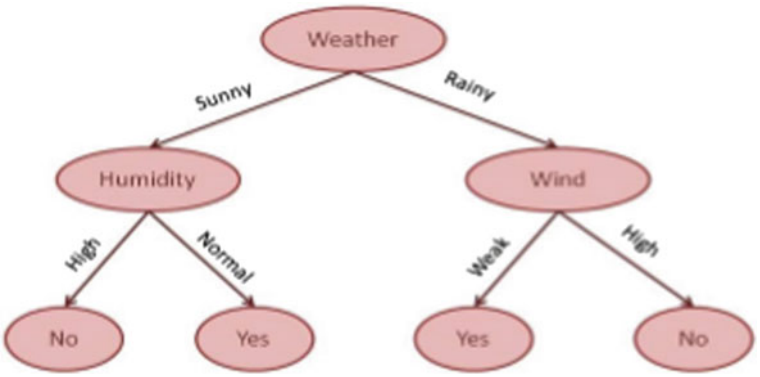


Fig. 3 Depicting weather forecasting using decision tree

(iii) Database Management Techniques: Such techniques are mainly used to develop characteristics of the existing data. Data mining uses techniques, such as iterative database scanning for frequent item sets, attribute focusing and attribute-oriented induction (AOI) [26, 27].
(iv) Data Visualization: An efficient way for end users to identify trends, patterns, correlations, outliers and easily understand the retrieved information from large data sets.

1.3 Overview of Data Mining

Data mining is not an age-old technology. It is relatively new technology, but it is already made in use by a number of industries [25, 26]. In an entire knowledge discovery from database (KDD) process, data mining can be depicted in Fig. 4 as [28].

KDD, an iterative process, initially begins by collecting raw data and finally convert it into a new representation of knowledge. Data mining is not a single-step process as it is very complex and involves a number of steps [26, 27].

Fig. 4 Process of KDD with data mining step

Table 1 Steps of data mining

(1)	Clean the data by removing noise or unwanted data
(2)	Combined data from multiple heterogeneous resources and kept at one place
(3)	Data get transformed (or consolidated) by using summary/aggregation operations
(4)	It is the data mining processing step, a critical practice to extract data patterns
(5)	The extracted patterns get evaluated on the basis of some interestingness measures which recognize the remarkable patterns that represent knowledge
(6)	Finally, present the mined knowledge to user in a visualized manner

1.3.1 Classification of Data Mining Process

The two types of data mining process are data preparation or data preprocessing and data mining.

(i) Data preprocessing includes data cleaning, data integration, data selection and data transformation.
(ii) Data mining includes the integration of data mining, pattern evaluation and knowledge representation [5, 23].

Entire process of data mining can be prescribed in the table as (Table 1).

1.3.2 Data Mining Architecture

The important components involved in building the architecture of a data mining system are data repositories, data warehouse server, data mining engine, pattern evaluation module, GUI and knowledge base. It can be depicted as in the figure [21] (Fig. 5):

(a) *Data Repositories*: It consists of database, data warehouse, WWW, text files and other documents. It requires data cleaning, integration and selection before transferring data to the database or data warehouse server.
(b) *Database or Data Warehouse Server*: It handles the relevant data retrieval based on the users request [2]
(c) *Data Mining Engine*: It is the heart or core component of any data mining system. It is responsible to perform the tasks such as association, classification, characterization, clustering, prediction and time-series analysis [15, 26]
(d) *Pattern Evaluation*: It is associated with the data mining engine to satisfy the users demand [27].
(e) *Graphical User Interface*: It acts as an interface between the user and the data mining system by displaying the result in user-friendly manner [13, 29].
(f) *Knowledge Base*: It guides the search or evaluates the interestingness of the result patterns. It gets interacted by the pattern evaluation module on regular basis so as to update itself accordingly [30].

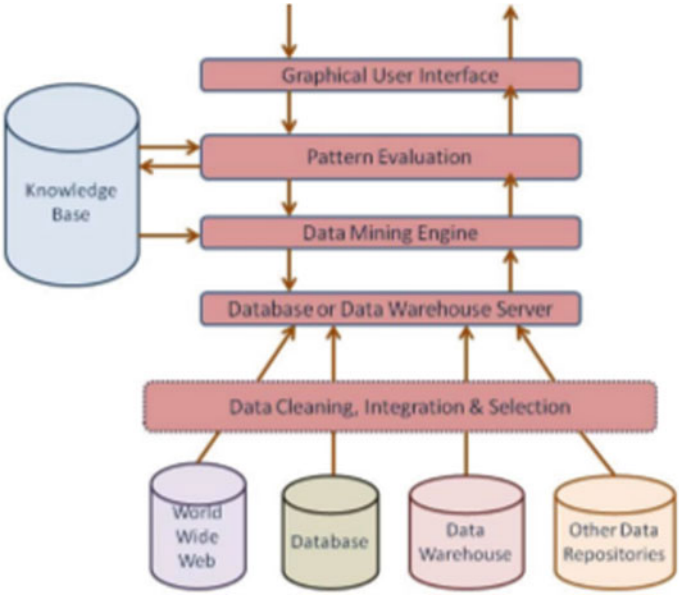


Fig. 5 Components of any data mining system

1.3.3 The Fundamentals of Data Mining

It became difficult for a person (an expert) to analyze huge data and find out valuable information out of it. This is made possible with the evolution of data mining and its three strong technologies. They are data mining algorithms, enormous data collection and computers with powerful multiprocessor [1]. Traditional data analysis techniques found challenges in handling large as well as new types of datasets. Thus, researchers focused on developing 16 proficient and more scalable tools to handle diverse data easily. Concepts given by data mining are shown in Fig. 6 as: [19].

Algorithms of data mining are comprised of techniques which existed few years back, i.e., at least 10 years [29]. Now, they are refined with matured, reliable and user-friendly tools in such a manner that they have consistently outperformed the previous

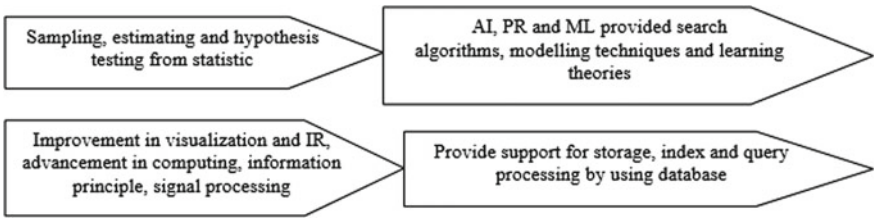


Fig. 6 Concepts of data mining related to other areas

Fig. 7 Data mining and its relationship with other areas

methods. Data mining produced information and knowledge that got used in several areas such as education, health care, finance, science, market analysis, intelligence agencies, internal revenue service, sports, Web education, credit scoring, engineering design and many more. The significant use of data mining in these special areas affects our life in one way or other. It is improved due to the rise in information technology [30]. Relationship of data mining with other areas can be depicted in Fig. 7.

1.4 Patterns to Accommodate Different Applications

In general, there are two types of data mining tasks [12, 22], given in Fig. 8 as:

- Prediction: To presume the forthcoming result on the basis of the experience of other existing values present in the database.
- Description: Emphasis on finding patterns that describe the data in very simple and human-interpretable form [31].

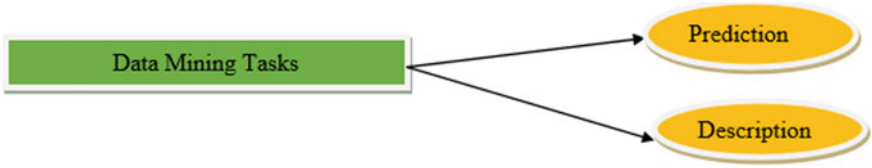


Fig. 8 Two types of task in data mining

The description is considered to be more important than prediction in the KDD process [15].

1.4.1 Data Mining Tasks

Following functionalities are kept under data mining tasks, such as classification, prediction, time-series analysis, association, clustering and summarization. All these functionalities can be further classified into predictive data mining tasks or descriptive data mining tasks. The following data mining tasks help prediction and description in achieving their objectives [11, 28] as shown in Fig. 9 as:

- Predictive tasks are helpful in predicting unknown or future values of another data set of interest.
- Descriptive tasks usually find data describing patterns and come up with new, significant information from the available data set [11, 16].

(a) ***Classification***: Categorize an object based on its attributes.
(b) ***Prediction***: Predicts future values of data on the basis of available data set.
(c) ***Time-Series Analysis***: Analyze time-series data in order to extract useful patterns, trends, rules and statistics.
(d) ***Association***: Helpful in identifying the relationships between objects.
(e) ***Clustering***: Identify and collect data objects based on the similarity of data set [27].
(f) ***Summarization***: The generalization and organization of relevant data in a smaller set that gives aggregated information of the data.

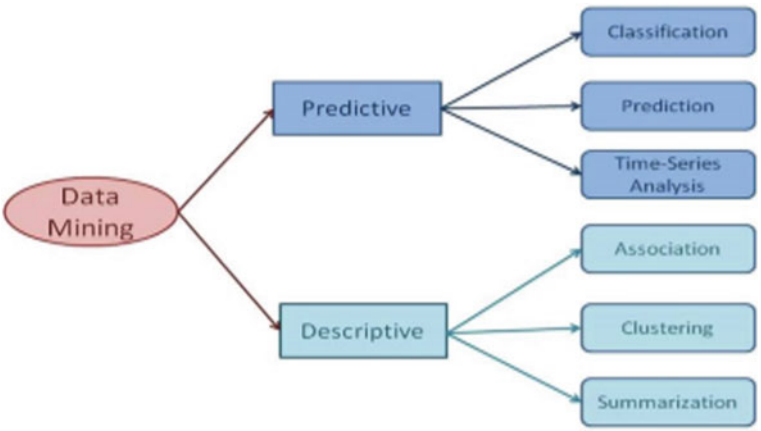


Fig. 9 Categorization of data mining tasks into prediction and description

1.4.2 Major Issues and Challenges in Data Mining [7, 28, 29]

Issues

- Considering diverse data to mine different kinds of knowledge.
- Lower performance.
- Inclusion of background knowledge
- Problem with data quality
- Absence of efficient parallel, distributed and incremental mining methods
- Constrained resource for data mining
- Gap between the discovered knowledge and existing one.

Challenges
Main reasons for challenges can be as follows [1, 9, 16]:

(a) Due to the fact that the data available is of heterogeneous types, incomplete and noisy.
(b) As the data is mainly available on different platforms in distributed computing environments, it becomes practically very difficult to bring all the data to a centralized data repository. Thus, the lack of tools and algorithms that enable mining of distributed data is a challenge.
(c) Data accumulated in today's world is heterogeneous. It can be in any form, such as images, audio and video, complex data, temporal data, spatial data, time series, natural language text and so on. It is really difficult to handle and extract required information from such dynamic and heterogeneous kinds of data.
(d) Lack of efficient algorithms and techniques adversely affects the performance of the data mining process.
(e) Collecting and incorporating background knowledge is a complex process.
(f) Inability to represent the information in an accurate and easy-to-understand way to the end user.
(g) Serious issues in terms of data security, privacy and governance.

1.5 Data Mining to Big Data Mining: Key Idea

The phenomenon big data has sharply accelerated the data mining. Many organizations have converted their paper-based systems to electronic systems. This transformation has brought several benefits to the organizations, such as time savings, a better management and making the tasks easier. Efficient and successful handling of the big data requires parallel processing.

Parallel processing aims to speedup the execution time of a program by sharing the work to be done among several processors [22]. In sequential processing, a single

processor is responsible to execute the entire data sets. It takes extremely long time to produce the result [17]. So, it is not suitable to process very large databases with a single processor. Therefore, the sequential program is required to get decompose into several threads. Threads are the independent units of computation and can be executed simultaneously on different processors to achieve a substantial reduction in execution time. As soon as threads are introduced during processing, several numbers of critical points are needed to be considered, such as the mechanisms to generate threads, synchronize the threads, performing communication of data between threads and after completion of the task the threads have to be terminated. It is a fact that these aspects of a parallel program are significantly more complex than those of a sequential program. But, when there is the task of processing big databases in a speedy and efficient manner, then the parallel program becomes very important [30, 31]. Thus, to bring data mining in practicality, it becomes crucial to introduce parallelism in its working.

2 Conclusion

Several data mining techniques are available to perform the processing of large databases. Data is not necessarily to be of same type. Appropriate techniques are available for information extraction from each type of data. When single technique is not efficient for such job, then the combination of two or more techniques is effective. Regardless of the many issues and challenges, data mining should still be considered as a valuable tool to many corporations. Data mining has laid down the foundation for other crucial techniques that can work on distributed and parallel environment in an efficient manner.

References

1. DeWitt DJ, Gray J (1992) Parallel database systems: the future of high performance database processing. Appear Commun ACM 36(6), June 1992
2. Fayyad U, Piatetsky-Shapiro G, Smyth P (1996) From data mining to knowledge discovery in databases. http://www.kdnuggets.com/Gpspubs/aimag-kdd-overview-1996-Fayyad.pdf. Retrieved 2008-12-17
3. Freitas AA, Lavington SH (1998) Mining very large databases with parallel processing. Kluwer Academic Publishers, Boston
4. Berson A, Smith SJ, Thearling K (1999) Building data mining applications for CRM. McGraw-Hill, New York
5. Owrang OMM (2000) Handling large databases in data mining. In: Proceedings of the information resources management association international conference on challenges of information technology management in the 21st century. Copyright 2000, Idea Group Inc
6. Han J, Kamber M (2001) Data mining: concepts and techniques. Morgan Kauffman, San Francisco, California
7. Hand DJ, Mannila H, Smyth P (2002) Principles of data mining. MIT Press, Cambridge Massachusetts

8. Dunham MH (2003) Data mining: introductory and advanced topics. Prentice Hall
9. Roiger RJ, Geatz MW (2003) Addison wesley data mining: a tutorial-based primer. Boston, Massachusetts
10. Xu R (2005) Survey of clustering algorithms. IEEE Trans Neural Netw 16(3), May 2005
11. Data mining tasks, techniques, and applications (2006) In: Introduction to data mining and its applications. Studies in computational intelligence, vol 29. Springer, Berlin, Heidelberg
12. Fox GC, Bae SH et al (2008) Parallel data mining from multicore to cloudy grids. High performance computing and grids workshop
13. Wang H, Wang S (2008) A knowledge management approach to data mining process for business intelligence. Ind Manag Data Syst 108(5):622–634
14. Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q, Motoda H, McLachlan GJ, Ng A, Liu B, Philip SY, Zhou ZH, Steinbach M, Hand DJ, Steinberg D (2008) Top 10 algorithms in data mining. Knowl Inf Syst 14: 1–37. https://doi.org/10.1007/s10115-0070114-2,at Springer
15. Pingshui W (2010) Survey on privacy preserving data mining. Int J Digit Content Technol Its Appl 4(9), December 2010
16. Deshpande SP, Thakare VM (2010) Data mining system and applications: a review. Int J Distrib Parallel Syst (IJDPS) 1(1), September 2010
17. Larose DT (2010) Discovering knowledge in data: an introduction to data mining. ISBN 0-471-66657-2, Wiley, Inc, 2005. Int J Distrib Parallel Syst (IJDPS) 1(1), September 2010
18. Dawei J (2011) The application of date mining in knowledge management. In: 2011 international conference on management of e-commerce and e-government, IEEE Computer Society, pp 7–9. https://doi.org/10.1109/icmecg.2011.58
19. Song S (2011) Analysis and acceleration of data mining algorithms on high performance reconfigurable computing platforms [Ph.D. thesis], Iowa State University
20. Silwattananusarn T, Tuamsuk K (2012) Data mining and its applications for knowledge management: a literature review from 2007 to 2012. Int J Data Min & Knowl Manag Process(IJDKP) 2(5), September 2012
21. https://www.wideskills.com/data-mining-tutorial/data-mining-architecture
22. Maitrey S, Jha CK (2012) A Survey: hierarchical clustering algorithm in data mining. IJESR 2(4), Article No 6/204-221. ISSN 2277-2685
23. Hilage TA, Kulkarni RV (2012) Review of literature on data mining. IJRRAS 10(1), January 2012
24. Maitrey S, Jha CK (2013) An integrated approach for CURE clustering using map-reduce technique. In: Proceedings of Elsevier, ISBN 978-81-910691-6-3, 2nd August 2013
25. Riondato M (2014) Sampling-based data mining algorithms: modern techniques and case studies. In: European conference on Machine learning and knowledge discovery in databases, ECML PKDD 2014, Nancy, France, September 15–19, 2014. Proceedings, Part III. Publisher Springer Berlin Heidelberg
26. Ghuman SS (2014) Int J Comput Sci Mob Comput 3(4), April 2014, pp 1401–1406 © 2014, IJCSMC All Rights Reserved 1401 Available Online at www.ijcsmc.com Int J Comput Sci Mob Comput. A Monthly Journal of Computer Science and Information Technology. ISSN 2320–088X IJCSMC, 3(4), April 2014, pp 1401–1406
27. Zheng Y (2015) Trajectory data mining: an overview. ACM Trans Intell Syst Technol 6(3), Article 29, Publication date: May 2015
28. Day MY Big data mining, http://mail.tku.edu.tw/myday/teaching/1062/BDM/1062BDM03_Big_Data_Mining.pdf, 2018-03-21
29. Mdaghri ZA (2016) Faculty of Science Rabat, Morocco, Rabat; Mourad El Yadari; Abdelillah Benyoussef; Abdellah El Kenz in Study and analysis of data mining for healthcare. Published In: 2016 4th IEEE international colloquium on information science and technology (CiSt)
30. Suresh R, Harshni SR (2017) Data mining and text mining—a survey. In: International conference on computation of power, energy information and communication (ICCPEIC)
31. Fernandes E, Holanda M, Victorino M, Borges V, Carvalho R, Van Erven G (2019) Educational data mining: predictive analysis of academic performance of public school students in the capital of Brazil. J Bus Res 94:335–343

Chapter 13
Improving the Training Pattern in Back-Propagation Neural Networks Using Holt-Winters' Seasonal Method and Gradient Boosting Model

S. Brilly Sangeetha, N. R. Wilfred Blessing, N. Yuvaraj, and J. Adeline Sneha

1 Introduction

From the past decades, machine learning tends to become a backbone in the field of information technology. The machine learning process considers a set of input and its desired output for updating the internal layers, which ensures that the predicted output is closer to the actual output. A machine learning model uses its internal layers to produce the most likely output based on its training states. Hence, it is referred as model fitting. The applications of machine learning model vary widely from several application areas from military to medicine, from advertising to product recommendation, etc. The machine learning models tend to grow likely as the application areas dealing with the number of data increase. Among several empirical models, artificial neural network (ANN) [1] is considered as the most advantageous machine learning model due to its accurate decisions making based on its learning state provided at the time of training. It offers wider capability of providing desired results from incomplete knowledge or missing information. There are several other factors that ensure ANN is a better model for prediction, which includes higher fault tolerance

S. Brilly Sangeetha (✉)
Department of Computer Science and Engineering, IES College of Engineering, Thrissur, India
e-mail: brillyvino82@gmail.com

N. R. Wilfred Blessing
Department of Information Technology, Salalah College of Technology, Salalah, Sultanate of Oman
e-mail: wilfred.b@sct.edu.om

N. Yuvaraj
Department of Computer Science and Engineering,
St. Peter's Institute of Higher Education and Research, Chennai, India
e-mail: yraj1989@gmail.com

J. Adeline Sneha
Satyabhama Institute of Science and Technology, Chennai, Tamil Nadu, India

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_13

even at the time of network nodes getting corrupted. The rate of corruption is slower than other machine learning models.

A multilayered ANN performs wider Boolean function than other computing units with single layer. The identification of correct weights in multilayered ANN substantially increases with complicated topologies and increasing parameters in the network. These multilayered ANN suffers mostly from the problem of identifying correct combination of node weights in relation with massive data flow. This prominently leads to the higher error ratio that produces poor prediction results. To reduce the percentage of errors during weights computation, ANN uses back propagation. The back propagation used in ANN estimates the gradient for finding the weights needed to be used in the network.

In BPNN, the computation of error takes place at the output node, and it is distributed backward along the layers. Back propagation in general is a supervised learning that tends to train the deep neural network using four-layered approach. It aims at reducing the errors, which can be fed at input to produce the desired prediction result, and the variations with actual output are very minimal. However, back-propagation neural networks (BPNNs) [2–4] suffer from the problem of network immobility. The network immobility occurs due to the adjustment of weights for larger input values at the time of training. These large values force the internal units to function with extreme values, due to the presence of derivative of very small activation function. Such immobility adjusts the network weights more rapidly that does not allow the network to acquire an optimal solution. This leads to higher computational burden in the network for finding the correct combination of weights when more parameters are considered. Further, as the number of hidden layers increases, the BPNN learning slows down exponentially. This is due to error signal attenuation as it propagates along the hidden layers. Thus, in multilayered BPNN, the optimization of prediction tends to reduce with increasing hidden layers and its weights.

In this paper, the errors occurring due to BPNN disabilities are removed and predictive performance is improved by increasing the weights and decaying the error signal using Holt-Winters' seasonal method and gradient boosting model. The design is made in such a way that it avoids longer convergence time on a plateau region.

The main contribution of the work involves the following:

1. BPNN is combined with Holt-Winters' seasonal method and gradient boosting model.
2. Holt-Winters' seasonal method forecasts the weights as inputs to train the hidden layers. It optimizes the weights required to train the hidden layers with a threshold limit that sets to reduce the use of hidden layers, and this tends to reduce the computational complexity.
3. Gradient boosting model acts as a learning rate parameter to compensate the signal attenuation errors in hidden layers using different learning rates. This parameter controls the network weight adjustment w.r.t the gradient loss.

Lesser the value is, slower is the movement along the downward slope. Hence, the gradient boosting model uses low learning rate and ensures that no local minima points are missed. This avoids longer time convergence, especially on a plateau

region. The tree boosting method provides highly effective solutions by combining several weak predictors (in the present study, we consider predictors that are error signal).

2 Related Works

Mason et al. [5] consider the three different prediction methods, namely multilayer ANN, logistic regression and probabilistic NN, for engineering students in university case studies. The study introduces PNN for student retention in engineering and comparing the results of PNN with other methods.

Sun and Wang [6] study a forecasting model for wind speed that includes an empirical decomposition ensemble, phase space reconstruction, sample entropy, and two hidden-layer neural re-propagation networks that improve the prediction of wind speed. First, the collected and sampled data is preprocessed by quick decomposition method, and then, entropy of samples is applied on empirical mode. The prediction model is improved using neural back-propagation network that was subsequently developed for forecasting the time series.

Ye and Kim [7] proposed neural back-spreading network Levenberg–Marquardt to improve the prediction accuracy that is combined with quasi-Newton and descent gradient methods to achieve quick convergence and better overall performance. In addition, the weight of the network can be adaptively adjusted to ensure that the system can converge efficiently.

3 Proposed Method

In this section, we study the disabilities in BPNN and the efforts to improve the predictive performance of BPNN. This chapter identifies the solution to these two problems: increasing weights and error signal decay using Holt-Winters' seasonal method and gradient boosting, respectively.

First, the Holt-Winters' seasonal method is a triple exponential smoothing model that forecasts the weights required for the hidden layers. This method is effective in dealing huge weights that tremendously required to train the hidden layers. The Holt-Winters' seasonal method optimizes the required weights and eliminates redundant weights required for training the inner layers. As the number of layers increases, the computational complexity increases; hence, a threshold (redundant) limit is set in BPNN regarding the usage of hidden layers. Instead, the predictions or forecasting is made by Holt-Winters' seasonal method that simplifies the predictions with reduced weights.

Second, to compensate the attenuation of error signal in deep layers, gradient boosting model is used. This method uses different learning rates for different hidden layers. The gradient boosting algorithm acts as a learning rate parameter, which is

regarded as a hyper parameter used for controlling the adjustment of network weights w.r.t gradient loss. Lesser the value is, slower is the movement along the downward slope. Hence, the gradient boosting model uses low learning rate and ensures that no local minima points are missed. This avoids longer time convergence, especially on a plateau region. The tree boosting method provides highly effective solutions by combining several weak predictors (in the present study, we consider predictors that are error signal).

Holt-Winters' Seasonal Method for Optimizing the Hidden Layers

Exponential smoothing is defined as a process that reviews the weights in continuous manner based on past observations. Exponential smoothing attributes reduce the weights in an exponential manner if the observation is from past iterations. Recent observations, in other words, are more important than the past iterations while updating the hidden layers.

Using BPNN input data, α, β, and μ, the model parameters are initialized. For choosing the best parameters, the MAPE is used as an error measure. These parameters are constantly updated in the adaptive Holt-Winters' method, taking into account the recent weights from the hidden layers. The motivation behind using the adaptive method is that the weights can change the behavior and model parameters required for training the BPNN adapt to the behavioral change against non-adaptive technique. This assumption has been confirmed by tests conducted on certain standard time-series data.

This method is used where the data indicate needed inputs and a threshold value for the hidden layers. Triple exponential smoothing a third parameter is added to handle this input to set the threshold value. We are now introducing a third seasonality equation. The resulting equations are called as Holt-Winters.

The multiplicative model generates the threshold value required for the hidden layers. We assume in this model that the weights are updated in following manner:

$$y_t = (b_1 + b_2 t) S_t + \varepsilon_t \tag{1}$$

where

b_1 is regarded as the permanent component
b_2 is regarded as the linear trend component
S_t is regarded as the multiplicative threshold factor
ε_t is regarded as the random error component.

Consider L as the weights required to train the hidden layers of BPNN over Tth iteration. The seasonal factors are the sum to the season length, i.e.,

$$\sum_{1 \le t \le L} S_t = L \tag{2}$$

The $b2$ component can be deleted from the model if considered unnecessary. In addition the hidden layers of BPNN with input and threshold values are updated using different processes, which are given below:

Overall Smoothing of Weights

$$G_t = \beta * (S_t - S_t - 1) + (1 - \beta) * G_{t-1} \tag{3}$$

where $0 < \beta < 1$ is considered as a smoothing constant factor.

The estimation of trend component is defined as the difference of smoothed value between the de-threshold-level estimates.

Updated Threshold Value

Update for the next iteration. The update over next iteration is given as follows:

$$y_t = (R_{t-1} + G_{t-1})S_{t-L} \tag{4}$$

It is noted that the optimal estimation of the threshold value at Tth iteration is last updated at time L.

Multiple-step-ahead forecasts (for $T < q$). The forecasted weight value over the iteration T is given as:

$$y_{t+T} = (R_{t-1} + T * G_{t-1})S_{t+T-L} \tag{5}$$

Thus, Holt-Winters' seasonal method forecasts the required weights in the form of inputs to train the hidden layers. It optimizes the weights required to train the hidden layers with a threshold limit that sets to reduce the use of hidden layers, and this tends to reduce the computational complexity.

Gradient Boosting Model

Gradient boosting model is regarded as a learning rate parameter that helps in compensating the errors occurring due to signal attenuation in hidden layers using different learning rates. This parameter controls the network weight adjustment w.r.t the gradient loss.

The present study describes XGBoost operation, where an efficient and highly scalable open-source implementation is easy to implement, and this works specifically on slick data [8].

The minimization of residual error from the previous model, XGBoost trains many sequential models. Each XGBoost model is a regression tree [9]. The formation of a decision tree means the identification, by classification, of a number of rules-based splits in the input. This is generalized by the CART algorithm by continuously taking valued weights on each decision tree leaf.

For a true response y_i and predictive model $y^{(1)} = f_1(x_i)$, a loss function is defined as $l(y^{(1)}, y_i)$ between the response obtained and the predicted instances.

The aim is then simply to minimize the loss over each hidden layer say i, together with some regularization function Ω, which biased toward simple models.

$$L = \sum_i l\left(y_i^{(1)}, y_i\right) + \Omega(f_1) \tag{6}$$

XGBoost is used to construct another tree, $f_2(x_i)$ for residual approximation after minimizing the value of L for a single tree. The minimization is regarded as the total loss L between the true response y_i and the sum of initial predictions of the training tree.

$$L = \sum_i l\left(y_i^{(1)} + f_2(x_i), y_i\right) + \Omega(f_2) \tag{7}$$

For predetermined trees, each input is trained for approximating the residual sum of previous trees, and this process is considered to be sequential.

XGBoost is designed with a tree in order to reduce the total loss in gradual terms. After the process of training at the end of all iteration, the output sum of all trees produces new predictions.

$$y = \sum_{k=1}^{N} f_k(x) \tag{8}$$

In reality, it is easier to select the functions f_k by increasing the gradient boosting and reducing the loss function by a second-order approximation [10].

To optimize performance and prevent overfit, XGBoost offers a number of additional parameters.

Many of these describe the criteria of training each tree. The parameters are train trees, maximum depth and decision of each tree, and allowed minimum weight on decision leaf, ratio of subsampling data, and minimum gain to acquire new branch decision.

4 Results and Discussions

The BPNN with gradient boosting and Holt-Winters' exponential smoothening (BPNN-GB-HWES) is tested against several other machine learning models that includes: BPNN, ANN, feedforward neural network (FNN), Kohonen self-organizing neural network (KSO-NN), radial basis function neural network (RBF-NN), convolutional neural network (CNN), recurrent neural network—long short-term memory (RNN-LSTM), and modular neural network (MNN).

The proposed method is evaluated on two datasets that include sunspot activities prediction [11] and breast cancer diagnosis [12].

Table 1 Performance evaluation on detecting the breast cancer over 450 runs

Algorithms	RMSE	Average initial error
BPNN-GB-HWES	0.092342	0.1932656
BPNN	0.095724	0.2013644
ANN	0.099885	0.2154658
FNN	0.125884	0.0232154
RBF-NN	0.154874	0.0254045
KSO-NN	0.198557	0.0257929
RNN-LSTM	0.215128	0.2918460
CNN	0.235485	0.2990850
MNN	0.264587	0.3254982
BPNN-GB-HWES	0.293765	0.3546844
BPNN	0.315478	0.4282790

The BPNN-GB-HWES with other methods are simulated in MATLAB platform against several performance metrics including mean square error (MSE), root MSE (RSME), and average initial error. Further, the performance evaluation is carried out using several real-time datasets that include the prediction of automobile sales data and the prediction of stock markets. The results are measured in terms of classification accuracy, specificity, sensitivity, and F-measure. The result shows that the proposed BPNN-GB-HWES outperformed other existing NN methods in terms of MSE, RSME, average initial error, accuracy, specificity, sensitivity, and F-measure. Finally, the proposed method is tested in terms of number of nodes or networks getting stuck by initial weights.

It is seen that at the end of 450 iterations, the RMSE of BPNN-GB-HWES is 0.092342 (Table 1), which is marginally lesser than other methods for breast cancer dataset. Similarly, for the sunspots activity detection, the RMSE is 0.090215 (Table 2), which is lesser than other methods. Hence, the study concludes that with increasing iterations, the reduction in RMSE may by marginally high.

In Table 3, we found that the accuracy of detecting the breast cancer using BPNN-GB-HWES is 0.950, which is higher than other methods. Likewise in Table 4, the accuracy of detecting the breast cancer using BPNN-GB-HWES is 0.925, which is higher than other methods.

The result shows that the proposed method has reduced number of nodes that do not stuck with the initial weights as compared with other approaches. The study further reduces RMSE rate by 0.09 in both the datasets, we found that after several iterations, the proposed method achieved the rationale of lesser RMSE. However, the number of iterations to achieve this RMSE rate was quite higher for other NN algorithms and certain algorithms failed before reaching that particular rate.

Table 2 Performance evaluation of dataset on detecting the sunspots problem over 450 runs

Algorithms	RMSE	Average initial error
BPNN-GB-HWES	0.090215	0.1932143
BPNN	0.094242	0.2087965
ANN	0.099548	0.2102152
FNN	0.124265	0.0238485
RBF-NN	0.152151	0.0253565
KSO-NN	0.195458	0.0259658
RNN-LSTM	0.213251	0.2918524
CNN	0.232955	0.2992212
MNN	0.262158	0.3259964
BPNN-GB-HWES	0.290889	0.3540235
BPNN	0.312548	0.4280218

Table 3 Performance evaluation of dataset on detecting the breast cancer over 450 runs

Algorithms	Accuracy	Specificity	Sensitivity
BPNN-GB-HWES	0.950	0.995	0.985
BPNN	0.925	0.972	0.972
ANN	0.916	0.933	0.952
FNN	0.875	0.912	0.895
RBF-NN	0.860	0.882	0.833
KSO-NN	0.835	0.853	0.821
RNN-LSTM	0.770	0.812	0.801
CNN	0.765	0.793	0.750
MNN	0.740	0.745	0.745
BPNN-GB-HWES	0.725	0.714	0.714
BPNN	0.710	0.695	0.778

Table 4 Performance evaluation of dataset on detecting the sunspots problem over 450 runs

Algorithms	Accuracy	Specificity	Sensitivity
BPNN-GB-HWES	0.952	0.971	0.962
BPNN	0.921	0.962	0.954
ANN	0.902	0.958	0.929
FNN	0.862	0.900	0.881
RBF-NN	0.851	0.874	0.849
KSO-NN	0.810	0.845	0.803
RNN-LSTM	0.762	0.839	0.785
CNN	0.758	0.802	0.769
MNN	0.733	0.795	0.721
BPNN-GB-HWES	0.712	0.751	0.705
BPNN	0.690	0.701	0.665

5 Conclusions

In this study, we improve BPNN to avoid two problems increasing weights and error signal decay using Holt-Winters' seasonal method and gradient boosting model, respectively. The Holt-Winters' seasonal method is a triple exponential smoothing model that forecasts the weights required for the hidden layers. In order to compensate the attenuation of error signal in deep layers, gradient boosting model is used. Tree boosting is used to provide highly effective solutions by combining several weak predictors. The simulation results show that the proposed method is higher in terms of accuracy and reduced error than other methods. For breast cancer dataset and sunspots activity detection, the RMSE of BPNN-GB-HWES is marginally lesser than other methods. Further, the accuracy of BPNN-GB-HWES is higher than other existing methods. The BPNN-GB-HWES has reduced the total nodes in the network, and it avoids being stuck by initial weights, which has caused reduced RMSE and increased accuracy. Hence, the study concludes that the tuning the weights as per the requirement and eliminating the decay of error signals helps to improve the prediction patterns of NN through back propagation. This is considered to be an improved method than other existing methods and can be applied on modern machine learning classification techniques. The study can further be enhanced by forecasting both time and frequency domain models to study the real-time estimation of heart rate of patients. Also, deep learning methods can be adopted to adaptively learn the network.

References

1. Hill T, Marquez L, O'Connor M, Remus W (1994) Artificial neural network models for forecasting and decision making. Int J Forecast 10(1):5–15
2. Wang L, Zeng Y, Chen T (2015) Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. Expert Syst Appl 42(2):855–863
3. Zeng YR, Zeng Y, Choi B, Wang L (2017) Multifactor-influenced energy consumption forecasting using enhanced back-propagation neural network. Energy 127:381–396
4. Bai Y, Li Y, Wang X, Xie J, Li C (2016) Air pollutants concentrations forecasting using back propagation neural network based on wavelet decomposition with meteorological conditions. Atmos Pollut Res 7(3):557–566
5. Mason C, Twomey J, Wright D, Whitman L (2018) Predicting engineering student attrition risk using a probabilistic neural network and comparing results with a back propagation neural network and logistic regression. Res Higher Educ 59(3):382–400
6. Sun W, Wang Y (2018) Short-term wind speed forecasting based on fast ensemble empirical mode decomposition, phase space reconstruction, sample entropy and improved back-propagation neural network. Energy Convers Manag 157:1–12
7. Ye Z, Kim MK (2018) Predicting electricity consumption in a building using an optimized back-propagation and Levenberg–Marquardt back-propagation neural network: Case study of a shopping mall in China. Sustain Cities Soc 42:176–183; Fröhlinghaus T, Weichert A, Rujan P (1994) Hierarchical neural networks for time-series analysis and control. Netw Comput Neural Syst *5*(1):101–116

8. Chen T, Guestrin C (2016) Xgboost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 785–794
9. Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Stat: 1189–1232
10. Friedman J, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Ann Stat 28(2):337–407
11. Murphy PM (1992) UCI Repository of machine learning databases [Machine-readable data repository]. In: Technical report. Department of Information and Computer Science, University of California
12. Zhao X, Han M, Ding L, Calin AC (2018) Forecasting carbon dioxide emissions based on a hybrid of mixed data sampling regression model and back propagation neural network in the USA. Environ Sci Pollut Res 25(3):2899–2910

Chapter 14
Ensemble of Multi-headed Machine Learning Architectures for Time-Series Forecasting of Healthcare Expenditures

Shruti Kaushik, Abhinav Choudhury, Nataraj Dasgupta, Sayee Natarajan, Larry A. Pickett, and Varun Dutt

1 Introduction

Predicting expenditure on medications is one of the crucial issues in the world as it may help patients to better manage their spending on healthcare [1]. Also, it may help pharmaceutical companies optimize their manufacturing process and determine attractive pricing for their medications [2]. Machine learning (ML) offers a wide range of techniques to predict future medicine expenditures using data of historical expenditures as well as other healthcare variables. For example, literature has developed autoregressive integrated moving average (ARIMA), multi-layer perceptron (MLP), long short-term memory (LSTM), and convolutional neural network (CNN) models to predict the future medicine expenditures and other healthcare outcomes [1, 3, 4, 5]. Researchers have also utilized traditional approaches like k-nearest neighbor and support vector machines' frameworks for time-series predictions [6].

S. Kaushik (✉) · A. Choudhury · V. Dutt
Applied Cognitive Science Laboratory, Indian Institute of Technology Mandi, Mandi, Himachal Pradesh 175005, India
e-mail: shruti_kaushik@students.iitmandi.ac.in

A. Choudhury
e-mail: abhinav_choudhury@students.iitmandi.ac.in

V. Dutt
e-mail: varun@iitmandi.ac.in

N. Dasgupta · S. Natarajan · L. A. Pickett
RxDataScience, Inc., Durham, NC 27709, USA
e-mail: nd@rxdatascience.com

S. Natarajan
e-mail: sayee@rxdatascience.com

L. A. Pickett
e-mail: larry@rxdatascience.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_14

However, the non-stationary and non-linear dynamics of the time-series poses major challenges in predicting the underlying time-series accurately [7].

Literature has shown the advantages of using LSTM and CNN models for performing time-series predictions when the underlying time-series depicts non-linear and/or non-stationary behavior [7]. However, prior research has not utilized the benefits of convolutional long short-term memory (ConvLSTM) models and convolutional neural network models combined with long short-term memory (CNN-LSTM) models for predicting the non-linear and non-stationary time-series in healthcare domains.

Second, an investigation involving LSTM, ConvLSTM, and CNN-LSTM multi-headed neural network architectures has not been explored for predicting the time-series in healthcare domains. In these multi-headed architectures, each independent variable (input series) is handled by a separate neural network model (head), and the output of each of these models (heads) is combined before a prediction is made about a dependent variable [8].

Third, prior research in the field of time-series forecasting suggests that significant improvement in performance can be attained by merging predictions from various models [9, 10]. Prior research has used the weighted approaches to train the ensemble models [10, 11]. However, combination of multi-headed neural network architectures via ensemble models has been less explored in the literature for predicting healthcare outcomes.

Fourth, prior research has shown the advantages of shuffling the supervised mini-batches while training the univariate time-series LSTM models [10]. Shuffling is helpful because it avoids the model getting trapped in local minima during training due to the repeated presentation of data in the same order. In addition, prior research has shown a regularization technique that called dropout to be helpful in reducing the overfitting in neural network architectures [12]. However, the effect of shuffling and dropout has not been evaluated so far on multi-headed architectures for time-series prediction problems.

Overall, building upon these gaps in the literature, the primary objective of this research is to evaluate multi-headed architectures involving LSTM, ConvLSTM, and CNN-LSTM models as well as evaluate an ensemble of these multi-headed models. The ensemble model combines the predictions of the LSTM, ConvLSTM, and CNN-LSTM models to predict patients' expenditures on certain pain medications.[1] Moreover, we also compare four different variations across different multi-headed neural architectures: shuffle with dropout, shuffle without dropout, no-shuffle with dropout, and no-shuffle without dropout. When shuffling is present (shuffle), smaller supervised sets (mini-batches) containing attributes corresponding to the chosen look-back (lag) period are created and shuffled across the time-series during network training. However, when shuffle is not present (no-shuffle), the mini-batches are created in the order they occur in data and inputted into the network without shuffling. Also, when dropout is present, certain proportion of nodes in the network is randomly discarded during model training. When dropout is absent, no nodes are discarded during training in the network.

[1] Pain medications were chosen as they cut across a number of patient-related ailments.

In what follows, we first provide a brief review of the related literature. Next, we explain the methodology of applying different multi-headed architectures consisting of LSTM, CNN-LSTM, ConvLSTM, and their ensemble for multivariate time-series prediction of healthcare outcomes. In Sect. 4, we present our experimental results, where we compare different models' predictions. Finally, we conclude our paper and provide a discussion on the implication of this research and its future scope.

2 Background

In recent years, ML algorithms (e.g., LSTM, CNN) have gained lot of attention in almost every domain [4, 7, 13–15]. The ML neural networks can automatically learn the complex and arbitrary mappings from inputs to outputs [7]. Neural network models like LSTMs can handle the ordering of observations (important for time-series problems), which is not offered by multi-layer perceptron or CNN models [4].

Recently, LSTMs and their variants ConvLSTM and CNN-LSTM have been used by researchers to solve different problems across a number of domains [13–16]. For example, Zhao et al. have used LSTMs for traffic forecasting [13]. These authors used the temporal–spatial information and proposed an LSTM network for short-term traffic forecast. Xingjian et al. have used convolutional LSTM (ConvLSTM) to predict the rainfall intensity for a short period of time [14]. They showed that ConvLSTM performed better than a fully connected LSTM to forecast rainfall intensity.

Researchers have also developed cascades of CNN and LSTM models to learn the upward and downward trend in a time-series [15]. For example, Ref. [15] showed that the CNN-LSTM (a cascade model) outperformed the stand-alone CNN and LSTM models to learn the trend in a time-series.

Moreover, there have been multi-headed neural network models proposed in the recent literature, where a head (a neural network) is used for each independent variable and outputs of each head are combined to give the final prediction for the dependent variable [8, 17]. Researchers have developed multi-headed recurrent neural networks in certain identification tasks [8] and clustering tasks [17]. However, to the best of our knowledge, multi-headed architectures of LSTM, ConvLSTM, and CNN-LSTM have not been evaluated yet for the multivariate time-series forecasting in the healthcare domain. Moreover, shuffling the mini-batches while training the neural networks and adding regularization to reduce overfitting has proven to be effective training mechanisms in the literature [10, 12].

Additionally, researchers have used ensemble techniques to further improve the performance of the individual models [9, 11, 18]. Prior research has shown that the ensemble approach performs better than the individual ML models [9]. Thus, we also create an ensemble model using the predictions of these three multi-head models and compare its performance with the individual models for multivariate time-series prediction of patients' expenditures. Moreover, we expect the ensemble model to perform better than the individual models. One likely reason for this expectation

is that the ensemble method is likely to give more weight to those model predictions that are accurate compared to those that are less accurate. Also, prior research shows that the ConvLSTM and CNN-LSTM architectures perform better compared to individual LSTM or CNN architectures on image datasets, where both spatial and temporal information contribute toward predictions [14]. In this research, we deal with only temporal information in a multi-headed architecture. Thus, by projecting from individual architectures to multi-headed architectures in the absence of spatial information, we expect the multi-headed LSTM model to perform better compared to the ConvLSTM and CNN-LSTM models. Also, we expect that shuffling between the supervised mini-batches or dropout mechanisms to likely help models to improve their prediction on test data.

Overall, the main contribution of our research is to evaluate the performance of three multi-headed architectures, i.e., LSTM, ConvLSTM, and CNN-LSTM to predict the weekly average expenditure on certain medications. The second contribution is to compare the individual multi-headed models with their ensemble model. The third contribution is to evaluate the performance of shuffle and dropout variations while training the multi-headed neural architectures.

3 Method

3.1 Data

In this paper, we selected two pain medications (named "A" and "B") from the Truven MarketScan dataset for our analyses [19].[2] These two pain medications were among the top-ten most prescribed pain medications in the USA [20]. Data for both medications range between January 2, 2011, and April 15, 2015 (1565 days). For our analyses, across both pain medications, we used the dataset between January 2, 2011, and July 30, 2014 (1306 days), for model training and the dataset between July 31, 2014, and April 15, 2015 (259 days), for model testing. On average, each day, about 1428 patients refilled medicine A, and about 550 patients refilled medicine B. For both medicines, we prepared a multivariate time-series containing the daily average expenditures by patients on these medications, respectively. The time-series was stationary for medicine A; however, it was not stationary for medicine B (one time-differencing was performed on medicine B's time-series, to make it stationary). We used 20 attributes across both medicines for performing time-series analyses. These attributes provided information regarding the number of patients of a particular gender (male, female), age group (0–17, 18–34, 35–44, 45–54, and 55–65), region (south, northeast, north central, west, and unknown), health-plan (two type of health plans), and different diagnoses and procedure codes (six ICD-9 codes) who consumed medicine on a particular day. These six ICD-9 codes were selected from the frequent

[2]To maintain privacy, the actual names of the two pain medications have not been disclosed.

pattern mining using Apriori algorithm [21]. The 21st attribute was the average expenditure per patient for a medicine on a day t which was defined as per the following equation:

$$\text{Daily Average Expenditure}_t = i_t \Big/ j_t \tag{1}$$

where i was the total amount spent in a day t on the medicine across all patients and j was the total number of patients who refilled the medicine in day t. This daily average expenditure on a medicine along with the 20 other attributes was used to compute the weekly average expenditure, where the weekly average expenditure was used to evaluate model performance.

3.2 Evaluation Metrics

All the models were fit to data at a weekly level using the following metrics: root mean squared error (RMSE; error) and R-square (R^2; trend) [22]. As weekly average expenditure predictions were of interest, the RMSE and R^2 scores and visualizations for weekly average expenditures were computed in weekly blocks of seven days. Thus, the daily average expenditures per patient were summed across seven days in a block for whole dataset. This resulted in the weekly average expenditure across 186 blocks of training data and 37 blocks of test data. We calibrated all models to reduce error and capture trend in data. Thus, all models were calibrated using an objective function that was defined as the following[3]: [RMSE/10 + (1 − R^2)]. This objective function ensured that the obtained parameters minimized the error (RMSE) and maximized the trend (R^2) on the weekly average expenditure per patient between model and actual data. The R^2 (between 0 and 1) accounts for whether the model's predictions follow the same trend as that present in the actual data. The larger the R^2 (closer to 1), the larger the ability of the model to predict the trend in actual data.

3.3 Experiment Design for Multi-headed LSTM

Figure 1a shows the multi-headed LTSM architecture used in this paper. The first layer across all heads is the input layer where mini-batches of each feature in data are put into a separate head. As shown in Fig. 1a, for training the multi-headed LSTM on a medicine, each variable (20 independent variables and one dependent variable) for the medicine was put into a separate LSTM model (head) to produce a single combined concatenated output. The dense (output) layer contained 1 neuron which gave the expenditure prediction about the medicine for a timeperiod. We used

[3]RMSE was divided by 10 in order to bring both RMSE and 1 − R^2 on the same scale. RMSE captures the error, and R^2 captures the trend.

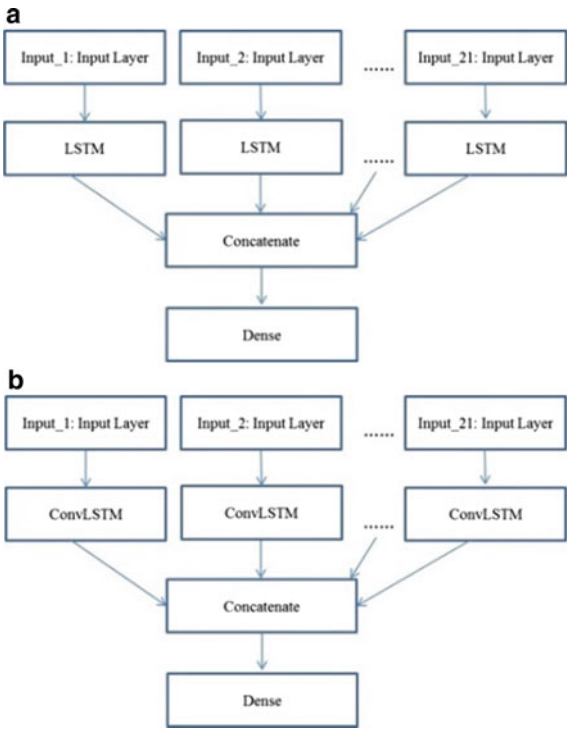


Fig. 1 **a** Multi-headed LSTM and **b** multi-headed ConvLSTM

a grid search procedure to train different hyper-parameters in the LSTM block in each head. The hyper-parameters used and their range of variation in the grid search were the following: hidden layers (1, 2, 3, and 4), number of neurons in a layer (4, 8, 16, 32, and 64), batch size (5, 10, 15, and 20), number of epochs (8, 16, 32, 64, 128, 256, and 512), lag/look-back period (2–8), activation function (tanh, adam, and adagrad), and dropout rate (20–60%).[4] Additionally, the training was done in four ways: shuffle with dropout, shuffle without dropout, no-shuffle with dropout, and no-shuffle without dropout. When shuffling was present, the mini-batches were shuffled randomly across time-series for each medicine. When shuffling was absent, we did not shuffle the mini-batches and presented these batches in sequential order to the neural network. For dropout present conditions, we put a dropout layer after each hidden layer in the LSTM block. For dropout absent conditions, we did not apply any dropout layer.

[4] A 20% dropout rate means that 20% connections will be dropped randomly from this layer to the next layer.

3.4 Experiment Design for Multi-headed ConvLSTM

The multi-headed ConvLSTM architecture was also trained exactly in a same manner as multi-headed LSTM. Figure 1b shows the example of a multi-headed ConvLSTM architecture in which the first layer across all heads is the input layer where mini-batches of each feature in data are put into a separate head. Here also, each feature variable was put into a separate model (head) to produce a single combined concatenated output. The ConvLSTM is different from the LSTM in a manner that in ConvLSTM layers, the internal matrix multiplications (present in LSTM) are replaced with convolution operations [14]. Convolution is a mathematical operation which is performed on the input data with the use of a filter (a matrix) to produce a feature map [14]. The ConvLSTM block in each head has contained first the ConvLSTM layer where we passed (32, 64, or 128) filters with different kernel sizes (1, 3, 5, and 7). The output of this ConvLSTM layer was passed to different fully connected or dropout layers. At last, the output from each head after training was then concatenated to predict the expenditure (21st feature) on a medicine on a day. The dense (output) layer contained 1 neuron which gave the expenditure prediction about the medicine for a timeperiod.

3.5 Experiment Design for Multi-headed CNN-LSTM

The multi-headed CNN-LSTM architecture was also trained in a same way as other two multi-headed architectures. The example of multi-headed CNN-LSTM architecture is shown in Fig. 2. As shown in Fig. 2, the CNN-LSTM architecture contained

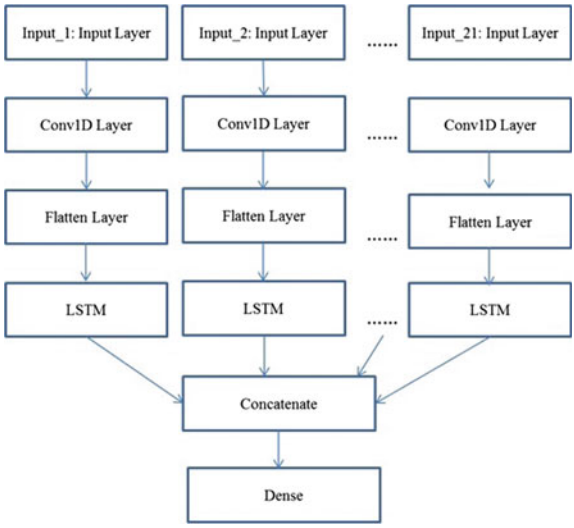


Fig. 2 Multi-headed CNN-LSTM architecture

both a CNN model as well as an LSTM model in each head. The CNN block is used for features extraction, and it is followed by the LSTM block for the sequence prediction of data [15]. There was a flatten layer just after the CNN block to flatten the 3D output from convolution layer in a 1D vector (input to LSTM). The outputs from each LSTM head were concatenated and passed through a dense layer, which then produce the daily expenditures for a medicine. Different number of filter (32, 64, or 128) and different kernel sizes (1, 3, 5, and 7) were varied as hyper-parameters in the CNN block. The hyper-parameters in the LSTM block were varied similar to those in the multi-headed LSTM model.

3.6 Ensemble Model

We used the normalized exponential weighted (NEW) algorithm [9] to ensemble the predictions of the multi-headed LSTM, ConvLSTM, and CNN-LSTM architectures. The working of the NEW algorithm is presented in the box below. Given a set of predictions on the training data from different models, the NEW algorithm starts with equal weights to all predictions (i.e., half weight to the multi-headed LSTM, ConvLSTM, and CNN-LSTM model predictions) and computes the ensemble prediction. Then, for the first training sample (i.e., the first point out of 1306 points), the squared error between the ensemble model's prediction and actual data is computed (line 4). Next, the weights are updated using the squared error for different model predictions (line 5), where the parameter η is the learning-rate parameter. Finally, we normalize the weights obtained for each model's predictions by dividing it with the total sum of the weights across all models (line 6). This process continues until all the 1306 training samples are covered. We calibrated the value of the η parameter by varying its values from 0.01 to 1.0 in steps of 0.01.

Normalized Exponential Weighted Algorithm

1. Input: *N models each predicting outcomes* f_i^t *for round t, and a free parameter* η
2. $w_i^1 \leftarrow 1/N$ *for* $i = 1, \ldots, N$ (Set initial weights of each model *i*'s predictions to 1/N)
3. *for training samples* $t = 1, 2, \ldots$ *do*
4. $l\left(f_i^t, y_t\right) = (f_i^t - y_t)^2$ (Calculation of squared error where y_t is the actual value)
5. $w_i^{t+1} \leftarrow w_i^t e^{\eta l(f_i^t, y_t)}$
6. $w_i^{t+1} \leftarrow w_i^{t+1} / \sum_{i=1}^{N} w_i^{t+1}$ (normalization of weights)

4 Results

4.1 Multi-headed LSTM Model

Table 1 shows the multi-headed LSTM model's RMSE and R^2 on training and test data for different shuffle and dropout combinations on medicines A and B (shuffle with dropout, shuffle without dropout, no-shuffle with dropout, and no-shuffle without dropout). As given in Table 1, the best RMSE on test data (=USD 318.82 per patient) was obtained for the no-shuffle and dropout combination for medicine A, and this model was trained with two lag periods, 64 epochs, 20 batch size, and tanh as the activation function. All the heads contained two LSTM layers, two dropout layers, and one output layer. The architecture description is as follows: first LSTM layer with 64 neurons, dropout layer with 30% dropout rate, second LSTM layer with 64 neurons, another dropout layer with 30% dropout rate, and finally the output layer with 1 neuron. The output from each head was then merged using a dense layer with 64 neurons, a dropout layer with 30% dropout rate, another dense layer with 64 neurons, dropout layer with 30% dropout rate, and finally one output layer with 1 neuron (concatenated output). On medicine B, we obtained the best RMSE on test data (=USD 40.31 per patient) for no-shuffle and with dropout combination. The model was trained with two lag periods, 64 epochs, 15 batch size, and relu activation function. First, we trained all the 21 heads, and then, the output from each head was concatenated using five dense layers, five dropout layers, and finally one output layer at the end. The 20 heads contained one LSTM layer with 64 neurons, one dropout layer with 30% dropout rate, and a second LSTM layer with 64 neurons.

Table 1 Multi-headed LSTM results during training and test

Medicine name	Combinations of shuffle and dropout	Train RMSE	Train R^2	Test RMSE	Test R^2
A	Shuffle with dropout	230.23	0.46	365.49	0.01
A	Shuffle without dropout	233.97	0.45	396.34	0.01
A	**No-shuffle with dropout**	237.81	0.45	318.82	0.05
A	No-shuffle without dropout	234.29	0.43	340.42	0.01
B	Shuffle with dropout	43.35	0.98	40.32	0.91
B	Shuffle without dropout	43.36	0.98	40.33	0.91
B	**No-shuffle with dropout**	42.92	0.98	40.31	0.93
B	No-shuffle without dropout	43.41	0.98	40.36	0.92

Note The bold text highlights the variation with the lowest RMSE on test data

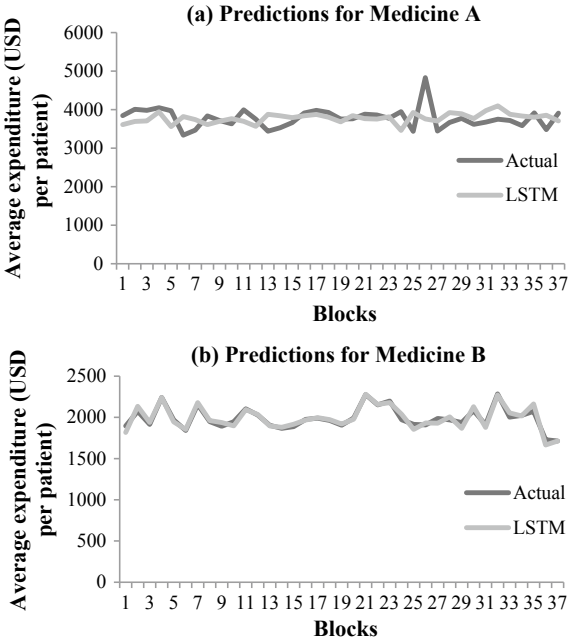


Fig. 3 Average expenditure (in USD per patient) from the multi-headed LSTM model for medicine A (**a**) and for medicine B (**b**) in test data

The 21st head contained four LSTM layers with 64 neurons and three dropout layers having 50% dropout rate between each LSTM layer. The output from each head was concatenated using first dense layer containing 128 neurons, followed by one dropout layer with 60% dropout rate, second dense layer with 64 neurons, second dropout layer with 40% dropout rate, third dense layer with 32 neurons, third dropout layer with 60% dropout rate, fourth dense layer with 16 neurons, fourth dropout layer with 60% dropout rate, fifth dense layer with 8 neurons, fifth dropout layer with 60% dropout rate, and finally the output layer with 1 neuron. Figure 3 shows that the LSTM model fits for medicine A (Fig. 3a) and medicine B (Fig. 3b) in test data for the best-performing model combinations. As shown in Fig. 3, the LSTM model fits were reasonably accurate for medicine B compared to medicine A.

4.2 Multi-headed ConvLSTM Model

Table 2 shows the multi-headed ConvLSTM model's RMSE and R^2 on training and test data for different shuffle and dropout combinations on medicines A and B (shuffle with dropout, shuffle without dropout, no-shuffle with dropout, and no-shuffle without dropout). As given in Table 2, the best RMSE on test data (=USD 326.91 per patient) was obtained for the no-shuffle and with dropout combination

Table 2 Multi-headed ConvLSTM results during training and test

Medicine name	Combinations of shuffle and dropout	Train RMSE	Train R^2	Test RMSE	Test R^2
A	Shuffle with dropout	228.29	0.47	331.37	0.02
A	Shuffle without dropout	230.55	0.46	361.57	0.03
A	**No-shuffle with dropout**	220.69	0.51	326.91	0.02
A	No-shuffle without dropout	221.32	0.50	334.19	0.02
B	Shuffle with dropout	56.84	0.97	64.49	0.77
B	Shuffle without dropout	63.05	0.96	86.81	0.62
B	**No-shuffle with dropout**	56.37	0.97	62.40	0.79
B	No-shuffle without dropout	57.54	0.96	81.90	0.65

Note The bold text highlights the variation with the lowest RMSE on test data

for medicine A, and this model was trained with two lag periods, 128 epochs, 20 batch size, and tanh activation layer. All the 21 heads contained one ConvLSTM layer with 128 filters, kernel (1, 5), followed by one dropout layer with 30% dropout rate, and the flatten layer at the end. The output from each head was merged using three dense layers, two dropout layers, and one output layer at the end. The output from each head was combined using one dense layer with 128 neurons, followed by one dropout layer with 30% dropout rate, second dense layer with 64 neurons, second dropout layer with 30% dropout rate, third dense layer with 32 neurons, and finally one output layer with 1 neuron. On medicine B, we obtained the best RMSE on test data (=USD 62.40 per patient) for no-shuffle and with dropout combination. The corresponding model was trained with two lag periods, 64 epochs, 15 batch size, and tanh activation function. All the heads possessed one ConvLSTM layer with 32 filters, kernel (1, 5), followed by a dropout layer with 30% dropout rate, and a flatten layer at the end. The output from each head was merged using four dense layers, four dropout layers, and one output layer at the end. The output from each head was combined using the following architecture: first dense layer with 64 neurons, first dropout layer with 50% dropout rate, second dense layer with 32 neurons, second dropout layer with 60% dropout rate, third dense layer with 16 neurons, third dropout layer with 60% dropout rate, fourth dense layer with 8 neurons, fourth dropout layer with 60% dropout rate, and finally the output layer with 1 neuron. Figure 4 shows the ConvLSTM model fits for medicine A (Fig. 4a) and medicine B (Fig. 4b) in test data for the best-performing model combinations.

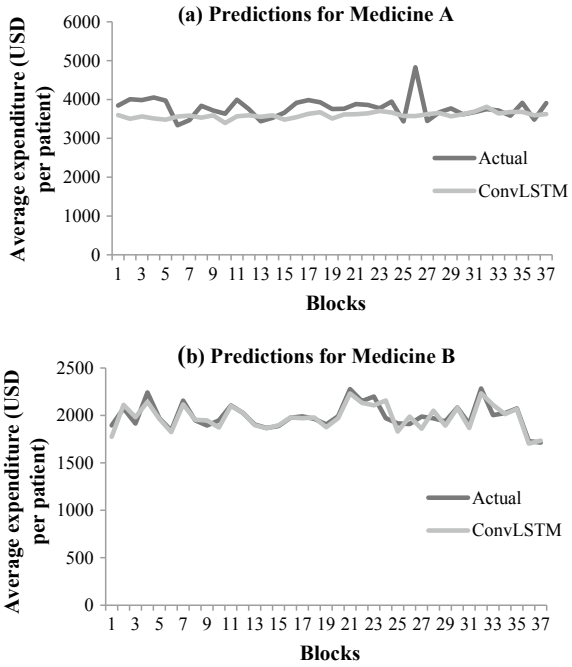


Fig. 4 Average expenditure (in USD per patient) from the multi-headed ConvLSTM model for medicine A (**a**) and for medicine B (**b**) in test data

4.3 Multi-headed CNN-LSTM Model

Table 3 gives the multi-headed CNN-LSTM model's RMSE and R^2 on training and test data for different shuffle and dropout combinations on medicines A and B (shuffle with dropout, shuffle without dropout, no-shuffle with dropout, and no-shuffle without dropout). As given in Table 3, the best RMSE on test data (=USD 312.24 per patient) was obtained for the no-shuffle and with dropout combination for medicine A. All the heads in this model were comprised of 1D convolution layer with 64 filters having 3 kernel size, followed by a flatten layer, one dropout layer with 20% dropout rate, and the LSTM layer with 16 neurons. These heads were trained with two lag periods, 64 epochs, 15 batch size, and relu activation function. The output from all the 21 heads was merged using two dense layers, one dropout layer, and one output layer at the end. The output from different heads was combined using the following architecture: first dense layer having 64 neurons and tanh activation function, first dropout layer with 20% dropout rate, second dense layer with 32 neurons and tanh activation, and finally the output layer with 1 neuron. On medicine B, we obtained the best RMSE on test data (=USD 45.09 per patient) for no-shuffle and with dropout combination. All the 21 heads in this model were comprised of 1D convolution layer with 32 filters having 3 kernel size, followed by

Table 3 Multi-headed CNN-LSTM results during training and test

Medicine name	Combinations of shuffle and dropout	Train RMSE	Train R^2	Test RMSE	Test R^2
A	Shuffle with dropout	214.67	0.53	349.16	0.03
A	Shuffle without dropout	200.97	0.60	314.68	0.01
A	**No-shuffle with dropout**	218.28	0.52	312.24	0.03
A	No-shuffle without dropout	215.99	0.53	315.13	0.01
B	Shuffle with dropout	57.20	0.98	54.07	0.89
B	Shuffle without dropout	49.39	0.97	47.71	0.89
B	**No-shuffle with dropout**	47.87	0.98	45.09	0.89
B	No-shuffle without dropout	50.32	0.97	48.84	0.87

Note The bold text highlights the variation with the lowest RMSE on test data

a flatten layer, one dropout layer with 30% dropout rate, and a LSTM layer with 8 neurons. These heads were trained with two lag periods, 64 epochs, 20 batch size, and tanh activation function. The output from all the 21 heads was merged using two dense layers, two dropout layers, and one output layer at the end. The output from different heads was combined using the following architecture: first dense layer having 32 neurons and tanh activation function, first dropout layer with 30% dropout rate, second dense layer with 32 neurons and tanh activation, second dropout layer with 30% dropout rate, finally the output layer with 1 neuron. Figure 5 shows the CNN-LSTM model fits for medicine A (Fig. 5a) and medicine B (Fig. 5b) in test data for the best-performing model combinations. As shown in Fig. 5, the CNN-LSTM model fits were reasonably accurate for medicine B compared to medicine A.

4.4 Ensemble Model

Table 4 shows the ensemble model's RMSE and R^2 on training and test data on both the medicines. These results were obtained using the best predictions (the results which are shown in bold in Tables 1, 2, and 3) from each of the LSTM, ConvLSTM, and CNN-LSTM models. The ensemble results on medicine A were obtained with the following weights: 0.15 for LSTM, 0.00 for ConvLSTM, and 0.85 for CNN-LSTM, with $\eta = 0.01$ as the learning-rate parameter from the normalized exponential weighted algorithm. On medicine B, the ensemble results were obtained with the following weights: 0.80 for LSTM, 0.00 for ConvLSTM, and 0.2 for CNN-LSTM. The ensemble model weights for medicine B were obtained for $\eta = 0.05$ using

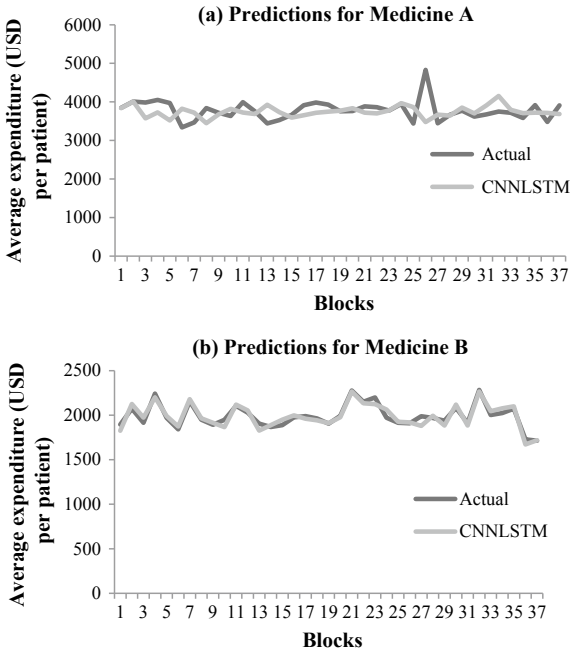


Fig. 5 Average expenditure (in USD per patient) from the multi-headed CNN-LSTM model for medicine A (**a**) and for medicine B, (**b**) in test data

Table 4 Ensemble model results during training and test

Ensemble model for medicines	Train RMSE	Train R^2	Test RMSE	Test R^2
A	220.06	0.41	311.05	0.05
B	43.64	0.98	39.93	0.92

the NEW algorithm. Figure 6 shows the model fits from the ensemble model for medicine A (Fig. 6a) and medicine B (Fig. 6b). As given in Table 4, the test RMSE has improved from stand-alone models for both the medicines.

5 Discussion and Conclusions

Time-series architectures have gained popularity among researchers across various disciplines [4, 13–15]. Specifically, the recurrent architectures such as long short-term memory (LSTM) have been utilized for time-series predictions [4, 10]. However, the variants of LSTM such as ConvLSTM and CNN-LSTM have not been evaluated for time-series predictions in the healthcare domain. Additionally, researchers have utilized single-headed neural network architectures to predict the future time-series in

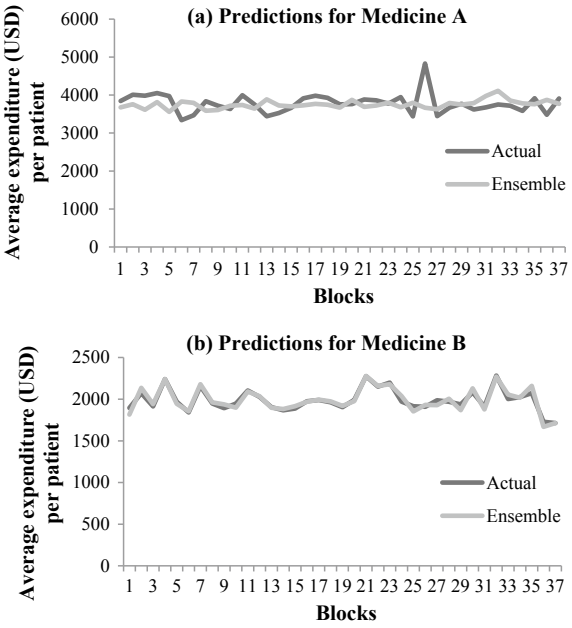


Fig. 6 Average expenditure (in USD per patient) from the ensemble model for medicine A (**a**) and for medicine B, (**b**) in test data

the healthcare domain [10]. However, the potential of multi-headed neural network architectures had not been utilized for multivariate time-series predictions in the healthcare domain. In multi-headed architectures, each head may take one predictor as input, and finally, the output from each head (model) may be merged to provide a single output for the predicted variable. In addition, outside of the healthcare domain, prior research has shown that ensemble of different architectures can improve the overall performance [11]. The main contribution of our research was to evaluate the performance of multi-headed LSTM, ConvLSTM, CNN-LSTM, and ensemble of all these three multi-headed models to predict the weekly average expenditure by patients on two pain medications. Another objective of this paper was to evaluate the advantages of shuffling training data and adding of dropouts while training the neural network models.

First, as per our expectation, we found that the best value of test RMSE and test R^2 was obtained from the ensemble model for both the medications. Ensemble model is likely to give more weight to the model that gives accurate predictions. For example, in this research, ensemble model gave a zero weight to the ConvLSTM model, which was the worst-performing model across both medicines. Overall, we obtained the best performance on test data using the ensemble model by combining the predictions from the multi-headed LSTM and CNN-LSTM models.

Second, we found that for medicine A, the multi-headed CNN-LSTM model performed better compared to the multi-headed ConvLSTM and LSTM models.

However, for medicine B, the multi-headed LSTM model performed better compared to the multi-headed ConvLSTM and CNN-LSTM models. A likely reason for this result could be that the extracted features from the CNN layers across different heads were beneficial for medicine A compared to medicine B.

Third, we found that the performance of the multi-headed ConvLSTM model was poor in terms of test RMSE and test R^2 across both medicines. A likely reason for the poorer performance of the multi-headed ConvLSTM model is because the ConvLSTM models are known for learning the spatial feature representations in image datasets and the datasets used in this manuscript mainly dealt with temporal features [14]. Thus, the ConvLSTM model did not perform as well as the other multi-headed models.

Fourth, we found that all models performed better on test data and reduced overfitting in dropout present conditions across both medicines. A likely reason is that dropouts caused our neural network architectures to reduce over-fitting in the dataset. However, the reduction in overfitting was small across all models as the multi-headed architectures are more complex compared to single-headed counterparts.

Fifth, we found that the multi-headed neural network models gave better performance when training time-series data was not shuffled. This result disagrees with the prior literature, where shuffling of mini-batches is helpful for LSTM models in univariate time-series predictions [10]. A likely reason could be that the random shuffling between mini-batches caused a loss of temporal information among sequences, where this temporal information was needed by our recurrent models.

Furthermore, in this paper, we tried to reduce overfitting using the dropout regularization technique. We found that in dropout present conditions, the difference in the test RMSE and training RMSE was reduced compared to the dropout absent conditions. However, adding dropout layers only reduced overfitting by a small amount, perhaps, due to the complexity of the multi-headed models. Prior research has proposed other regularization techniques such as l1 and l2 regularization for the problem of overfitting [12]. These regularization techniques add a regularization term to the cost function to penalize the model for having several parameters. The parameter reduction leads to simpler models that likely reduce overfitting. In future, we plan to apply the l1 and l2 regularization to evaluate their ability to reduce overfitting in data.

Overall, we believe that the multi-headed architectures and weighted ensemble approaches could be helpful to caregivers, patients, and pharmaceutical companies to predict per-patient expenditures where we can utilize the demographic details and other patient variables in predicting their future expenditures. In future, we plan to perform long-range bi-weekly or monthly predictions to evaluate the capacity of multi-headed neural network architectures for making such predictions. Also, as part of our future research, we plan to evaluate other novel networked architectures (e.g., generative adversarial networks) and their ensembles for time-series forecasting of healthcare expenditure data.

Acknowledgements The project was supported by grant (award: #IITM/CONS/RxDSI/VD/33) to Varun Dutt.

References

1. Pham T, Tran T, Phung D, Venkatesh S (2016) Deepcare: a deep dynamic memory model for predictive medicine. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, Cham, pp 30–41
2. Hunter J (2016) Adopting AI is essential for a sustainable pharma industry. Drug Discov World: 69–71
3. Xing Y, Wang J, Zhao Z (2007) Combination data mining methods with new medical data to predicting outcome of coronary heart disease. In: 2007 international conference on convergence information technology (ICCIT 2007), IEEE, pp 868–872
4. Kaushik S, Choudhury A, Dasgupta N, Natarajan S, Pickett LA, Dutt V (2017) Using LSTMs for predicting patient's expenditure on medications. In: 2017 international conference on machine learning and data science (MLDS), IEEE, pp 120–127
5. Feng Y, Min X, Chen N, Chen H, Xie X, Wang H, Chen T (2017) Patient outcome prediction via convolutional neural networks based on multi-granularity medical concept embedding. In: 2017 IEEE international conference on bioinformatics and biomedicine (BIBM), IEEE, pp 770–777
6. Huang Z, Shyu ML (2012) Long-term time series prediction using k-NN based LS-SVM framework with multi-value integration. In: Recent trends in information reuse and integration. Springer, Vienna, pp 191–209
7. Gamboa JCB (2017) Deep learning for time-series analysis. arXiv preprint: arXiv:1701.01887
8. Bagnall D (2015) Author identification using multi-headed recurrent neural networks. arXiv preprint: arXiv:1506.04891
9. Adhikari R, Verma G, Khandelwal I (2015) A model ranking based selective ensemble approach for time series forecasting. Procedia Comput Sci 48:14–21
10. Kaushik S, Choudhury A, Sheron PK, Dasgupta N, Natarajan S, Pickett LA, Dutt V (2019) AI in healthcare: time-series forecasting using statistical, neural, and ensemble architectures. In: Frontiers in big data (under review)
11. Adhikari R, Agrawal RK (2014) Performance evaluation of weights selection schemes for linear combination of multiple forecasts. Artif Intell Rev 42(4):529–548
12. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
13. Zhao Z, Chen W, Wu X, Chen PC, Liu J (2017) LSTM network: a deep learning approach for short-term traffic forecast. IET Intel Transp Syst 11(2):68–75
14. Xingjian SHI, Chen Z, Wang H, Yeung DY, Wong WK, Woo WC (2015) Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Advances in neural information processing systems, pp 802–810
15. Lin T, Guo T, Aberer K (2017) Hybrid neural networks for learning the trend in time series. In: Proceedings of the twenty-sixth international joint conference on artificial intelligence, pp 2273–2279
16. Kim TY, Cho SB (2019) Predicting residential energy consumption using CNN-LSTM neural networks. Energy
17. Bagnall D (2016) Authorship clustering using multi-headed recurrent neural networks. arXiv preprint: arXiv:1608.04485
18. Jose VRR, Winkler RL (2008) Simple robust averages of forecasts: some empirical results. Int J Forecast 24(1):163–169
19. Danielson E (2014) Health research data for the real world: the MarketScan: registered: databases. Truven Health Analytics, Ann Arbor, MI
20. Scott G (2014) Top 10 Painkillers in the US. MD magazine. Retrieved from https://www.mdmag.com/medical-news/top-10-painkillers-in-us

21. Kaushik S, Choudhury A, Dasgupta N, Natarajan S, Pickett LA, Dutt V (2018) Evaluating frequent-set mining approaches in machine-learning problems with several attributes: a case study in healthcare. In: International conference on machine learning and data mining in pattern recognition, Springer, Cham, pp 244–258
22. Yilmaz I, Erik NY, Kaynar O (2010) Different types of learning algorithms of artificial neural network (ANN) models for prediction of gross calorific value (GCV) of coals. Sci Res Essays 5(16):2242–2249

Chapter 15
Soft Computing Approaches to Investigate Software Fault Proneness in Agile Software Development Environment

Pooja Sharma and Amrit Lal Sangal

1 Introduction

In current eras, agile software development (ASD) practices like scrum, extreme programming, the agile unified process, Kanban, lean software development, dynamic systems development method (DSDM) and others techniques [1–5] have replaced the traditional software development (TSD) approaches. Application of agile methodologies is related to enhancement in progress practices, i.e., quicker delivery, effective communication, improved class product with reduced bugs, and less overhead costs [3, 5]. In the literature, various articles focus on achievement features of ASD [6], agile project achievement with respect to traditional methods [7], advanced product frameworks for ASD maturity [8], organizational and people considerations [9, 10]. According to Beck et al. [11], the highest priority in ASD is to pacify the client with the incessant release of the projects. The projects developed are continuously produced as updates in which testing and restructuring actions are performed in parallel. If developments are incessantly being made in a spiral development manner, modules will undergo changes continuously, and newfangled units will be added further to sustain new necessities for components. It may create new defects [12]. Owing to a different association among diverse updates of the identical software development if faultiness in components/methods/class happens in an update, it may be innate later. As a result, determination of faultiness in software components becomes significant as it helps to categorize components that necessitate restructuring or complete testing. According to [13], there are numerous benefits of SFP, as it supports to improve the testing and maintenance procedure, thus surge the product value.

P. Sharma (✉) · A. L. Sangal
Department of Computer Science & Engineering, Dr. B.R. Ambedkar National Institute of Technology, Jalandhar, Punjab, India
e-mail: poojanitjal@gmail.com; poojas.cs.16@nitj.ac.in

A. L. Sangal
e-mail: sangalal@nitj.ac.in

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_15

Errors, bug, defects, faults, and failures are interrelated terms which create confusion in their descriptions [12]. According to the IEEE standard '610-1990,' these terms are defined as error: Human mistake which produces inappropriate outcome. Bug: An unanticipated outcome or indefinite aberration operation found by a programmer, afterward compile of code, the test stage is known as a bug. Fault: An inappropriate step, method, or information description in a computer creates the program code to accomplish in unsuspected style, which is generally recognized as defect, and these are initiated by the mediator, not by the programmer. Failure: Incapability of a process or software module to achieve its essential purposes within definite performance necessities. Faults are unlike from failures. Faults indicate the superiority characteristic which elucidates a situation that software fails to achieve its anticipated occupations.

Since last two years, SFP has addressed problems in software faults from two viewpoints: First is to suggest a new technique or method-combination technique to surge fault prediction efficiency, and second is to use new constraints to recommend extreme operative features for the fault estimation [14]. The SFP models are constructed based on the approaches of product metrics which already present in the literature in abundance [15, 16], but models employ both products, and process metrics are little known. Moreover, significant work has been made for the classification of the process metrics like number of revisions (NR), number of distinct committers (NDC), number of modified lines (NML), and number of defects in the previous versions (NDPV), which affects the effectiveness of fault proneness [17–19]. In order to develop advanced models and launch indication, more studies need to be conducted. Although numerous approaches (e.g., logistic regression, Naive Bayes, neural networks) are employed in fault prediction methods, it is not apparent whether the features which perform better in one practice will also be convenient in other techniques. So, the investigations are repeated using other methods [20].

To this effect, authors in this work are using soft computing methods to generate fault prediction models as an application of these methodologies and make use of the concepts of indecisiveness, inaccuracy, guesstimate, etc., [21]. To attain the goal of the study, experiments are performed on different versions of Ant project from the PROMISE repository [22, 23]. Object-oriented metrics' suites along with process metrics are employed to identify fault-proneness classes in software modules developed in multiple, sequential releases using a highly iterative or agile process. Further, to assess the results of the proposed prediction model, evaluation criteria based upon ROC (AUC) is applied.

2 Related Work

In these days, SFP is the most emergent research area. Numerous works conducted on fault prediction with several methodologies. Various methodologies for fault prediction are logistic regression (LR) [24, 25], naive Bayes (NB) [26, 27], support vector machines (SVM) [28], FIS [29], KNN [30], decision tree [16, 30], random forest (RF)

[31], linear or multiple regression (L/MR) [32–34], neural networks [35, 36]. SVM and naïve Bayes (NB) methods are compared [37] by performing experimentations on datasets of NASA MDP repository. Erturk and Sezer [14] provided an assessment of soft computing methods for fault determination using McCabe metrics. Czibula et al. [38] offered a new cataloguing method created on 'relational association rule mining' in SFP. Khosgoftaar et al. [39] used 'rule-based' and 'case-based-learning' approaches to distinguish model as faulty/non-faulty. Lu et al. [40] discussed the active learning method, which estimates for initial-level updates, with the exemption of the preliminary updates of the software. Additionally, complete reviews on defect prediction were presented by Catal and Diri [41], Kitchenham [42] and Hall et al. [43].

As observed from the above studies, statistical model has been discovered much for the evaluation of software fault attributes. Soft computing methods are effectively applied transversely in several fields like investment, medication and other commerce zones in the literature. Recently, the authors have exposed inordinate attention in the direction of soft computing methods to resolve the indeterminate problems in software fault prediction.

2.1 Soft Computing Approaches

The proposed research framework makes use of FIS, ANN, and ANFIS as it provides a hybrid approach which makes use of skilled judgment and data to learn to change estimate gradually. FIS is a Mamdani style rule-based method which is reliant on the fuzzy concept. FIS structures are based on linguistic values. This chapter discussed Mamdani-based inference process which is processed in following phases: (i) fuzzification method, (ii) rule-valuation based on input metrics, (iii) aggregation of the rule-evaluation output, and (iv) defuzzification.

ANN is a soft computing method stimulated like the working of the human mind. ANN belongs to a collection of information-processing systems which can be used to find information, outlines or models from a huge volume of data. ANN models can be run with predictable or lost values in datasets. It holds large numeral of neurons which are interrelated to one another. In the proposed methodology, ANN used three-layered feed-forward method and also used supervised machine learning procedures. Inputs are provided using input vectors at the input layer to ANN. Weights are restructured in each repetition for matching the goal data or decrease mean square error (MSE). The fault is taken as goal value in the Ant dataset from promise repository. ANN system is trained using the Levenberg–Marquardt (LM) algorithm because it works like a black box, and its readability and interpretability are difficult to find. Additional details of ANN are given by Hecht-Nielsen [44].

ANFIS is a graphical system based on Takagi–Sugeno fuzzy type and trained using a neural network. Nodes are connected to complete the functionality in the collected layers in the network. Thus, it associates the features of both ANN and

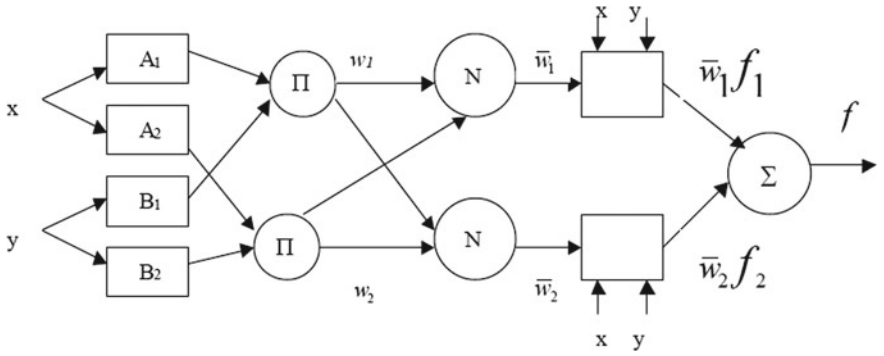


Fig. 1 ANFIS structure

FIS. The outcome of each Takagi–Sugeno rule is the addition of linear association of inputs and a constant value of ANFIS.

Let us take $z = f(x, y)$, which is a first-order Takagi-FIS and consists of two rules.

1. If x is a_1 and y is b_1, then $f_1 = a_1x + b_1y + c_1$
2. If x is a_2 and y is b_2, then $f_2 = a_2x + b_2y + c_2$

$[x, y]$ are input vectors, as shown in Fig. 1. The shooting assets of w_1 and w_2 are multiplication of membership scores. The output (f) is the average weighted score for every rule's output $\bar{w}_1$, and $\bar{w}_2$, are the fraction of each fire asset to the total of all firing assets. Sugeno fuzzy model is learned by the integration of fuzzy model with adaptive neural networks, and we get the network known as ANFIS, which is represented in Fig. 1.

ANFIS is constructed using five layers exclusive of input layer, i.e., layer 0:

Layer 0, i.e., input layer contains 'n' number of nodes, where n denotes the total number of inputs.

Layer 1 is the fuzzification layer with a linguistic value like Gaussian function with mean.

$$A_i(x_0) = \frac{1}{1 + \left[\frac{x - c_i}{a_i}\right]^{2b_i}} \tag{1}$$

$\{a_i, b_i, c_i\}$ are the number of parameters. When the parameter's value varies, the bell-shaped function differs in shape. Parameters in the respective layers are called premise parameters, and the back-propagation algorithm improves their values during the learning phase. As the value of the parameters varies, the value of the linguistic term A_i also varies.

Layer 2 is known as the product layer. The value of each node is obtained by multiplying linguistic inputs which are calculated in the previous layer, and node value is fixed. The membership values $\mu A_i(x_0)\mu B_i(x_1)$ are multiplied to find the weighting strength of rules, where x_0 has the linguistic value of A_i, and x_i has the

linguistic value of B_i, in the predecessor part of Rule i. Here are p^n nodes which denote the number of rules in layer 2. Every node has the forerunner part of the rule (if part of the if-then rule).

$$w_i = \mu A_i(x_0) * \mu B_i(x_1) \tag{2}$$

Layer 3 is called the normalized layer, which is used to stabilize the power of rules using

$$\overline{W_i} = \frac{w_i}{\sum_{j=1}^{R} w_j} \tag{3}$$

Here, w_i is the weighting asset of the ith rule.

Layer 4 is the defuzzification layer. These rules are applied in the weighted output of each node.

$$\overline{W_i} f_i = \overline{W_i}(p_0 x_0 + p_1 x_1 + p_2 x_2) \tag{4}$$

P_is is the list of parameters, the value of $i = n + 1$, where n is the total inputs assigned to the model. The equation written above is only for two inputs and can be extended for more inputs also. Finally, $\overline{W_i}$ assigns the output 'O' of layer 3. The parameters of the layers are restructured using learning algorithm in each step. The least square approximation method is used in ANFIS and a sequential model; the back-propagation process is used to train the network.

Layer 5 is called the output layer. It is the summation of all incoming nodes as a single node which is represented as the modelled output by ANFIS network.

$$\sum_i \bar{w}_i f_i = \frac{\sum_i \bar{w}_i f_i}{\sum w_i} \tag{5}$$

where $\bar{w}_i f_i$ represents the weighted output of node i.

3 Dataset and Metrics Definitions

NASA MDP [22] and PROMISE [23] are renowned repositories for envisaging the software faults. The PROMISE repository contains datasets about the SFP problem. We have chosen versions of Ant project from the PROMISE repository. For performing the fault prediction, four software attributes are taken as input, and three CK metrics, i.e., CBO, WMC, and RFC [14, 24, 25] and one process metric NR [20] are considered.

The definitions of the software metrics are as follows:

1. CBO: Coupling between objects is the total number of other classes which is coupled to a class. If the value of CBO is high, then maintainability, testability, modularity, and reusability values decrease and the complication of the system increases. So, the value of CBO should be less.
2. Weighted method count (WMC): It is the total of complexities of all the methods in a class. As the value of WMC increases, the specialty degree also rises.
3. Response for class (RFC): When a class object receives a call, then the total number of modules executed in reply is called the value of RFC.
4. Number of revisions (NR): It deals with the numeral of revisions retrieved from the main program in a version control system.

Table 1 presents the detail of the dataset along with the range of values used.

4 Proposed Methodology

In the study, first authors build a fuzzy inference system (FIS) and then used ANN and ANFIS's both data-driven methods for the evaluation and explanation of results. ANN is the best method for fault prediction because the achievement of the ANN for predicting the faults has been presented in the literature [44–49].

In this chapter, the attributes (CBO, RFC, WMC, and NR) are used as input attributes, and bug characterizes as output linguistic attribute. Input attributes are classified into three ranges that are low, medium, and high, and output attribute is specified with two values as low and high. The initial value for all input attributes ('*a*') is zero for all software metrics; the largest range value ('*c*') is the present maximum value of the relevant metric, and moderate value ('*b*') is half of '*c.*' The crisp values (metric values) of linguistic attributes to fuzzy values are converted using membership functions. In this study, we used triangular membership functions. If-then rules are composed in rule-based FIS model. Membership functions range, used for the selected metrics from project datasets, is presented in Table 1. As there are four inputs (WMC, CBO, RFC, and NR) and one output (faultiness/bug), a total of 81 rules are generated and applied. Table 2 presents the sample details of the if-then rules used in the FIS system. The common execution details of the fuzzy inference system are discussed in Table 3. The data gathered for four software releases of projects Ant 1.4, 1.5, 1.6, 1.7 are being used for validation and implementation of proposed techniques.

The ANN models were built for product metrics (WMC, CBO, and RFC) and product plus process metrics (WMC, CBO, RFC, and NR). Figure 2 shows the ANN model structure with four inputs. For four inputs model, a total of 10 neurons are used in the hidden layers. Due to inequality $\leq 2N + 1$ (N is the input) the neuron count of the hidden layer is six. The training method for ANN models is scaled as conjugate gradient and run as a pattern recognition network. It is a feed-forward network which can be trained to categorize inputs according to the goal. The classified output data

Table 1 Range of values for the metrics for different releases of Ant project dataset

Dataset Ant v1.4	Metrics											
Total instances = 178 Defective instances = 40 Defect rate = 0.16	CBO			WMC			RFC			NR		
	L	H	M	L	H	M	L	H	M	L	H	M
	0	136	10.75	0	77	10.49	0	196	33.83	0	22	2.843
Dataset Ant v1.5	Metrics											
Total instances = 293 Defective instances = 32 Defect rate = 0.225	CBO			WMC			RFC			NR		
	L	H	M	L	H	M	L	H	M	L	H	M
	0	193	10.64	0	91	10.9	0	213	30.66	0	40	7.894
Dataset Ant v1.6	Metrics											
Total instances = 351 Defective instances = 92 Defect rate = 0.109	CBO			WMC			RFC			NR		
	L	H	M	L	H	M	L	H	M	L	H	M
	0	243	11.45	0	100	11.14	0	247	34.22	0	46	7.852
Dataset Ant v1.7	Metrics											
Total instances = 745 Defective instances = 166 Defect rate = 0.26	CBO			WMC			RFC			NR		
	L	H	M	L	H	M	L	H	M	L	H	M
	0	499	11.04	0	120	11.07	0	288	34.36	0	63	8.673

L—low, H—high, M—mean

Table 2 Sample of fuzzy 'if-then' rules in the FIS model

	Input metrics					
If	WMC	CBO	RFC	NR	Then	Faultiness
R-1	L	L	L	L		L
R-2	M	L	L	L		L
R-3	H	L	L	L		L
R-4	L	M	L	L		L
R-5	H	M	M	L		M
.	.	.	.	.		.
.	.	.	.	.		.

R—rule, L—low, M—medium, H—high

Table 3 Fuzzy inference system details

Fuzzy method	Mamdani
Fuzzy input variables	Minimum Medium Maximum
Fuzzy output variables	Minimum Maximum
Membership function	Triangular
Input function range	Minimum value of metrics Maximum value of metrics
Output function range	0 1
Rule count	81

for pattern recognition system consist of bug data. Supervised machine learning algorithms are used by ANN and train the data using Levenberg–Marquardt (LM). The detailed description of the ANN is found in [46]. The ANN model performs the classification task, as a result of approximately fifty repetitions with 0.134 mean squared errors. The training process will be aborted when the performance gradient reduces below the lowest performance gradient.

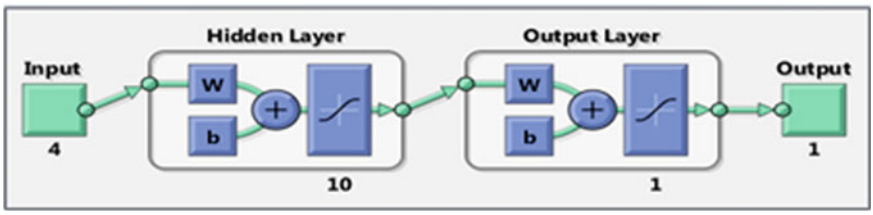


Fig. 2 Artificial neural network model structure

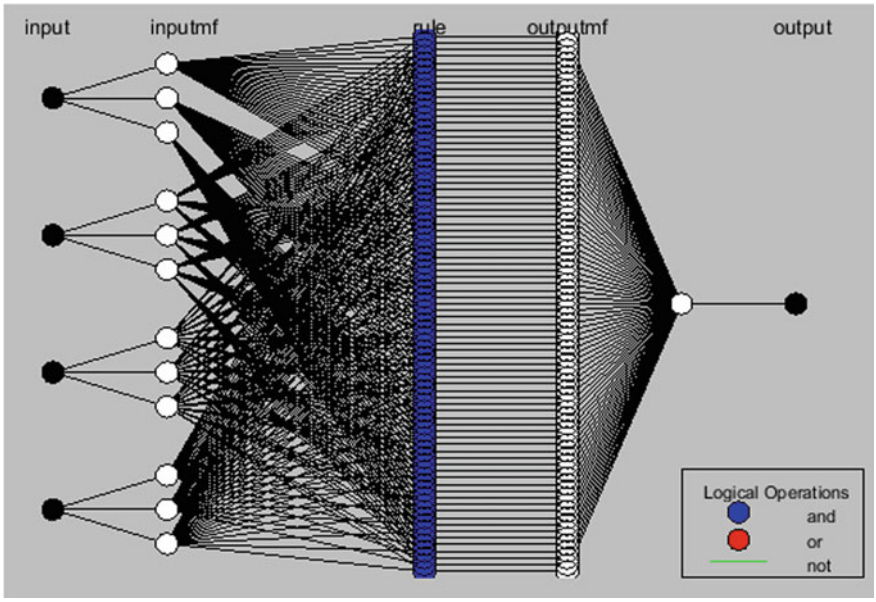


Fig. 3 ANFIS structure

ANFIS architecture (having four inputs) consists of five layers as shown in Fig. 3, and total stages with execution information are as follows.

From initial layer (layer 0), input parameters with respect to metrics (CBO, RFC, WMC, and NR) are taken for input. In layer 1, neurons are fuzzified with the crisp inputs using membership function. Triangular membership functions are considered with three fuzzy sets low, medium, and high. In layer 2, the rule for the input neurons is defined, and a total of 81 rules are generated and fired. Train dataset consists of approximately 777 cases. Layer 3 normalizes the weight factor of every rule coming from layer 2. The defined rules are applied with the normalized strength to layer 4, and the impact of each rule toward the output is calculated. The crisp output in layer 5 is obtained by adding all the results coming from layer 4. When epoch is observed for the forward pass, the least-squares estimator is employed, and the gradient descent method is implemented in the backward pass. In the forward pass, coefficients in the polynomial expression are tuned, and in the backward pass, ranges and parameters of membership functions are adjusted. Model training is continuing until the change in errors between the current and earlier iteration is obtained, or the preferred epoch touched. The model runs up to 100 iterations and then stopped. Figures 4 and 5 provides the details of testing and training errors.

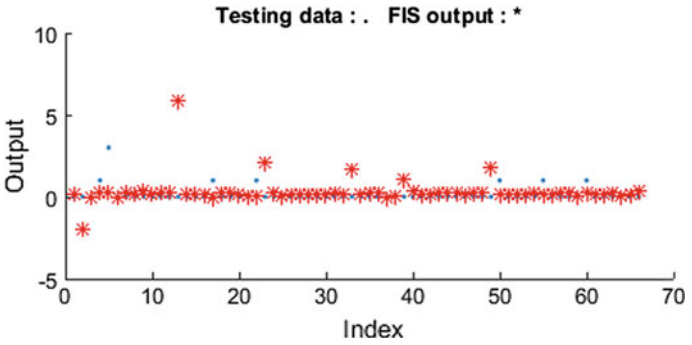


Fig. 4 Testing error mapping Takagi–Sugeno FIS to ANFIS

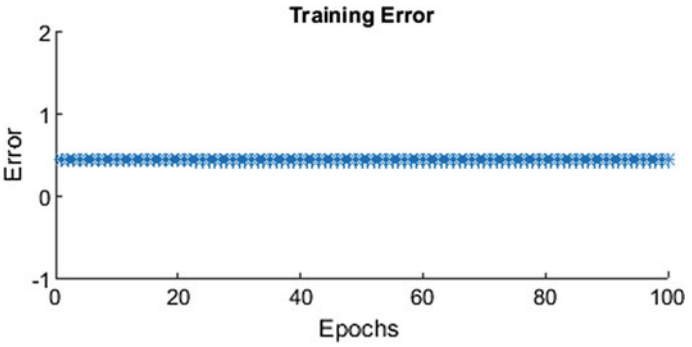


Fig. 5 Training error mapping Takagi–Sugeno FIS to ANFIS

5 Results and Discussion

We analyzed the result using ROC (AUC), accuracy, and standard error. These performance measures measure the performance evaluation for Ant project. To attain these evaluations, experimentations are performed in the MATLAB (R2015b). In the proposed study, the experiments are performed on three methods: (i) FIS (ii) ANN (iii) ANFIS models, which are trained and tested using Ant-projects versions.

Accuracy: Act as primary constraint used for authenticating any prediction or classification model. Definition of accuracy is the number of correct assessments completed to the overall assessments. The formula for accuracy evaluation is as:

$$\text{Accuracy} = \text{Number of correct evaluations completed/Overall evaluations}$$

$$\text{Accuracy} = (\text{TP} + \text{TN})/\text{Total instances} * 100$$

Here, TP is true positive, and TN is true negative.

Receiver operating characteristics curve (ROC): It defines the FP and TP rates of the prediction. ROC is a probability curve, and AUC represents the degree or measure of reparability. ROC (AUC) is capable of distinguishing between the classes. More the value of AUC, better the model is at predicting false as 0 s and true as 1 s.

$$\text{If the value of AUC} = \text{true means there is no error in the prediction.}$$

$$\text{If AUC} > 0.5, \text{ means it is useful for SFP model.}$$

Standard error of the mean: How much the predicted fault value deviates from the actual fault (accuracy) rate?

$\text{SEM} = \text{SD}/\sqrt{n}$, where SD is predicted standard deviation and 'n' is the number of faulty modules.

Tables 4 and 5 present the values of accuracy, standard error, and ROC for different versions of the selected project datasets. Accuracy is recorded between 81.6 and 89.3 for models which make use of product metrics and process metrics and are based on ANFIS approach with standard error ranging from 0.044 to 0.049, respectively, as compared to the accuracy range 79.1–84.8 for models which considered only product metrics. For ANN models (with product + process metrics), the accuracy range is 79.1–92.6 as compared to accuracy range 71.6–88.9 (with product metrics). For FIS-based models (with product + process metrics), the accuracy range is 66.4–79.1 as compared to accuracy range 61.7–77.1 (with product metrics). This shows that models developed using neural networks have better accuracy, followed by models developed using ANFIS. Fuzzy models have the lowest accuracy among them. Also, the models developed using product and process metrics perform better than the models developed using only product metrics.

6 Conclusion and Future Research Scope

SFP is a classification method which signifies that the software module is faulty/non-faulty by taking some metrics of software projects. It helps us to make the software product profitable, effective, and consistent.

In the proposed study, FIS, ANN, and ANFIS-based fault prediction methods are employed to organize the faulty modules. Results of the projected methods are explored in ROC (AUC); accuracy and standard error. For models with three inputs, i.e., WMC, CBO, and RFC, the values of accuracy obtained are 84.8% for ANFIS, 88.8% for ANN, and 77% for FIS. For models with four inputs (WMC, CBO, RFC, and NR), the values of accuracy obtained are 88.2% for ANFIS, 92.6% for ANN, and 79.1% for FIS. It is found from the experimentation that the results of the ANN

Table 4 Description of results with product metrics

Predicted dataset	ANFIS			ANN			FIS		
	ROC	Std. error	Accuracy	ROC	Std. error	Accuracy	ROC	Std. error	Accuracy
ANT 1.4	0.791	0.050	79.1	0.716	0.053	71.6	0.617	0.058	61.7
ANT 1.5	0.810	0.036	92.5	0.874	0.033	87.4	0.768	0.048	76.8
ANT 1.6	0.861	0.047	86.1	0.889	0.029	88.9	0.767	0.043	76.7
ANT 1.7	0.848	0.046	84.8	0.888	0.031	88.8	0.771	0.044	77.1

Table 5 Description of results with product and process metric

Predicted dataset	ANFIS			ANN			FIS		
	ROC	Std. error	Accuracy	ROC	Std. error	Accuracy	ROC	Std. error	Accuracy
ANT 1.4	0.816	0.049	81.6	0.791	0.046	79.1	0.664	0.053	66.4
ANT 1.5	0.825	0.040	89.3	0.897	0.024	89.7	0.760	0.042	76
ANT 1.6	0.855	0.043	88.2	0.919	0.020	91.9	0.781	0.037	78.1
ANT 1.7	0.882	0.044	85.8	0.926	0.019	92.6	0.790	0.035	79

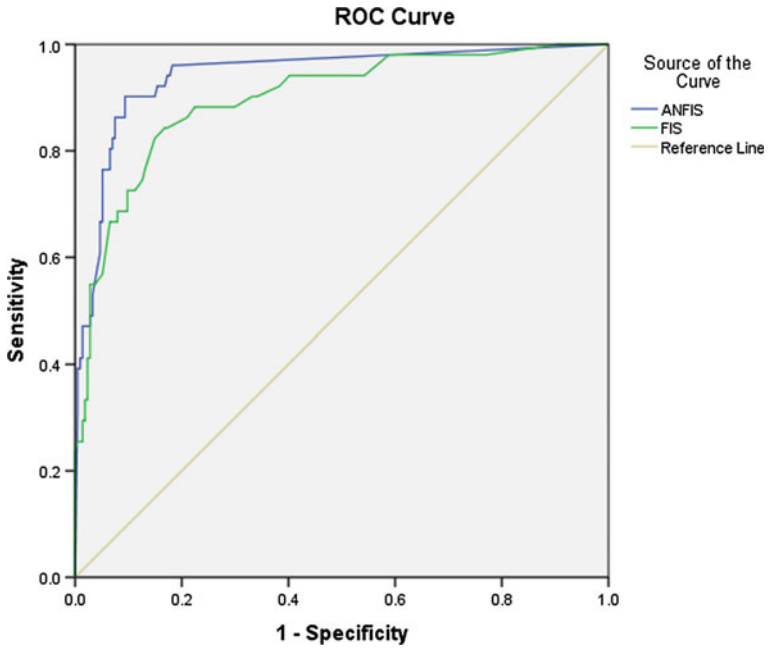


Fig. 6 ROC curve for the proposed methods

approach outperform than ANFIS approach. Figure 6 presents the ROC curve for the proposed methods. Fault-proneness modules' accuracy is found significantly lower, ranging from 70 to 85%, with a higher range of misclassification rate [14, 15]. Another concern of software fault prediction research is uneven dispersal of faults in the fault datasets which may clue to partial learning [17].

Moreover some topics like optimal software metric measures should be included in fault estimation methods along with investigation of cost efficiency of the proposed models. For future work, the proposed methodology is to be extended for developing more advanced models with high accuracy for assessing the faultiness of the modules in the agile software development environment. Also, the cross-project and cross-company projects' defect datasets can be considered for experimentation and to generalize the results.

References

1. Çetin E, Onay Durdu P (2019) Blended Scrum model for software development organizations. J Softw Evol Proc 31:e2147. https://doi.org/10.1002/smr.2147
2. Alzoubi YI, Gill AQ, Moulton B (2018) A measurement model to analyze the effect of agile enterprise architecture on geographically distributed agile development. J Softw Eng Res Dev 6(1). https://doi.org/10.1186/s40411-018-0048-2

3. Rasnacis A, Berzisa S (2017) Method for adaptation and implementation of agile project management methodology. Procedia Comput Sci 104(43):50. https://doi.org/10.1016/j.procs.2017.01.055
4. Schwaber K, Sutherland J (2013) The scrum guide. SCRUM. org, July 2013
5. Kupiainen E, Mantyla MV, Itkonen J (2015) Using metrics in agile and lean software development—a systematic literature review of industrial studies. Inf Softw Technol 62:143–163. https://doi.org/10.1016/j.infsof.2015.02.005
6. Chow T, Cao DB (2008) A survey study of critical success factors in agile software projects. J Syst Softw 81:961–971
7. Ambler S (2014) Lean and agile software development is more successful than waterfall. Retrieved March 01, 2016, from Scott Ambler and Associates: http://scottambler.com/backup_muse/lean-and-agile-softwaredevelopment-is-more-successful-than waterfall.html
8. Fontana RM, Meyer VJ, Reinehr S, Malucelli A (2015) Progressive outcomes: a framework for maturing in agile software. J Syst Softw 102:88–108
9. Iivari J, Iivari N (2011) The relationship between organizational culture and the deployment of agile methods. Inf Softw Technol 53:509–520
10. McHugh O, Conboy K, Lang M (2012) Agile practices: the impact on trust in software project teams. Software, IEEE 29(3):71–76
11. Beck K, Beedle M, van Bennekum A, Cockburn A, Cunningham W, Fowler M, Grenning J, Highsmith J, Hunt A, Jeffries R, Kern J, Marick B, Martin RC, Mellor S, Schwaber K, Sutherland J, Thomas D (2007) Manifesto for agile software development
12. Huizinga D, Kolawa A (2007) Automated defect prevention: best practices in software management. Wiley
13. Catal C (2011) Software fault prediction: a literature review and current trends. Expert Syst Appl 38(4):4626–4636
14. Erturk E, Sezer EA (2015) A comparison of some soft computing methods for software fault prediction. Expert Syst Appl 42(4):1872–1879. https://doi.org/10.1016/j.eswa.2014.10.025
15. Denaro G, Pezze' M (2002) An empirical evaluation of fault-proneness models. In: Proceedings of the 24th international conference on software engineering, ICSE'02, New York, NY: ACM, pp 241–251. https://doi.org/10.1145/581339.581371
16. Gyimothy T, Ferenc R, Siket I (2005) Empirical validation of object-oriented metrics on open source software for fault prediction. IEEE Trans Softw Eng 31(10):897–910. https://doi.org/10.1109/TSE.2005.112
17. Hassan AE (2009) Predicting faults using the complexity of code changes. In: ICSE'09: Proceedings of the 31st international conference on software engineering, IEEE Computer Society, Washington, DC, pp 78–88. https://doi.org/10.1109/icse.2009.5070510
18. Illes-Seifert T, Paech B (2010) Exploring the relationship of a file's history and its fault-proneness: an empirical method and its application to open source programs. Inf Softw Technol 52(5):539–558. https://doi.org/10.1016/j.infsof.2009.11.010
19. Jureczko M, Madeyski L (2011) A review of process metrics in defect prediction studies. Metody Informatyki Stosowanej 30(5):133–145. http://madeyski.e informatyka.pl/download/Madeyski11.pdf
20. Madeyski L, Jureczko M (2014) Which process metrics can significantly improve defect prediction models? An empirical study. Softw Qual J 23(3):393–422. https://doi.org/10.1007/s11219-014-9241-7
21. Zadeh LA (1994) Soft computing and fuzzy logic. IEEE Softw 11(6):48–56. http://ieeexplore.ieee.org/abstract/document/329401
22. NASA MDP, http://mdp.ivv.nasa.gov/. Tera-Promise, http://openscience.us/repo/, visit date 01.08.2015. https://doi.org/10.1109/icist.2013.6747602
23. Burrows R, Ferrari FC, Lemos OA, Garcia A, Taiani F (2010) The impact of coupling on the fault-proneness of aspect-oriented programs: an empirical study. In: 2010 IEEE 21st international symposium on software reliability engineering (ISSRE), pp 329–338
24. Kapila H, Singh S (2013) Analysis of CK metrics to predict software fault-proneness using bayesian inference. Int J Comput Appl 74(2):1–4

25. Dejaeger K, Verbraken T, Baesens B (2013) Towards comprehensible software fault prediction models using Bayesian network classifiers. Inst Electr Electron Eng IEEE Trans Softw Eng 39(2):237–257
26. Pai GJ, Dugan JB (2007) Empirical analysis of software fault content and fault proneness using Bayesian methods. Inst Electr Electron Eng (IEEE) Trans Softw Eng 33(10):675–686
27. Mishra B, Shukla KK (2012) Defect prediction for object oriented software using support vector based fuzzy classification model. Int J Comput Appl 60(15):8–16
28. Singh P, Pal NR, Verma S, Vyas OP (2017) Fuzzy rule-based approach for software fault prediction. Inst Electr Electron Eng (IEEE) Trans Syst Man Cybern Syst 47(5):826–837
29. Goyal R, Chandra P, Singh Y (2014) Suitability of KNN regression in the development of Interaction based software fault prediction models. IERI Procedia 6:15–21
30. Fokaefs M, Mikhaiel R, Tsantalis N, Stroulia E, Lau A (2011) An empirical study on web service evolution. In: IEEE international conference on web services (ICWS 2011), pp 49–56
31. Malhotra R, Jain A (2012) Fault prediction using statistical and machine learning methods for improving software quality. J Inf Process Syst 8(2):241–262
32. Nagappan N, Williams L, Vouk M, Osborne J (2005) Early estimation of software quality using in-process testing metrics. In: Proceedings of the third workshop on software quality—3-WoSQ
33. Pai GJ, Bechta Dugan J (2007) Empirical analysis of software fault content and fault proneness using bayesian methods. IEEE Trans Softw Eng 33(10):675–686
34. Gondra A (2008) Applying machine learning to software fault-proneness prediction. J Syst Softw 81(2):186–195
35. Lu H, Cukic B (2012) An adaptive approach with active learning in software fault prediction. In: Proceedings of the 8th international conference on predictive models in software engineering—PROMISE 2012
36. Abaei G, Selamat A, Fujita H (2015) An empirical study based on semi-supervised hybrid self-organizing map for software fault prediction. Knowl Based Syst 74:28–39
37. Cahill J, Hogan JM, Thomas R (2013) Predicting fault-prone software modules with rank sum classification. In: Proceedings of the 22nd Australian conference on software engineering (ASWEC 2013). IEEE Press, 2013, pp 211–219. http://dx.doi.org/10.1109/ASWEC.2013.33
38. Czibula G, Marian Z, Czibula IG (2014) Software defect prediction using relational association rule mining. Inf Sci 264:260–278
39. Khosgoftaar TM, Xiao Y, Gao K (2014) Software quality assessment using a multi-strategy classifier. Inf Sci 259:555–570
40. Lu H, Kocaguneli E, Cukic B (2014) Defect prediction between software versions with active learning and dimensionality reduction. In: Proceedings of IEEE the 25th international symposium on software reliability engineering (ISSRE 2014). IEEE Press, 2014, pp 312–322. http://dx.doi.org/10.1109/ISSRE.2014.35
41. Catal C, Diri B (2009) A systematic review of software fault prediction studies. Expert Syst Appl 36(4):7346–7354. https://doi.org/10.1016/j.eswa.2008.10.027
42. Kitchenham B (2010) What's up with software metrics? A preliminary mapping study. J Syst Softw 83(1):37–51. https://doi.org/10.1016/j.jss.2009.06.041
43. Hall T, Beecham S, Bowes D, Gray D, Counsell S (2012) A systematic literature review on fault prediction performance in software engineering. IEEE Trans Software Eng 38(6):1276–1304. https://doi.org/10.1109/TSE.2011.103
44. Hecht-Nielsen R (1987) Kolmogorov's mapping neural network existence theorem. In: Proceedings of the first IEEE international conference on neural networks. IEEE Press, 1987, pp 11–14
45. Chatterjee S, Nigam S, Roy A (2016) Software fault prediction using neuro-fuzzy network and evolutionary learning approach. Neural Comput Appl. https://doi.org/10.1007/s00521-016-2437-y
46. Negnevitsky M (2005) Artificial intelligence a guide to intelligent systems, 2nd edn. Addison-Wesley, Harlow, England
47. Jang JSR (1993) ANFIS: Adaptive-network-based fuzzy inference system. IEEE Trans Syst, Man Cybern 23(3):665–685

48. Sen S, Sezer EA, Gokceoglu C, Yagiz S (2012) On sampling strategies for small and continuous data with the modeling of genetic programming and adaptive neuro-fuzzy inference system. J Intell Fuzzy Syst 23(6):297–304
49. Kanmani S, Uthariaraj VR, Sankaranarayanan V, Thambidurai P (2007) Object-oriented software fault prediction using neural networks. Inf Softw Technol 49(5):483–492. https://doi.org/10.1016/j.infsof.2006.07.005

Chapter 16
Week Ahead Time Series Prediction of Sea Surface Temperature Using Nonlinear Autoregressive Network with and Without Exogenous Inputs

Geetali Saha and N. C. Chauhan

1 Introduction

Oceans make up more than two-thirds of the Earth surface and house innumerable species throughout the journey of evolution. They themselves are also evolving as the Earth continues to age through the centuries. The warm and cold currents developing in the oceans cause various changes on the surface of the Earth transgressing all man-made boundaries. Many parameters of the ocean are found to reflect such changes. However, the foremost among them is the sea surface temperature and the winds over ocean water that is found to have been correlated to various other biotic parameters affecting the overall global climate.

The National Oceanic and Atmospheric Administration (NOAA) [1], headquartered at USA, plays an important role toward storing, analyzing, and distributing observations related to the ocean. It includes land-based station data for location-based collection of weather parameters, satellite data that is raw data collected by the satellites and archived by NCEI—National Centers for Environmental Information (NCEI) formerly known as National Climatic Data Center (NCDC), radar data for tracking moving targets and Marine or Ocean data collected from ships at sea, moored buoys, and coastal stations among others. NOAA's Pacific Marine Environmental Laboratory (PMEL) [2] ensures measurement of critical observations and advance research in the domains of Earth atmosphere climatic interactions.

Several indices, based on sea surface temperature anomalies (SSTA), are proposed for the tropical Pacific measurement and monitoring for duration of 30 years. Most common are the Niño 3.4 index and the Oceanic Niño Index (ONI) that characterize

G. Saha (✉)
G.H. Patel College of Engineering and Technology, Vallabh Vidya Nagar, Anand, Gujarat, India
e-mail: geetalisaha@gcet.ac.in

N. C. Chauhan
A.D. Patel Institute of Technology, New Vallabh Vidya Nagar, Anand, Gujarat, India
e-mail: narendracchauhan@gmail.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_16

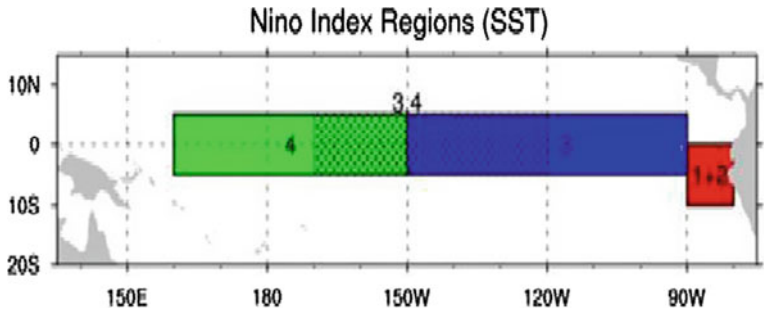


Fig. 1 Region outline [4]

Table 1 Niño indices with their locations and characteristics [5]

Type	Location	Characteristics	Remarks
Niño 1 + 2	0–10 S, 90 W–80 W	Smallest region Most Eastern Niño SST regions	Largest variance of Niño SST
Niño 3	5 N–5 S, 150 W–90 W	Was primary focus for monitoring and predicting El Niño	Trenberth (1997) proved that the ENSO lies further west
Niño 3.4	5 N–5 S, 170 W–120 W	Represents the average equatorial Pacific SSTs	5-month running mean, SSTAs $\geq\pm$ 0.4 °C for six months
ONI	5 N–5 S, 170 W–120 W	Operational definition used by NOAA, widely accepted	3-month running mean, SSTAs $\geq\pm$ 0.5 °C for five months
Niño 4	5 N–5 S, 160 E–150 W	Captures SSTAs in the Central Equatorial Pacific	Less variance than the rest of the Niño regions

the El Niño and La Niña events. Various numbers are used to label corresponding regions by ship tracks since 1949 by Rasmusson and Carpenter [3] (Fig. 1; Table 1).

El Niño (warm) and La Niña (cool) events in the tropical Pacific are identified using the Oceanic Niño Index (ONI), a standard by NOAA. An overlapping consecutive 3-month period at and beyond $\pm$0.5 °C anomaly classifies them to be a warm (El Niño) event or cold (La Niña) event. Further classification in the range of 0.5–0.9 SSTA is weak, 1.0–1.4 is moderate, 1.5–1.9 is strong, and beyond 2.0 is very strong (shown in Fig. 2 in terms of increasing color intensities).

The present paper is organized as follows: Section 2 covers the literature survey. Section 3 gives an insight into the basic nonlinear autoregressive model with and without exogenous input for time series prediction. Section 4 shows the dataset details used in the proposed work. Section 5 is about the proposed methods. Section 6 is about the result obtained using the proposed methods. Section 7 summarizes the results obtained.

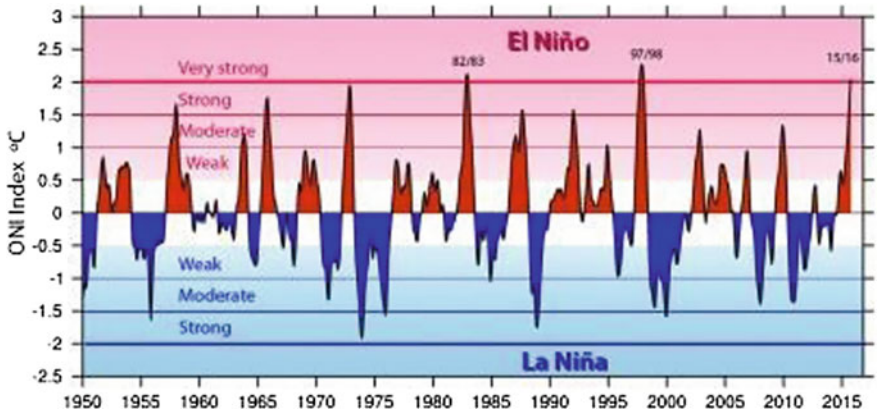


Fig. 2 El Niño (warm in red) and La Niña (cool in blue) events in terms of SSTA across 1949–2015 [5]

2 Literature Review

Climatic variations are circulation driven based on the exchange of energy measured in terms of the sea surface temperature (SST) and this exchange depends on the interactions of the various radiative flux and heat flux at the sea surface. Sea surface temperature anomalies act as a mirror reflecting the dynamic behavior of the marine world, the overall weather and the Earth at large. Many researchers have correlated the variations in the SST values to the occurrence of various other events.

Hu and He [6] have investigated the early appearance of Green Tide in the Yellow Sea during Olympic sailing competition in the coastal waters of Qingdao in Summer 2008. With the help of Floating Algae Index (FAI) and the marine environmental parameters during 2008–2013, they established a green tide monitoring system and concluded that the rising SST anomaly caused the massive algae bloom.

Carbone et al. [7] observed the Hg^0 concentrations and the relative humidity measured at MLO—Mauna Loa Observatory, Hawaii and compared these to the Pacific sea surface temperature (SST) periodicities between 2002 and 2009 with the use of Empirical Mode Decomposition technique. They developed a direct effect of SST on Hg^0 concentrations.

Mustapha et al. [8] have utilized 11 years (1998–2008) of AVHRR—Advanced Very High Resolution Radiometer data consisting of 6602 individual images (via day and night passes) analyzed to derive SST fronts from SST fields using SIED—Single Image Edge Detection algorithm. Their analysis of the Amundsen Gulf Region and the Mackenzie Shelf, recurrent SST patterns, especially at the peak of Summer is detected.

Stock et al. [9], with his team of twelve NOAA faculties, have investigated seven Large Marine Ecosystems (LMEs) adjacent to the USA. They established an indirect link of SST Anomaly on species of interest via mixed layer depths, stratification

and horizontal transports using two dynamical seasonal forecast systems, namely NOAA's, GFDL's, CM 2.5-FLOR prediction system and the NCEP System.

Dinesh Kumar et al. [10] have used monthly SST data from NOAA ERSST at two locations: Arabian Sea (10°–20° N and 60°–70° E) and Bay of Bengal (10°–20° N and 85°–95° E) from 1880 to 2010. They have concluded warmer SSTs characteristics in the Bay of Bengal as compared to the Arabian Sea.

Cañadas and Vázquez [11] have tried to establish a link between the sub-population of short-beaked common Dolphin with SST at the Alboran Sea and the Gulf of Vera. Using data obtained by a survey on common dolphins from 1992 to 2011, and ocean parameters from NOAA Ocean Watch, it was concluded that an increase in SST leads toward declining population density and this was found to be more true in cases of extreme SSTA, emphasizing the need for SSTA calculation.

Lins et al. [12], using PIRATA in the Atlantic Dataset, have performed year long multistep prediction coupled with SVM models. They have obtained MAPE less than 2% and all MAE lower than 0.37 °C.

Picone et al. [13] have investigated the various measurement collection methods that contribute to observations of sea surface temperature in the Italian Sea. ICOADS Dataset (reconstructed time series having temporal data for 106 years starting from 1900 and for the location boundary of 35° N–46° N, 7° E–20° E) is compared with CNR-MED dataset and shows good agreement of SST values at basin scales.

Salles et al. [14] have collected SST data from PIRATA using different sensor buoys, making a dataset containing 50,000 entries; of which every valid test dataset must have at least two years duration of uninterrupted data. Considering variable size of training dataset, the future values of SST are predicted with the help of ARIMA model clubbed with Random Walk as a baseline prediction model.

Using satellite and in situ measurements over a span of 29 years (1981–2009), a relationship between dominant Empirical Orthogonal Function (EOF) and long-term trends of SST variation is established by Park et al. [15]. Differential surface warming is observed in the Yellow Sea where warming rates reduced monotonically with depth having most remarkable warming trends in the neighborhood of Yangtze river, contributed significantly by the river discharge.

Using reanalysis datasets for atmosphere and oceans on daily, 10-day and monthly basis for the July–October months from 2002 to 2005, Wada et al. [16] tried to establish a relationship between maximum Tropical cyclone intensity and SST. Using Argo profiling float observations accumulated Tropical Cyclone Heat Potential (TCHP) since genesis to minimum central pressure (MCP) is calculated and a stronger tropical cyclone is detected in the Western Pacific and validated using TCHP values derived from three oceanic reanalysis datasets.

With the help of approximately 25,000 in situ subsurface temperature profiles during 1997–2007, Ali et al. [17] have tried to develop an ANN model to derive TCHP in the Indian Ocean using satellite-derived SST, sea surface height anomalies at 26 °C isotherm. For validation, 8000 independent in situ profiles in the years 2008–09 are used and the ANN approach is observed to be providing a better performance than the other techniques discussed.

An attempt to synthesize Ocean Subsurface Thermal Structure (OSTS) by ANN approach from surface parameters, like net radiation, net surface heat flux, wind stress, SST and dynamic height from Arabian Sea mooring (15.5° N and 61.5° E), deployed by the Woods Hole Oceanographic Institution during October 1994–October 1995 is made by Ali et al. [18]. Predicted profiles were found to have very low error when compared with the in situ results.

Thus, prediction of SST values helps in forecasting various other occurrences found to be correlated. Different techniques are gaining popularity in this direction, right from the basic numerical techniques to advanced neural network-based approaches.

Neural networks in particular date back its origin to the early 1940s when in 1943, a neurophysiologist Warren McCulloch with a mathematician Walter Pitts published an article describing how the neurons might work by modeling it in using electrical circuits. However, it was later, in 1972 that Kohonen and Anderson, both working independent of each other proposed ADAptive LINear Elements (ADALINE) circuits, thereby reigniting interest in this domain which also resulted into the advent of multi-layer neural networks. It was in 1986 that Rumelhart et al. [19] proposed backpropagation of errors. Later during 1989, Chen and Billings [20] proposed the NARMAX model for input–output representations of nonlinear discrete-time systems.

The nonlinear autoregression method finds versatile applications. Using NARX model, Du et al. [21] have proposed a prediction model for analysis of Surge Protective Devices (SPD) response characteristics. Xie et al. [22] have used NARX neural network and tested it for Co_2 compressor using Gamma Test for single-step ahead prediction. Ercan [23] has targeted day ahead prediction of the exchange rate for OMX Baltic Benchmark GI (OMXBBGI) market index value using previous day's index value. Jawad et al. [24] have proposed a GA-based NARX-NN algorithm and tested the same on real-world datasets of ERCOT (for electrical load prediction) and CDMO (for wind speed prediction). Lipu et al. [25] have used NARX-NN clubbed with lighting search algorithm (LSA) to estimate the state of charge for lithium-ion battery.

Saha and Chauhan [26] have investigated the step ahead prediction of weather parameters: average low temperature, average high temperature, average humidity, average wind speed of the Manaus region, Brazil from 1970 to 2015 using a NAR network.

Saha and Chauhan [27] have proposed Dependency Investigation of Sea Surface Temperature on Sea Bottom Temperature and Sea Surface Salinity using data from Scripps Pier on the Western Coast of California (1916–2015). The SST data has multistep prediction using NAR network and the same using Sea Bottom Temperature and Sea Surface Salinity has single-step ahead prediction using NARX network.

3 Time Series Models

A time series is made of a set of time stamped data, usually collected at predefined regular intervals of time. Its formation is attributed by repeated observations at fixed intervals of time. Many time series data of variable types, dimensions, and utilities are available to researchers for analysis. The UCI Machine Learning Repository [28] is one such collection that is highly recognized by the research scholars of the machine learning community for validating and testing of proposed algorithms. David Aha in 1987 along with fellow graduate students at UC Irvine created this as an ftp archive which in 2007 was updated to its present version by Arthur Asuncion and David Newman. The National Science Foundation funds it. The El Nino dataset used in the proposed algorithm has readings collected from various buoys positioned across the equatorial Pacific.

Most time series are temporal sequential data recorded over time either on an hourly, daily, monthly or yearly basis—but the duration remains constant over the complete series. Through many years, the analysis of these time series has revealed that many characteristics stay hidden in the patterns of the time series. Often, they occur naturally in various application areas. The aim of time series analysis is to mimic, identify, and explore potential hidden characteristics that make a forecast possible. Here, data collected from the mentioned site is fed and simulated using MATLAB tool for data analysis. Later, they are compared to judge its accuracy and relevance using various error measures.

3.1 Basic Nonlinear Autoregressive Models

3.1.1 The Nonlinear Autoregressive Network

This model can accept dynamic inputs represented by time series sets and it is particularly useful to describe the nonlinear dynamic in a wide variety of systems.

Refer Fig. 3.

Its general formulation is given by

$$y(t+1) = F(u(t), u(t-1), u(t-2) \ldots u(t-D)) \tag{1}$$

where

$u(t)$, $u(t-1)$, $u(t-2)$ … $u(t-D)$ are the inputs at time t, $(t-1)$, $(t-2)$, … $(t-D)$ respectively.

F function describes the nonlinear correlation between the model output y $(t+1)$ and the input $u(t)$ based on previous states from time t, $t-1$ to $t-D$ only.

The weights of the neural network are tuned by the Levenberg–Marquardt rule.

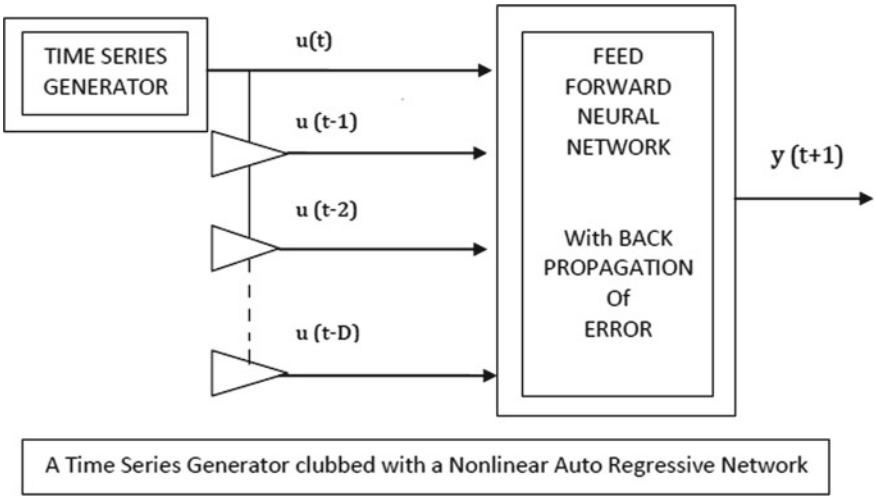


Fig. 3 Nonlinear autoregressive neural network

3.1.2 The Nonlinear Autoregressive Network with Exogenous Inputs

This model is modified to accept dynamic inputs along with any other exogenous inputs represented by time series sets and it is particularly useful to establish a relationship between both time series.

Refer Fig. 4.

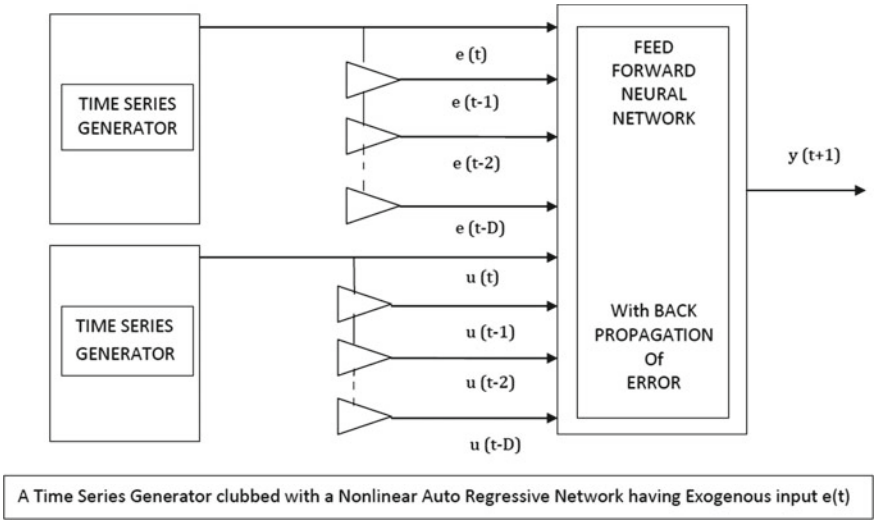


Fig. 4 NARX neural network with $y(t)$ inputs and $u(t)$ exogenous inputs

A nonlinear NARX can be mathematically represented as

$$y(t+1) = G[u(t); e(t)] \tag{2}$$

where

the vectors $u(t)$ and $e(t)$ denote the input and exogenous time series, respectively.

G function describes the nonlinear correlation between the model output y (t + 1) and the inputs ($u(t)$ and $e(t)$) based on previous states from time t, $t-1$ to $t-D$.

3.1.3 The Exogenous Inputs in Parallel Mode

A nonlinear autoregressive network with an additional input has two possible modes of operation. Both of them are shown in Fig. 5. Our proposed algorithm uses parallel mode where the estimated outputs are fed back and included in the output's regressor.

The mathematical expression is given by

$$y(t+1) = H[y(t), \ldots y(t-D); e(t), e(t-1) \ldots e(t-D)] \tag{3}$$

3.2 Error Measures

Comparison of the predicted values with the actual data is error. And there are innumerable error parameters that have various interpretations in the domain of statistical analysis.

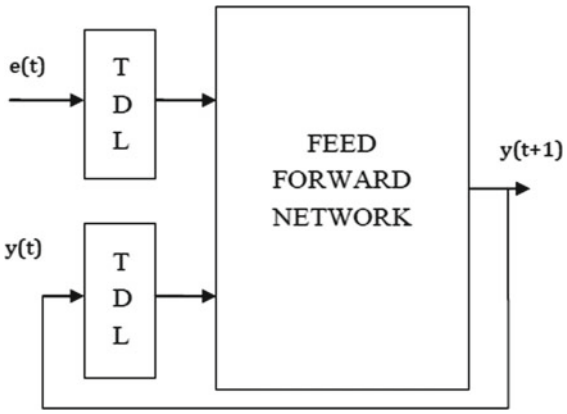


Fig. 5 NARX neural network architecture; TDL is time delay layer

Following are the error parameters associated with our experimental setup and analysis:

Mean square error (MSE),
Normalized mean square error (NMSE)
Root mean square error (RMSE),
Normalized root mean square error (NRMSE),
Maximum absolute error (MaxAE),
Mean absolute error (MeanAE),
Mean absolute percentage error (MAPE),
Standard deviation of data (SDD),
Standard deviation of error (SDE),
Residual standard deviation (SRes)

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_t - f_t)^2 \tag{4}$$

$$\mathrm{NMSE} = \frac{\mathrm{MSE}}{y_{\max} - y_{\min}} \tag{5}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_t - f_t)^2} \tag{6}$$

$$\mathrm{NRMSE} = \frac{\mathrm{RMSE}}{y_{\max} - y_{\min}} \tag{7}$$

$$\mathrm{MaxAE} = \max(\mathrm{abs}(y_t - f_t)) \tag{8}$$

$$\mathrm{MeanAE} = \mathrm{mean}(\mathrm{abs}(y_t - f_t)) \tag{9}$$

$$\mathrm{MAPE} = \frac{1}{m}\sum_{i=1}^{m}(\mathrm{abs}((y_t - f_t)/y_t)) \tag{10}$$

$$\mathrm{SDD} = \mathrm{std\ dev}(\mathrm{DATA}) \tag{11}$$

$$\mathrm{SDE} = \mathrm{std\ dev}(y_t - f_t) \tag{12}$$

$$\mathrm{SRes} = \sqrt{\sum_{i=1}^{m}(y_t - f_t)^2 \frac{1}{m-2}} \tag{13}$$

where

y_t actual data
f_t forecasted data
$y_{\max}$ maximum value of the dataset
$y_{\min}$ minimum value of the dataset
n total count of dataset
m count of multistep; for one week $m = 7$.

4 Dataset Used

The ownership of the data [28] lies with PMEL laboratory of NOAA. A set of buoys, stationed by International Tropical Ocean Global Atmosphere (TOGA)'s Tropical Atmosphere Ocean (TAO) program, are installed for recording.

Donated by Dr. Di Cook, Department of Statistics, Iowa State University in June 1999, it is made up of 178080 instances and 12 attributes having some missing values.

About 70 moored buoys across the equatorial Pacific are installed to measure various parameters—subsurface temperatures down to 500 meters depth, surface winds, sea surface temperatures, air temperature, and relative humidity. The present dataset is composed of the following components date of measurement, location of the buoy, zonal winds (for west < 0, for east > 0), meridional winds (for south < 0, for north > 0), air temperature, relative humidity, and SST. Variations in wind data are $\pm$10 m/s and that in relative humidity is 70–90%. Fluctuations in air temperature and sea surface temperature are both in the range of 20–30 °C. All readings are taken at the same time of the day.

A section of the data having very few missing values is identified. The location identified is at 0° N latitude and −110 E longitudes. It is slightly larger than 10 years data and is dated from 10/05/1985 to 20/07/1995.

As identified in many previous literatures, to avoid the problem of overfitting, training and testing datasets are segregated at the very beginning of the experimental setup. The training data is made up of the values dated from 10/05/1985 to 10/05/1995. The duration from 11/05/1995 to 20/07/1995 is used as an independent test dataset.

5 Proposed Methods

The training data comprising of 10 years of daily data is rearranged and the mean of daily SST value is calculated using the respective last ten years data. This is the mean value of SST. And it is subtracted from the present SST reading giving the sea surface temperature anomaly (SSTA). This is compared to the forecasted value.

5.1 SSTA

5.1.1 Sea Surface Temperature Anomaly (SSTA)

A set of time stamped SST data is converted to time series data. The sea surface temperature time series is analyzed against different possible values of delay and hidden neurons in an open-loop configuration using a nonlinear autoregressive network for single-step prediction.

The purpose of this implementation is meant to obtain the optimized values of H (number of hidden neurons) $= 20$ and D (number of delay neurons) $= 2$. Refer Fig. 6.

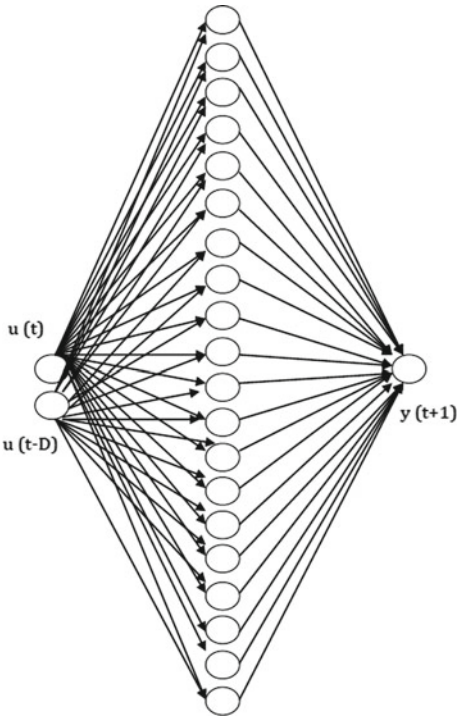


Fig. 6 Proposed NAR NN using optimized values of D and H in open-loop configuration

5.2 SSTA_NAR

5.2.1 Nonlinear Autoregression Network—Multistep Prediction for SSTA (SSTA_NAR)

Using the optimized values of H and D, the purpose of closed-loop implementation is to retain predicted values and continue the forecasting for 7 days of SSTA. TDL represents the Time Delay Layer.

In the present case, last ten years of SST data is fed to the neural network as the training dataset. The testing dataset is a total of 70 days, one week each at a time. Results show 10 iterations compiled one after the other forming a time series representation of 70 timestamps of SST data. Refer Fig. 7.

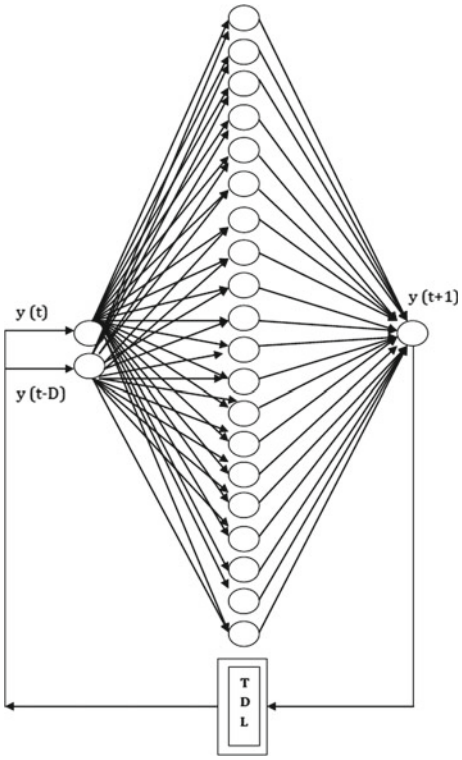


Fig. 7 Proposed NAR NN using optimized values of *D* and *H* in closed-loop configuration

5.3 SSTA_AirT

5.3.1 Nonlinear Autoregression Network—Multistep Prediction Using Air Temperature as an Exogenous Input (SSTA_AirT)

Using the optimized values of H and D, we predict a week ahead values of SSTA using previous values of the SST time series and the air temperature time series.

In the present case, last ten years of (SST and Air Temperature) data is fed to the NN as the training dataset. The testing dataset is a total of 70 days, one week each at a time. Results show 10 iterations compiled one after the other forming a time series representation of 70 timestamps of SST data calculated with the help of Air Temperature data. Refer Figs. 8 and 9 for the open and closed-loop configurations, respectively and Table 3 for the individual week wise results.

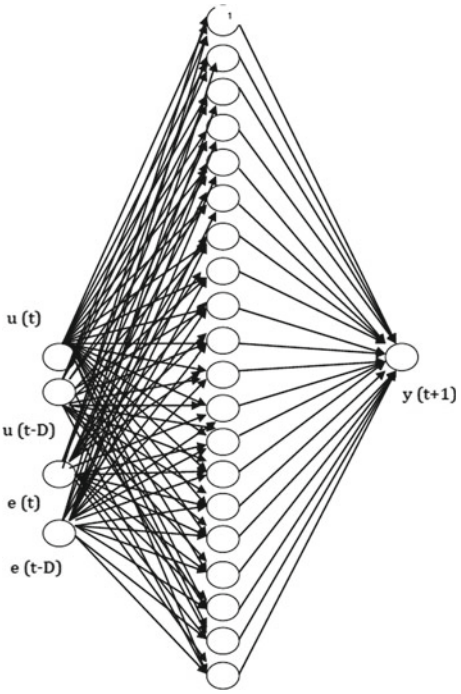


Fig. 8 Proposed NARX-NN with exogenous input $e(t)$ in open-loop configuration

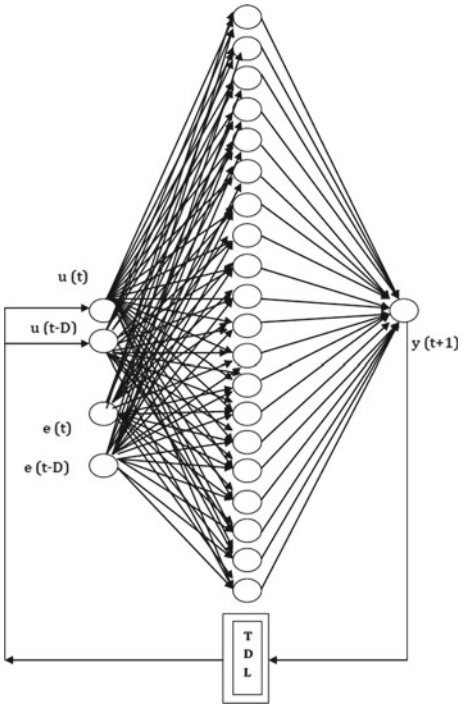


Fig. 9 Proposed NARX-NN with exogenous input $e(t)$ in closed-loop configuration

5.4 SSTA_Z

5.4.1 Nonlinear Autoregression Network—Multistep Prediction Using Zonal Winds as an Exogenous Input (SSTA_Z)

Using the optimized values of H and D, we predict multiple step ahead values of SSTA using previous values of the SST time series and the Zonal wind time series. In the present case, last ten years of (SST and Zonal wind) data is fed to the neural network as the training dataset. The testing dataset is a total of 70 days, one week each at a time. Results show 10 iterations compiled one after the other forming a time series representation of 70 timestamps of SST data calculated with the help of zonal winds data. Refer Figs. 8 and 9 for the open-loop and closed-loop configurations, respectively and Table 4 for the individual week wise results.

5.5 SSTA_M

5.5.1 Nonlinear Autoregression Network—Multistep Prediction Using Meridional Winds as an Exogenous Input (SSTA_M)

Using the optimized values of H and D, we predict multiple step ahead values of SSTA using previous values of the SST time series and the meridional wind time series. In the present case, last ten years of (SST and meridional wind) data is fed to the neural network as the training dataset. The testing dataset is a total of 70 days, one week each at a time. Results show 10 iterations compiled one after the other forming a time series representation of 70 timestamps of SST data calculated with the help of Meridional winds data. Refer Figs. 8 and 9 for the open-loop and closed-loop configurations, respectively and Table 5 for the individual week wise results.

6 Results

Using a 10-year dataset of the equatorial Pacific region in a site-specific approach, the proposed network is used to forecast the next week SST values.

The legends are explained as follows:

SSTA $= T - M$ is the SSTA calculated by taking the difference of mean from actual.
SSTA_NAR is the SSTA calculated using the SST series only.
SSTA_AirT—SSTA calculated using the Air Temperature data as an exogenous input (Figs. 8, 9 and 10; Table 3).
SSTA_Z—SSTA calculated using the Zonal winds as an exogenous input (Figs. 8, 9 and 10; Table 4).
SSTA_M—SSTA calculated using the Meridional winds as the exogenous input (Figs. 8, 9 and 10; Table 5).

The error parameters are tabulated for reference and understanding. It is observed that for all the test datasets, the value of standard deviation of error is far smaller than the standard deviation of data. The values of normalized mean square error (NMSE) and normalized root mean square error (NRMSE) are found to never exceed 0.13 °C and 0.1 °C, respectively. The maximum absolute error (MaxAE) of 2.1496 °C is observed when no exogenous input is provided to the neural network. The value of standard deviation of the residuals (SRes) never exceeds 0.5213 °C. The mean absolute percentage error (MAPE) and mean absolute error (MeanAE) have a maximum value of 0.0369 °C and 0.9847 °C, respectively, which is significantly low.

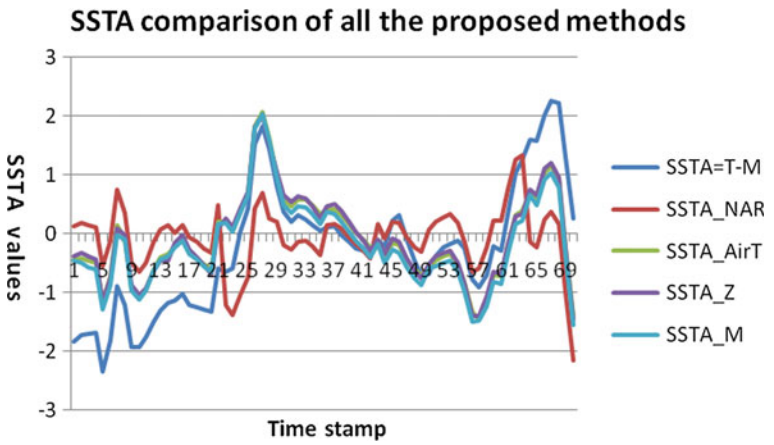


Fig. 10 SSTA comparative of all the proposed methods amounting to 70 timestamps

7 Summary

A comparative of all the methods is tabulated in terms of errors and the time series plots give a pictorial view of the agreement in values.

It is to be noted here that all the methods provide forecasted values very near to the actual SSTA. However, it is the performance of the model in presence of the Zonal and the Meridional winds that provide least error—both in the error comparative and in the time series plots. This is in agreement to the basic fact that this warming and cooling of the sea surface temperature is subject to the wind direction and intensities prevailing at different places at different instances of time.

Table 2 Week (W) wise details of error components calculated using NAR multistep-SSTA_NAR

W	MSE	NMSE	RMSE	NRMSE	MaxAE	MeanAE	MAPE	SDD	SDE	SRes
1	0.1388	0.0117	0.3726	0.0315	0.7518	0.2872	0.012	2.0136	0.389	0.1972
2	0.1502	0.0127	0.3876	0.0328	0.6352	0.3331	0.0141	2.0135	0.3801	0.2051
3	0.0647	0.0055	0.2544	0.0215	0.4976	0.2023	0.0084	2.0135	0.2745	0.1346
4	0.8155	0.069	0.903	0.0764	1.3796	0.8216	0.0328	2.0134	0.8626	0.4778
5	0.0487	0.0041	0.2206	0.0187	0.3495	0.2069	0.0083	2.0146	0.1763	0.1167
6	0.0508	0.0043	0.2254	0.0191	0.418	0.1943	0.0079	2.015	0.2311	0.1193
7	0.0369	0.0031	0.1921	0.0163	0.3084	0.1744	0.0073	2.017	0.207	0.1016
8	0.1009	0.0085	0.3176	0.0269	0.6354	0.2663	0.0113	2.0137	0.3392	0.168
9	0.6445	0.0545	0.8028	0.0679	1.3429	0.6661	0.0275	2.0128	0.7299	0.4248
10	0.846	0.0716	0.9198	0.0778	2.1496	0.6136	0.0257	2.0127	0.9011	0.4867

Table 3 Error components using NARX multistep prediction using additional Air temperature input-SSTA_AirT

W	MSE	NMSE	RMSE	NRMSE	MaxAE	MeanAE	MAPE	SDD	SDE	SRes
1	0.3829	0.032	0.6188	0.0521	1.1558	0.545	0.0231	2.0136	0.3918	0.3274
2	0.5146	0.0435	0.7174	0.0607	1.1055	0.6191	0.0263	2.0135	0.3914	0.3796
3	0.1422	0.0119	0.377	0.0319	0.6115	0.3349	0.0139	2.0135	0.2767	0.1995
4	1.58	0.1337	1.257	0.1063	2.0783	0.9847	0.0378	2.0134	0.8439	0.6651
5	0.3701	0.0313	0.6083	0.0515	0.9968	0.5774	0.0231	2.0146	0.2069	0.3219
6	0.0777	0.0066	0.2788	0.0236	0.4216	0.2385	0.0097	2.015	0.2557	0.1475
7	0.1851	0.0157	0.4302	0.0364	0.6978	0.3774	0.0158	2.017	0.2232	0.2277
8	0.4986	0.0422	0.7061	0.0597	1.3287	0.6291	0.0266	2.0137	0.3464	0.3737
9	0.6732	0.057	0.8205	0.0694	1.4532	0.7027	0.0299	2.0128	0.702	0.4342
10	0.8724	0.0738	0.934	0.079	1.3896	0.8616	0.0348	2.0127	0.9129	0.4942

Table 4 Error components using NARX multistep prediction using zonal winds as an exogenous input-SSTA_Z

W	MSE	NMSE	RMSE	NRMSE	MaxAE	MeanAE	MAPE	SDD	SDE	SRes
1	0.3521	0.0298	0.5934	0.0502	1.1439	0.5011	0.0212	2.0136	0.3833	0.314
2	0.5009	0.0424	0.7078	0.0599	1.0422	0.628	0.0266	2.0135	0.3526	0.3745
3	0.1428	0.0121	0.3779	0.032	0.6782	0.3042	0.0127	2.0135	0.2765	0.2
4	1.4944	0.1264	1.2512	0.1034	2.0118	0.9835	0.0379	2.0134	0.7842	0.6469
5	0.412	0.0349	0.6419	0.0543	1.0528	0.6026	0.0241	2.0146	0.2389	0.3396
6	0.1113	0.0094	0.3336	0.0282	0.5034	0.2897	0.0118	2.015	0.3084	0.1765
7	0.1862	0.0158	0.4315	0.0365	0.7618	0.3518	0.0147	2.0147	0.2699	0.2283
8	0.497	0.042	0.705	0.0596	1.391	0.6028	0.0256	2.0137	0.3949	0.373
9	0.592	0.0501	0.7694	0.0651	1.3971	0.6516	0.0277	2.0128	0.6548	0.4071
10	0.9673	0.0818	0.9835	0.0832	1.4504	0.9128	0.0369	2.0127	0.9581	0.5204

Table 5 Error components using NARX multistep prediction using Meridional winds as an exogenous input-SSTA_M

W	MSE	NMSE	RMSE	NRMSE	MaxAE	MeanAE	MAPE	SDD	SDE	SRes
1	0.5189	0.0439	0.7203	0.0609	1.2998	0.6207	0.0262	2.0136	0.3948	0.3812
2	0.556	0.047	0.7456	0.0631	1.1213	0.6599	0.028	2.0135	0.375	0.3945
3	0.1605	0.0136	0.4006	0.0339	0.651	0.3547	0.0148	2.0135	0.2744	0.212
4	1.5094	0.1277	1.2286	0.1039	2.0399	0.9521	0.0366	2.0134	0.8387	0.6501
5	0.2624	0.0222	0.5122	0.0433	0.963	0.4577	0.0183	2.0146	0.2484	0.2711
6	0.0767	0.0065	0.2769	0.0234	0.3962	0.2464	0.01	2.015	0.2976	0.1465
7	0.3037	0.0257	0.5511	0.0466	0.8668	0.4967	0.0207	2.0147	0.2578	0.2916
8	0.6785	0.0574	0.8237	0.0697	1.4975	0.7485	0.0317	2.0137	0.3715	0.4359
9	0.7569	0.064	0.87	0.0736	1.487	0.7292	0.0311	2.0128	0.6592	0.4604
10	0.8288	0.0701	0.9104	0.077	1.5577	0.835	0.0339	2.0127	0.936	0.4817

References

1. National Climatic Data Center. National Center for Environmental Information, Homepage https://www.ncdc.noaa.gov. Last accessed on 2019/08/29
2. Pacific Marine Environmental Laboratory Homepage. https://www.pmel.noaa.gov/aboutpmel. Last accessed on 2019/08/29
3. Rasmusson EM, Carpenter TH (1982) Variations in tropical sea surface temperature and surface wind fields associated with the southern oscillation/El Niño. Mon Weather Rev 10:354–384
4. Shea D Climate data guide. https://climatedataguide.ucar.edu. Last accessed on 2019/08/29
5. Trenberth K & National Center for Atmospheric Research Staff (Eds): The climate data guide: Nino SST Indices (Nino 1 + 2, 3, 3.4, 4; ONI and TNI). Retrieved from https://climatedataguide.ucar.edu/climate-data/nino-sst-indices-nino-12-3-34-4-oni-and-tni
6. Hu L, He M (2014) Impacts of sea surface temperature anomaly to the coverage area and early appearance time of green tide in the yellow sea. In: IEEE geoscience and remote sensing symposium. Quebec City, QC, Canada, (13–18 July'14), pp 4465–4468
7. Carbone F, Landis MS, Gencarelli CN, Naccarato A, Sprovieri F, De Simone F, Hedgecock IM, Pirrone N (2016) Sea surface temperature variation linked to elemental mercury concentrations measured on Mauna Loa. Geophys Res Lett 43:7751
8. Mustapha SB, Larouche P, Dubois J-M (2016) Spatial and temporal variability of sea-surface temperature fronts in the coastal Beaufort Sea. Cont Shelf Res 124:141
9. Stock CA, Pegion K, Veechi GA, Alexander MA, Tommasi D, Bond NA, Fratantoni PS, Gudgel RG, Kristiansen T, Brien TD, Xue Y, Yang X (2015) Seasonal Sea surface temperature anomaly prediction for coastal ecosystems. Progr Oceanogr 137:219–236
10. Dinesh Kumar PK, Steeven Paul Y, Muraleedharan KR, Murthy VSN, Preenu PN (2016) Comparison of long-term variability of sea surface temperature in the Arabian Sea and the Bay of Bengal. Reg Stud Mar Sci 3:67–75
11. Cañadas A, Vázquez JA (2017) Common dolphins in the Alboran Sea: facing a reduction in their suitable habitat due to an increase in sea surface temperature. Deep-Sea Research Part II
12. Lins ID, Araujo M, Moura MDC, Silva MA, Droguett EL (2013) Prediction of sea surface temperature in the tropical Atlantic by support vector machines. Comput Statist Data Anal 61:187–198
13. Picone M, Orasi A, Nardone G (2018) Sea surface temperature monitoring in Italian Seas: analysis of longterm trends and short-term dynamics. Measurement 129:260–267
14. Salles R, Mattos P, Iorgulescu AMD, Bezerra E, Lima L, Ogasawara E (2016) Evaluating temporal aggregation for predicting the sea surface temperature of the atlantic ocean. Ecol Inform 36:94–105
15. Park K-A, Lee E-Y, Chang E, Hong S (2015) Spatial and temporal variability of sea surface temperature and warming trends in the Yellow Sea. J Mar Syst 143:24–38
16. Wada A, Usui N, Sato K (2012) Relationship to maximum tropical cyclone intensity to sea surface temperature and tropical cyclone heat potential in the North Pacific Ocean. J Geophys Res 117:D11
17. Ali MM, Jagadeesh PSV, Lin II, Hsu J-Y (2012) A neural network approach to estimate tropical cyclone heat potential in the Indian Ocean. IEEE Geosci Remote Sens Lett 9(6):1114–1117
18. Ali MM, Swain D, Weller RA (2004) Estimation of ocean subsurface thermal structure from surface parameters: a neural network approach. IEEE Geosci Rem Sens Lett 31:20
19. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323:533–536
20. Chen S, Billings SA (1989) Representations of non-linear systems: the NARMAX model. Int J Contr 49(3):1013–1032
21. Du L, Zhang Q, Gao C, Chen H, Yin Q, Dang K, Fu Y, Qu D, Guo F (2018) Response characteristics prediction of surge protective device based on NARX neural network. IEEE Trans Electromagn Compat (2018)

22. Xie H, Tang H, Liao Y-H (2009) Time series prediction based on NARX neural networks : an advanced approach. In: Proceedings of eighth international conference on machine learning and cybernetics, pp 1275–1279
23. Ercan H (2017) Baltic stock market prediction by using NARX. In: Proceedings of CSIT, pp 464–467
24. Jawad M et al (2018) Genetic algorithm based non-linear autoregressive with exogenous inputs neural network short—term and medium-term uncertainty modelling and prediction for electrical load and wind speed. Proc J Eng 8:721–729
25. Lipu MS, Hannan MA, Hussain A, Saad MH, Ayob A, Blaabjerg F (2018) State of charge estimation for lithium–ion battery using recurrent NARX neural network model based lighting search algorithm. In: IEEE access, pp 28150–28161
26. Saha G, Chauhan NC (2017) Numerical weather prediction using nonlinear auto regressive network for the Manaus region, Brazil. In: Proceedings international conference on innovations in power and advanced computing technologies, pp 1–4
27. Saha G, Chauhan NC (2019) Dependency investigation of sea surface temperature on sea bottom temperature and sea surface salinity. In: Accepted and Presented at In Proceedings of international conference on innovations in power and advanced computing technologies'19, March 2019, VIT, Vellore
28. https://archive.ics.uci.edu/ml/index.php. Last accessed on 30th Aug 2019

Chapter 17
Regression Model of Frame Rate Processing Performance for Embedded Systems Devices

Yaroslav Krainyk

1 Introduction

Contemporary Internet-of-Things (IoT) devices [1, 2] are having momentum of a great success in the market and have spread to almost all spheres of information communication technologies applications. Such inquiry causes that IoT devices obtain new functionality, and customers are expecting increased performance and higher service level from them. However, base device constraints should be taken into account under those circumstances. Typically, top-segment devices are oriented on the maximum autonomous work time alongside with high-intensive computational tasks. The primary example of such devices is multimedia-enabled IoT modules [3, 4].

The most common hardware platform for IoT devices with multimedia capabilities is microcontrollers with dedicated modules or system-on-chips (SoC) [5] with integrated processing cores. The last ones may contain graphics processing core and facilitate necessary computational power even for complex scenes. They also include necessary video interfaces. However, at the same time, output of the processed frame to the custom display (not via standard video interface) requires low-level implementation of the procedure. Due to this culprit, achievable frame rate for video depends on the peculiarities of software implementation and interface selected for data transfer.

Another point to mention about the problem is usage of the existing display modules in new systems. High pace of development cycles and shortened time-to-market value cause adoption of the existing components that is not always the best option for performance. Transition from one display to another obviously implies changes in the performance parameters. Thus, mathematical apparatus for performance estimation in terms of video frame rate for multimedia IoT module is required. This will allow the designer to have necessary information to make the decision about component substitution.

Y. Krainyk (✉)
Petro Mohyla Black Sea National University, 10. 68 Desantnykiv str., Mykolaiv, Ukraine
e-mail: yaroslav.krainyk@chmnu.edu.ua

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_17

In this work, we investigate multimedia-enabled IoT device that comprises SoC module and display to devise performance estimation model for such kind of systems.

2 Background and Related Works

Multimedia-enabled IoT devices [6–10] can be applied in different fields of social life and can provide advanced user experience. They are tightly connected with concept of smart environments where suck kind of devices automate numerous routine actions for customers [6, 7].

SoC architectural solution for hardware organization was a breakthrough point in the development of modern computer systems. SoCs offer all main components within a single die and therefore reduce total number of parts in the final product. SoCs have become a hardware platform for single-board computers (SBCs). This type of devices combines decent computational power with low level of energy consumption due to SoC usage. They also are a fundamental part for multimedia-enabled IoT devices. Hence, in this work, we exploit SBC Friendly ARM Nano-Pi Neo Air [11] as the control module for the system.

From the point of view of content flow to the display device, we can distinguish the next two classes of devices:

1. Devices that receive multimedia data from the external source. These devices are not responsible for content processing and just direct them from the input to the output.
2. Content-generating devices that perform content processing directly on the target device.

The performance of the devices that belong to the first type depends on the throughput of the interfaces involved into multimedia transfer. Therefore, it can be estimated quite precisely under different conditions. At the same time, performance of the content-generating device relies on the complexity of the scene and software implementation of the rendering procedure. With this statement in mind, in this paper, we focus on the performance of the devices that belong to the second class and intend to provide mathematical model suitable for productivity estimation under circumstances of software-based content processing.

In opposite to conventional video interfaces, universal interfaces can communicate with almost any device type that has corresponding functionality. However, this feature is achieved at the expense of performance. Performance plays indispensable role for graphics presentation and serves as the major indicator for user experience. Serial Peripheral Interface (SPI) is one of the most widespread communication interfaces and display modules include SPI as one of the possible options for data transmission. In general, SPI can be characterized as synchronous one-bit wide connection. This interface is quite ubiquitous and can be found as a compound part in modern SoCs. In the current work, we assume that display is connected to SBC via SPI.

Linear regression [12–15] is a powerful and simple method for dependency identification between two variables (in classic case). In this paper, we assume that dependent variable is number of frames displayed on the screen per unit of time. The independent variable is tightly coupled with software and will be defined in the main section.

The contribution of the paper is the developed linear regression model for performance estimation of the content-generating multimedia IoT device. The model provides the opportunity to estimate performance in terms of frame rate according to area of graphical objects that are to be shown on the screen.

3 Model of Performance Estimation

Let us provide more details about organization of the system under investigation. The system contains module of the single-board computer Nano-Pi Neo Air and display module that is based on ILI9163 display controller. The display itself has resolution of 128×128 pixels with maximal 18-bit-per-pixel color presentation scheme. This is feasible to consider this testbed as a small IoT device. The interconnection between computer and display is made via SPI. The general scheme of elements connection is demonstrated in Fig. 1.

Speaking of software organization, we assume that a single-thread model of software implementation is employed. Thus, it is guaranteed that processor is occupied by a single processing stage at the moment. The following processing pipeline has been used in the test implementation:

1. Calculations of the new parameters of the scene.
2. Rendering of the scene (this task is submitted to the graphical core).
3. Copying of the processed image into memory region available for processor.
4. Transmission of the data array to the display.

The built-in graphical core is responsible for rendering of a graphical scene. It is easy to observe that during rendering stage, the main processing core is idle. As image data are ready, processor duplicates into memory region and adopts image presentation to the one that is suitable for display.

From the point of view of processor time usage, the last two stages consume the same amount of time without regard of what actually is rendered by the software. The first stage supposes calculation of new geometrical positions, scale factors, and rotation angles. The performance of this stage actually depends on the quantity of the graphical objects and transformation complexity. However, as size of the display is

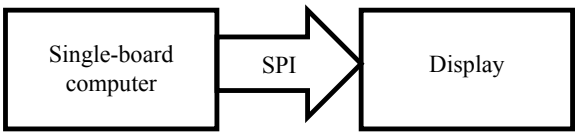


Fig. 1 General scheme of elements connection

relatively small and, hence, number of objects to display is also quite limited, we can assume that it is not much impactful on the performance. Another point to mention is that those calculations for transformation can be performed relatively fast even for dynamic scenes. Therefore, time consumed by this stage can also be deemed as stable interval during different pipeline iterations.

Thus, it brings us to the situation when the second stage is the stage that causes most variations in the pipeline performance. Generation of multimedia content on IoT device is a complex task that demands cooperation of main processing modules. To provide necessary user experience, first of all, it is necessary to assure performance and quality of graphics. This is the reason for selection of OpenGL [16] technology for graphics processing. It effectively harnesses graphical core and is capable of guaranteeing decent performance for this task. In context of user interface and graphics quality, it is worth to mention that textures are the first option to build advanced interface and content with OpenGL. In the further part of the work, it is presumed that graphical objects are represented as textures.

OpenGL employs complex and sophisticated pipeline that allows developers to perform 3D processing of graphics. It includes transformations from object space to image space with several intermediate stages. General flowchart of the OpenGL graphics processing in the user-space application is shown in Fig. 2. The pipeline incorporates numerous parameters to set up the final view of the scene. In this work, the following assumption is put forward. The total time required for processing depends on the area of objects in the rendering process.

In context of the previous statement, the virtual display area can be taken as a conventional unit of area for further investigation. This unit is represented as a final image that is visible according to the objects' positions and camera settings. The typical recommendation for display coordinate space assignment is different and is quite customizable. Hereinafter, coordinate space of the display is assigned to the values $[-1; 1]$ for both dimensions. Which means that if object's corner points are $(-1; -1)$ and $(1; 1)$, then it will cover all area of the display. Object with corner coordinates $(0; 0)$ and $(1; 1)$ occupies only quarter of display area.

Let us assume that at the specific moment, a set of graphical objects $O = \{o_1, o_2, \ldots, o_n\}$, should be displayed on the screen. All the elements of O can be characterized by their areas $S_i, i = \overline{1, n}$ relatively to the display area. The performance of graphical output system is typically expressed in frames-per-second (FPS). The device developer should assess this parameter at the testing stage. The performance should not plummet lower the specific threshold. Additionally, generation of graphical content might imply quite complex dynamic scenes with high quantity of objects where performance can change tremendously from one scene to another. Although, performance of the graphics core can be retrieved from the specification, it is a tedious task and calculation for each scene is not rational. Therefore, the proposed model simplifies assessment of the performance according to the total area of rendered objects in the scene. The model allows approximating performance of the system and formulating basic rules to follow for software responsible for graphics processing. First, we introduce total objects' area parameter that is calculated as

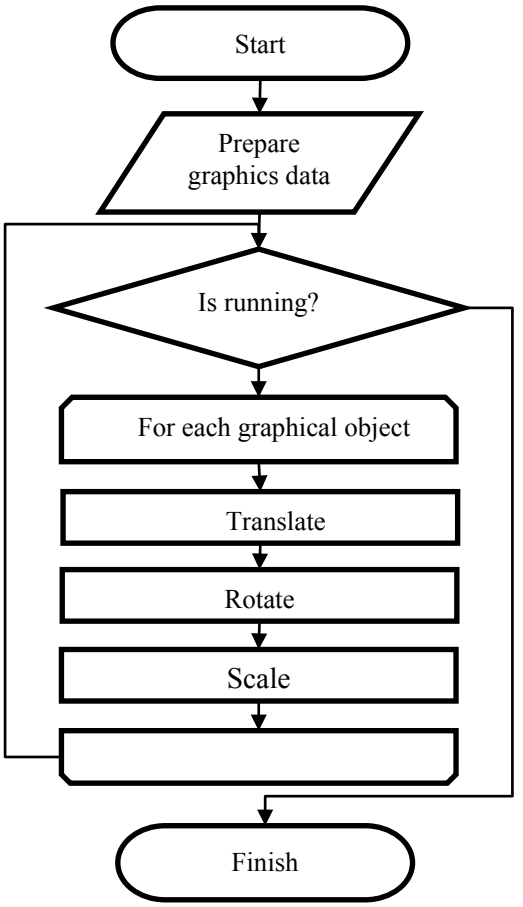


Fig. 2 General organization of graphics processing in OpenGL application

$$\mathrm{Sum}_S = \sum_{i=1}^{n} S_i. \tag{1}$$

Providing that display area is denoted as S_{screen}, the ratio of total objects' area to display area can be defined as a metric

$$M = \frac{\mathrm{Sum}_S}{S_{\mathrm{screen}}}. \tag{2}$$

This metric is used as independent parameter in the model. The idea behind this metric is to identify how many layers of objects can be rendered on the screen with area S_{screen} that is perceived as the single layer of visual content. As graphical applications employ concept of layer extensively, this metric becomes intuitively

understandable. Thus, basically, the rule to guarantee performance during specific scene rendering is that the value of M should be higher than the threshold.

The general representation of the regression model with framerate as dependent variable can be written as follows

$$F = a \cdot \frac{\mathrm{Sum}_S}{S_{\mathrm{screen}}} + b. \tag{3}$$

The peculiarity of the proposed model is that intercept component b actually includes performance expenses at other three stages of content generation. Although the exact values can be calculated separately, previously made assumption that those values are not subject to significant changes allows concealing them in b value. Because screen parameters are indirectly involved into the equation, substitution of the screen will result in different model numerical values.

The software for the testing has been implemented in Java programming language and using LWJGL library [17]. LWJGL implements OpenGL ES standard and can be deployed on the devices with ARM architecture that include Nano Pi Neo Air board. LWJGL can also be considered as an enabler of OpenGL ES functionality for software run by Java virtual machine.

4 Results

To measure processor time spent on one iteration, we employ system functions for measurements of time intervals available in Java. For this purpose, the moment of the iteration start is to be fixed directly before the beginning of the cycle as well as the moment of the finish. Difference of the values stored in the corresponding variable contains information about time interval. Adequate amount of statistical information has been collected during this stage to prove correctness of the selected approach. The statistics includes time data that match to scenes with different values of area metric. We also modeled scenes with different number of objects to render. They have been further converted to FPS parameter. Moreover, the data also includes measurements for consequential dynamic scenes where total area parameter is changing permanently. Traditionally, graphical scene can be divided into two layers, background and foreground. Thus, it can be presumed that background layer should occupy full possible layer area on the display, and total area of foreground objects may vary. Due to this point, the measurements have been taken with one background layer (4 conventional area units) and variable number of foreground objects that consume from 1.5 to 7.25 area units. The higher boundary of this range indicates that almost three layers are to be rendered for the scene.

To build the final regression model, all the obtained data has been have been processed by Python script that employs scikit-learn library [18] as a main tool to calculate coefficients of the model. Matplotlib library has been used to visualize data set and final model for the peculiar hardware configuration. Horizontal axis denotes

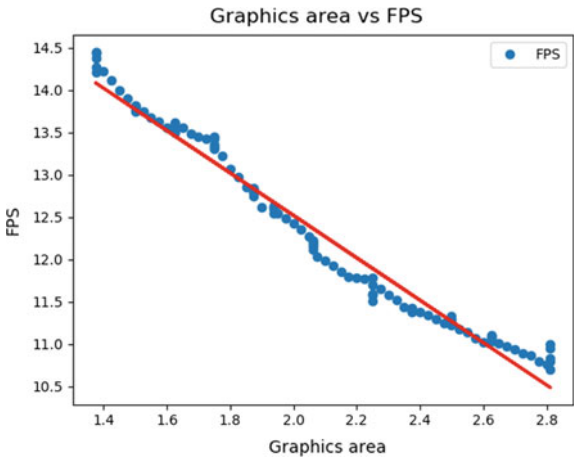


Fig. 3 Visualization of statistics data points and corresponding regression model with calculated coefficients

area of graphical data, and vertical axis defines performance in FPS. The final plot is illustrated in Fig. 3.

The obtained statistics and chart shown in Fig. 3 clearly indicate that usage of linear regression model to estimate performance is adequate. To prove this even further, conventional root mean squared error (RMSE) metric has been calculated for predicted values. With RMSE equals to value 0.177, it can be stated that the model is sufficient to estimate performance of the system.

Moreover, to prove correctness of linear regression selection, coefficient of determination has been computed for the acquired data set according to

$$R^2 = 1 - \frac{\mathrm{SS}_{\mathrm{res}}}{\mathrm{SS}_{\mathrm{tot}}}, \tag{4}$$

where $\mathrm{SS}_{\mathrm{res}}$ denotes residual sum of squares and $\mathrm{SS}_{\mathrm{tot}}$ denotes total sum of squares. The calculated value of the R^2 metric is equal to 0.978 on the selected interval. As this statistics is close to 1, it can be declared that dependent and independent variables have been chosen correctly, and the model fits for prediction of IoT device performance.

It is also easy to observe recommendation about the maximum area of graphical elements in the scene to keep necessary performance mark using the plot. The final presentation of the model is next.

$$F = -2.501 \cdot \frac{\mathrm{Sum}_S}{S_{\mathrm{screen}}} + 17.523. \tag{5}$$

From the practical point of view, plot in Fig. 3 demonstrates that performance of the system under test never gets lower than 10 FPS if total area of rendered objects does not go higher approximately 11 area units.

5 Discussions

The attained results prove that the linear regression model can be applied to estimate performance of content-generating multimedia IoT devices. It can be exploited to declare basic rule about software implementation regarding areas of graphical objects. However, due to possible differences in hardware parts (throughput during the access to SD-card, different display, etc.), retrieved parameters of the model may require calculation and justification to match specific configuration.

The implication of the proposed approach is that it can be used for the system that includes SBC as the main module, and all operations are performed by SoC. However, strict recommendations can be alleged for the approach use. First, SoC architecture is supposed to be a combination of processor core and graphics core that have an access to single memory region. Second, transfer of the generated multimedia content to the display is a responsibility of user-space application. While conventional devices have built-in functionality for multimedia output (video ports, dedicated memory, preinstalled drivers, etc.), the device under consideration transfers data via general-purpose SPI interface. Third, all graphics objects are going through the same processing pipeline at every loop of the scene generation.

The proposed instrument should be used cautiously when applied to other multimedia devices. More advanced architectures depend on much more parameters that should be taken into consideration. Therefore, more sophisticated model should adopted for this class of devices.

6 Conclusions

The proposed regression model aims to provide reliable apparatus for performance estimation for content-generating multimedia IoT-device. Classic linear regression has been employed for this purpose. The results obtained using developed testbed were used for model inference and they proved feasibility of the assumption made in the paper. From the practical point of view, it can be used to ensure that output frame rate will not be lower than it is declared in the product specification. It can be achieved by limiting amount of multimedia information to be processed by the device.

References

1. Gubbi J, Buyya R, Marusic S, Palaniswami M (2013) Internet of things (IoT): a vision, architectural elements, and future directions. Future Gener Comput Syst 29(7):1645–1660. https://doi.org/10.1016/j.future.2013.01.010
2. Miraz MH, Ali M, Excell PS, Picking R (2015) A review on internet of things (IoT), internet of everything (IoE) and internet of nano things (IoNT). In: 2015 internet technologies and

applications (ITA), IEEE, pp 219–224. https://doi.org/10.1109/itecha.2015.7317398
3. Al-Turijan F, Alturijan S (2018) 5G/IoT-enabled UAVs for multimedia delivery in industry-oriented applications. Multimedia Tools Appl 1–22. https://doi.org/10.1007/s11042-018-6288-7
4. Alvi SA, Aflaz B, Shah GA, Atzori L, Mahmood W (2015) Internet of multimedia things: vision and challenges. Ad Hoc Netw 33:87–111. https://doi.org/10.1016/j.adhoc.2015.04.006
5. Conti F, Schilling R, Schiavone PD, Pullini A, Rossi D, Gurkaynak FK, Muehlberghuber M, Gautschi M, Loi I, Hauhou G, Mangard S, Benini L (2017) An IoT endpoint system-on-chip for secure and energy-efficient near-sensor analytics. IEEE Trans Circ Syst I Regul Pap 64(9):2481–2494. https://doi.org/10.1109/TCSI.2017.2698019
6. Chianese A, Piccialli F (2014) Designing a smart museum: when cultural heritage joins IoT. In: 2014 eighth international conference on next generation mobile apps, services and technologies, IEEE, pp 300–306. https://doi.org/10.1109/ngmast.2014.21
7. Sivanathan A, Sherratt D, Gharakheili HH, Radford A, Wijenayke C, Vishwanath A, Sivarman V (2017) Characterizing and classifying IoT traffic in smart cities and campuses. In: 2017 IEEE conference on computer communications workshops (INFOCOM WKSHPS), IEEE, pp 559–564. https://doi.org/10.1109/infcomw.2017.8116438
8. Wan J, Al-awlaqi M, Li M, O'Grady M, Gu X, Wang J, Cao N (2018) Wearable IoT enabled real-time health monitoring system. EURASIP J Wirel Commun Netw. https://doi.org/10.1186/s13638-018-1308-x
9. Patel S, Park H, Bonato P, Chan L, Rodgers M (2012) A review of wearable sensors and systems with application in rehabilitation. J NeuroEng Rehabil 9:21. https://doi.org/10.1186/1743-0003-9-21
10. Albagory Y, Nofal M, Al Raddadym F (2018) IoT-RTP and IoT-RTCP: adaptive protocols for multimedia transmission over internet of things environments. IEEE Access 5:16757–16773. https://doi.org/10.1109/ACCESS.2017.2726902
11. NanoPi Neo Air. https://www.friendlyarm.com/index.php?route=product/product&product_id=151. Last accessed 2019/08/03
12. Yang Y, Sun L, Guo C (2018) Aero-material consumption prediction based on linear regression model. Procedia Comput Sci 131:825–831. https://doi.org/10.1016/j.procs.2018.04.271
13. Permai SD, Tanty H (2018) Linear regression model using bayesian approach for energy performance of residential building. Procedia Comput Sci 135:671–677. https://doi.org/10.1016/j.procs.2018.08.219
14. Austin PC, Steyerberg EW (2015) The number of subjects per variable required in linear regression analyses. J Clin Epidemiol 68(6):627–636. https://doi.org/10.1016/j.jclinepi.2014.12.014
15. Ibrahim S, Daut I, Irwan YM, Irwanto M, Gomesh N, Farhana Z (2012) Linear regression model in estimating solar radiation in Perlis. Energy Procedia 18:1402–1412. https://doi.org/10.1016/j.egypro.2012.05.156
16. Shreiner D, Sellers G, Kessenich J, Licea-Kane B (2013) OpenGL programming guide: the official guide to learning OpenGL, Version 4.3. Addison-Wesley
17. LWJGL—Lightweight Java Game Library. https://www.lwjgl.org/. Last accessed 2019/08/05
18. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830

Chapter 18
Time Series Data Representation and Dimensionality Reduction Techniques

Anshul Sharma, Abhinav Kumar, Anil Kumar Pandey, and Rishav Singh

1 Introduction

A time series can simply be understood as an ordered sequence of numbers, representing data collected over a regular time interval which are usually obtained from different application domains like scientific, health care, industries, financial domains, etc. In concordance with researches in the area of data mining, time series data mining methods help in certain essential tasks. These tasks can be categorized as clustering, classification, indexing, motif discovery, and rules discovery. While performing these tasks, different approaches need to be followed. Representation and similarity measures are key elements of time series data mining which eventually assist to achieve greater efficiency and effectiveness in information processing.

Essentially, time series is a wide spectrum high-dimensional data. The processing and storage of this kind of data in their raw format are very tedious, cumbersome as well as expensive. This leads to the importance of developing relevant representation techniques with the aim of reducing the high dimensionality and also preserving the fundamental characteristics of the data. This has to be done with a certain amount of precision and acumen because time series data is not like canonical data (data with

A. Sharma (✉) · A. Kumar
Department of Computer Science and Engineering, Indian Institute of Technology (Banaras Hindu University), Varanasi 221005, India
e-mail: anshul.rs.cse16@iitbhu.ac.in

A. Kumar
e-mail: abhinav.rs.cse17@iitbhu.ac.in

A. K. Pandey
Computer Centre, Banaras Hindu University, Varanasi 221005, India
e-mail: akpandey@gmail.com

R. Singh
Department of Computer Science Engineering, NIT Delhi, Delhi, India
e-mail: rishav.singh@nitdelhi.ac.in

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_18

nominal/categorical or ordinal variables) with straightforward distance definition between values. This is desirable for querying, clustering, classification, and other mining tasks if we are looking for similarity-based retrieval in time series data.

The way, representation is selected, depends on how appropriate it is to impact the ease and effectiveness of time series data mining task to obtain information and knowledge that is relevant and valuable. The problem of the high dimensionality of information needs to be resolved because it adversely affects the outcomes of time series data mining, as an inverse proportionality relationship exists between query precision and dimensional effectiveness. There are various methods suggested in the literature to achieve reduced dimensionality in time series data representation [1–10]. These representation techniques can be characterized as data adaptive and non-data adaptive [1]. In methods, that are data adaptive, the transformation parameters rely on information such as APCA [2], while the transformation parameters do not alter in non-data adaptive methods, or in other words, the parameter does not rely on information such as PAA [1]. In the literature, various kinds of representations for time series are presented [1–4]; some of them are specific to particular domain application, while others are general to be used in any application. The challenges of indexing time series for fast retrieval have been addressed in the majority of works, but this chapter is more inclined towards those time series representation methods that are more appropriate for classification/clustering.

In the literature, many variations of the primary method are proposed, but here, we have considered only basic and fundamental methods. The systematic workflow used in the chapter is summarized in the following way: Sect. 2 represents the proposed taxonomy for representation and dimensionality reduction. Section 3 introduces the piecewise linear methods, which include five methods. Section 4 describes the symbolic-based methods, which include three fundamentals symbolic methods. In Sect. 5, feature-based methods are described that are specially designed for classification/clustering. Section 6 includes transformation-based methods that are used in diversified field of applications. Section 7 discusses the cutting edge and the latest techniques for time series classification. Lastly, Sect. 8 provides a conclusion and directions for future research.

2 A Taxonomy of Representation/Dimensionality Reduction Techniques

As stated earlier, the taxonomy that we suggest is categorized in the representation and dimensionality reduction processes intended for time series data along with an inclination towards classification and clustering tasks. However, these representations are not restricted to this scope; they can also be used for other time series data mining endeavours. The proposed taxonomy is presented in Fig. 1 and divides the methods into four categories.

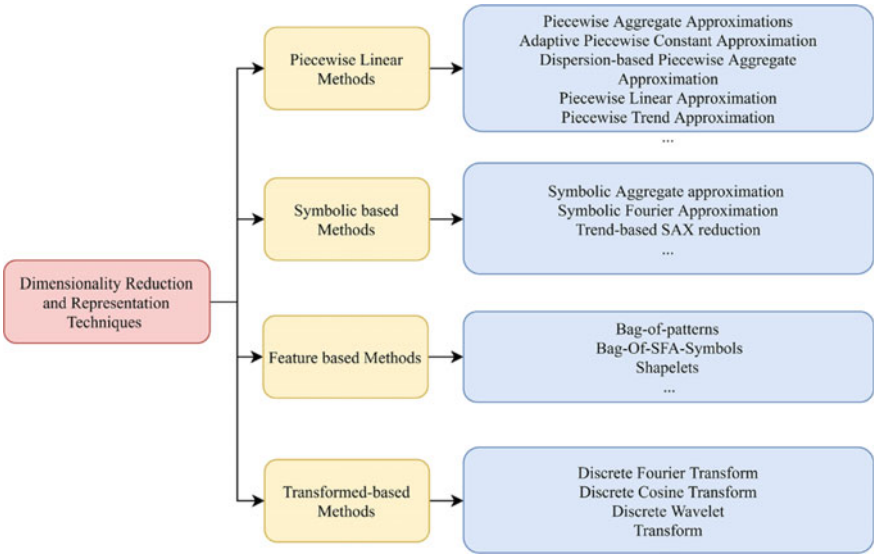


Fig. 1 Taxonomy of time series representation and dimensionality reduction techniques

- *Piecewise linear methods*: in this, the time series is usually divided into different segments, and approximation of these segments is taken to represent the series. These methods include piecewise aggregate approximation (PAA) [3], adaptive piecewise constant approximation (APCA) [2], piecewise linear approximation (PLA) [5], dispersion-based piecewise aggregate approximation (DPAA) [6], and piecewise trend approximation (PTA) [7].
- *Symbolic-based methods* represent the time series into a symbolic format, and they provide a higher level of an approximation than piecewise linear methods; these include symbolic aggregate approximation (SAX) [4], symbolic Fourier approximation (SFA) [8], and trend-based SAX reduction [9].
- *Feature-based methods* first learn the unique features set from the training data and then represent the time series in terms of these features. These methods include Bag of patterns (BOP) [10], Bag-Of-SFA-Symbols (BOSS) [11], and shapelets [12].
- *Transformation-based methods* considered the time series as signal and convert it into other domains such as frequency or time–frequency and extract the informative information that is hidden in the time domain. These methods include discrete Fourier transformation (DFT) [13], discrete cosine transformation (DCT) [14], and discrete wavelet transformation (DWT) [4, 14].

3 Piecewise Linear Methods

If we depict the time series in different ways, we can achieve a high level of reduced dimensionality, i.e. by reducing the sample size of raw data. Sampling is one of the easiest ways to represent time series. A rate of p/q is used in this strategy, where q represents a segment of raw time series and p is the selected sample points from the segment. However, it has a drawback; if the sampling rate is too small, it distorts the shape of the sampled time series. In this section, the piecewise linear model for time series representation is taken into consideration, which usually splits the time series into equal length and provides the relevant information from them.

3.1 PAA

PAA aims at dimensionality reduction by taking the mean of frames which are of equal size. If $X = \{x_1, x_2, \ldots, x_p\}$ is considered a time series whose length is p, the PAA of X can be calculated by dividing X into q number of segments of equal lengths of size p/q, where $q \ll p$. In this transformation, each segment of the raw time series is represented by the mean of the value of its elements.

$$\overline{X} = \frac{q}{p} \sum_{j=p/q(i-1)+1}^{(p/q)} x_j \tag{1}$$

Figure 2 illustrates the notion of PAA. It ensures the division of the original series into correct size of frames. Moreover, two specific cases of this approach are noticed. When p is equal to q, then we found no difference between transformed representation and original time series, or we can say both are an identical representation. If q is equivalent to 1, then the mean value of the original sequence is obtained from the transformed representation. In general, it can be said that a piecewise constant approximation of the initial sequence is produced by this transformation.

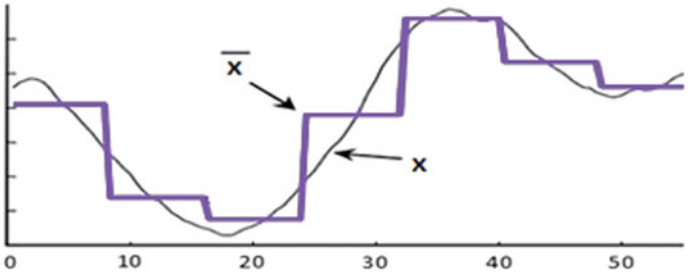


Fig. 2 X is the raw time series and $\overline{X}$ is the PPA representation of X

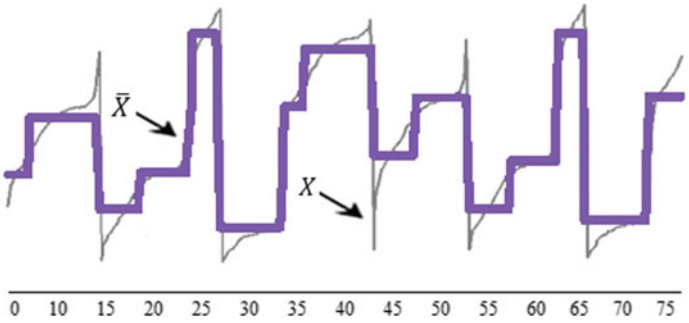


Fig. 3 X is the original time series and $\overline{X}$ is the APCA representation of X

3.2 APCA

APCA can be considered as an extended version of PAA. To attain the APCA representation, the time series is stratified into K pieces of unequal length based on its size. Here, low activities are represented by long segments, and region of high activities is represented by short segments. The average value and the index of the segment's right endpoint are used here to depict each segment in the sequence. Hence, the time series $X = \{x_1, x_2, x_3, \ldots, x_p\}$ is represented as $\{\bar{x}_1, x_{r1}, \bar{x}_2, x_{r2}, \ldots, \bar{x}_k, x_{rk}, \ldots, \bar{x}_m, x_{rq}\}$, where $\bar{x}_k$ is the elements' mean value in the kth segment and x_{rk} is the index of the rightmost element of the kth segment. As shown in Fig. 3, segments of series are of different lengths.

3.3 DPAA

DPAA can be understood as another form of PAA. DPAA splits a series into sequences of same length. Further, it records the mean value and standard deviation for each segment. The primary distinction between PAA and DPAA is that it takes into account both central tendency and dispersion, which is available in each subsequence. For a given time series $X = \{x_1, x_2, x_3, \ldots, x_p\}$, the DPAA representation of X can be expressed formally by $\overline{X} = (\bar{x}_1, \sigma_{x,1}), (\bar{x}_2, \sigma_{x,2}), \ldots, (\bar{x}_k, \sigma_{x,q})$ where

$$\bar{x}_i = \sum_{j=(i-1)\frac{p}{q}+1}^{i\frac{p}{q}} \frac{x_j}{\left(\frac{p}{q}\right)} \tag{2}$$

$$\sigma_{x,i} = \sqrt{\sum_{j=(i-1)\frac{p}{q}+1}^{i\frac{p}{q}} \frac{\left(x_j - \bar{x}_i\right)^2}{\left(\frac{p}{q}\right)}} \tag{3}$$

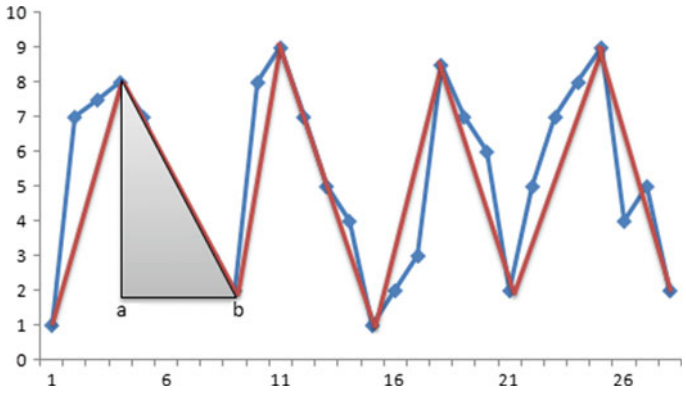


Fig. 4 Linear fit between two endpoints (**a**) and (**b**) of a segment

We can call these two statistical measures [defined in Eqs. (2) and (3)] as features. In the above expressions, p denotes the length of original time series and q represents the number of segments. It is assumed that the term $\frac{p}{q}$ should be an integer value that represents the number of points in each segment. Therefore, the original time series should be padded with suitable numbers of zeros before the transformation. Finally, the reduced dimensionality of time series is $2q$ because each segment carries two feature values.

3.4 PLA

PLA representation of time series is the most widely used method. In PLA, we partition a time series X of size p into q parts ($q \ll p$), as we do in PAA, but in this case, each segment is represented by a straight line. Equal length segments, e.g. $l = p/q$ is produced in uniform segmentation, whereas in non-uniform segmentation, X is partitioned into parts of unequal length so that they can best fit the shape of the time series. In a further process, linear interpolation is used to approximate the segment $X_{a:b}$ by joining a line between x_a and x_b. Here, x_a and x_b are the end points of the segment as shown in Fig. 4.

3.5 PTA

Local trends of time series data are used as a base for PTA representation. First of all, this process involves the transformation of raw data into local trends (ratios). Secondly, it performs segmentation to discriminate time series into distinct segments

of different trends. Finally, the ratios of the first and the last data points are taken from each segment for approximation.

For any given time series $X = \{(x_1, t_1), (x_2, t_2), \ldots, (x_p, t_p)\}$, where x_i is a real value and t_i is the time stamp, the PTA representation of X is presented by the following expression:

$$\overline{X} = \{(r_1, r_{t_1}), (r_2, r_{t_2}), \ldots, (r_q, r_{t_q})\}, q \leq p, \tag{4}$$

where r_{t_i} is the rightmost element of the ith segment, $r_i (1 < i \leq q)$ is the proportion between $r_{t_{i-1}}$ and r_{t_i} in ith segment, r_1 is the proportion between t_1 and r_{t_1}, ith segment length is computed using $r_{t_i} - r_{t_{i-1}}$.

PTA uses the piecewise discontinuous function to obtain dimensionality reduction of time series. This algorithm contains three major steps:

- *Local trend transformation*: data points of the new transformed series are obtained by computing the ratio between any two successive data point of the original series.
- *Segmentation*: local trends in the time series are transformed by division into segments of variable length in a way that the different trends are represented by two conjunctive segments.
- *Segment approximation*: the ratio of the leftmost and rightmost data points inside the segment is used to represent each segment that indicates the trend characteristic.

4 Symbolic-Based Methods

The past decades have seen the introduction of many symbolic representations of time series. This method basically uses pre-designated mapping rules to convert the time series with a numerical form into a sequential series of discrete symbols. Symbolic methods are more adaptive to noise. SAX is the most widely accepted technique of symbolic representation on time series data mining. This representation method guarantees a significant reduction in dimensionality and a reduced bounding property, which in turn improves the efficiency of algorithms.

4.1 SAX

SAX can be considered as one of the most competent, pertinent, and competitive approaches for the dimensionality reduction in time series. In this, a set of symbols are used to represent a time series of length p into q where $q \ll p$. First, we normalize the time series so that its mean becomes zero and the standard deviation converted into unity. Then, the time series is partitioned into an appropriate number of segments with the sequence of q real values to compute a PAA representation of the

Table 1 A look-up table that contains the brake points for alphabets between 3 and 8

v	3	4	5	6	7	8
β_1	−0.043	−0.67	−0.84	−0.97	−1.07	−1.15
β_2	0.43	0	−0.25	−0.43	−0.57	−0.67
β_3		0.67	0.25	0	−0.18	−0.32
β_4			0.84	0.43	0.18	0
β_5				0.97	0.57	0.32
β_6					1.07	0.67
β_7						1.15

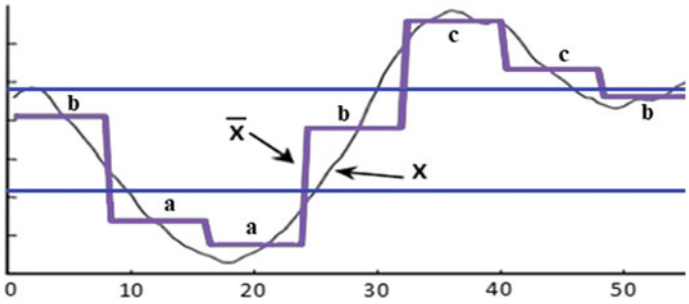


Fig. 5 SAX representation: firstly, time series is discretized using PAA; after that, PAA coefficient is mapped with SAX symbols. In the example, the series is represented by **baabccb**

normalized series. Further, the discretization process assigns the discrete symbols from alphabet v of size l to these real numbers. This process usually determines breakpoints resulting in the assigned symbols of v occurring with equal probabilities in the representation; this is shown in Table 1. As the time series forms Gaussian distribution after normalization, the breakpoints $\beta_1, \beta_2, \beta_3, \ldots, \beta_v$ are determined from the statistical table in such a way that the area under the normal curve $\mathbf{N}(0, 1)$ from β_k to β_{k+1} is $1/\boldsymbol{l}$, for all k. The SAX representation of time series X is shown in Fig. 5.

One slightly modified version of SAX is extended symbolic aggregate approximation [15]. In this approximation, three symbols are used for each segment, in which the first symbol represents minimum value, the second symbol represents mean value, and the third symbol represents the maximum values of the segment.

4.2 SFA

The SFA representation of a time series is performed by the use of a set of symbols, (called SFA word). These alphabets of symbols are of a finite set. The transformation

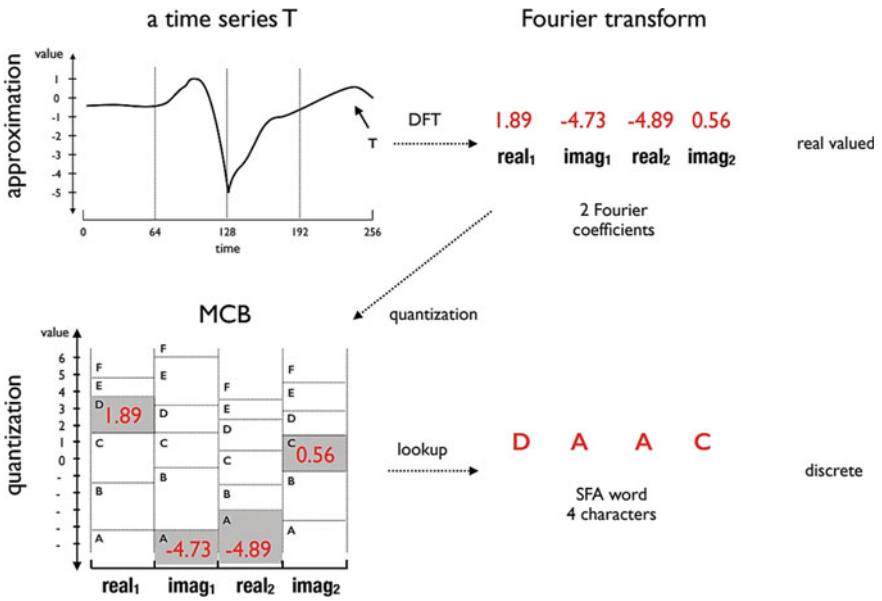


Fig. 6 SFA representation of time series T, approximated by DFT and discretized using MCB [16]

process of a time series can be generalized in two parts from numerical to symbolic representation: (i) *approximation* and (ii) *quantization*. Approximation maps a time series, with reduced dimensionality into lower dimensional space and produces a vector, representing real values. Discretization maps assign each real value to a symbolically interpreted discrete value.

As said above, SFA is constituted of a couple of operations, as shown in Fig. 6. First, approximation is performed with the use of DFT, configured by the SFA word length $L \in N$. Quantization is performed with the use of Multiple Coefficient Binning (MCB) technique; this is configured by alphabet size $C \in N$ equal to the number of bins.

The aim of the approximation process is to represent a signal of length n by a reduced length signal of length L. The uses of Fourier higher-order coefficients are often connected with time series where there are rapid changes in a signal such as noise or dropouts. The first $L \leq n$ Fourier coefficients are used for low-pass filtering of the signal. In the frequency domain, quantization helps in noise reduction by separating the series into frequency bins, and each bin is represented by a Fourier coefficient. Mainly, the quantization is performed by Multiple Coefficient Binning that helps in determining equi-depth bins which in turn help in mapping each imaginary as well as a real part of the Fourier coefficients to symbols. Further, a separate histogram is built for imaginary as well as for real part in MCB using all training samples, and then, partitioning of histograms is done with the use of equi-depth binning.

In Fig. 6, SFA transformation is illustrated at the top right where DFT is used to transform a time series and gets a real-valued vector (1.89; −4.73; −4.89; 0.56). Here, the MCB bins are used to quantize the vector to the SFA word DAAC.

4.3 Trend-Based SAX Reduction

This method follows a trend-based SAX representation approach which is also termed as SAX_CP. In this approach, abrupt change points and variations in data are taken as a base to capture the trend information, presented in the time series. In SAX_CP, discrimination between two time series is done on the basis of adaptation of a segment size that is variable, relying on the change points in time series. The representation of SAX_CP covers the following steps:

- *Abrupt change point's detection*: here, *trend change* points in the time series are considered as the *change points*. They have to be detected for the identification of the sizes of different segments that are used in the SAX_CP representation. In this approach, the segment size is not constant but variable; this, in turn, makes the representation and classification of sequences in a time series far more challenging.
- *Trend computation*: the trends are identified based on differences between segments and its mean. In this process, a specific difference (variations) is generated by each segment which is coded in a binary string.
- *Sequences generation*: SAX symbols are generated, and each one of them is explained and marked with the trend, and the sum of the segments points variations with respect to the segment mean.

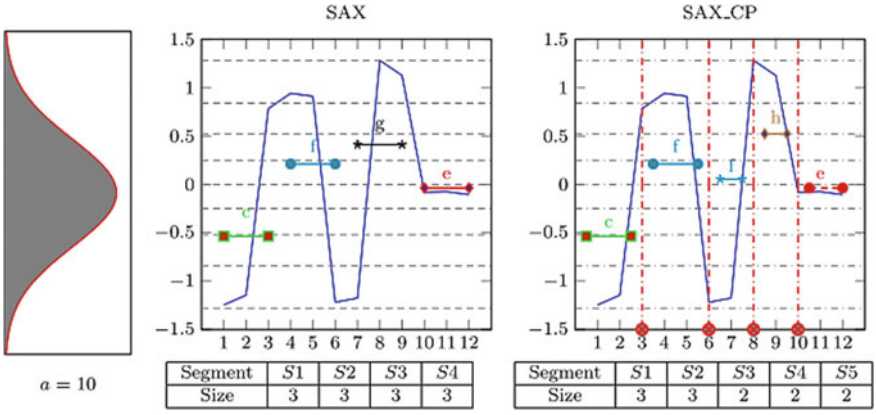


Fig. 7 Segment representation for SAX and SAX_CP for the time series *S* [9]

As shown in Fig. 7, SAX has fixed length size window, while SAX_CP has variable size window. SAX_CP also considers the change point information; therefore, symbol sequence for SAX and SAX_CP is different as *cfge* and *cffhe*, respectively.

5 Feature-Based Methods

These methods are specifically defined for classification or clustering task. In this approach, features are extracted from the dataset such as symbolic sequence or shapelets. Further, the time series is represented by these features. Feature-based representation of time series is more robust to time invariant. In the next subsection, we explain the three feature-based techniques, which include (BOP) [10], (BOSS) [11], and shapelets [12].

5.1 BOP

BOP is a representation technique similar to a bag of words (BOW) for time series data based on the histogram. BOW is a widely used representation technique in the area of text mining and information retrieval. This approach is more appropriate for high-dimensional time series where similarities, based on higher-level structures, need to be considered. The core feature used by this approach is SAX.

In this approach, firstly, the "vocabulary" patterns are constructed from a time series dataset. A sliding window approach is used to accomplish this task; it extracts subsequences from the time series of length q. Each of these subsequences is then given a mean zero through normalization with unity as the standard deviation. Subsequently, these subsequences are converted to a SAX string. Thus, a set of strings was obtained in which each string corresponds to a time series subsequence.

Whenever any subsequence P_i is taken, the likelihood of P_{i-1} and P_{i+1} being similar to its immediate neighbouring subsequences is very high, if P_i happens to be in the smooth region of the time series. Consequently, a number of consecutive subsequences can be mapped to the same string; subsequences thus mapped are known as trivial matches of P_i. They are sometimes counted as true independent patterns; in order to avoid this over-counting, only the first occurrence of the string is taken into account for recording, and the rest are ignored until a significantly different string is found. It may be concluded that only the first occurrence is taken into account for each group of consecutive identical strings, and the group of occurrences is counted as one. Let us assume, for example, that the sliding window technique is used to obtain the sequence of SAX string:

$$P = \text{aab aab abb abb abb bab bab aba aba} \ldots$$

With the option of numerosity reduction option detailed above, the above-mentioned sequence would be recorded as $P =$ aab abb abb bab aba instead.

Once the string set is obtained for each time series, then the transformed series will contain the word frequency for the words in the BOP. For example, SAX word sequence S for a given time series is represented as:

Aaa	aab	aba	abb	baa	bab	bba	bbb
0	2	2	2	0	2	0	0

5.2 *BOSS*

The BOSS model follows an analogy similar to the BOP model, but this model uses SFA strings to represent every time series as a set of substructures that is unordered. The systematic workflow of the BOSS model is presented in Fig. 8, which follows the three-step processes: (i) a sliding window is used to convert time series into subsequences; (ii) these subsequences are converted into SFA words; (iii) reduction of numerosity and representation of histograms are used to depict time series.

BOSS model is defined by the following three parameters: (i) the length of the substructures, represented by the window size w; (ii) mean normalization, i.e. flag $\in$ [False, True]: set to true for offset invariance, and (iii) SFA with two parameters S, L for the size of the alphabet and word length, respectively. These parameters are used for string representation as well as low-pass filtering. Figure 9 demonstrates the complete process for BOSS representation for a given sample.

When we extract length w sliding windows from a time series, it is intuitive that w approximately represents the length of substructures inside the time series. Now, every sliding window is given a standard deviation of "unity". The mean becomes a parameter for this model for the purpose of obtaining offset invariance, and it determines whether or not we need to subtract the mean value from each sliding window. Finally, using the BOSS model, a transformation of the time series into

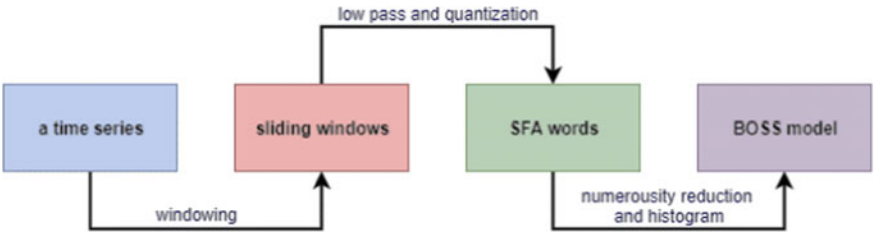


Fig. 8 Workflow of the BOSS representation

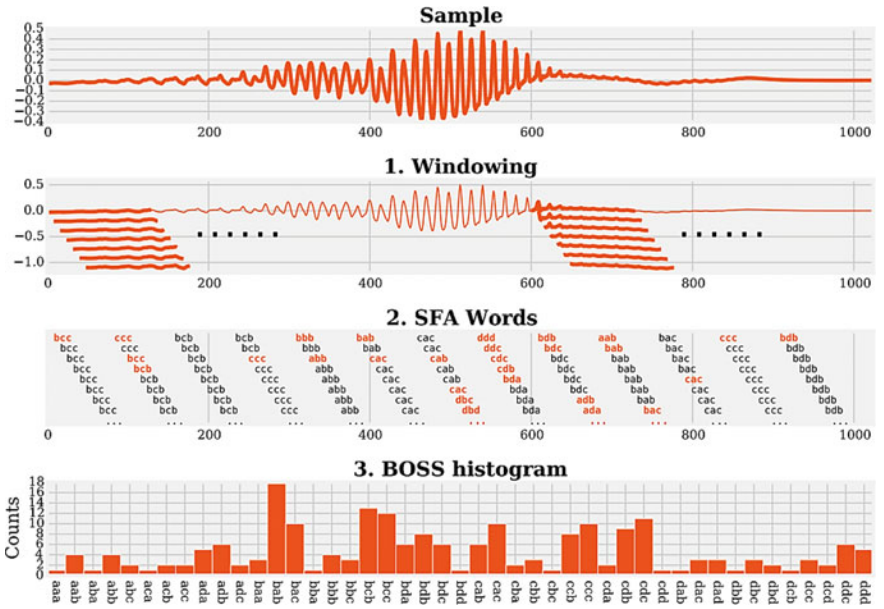


Fig. 9 Example of BOSS representation for a sample time series [16]

an unordered set of SFA words is achieved by applying the SFA transformation to each sliding window. Using an unordered set guarantees a phase-shift invariance, i.e. invariance to each substructure's horizontal alignment within the time series. It is extremely likely that the SFA words for two adjacent sliding windows are same in stable parts of the signal. Thus, numerosity reduction can be applied to avoid weighing out stable sections of the signal, i.e. only the first occurrence of the SFA word is recorded with all subsequent identical duplicates being ignored unless we find a new SFA word. As we can see from Fig. 9, the first SFA words are the same; the only *bcc* is used and the remainders are ignored.

The following are the advantages of this approach:

- High speed of execution, as hashing is being used for the purpose of measuring the similarity of SFA words.
- Adaptability to noise.
- Invariance to amplitudes, occlusions, offsets, and phase shifts.

5.3 Shapelets

Shapelets are the subsequences of the time series that can be recognized as a representative of class membership [17]. In a recent study, shapelets are used as features

and represent the time series by transforming it into a distance vector of shapelets, also known as shapelet conversion. First of all, the best informative shapelets are obtained from the dataset in this strategy, and then, we use these shapelets to convert information. This is achieved by calculating the distances to each shapelet from a sequence. This depiction of the time series captures the local class-specific data that helps to improve classification/clustering effectiveness.

6 Transformed-Based Methods

By these methods, time series is transformed from one domain to another domain like frequency to time–frequency which gives a better and informative representation of data that is difficult to capture in the time domain. These representations greatly help in dimensionality reduction while preserving the salient characteristics/features of data. In the following subsections, we explain the well-known traditional methods for reduction of dimensionality, which includes DFT, DCT, and DWT. These methods achieve time series' dimensionality reduction by considering the few high informative coefficients of the Fourier transforms, cosine transforms, or wavelet transforms. Hence, few coefficients adequately represent the time series in transform domain.

6.1 DFT

DFT is originally applied on time series for dimensionality reduction. This transformation decomposes a signal into Fourier series, i.e. a summation of series sine and cosines. Moreover, a specific complex number represents each series, namely the Fourier coefficient. Compression of data is the most valuable feature of this method, as we can reconstruct the original signal from transformed signal because high-valued Fourier coefficient carries more information and discarding low Fourier coefficient has no significant loss.

More formally, for a given signal $\boldsymbol{x} = \{x_1, x_2, x_3, \ldots, x_p\}$ of length p, DFT of $\boldsymbol{x}$ is represented by $\mathbf{X} = \{X_1, X_2, X_3, \ldots, X_p\}$ where X_k is a complex number for $k \leq p$. Here, $\boldsymbol{x}$ represent the signal in the time domain and $\mathbf{X}$ represent the signal in the frequency domain. The each X_k component of $\mathbf{X}$ is defined as:

$$X_k = \frac{1}{\sqrt{p}} \sum_{j=1}^{p} x_j \mathrm{e}^{\frac{i2\pi}{p} jk}, \quad i = \sqrt{-1}, k = 1, 2, 3, \ldots, p \tag{5}$$

The original representation of $\boldsymbol{x}$ can be retrieved by the following inverse function:

$$x_j = \frac{1}{\sqrt{p}} \sum_{k=1}^{p} X_k \mathrm{e}^{\frac{i2\pi}{p} jk}, \quad i = \sqrt{-1}, j = 1, 2, 3, \ldots, p \tag{6}$$

If we want to reduce the dimensionality of $\boldsymbol{x}$ from length p to q, then only $q/2$ coefficient of transform vector is needed to be stored because each coefficient in transform space is a complex number. Thus, only one dimension is required to be stored for the coefficients' real as well as imaginary parts.

6.2 DCT

DCT is also a Fourier-based transform similar to DFT, but the difference is that it uses only cosine functions and leaves out the sines. DCT transform the signal from the time domain to the frequency domain by highlighting the periodicity of the signal. Formally, DCT for a time series $\mathbf{x}$ of length p is defined as:

$$X_k = \sqrt{\frac{2}{p}} \sum_{j=1}^{p} x_j \cos \frac{\pi k(j - 0.5)}{p}, \quad k = 1, 2, 3, \ldots, p \tag{7}$$

6.3 DWT

Wavelet transform (WT) is one of the techniques used for signal decomposition. It represents data as a sum of functions or as a difference of functions. DWT is simply a version of WT that is discrete. It shows similarity with DFT in certain aspect such as in DWT, wavelet coefficients give the local contribution in signal reconstruction, while the contribution of Fourier coefficients to the signal is always global. This makes DWT useful for analysing the data with multi-resolution. The application of this method is defined for sequences where lengths are the power of two.

The *Haar* wavelet is considered to be wavelet in the simplest possible form. Its exact definition is provided in [18]. In a time series $\boldsymbol{x}$ of length $\boldsymbol{p}$, *Haar* transform $H_L(\boldsymbol{x})$ is defined as:

$$A_{l+1}(k) = \frac{A_l(2k) + A_l(2k + 1)}{2}, \tag{8}$$

$$D_{l+1}(i) = \frac{D_l(2k) - D_l(2k + 1)}{2}, \tag{9}$$

$$H_l(N) = (A_l, D_l, D_{l-1}, D_{l-2}, \ldots, D_0) \tag{10}$$

where $l \in [0, L]$ and $k \in [1, n]$.

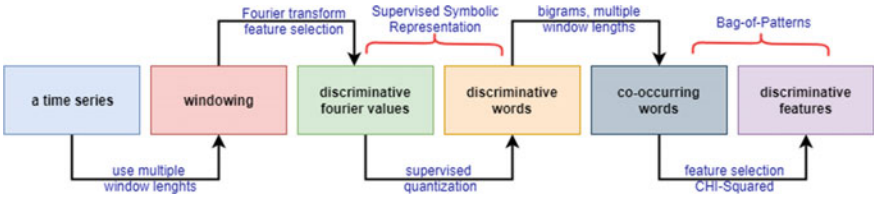


Fig. 10 Workflow of time series representation using WEASEL

7 State of the Art

The current trends and state-of-the-art representation methods for time series, concerning classification/clustering framework, include symbolic methods such as *trend feature symbolic aggregate approximation* (*TFSAX*) [19], SAX_CP [9], feature-based methods such as *word extraction for time series classification* (WEASEL) [20], *bag of recurrence patterns* (BOR) [21].

TFSAX is a modified version of SAX. In this method, dimensionality is reduced by utilizing the PAA approach, and then, the average value of each segment is discretized by SAX; now the trend feature in each segment is extracted by the use of trend distance factor and trend shape factor. Finally, trend information is discretized into symbols by designing appropriate multi-resolution symbolic mapping rules.

WEASEL represents the time series similar to a bag of pattern that extracts the discrete features from each window, sliding over the time series. However, in this approach, the filtering process of features is dferent, as depicted in Fig. 10.

BOR representation method utilizes the visualization techniques known as recurrence plots (RP) under the BoF model for classification of time series. In this approach, the transformation of time series into a 2D texture image is done by using primary RP, followed by the application of BoF approach for representing the time series.

8 Conclusion and Future Research Direction

In this chapter, we provided an overview of both basic and recent trends in time series representation and dimensional reduction in the context of data mining, which are more preferably applied in time series classification. We also discussed representation methods which include piecewise linear methods, symbolic-based methods, transformation-based methods, and feature-based methods. In the growing generation of sequence data, there is a need to develop new methodologies and tools to analyse complex time series data more accurately and efficiently. The representation of time series is the heart of these techniques to take the benefit from them. Time series representation also addresses the problem of noisy data and the curse of dimensionality. On the basis of our observations, we conclude that the current

research direction for time series representation is moving towards symbolic-based and feature-based which are providing a good approximation of time series. Moreover, 2D-based representation of time series is also a good research direction from deep learning point of view.

References

1. Lin J, Keogh E, Wei L, Lonardi S (2007) Experiencing SAX: a novel symbolic representation of time series. Data Min Knowl Disc 15(2):107–144
2. Keogh E, Chakrabarti K, Pazzani M, Mehrotra S (2001) Locally adaptive dimensionality reduction for indexing large time series databases. ACM Sigmod Record 30(2):151–162
3. Keogh E, Chakrabarti K, Pazzani M, Mehrotra S (2001) Dimensionality reduction for fast similarity search in large time series databases. Knowl Inf Syst 3(3):263–286
4. Chan KP, Fu AWC (1999) Efficient time series matching by wavelets. In: Proceedings of the 15th IEEE Int'l conference on data engineering, Sydney, Australia, 23–26 Mar 1999, pp 126–133
5. Shatkay H, Zdonik SB (1996) Approximate queries and representations for large data sequences. In: Proceedings of the twelfth international conference on data engineering, IEEE, pp 536–545
6. Karamitopoulos L, Evangelidis G (2009) A dispersion-based PAA representation for time series. WRI World Congr Comput Sci Inf Eng 4:490–494
7. Dan J, Shi W, Dong F, Hirota K (2013) Piecewise trend approximation: a ratio-based time series representation. In: Abstract and Applied Analysis, vol 2013. Hindawi
8. Schäfer P, Högqvist M (2012) SFA: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In: Proceedings of the 15th international conference on extending database technology, ACM, pp 516–527
9. Yahyaoui H, Al-Daihani R (2019) A novel trend based SAX reduction technique for time series. Expert Syst Appl 130:113–123
10. Lin J, Khade R, Li Y (2012) Rotation-invariant similarity in time series using bag-of-patterns representation. J Intell Inf Syst 39(2):287–315
11. Schäfer Patrick (2015) The BOSS is concerned with time series classification in the presence of noise. Data Min Knowl Disc 29(6):1505–1530
12. Hills J, Lines J, Baranauskas E, Mapp J, Bagnall A (2014) Classification of time series by Shapelet transformation. Data Min Knowl Disc 28(4):851–881
13. Agrawal R, Faloutsos C, Swami A (1993) Efficient similarity search in sequence databases. In: Proceedings of the 4th conference on foundations of data organization and algorithms, pp 69–84
14. Batal I, Hauskrecht M (2009) A supervised time series feature extraction technique using DCT and DWT. In: International conference on machine learning and applications, IEEE, pp 735–739
15. Lkhagva B, Suzuki Y, Kawagoe K (2006) New time series data representation ESAX for financial applications. In: 22nd international conference on data engineering workshops (ICDEW'06), IEEE, pp x115–x115
16. Schafer P (2015) Scalable time series similarity search for data analytics, Ph.d. Thesis, Humboldt University, Berlin
17. Ye L, Keogh E (2009) Time series shapelets: a new primitive for data mining. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 947–956
18. Chan K-P, Fu AWC (1999) Efficient time series matching by wavelets. In: Proceedings 15th international conference on data engineering, IEEE, pp 126–133

19. Yu Y, Zhu Y, Wan D, Liu H, Zhao Q (2019) A novel symbolic aggregate approximation for time series. In: International conference on ubiquitous information management and communication, Springer, Cham, pp 805–822
20. Schäfer P, Leser U (2017) Fast and accurate time series classification with weasel. In: Proceedings of the ACM on conference on information and knowledge management, pp 637–646
21. Hatami N, Gavet Y, Debayle J (2019) Bag of recurrence patterns representation for time-series classification. Patt Anal Appl 22(3):877–887

Chapter 19
Simultaneous Localization and Mapping with Gaussian Technique

Sai Prabanjan Kumar Kalvapalli and C. Mala

1 Introduction

Estimating the properties of the world from observations of the sensors over time and learning from their prior experience form the core of statistical or probabilistic robotics. Statistical robotics is a nascent area to the field of robotics that considers valid data along with the noisy data in the robot perception and action using statistical and probabilistic approaches.

The need for estimation and learning arises from the fact that real world is abundant with the presence of uncertainty, and the design of probabilistic algorithms should estimate in such a scenario. The probabilistic algorithms contain training part and testing part, where during the training part, the robots learn the appropriate parameters that enable them to complete the task that is given to them accurately and efficiently in the presence of uncertainty. The testing part involves cross-validating the learnt parameters of the previously existing models, and any error crept in will be minimized iteratively.

The subtle difference between estimation and learning can be better explained by formally defining the respective terms. Estimation involves predicting some aspects of the state of the world from noisy, incomplete and uncertain data, whereas learning involves improving the robot performance iteratively by comparing the model parameters with the parameters of the prior experience under the presence of uncertainty.

Uncertainty and incomplete data are the key reason for the deployment of estimation and learning in the field of robotics. The sources of uncertainty in robotics can be listed as

S. P. K. Kalvapalli (✉) · C. Mala
NIT Trichy, Trichy, Tamilnadu 620015, India
e-mail: saiprabanjan@gmail.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_19

- Sensor noise—sensors providing inaccurate information
- Lack of knowledge about the environment—objects may be hidden from you which are not possible to perceive
- Dynamic changes in robot motion and environment—objects in environment may be moving over time and robot cannot move where they are precise.

There are several ways of handling the uncertainty in robotics, and among them, probabilistic modelling and machine learning are the popular ones. Probabilistic modelling uses various types of probability distributions for handling the uncertainty, whereas machine learning predicts the future uncertain world by learning from the previous experience.

2 Statistical Estimation and Learning in Robotics

Statistical learning or probabilistic modelling uses various types of probability distributions to predict the future uncertainty in the environment of the robot. The most commonly used probability distribution over real numbers is the normal distribution also known as the Gaussian distribution [1].

Gaussian distributions are used in various forms to learn and estimate from the data, and the most popular among them are single-dimensional Gaussian model, multivariate Gaussian model and Gaussian mixture model. All the Gaussian models have a single binding property in spite of their implicit differences that make it easy for performing estimation and maximum likelihood estimation on the data, and the property is parameters of the models.

Single-dimensional Gaussian model is a canonical model to measure and express uncertainty in the probability distributions [2]. Multivariate Gaussians are the generalization higher to higher dimensions of Gaussians and their mean, co-variance properties can be studied. Gaussian mixture models are the combinations of Gaussians to model more complex distributions.

For statistical modelling in robotics, Gaussian distributions are a natural choice for modelling noise and uncertainty. The properties of Gaussian distribution that make it more suitable among the many probability distributions can be listed as

- Only two parameters mainly mean and variance are enough to specify any Gaussian model like single-dimensional, multivariate and mixture models.
- Gaussian distributions have good mathematical properties that make it suitable for performing complex analytical calculations without going out of domain. One of such properties is multiplication operation over Gaussian distribution that is closed.
- Central limit theorem is another mathematical property that makes Gaussian distribution suitable for statistical modelling of robotics, which states that the expectation of the mean of any random variable converges to Gaussian.

3 Simultaneous Localization and Mapping [SLAM]

Bayesian filtering methods are used for estimating the true state of the robot from the noisy measurements. Bayesian modelling uses Gaussian models for calculating the mean and co-variance of the state dynamics model X_t, and uses probabilistic inference for predicting future states from noisy measurements.

Occupancy grid mapping algorithm is used for mapping based on range measurements, and it can be converted to localization by using a three-dimensional version of it. A map for a robot is a spatial model of its environment unlike the metric maps that are used in day-to-day life.

Map representation depends upon two conditions, namely available sensors and the purpose of mapping, and the purpose of mapping can be either for world coordinate system or for the robot coordinate system. The occupancy grid mapping algorithm can be used for SLAM application only in the robot coordinate system [3].

The challenges for simultaneous localization and mapping task can be listed as

- Noisy measurement in robot coordinate system
- Converting measurements from robot coordinate system to global coordinate system
- Related to other robotic problems like planning and navigation
- Environment changes over time.

To make the coordinate transformation from robot coordinate to world coordinate measurement, it is required for having good estimation techniques and probabilistic methods, and occupancy grid mapping is one such algorithm which is called occupancy grid mapping (OGM) algorithm.

OGM views world as a grid of cells where each cell contains the occupancy value provided by the range sensor. OGM algorithm requires Bayesian filtering algorithm to maintain an occupancy grid map that means recursive update of probabilistic notion of occupancy rather than a definite one. Given the range sensor measurements for the occupancy values, the updating of the cell value will be a recursive update of probabilistic model calculated on the measurements rather than definite ones.

For updating, occupancy probability of each cell is done by calculating the respective measurements in a Bayesian framework. Keeping track of each cell and updating it using Bayesian framework is a computationally intensive technique, so a new method called odds of a cell occupied technique is utilized.

4 Kalman Filter, Extended Kalman Filter

Kalman Filters follow a notion called notion of uncertainty which considers all measurements are noisy and the robot's decision should not be based on single measurement [4]. Finding the state of the robot in a noisy environment is as precise to the true state of the robot in the world. Delineating the state and the measurements is

a viable option of achieving the goal stated, as measurements can only give a shadow of the true world. And Kalman Filters provide the stated functionality successfully.

Kalman filter has two important factors that underline its functionality, namely dynamical systems and measurements model. The mathematical model of the dynamical systems and probabilities help to model motion and noise, as providing inspiration robot tracking task is considered as illustration for Kalman filter functionality.

Dynamical system of model is required for modelling a robot in movement unlike a robot identification task where probability model is enough. Dynamical system is helpful in describing how the state of the object changes in time as well as how the robot measures the state.

In the linear model, the state X will be indexed by the time steps t. The state vector contains two terms, v and $\frac{\mathrm{d}v}{\mathrm{d}t}$. V is the position term calculated in metres, and the velocity term $\frac{\mathrm{d}v}{\mathrm{d}t}$ is measured in metres per second. Dynamics is the reason that state changes in each unit time step, from the current time step 't' to the next time step $t + 1$.

The transition in a time difference is captured by the state transition matrix, and state transition matrix helps in describing the common transition in formal form for a robot in a unit time step. Transition matrix simplifies the mathematical calculations of the state metrics to be dependent upon the previous state by dividing the time unit-wise.

At any given time because of the inherent character of the nature, the sensors of the robot cannot observe the true state of the robot but only a shadow of it. They are mathematically modelled as:

$$X_{t+1} = AX_t + Bu_t \tag{1}$$

$$Z = CX_u \tag{2}$$

where U term represents nonlinear influence on the robot model that is not dependent on the state 'X', which is called as state term.

Even in linear model, both X and Z contain noise as the state term X cannot capture all possible physical interactions. Observation term Z is also noisy as sensors contain noise in their measurements.

Particle filter is a probabilistic state estimation technique using a sampling-based distribution representation. Instead of a fully defined function, the particle filter represents a distribution with a set of samples named as particles, and these samples represent the distribution.

The statistics of the samples match the statistics of the distribution such as the mean and standard deviation, and also, there is a scope for more complicated metrics. As with Gaussian models, the parameters such as mean and co-variance are not present.

Instead, a full population is tracked, as the particle filter population represents a mixture of Gaussian distributions [5]. But, the variance of Gaussian filters is taken as zero which makes the Gaussian filters become Dirac delta functions.

Initial group of particles represents the underlying distribution of the belief state, and each particle is comprised of a pair of the pose and the weight of the pose. Higher the weight of the pose, more is its priority in the underlying probability distribution.

Odometry Update:

Is useful in moving the particles based on odometer information, similar to the Kalman filter, a motion model will move the underlying distribution, where the particles move based on the odometer measurements taken from the robot.

A correlated uncertainty model captures the noise underlying the motion model such as wheel slip or friction change. The motion model is not tracked in particle filter, and in explicit parameters, a sample noise is added from the motion noise model.

As an illustration, a Gaussian distribution to model noise with '0' mean and nonzero co-variance is taken. Noise is uniquely added to each particle and separate samples are made for each particle, after adding the noise in such a way, the data contains the dispersion of the particles such that it captures the uncertainty due to motion. To constrain the noise and belief distribution similar to Kalman filter, a set of observations can be considered.

Kalman Filter:

$$\text{Prediction step: } X = Ax_{k-1} + Bu_{k-1}$$
$$P = APA^{\mathrm{T}} + Q$$

$$\text{Update step: } K = pH^{\mathrm{T}}\left(H_p H^{\mathrm{T}} + R\right)^{-1}$$

Extended Kalman Filter:

$$\text{Prediction step: } X_u = X + K(Z - HK)P$$

$$\begin{aligned}\text{Update step: } X_u &= (1 - KH)_p \\ \widehat{X_K} &= g(X_{k-1}, U_K) \\ \widehat{P_K} &= G_k P_{k-1} C_k^{\mathrm{T}} + Q_k \\ K &= \left(\widehat{P_K}\right) C_K^{\mathrm{T}} \left(C_K \widehat{P_K} C_K^{\mathrm{T}} + R_K\right)^{-1} \\ \widehat{X'_K} &= \widehat{X_K} + K\left(Z_K - C\left(\widehat{X_K}\right)\right) \\ P_K^{'} &= P_K - KC_K P_K\end{aligned}$$

where X represents the state of the robot, u represents the actions, P is the confidence matrix, Q is co-variance of the noise, K is the updated matrix, H is the sensor model, R is co-variance of the noise useful for sensor fusion, Z is the measurements returned from the sensors and I is the identity matrix.

Robot Motion Model in 2D:

$$\text{State } X = \begin{pmatrix} X_x & Y_X & \gamma_k \end{pmatrix}^{\mathrm{T}}$$

$$\text{Odometry } u = \left(\dot{X}_u \ \dot{Y}_u \ \dot{\gamma}_u\right)^{\mathrm{T}}$$

Motion Model:

$$\begin{aligned} G(x,u) = x + (\cos(\gamma)\dot{x} - \sin(\gamma)\dot{y})\Delta t \\ y + (\sin(\gamma)\dot{x} + \cos(\gamma)\dot{y})\Delta t \\ \gamma + \dot{\gamma}\Delta t \end{aligned}$$

Extended Kalman filter for SLAM Pseudo-Code:

EKF_SLAM(μ_{t-1}, ε_{t-1}, μ_t,Z_t)

1: $\mu_t = g(u_t, \mu_{t-1})$

2: $\varepsilon_t = G_t \varepsilon_{t-1} G_t^{t} + R_t$

3: $k_t = \varepsilon_t H^{t}(H_{tt}\varepsilon_t H^{t}{}_{t} + Q_t)^{-1}$

4: $\mu_t = \mu_t + K_t (Z_t - h(\mu_t))$

5: $\varepsilon_t = (I - K_t H_t) \varepsilon_t$

6: return μ_t, ε_t

5 Future Work

The extended Kalman filters and particle filters are utilized for handling nonlinearity of the robot environment and uncertainty in the modelling of robots and their environments. But, they linearize the noise independently using mathematical models like Taylor series and add later to the linear estimations made using statistical techniques. Instead, noise along with the ground truth data needs to be modelled accurately and consistently by the future statistical algorithms for robot modelling.

References

1. Schulz E, Speekenbrink M, Krause A (2018) A tutorial on Gaussian process regression: modelling, exploring, and exploiting functions. J Math Psychol 85:1–16
2. Pulido M, van Leeuwen PJ (2018) Kernel embedding of maps for Bayesian inference: the variational mapping particle filter. In: EGU general assembly conference abstracts, vol 20
3. Sen D, Thiery AH, Jasra A (2018) On coupling particle filter trajectories. Statist Comput 28(2):461–475

4. Roumeliotis SI, Mourikis AI (2018) Extended Kalman filter for 3d localization and vision-aided inertial navigation. U.S. Patent Application No. 15/706,149
5. Ljung L (1979) Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems. IEEE Trans Autom Contr 24(1):36–50

Chapter 20
Unsupervised Learning of the Sequences of Adulthood Transition Trajectories

Jayanta Deb and Tapan Kumar Chakrabarty

1 Introduction

Over the last two decades, extracting patterns from a large complex real dataset has been the focus in several fields of scientific research to arrive at strategic decisions within a reasonable time frame and optimum cost. To that end, in place of rule-based approach of pattern recognition or clustering, the unsupervised machine learning is becoming increasingly popular method. For problems where patterns are unknown and time variant or for which the dataset at hand does not contain sufficient labels, unsupervised machine learning which is a subset of artificial intelligence provides us plausible as well as significant outcomes. Instead of being guided by labels, unsupervised machine learning works by learning the underlying structure of the data, which means learning intermediate concepts, features or latent variables that are useful to capture the statistical dependencies that we need to care about. Unsupervised machine learning can be defined as a program or algorithm that is capable of capturing all of the possible dependencies between all of the observed events in a dataset with minimum or no additional support.

Many surveys nowadays collect data which are simply longitudinal records of when events occurred for an individual or a sample of individuals. Information on the history of event occurrences, timings and their types, timing of transition from one particular event to another is collected and recorded systematically termed as event history data. Such data are characterized by the chronology of the event outcomes. Thus, a list of outcomes for all events of a person is known as event sequence. Notably, even though the total quantity of events is relatively small, huge number of diverse sequences are achievable.

J. Deb · T. K. Chakrabarty (✉)
Department of Statistics, North-Eastern Hill University, Shillong, India
e-mail: tapankumarchakrabarty@gmail.com

J. Deb
e-mail: jdeb888@gmail.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_20

In recent years, one of the many research study areas that has become increasingly popular is life course research. In life course studies, researchers are particularly interested in studying big events which outlines an individuals' living, for instance, schooling, entry into workforce, entry into marital union, entry into parenthood, etc. The study of demographic behaviour in life course research hence uniquely characterized by an overall perspective. Nevertheless, the techniques used so far do not permit such an universal prospect.

In demographic life course research, the investigation of change in adulthood events and the pattern of the sequences of the adulthood events are considered priority areas in many countries. Here, the primary purpose is to investigate the timing of events and secondly, sequencing of events. Also occasionally, the amount of certain events that generally occur in the course of early adulthood. These events which occur during the adulthood are typically considered as an index of transition from puberty to adulthood. In order to make things simpler, the measure of timing is considered as the age, at which, the events are experienced. The audited series of events is considered as an index of sequencing. Also, total count of observed events is treated as the index of quantum. When someone focuses on adulthood transition, the main problem is that, throughout the lifetime, whether the person has experienced an event or not. Moreover, experiences of different events at the same time-point create a critical problem, in sequencing.

For visualizing and summarizing, these types of event history data, sequence analysis in machine learning represent a potentially effective mechanism. It examines such data structures more easily and decreases the complexities. It provides us comparatively simpler descriptive technique that perhaps efficiently applied in practice. Especially, this method can construct the principal constituent or basis for further modelling because this provides a technique to figure out the basic sequences that may assist in subsequent investigations. Using sequence analysis within the events is an innovative approach in this scenario. It helps us in identifying most usual and atypical event sequences. Moreover, it provides us the information on how people behaving in adulthood and family formation trajectories.

Only sequencing of the specific events is not adequate in a study on the transition to adulthood and family formation. However, this method gives an extensively used structure to investigate the determinants through distinguishing the genders, place of residence, religion, caste, different societies, etc. It also helps us in examining the dynamics of behavioural changes in a group of individuals. Therefore, in the study of transition to adulthood and family formation, the timing, sequencing, quantum and clustering should be considered as a benchmark for the analysis.

Nowadays, a collection of statistical techniques broadly recognized as cluster analysis comprises one of the prominent mechanisms in life course research. The purpose of the life course analysis is to focus on the causal relationship of the formation of individual trajectories. In addition to providing the ideal types of trajectories and exploration tools, it allows researchers to explore the complexities of life courses in an adequate way. Without particularly considering the order of events, several methods have been suggested to group individuals in different states at different time points.

In this study, the technique of sequence analysis is put in an application to the data obtained from a national-level survey entitled 'Youth Study in India: situation and needs 2006–2007' [34] conducted by IIPS Mumbai and Population Council, New Delhi, during the year 2006–2007, to primarily find out the dominating principal patterns which effect the adulthood and family formation structures in India. Secondly, cluster analysis is performed for finding out ideal types of adulthood and family formation trajectories which are homogeneous in nature, i.e. to group individuals with respect to their states at various time points.

The rest of the paper is structured as follows. In Sect. 2, we introduce a brief literature review on unsupervised learning of sequences. In Sect. 3, we present the dataset and methods, and Sect. 4 uses some descriptive methods to visualize the fundamental characteristics of the adulthood transitions in India. Section 5 brings in the sequence analysis based on optimal matching and gives a brief discussion on the results of the respective analyses. Finally, Sect. 6 contains some of the concluding remarks.

2 Literature Review on Unsupervised Learning of Sequences

In statistics and machine learning, the problem of forming representations of a dataset without any explicit training is called 'unsupervised learning'. Many of the machine learning rules allow to perform analyses equivalent to standard statistical techniques. Sequencing the biological and social processes and clustering of these sequences thereafter have attracted the attention of researchers in the present days. The sequence analysis which is a subset of unsupervised machine learning was originally developed in biology for the analysis of DNA, RNA or peptide sequences to understand the evolution, features and typologies of various biological processes. Methodologies such as discrete Markov models, sequence alignment and optimal matching-based clustering are used to explore how the biology of an organism from which the new sequence are generated in different from those in existing biological databases.

2.1 Sequences of Biological Processes

Classifying various biological trajectories is a principal problem in biological sciences. Techniques involved in the field of machine learning, such as hidden Markov models, support vector machines and clustering algorithms, make an effort to classify these kind of biological trajectories such as trajectories of moving keratocyte cells. In biology, the scarcity of predictive models for the core processes such as, quantitative evaluation and learning from the data, creates a space for developing new knowledge.

In sequencing and classification of new biological data, an application of unsupervised learning techniques principally deals with three major goals [41]. Firstly, if

someone wants to study further about the biological or biochemical processes behind the observed phenomenon by identifying the parameters in the observation that are significantly influenced by a certain change in experimental conditions. Secondly, the information contents of a given data set with respect to a certain property of interest can be estimated. Third, automatic identification and classification of vast amounts of experimental data could facilitate the process of interpretation.

Sequencing and clustering are two of the basic fundamental problems occur in biological sciences. In biology, sequence analysis is often used for refraining from expensive as well as prolonged experiments. As an example, it is frequently used for the identification of the sequences and activities of ribonucleic acid, deoxyribonucleic acid and proteins. Thus, the particulars regarding the framework as well as activity of sequences previously established through experiments are passed onto the most similar sequences.

One of the tools of unsupervised machine learning which is highly used to identify this similarity between two sequences in biology is optimal matching (OM). Optimal matching approach is primarily formed in computer sciences. Later on, for determining the identical biological sequences, the method is widely used and expanded in biological sciences. For analysing the sequences in biology, this approach is applied and linked through cost settings. Generally, identifying the similarities between output sequences obtained from an experiment, and newly introduced sequences is the principal purpose of an analysis in biology. But, in the field of computational biology, the major challenges are to analyse the significant similarities and differences in their genomic sequences between previously diverse species, rate of mutational changes, natural selection and genetic drift [14]. As a result, the cost of substitution certainly shows the evolutionary preferences of some particular evolutions as compared to rest of the evolutions. The higher probability of connectedness between two sequences based on some phylogenetic hypothesis is generally indicated by a lower 'substitution cost' obtained from two states. Consequently, biologists are to focus more on 'substitution matrices' rather than enquiring about the estimation of probabilities. Thus, to estimate these probabilities based on some phylogenetic hypothesis, an ideal collection of confirmed alignment is to be constructed by biologists [29]. Moreover, when there are protein sequences, a quite complex operation in practice is needed. Constituting matrices based on evolutionary distances in protein sequences requires considerable work, and the use of sequence analysis is an essential step.

Several methods such as K-means clustering, hierarchical clustering, density-based scan clustering, Gaussian clustering model were applied to the task of identifying ideal number of clusters of the biological trajectories of living cells. Recently, a Convallis learning rule for unsupervised learning is discovered which allows a recurrent network of spiking excitatory and inhibitory neurons to develop selective representations of spoken digits [47]. An interactive bioconductor visual application for quality control, filtering and trimming of FASTQ files was done with the help of sequence analysis [40]. A rapid sorting algorithm was developed to arrange unordered genomic fragments into a sequence representing distance from the causative mutation [35]. Hidden Markov models are used for analysing biological sequences [14]. Sequence analysis was used to determine the base recognition positions of zinc fingers

by Jacobs [26] in molecular biology. The epidemiology of tick-borne encephalitis virus was investigated by comparative sequence analysis of virus strains isolated in endemic areas of Europe and Asia [15]. Sequence analysis of the L1 metallo–lactamase from *Xanthomonas maltophilia* shows a significant variation from that of the CphA and Blm metallo–lactamases of *Aeromonas hydrophila* and *Bacillus cereus* [43]. Isolation and sequence analysis of an immunologically active thymic polypeptide was considered by Goldstein et al. [23]. The sequence determination of the entire genome of the Synechocystis sp. strain PCC6803 was completed by Kaneko et al. [27]. The pig genome [7] is being sequenced and characterized under the auspices of the Swine Genome Sequencing Consortium.

The method of sequence analysis followed by optimal matching not only carried out as an usual procedure in the field of biology, however, it is used in discovering novel resemblance within a fresh collection of observation, in a highly complex as well as an exceptional pathway. Not only biologists successfully used this statistical method for their faced problems, rather the use of these theories in social processes is not far behind.

2.2 Social Processes

Unlike the analysis of biological sequences, the quantitative analysis of social processes deals with more complicated sequences of interconnected events along with complex social trajectories. These types of social and demographic transitions, for example, transition from leaving study to entry into workforce, from workforce to marital union, from marital union to parenthood, etc., could be observed as complex processes [9]. Moreover, demographic events are highly consistent, and as a result, the period around is 'demographically dense' [37]. Such processes are also associated with related events. These events are treated as indicators of entry into social roles and are seen as symbolic for adults [33]. Eventually, this kind of a situation obviously gives an indication for adopting a life course aspect.

Life course approach has been developing since the mid-1960s. Four chief elements were identified [22] as fundamentally shaping life courses: development of the individual, history and culture, social relations and the intersection of age, period and cohort. In life course research, each individual trajectory itself formed by social events. Therefore, if we consider the process of to be an adult, as a trajectory, then it will be seen that this trajectory is formed by a number of social and family formation events.

The main issue in life course approach is then the proper selection of events from a given longitudinal record. Thus, in recent years, the life course research has given an increasing attention on event histories [10, 12, 31, 45]. It is naturally seen as a productive method in the way in which life course research stream provides the ultimate explanation of life events [31]. Examining the mutual interdependencies and intersection between simultaneous careers of individuals is possible through event history analysis. Secondly, one can analyse the impact of variables on individual

behaviour [12]. But, each and every issue which occurs during a life course research cannot be addressed only through event history analysis. It depends on the timing of specific events in the life course [46]. Event history analysis fails to provide a common outline of life courses as it concentrates only on time-to-event. Thus, it cannot visualize life courses as a significantly conceptual and meaningful unit, and hence, it misleads us to adopt the life courses as an holistic perspective.

In studying life course trajectories, sequence analysis has become a fundamental as well as holistic strategy, ever since Abbott [2] introduced it in social processes. In social processes, sequence analysis is used as 'fishing for patterns' [3]. It means that this method considers the complication of the sequences and focuses only on the events. Exploring dissimilarities between the pairs of social sequences through distance measure is the principal objective of sequence analysis. Larger distance between two sequences indicates that the amount of dissimilarity between them is higher and vice-versa [20]. Therefore, the first decision is to choose an appropriate distance metric. Among the two main classes of distance metrics, edit-based metric measures the distance between two sequences by counting the minimum number of edit operations needed to turn one sequence into a perfect copy of the another and is broadly known as optimal matching [4]. But, due to the insensitiveness of edit-based metrics to the differences in the ordering of event states [18], motivation in the development of so-called subsequence-based metrics has taken place [16, 19]. By calculating the number of common subsequences, these metrics quantify the distance between a pair of sequences [39, 42]. Since in modern societies, the social transition types differ comparatively a little in the ordering of events [11], optimal matching becomes a suitable choice in these studies of social processes [1, 13, 38]. In the next step, sequence analysis takes into account the distance matrix to divide these sequences into ideal number of homogeneous clusters. Among numerous clustering algorithms, Ward hierarchical clustering method is most widely used in social processes as Wards method [44] iteratively merges ever-bigger clusters of sequences such that, in each iteration, the increase of the total within cluster distance is minimized [24].

From the above discussion, finally, we observe that sequences in social processes are made of social events. This scenario signifies that, in this social and demographic processes, the 'timing' as well as 'coding of events' are extremely critical. In order to represent this hierarchy of social events, sequencing and clustering are adequately appropriate. Sequences differ in social processes based on the timing of events. As a consequence, determining the alikeness between the sequences through sequence manipulation indicates that time manipulation is principal in clustering. The method of recognizing the subsequences of uniformly coded events in a social process is time consuming because of longer duration of the process. However, the duration can be reduced for a rapid identification of the subsequences, through deleting or inserting an event. In other ways, an event is approximated by another possible event, which is defined as 'substitution', and in this way, it preserves the timing of the process. In a nutshell, the 'events' are preserved in 'insertion and deletion' activities irrespective of time, but the 'time' is conserved in 'substitution' method without taking care of the changes in events. Thus, matching of an event's almost indistinguishable

subsequences is carried out through 'clustering' and 'optimal matching' which is an integration of accelerations/decelerations of time as well as an approximations of events [29]. But topical link among the sequences ruins because of warping time when they occupy identical time scale. Consequently, not only the timing of events but a proper sequence analysis with the sequences of social events is of great importance.

3 Data and Methods

Young people are the future of a nation, and contribution to their development is important but critical. The Government of India has proposed several national-level programmes like the National Youth Policy, the National Population Policy 2000, the National Rural Health Mission, etc., to address the multiple needs of young people. However, there has been a lack of comprehensive evidence on young people's situation and needs. The report [36] of the Youth in India: Situation and Needs study produces an extensive overview of youths in the country and offers a roadmap for programmes and priorities that aim to address youth needs. Thus, the project entitled Youth in India: Situation and Needs is a resourceful project intended to provide this evidence, and the data from this project are used and analysed briefly in this paper.

The household-level data from the study 'Youth in India: Situation and Needs 2006–2007' [34] have been used in this article. Being implemented by the International Institute for Population Sciences, Mumbai and the Population Council, New Delhi, it is the first-ever subnationally representative study that provides data on young people's transition to various adulthood events. Survey was conducted in a total of six geographical regions (states) in India. Names of the geographical regions are Andhra Pradesh, Bihar, Jharkhand, Maharashtra, Rajasthan and Tamil Nadu. These six states were purposively selected to represent the different geographic and sociocultural regions within the country. Altogether, they represent two-fifths of the country's population. It also provides a wealth of evidence on married and unmarried young women and men (aged 15–24 and 15–29) from both rural and urban settings of each state. Surveys were undertaken in a phased manner and took place between January 2006 and April 2008. In all, 58,728 young people were contacted, of which a total of 50,848 married and unmarried young women and men were successfully interviewed. Using the information on profile of surveyed youths, the present article has made an attempt to identify potential sequences of adulthood events and homogeneous clusters, accounting for such transitions.

Youth study data provide an enormous amount of information on almost every major dimension of youth life: education, workforce participation, family life, sexual activity, marriage, health and civic participation. The analysis sample includes all households with some of the selected dimension. These are educational attainment, workforce participation and marriage. Among these key variables of interest here,

first two were recorded by the survey as a binary variable: 'yes' or 'no'. Marriage was recorded as a five-categorical variable: 'currently married', 'married but no gauna', 'separated', 'widowed' and 'never married'.

3.1 Method

Sequence analysis is highly intertwined with event history analysis as it provides the ability to model entire life paths or event history trajectories. A typical sequence analysis consists of five key stages, namely describing key sequences via aggregated measures, visualizing key sequences via sequence index plots, comparing sequences via optimal matching, grouping sequences into clusters and associating patterns with other variables within regression models.

Sequence Plots Sequence plots generally show the distribution of sequences. Sequences are formed of categorical data. Let there be an ordered series of l components which are picked out sequentially from a finite set A and be called as state sequence having length l. This is explicitly defined as an alphabet where the size of the set A is $a = |A|$. Let us consider, there are l successive elements namely $x_1, x_2, \ldots, x_l$, then the sequence x is represented as $x = (x_1, x_2, \ldots, x_l)$, where $x_j \in A$. By two features, these state sequences are characterized. Primarily, these sequences are developed by the 'states'. At second, the location of each component provides significant information in respect of elapsed time (days, period, age, etc.). Another effective form of defining a 'state sequence' is through creating an ordered series of specific sequence states with their respective time periods. Hence, a sequence of the pair (x_j, t_j) is generated, where x_j is a state and t_j is its duration. Many other compact forms are available to represent the sequence data like DSS, etc, and are described in [21].

Transition Rate Next, the transition rate gives us an impressive information about the series of sequences. The probability to switch at a given position from state s_i to state s_j is termed as 'transition rate' between a pair of states (s_i, s_j). Consider $n_t(s_i)$ as the total count of sequences at position t with state s_i that does not terminate in t. Also, let $n_{t,t+1}(s_i, s_j)$ is the total count of sequences at position t with state s_i and at position $t + 1$ with state s_j. Then, the transition rate $p(s_j|s_i)$ between states s_i and s_j is obtained as

$$p(s_j|s_i) = \frac{\sum_{t=1}^{L-1} n_{t,t+1}(s_i, s_j)}{\sum_{t=1}^{L-1} n_t(s_i)} \tag{1}$$

where L is the length of the sequence that is maximally observed. Therefore, following a specific outcome, the inclination of the choice of next outcome is determined by transition rate matrices. Transition probability of origin state s_i at t to states at $t + 1$ is generated by every i^{th} row of the resulted matrix. For every positions $t = 1, 2, \ldots, L - 1$, if an independent transition rate matrix is formed with three-dimensional set up, then a time-varying transition distribution could be produced. Simply, taking into account the states at t along with the states at $t + 1$, the matrix

with respect to position t is computed. And, position t index makes reference to the third dimension related to an array.

Entropy Measures Shannon's entropy is a technique of quantifying heterogeneity of observed states at a determined position. A graphical representation of these entropies gives us an indication how the states are becoming heterogeneous over time. Let us consider at a given position in state i, the proportion of cases is p_i. Then, Shannon's entropy can be written as

$$h(p_1, p_2, \ldots, p_a) = -\sum_{i=1}^{a} p_i \log_{p_i} \tag{2}$$

where size of an alphabet is denoted by a. If entire cases belong to the same state, then the entropy is 0. The entropy is maximized when each state has the equal proportion of cases.

Turbulence To account simultaneously for both the aspects, i.e. sequencing and durations, Elzinga [17] has suggested a compound method of measuring turbulence. Firstly, this measure of turbulence considers the total count $\phi(x)$ of different subsequences obtained from DSS sequences. Secondly, it takes into account the variance $s_t^2(x)$ which is obtained from the successive times t_j spent in $l_d(x)$ different states. Thus, the measure of turbulence $T(x)$ of a generated sequence x is given by,

$$T(x) = \log_2(\phi(x)\frac{s_{t,\max}^2(x) + 1}{s_t^2(x) + 1}) \tag{3}$$

Given a total duration $l(x) = \sum_j t_j$ of a sequence, $s_t^2(x)$ could reach upto a maximum value of $s_{t,\max}^2(x)$, where $s_{t,\max}^2(x) = (l_d(x) - 1)(1 - \bar{t}(x))^2$.

Dissimilarity Measures and Clustering For visualizing different patterns of the event history sequences in an effective way, researchers often want to classify those sequences in distinct groups based on their similarities or dissimilarities so that the resulting groups become homogeneous in nature. In the sake of estimating the similarity over the sequences, a composite measure of numerous characteristics is to be essential as individual estimates cannot properly explain the sequences altogether.

A distance matrix is a square symmetric matrix whose ij^{th} element is equal to the ji^{th} element and signifies the distances among a group of paired objects [8]. This assists us in investigating the relationships as well as in obtaining meaningful characteristics of the data. Optimal matching in unsupervised learning provides us the best way of creating this kind of distance matrix [5, 6, 25, 30]. This method converts one sequence to another through the distances which is generated by estimating the total count of elementary operations. If the obtained differences are large enough, indicates that, a lot number of operations have been performed. A cost is certainly associated with every operation. Converting one sequence into another requires a minimal cost. This is popularly known as edit distance. Generally, these operations are mainly of two types. One of them is known as substitution, and the other one is

recognized as indel operation. When one element is replaced by the other element, then this process is called as the method of substitution. Sometimes, it is possible to shift a position for all components on the right side of an deleted or inserted component, and this process is termed as indel operation. The optimal matching distance was first considered by [30] and used by [4]. Owing to a position varying substitution cost, a three-dimensional matrix is essential when the cost in an indel operation is fixed as sufficiently high which does not depend on related position as well as state. Thus, taking into account the estimated transition rates, the cost is measured by

$$2 - p(s_i|s_j) - p(s_j|s_i) \tag{4}$$

Given that the state s_j has been observed at time t, the probability of observing state s_i at time $t + 1$ is $p(s_i|s_j)$. If substitution of each state is done by itself, then the cost is 0 which is minimum and the maximum is less than 2, otherwise. For normalized distances, authors are referred to Gabadinho et al. [21].

The distance matrix is extensively used in performing cluster analysis. Based on numerous dimensions, it helps us in identifying the similarities among the cluster of sequences [8, 28]. Instead of using clustering algorithms like centroid-based clustering, distribution-based clustering and density-based clustering, we use Wald hierarchical clustering for the sequences. Wald suggested a procedure where the criterion for choosing the pair of clusters to merge at each step is based on the optimal value of an objective function. The initial cluster distances in Ward's method are therefore defined to be the squared Euclidean distance between points and is expressed as

$$d_{ij} = d(X_i, X_j) = \| X_i - X_j \|^2 . \tag{5}$$

4 Exploratory Analysis of Distribution of Adulthood Events by Geographical Region

4.1 Proportion of Events by Geographical Region

The period of transition to adulthood is marked by several events viz. discontinuation of school, entry into workforce, entry into marital union, etc. Data collected through life event calender component of Youth Study [34] provide us an opportunity to explore the pattern of these events in young people's lives through descriptive measures. Table 1 shows proportions for different events for the six geographical region.

Proportions varied widely by sex and place of residence. The first panel of Table 1 shows that 42.6% (lowest) of the total people in Bihar quit studying during the age interval 12–29, whereas 76.5% (highest) of the total people in Tamil Nadu have done the same. 30.9% of Tamil Nadu's urban male and 28.4% of the Maharashtra's rural male left study during the age interval 12–29. Urban and rural females staying in these six geographical region show higher percentages of leaving study than their

Table 1 Proportions of events by geographical region

	Overall (%)	Urban male (%)	Urban female (%)	Rural male (%)	Rural female (%)
Event: leaving study					
Rajasthan	49.5	22.7	30.3	22.4	24.7
Bihar	42.6	18.3	28.5	15.8	22.8
Jharkhand	71.9	28.2	35.4	23.8	56.6
Maharashtra	70.0	28.3	36.7	28.4	47.2
Andhra Pradesh	64.8	30.7	35.0	25.9	38.2
Tamil Nadu	76.5	30.9	41.1	28.1	52.1
Event: work					
Rajasthan	55.7	26.8	17.5	29.2	34.5
Bihar	48.5	25.5	14.3	24.2	32.6
Jharkhand	54.0	29.6	15.1	25.1	38.5
Maharashtra	52.1	30.1	12.7	30.9	31.5
Andhra Pradesh	65.1	34.8	20.1	32.9	41.3
Tamil Nadu	58.8	31.0	22.1	28.2	35.3
Event: marriage					
Rajasthan	44.9	15.4	25.4	21.2	26.5
Bihar	42.5	13.8	28.7	13.6	28.9
Jharkhand	44.7	13.5	23.1	15.0	38.0
Maharashtra	39.8	12.7	22.6	15.6	29.1
Andhra Pradesh	44.8	16.6	26.5	17.1	29.2
Tamil Nadu	41.6	18.2	22.4	15.2	27.3

male counterparts. Among these six regions, 41.1% of the total urban females of Tamil Nadu and 56.6% of the total rural females of Jharkhand left study during the above-mentioned age interval. Females in Bihar state show the lowest proportion in this event.

Second panel of Table 1 shows the proportions with respect to the event 'entry into workforce'. More than half of the total population in all the states show a tendency to entering into workforce. Male respondents in Bihar (25.5% in urban and 24.2% in rural) show the lowest percentages for workforce-entry, whereas male respondents in Andhra Pradesh (34.8% in urban and 32.9% in rural) have shown the highest percentages in workforce-entry. 12.7% urban and 31.5% rural females in Maharashtra show the lowest percentages in entry into workforce than that of females in other states. However, urban females enter into work at a much lower rate than urban males, whereas rural females tend to enter into work at a much higher rate than rural males.

State-wise proportions of the respondents in marital union are shown in the third panel. Rural and urban males show significantly lower percentages in marriage than the females. 22.4% (lowest) of urban females in Tamil Nadu and 26.5% (lowest) of rural females in Rajasthan have entered into marital union during the age interval 12–24, while 28.7% (highest) of urban females in Bihar and 38% rural females in Jharkhand show the highest percentages in marriage. However, only 12.7% (lowest) of the total urban males in Maharashtra and 13.6% rural males in Bihar entered into marital union during 12–29 years of age. Overall, we can conclude that females show a higher tendency in both the events, namely leaving education and entry into marriage. But, male respondents show significantly higher percentages for entry into workforce, i.e. male respondents are showing greater tendency to entering into workforce as soon as they leave study.

4.2 Failure Rate Function

Kaplan–Meier failure estimates (Fig. 1) are used to visualize the patterns of the events namely leaving education, entry into workforce and entry into marital union. Figure 1 shows that patterns of these three events not only differ by areas of residence but also considerably by sex and states. The timing at experiencing these events generated different patterns specific to different groups of population.

The people of Rajasthan, Bihar and Andhra Pradesh are inclined to go to work before leaving study though the urban females are showing an opposite pattern. Male respondents of Jharkhand and male respondents in the urban area of Maharashtra have shown nearly similar timing of leaving study and entry into work. However, something special is seen about the people of Tamil Nadu, where everyone is first leaving their studies and then getting into work. When considering marriage, it is seen that women in rural areas are entering into the marital union earlier than other groups of people. But, it turns out that men are extensively delaying their entry into marital union as compared to their female counterparts in each and every region. This is very reasonable in Indian scenario.

5 Sequence Analysis

5.1 Index Plot

Sequences across the ages with respect to each individual are represented through basic index plot in Fig. 2. Each event's outcome is colour coded in index plots as well as in state frequency plot and state distribution plot. Each sequence, i.e. the record of events for a single individual, is represented by an horizontal line in the plot. Near about 10% of the total individuals are of age 15, and they experience schooling,

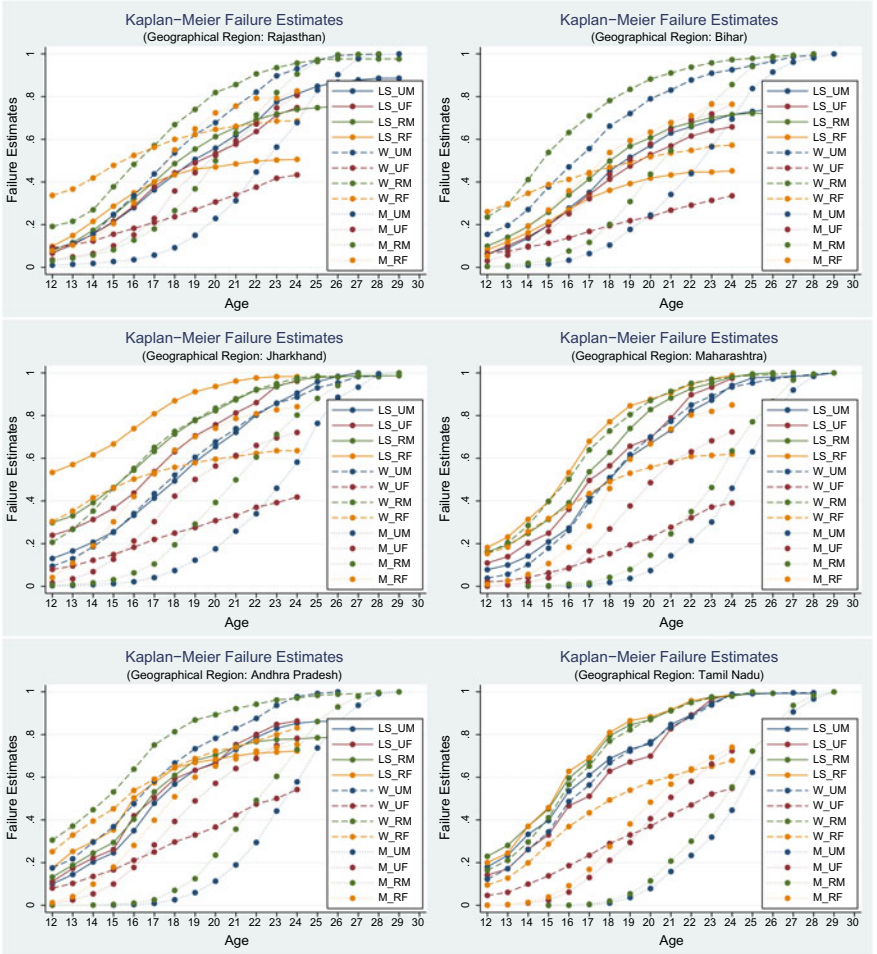


Fig. 1 Failure rate function for all the three event types by sex and place of residence separately for all the six geographical region

workforce and both schooling and working at the same time. A small proportion of them are not in school, nor in workforce and remained unmarried. There is a decline in this trend after the age of 18 years and more aged individuals dominantly driven by entry into marital union and workforce. The majority of individuals in the age group of 20–25 have shown the risk of entering into the workforce, into the marital union and simultaneously in both.

Sequence index plots for all the six geographical regions, i.e. Rajasthan, Bihar, Jharkhand, Maharashtra, Andhra Pradesh and Tamil Nadu are presented in Fig. 3a, b. A little proportion of the people currently aged 15 years are working and studying at the same time except the individuals staying in Rajasthan. A small percentages of

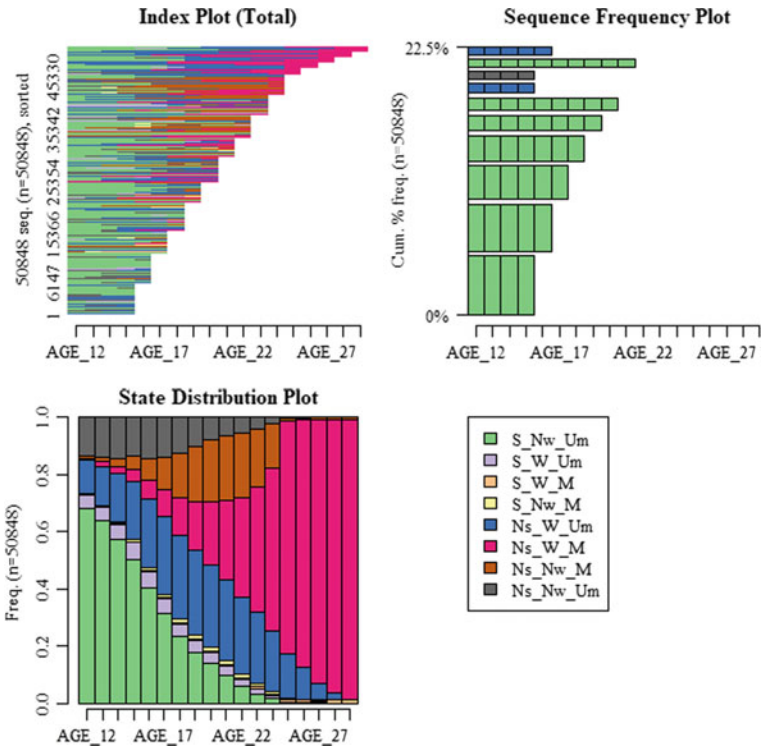


Fig. 2 Exploring the data via a plot of the sequences (top-left), most frequent sequences (top-right) and state distribution plot (bottom-left)

them are only in workforce except individuals staying in Tamil Nadu, and also, a little proportions of the people aged 15 years in Jharkhand are already married. Some of the proportions showing that they are neither in school nor working and unmarried. Total proportions of the individuals currently aged 15 in Tamil Nadu show that all of them are in education. The proportions are getting higher in marriage after the age of 20 years in Rajasthan, after 22 years of age in Andhra Pradesh, after 23 years of age in Jharkhand, Maharashtra, Tamil Nadu and after the age of 24 years in Bihar. At the end, total proportions of the people aged between 25 and 29 years in all the regions are married, which is naturally common in modern Indian society.

5.2 State Frequency Plot

The sequence index plot, reported in Fig. 2, generally suffers from the problem of over-plotting by providing a large number of sequences; and thereby it may hide particular patterns in the sequences if not carried out carefully. To avoid these limita-

tions, plotting of cumulative percentages of individual sequences through sequence frequency plot is suggested in the literature. Top right panel of Fig. 2 is the sequence frequency plot showing the cumulative percentages of individuals in each state and over time. The largest group dominated by schooling is represented by a wider row. Also, first six groups dominated only by those who are in school. Those who entered into workforce follow this.

Geographical region-wise sequence frequency plots show that 80–85% of the individuals are in study for the entire period in Maharashtra, Andhra Pradesh and Tamil Nadu, where 60–70% of the individuals are in study for the entire period in Rajasthan, Bihar and Jharkhand. Those who engaged in workforce for the entire period in Rajasthan, Bihar, Jharkhand and Andhra Pradesh follow this. Those who left study followed by neither studying, not working, nor even married, are in the second largest group for Tamil Nadu and Maharashtra. The third largest group for these two states considers those who first left study followed by entry into workforce.

5.3 State Distribution Plot

The bottom panel of Fig. 2 shows the state distribution plot. It describes the distribution across all of the eight states over the entire period. We see that the proportion of individuals who become married over time increases to the extent that it becomes the most prominent state by the end of the period. We also observe that, at the early ages, proportion of the individuals who are studying is higher followed by those who are in workforce. The top of the graphic wane indicates that a little proportion of all the respondents are in notstudying-notworking-unmarried state.

The middle-right panels of Fig. 3a, b shows that almost 60% of the total individuals are in education at the beginning for Rajasthan, Bihar and Jharkhand, whereas almost 80% of the totals are in education for Maharashtra and Tamil Nadu, and near about 70% of the total people in Andhra Pradesh are in study at the beginning of the observation period. This is followed by a small period of unemployment as well as unmarried individuals, followed by a comparatively larger group of working-unmarried individuals. We also observe that the proportion of individuals who are married and working simultaneously over time increases to the extent that it becomes the most prominent state by the end of the observation period. However, Tamil Nadu individuals occupy the largest part in workforce among all the regions. By the end of the observation period, a very little proportions of the individuals in Bihar and Jharkhand show that after entering into marital union, they are studying and working at the same time. Also, the individuals in Bihar region occupy the largest part in marriage (neither studying-not working) state among all the regions.

5.4 Entropy and Turbulence Measures

Another option to understand the data is via a calculation of the Shannon entropy of the observed state distribution also shown and briefly studied in Billari [9]. Calculated Shannon entropy of the state distribution is represented here in Fig. 4. We can see from Fig. 4 that when all the individuals have the same state, then the entropy is zero, it means that, there is no heterogeneity present in the data. Figure 4 shows that the entropy is high at the beginning and then levels off across time, which is a common finding in research of the transition to adulthood [32]. The entropy is 0.5 at the beginning, reaches top after eight years from beginning, i.e. 19 years of age, and then levels off towards zero as the time moves on. This matches our previous observations from Fig. 2, which showed almost 50% of the variation at the beginning, more

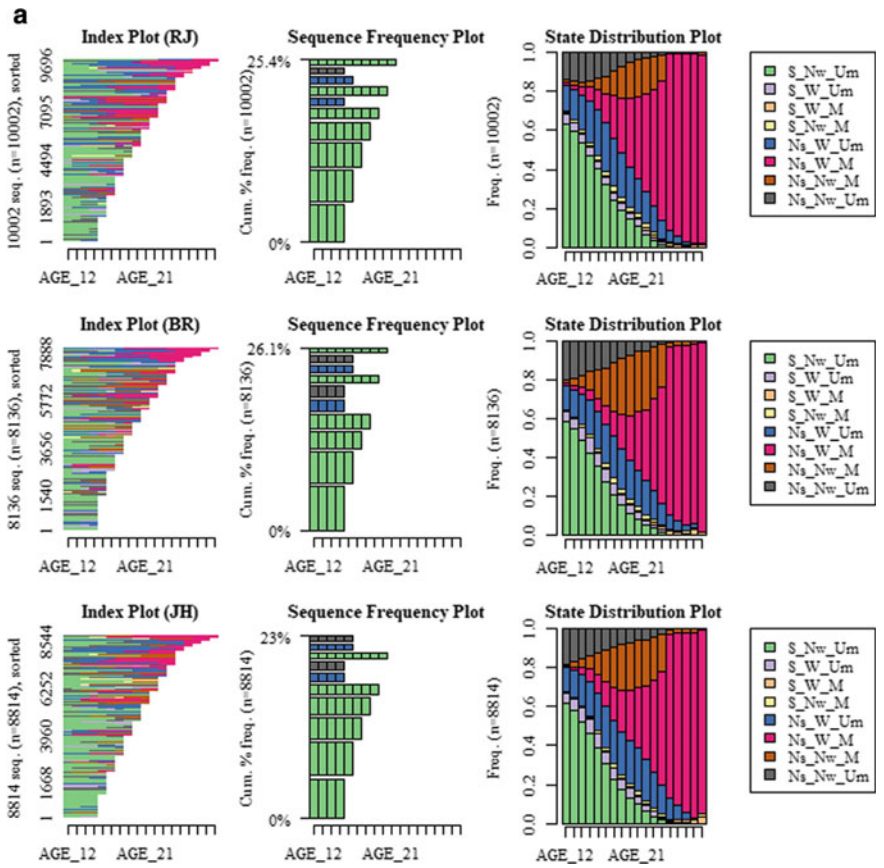


Fig. 3 **a** Geographical region-wise plots of the sequences (left), most frequent sequences (middle-left) and state distribution plot (middle-right) are presented (Rajasthan, Bihar and Jharkhand). **b** Geographical region-wise plots of the sequences (left), most frequent sequences (middle-left) and state distribution plot (middle-right) are presented (Maharashtra, Andhra Pradesh and Tamil Nadu)

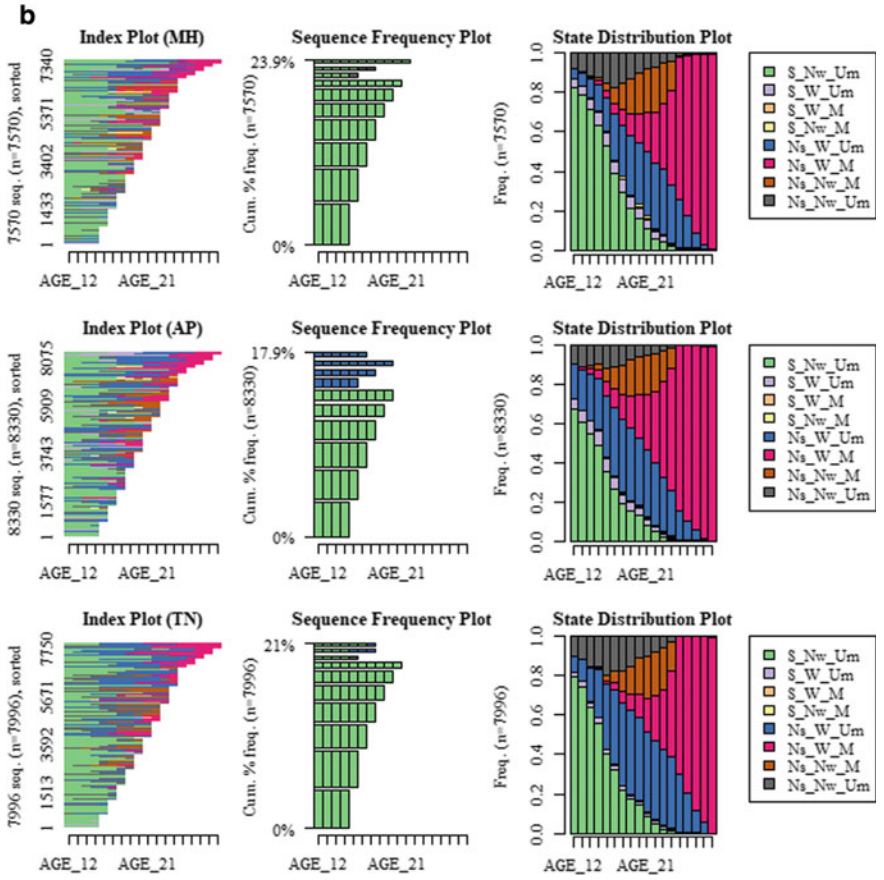


Fig. 3 (continued)

variation in the states at the middle and progressively more individuals becoming employed and married by the end of the observation period leads to least variation at the end. Plotted entropies for the various geographical regions are shown in Fig. 5a, b. Variation of the state entropies is observed among these six regions. Steady rise in the entropies is seen in Andhra Pradesh region, whereas a more gradual rise is seen in Tamil Nadu and Maharashtra. Lowest unobserved heterogeneity is present in Maharashtra at the beginning years, whereas the highest heterogeneity is observed for Bihar region at that time. It is also seen that after the age of 25 years, there are steady decrease in entropies for all the geographical regions.

The right panel of Figs. 4 and 5a, b represents Elzinga'z measure of turbulence. From Table 2 we see that the mean of the turbulence is 3.29 with a minimum value of 1 and a maximum value of 9.8, echoed in the histogram in the right panel of Fig. 4. Mean turbulence differs across geographical regions with a minimum mean value of 2.99 for Bihar and with a maximum mean value of 3.583 for Tamil Nadu. Median,

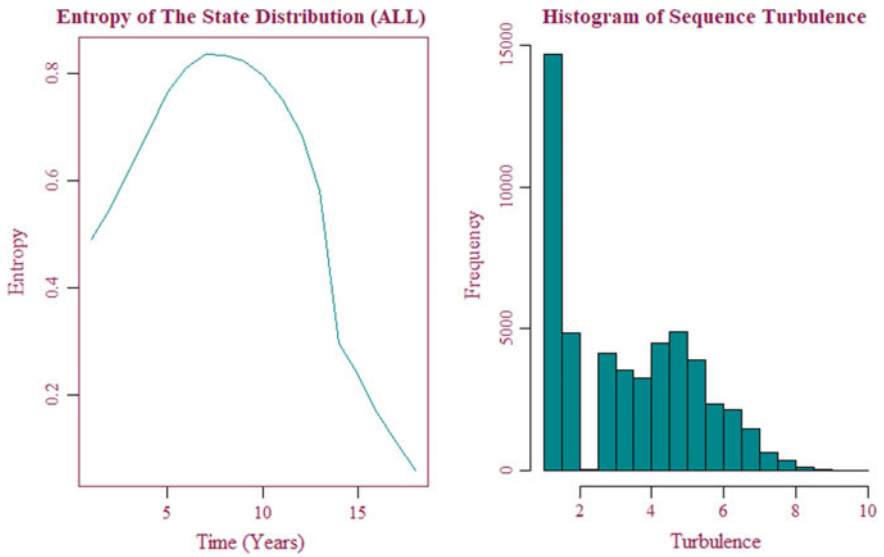


Fig. 4 Entropy of state distribution and histogram of sequence turbulence

third quartile and maximum value also differ across geographical regions. Maximum value ranges from 8.428 (for Rajasthan) to 9.8 (for Andhra Pradesh) although the minimum value for all the regions gives a value 1. Hence, we can conclude that the geographical regions differ in respect of their turbulence measures which means unobserved heterogeneity and complexity both are present in the data.

5.5 Cluster Analysis

Finally, sequences are clustered depending upon the distance matrix obtained through optimal matching and are popularly known as 'Wald hierarchical clustering'. Using the 'constant' method for optimal matching in TraMineR, it is found that all probable event outcomes have an equal substitution cost. It is also observed that two indel operations are approximately identical to a single substitution operation. But, determining the costs in accordance with 'transition rates' is more instinctive. Hence, for simplicity, 'TRATE' approach of TraMineR is used here.

For a better interpretable result, an appropriate choice of the number of clusters is to be considered while performing a cluster analysis. In this case, a four-class solution is found to be the best among a series of possible number of clusters ($k = 3, 4, 5$). Figure 6 displays the four cluster solution separately for each geographical region. The first cluster (Type 1) ($n = 4557$) of Rajasthan is mainly dominated by those who are studying with a little number of working-married sequences. Cluster 2 (Type 2) contains those individuals who are working and are in both workforce as well as

marital union at the same time. The third cluster (Type 3) is dominated by mostly of working-married individuals. A little part of them are working and studying. Cluster 4 (Type 4) illustrates a more mixed cluster, such as considerably more being in the married, notstudying-notworking-unmarried states. Surprisingly, a greater part of studying-working-married individuals is in this cluster. A similar pattern is visible for Tamil Nadu.

In Bihar, the first cluster is dominantly driven by those individuals who are only in workforce but also contain those who are in both workforce and marital union at the same time. Second cluster consists of mainly the working-married individuals. Third cluster is a more mixed one but mostly dominated by those who are in education. The fourth cluster considerably more being in married, notstudying-notworking-unmarried states.

The first cluster of Jharkhand shows that those individuals who are in working-married category are in this cluster. Second cluster is dominated by working and working-married individuals, third cluster consists primarily of students, and the fourth cluster is considerably built by married, notstudying-notworking-unmarried and working-married individuals. Maharashtra shows that maximum number of students are in cluster 4, whereas cluster 1 gives similar result to cluster 1 of Rajasthan. Here, cluster 2 consists of married, notstudying-notworking-unmarried and working-married individuals. Also, cluster 3 seems to be dominated by working-married individuals. A more mixed cluster (cluster 1) is observed for Andhra Pradesh, while the second cluster shows that those individuals who are in working-married and working state are in this cluster. Most of the students are in the third cluster, and mostly married as well as notstudying-notworking-unmarried individuals are in the fourth cluster. Most importantly, those clusters which have the maximum number of individuals are dominated by students.

6 Conclusion

This article contributes to the methodological aspects in two ways. Initially, we describe the way, in which sequence analysis method could be used in investigating the event history data explicitly as well as simultaneously considering the 'quantum', 'timing' and 'sequencing' of events. Secondly, in analysing event history data, techniques established in unsupervised machine learning are elaborately used. In investigating these types of event history data, a high amount of resiliency is acceptable by these techniques. These techniques allow us in extracting accessible information from the data and provide us a valid representation.

This present research admits principal role of 'timing', 'sequencing', 'quantum' and 'clustering' on the study of transition to adulthood. To extract the complex relationship between the sequences of events, we suggest a way to analyse the event history data which are able to find the hidden information. We implement the method of unsupervised machine learning to the transition to adulthood data obtained from a national-level survey called 'Youth Study'. More specifically, we extract and vi-

sualize the most frequent sequences which allow us in identifying the principal sequences among the patterns. Moreover, the indication of an event's occurrence that is the 'quantum' is often limited to highlighting the basic characteristics of Indian Youths.

Acceptance of clustering algorithms permits us to observe the event histories at an unique viewpoint. Outcomes of cluster analysis provide a distinct view when an attempt is made to distinguish among the group of individuals. Although survival and event history analysis investigates event history data, they are unable to operate in these types of complex situations. Summarily, these techniques presented in this paper are capable of performing variety of comparative studies such as comparison of communities, comparison of genders and comparison of different geographical

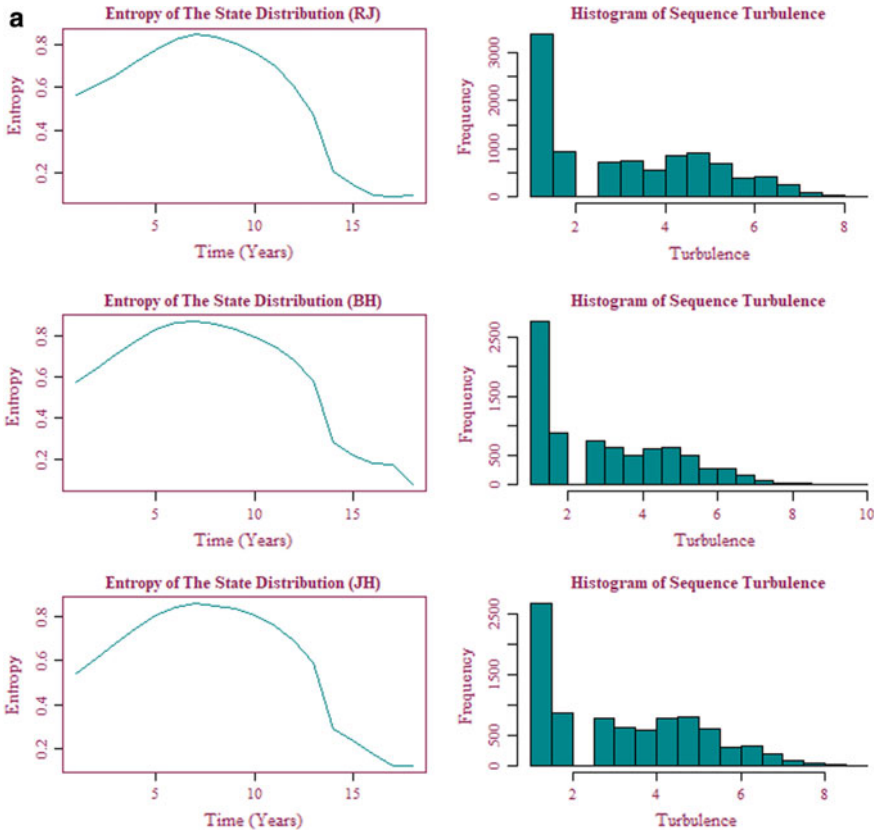


Fig. 5 **a** Geographical region-wise entropy of state distribution and histogram of sequence turbulence (Region: Rajasthan, Bihar and Jharkhand). **b** Geographical region-wise entropy of state distribution and histogram of sequence turbulence (Region: Maharashtra, Andhra Pradesh and Tamil Nadu)

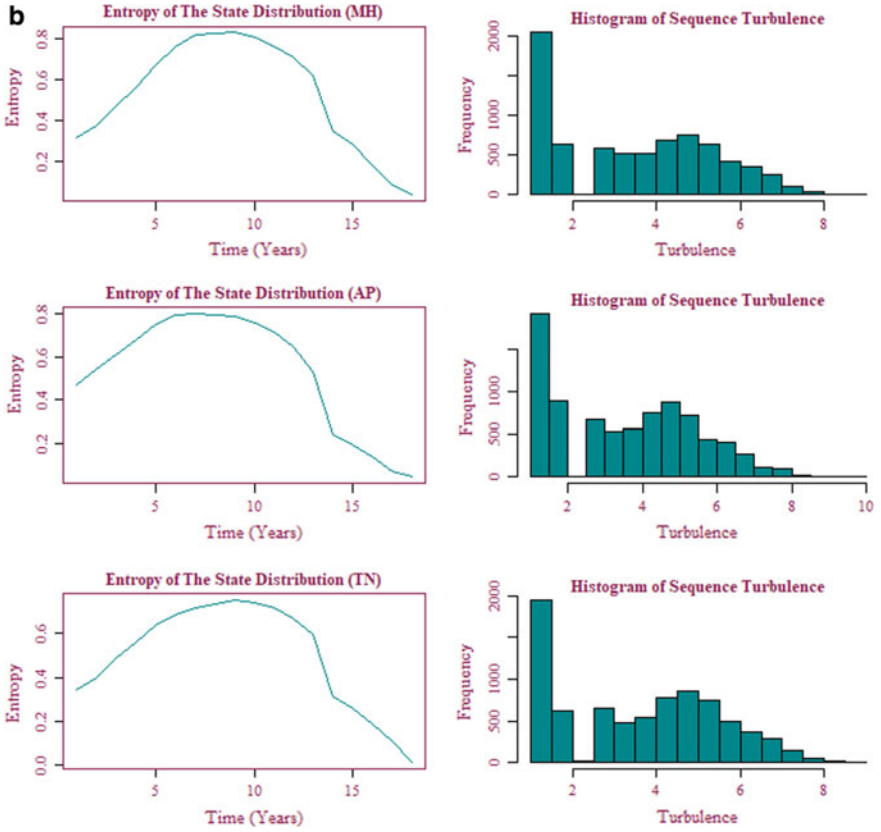


Fig. 5 (continued)

Table 2 Summary measures of turbulence

Turbulence							
Measures	All	RJ	BH	JH	MH	AP	TN
Min.	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1st Quartile	1.000	1.000	1.000	1.000	1.000	2.000	2.000
Median	3.322	3.000	3.000	3.158	3.528	3.628	3.807
Mean	3.294	3.113	2.990	3.166	3.437	3.539	3.583
3rd Quartile	4.745	4.585	4.441	4.543	4.949	5.048	5.087
Maximum	9.800	8.428	9.547	8.783	8.916	9.800	8.783

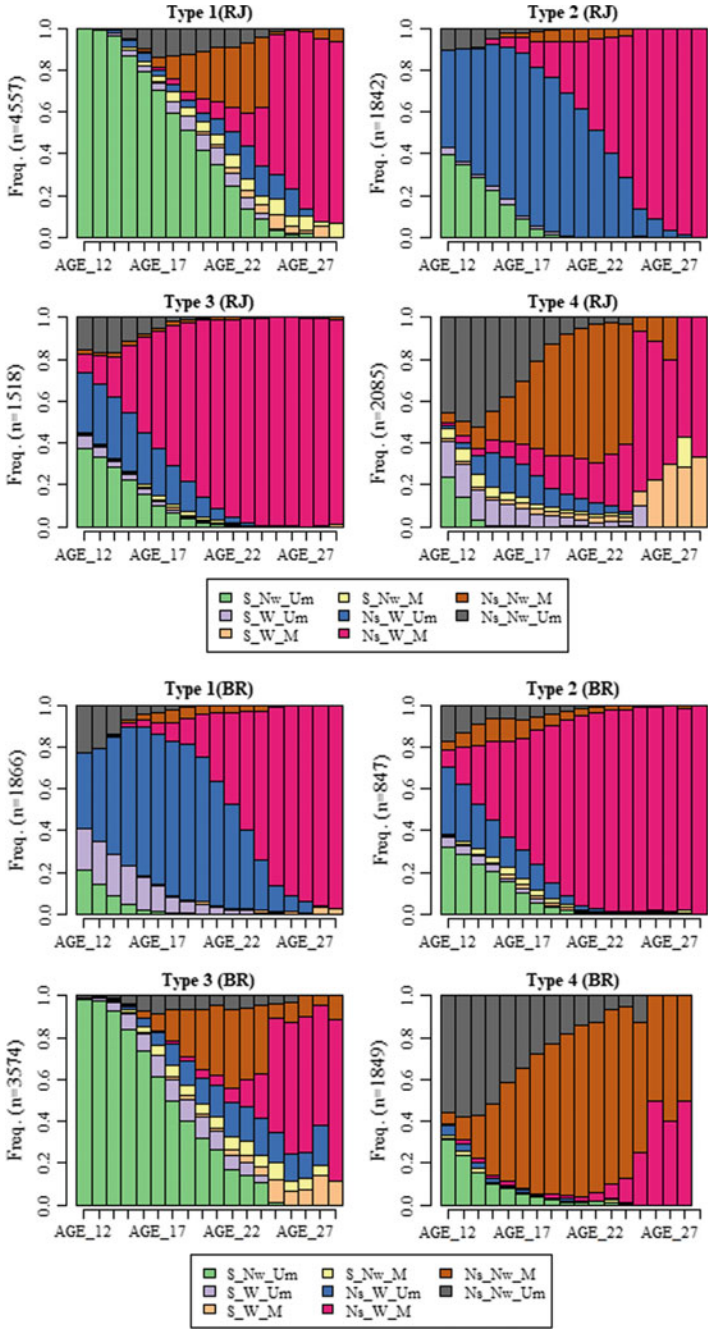


Fig. 6 Geographical region-wise state distribution across time of Wald hierarchical clustering of sequences from optimal matching distances with a four-class solution

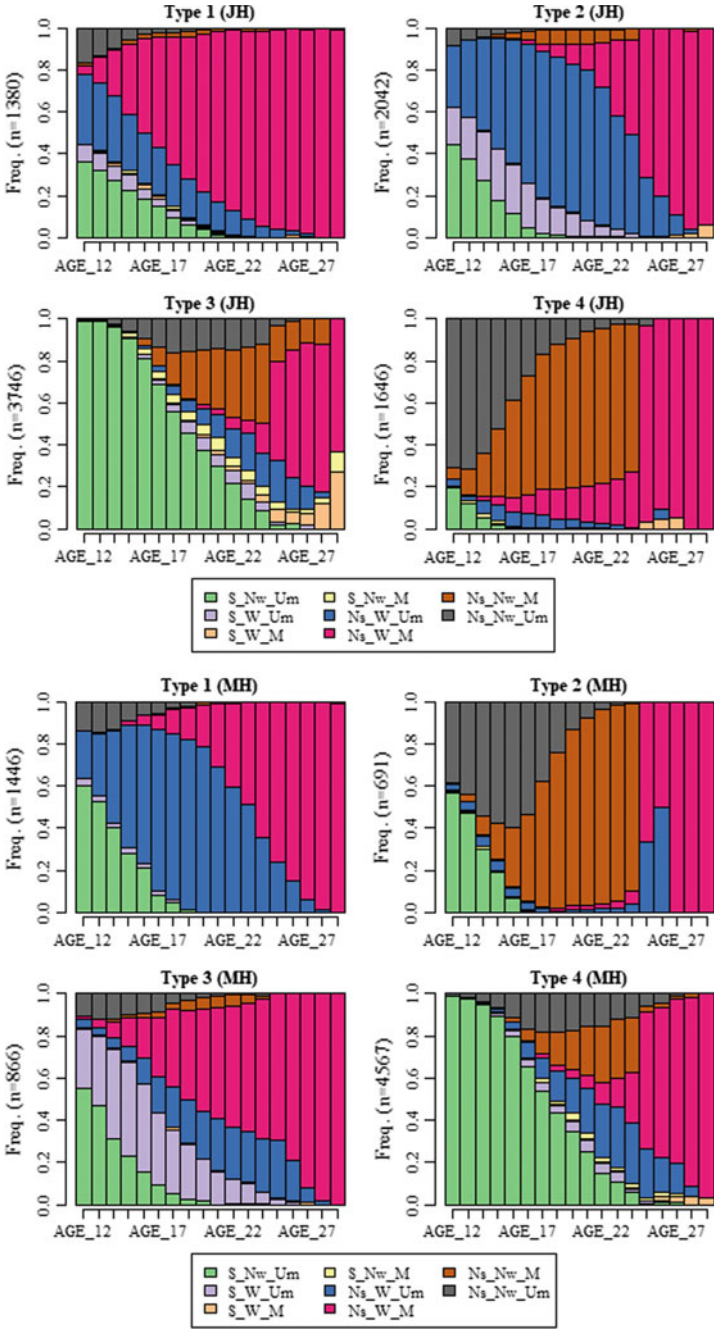


Fig. 6 (continued)

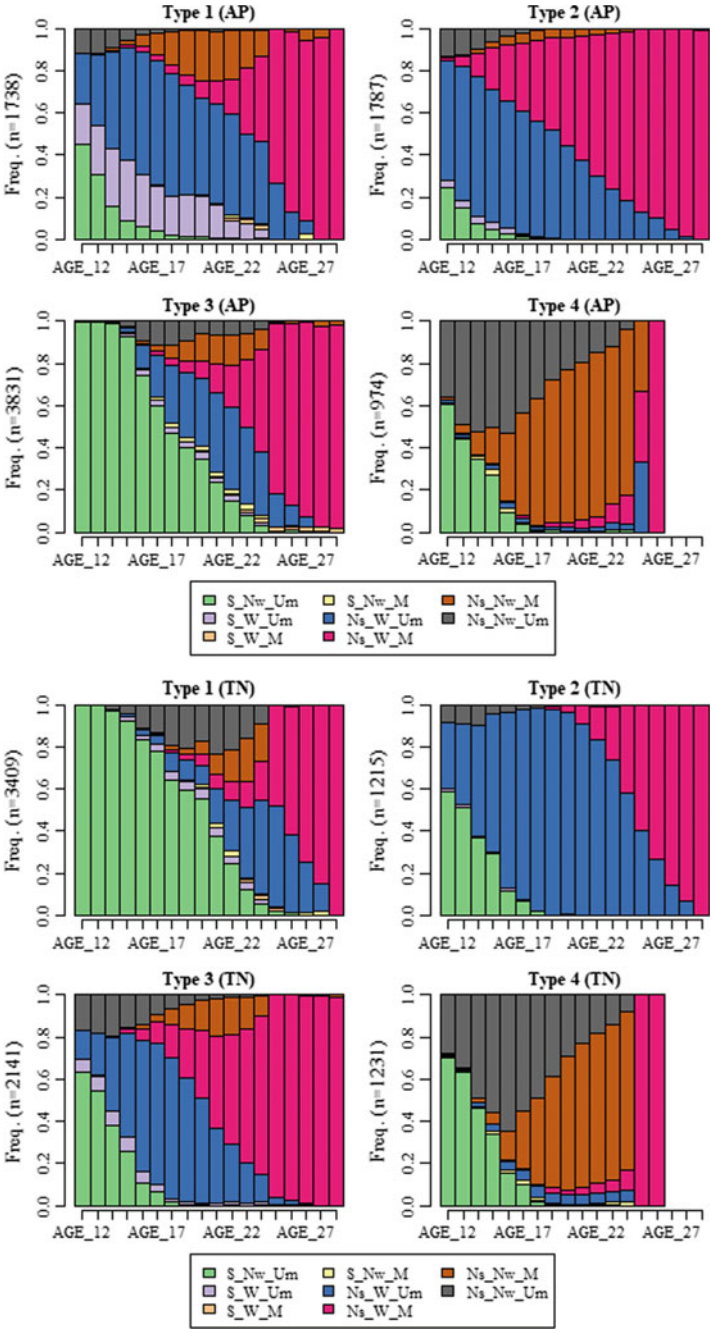


Fig. 6 (continued)

regions. we have also developed the comparison of the characteristics of the youths in six geographical regions.

Finally, we can conclude that this current paper establishes sequence analysis as a mechanism to investigate the transition to adulthood sequences of Indian Youths. It helps us in better understanding as well as improving knowledge about the youths in India. For visualizing and summarizing complex event history dataset, the above technique provides a potentially powerful tool. It also helps in detecting unusual cases as well as subgroups. Summarily, not only for an interpretable and significantly meaningful outcome, also for representing a valid classification, cluster analysis appears to be a better method, and the results obtained could be adopted for further analysis in future.

References

1. Aassve A, Billari FC, Piccarreta R (2007) Strings of adulthood: a sequence analysis of young british women's work-family trajectories. Euro J Popul/Revue européenne de Démographie 23(3–4):369–388
2. Abbott A (1983) Sequences of social events: concepts and methods for the analysis of order in social processes. Hist Methods: J Quant Interdisciplinary History 16(4):129–147
3. Abbott A (2000) Reply to Levine and Wu. Sociol Methods Res 29(1):65–76
4. Abbott A, Forrest J (1986) Optimal matching methods for historical sequences. J Interdisciplinary History 16(3):471–494
5. Abbott A, Hrycak A (1990) Measuring resemblance in sequence data: an optimal matching analysis of musicians' careers. Am J Sociol 96(1):144–185
6. Abbott A, Tsay A (2000) Sequence analysis and optimal matching methods in sociology: review and prospect. Sociol Methods Res 29(1):3–33
7. Archibald AL, Bolund L, Churcher C, Fredholm M, Groenen MA, Harlizius B, Lee KT, Milan D, Rogers J, Rothschild MF et al (2010) Pig genome sequence-analysis and publication strategy. BMC Genom 11(1):438
8. Bartholomew DJ, Steele F, Moustaki I (2008) Analysis of multivariate social science data. Chapman and Hall/CRC, Boca Raton
9. Billari FC (2001) The analysis of early life courses: complex descriptions of the transition to adulthood. J Popul Res 18(2):119–142
10. Billari FC (2001) Sequence analysis in demographic research. Can Stud Popul 28(2):439–458
11. Billari FC, Liefbroer AC (2010) Towards a new pattern of transition to adulthood? Adv Life Course Res 15(2–3):59–75
12. Blossfeld HP et al (2001) Techniques of event history modeling: New approaches to casual analysis. Psychology Press
13. Brzinsky-Fay C (2007) Lost in transition? labour market entry sequences of school leavers in europe. Eur Sociol Rev 23(4):409–422
14. Durbin R, Eddy SR, Krogh A, Mitchison G (1998) Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge
15. Ecker M, Allison SL, Meixner T, Heinz FX (1999) Sequence analysis and genetic classification of tick-borne encephalitis viruses from Europe and Asia. J Gen Virol 80(1):179–185
16. Elzinga CH (2005) Combinatorial representations of token sequences. J Classif 22(1):87–118
17. Elzinga CH, Liefbroer AC (2007) De-standardization of family-life trajectories of young adults: a cross-national comparison using sequence analysis. Euro J Popul/Revue européenne de Démographie 23(3–4):225–250

18. Elzinga CH, Studer M (2015) Spell sequences, state proximities, and distance metrics. Sociol Methods Res 44(1):3–47
19. Elzinga CH, Wang H (2013) Versatile string kernels. Theoret Comput Sci 495:50–65
20. Elzinga C, Studer M (2016) Normalization of distance and similarity in sequence analysis. LaCOSA II, Lausanne pp, pp 445–468
21. Gabadinho A, Ritschard G, Mueller NS, Studer M (2011) Analyzing and visualizing state sequences in R with traminer. J Stat Softw 40(4):1–37
22. Giele JZ, Elder GH (1998) Methods of life course research: qualitative and quantitative approaches. Sage, Thousand Oaks
23. Goldstein AL, Low TL, McAdoo M, McClure J, Thurman GB, Rossio J, Lai CY, Chang D, Wang SS, Harvey C et al (1977) Thymosin alpha1: isolation and sequence analysis of an immunologically active thymic polypeptide. Proc Nat Acad Sci 74(2):725–729
24. Han Y, Liefbroer AC, Elzinga CH (2017) Comparing methods of classifying life courses: sequence analysis and latent class analysis. Longitudinal Life Course Stud 8(4):319–341
25. Hollister M (2009) Is optimal matching suboptimal? Sociol Methods Res 38(2):235–264
26. Jacobs GH (1992) Determination of the base recognition positions of zinc fingers from sequence analysis. EMBO J 11(12):4507–4517
27. Kaneko T, Sato S, Kotani H, Tanaka A, Asamizu E, Nakamura Y, Miyajima N, Hirosawa M, Sugiura M, Sasamoto S et al (1996) Sequence analysis of the genome of the unicellular cyanobacterium Synechocystis sp. strain PCC6803. II. sequence determination of the entire genome and assignment of potential protein-coding regions. DNA Res **3**(3), 109–136
28. Kaufman L, Rousseeuw PJ (2009) Finding groups in data: an introduction to cluster analysis, vol 344. Wiley, Hoboken
29. Lesnard L (2006) Optimal matching and social sciences
30. Levenshtein VI (1966) Binary codes capable of correcting deletions, insertions, and reversals. Soviet Phys Doklady 10:707–710
31. Mayer KU, Tuma NB (1990) Event history analysis in life course research. University of Wisconsin Press, Madison
32. Mills M (2010) Introducing survival and event history analysis. Sage, Thousand Oaks
33. Modell J, Furstenberg FF Jr, Hershberg T (1976) Social change and transitions to adulthood in historical perspective. J Family History 1(1):7–32
34. International Institute for Population Sciences (IIPS), Population Council (2010) Youth in India: situation and needs 2006–2007
35. Rallapalli G, Corredor-Moreno P, Chalstrey E, Page M, MacLean D (2019) Rapid fine mapping of causative mutations from sets of unordered, contig-sized fragments of genome sequence. BMC Bioinform 20(1):9
36. Ram U, Mohanty S, Singh A (2010) Youth in India: situation and needs
37. Rindfuss RR (1991) The young adult years: diversity, structural change, and fertility. Demography 28(4):493–512
38. Robette N (2010) The diversity of pathways to adulthood in france: evidence from a holistic approach. Adv Life Course Res 15(2–3):89–96
39. Robette N, Bry X (2012) Harpoon or bait? a comparison of various metrics in fishing for sequence patterns. Bull Sociol Methodol 116(1):5–24
40. Roser LG, Agüero F, Sánchez DO (2019) Fastqcleaner: an interactive bioconductor application for quality-control, filtering and trimming of FASTQ files. BMC Bioinform 20(1):361
41. Sbalzarini IF, Theriot J, Koumoutsakos P (2002) Machine learning for biological trajectory classification applications
42. Studer M, Ritschard G (2016) What matters in differences between life trajectories: a comparative review of sequence dissimilarity measures. J R Stat Soc: Sers A (Stat Soc) 179(2):481–511
43. Walsh TR, Hall L, Assinder SJ, Nichols WW, Cartwright SJ, MacGowan AP, Bennett PM (1994) Sequence analysis of the L1 metallo-β-lactamase from *Xanthomonas maltophilia*. Biochimica et Biophysica Acta (BBA)-Gene Struct Exp **1218**(2), 199–201
44. Ward JH Jr (1963) Hierarchical grouping to optimize an objective function. J Am Stat Assoc 58(301):236–244

45. Wehner S et al (1999) Exploring and visualizing event history data
46. Yamaguchi K (1991) Event history analysis, vol 28. Sage, Thousand Oaks
47. Yger P, Harris K (2013) The convallis learning rule for unsupervised learning in spiking neuronal networks. BMC Neurosci 14(1):P426

Chapter 21
A Quantile-Based Approach to Supervised Learning

Dreamlee Sharma and Tapan Kumar Chakrabarty

1 Introduction

Supervised statistical learning aims at building a predictive model for a response based on one or more inputs known as features. A very simple approach of supervised learning for predicting a quantitative response is the linear regression model, also known as predictive model in terms of statistical learning. Regression analysis is one of the core statistical techniques in scientific applications. It examines the relationship between two or more variables of interest by modeling the influence of one or more independent variables, called regressors or predictors through some function, on a response variable of interest. The conceptual model with additive deterministic and stochastic components is written in the form

$$\text{observation} = \text{deterministic component} + \text{stochastic component} \tag{1}$$

In the development of regression, analysis of the model (1) is carried out using two distinct approaches. The first approach defines the deterministic component parametrically without any specific reference to the distribution of the stochastic term and is termed as semi-parametric. The modeling and estimation of the two components have been traditionally performed separately. The procedure of fitting the deterministic component is ruled by the procedure of least squares and then by the procedure of least absolutes. The procedure of least absolute [25] first introduced by [4] is a method in which the parameters are estimated so that the sum of absolute differences

D. Sharma
Department of Mathematics, Adamas University, Kolkata 700126, India

D. Sharma (✉) · T. K. Chakrabarty
Department of Statistics, North-Eastern Hill University, Shillong 793022, India
e-mail: dreamleesharma@yahoo.in

T. K. Chakrabarty
e-mail: tapankumarchakrabarty@gmail.com

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_21

of the observed and the predicted values is minimized. Predictive modeling with least absolute error criteria is also referred to as conditional median modeling, or simply median regression. The least absolutes method scarcely gives specific algebraic answers [9]. The method of least squares [30] on the other hand chooses the parameters such that the sum of squared differences of the observed and the predicted values is minimized. Least squares method is the most commonly used method of predictive modeling and is also known as conditional mean function modeling. The method of quantile regression [1, 21] is simply a generalization of the median regression where one models the pth conditional quantile of the variable Y, the response variable as a linear function of the predictor variables. Estimation using these methods does not require any specific reference to the distribution of the stochastic term and hence are also referred to as semi-parametric.

In the second approach, the distribution of the stochastic component has been ruled by assuming or transforming to normal distribution with mean 0 and variance σ^2. The least squares method is often preferred as the principle of prediction since it is similar to using the method of maximum likelihood when the stochastic term is normally distributed. The estimates satisfy various optimum properties with this assumption. It is shown that using linear check function in quantile regression is similar to the use of maximum likelihood under the assumption that the errors of the regression model are from asymmetric Laplace distribution [10].

As it follows from above, in many situations the distributional form of the stochastic component and its parameter(s) is validly ignored. While in many prediction situations, assimilating the nature of the distribution of the stochastic element is principal to recognizing the model of the response variables conditionally on the values of the predictors.

In this paper, a quantile-based approach to supervised statistical learning will be employed where we shall model the stochastic element of the setup using quantile functions. This approach to regression will be termed as 'Parametric Regression Quantile' [11, Chap. 12]. The stochastic term will be modeled by using the quantile-based flattened logistic distribution (FLD) and its asymmetric generalization, viz. the quantile-based flattened generalized logistic distribution (FGLD).

The paper is organized under the following sections. Section 2 introduces the two quantile-based distributions and presents the matching L-moments estimators of their parameters. Section 3 describes the formulation of the predictive model or regression setup, Sect. 4 elaborates the method of fitting the predictive model and describes an algorithm to obtain the best-fitted model. The convergence of the algorithm is discussed in Sect. 4.1 and the goodness of fit of the parametric regression quantile model is discussed in Sect. 4.2. Parametric regression quantile (PRQ) models using the FLD and FGLD have been introduced in Sect. 5. Section 5.1 presents a simulation study to prove the validity of both the PRQ models. Applications based on real-life data have been shown in Sect. 6. Section 7 gives a short conclusion of the present study.

2 An Introduction to the Quantile-Based Distributions

The quantile-based flattened logistic distribution (FLD) introduced by [11] and discussed in detailed by [32] is defined by the quantile function (QF) in (2)

$$Q(p) = \beta\left[\ln\left(\frac{p}{1-p}\right) + \kappa p\right] \tag{2}$$

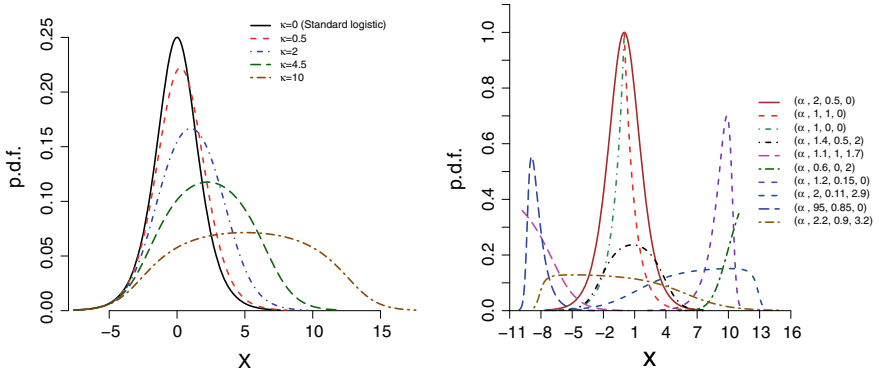
where $\beta \geq 0$ and $\kappa \geq 0$ are, respectively, the scale parameter and the peakedness parameter. The distribution has an interesting property; it is mesokurtic for $\kappa = 2.892927$, leptokurtic for $\kappa < 2.892927$ and platykurtic for $\kappa > 2.892927$.

We propose an asymmetric generalization of the distribution in (2) by replacing the standard logistic QF with the standard form of the QF of quantile-based skew logistic distribution [11, 35] and obtain the flattened generalized logistic distribution (FGLD) given in (3)

$$Q(p) = \beta\left[(1-\delta)\ln(p) - \delta\ln(1-p) + \kappa p\right] \tag{3}$$

where $\beta > 0$ is the scale parameter, $0 \leq \delta \leq 1$ is a skewness parameter, and $\kappa \geq 0$ is a parameter regulating the flatness of the peak of the distribution. The FLD is a special case of FGLD for $\delta = 0.5$. To have an idea of the shape of these distributions, density plots of these distributions have been shown in Fig. 1a, b, from which it can be clearly seen that the flatness of the peak increases with increase in the value of κ for both the distributions. Also, from Fig. 1b, it is clear that the FGLD has a flexible range of skewed shapes in addition to the flexibility in peakedness.

For a comprehensive study on the distributional properties and applications of the flattened logistic distribution, one can refer to [32]. We have derived the first four L-moments of the FGLD, and these are given by $\lambda_1 = \beta\left[(2\delta - 1) + \frac{\kappa}{2}\right]$,



(a) Density plots of FLD (b) Density plots of FGLD

Fig. 1 Possible shapes of FLD and FGLD

$\lambda_2 = \frac{\beta}{2}\left(1 + \frac{\kappa}{3}\right)$, $\lambda_3 = \frac{\beta}{6}(2\delta - 1)$ and $\lambda_4 = \frac{\beta}{12}$. The L-skewness ratio and the L-kurtosis ratio of the FGLD are given, respectively, by $\tau_3 = \frac{(2\delta - 1)}{3 + \kappa}$ and $\tau_4 = \frac{1}{2(3 + \kappa)}$.

Since δ lies between 0 and 1 and κ is positive, this clearly indicates that $-\frac{1}{3} \leq \tau_3 \leq \frac{1}{3}$. The plot of L-skewness ratio in Fig. 2 shows that for $\delta < 0.5$, the FGLD is negatively skewed, for $\delta > 0.5$, the FGLD is positively skewed and for $\delta = 0.5$, the FGLD is symmetric, this corresponds to the black line at $y = 0$ in Fig. 2b. The L-kurtosis ratio of the FGLD is $\frac{1}{6}$ for $\kappa = 0$, so $\tau_4 \leq \frac{1}{6}$. Also, it is clear that $\tau_4 > 0$ since $\kappa \geq 0$. Hence, $0 < \tau_4 \leq \frac{1}{6}$. The FGLD has an L-kurtosis ratio which is skewness invariant in the sense that it is not dependent on the value of δ, the skewness parameter. Figure 3 is a plot of the L-kurtosis ratio of the FGLD for various values of κ.
The black dotted line indicates the line where the value of y-axis is $\frac{1}{6}$. The L-kurtosis ratio of the FGLD lies below the black dotted line. It is clear from the figure that, with increases in the value of κ, the L-kurtosis ratio of the FGLD decreases. Thus, it can be seen that the FGLD has a great flexibility in terms of skewness and kurtosis. We avoid any other expansive studies of these distributions and mainly focus on the parameter estimation which will be required in the process of fitting the regression lines. The parameters of these distributions are obtained by using the matching L-moments estimation (MLM) method [16–18]. Let $\lambda_1, \lambda_2, \ldots$ be the L-moments of a distribution for which we are to estimate the parameters and let the sample L-moments be $l_1, l_2, \ldots$ which are calculated from a given data, then one can obtain set of equations by matching the rth order L-moment of the distribution with the

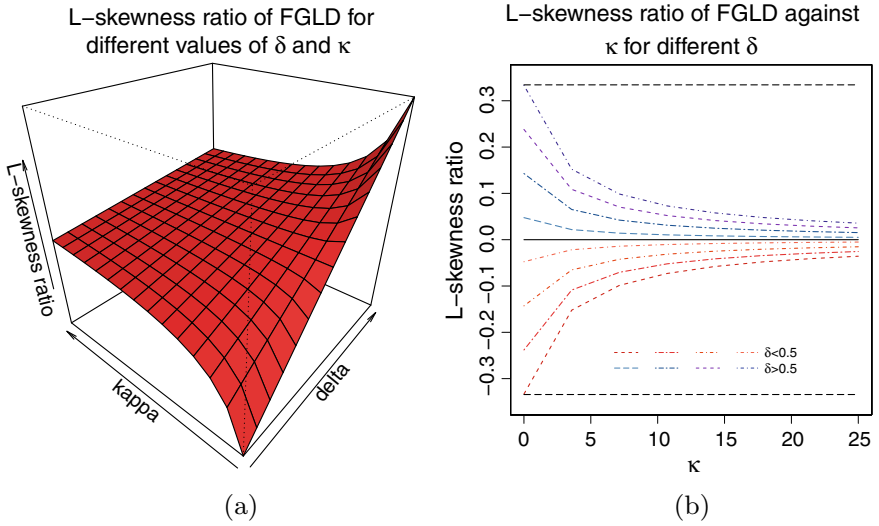


Fig. 2 Plot of L-skewness ratio against κ and δ

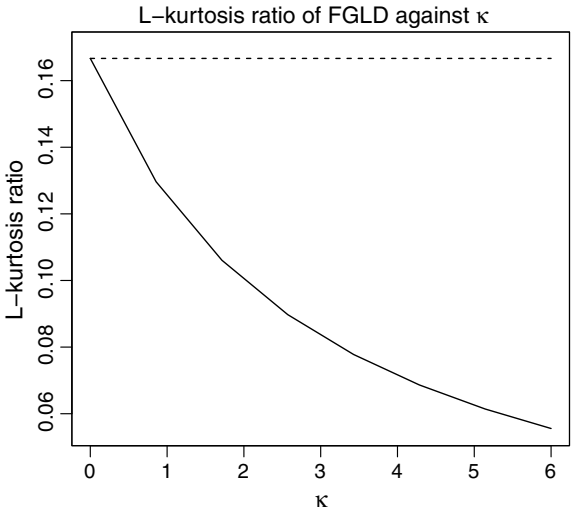


Fig. 3 Plot of L-kurtosis ratio for different values of κ

corresponding sample L-moment, the number of equations being equal to the number of parameters to be estimated. By solving these set of equations, the parameters of the distribution can be estimated. Thus, the matching L-moments estimators of the parameters of FLD [32] are given by (4).

$$\begin{aligned} \widehat{\beta} &= 6l_4 \\ \widehat{\kappa} &= \frac{l_2 - 6l_4}{l_4} \end{aligned} \tag{4}$$

Similarly, the matching L-moments estimators of the parameters of FGLD have been derived and given in (5)

$$\begin{aligned} \widehat{\beta} &= 12l_4 \\ \widehat{\delta} &= \frac{l_3}{4l_4} + \frac{1}{2} \\ \widehat{\kappa} &= \frac{l_2}{2l_4} - 3 \end{aligned} \tag{5}$$

3 Formulation of the Regression Model

Parametric regression quantile (PRQ) or quantile-based parametric predictive modeling first introduced by [11] is a method of regression where quantile functions are used to describe the predictive models. Quantile-based parametric regression analysis is not a new term and has been used in the works of many including, [5, 24, 28, 29, 34]. [19] has beautifully related the various works on such predictive models. Unlike

the classical linear regression, we model the quantiles of the dependent variable y as an additive model consisting of (i) deterministic component linear in predictors $\boldsymbol{x}$ and (ii) stochastic component expressed in quantile function form by $\eta S(p)$, where $S(p)$ is a quantile function which need not necessarily be symmetric, and η is the scale parameter. Thus the p-quantile of y given the specified $\boldsymbol{x}$, i.e., the regression quantile function of y on $\boldsymbol{x}$ is expressed by

$$Q_y(p|\boldsymbol{x};\lambda,\boldsymbol{\theta},\eta) = \lambda + \boldsymbol{\theta}'\boldsymbol{x} + \eta S(p) \tag{6}$$

where $\boldsymbol{\theta}$ is a vector of parameters or coefficients of the predictors, λ is the intercept of the line and η is the scale parameter. One can view (6) as the conditional quantile function of y for a given $\boldsymbol{x}$ and can be considered as conditional parametric distribution or model expressed as quantile function. Furthermore, the quantile function is standardized to have zero mean or median. In this article, we shall adjust the quantile function, $S(p)$ to have zero median, i.e., $S(0.5) = 0$, due to which the fitted regression line turns into Galton's median regression line. The model is thus a fully parametric linear regression or predictive model.

The distribution of the error term may contain nonlinear shape parameter(s), $\boldsymbol{\alpha}$ so that the regression quantile model takes the following form

$$Q_y(p|\boldsymbol{x};\lambda,\boldsymbol{\theta},\eta,\boldsymbol{\alpha}) = \lambda + \boldsymbol{\theta}'\boldsymbol{x} + \eta S(p,\boldsymbol{\alpha}) \tag{7}$$

Such models that include shape parameters are called semi-linear models. The variable $\boldsymbol{x}$ affects y through a linear relation $\lambda + \boldsymbol{\theta}'\boldsymbol{x}$, and the quantile function, $\eta S(p,\boldsymbol{\alpha})$ describes the error distribution in quantile form. The various parameters are λ, which is the intercept of the line; $\boldsymbol{\theta}$, the coefficients of the predictors; η, the scale; and $\boldsymbol{\alpha}$, the shape parameters of the distribution $S(p,\boldsymbol{\alpha})$. The virtue of this formulation as compared to the standard one is that all the parameters (equational and distributional) are distinctive in the equation that defines the model. [13, 14] have shown that distributions having explicit QFs can be used in regression in an effortless manner no matter whether the distributions are symmetric or not.

Using the ideas that go back to [6–8] and [26, 27] in respect to modeling, and [22] in respect to estimation, [11, 13, 14] formulated such approach to regression. Galton established the ideas of using the median regression, order statistics, quantile plots and the QF [see 11, 12, 15, 33]. Lloyd pioneered the idea of expressing the rth order statistic of a sample of size n of variable X, details on which can be explored from [14]. Thus, [11] symbolized that a regression quantile function at the most universal form can be written as

$$Q_y(p|\boldsymbol{x};\lambda,\boldsymbol{\theta},\eta,\boldsymbol{\alpha}) = Q(p;\boldsymbol{x},\boldsymbol{\phi}) \tag{8}$$

where the $\boldsymbol{x}$ and $\boldsymbol{\phi}$, respectively, denote the predictor(s) and parameter(s). This universal model can be fitted by minimizing the sum of absolutes of the distributional residuals. In the next section, the term 'Distributional Residual' is delineated and

the method of minimization is discussed. Quantile formulation to parametric regression analysis has several desirable properties that make it more likable in terms of regression. These properties can be explored in details from [11, 14].

4 Method of Fitting the Best Line of Regression

We have seen that a parametric regression quantile model can be regarded as a conditional distribution of the response y given a vector of predictors ($\boldsymbol{x}$) expressed as a quantile function. Thus, the method of fitting quantile distribution can be adapted to fitting quantile regression here. The method of maximum likelihood and simple extensions of distributional least squares (DLS) and distributional least absolutes (DLA) are the available methods for fitting models where the distributional element is explicit [11, 19]. Although in principle, the methods developed for fitting the distributions expressed in the form of density or distribution functions can be used to distributions in quantile functions form, and a few methods are designed specially to be used with quantile function models [11].

The standard maximum likelihood method requires an implicit conversion from quantile-based methods to density-based methods and so cannot be easily applied unless there is a clear density function. Moreover, the calculation of likelihood, $L(\boldsymbol{\phi})$ (or log-likelihood) requires derivation of the p from $y = Q(p; \boldsymbol{\phi})$ and the parameters, $\boldsymbol{\phi}$ are inherently present in the terms containing p, adding a fresh level of complexity to the situation and hence the method is difficult to implement.

In the remaining part of this section, we shall discuss the method of distributional least absolute which will be implemented for fitting the proposed model in Sect. 3. This method that equals to the ordinary least squares is based on the differences between the observations arranged in terms of magnitude and their predicted positions provided by the predicted model, termed as distributional residuals. The predicted positions may be the expectations of the order statistics under the model, called the rankits, which are used in DLS or the median values of population, called the median rankits, which are used in DLA. We now consider these in detail.

Let $(y_i, \boldsymbol{x}_i)$, $i = 1, 2, \ldots, n$ be the given set of data which is postulated as realization of the population modeled as

$$y = Q(p; \boldsymbol{x}, \boldsymbol{\phi}) = \lambda + \boldsymbol{\theta}'\boldsymbol{x} + \eta S(p, \boldsymbol{\alpha}) \tag{9}$$

In other words, the population distribution of the random variable Y conditional on $\boldsymbol{X} = \boldsymbol{x}$ is given by

$$Q_y(p|\boldsymbol{x}; \boldsymbol{\phi}) = \lambda + \boldsymbol{\theta}'\boldsymbol{x} + \eta S(p, \boldsymbol{\alpha}) \tag{10}$$

In order to fit this model using the given data, the method of DLA seeks to minimize the sum of the absolute values of the distributional residuals for values of the parameters $\boldsymbol{\phi}$. Having identified model (10), we create the fitted values of the deterministic

component $\tilde{y}_i = \widehat{\lambda} + \widehat{\boldsymbol{\theta}}' \boldsymbol{x}_i$; $i = 1, 2, .., n$ and obtain the initial residuals $\tilde{e}_i = y_i - \tilde{y}_i$. Let $r_i = \text{rank}(\tilde{e}_i;\ i = 1, 2, \ldots, n)$ and define median-p_i as $p_i^* = I_{0.5}^{-1}(r_i, n + 1 - r_i)$. Then, applying median rankits, the fitted error terms can be obtained as $\widehat{\eta} S(p_i^*, \widehat{\boldsymbol{\alpha}})$, so that the distributional residuals are given by

$$e_i* = y_i - (\tilde{y}_i + \widehat{\eta} S(p_i^*, \widehat{\boldsymbol{\alpha}})) = y_i - \widehat{\lambda} - \widehat{\boldsymbol{\theta}}' \boldsymbol{x}_i - \widehat{\eta} S(p_i^*, \widehat{\boldsymbol{\alpha}})\ ;\ i = 1, 2, \cdots, n \quad (11)$$

since, the fitted regression quantiles are $\tilde{y}_i + \widehat{\eta} S(p_i^*, \widehat{\boldsymbol{\alpha}});\ i = 1, 2, \cdots, n$. The criteria of fitting the regression setup is to search for estimates for which the sum of absolute distributional residuals is minimized; i.e., we are to minimize,

$$\sum_{i=1}^{n} |e_i*| = \sum_{i=1}^{n} |\tilde{e}_{(i)} - \widehat{\eta} S(p_i^*, \widehat{\boldsymbol{\alpha}})| \quad (12)$$

with respect to p_i^*. [31] has shown that, for a given value of η, the vector that minimizes (12) over $S(p_i^*, \boldsymbol{\alpha})$ is a vector whose ith element is the median of the ith order statistic of the estimated residuals $\tilde{e}_i$. The median of the ith order statistic of the estimated residuals can be obtained by using the uniform transformation rule [11] on the median of the ith order statistics from standard uniform distribution, which is given by $I_{0.5}^{-1}(r_i, n + 1 - r_i)$, where r_i is the rank of the ith observation. Thus, with $p_i^* = I_{0.5}^{-1}(r_i, n + 1 - r_i)$, $S(p_i^*, \widehat{\boldsymbol{\alpha}})$ gives the required median.

To obtain the best-fitted line of regression requires four steps. Step 1 obtains the initial fitted y's, $\tilde{y}$. From the fitted y, we obtain the initial residuals $\tilde{e} = y - \tilde{y}$ in step 2. The initial residuals are then fitted in step 3. In step 4, the final fitted y, i.e., $\widehat{y}$ are obtained by the addition of $\tilde{y}$ and the fitted initial residuals obtained in step 3.

To obtain the estimates of the parameters of the regression setup in (7), a search algorithm is required to be used. A search algorithm is an algorithm which solves the search problem. Using the ideas from [20], we propose a search algorithm based on comparisons. The proposed algorithm does not require a gradient. Thus, it can be used on functions that are not continuous or differentiable. This algorithm will be discussed by considering the simplest semi-linear model containing only one predictor, x and one shape parameter in the error term. The model is of the form,

$$Q_y(p|x;\ \boldsymbol{\phi}) = \lambda + \theta x + \eta S(p, \alpha) \quad (13)$$

Suppose that, $(y_i, x_i);\ i = 1, 2, \ldots n$, is a set of data, then a median regression model can be fitted to the data from which we can obtain the initial estimates say, $\widehat{\lambda}_1$ and $\widehat{\theta}_1$ of λ and θ of the deterministic part of the model. From the fitted model, we obtain the residuals which can be fitted to $\eta S(p, \alpha)$ by some appropriate quantile-based methods like method of matching L-moments estimation (MLM), method of matching quantiles estimation (MQE), etc. and initial guess $\widehat{\eta}_1$ and $\widehat{\alpha}_1$ of η and α, respectively, can be chosen for the stochastic model fitted to residual data. Thus, we have the initial guess of all the four required parameters. Using these initial estimates, initial set of distributional residuals is obtained. Let this be e_{1i}^*. Let s_1 be the sum

of the first set of absolute distributional residuals, i.e., $s_1 = \sum_i |e^*_{1i}|$. Now, the initial estimates, $\widehat{\theta}_1$, $\widehat{\eta}_1$ and $\widehat{\alpha}_1$ are fixed and we vary the value of $\widehat{\lambda}$, so that we have a sequence S_λ of $\widehat{\lambda}_j$'s; $j = 1, 2, 3, ...$ such that $\widehat{\lambda}_1$ is the median of the sequence. The sequence S_λ is the search space for the estimate of λ. Let the corresponding sets of sum of absolute distributional residuals be $s_j = \sum_{i=1}^{n} |e^*_{ji}|; j = 1, 2, 3, ...$ Among the obtained s_j's there will be one smallest s_j, say s_k. We replace the value of $\widehat{\lambda}_1$ by $\widehat{\lambda}_k$ such that $s_k = \min_{j \in n} s_j$. Thus, we have a new initial estimate for $\widehat{\lambda}$. Again fixing the value of $\widehat{\lambda}_1$, $\widehat{\eta}_1$ and $\widehat{\alpha}_1$ and varying value of $\widehat{\theta}$ we can choose the new estimate $\widehat{\theta}_1 = \widehat{\theta}_k$ such that $s_k = \min_{j \in n} s_j$. The same process is repeated for the other estimates. The whole process is repeated several times until we obtain the best set of estimates that minimize the sum of absolute distributional residuals. The algorithm has been implemented in "*R*". The step by step algorithm is given in Algorithm 1.

Algorithm 1 Search algorithm to obtain the estimates of best fit

1: Allocate reasonable initial estimates $\widehat{\lambda}_1, \widehat{\theta}_1, \widehat{\eta}_1$ and $\widehat{\alpha}_1$
2: Obtain $s_1 = \sum_i |e^*_{1i}|$
3: Define the search space by a sequence S_λ of estimates of λ such that λ_1 is the median of the sequence. Also let this sequence cover the confidence interval of λ_1.
4: For fixed $\widehat{\theta}_1, \widehat{\eta}_1$ and $\widehat{\alpha}_1$, obtain s_j corresponding to $\lambda_j; j = 1, 2, 3, ...$
5: Search for λ_j corresponding to the smallest distributional residual and replace λ_1 by this λ_j.
6: Define the search space for θ by a sequence S_θ of estimates of θ such that θ_1 is the median of the sequence. Also let this sequence cover the confidence interval of θ_1.
7: For fixed $\widehat{\lambda}_1, \widehat{\eta}_1$ and $\widehat{\alpha}_1$, obtain s_j corresponding to $\theta_j; j = 1, 2, 3, ...$
8: Search for θ_j corresponding to the smallest distributional residual and replace θ_1 with θ_j so obtained.
9: Repeat the algorithm to obtain other estimates.
10: Loop until the difference between the sum of absolute distributional residual in a few successive iterations continues to be negligible.

The algorithm can be extended to include several predictors and several shape parameters of the error term. The approximation is better for smaller common difference of the sequence.

4.1 Convergence of the Algorithm

The algorithm uses a simple idea that the estimates will be contained in a certain range of the initial estimates and of course, will not be far away from the initial estimates. For any reasonable initial estimate say, $\widehat{E}_i$ we can obtain a 95% or 99% confidence

interval $C_i = (a, b) \mid \widehat{E}_i \in (a, b)$, i being λ or θ or α, etc. Let S_i be the search space used in Algorithm 1 for ith estimate such that $C_i \subset S_i$, then we can always find an estimate E_i in S_i such that the sum of the absolute distributional residual is minimum among all the estimates in the defined search space. If the search space is large enough, then E_i will not surely be either of the bounds of S_i.

In the next step, we again consider the estimate obtained in the previous step as the mid value of sequence S_i, then the distributional residual will be minimum for an estimate that will surely be within the sequence, as the sequence has large bounds. Thus in every step, we obtain lesser distributional residual than the previous step and the estimate in the current step gets closer to the estimate obtained in the previous step. This continues in each iteration and after a certain number of iterations it happens that the same set of estimates continue to loop or in other words, the difference between the sum of absolute distributional residual in a few successive iterations continues to be negligible when the iteration finally breaks and returns the best set of estimates for a certain set of initial estimates. Thus the algorithm always converges.

The initial estimates can be guessed by any methods, method of least squares, method of least absolutes, graphical method, percentile method, etc.

4.2 Validation

In statistics, only visual method to claim goodness of fit is not enough. Some theoretical validation is required to prove the visual validation. There are several goodness of fit tests existing in the literature, some of which are explained briefly by [19]. In this article, the goodness of fit (GOF) of the proposed regression technique will be assessed through the Bonferroni I_{B} index [3, 19, 34]

$$I_{\mathrm{B}} = \frac{\sum_{i=1}^{n} |\widehat{y_i} - \mathrm{med}(\widehat{y})|}{\sum_{i=1}^{n} |y_i - \widehat{y_i}| + \sum_{i=1}^{n} |\widehat{y_i} - \mathrm{med}(\widehat{y})|} \tag{14}$$

where $\mathrm{med}(\widehat{y})$ is the median of the fitted $y's$. The index takes values in the closed interval [0, 1]. $I_{\mathrm{B}} = 1$ implies $y = \widehat{y}$, i.e., the model fits the observed data perfectly, whereas, $I_{\mathrm{B}} = 0$ implies $\widehat{y_i} = \mathrm{med}(\widehat{y})\ \forall\, i = 1, 2, 3, ..., n$, which does not provide any explanation of the data and hence is an indication of a non-informative model. In practice, the Bonferroni index of adequacy measures is the unit scale of the data, the reduction which takes proportionately in the absolute deviation owing to fitting the complete model.

5 Regression Quantile Models Using Quantile-Based Distributions

We propose two simple parametric regression quantile models; one of which is modeled with symmetric errors, while the other with asymmetric errors. Let the stochastic term in (7) be modeled by the FLD, then the regression quantile function of y on x (Model I) is expressed by

$$Q_y(p|x) = \lambda + \theta x + \beta\left[\ln\left(\frac{p}{1-p}\right) + \kappa p - \frac{\kappa}{2}\right] \tag{15}$$

where the term $\frac{\beta\kappa}{2}$ has been subtracted to ensure that the error distribution has 0 median. Thus, there are four parameters for which initial estimates need to be allocated to begin with the minimization process. The regression quantile model expressed in (15) is convenient for regression situations where the stochastic component is not only non-normal but also deviates from the mesokurtic property with a more prominent platykurtic property for $\kappa > 2.892927$.

As per Algorithm 1, the initial estimates of λ and θ can be obtained by fitting median regression model to the data. From this we obtain the initial set of residuals $\tilde{e}_{(i)}; i = 1, 2, .., n$. We can fit the FLD to this set of residuals using the method of matching L-moments estimation from which we can get the initial estimates of β and κ. Note that, the FLD used here lacks the location parameter and thus contain only two parameters which need to be estimated. We shall be using here the second and the fourth order L-moments to estimate the scale, β and the peakedness parameter, κ Then proceeding with the algorithm, the final estimates can be easily obtained.

The FLD being a symmetric distribution may not provide a good fit for regression models with asymmetric distributional residual. Hence, to fit such regression models, we propose modeling the stochastic term using the asymmetric generalization of FLD introduced in Sect. 2. The regression quantile function of y on x in this case (Model II) is expressed by

$$Q_y(p|x) = \lambda + \theta x + \beta\left[(1-\delta)\ln(p) - \delta\ln(1-p) + \kappa p\right] + \beta\left[-(2\delta - 1)\ln(2) - \frac{\kappa}{2}\right] \tag{16}$$

where the adjustment to confirm 0 median of the error distribution is ensured in the last two terms. Model II is convenient for regression situations where the stochastic component is not only non-normal, but also deviates from symmetry as well as the mesokurtic property with a more prominent flattened peak. In Model II, 5 parameters are required to be estimated to obtain the line of best fit. The same algorithm will be used here as in Model I.

5.1 Simulation Study

Model I and Model II have been applied to simulated regression datasets and the results obtained have been compared with mean and median regression on the basis of Bonferroni Index. 5 sets of regression data have been simulated with different sets of symmetric as well as asymmetric errors. The X values have been simulated from $N(\mu, \sigma)$ distributions by changing the μ and σ for each dataset. The errors have been simulated from the standard normal distribution, $N(0, 1)$; normal distribution, $N(0.5, 5)$; mixture normal distribution, $0.6N(3, 5) + 0.4N(0, 1)$; flattened logistic distribution, $FLD(0.5, 5.5)$; and skew normal distribution, $SKN(0, 1.2, 2.5)$. The constant, λ and coefficient, θ of regression model have been changed for each of these 5 datasets. The regression datasets were then obtained separately using these five sets of simulated errors and both Model I and Model II have been fitted to each of these datasets.

All the five sets of simulated regression data have also been fitted using mean regression (least square regression, LSR) and median regression (least absolute regression, LAR). The goodness of fit for each of the cases has been performed using I_B defined in (14). The result obtained has been summarized in Table 1, from which it can be clearly seen that in terms of Bonferroni Index, both Model I and Model II provide good fit to all the simulated data followed by LAR and LSR. Model II being the generalized version of Model I provides even better fit than Model I with a little higher value of I_B. LAR and LSR provide good fit only for the case of errors simulated from standard normal distribution. Tarsitan et al. [34] have shown much earlier that the method of fitting regression line by minimizing the sum of absolute distributional residuals provide much better fit than either the LSR or LAR. Thus, we can use these models to fit some real-life data. This will be done in the next section.

6 Empirical Applications

In this section, we apply the regression models in two real-life dataset. A set of data quoted by [23] provides the stopping distance (d) of vehicles as a function of their speed (s). It has been shown that by taking the square root of stopping distance as the predictor variable, a good fit is obtained. The data were given as an assignment problem at the end of [11, Chap. 12] where it was asked to propose a suitable model for using the stopping distance (d) as the predictor variable. The scatter plot for the dataset is depicted in Fig. 4.

We shall consider stopping distance as the independent variable. Hence, using Model in (15) the regression quantile function of speed, s on stopping distance d is expressed by

$$Q_s(p|d) = \lambda + \theta d + \beta\left[\ln\left(\frac{p}{1-p}\right) + \kappa p - \frac{\kappa}{2}\right] \tag{17}$$

Table 1 Fitted regression Model I and Model II to simulated datasets

Errors simulated from	θ and λ	Model	No. of iterations	$\widehat{\theta}$	$\widehat{\lambda}$	$\widehat{\beta}$	$\widehat{\kappa}$	$\widehat{\delta}$	I_{B}
$N(0, 1)$	$\theta = 1$ and $\lambda = 2.5$	Model I	4	0.9887895	2.508357	0.2632254	7.104061		0.9969461
		Model II	4	1.01179	2.497357	0.5334509	3.551531	0.5100878	0.9966908
		LSR		1.028	2.491				0.9425761
		LAR		1.028790	2.498357				0.9427546
$N(0.5, 5)$	$\theta = 1$ and $\lambda = 3$	Model I	5	1.48807	2.955226	1.824936	4.028261		0.9734774
		Model II	4	1.40807	2.895226	3.717871	2.00013	0.6079048	0.9784496
		LSR		1.588	3.005				0.7363468
		LAR		2.000926	2.887486				0.7300372
$0.6N(3, 5) + 0.4N(0, 1)$	$\theta = 2$ and $\lambda = 1$	Model I	6	3.537607	1.019744	0.7725235	7.423497		0.9556737
		Model II	7	3.548607	1.020744	1.574047	3.643749	0.4719869	0.9570194
		LSR		3.5676	0.9647				0.5338512
		LAR		3.3720245	0.9105299				0.5198202
$FLD(0.5, 5.5)$	$\theta = 0.5$ and $\lambda = 2.5$	Model I	8	0.7758293	2.436195	0.3809118	5.267012		0.9928972
		Model II	10	0.7598291	2.437195	0.7638237	2.600006	0.565925	0.993333
		LSR		0.8653896	2.50593				0.8851149
		LAR		0.7058293	2.486195				0.8853278
$SKN(0, 1.2, 3.8)$	$\theta = 2$ and $\lambda = 0.25$	Model I	8	2.774567	0.2873956	0.455201	0.5417663		0.941475
		Model II	3	2.791567	0.2533956	0.9134019	0.2663832	0.761912	0.9657754
		LSR		2.9346	0.2534				0.6926239
		LAR		2.8404865	0.2628917				0.7016454

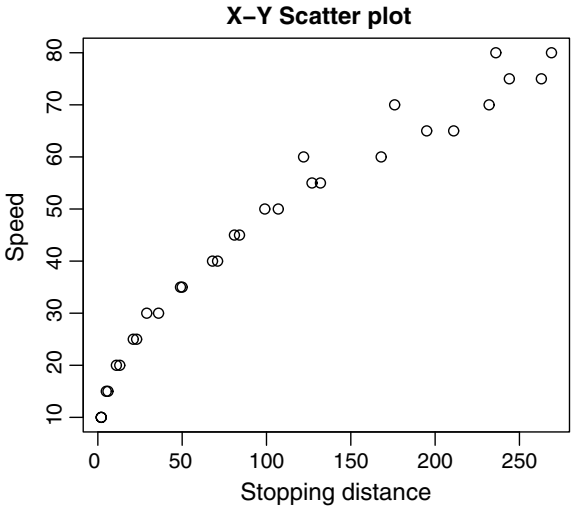


Fig. 4 Scatter plot for stopping distance data

whereas, using Model (16), the regression quantile function of speed, s on stopping distance d is expressed by

$$Q_{\mathrm{s}}(p|d) = \lambda + \theta d + \beta\left[(1-\delta)\ln(p) - \delta\ln(1-p) + \kappa p\right] + \beta\left[-(2\delta-1)\ln(2) - \frac{\kappa}{2}\right] \tag{18}$$

Using median regression, initial estimates of λ, θ are allocated. Using these initial estimates, we obtain the initial set of residuals to which we fit the FLD or FGLD using the method of matching L-moments and the estimates of β, κ and δ are obtained. Then using Algorithm 1, we obtain the estimates for which the sum of absolute distributional residual is minimum. GOF is performed and the Bonferroni index is obtained for each of the models. The estimated values corresponding to the minimum sum of absolute distributional residuals and the GOF index is given in Table 2. The fit versus observation plot and the distributional residual plot of stopping distance data fitted, respectively, using Model I and Model II are given in Figs. 5 and 6. It can be seen from both Figs. 5 and 6 that both the models (Model I and Model II) provide visually good fit to the stopping distance data which have been further verified by the results of the Bonferroni index shown in Table 2.

Another dataset associated with the transport of sulfite ions from blood cells suspended in a salt solution was taken from [2] which was originally collected by W. H. Dennis and P. Wood at the University of Wisconsin. The response variable is the chloride concentration which was measured (in %) over a period of about 8 min as a continuous curve generated from electrical potentials. This scatter plot for the

Table 2 Estimates for minimum sum of absolute distributional residuals for stopping distance data

Model used	Model para	Initial guess	CI of initial est	Final estimate	Iterations	$\sum_i \lvert e_i^* \rvert$	GOF index, I_B
Model I	λ	21.81818	(17.87410, 25.42958)	21.42828	7	17.81279	0.9692426
	θ	0.22727	(0.20543, 0.27441)	0.2272727			
	β	2.141514	(0.3685449, 3.914482)	2.125344			
	κ	2.974697	(0, 10.31329)	3.004797			
Model II	λ	21.81818	(17.87410, 25.42958)	21.81818	12	11.87461	0.9793248
	θ	0.22727	(0.20543, 0.27441)	0.2272727			
	β	4.283027	(0.6318217, 7.934233)	4.740127			
	δ	0.3059912	(0.02195908, 0.5900234)	0.2827012			
	κ	1.487348	(0, 5.21589)	1.245748			

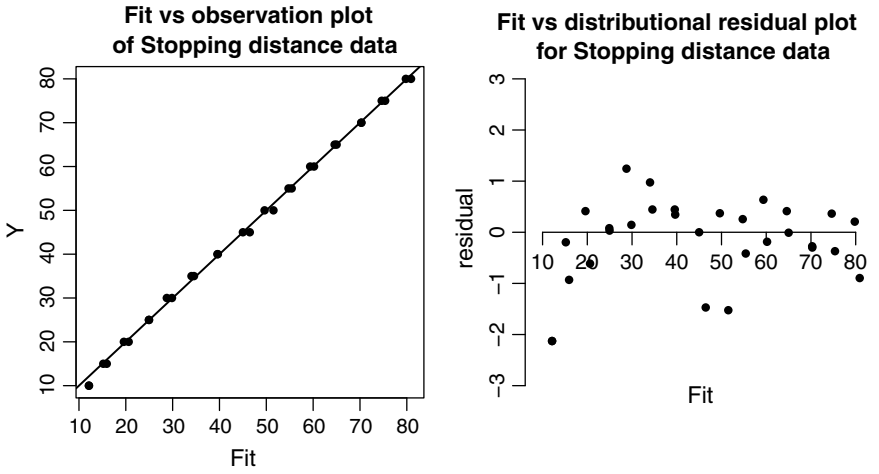


Fig. 5 Fit versus observation plot and distributional residual plot for stopping distance data using Model I

same is given in Fig. 7. The regression quantile function of chloride concentration, c on time interval t using Model I is expressed by

$$Q_c(p|t) = \lambda + \theta t + \beta \left[\ln\left(\frac{p}{1-p}\right) + \kappa p - \frac{\kappa}{2} \right] \tag{19}$$

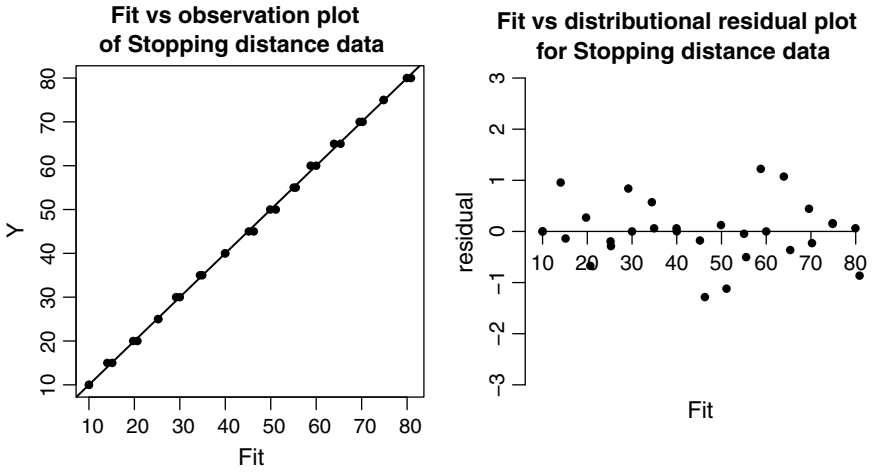


Fig. 6 Fit versus observation plot and distributional residual plot for stopping distance data using Model II

Fig. 7 Scatter plot for chloride concentration data

whereas, using Model in (16), the regression quantile function of chloride concentration, c on time interval t is expressed by

$$Q_c(p|t) = \lambda + \theta t + \beta\left[(1-\delta)\ln(p) - \delta\ln(1-p) + \kappa p\right] + \beta\left[-(2\delta-1)\ln(2) - \frac{\kappa}{2}\right] \tag{20}$$

Semi-linear regression model defined in (15) and (16) has been fitted to the dataset by following the Algorithm in 1. The result is summarized in Figs. 8 and 9 and Table 3.

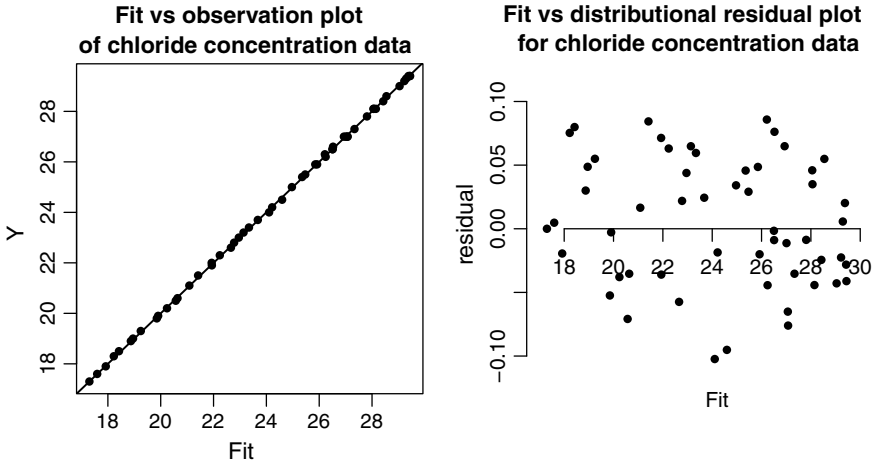


Fig. 8 Fit versus observation plot and distributional residual plot of chloride concentration data using Model-I

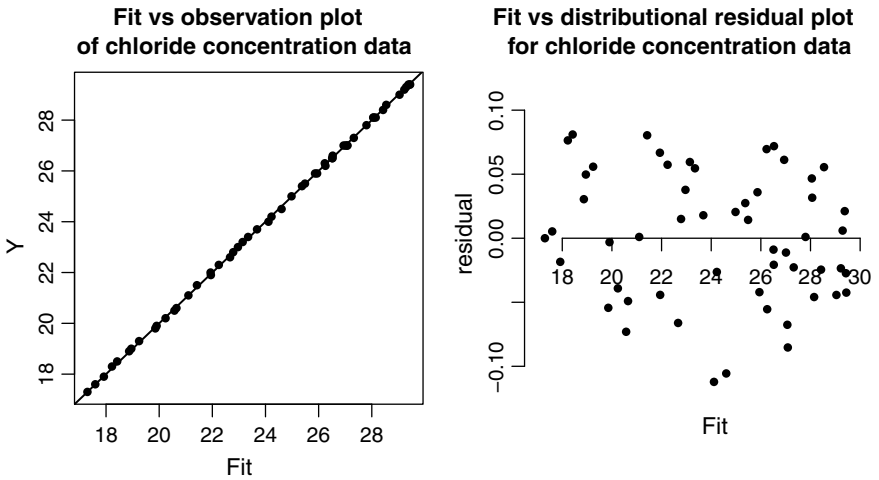


Fig. 9 Fit versus observation plot and distributional residual plot of chloride concentration data using Model II

Figures 8 and 9 clearly shows visually a good fit of the regression models to the chloride concentration data which is further proved by the Bonferroni index obtained in Table 3.

Table 3 Estimates for minimum sum of absolute distributional residuals for chloride concentration data

Model used	Model para	Initial guess	CI of ini est	Final estimate	Iterations	$\sum_i \lvert e_i^* \rvert$	GOF Index, I_B
Model I	λ	12.43621	(11.76666, 13.38273)	12.43231	3	2.292669	30.9868492
	θ	2.31034	(2.16798, 2.45951)	2.310345			
	β	0.09448149	(0.006804,0.182159)	0.09490149			
	κ	10.4551	(0, 26.40079)	10.45862			
Model II	λ	12.43621	(11.76666, 13.38273)	12.43521	3	2.265447	0.9870043
	θ	2.31034	(2.16798, 2.45951)	2.310345			
	β	0.188963	(0.03031217, 0.3476138)	0.192063			
	δ	0.51536	(0.153823, 0.8768971)	0.51534			
	κ	5.227551	(0, 13.13523)	5.226151			

7 Conclusion

Quantile-based parametric regression modeling is relatively recent in the history of statistics. In this paper, we have considered parametric regression quantile using flexible additive stochastic components modeled by flattened logistic distribution and its asymmetric generalization. The advantage of these models is to capture the flexibility in the distribution of the stochastic term. The models have been fitted using the criteria of minimizing the sum of distributional least absolutes, for the computation of which a search algorithm is proposed and successfully implemented. The superiority of the proposed models over the classical mean and median regression models has been demonstrated using simulated data. Two empirical examples of regression have been fitted by using both Model I and Model II and the goodness of fit test is performed using Bonferroni index. Both the regression models are shown to give visually and theoretically a good fit to simulated as well as empirical data. The approach to regression is both practical and instructive as a tool of data analysis in predictive analysis contexts and provides a robust approach to supervised learning.

Acknowledgements The Department of Science and Technology (DST), Government of India is acknowledged by the first author for the financial support provided to her through the DST-INSPIRE fellowship with award no. IF130343.

References

1. Bassett G, Koenker R (1982) An empirical quantile function for linear models with iid errors. J Am Stat Assoc 77(378):407–415
2. Bates DM, Watts DG (1988) Nonlinear regression analysis and its applications. Wiley
3. Bonferroni CE (1940-1941) Elementi di statistica generale. Universita Commerciale L. Bocconi, Milano
4. Boscovich RJ (1757) De litteraria expeditione per pontificiam ditionem, et synopsis amplioris operis, ac habentur plura ejus ex exemplaria etiam sensorum impressa. Bononiensi Sci Artum Inst Atque Acad Comment 4:353–396
5. Dean B, King AR (2009) Versatile regression: simple regression with a non-normal error distribution. In: Third anual applied statistics education and research collaboration (ASEARC) conference, 7-8. Dec. Newcastle
6. Galton F (1883) Inquiries into the human faculty and its development. Macmillan and Company, London
7. Galton F (1886) Regression towards mediocrity in hereditary stature. J Anthropol Inst G B Irel 15:246–263
8. Galton F (1889) Natural inheritance, vol 42. Macmillan and Company, London
9. Gentle JE (1977) Least absolute values estimation: an introduction. Commun Stat-Simul Comput 6(4):313–328
10. Gilchrist WG (1997) Modelling with quantile distribution functions. J Appl Stat 24(1):113–122
11. Gilchrist WG (2000) Statistical modelling with quantile functions. Chapman & Hall/CRC, Boca Raton, Florida
12. Gilchrist WG (2005) Galton misrepresented. Significance 2(3):136–137
13. Gilchrist WG (2007) Modeling and fitting quantile distributions and regressions. Am J Math Manag Sci 27(3–4):401–439

14. Gilchrist WG (2008) Regression revisited. Int Stat Rev 76(3):401–418
15. Hald A (1998) A history of mathematical statistics from 1750 to 1930. Wiley, New York
16. Hosking JRM (1986) The theory of probability weighted moments. Technical Report RC 12210, IBM Research Division, Yorktown Heights, New York
17. Hosking JRM (1990) L-moments: analysis and estimation of distributions using linear combinations of order statistics. J R Stat Society Ser B (Methodol) 52(1):105–124
18. Hosking JRM (1996) Some theoretical results concerning l-moments. Technical Report RC 14492 (revised), IBM Research Division
19. Karian ZA, Dudewicz EJ (2010) Handbook of fitting statistical distributions with R. CRC Press, Boca Raton, Florida
20. Knuth DE (1998) Sorting and searching, 2nd edn. the art of computer programming, vol. 3
21. Koenker R, Bassett G (1978) Regression quantiles. Econ J Econ Soc 46(1):33–50
22. Lloyd E (1952) Least-squares estimation of location and scale parameters using order statistics. Biometrika 39(1–2):88–95
23. Mosteller F, Fienberg SE, Rourke RE (2013) Beginning statistics with data analysis. Addison-Wesley Publishing Company, Massachusetts
24. Muraleedharan G, Lucas C, Soares CG (2016) Regression quantile models for estimating trends in extreme significant wave heights. Ocean Eng 118:204–215
25. Narula SC, Wellington JF (1982) The minimum sum of absolute errors regression: a state of the art survey. Int Stat Rev/Rev Int Stat 50(3):317–326
26. Parzen E (1979) Nonparametric statistical data modeling. J Am Stat Assoc 74(365):105–121
27. Parzen E (2004) Quantile probability and statistical data modeling. Stat Sci 19(4):652–662
28. Perri PF, Tarsitano A (2007) Partially adaptive estimation via quantile functions. Commun Stat-Simul Comput 36(2):277–296
29. Perri PF, Tarsitano A (2008) Distributional least squares based on the generalized lambda distribution. In: Brito P (ed) COMPSTAT 2008. Proceedings in computational statistics, 18th symposium, vol 2. Springer, Berlin, pp 341–348
30. Plackett RL (1972) Studies in the history of probability and statistics. xxix: the discovery of the method of least squares. Biometrika 59(2):239–251
31. Rao CR (1988) Methodology based on the l_1 norm in statistical inference. Sankhyā: Indian J Stat, Ser Ain Stat-Simul Comput 50(3):289–313
32. Sharma D, Chakrabarty TK (2018) The quantile-based flattened logistic distribution: some properties and applications. Commun Stat-Theory Methods. https://doi.org/10.1080/03610926.2018.1481966
33. Stigler SM (1986) The history of statistics: the measurement of uncertainty before 1900. Harvard University Press, Cambridge
34. Tarsitano A, Perri P et al (2007) A quantile function-based approach for least absolute deviations. In: Fifth conference on complex models and computational intensive methods for estimation and prediction. Cleup, Padova, pp 386–391
35. Van Staden PJ, King RA (2015) The quantile-based skew logistic distribution. Stat Probab Lett 96:109–116

Chapter 22
Feature Learning Using Random Forest and Binary Logistic Regression for ATDS

Chandra Shekhar Yadav and Aditi Sharan

1 Introduction

At first, the Internet effectively interconnected laboratories occupied with government inquire about, and since 1994 it has been extended to serve giant number of clients and vast number of purposes everywhere throughout the world. As per the International Business Machines Corporation promoting cloud study, more than 90% of the information on the Web has been made post-2016. Individuals, organizations, and gadgets have all moved toward becoming information manufacturing plants siphoning out fantastic measures of data to the Web each day. This influx of information leads to an information overload and, this phenomenon is known as Information-Glut or Data-Smog.

The "Information Overload" problem generally happens whenever the input stream of a system surpasses system's process capability. Due to business constraints, the decision makers and executives have genuinely restricted psychological processing limits. Subsequently, when information overload happens, then certainly, a contraction in decision-making quality arises and that limits the business by many factors. Information overload can be partially dealt with by the summarization of text information. Since significant information available on Web is in text format, therefore, an effective text document summarizer system can represent text data concisely on the Web.

Author in [1–4] has seen summary as "a text that is produced from one or more texts, which conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually significantly less than that." An

C. S. Yadav (✉)
Standardisation Testing and Quality Certification, MeiTy, New Delhi, India
e-mail: chandrtech15@gmail.com

C. S. Yadav · A. Sharan
SC & SS, Jawaharlal Nehru University, Delhi, India
e-mail: aditishran@jnu.ac.in

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_22

android application of text summarization in domain of News summarization is "inshorts App."

Based on nature of the summarizer system, it can classify in two types: first extractive summarization and second abstractive summarization. This extractive-based summarization relies on statistical and lexical features, such as frequency of words present in a single or multiple document, sentence length, sentence position, combination of Local_Weights and Global_Weights like term frequency and inverse document frequency score are statistical features and lexical features contains selection of candidate word as cue words, verb, nouns, digits, or number's occurrence in the text document, bold letter words, uppercase, centrality, sentiment of words and sentences, aggregate similarity, etc. The target of feature dependent summarizer system is either as single or as a combination of various strategies is to find the most informative sentences that may include in the text summary. Proposed work in this paper deals with extractive summarization.

2 Related Work

Authors in [5] introduced a position-based statistical model, author in [6] proposed frequency-based model, author in [7] iterated that sentences appear in the starting and last paragraph more important compare to those part of between these, authors in [8] relatively assigned more weight to the first ten para and last five para in a text, authors in [9] proposed MEAD system based on centroid, TF-IDF, position, authors in [10] considered keyword-occurrence as a feature, author in [11] suggested importance of sentences varies on users and user writing styles, and author in [12] proposed six features like keywords headline word, word frequency, cue words, and number of cue words present in the sentence, length and location of sentence. Latent semantic analysis (LSA)-based work proposed by authors [13], in that they implemented all previous LSA-based model for ATS, they proposed a new entropy-based summary evaluation measure.

Authors in [14] presented a legitimate portrayal for the usage of lexical chain by exploit of Roget Thesaurus, whereas by motivation of their work [15] built up the early text content analyzer for summarization utilizing lexical chain. Authors in [16] came with a new model for lexical chains development wherein competitor words are chosen dependent on POS labeling such as nouns, and WordNet1.7 utilized for implementation. In work [17] also assumed noun as a competitor unites for lexical chain design and the connection among nodes, i.e., sentences. Authors [18] were pursued the examination of [15] to create lexical chains, and they suggested an algorithm to design a lexical chain creation in linear time.

Another work proposed in direction of extractive summarization by authors [1, 2] in that summary extraction was depended on two types of features one is syntactic features and other is semantic-based features. First time they considered overall sentiment of sentence (evaluated based on all entities' sentiment present in that sentences) for sentence's score evaluation. In other work the "SentiWordNet" used by authors

[19, 20] to find sentence's features, syntactic-dependent summarizer system was presented by author [21], the "centroid-based" summarizer system by authors [22]. Yet their methodology is equivalent as proposed by authors [23]. These summarizer systems result can utilize in various domains like accounting, research, efficient usage of results, and accelerated of Web search by search engines as presented by authors [24].

3 Background

This section deals with detailed discussion about random forest, logistic regression, and dataset presented.

3.1 Random Forest

Random forest uses machine learning and predictive modeling-based techniques. This is kind of ensemble classifier that is used in decision modeling. According to [25] "Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest." It is a versatile algorithm generates arbitrary number of simple trees and fit to perform both tasks like regression and classification. This technique does an indirect feature selection and takes care about missing data using previous observations. Another feature of this technique is, due to huge no. of trees born/population in this technique, therefore, in this technique issue of overfitting does not occur and this is convergent every time.

The response of trees is based on a set of predictors values those are chosen with replacement, with the same kind of distribution and independently for each tree in the forest. If "M" is the number of inputs, then the optimal predictor size is $(\log_2 M + 1)$. For classification problem, the function computes the extent, in which the average counts of votes for the legitimate class outstrip the average count from remaining class available in the outcome variable. For regression problems, random forest is designed by growing trees are in capacity to produce a numeric response. At last, the prediction of the random forest is calculated by mean of all tree's predictions, that is given by Eq. 1. Here in Eq. 1 the index-K runs over all the distinct trees present in the forest. The mean error is given by Eq. 2.

$$\text{Random Forest Predictions} = \frac{1}{K} \sum_{K=1}^{K} K\text{th tree responce} \tag{1}$$

$$\text{Mean error} = (\text{observed} - \text{tree responce})^2 \tag{2}$$

The random forest algorithm/flow diagram is presented by Fig. 1, proposed by authors [26].

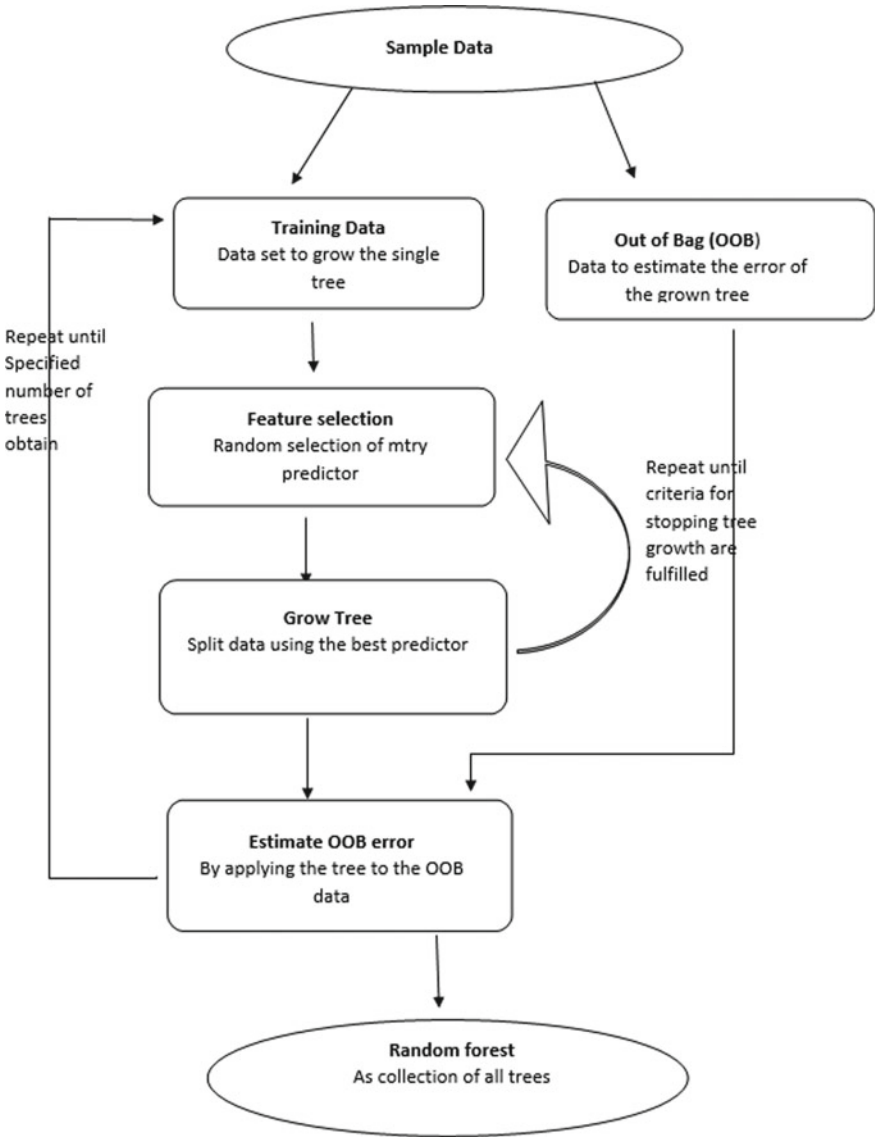


Fig. 1 Random forest algorithm

3.2 Binary Logistic Regression

The model is represented in Eqs. 3 and 4. The "Y" is a depended feature and a kind of binary response variable, "Y_i" is True/1, in case the predefined condition is satisfied. However, Y_i is False or 0. X is set of independent features as $X = (X_1, X_2, \ldots X_k)$, "$X_i$" is continuous, discrete, or mix. Here, x_i is the predicted value of independently for ith observation. β_0 and β_1 need to be learned. Equation 5 shows how parameter estimation is done by logistic regression.

$$\pi_i = \Pr(Y_i = 1 | X_i = x_i) = \frac{e^{(\beta_0 + \beta_1 \cdot x_i)}}{1 + e^{(\beta_0 + \beta_1 \cdot x_i)}} \tag{3}$$

$$\log it(\pi_i) = \log\left(\frac{\pi_i}{1 - \pi_i}\right) = (\beta_0 + \beta_1 x_i) \tag{4}$$

By maximizing given Eq. 5, parameter estimation can be performed,

$$L(\beta_0, \beta_1) = \prod_{i=1}^{N} \pi_i^{yi} (1 - \pi_i)^{n_i - y_i} = \prod_{i=1}^{N} \frac{e^{\{yi \cdot (\beta_0 + \beta_1 \cdot x_i)\}}}{1 + e^{(\beta_0 + \beta_1 \cdot x_i)}} \tag{5}$$

3.3 DUC 2002 Dataset

The DUC 2002 datasets are taken from the Text Retrieval Conference (TREC) disks used in track of question answering, i.e., in that conference track number TREC-9. This dataset contains data from various newspapers like, "Wall Street Journal" (from 1987 to 1992), "AP newswire" (from 1989 to 1990), "San Jose Mercury News" (1991), "Financial Times" (from 1991 to 1994), LA Times from disk-5, and FBIS from disk-5. Approximately, every set has ten documents, comprise with at least ten words, no upper limit of document is defined. For individual document, there is a single text document abstract with approximately the length of a 100 words. The multi-document abstract is categorized into four classes according to 200, 100, 50, and 10 words long. All documents are categorized into four classes, and the classes are defined like.

1. Single catastrophic event that is made inside the 7 day window.
2. Single event that is based on any domain and made in the 7 day window.
3. Multiple particular occasions of a solitary sort (no restriction on the time window).
4. Documents that contain for the most part historical data about a solitary person.

4 Proposed Approach

The previous proposed approach is given by linear combination of different features that are represented by Model-1 and Eq. 6. Now, modified approach model is given by Eq. 7, and weighted approach followed now. In this section, a logistic regression and random forest model are used for weight learning w_1, w_2, w_3, w_4, and w_5.

$$\begin{aligned}\text{Model} - 1 = (&\text{Location} + \text{TF} - \text{IDF} + \text{Centroid} \\ &+ \text{Aggregate Similarity} + \text{Sentiment})\end{aligned} \tag{6}$$

$$\begin{aligned}\text{Model} - 2 = &(w_1 \times \text{location}) + (w_2 \times \text{TF} - \text{IDF}) + (w_3 \times \text{Centroid}) \\ &+ (w_4 \times \text{Aggregate similarity}) + (w_5 \times \text{sentiment})\end{aligned} \tag{7}$$

Features used for different model are considered from various studies. The position-based feature considered from [5] given by Eq. 8, aggregation similarity feature taken from [27] presented by Eqs. 9, 10 and similarity between sentences find by cosine-based similarity score shown in Eq. 11, frequency-based feature from [6] presented in Eq. 12, centroid feature from [28] presented in Eq. 13, sentiment feature (score5) defined by authors [29] presented in Eq. 14.

$$\text{Score}(S_i) = 1 - \frac{i-1}{N} \tag{8}$$

$$\text{Similarity}(S_i,\ S_j) = \sum_{k=1}^{n} W_{ik} \cdot W_{jk} \tag{9}$$

$$\text{Score}(S_j) = \sum_{j=1, i \neq j}^{n} \text{Similarity}(S_i,\ S_j) \tag{10}$$

$$\text{Similarity}(S_i,\ S_j) = \frac{\sum_{k=1}^{m} W_{ik} \cdot W_{jk}}{\sqrt{\sum_{k=1}^{m} W_{ik}^2 \sum_{k=1}^{m} W_{jk}^2}} \tag{11}$$

$$W_i = \text{TF}_i \times \text{IDF}_i = \text{Tf}_i \times \log \frac{ND}{\text{d}f_i} \tag{12}$$

$$C_i(S_i) = \sum_{w} C_{w,i} \tag{13}$$

$$\text{Score5} = \sum_{i=1}^{n} \left|\text{Sentiment}(\text{Entity}_i)\right| \tag{14}$$

5 Experiment and Results

In this experiment best feature weights combination is find put using machine learning techniques and used to get optimal results. To get the optimal feature weights, different machine learning-based model such as supervised algorithm, random forest, and logistic regression has been used. Here, internal estimates monitor strength, error, correlation, and this is used for evaluation to get variable importance. This experiment was executed on DUC dataset. Figure 2 shows correlation between independent features used in the experiment. Figure 2 is self-explanatory of the fact that there is less correlation or no correlation between features. Since our feature weight learning approach based on regression and classification, therefore, we require to check dependency between variable. In case two variables are dependent and showing a linear relationship between it, then it is not suitable for regression analysis. Our, scatter plot matric is not showing linear relationship between variables, so we can use these methods now.

This data partitioned in two sets for training and testing. Developed models (random forest and logistic regression) are used to find best feature weight combination. Table 1 suggested features weight, and corresponding model accuracy is presented. K-fold cross-validation technique is considered to check the accuracy of regression model. In which data is divided into K segments, one segment is used for validation, and remaining "$K - 1$" left for model training, and this practice repeated up to K times. In our implementation, the K is decided to ten. For random forest, we placed 90% observations into training dataset and the remaining 10% of the observations keep for test data. We have generated five hundred trees, and the number of predictors

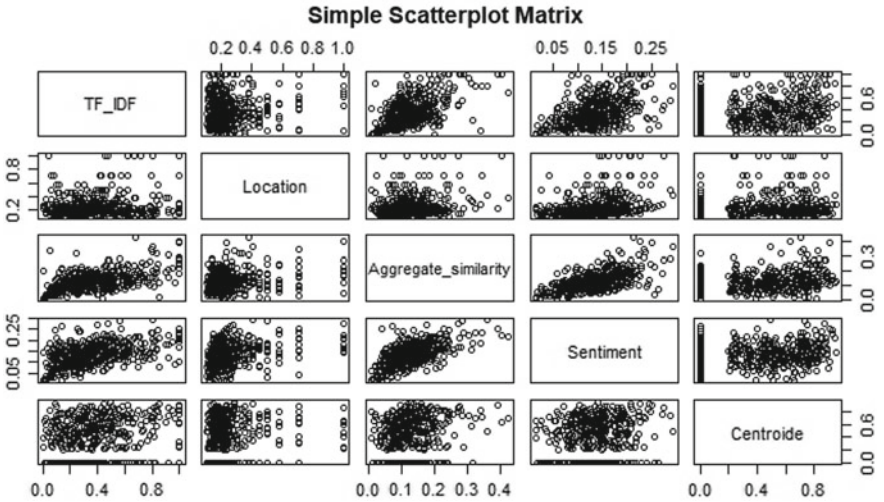


Fig. 2 Correlation between different features used for summarization task

Table 1 Computed optimal feature weight using random forest and logistic regression

Implemented model	(Feature-1) location	(Feature-2) TF-IDF	(Feature-3) centroid	(Feature-3) aggregate similarity	(Feature-4) sentiment	Model accuracy (%)
Random forest	20.1211	22.5391	12.60026	22.8421	39.1893	76.34
Logistics regression	0.4469	0.8440	0.4545	0.1878	0.4978	79.4

to randomly sample at each split is two. The accuracy of models and feature weights found out are shown in Table 1.

From, Table 1, it is evident logistic regression-based model gives weight for sentiment score is 39.18 which is the highest among all features weight, and it emphasizes the fact that the significance of sentiment-based feature is more significant than other features. In logistic regression-based model, the sentiment feature weight is second highest. In these two developed models logistic regression and random forest, features weights computed by logistic regression are $w_1 = 0.4446$, $w_2 = 0.8440$, $w_3 = 0.4545$, $w_4 = 0.1878$, $w_5 = 0.4978$ w.r.t. feature name location, term frequency–inverse documents frequency, centroid, aggregate similarity, and sentiment, with model accuracy 79.4%. So, both the models, Model-1 and Model-2, are represented by corresponding Eqs. 6 and 7.

In this section, these models are implemented on DUC dataset. In this experiment, the previous model (Model-1) and new model (Model-2) were implemented and later feature weight learned by regression-based model. Results are shown in Table-2 and Table-3. Model performance can be decided by R-recall, P-precision, and F-score. The 95% confidence interval signifies the probability of falling values of precision-P, recall-R, and F-score in the given domain. Figure-3 tells comparative performance analysis based on content-based ROUGE measure. Tables 2 and 3 and Fig. 3 present a proof of effective changes in results of Model-2 (Eq. 7) over Model-1 (Eq. 6).

6 Concluding Remark

This work exhibited an amalgam model for objective "Automatic Text Document Summarization," which depends over a direct blend of various statistical and semantic measures. In this approach, statistical estimates like sentence position, term frequency–inverse document frequency, i.e., TF-IDF, centroid, and semantic methodology that performing slant investigation of sentences. The semantic methodology depends on deep-level analysis, i.e., word-level examination. The sentence's sentiment score is computed by aggregation of estimation score of each entity present in that sentence. Every entity present in sentence has any of three polarities from class positive, neutral, and negative. In the event, where entity's sentiment score is negative, then that score is multiplied by minus one to regard it as positive value. The

Table 2 Showing performance of Model-1

	Precision	Recall	*F*-score	(95%-confidence interval *P*)		(95%-confidence interval *R*)		(95%-confidence interval *F*)	
ROUGE—1	0.2644	0.3164	0.2879	0.2136	0.3170	0.25620	0.3757	0.2319	0.3444
ROUGE—2	0.0800	0.0959	0.0872	0.0464	0.1142	0.0559	0.1373	0.0507	0.1241
ROUGE—3	0.0419	0.0503	0.0457	0.0168	0.0692	0.0205	0.0827	0.01840	0.0751
ROUGE—4	0.0238	0.0288	0.0261	0.0068	0.0448	0.0084	0.0546	0.0075	0.0491
ROUGE—L	0.2294	0.2748	0.250	0.1798	0.2818	0.2161	0.3350	0.1967	0.3058
ROUGE—W-1.2	0.0898	0.2015	0.1242	0.0708	0.1090	0.1585	0.2442	0.0978	0.1501
ROUGE—S*	0.0652	0.0930	0.0765	0.0411	0.090	0.0592	0.1273	0.0484	0.1052
ROUGE—SU*	0.0714	0.1013	0.0836	0.0465	0.0969	0.0664	0.1360	0.0546	0.1128

Table 3 Showing performance of Model-2

	Precision	Recall	*F*-score	(95%-confidence interval *P*)		(95%-confidence interval *R*)		(95%-confidence interval *F*)	
ROUGE—1	0.2743	0.3267	0.2980	0.2238	0.3293	0.2695	0.3853	0.2445	0.3553
ROUGE—2	0.0777	0.0922	0.0843	0.0463	0.1125	0.0555	0.1319	0.0504	0.1209
ROUGE—3	0.0379	0.0449	0.0411	0.0144	0.0637	0.0179	0.0758	0.0160	0.0689
ROUGE—4	0.0217	0.0260	0.0237	0.0058	0.0424	0.0071	0.0512	0.0064	0.0463
ROUGE—L	0.2394	0.2854	0.2602	0.1891	0.2934	0.2289	0.3444	0.2070	0.3159
ROUGE—W-1.2	0.0904	0.2012	0.1247	0.0724	0.1091	0.1633	0.2409	0.1002	0.1495
ROUGE—S*	0.0679	0.0956	0.0792	0.0436	0.0940	0.0644	0.1280	0.0516	0.1076
ROUGE—SU*	0.0734	0.1030	0.0855	0.0486	0.1003	0.0709	0.1361	0.0574	0.1144

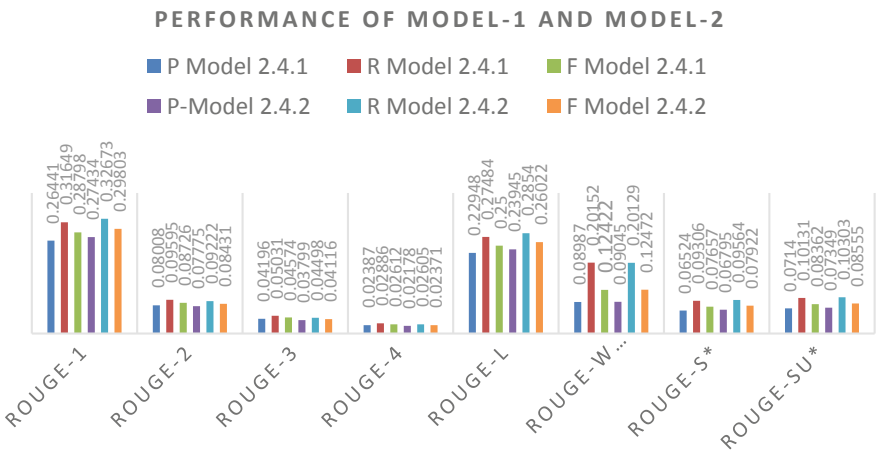


Fig. 3 Comparative performance of proposed Model-1 (presented in Eq. 6) and proposed Model-2 (presented in Eq. 7)

purpose behind performing this is to choose a sentence that has much strong sentiment that may come from both classes negative and positive, and both the classes have equivalent significance in our work. To compute the total score or importance of a sentence, all the computed scores for each sentence were added and the highest informative sentence based on score was identified. Further, if in case cosine similarity score between the sentences and in the text summary is below the defined limit, then to achieve desired diversity, it should be considered for summary text. The stopping condition of this process is summary length hindrance.

In the proposed model, the data is classified into two parts, one part to train the model and second part to test the proposed model over the trained model. A feature weighted approach is proposed using regression and random forest. The regression-based proposed model performs better and given more accuracy. Since logistic regression producing better model, therefore, feature weights are given according to the regression model. Another experiment is done by feature weight analysis to show the model improvement.

References

1. Yadav CS, Sharan A (2015) Hybrid approach for single text document summarization using statistical and sentiment features. Int J Inf Retr Res 5(4):46–70
2. Yadav CS, Sharan A, Kumar R, Biswas P (2016) A new approach for single text document summarization, vol 380
3. Lin C-Y (2004) Rouge: a package for automatic evaluation of summaries. Text Summ Branches Out
4. Radev DR, Hovy E, McKeown K (2002) Introduction to the special issue on summarization. Comput Linguist 28(4):399–408

5. Baxendale PB (1958) Machine-made index for technical literature—an experiment. IBM J Res Dev 2(4):354–361
6. Luhn HP (1958) The automatic creation of literature abstracts. IBM J Res Dev 2(2):159–165
7. Edmundson HP (1969) New methods in automatic extracting. J ACM 16(2):264–285
8. Kupiec J, Pedersen J, Chen F (1995) A trainable document summarizer. In: Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval, pp 68–73
9. Radev DR, Blair-Goldensohn S, Zhang Z (2001) Experiments in single and multi-document summarization using MEAD. Ann Arbor 1001:48109
10. Ganapathiraju K, Carbonell J, Yang Y (2002) Relevance of cluster size in MMR based summarizer: A Report 11-742: Self-paced lab in Information Retrieval
11. Ouyang Y, Li W, Lu Q, Zhang R (2010) A study on position information in document summarization. In: Proceedings of the 23rd international conference on computational linguistics: posters, pp 919–927
12. Tofighy SM, Raj RG, Javad HHS (2013) AHP techniques for Persian text summarization. Malays J Comput Sci 26(1):1–8
13. Yadav C, Sharan A (2018) A new LSA and entropy-based approach for automatic text document summarization. Int J Semant Web Inf Syst 14(4):1–32
14. Morris J, Hirst G (1991) Lexical cohesion computed by thesaural relations as an indicator of the structure of text. Comput Linguist 17(1):21–48
15. Barzilay R, Elhadad M (1997) Using lexical chains for text summarization
16. Gurevych I, Nahnsen T (2005) Adapting lexical chaining to summarize conversational dialogues. In: Proceedings of the recent advances in natural language processing conference, pp 287–300
17. Kulkarni AR, Apte SS (2014) An automatic text summarization using lexical cohesion and correlation of sentences. Int J Res Eng Technol 3(06):285
18. Silber HG, McCoy KF (2000) An efficient text summarizer using lexical chains. In: Proceedings of the first international conference on Natural language generation, vol 14, pp 268–271
19. Boudia MA, Hamou RM, Amine A (2016) A new approach based on the detection of opinion by SentiWordNet for automatic text summaries by extraction. Int J Inf Retr Res 6(3):19–36
20. Rautray R, Balabantaray RC, Bhardwaj A (2015) Document summarization using sentence features. Int J Inf Retr Res 5(1):36–47
21. Haggag MH (2013) Semantic text summarization based on syntactic patterns. Int J Inf Retr Res 3(4):18–34
22. Mohamed SS, Hariharan S (2016) A summarizer for tamil language using centroid approach. Int J Inf Retr Res 6(1):1–15
23. Erkan G, Radev DR (2004) Lexrank: graph-based lexical centrality as salience in text summarization. J Artif Intell Res 22:457–479
24. Takale SA, Kulkarni PJ, Shah SK (2016) An intelligent web search using multi-document summarization. Int J Inf Retr Res 6(2):41–65
25. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
26. Boulesteix A, Janitza S, Kruppa J, König IR (2012) 'Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics', *Wiley Interdiscip*. Rev Data Min Knowl Discov 2(6):493–507
27. Kim J-H, Kim J-H, Hwang D (2000) Korean text summarization using an aggregate similarity. In: Proceedings of the fifth international workshop on Information retrieval with Asian languages, pp 111–118
28. Radev DR, Jing H, Sty M, Tam D (2004) Centroid-based summarization of multiple documents. Inform Process Manag 40:919–938
29. Mani I, Maybury MT (1999) Advances in automatic text summarization reviewed by Mark Sanderson University of Sheffield 26(2):280–281

Chapter 23
MLPGI: Multilayer Perceptron-Based Gender Identification Over Voice Samples in Supervised Machine Learning

Meenu Yadav, Vinod Kumar Verma, Chandra Shekhar Yadav, and Jitendra Kumar Verma

1 Introduction

A human voice is a good carrier of information for others and as a universal way of contact on this planet. When sound is produced from any sound source, then it causes the air particles to vibrate so that the sound can be produced. This continuous vibration of air particles raises the echo. Ears of human identify the sound depending on the rate of speech signals and identify the voice of different persons. In real life, each human being has a distinct voice from everyone else's because of the different features of voice. A person can identify the other person voice or can distinguish between voices of numbers of people by recognizing the pitch, frequency and tone of the voice. Even with the voice person can guess the age and gender of the voice. We can utilize the application speech (sound) processing in various areas like voice dialing, phone conversation, doing route identification of call, controlling domestic gadget, conversion of speech (sound) to text and vice versa, lip style identification and synchronization, automation systems, etc. [1].

M. Yadav (✉) · V. K. Verma
Sant Longowal Institute of Engineering and Technology, Sangrur, Punjab, India
e-mail: meenu_yadav99@rediffmail.com

V. K. Verma
e-mail: vinod5881@gmail.com

C. S. Yadav
Standardization Testing and Quality Certification, MeitY, New Delhi, India
e-mail: chandrtech15@gmail.com

J. K. Verma
Department of Computer Science and Engineering, Amity School of Engineering and Technology, Amity University Haryana, Gurugram (Manesar), Haryana, India
e-mail: jitendra.verma.in@ieee.org

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_23

2 Gender Classification Related Work

The gender classification system is a system which identifies the gender of the speech files. Here, we are classifying the work of recognizing the gender in two parts in which first is based on different features extracted from speech files and other is the gender classification using different parameters means just not only based on speech but also other parameters of person. So the work which is done in the first method of using the different features is explained further which are extracted from speech files. In [2], author has proposed a new technique perceptual linear predictive (PLP). This technique mainly has three concepts to derive the auditory spectrum: (i) spectral resolution of critical-band, (ii) curve for equal-loudness and (iii) power law of intensity-loudness. Study of PLP was computationally useful and represents a voice (speech) in low dimensions. In [3], author has proposed that RASTA speech processing reduces the sensitivity of recognizers. RASTA does this with the help of band-pass filtering time trajectories of logarithmic parameters of speech. In [4], author discussed the correspondence between feature of RASTA and the nature of the identification models and also the relation of their features to delta and also to mean subtraction and cepstral. Features, like linear predictive codes (LPC), PLP-relative spectra (PLP-RASTA), perceptual linear prediction (PLP), Mel frequency cepstral coefficients (MFCC), etc., were extracted from the speech files. They used methods like support vector machine (SVM), dynamic time warping (DTW), vector quantization (VQ), hidden Markov model (HMM) and which are utilized for classification and identification or say recognition [5]. In [6], author proposed a LID system which identifies the language and also classifies the genders from the speech files. Machine learning classifiers were applied to the data matrix which is a supermatrix of all the features (attributes) which were processed out from speech (sound) files. In [7], author designed a language identification system using machine learning algorithms based on language models. They used six machine learning (ML) methods on their dataset and did comparison among their results and they found that the best-optimized result was given by random forest (RF). In [8], author proposed a new direct access framework for the identification system of speaker's voice (sound). This process has phases like: voice preprocessing, feature extraction of speaker's sound, normalizing feature, selecting feature, speaker modeling, direct accessing method and speaker matching. The MFCC feature-based accuracy level is 94.38% and the gender-based accuracy for classification of speaker is up to hundred percent based on flatness, pitch, roll of and brightness features. In [9], author used multilayer perceptron (MLP), Gaussian mixture model (GMM), learning vector quantization (LVQ) and vector quantization (VQ) classification model in their work along with Mel frequency cepstral coefficients (MFCC). Now the papers of gender classification on the basis of different parameters of person. In [10], author designed a computational model and evaluated this model on two things like first is linear support vector machines (SVM) and nonlinear feed-forward (FFNN), long short-term memory (LSTM) and convolutional neural network (CNN) models. Sometimes error occurred in automatic speech recognition (ASR) so variant recurrent neural network (V-RNN)-based models are

applied for detecting error and classification of error types [11]. They have shown the V-RNN which was trained on the recommended set of features which acts as a productive classifier for the ASR error detection. But in [12], author has first time predicted the gender of a person directly from the same binary iris code that could be used for identification. In [13], author worked with the number of convolutional neural network (CNN) to train them individually on predefined patches with partial cropping and many image resolutions. And here in the proposed work, we required a deep learning method based on multiple large patches.

3 Proposed Work

The following section contains the explanation regarding the dataset, preprocessing and the proposed architecture of the machine learning model which consist of the multiple classifiers like random forest, J48, bagging, REP Tree, multilayer perceptron, simple logistic, logistic and SMO for classifying the gender by analyzing the speech signals.

The paper contains the description of the following: used dataset, preprocessing of speech files, extraction of the feature of sound, concept of classifying voice and used models for classification (Fig. 1).

3.1 Used Dataset

The work which is done in this paper uses the dataset proposed in MSLT corpus [14]. The datasets have been referenced from Internet source like Kaggle. So it is a collection of general voice and some files from the predefined dataset. The dataset contains the speech signals of 1500 individuals with 680 males and 820 females their age is between 15 and 50 because it contains the voice files of child, adult and senior citizens. This dataset has ten languages-based speech files like English, German, Japanese, French, Spanish, Mandarin, Korean, Russian, Italian and Hindi.

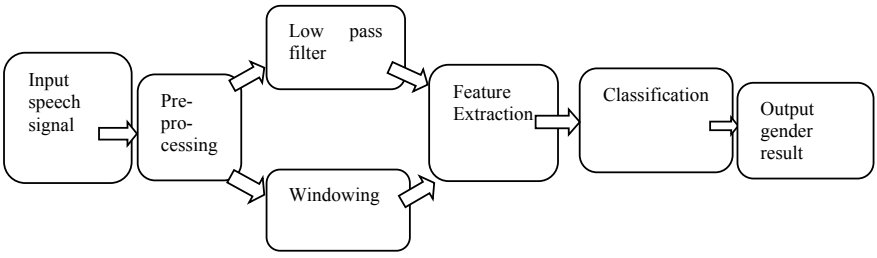


Fig. 1 Gender classification system

3.2 Voice Preprocessing

In this system, voice preprocessing is done by two methods. First is low pass filtering and the second is rectangular windowing. Windowing is a process to make the subsets of the larger set for analysis and processing. Rectangular window simply truncates the dataset before and after windowing but does not modify the contents of the window at all [15]. The rectangular window is defined by Eq. (1):

$$w_{\mathrm{R}}(n) \triangleq \begin{cases} 1, & -\frac{M-1}{2} \leq n \leq \frac{M-1}{2} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $w_{\mathrm{R}}(n)$ is a window function, M denotes window order and n is a number of voice samples.

3.3 Voice Features Extraction

The essential component of speech identification or say recognition is the extraction of the features of voice (sound). To recognize the speech, it is advised to find the accurate features of the voice for an accurate result. Features are of two types: Spectral features are like zero-crossing rates, short-time energy, etc., which are computed in this system. Temporal features are time-based domain features. These features have easy physical interpretation and easy to extract them. Temporal features contain fundamental frequency, F, M, cepstra, spectra, pspectrum, and lpcas.

The fundamental frequency represents the lowest frequency of the speech signal. It is the inverse of the length of pitch duration. The fundamental frequency is denoted as F_0 (F-zero). It is correlated with a pitch. The formula of fundamental frequency F_0 is given by Eq. (2):

$$F_0 = \frac{1000}{\tau_{\mathrm{ms}}} \tag{2}$$

where τ_{ms} represents length of pitch.

The concept of ZCR of speech signals states the changes in the value of a sign with a signal. It is the rate when the signal reaches to zero from positive value and then to negative value or from negative to zero to positive. One use of zero-crossing rate is in voice activity detection (VAD), it identifies whether the voice is present in speech or not [16]. The concept of ZCR is defined in Eq. (3):

$$\mathrm{ZCR} = \sum_{m=-\infty}^{\infty} |\mathrm{sgn}[s(m)] - \mathrm{sgn}[s(m-1)]| \tag{3}$$

In Eq. (3), sgn [] is sign function that can be defined as in Eq. (4):

$$\text{sgn}[x] = \begin{cases} 1, & (x \geq 0) \\ -1, & (x < 0) \end{cases} \tag{4}$$

Here, the speech signal is $s(m)$. A justifiable gist is that in case low ZCR, it voiced and in case high ZCR that is unvoiced speech.

The energy is related with speech or voice with the varying time nature so the speech is automatically processed to analyze the variation of energy with time or specifically we can say the energy is related to the short domain (area) of sound (voice) [16]. The overall energy of an energy (power) signal is shown by Eq. (5):

$$E(n) = \sum_{i=1}^{N} x_n^2(i) \tag{5}$$

where N represents for frame length, $x(i)$ for original speech signal and short-time energy $E(n)$ of frame n. To determine the cepstral features of the speech signal, first, we have to find the spectral features of the speech signal. The cepstral features can be determined by taking inverse Fourier transform of the logarithmic of earlier spectral feature. Equation is given by (6):

$$\text{Power cepstrum of signal } \left| F^{-1}\left\{ \log\left(|F\{f(t)\}|^2 \right) \right\} \right|^2 \tag{6}$$

3.4 Classification

Gender classification by speech analysis basically it aims to guess or determine the gender of the speaker by analyzing different parameters of the voice sample. The extraction of voice features is the key task in the speech recognition system. The meaning of feature extraction is that bring out those features of speech that help the system in identifying the language of speaker and gender of speaker. These features of sound (speech) are called as patterns. These patterns include the training set and they are used to apply the classification algorithm. The advantage of this algorithm in identifying pattern of saying of specific gender is feature matching along with extraction of features [1].

3.5 Used Models for Classification

There are many experiments which are performed using varying techniques of machine learning. Those machine learning techniques are random forest, J48, bagging, SMO, multilayer perceptron, logistic, simple logistic and REP Tree.

Random Forest Random forest is also called random decision trees which are used for regression, classification and other tasks which operates by constructing the decision trees of solutions for the individual problem at the training time of data and gives output in the form of classes (classification) [17]. The posterior probability that x which is a point lies to class c ($c = 1, 2, \ldots, n$) may denoted by $P\left(c|v_j(x)\right)$, shown in Eq. (7),

$$P\left(c|v_j(x)\right) = \frac{P\left(c, v_j(x)\right).}{\sum_{l=1}^{n} P\left(c_l, v_j(x)\right).} \tag{7}$$

The discriminant function is defined by Eq. (8):

$$g_c(x) = \frac{1}{t} \sum_{j=1}^{t} \hat{P}\left(c|v_j(x)\right) \tag{8}$$

The decision rule is to assign x to class c for which $g_c(x)$ is the maximum.

J48 The C4.5 algorithm is used for building the decision trees in WEKA tool as a classifier which is called as J48. J48 has a full name weka.classifiers.trees.J48. It is mainly used for classification of instances.

Bagging Bagging is used to reduce the variance and retaining the bias. It happens when we compute the average of predictions in different spaces of the input feature space [18]. Equation is defined in (9):

$$\varphi_B(x) = av_B\varphi\left(x, L^{(B)}\right) \tag{9}$$

L is a learning set, x is a input, φ_B denotes aggregation, $L^{(B)}$ is a repeated bootstrap samples.

SMO SMO is an algorithm which is used to solve the quadratic programming problems that occurred during the training period of SVM. SMO computes the constraints on the multipliers for two Lagrange multipliers [19]. SMO find the minimum along the direction of the constraint by computing the second Lagrange multiplier shown in Eq. (10):

$$\alpha_2^{\text{new}} = \alpha_2 + \frac{y_2(E_1 - E_2)}{\eta} \tag{10}$$

where ith training error E_i, Lagrange multiplier α_2, second derivative η of the objective function along the diagonal line, minimum constrained is found by clipping shown in Eq. (11):

$$\alpha_2^{\text{new, clipped}} = \begin{cases} H & \text{if } \alpha_2^{\text{new}} \geq H \\ \alpha_2^{\text{new}} & \text{if } L < \alpha_2^{\text{new}} < H \\ L & \text{if } \alpha_2^{\text{new}} \leq L \end{cases} \tag{11}$$

Let $= y_1 y_2$, the value of α_1 is computed from the new clipped α_2 defined in Eq. (12):

$$\alpha_1^{\text{new}} = \alpha_1 + s\left(\alpha_2 - \alpha_2^{\text{new, clipped}}\right) \tag{12}$$

REP Tree REP Tree is a fast decision learner tree. It constructs a decision tree or regression with the help of the obtained information and pruned it with the help of reduced error pruning with back fitting.

Multilayer Perceptron Multilayer perceptron can be defined as a class of feed-forward artificial neural network which consists of three or more layers in its figure. Basically, it has three layers as input layer, hidden layer and output layer. Multilayer perceptron uses supervised technique for training the model that is backpropagation. The degree of error in an output node j in the nth data point (training example) by $e_j(n) = d_j(n) - y_j(n)$, where d is the "target value and y is the value produced by the perceptron [20]. The node weights may be adjusted based on corrections that minimize the error in the entire output" that given by Eq. (13):

$$\varepsilon(n) = \frac{1}{2}\sum_j e_j^2(n) \tag{13}$$

Using gradient descent, the change in each weight is shown by Eq. (14):

$$\Delta w_{ji}(n) = -\eta \frac{\partial \varepsilon(n)}{\partial v_j(n)} y_i(n) \tag{14}$$

where y_j is "the output of the previous neuron and η is the learning rate, that ensure that the weights without oscillations quickly converge into the response. The calculated derivative to be depend on the induced local field y_j, which itself varies. It is easy to prove that for an output node this derivative can be simplified to,"

$$-\frac{\partial \varepsilon(n)}{\partial v_j(n)} = e_j(n)\emptyset'\left(v_j(n)\right) \tag{15}$$

Here, $\emptyset'$ "is the derivative of the activation function described above, which itself does not vary. The analysis is more difficult for the change in weights to a hidden node, but it can be shown that the relevant derivative is,"

$$-\frac{\partial \varepsilon(n)}{\partial v_j(n)} = \emptyset'\left(v_j(n)\right)\sum_k -\frac{\partial \varepsilon(n)}{\partial v_k(n)} w_{kj}(n) \tag{16}$$

Simple Logistic The condition for using simple logistic is that we should have one nominal variable and one measurement variable, and we want to know does the modification in the measurement variable bring the change in nominal variable [21]. A multi-class LLR score fusion model is presented in Eq. (17):

$$\bar{s}_{\mathrm{nl}} = \sum_{i=1}^{C} a^i s_{\mathrm{nl}}^{(i)} + \beta_l = \alpha^{\mathrm{T}} s_{\mathrm{nl}} + \beta_l \tag{17}$$

β_l and α_i are the biases and regression coefficients, respectively.

Logistic Regression Logistic regression is a supervised machine learning classification algorithm. In this function input values (x) are joined with the help of weights and the coefficients values to get the output values (y). It is shown by Eq. (18):

$$y = \mathrm{e}^{\frac{(b0+b1\times x)}{\left(1+\mathrm{e}^{(b0+b1\times x)}\right)}} \tag{18}$$

where y denotes the predicted output, $b0$ represents bias or intercept term and $b1$ denotes the coefficient of the single input value (x). There is an associated b coefficient (a constant real value) in our input data for every column that must be learned by training data.

4 Experiment and Result

The experiment of the gender classification system of speech files is introduced in this chapter. First, we should have a database of speech files, speech files would be preprocessed so that the files do not contain any noise and pause between the speeches. After voice preprocessing, nine voice features were extracted in this designed system as described in Fig. 2. The files will go into the RASTA feature extraction process where RASTA features are extracted and as well as RASTA filters the audio files. We have extracted nine features from the four main classes of features which are waveform, RASTA, spectrum and cepstrum. The nine features which are extracted named as fundamental frequency, lpcas, zero-crossing rates, F, M, spectrum, short-term energy, spectra and cepstra. The complete hierarchical structure of speech features is explained by Fig. 2.

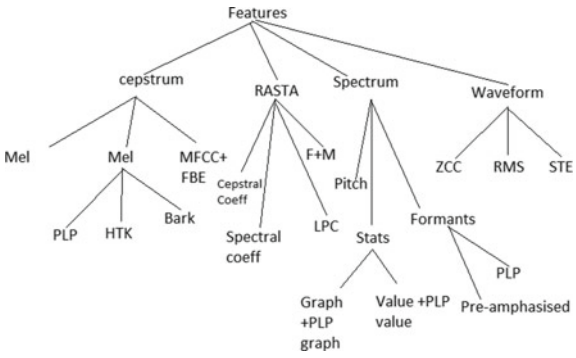


Fig. 2 Hierarchical structure of speech features

The filtered, silence free and noise-free speech files then enframed using the rectangular windowing process. Each frame has a size of 20 and 10 ms overlap (if the last frame has a size of less than 20 ms then this frame will be discarded). Then, these frames sent to the spectrum and cepstrum feature extraction phase as shown in Fig. 3. In this phase, remaining all the features were extracted. After getting all these features using the feature extraction process, all the features were arranged in a "feature structure" also called "feature matrix". In this, columns showing the type of features which is extracted from speech files and rows represent the number of speech files. At last, we did the column normalization of the feature matrix so that feature values lie between 0 and 1. This column normalization step increases the accuracy of classification.

We have taken 1500 speech files of ten languages: English, German, Japanese, French, Spanish, Mandarin, Korean, Russian, Italian and Hindi as used in [10], but we used three different languages from those ten languages which are used in [10]. In our dataset, we have taken each 150 files of ten languages which is a combination

Table 1 General information about the speech corpus

Language	# of speech files	# of male files	# of females files	Size of files in s (min $\leq x \leq$ max)
English	150	50	100	$2 \leq x \leq 60$
German	150	60	90	$30 \leq x \leq 60$
Japanese	150	60	90	$1 \leq x \leq 3$
French	150	140	10	$2 \leq x \leq 10$
Spanish	150	100	50	$2 \leq x \leq 10$
Mandarin	150	80	70	$40 \leq x \leq 60$
Korean	150	50	100	$1 \leq x \leq 4$
Russian	150	100	50	$5 \leq x \leq 9$
Italian	150	70	80	$59 \leq x \leq 90$
Hindi	15	3	12	$47 \leq x \leq 60$

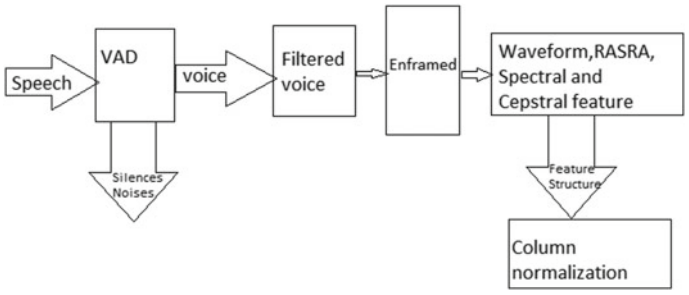


Fig. 3 Feature extraction process

of male and female files. Table 1 shows the general information on our used dataset. We have used speech files with random length in seconds.

We have experimented with 1500 speech files of ten languages that are English, German, Japanese, French, Spanish, Mandarin, Korean, Russian, Italian and Hindi. It is the combination of males and females speech files in which 680 are of male speech files and 820 are of female speech files. We have extracted nine features of speech files like cepstra, spectra, F, M, zero-crossing rate, fundamental frequency, short-time energy, pspectrum, lpcas. We get a large feature matrix of (1500 $\times$ 9), 1500 shows the number of speech files and 9 are features from the input files. Then, apply the machine learning classifiers. We have applied eight machine learning methods as used in [10] so that we can compare our results on our database. These machine learning methods are random forest, bagging, J48, SMO, REP Tree, multilayer perceptron, simple logistic and logistic. Table 2 shows the accuracies, *F*-measure, ROC area and PRC curve achieved by each machine learning method on dataset. We have tested the dataset using tenfold cross-validation methods. We got a surprising result that simple logistic acquired 87.33% which is highest among the eight machine learning methods used in this paper in all aspects like accuracy, *F*-measure, ROC area and PRC curve. It is better than the previous work in this area because the last paper [10] about language and gender classification have achieved only 76.29% accuracy using random forest. And the results are clearly shown in Table 2.

The eight machine learning methods which we have used in this work are the best methods for classification purpose. A good machine learning model should have a precision-recall curve as near to the line equal to one. Precision shows the correct values of class A amid all values that are categorized as A. Recall is the fraction of values that are categorized as class A, amid all the values which in fact have class A. Figures 4 and 5 describe the PR curves for male and females using best four accuracies of machine learning methods.

Table 2 Results obtained using six machine learning methods

ML method	*F*-measure	ROC area	PRC area	Accuracy
Random Forest	86.7	94.5	94.7	86.7
J48	81.3	84.5	80.4	81.3
Bagging	84.6	90	88.4	84.7
SMO	77.4	77.6	71.7	77.3
REP Tree	79.3	82.4	79.6	79.3
Logistic	82.6	87.9	87	82.7
Multilayer perceptron	83.3	93.3	93.6	83.3
Simple logistic	87.3	88	86.5	87.3

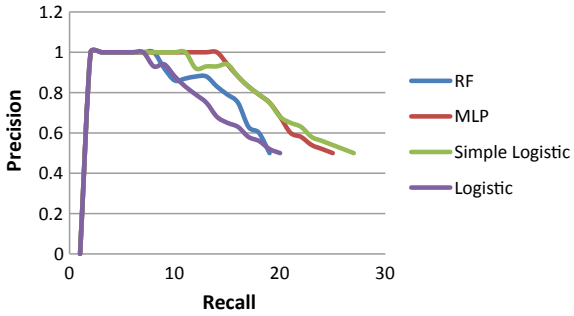


Fig. 4 PR curves of males for best four ML methods

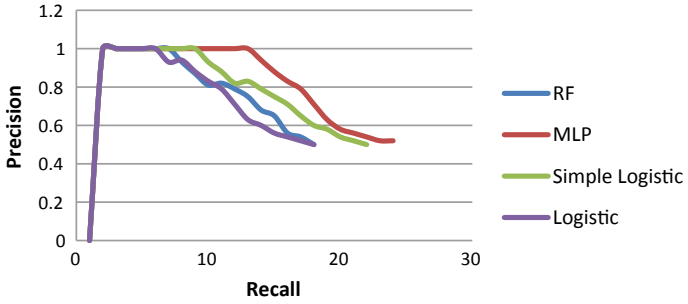


Fig. 5 PR curves of females for best four ML methods

5 Conclusion and Future Scope

In this research work, we have proposed an LID system which is different in many manners from the previous LIDs. (1) We computed four feature sets of speech files that are cepstrum features, spectrum features, RASTA features and waveform features. Unlike the previous work in this area, we have computed features directly from our speech files without any changes to intermediate features. (2) We achieved high results using the 20 ms frame size which is 500 times smaller than the frame size used in the previous LIDs experiments. (3) Experiment using eight machine learning methods which are not yet used in the previous work as per our best knowledge.

Future scope in this direction may be, performing experiments which enable comparative performance as a function of the frame size, and performing analysis by large speech files from other languages.

References

1. Hong Z (2017) Speaker gender recognition system. Master's thesis, degree programme in wireless communications engineering, University of Oulu
2. Hermansky H (1990) Perceptual linear predictive (PLP) analysis of speech. J Acoust Soc Am 87(4):1738–1752
3. Hermansky H, Morgan N (1994) RASTA processing of speech. IEEE Trans Speech Audio Process 2(4):578–589
4. Hermansky H, Morgan N, Hirsch H-G (2002) Recognition of speech in a additive and convolutional noise based on RASTA spectral processing, vol 2, pp 83–86
5. Dave N (2013) Feature extraction methods LPC, PLP and MFCC in speech recognition. Int J Adv Res Eng Technol 1(Vi):1–5
6. HaCohen-Kerner Y, Hagege R (2017) Language and gender classification of speech files using supervised machine learning methods. Cybern Syst 48(6–7):510–535
7. HaCohen-Kerner Y, Hagege R (2015) Automatic classification of spoken languages using diverse acoustic features, pp 275–285
8. Heryanto H, Akbar S, Sitohang B (2014) A new direct access framework for speaker identification system. In: Proceedings of 2014 international conference on data and software engineering (ICODSE) 2014
9. Djemili R (2012) A speech signal based gender identification system using four classifiers
10. Chowdhury SA, Stepanov EA, Danieli M, Riccardi G (2019) Automatic classification of speech overlaps: feature representation and algorithms. Comput Speech Lang 55:145–167
11. Errattahi R, EL Hannani A, Hain T, Ouahmane H (2019) System-independent ASR error detection and classification using Recurrent Neural Network. Comput Speech Lang 55:187–199
12. Tapia JE, Perez CA, Bowyer KW (2016) Gender classification from the same iris code used for recognition. IEEE Trans Inf Forensics Secur 6013(c):1–11
13. Cheng J, Li Y, Wang J, Yu L, Wang S (2019) Exploiting effective facial patches for robust gender recognition. Tsinghua Sci Technol 24(3):333–345
14. Federmann C, Lewis WD (2016) Microsoft speech language translation (MSLT) corpus: the IWSLT 2016 release for English, French and German
15. Alam SMS, Khan S (2014) Response of different window methods in speech recognition by using dynamic programming. In: 1st international conference on electrical engineering and information & communication technology ICEEICT 2014, no 2
16. Zaw TH, War N (2018) The combination of spectral entropy, zero crossing rate, short time energy and linear prediction error for voice activity detection. In: 20th international conference of computer and information technology ICCIT 2017, vol 2018, Jan 2018, pp 1–5
17. Ho TK (1995) Random decision forests, vol 47, pp 4–8
18. Breiman L (1994) Bagging predictors. Report no. 421. Department of Statistics, University of California, Berkeley
19. Platt JC (1998) Sequential minimal optimization: a fast algorithm for training support vector machines, pp 1–21
20. Rumelhart DE et al (1985) Learning internal representations by error propagation. Report no. V. Institute for Cognitive Science, University of California, San Diego, La Jolla, California
21. Sim KC, Lee K (2010) Adaptive score fusion using weighted logistic linear regression for spoken language recognition. In: Sim KC, Lee K-A (eds) Agency for science, technology and research (A STAR). 2010 IEEE international conference on acoustics, speech and signal processing, Singapore, pp 5018–5021

Chapter 24
Scrutinize the Idea of Hadoop-Based Data Lake for Big Data Storage

Arvind Panwar and Vishal Bhatnagar

1 Introduction

Data is the driving force for the economy of any country. The biggest task for any organization is to provide an infrastructure for data management. A few years ago in the late 1990s, the concept of a data warehouse is presented by Bill Inmon to store data and use that data for the decision-making process. A data warehouse is relational database storage that designs on the concept for querying the data and analyzes data [1]. According to Bill Inmon, a data warehouse is a collection of subjected oriented, non-volatile, time-variant, and integrated data from a different type of data source within or outside of organizations. A data warehouse is a highly successful concept and makes a very significant role to grow the business in this competitive world until the first decade of the twentieth century [2]. As the world is changing with the lightning speed, data warehouse faces some challenges. The prime challenge for the data warehouse is data itself because data warehouse stores data in relation form but enterprises need a repository to store relational as well as non-relation data. Another challenge for data warehouse is the core concept of data warehouse designing state that transforms the data before loading, its meaning enterprises must set the goal or business dimensions how to use data after storage, and if you change your requirements or business need you need to design or transform data again which is very costly and time-consuming process. This concept is known as a schema-on-write concept in data warehouse but data engineers have now new weapons like AI, machine learning, and deep learning to deal with data so data engineers need data in its raw format not in transformed. Data engineers need schema-on-read concept

A. Panwar (✉)
Guru Gobind Singh Indraprastha University, GGSIP University, Delhi, India
e-mail: arvind.nice3@gmail.com

Dr. Akhilesh Das Gupta Institute of Technology and Management, New Delhi, India

V. Bhatnagar
Ambedkar Institute of Advanced Communication Technologies and Research, Delhi, India

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0_24

data storage in which data store in its raw format means in the same format as the data generated. The next but not the last challenge for a data warehouse is big data.

Big data is the biggest challenge in the growth of data warehouse, and it is the main reason enterprises need to shift from data warehouse [3, 4]. Big data is not technology, it is a problem or it is the biggest headache for the data engineers as well as the whole world, but along with the problem, it is the sea of opportunities that need to explore. Big data is a data with velocity, data with huge volume, data with variety, data with veracity, and data with value. The traditional data warehouse or any other repository is not sufficient to store such kind of big data. Big data is the collection of structured data (e.g., table, excel sheet), unstructured data (e.g., PDF file, word file, customer review, social media comments), and semi-structured data (e.g., XML file). Data warehouse cannot handle this kind of data and its failure to fulfill the business need of current or future market demand, as a result, researchers try to find an alternative for data warehouse which can handle big data.

Hadoop and the Google file system are the only solution to store big data. Google file system and Hadoop offer the same kind of technology and infrastructure to handle big data but enterprises use Hadoop because Hadoop is an open-source product under Apache open-source community [5, 6] and Google file system a trademark product of Google. Hadoop is designed to store big data from different distributed data cluster, and it can process data from the distributed data center across the globe. Hadoop provides the infrastructure for big data storage, data processing, data accessing, securing data, data governance, and visualizing data insight [7].

After the first decade of twentieth century, organizations start moving toward the Hadoop system for data storage, but in the starting organization used Hadoop as a data warehouse. Many enterprises feel great success in business and good growth as well, but the limitations are the same with Hadoop-based data warehouse which is faced by traditional data warehouse systems. There are many confusion and some question about to move from a data warehouse. This confusion and question are regarding the performance of the new system, security offered by the new system, scalability of the new system, and many more, but all enterprises agree on one thing that they can continue with the traditional system because data engineers want data in raw format.

In the mid of the second decade of the twentieth century, a new term coined known as data lake by researchers. It is a technologically advanced version of the data warehouse. A data lake is also a repository like a data warehouse, but the core concept of both is different. Data warehouse works on the idea of schema-on-write, which means that before landing the data in the data warehouse enterprises must transform the data according to business needs. To store big data in data warehouse, an ETL (extract data from various data resources, transform data according to business requirement, and load the data on target storage media) process must execute [8–10]. With this kind of data storage architecture, the organization has to make a blueprint for the data model and make a data analysis strategy before the data loading phase. Or in other words, every enterprise, who wish to use data warehouse must know in the initial phase, how they are going to use data for decision-making process if they wish to change the analytical plan enterprise need to follow all the process again.

They need to transform data again, according to new business need and then load data again. This was the prime's reason to shift from data warehouse to data lake because data lake works on the schema-on-read concept, in which data is not transformed before loading, whereas data is stored in raw format [11, 12]. The data lake is only a data landing area, which stores the data without changes in the property of raw data. Whenever a data engineer has any business problem to solve, it is getting data from the data lake and transforms according to business needs and stores in a different area to get insights from data. To use data lake enterprises, it is not needed to make a blueprint for data model or any analytical plan before storing the data. The concept of data lake came from the idea of water lake where water is gathered from many resources like water from home, water from rain, water from the gutter, and many others. The same concept applies, here data is coming from the ever-growing data resources within the organization and outside the organization and stores it in raw format until some process not demands data from the lake [13–15]. Lake stores all kinds of data like structured, unstructured, and semi-structured at a single place.

After getting the knowledge about data lake, some organization thinks it can replace data warehouse because it supports big data and can remove data silos but on the other side there are some confusion and question about data lake. This chapter's author scrutinizes the data lake and tries to overcome confusion about data lake. After reading this chapter, reader can understand the data lake concept. Organization of the chapter is as follows: Sect. 2 describes the research methodology used in this work, Sect. 3 describes research classification, Sect. 4 gives a brief overview of big data and its storage, Sect. 5 explains data lake, and Sect. 6 provides data lake architecture concept.

2 Research Methodology

Our research is based upon the studies of different journals. Authors of this chapter select articles from journals, international conferences, edited books, and white chapters of the organization from different sectors. The author excludes master or doctorate thesis and unpublished work. Our research methodology is distributed into four different segments as given and as shown in Fig. 1:

- Verification of articles
- Analyze research
- Research definition
- Article searching.

2.1 Research Description or Definition

- Research area: The research area is data lake as shown in Fig. 2.

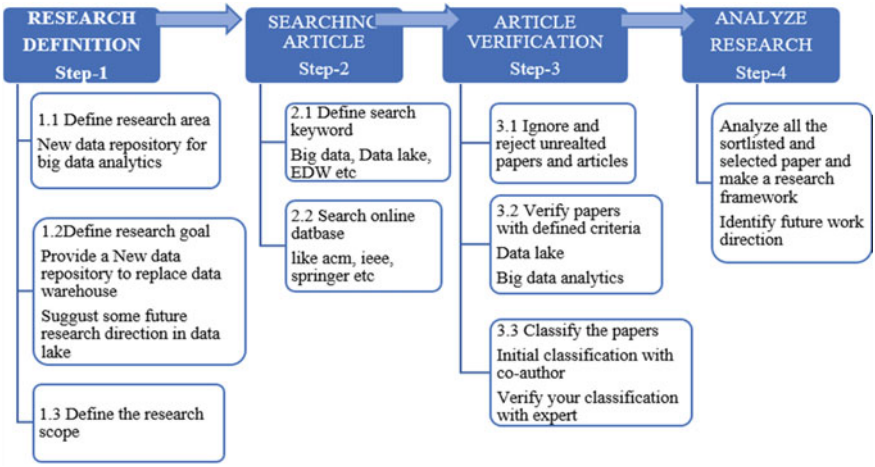


Fig. 1 Research methodology

Fig. 2 Research area in the data lake

- Research goal defined: The research objective is to offer a detailed survey on data lake and provide a reference architecture for big data storage which is an improvement version of an enterprise data warehouse for big data analytics.
- Research scope definition: The research possibility is the research work articles on data lake for big data storage and analytics.

2.2 Searching Articles

- Search terms and keywords selection: First, the author decides some search terms, and final search keywords are an enterprise data warehouse, big data storage, big data analytics, and data lake. Then, typical articles were originating as a reference by search keywords related to research space as shown in Fig. 2. Then, the search keyword was changed which depends upon information gathered from articles.
- Search an online database for articles: Author searches articles in the following online journals database:

- IGI Global
- IEEE Transactions and explorer
- ACM digital library
- Science Direct
- Hindawi journals
- MIT Press journals
- Wiley library
- Springer-Link Journals.

2.3 Article Verification

- Articles originate in Step 2 was cautiously and constantly verified by author and co-author. Only related articles with the research area are selected for classification. The verification process is as follows:
- Reject dissimilar papers: The research chapter that was totally unrelated to our research zone is rejected by author and co-author.
- Verified articles with defined criteria: After elimination or rejection of chapter in the last step, leftover articles are verified autonomously. The articles match with defined criteria that are selected, and others were discussed with co-author for the conclusion.
- Classify the articles: After verifying all articles in the last step, now the next step is to classify all articles. Author and co-author classify all the articles and prepare a result. After finalizing, the classification result is discussed with an expert in the research area.

2.4 Analyze Research

The author analyzed each and every selected chapter from diverse perceptions and recognized some forthcoming research direction and gives a detailed view of the data lake with the help of architecture.

3 Research Classification

The research nature of big data analytics and big data is difficult to restrain from particular disciplines; the related things are distributed through several journals. Accordingly, some online digital libraries were searched to deliver a comprehensive catalog of the theoretical literature on big data and big data storage.

- Taylor & Francis Online
- IGI Global
- IEEE Transactions and explorer
- ACM digital library
- Science Direct
- Hindawi journals
- MIT Press journals
- Wiley library
- Springer-Link Journals.

The search was depending upon the search keyword, "big data storage and big data" and "data warehouse and data lake," which initially created nearly 700 research papers. Each research paper was thoroughly reviewed to remove this paper which is not related to big data storage and data lake or data warehouse. The collection standards are as below:

- Articles published in big data, data storage, data warehouse, and data lake associated journals were selected because they offer the most suitable vents for data lake research.
- The research paper which clearly defines, how to store big data, data lake, and data warehouse method(s) could be assisted and applied in big data storage was selected.
- Conference paper, doctoral and master's thesis, textbooks, and unpublished working paper and chapters were omitted, as practitioners and scholars most frequently use journals to gain information and publicize fresh discoveries. Hence, journals denote maximum research.

Each research paper was cautiously studied and independently classified according to the diagram, as shown in Fig. 3. Even though this search was not complete, it assists as a wide range for an understanding of data lake.

4 What Are Big Data and Big Data Storage?

Big data is the term that is used to define the pool of data sets having a complex structure, voluminous in nature, and that are compiled from different sources such as online transactions, mobile systems, web applications, and data records [16, 17]. The data generated from different sources is stored in databases that significantly increase the volume of data. Big data analytics is the process to manage such a huge amount of data and use it for pattern discovery and the knowledge discovery process [18, 19].

Big data plays a significant role in the non-commercial use of pattern recognition in the research area for industries. Pattern discovery and relationship finding techniques are necessary for customer segmentation and market basket analysis to discover the relationship between business and customers for maintaining demand

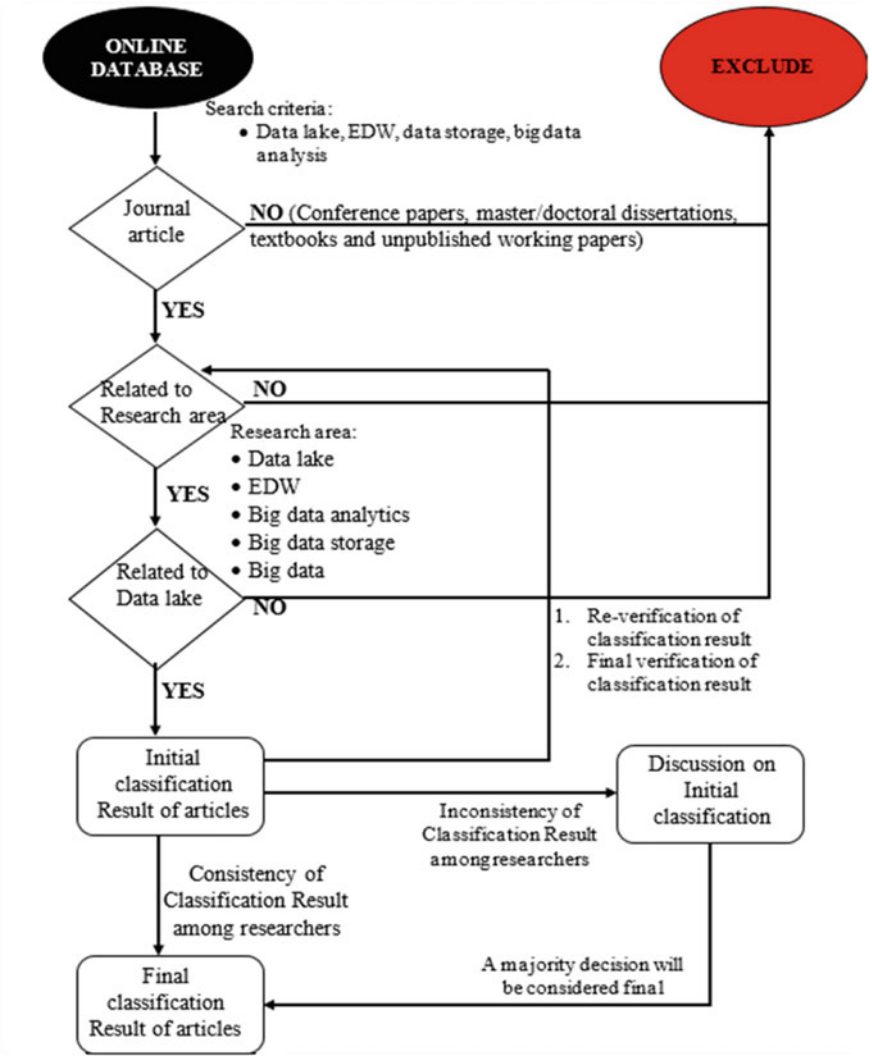


Fig. 3 Research classification and article selection criteria

and supply chain. The big data is potentially required in health care, public sector, retail, manufacturing, personal location data, and smart routing technologies.

There are three main components of big data that are necessary for revolutionary data analysis that is variety, velocity, and volume [20–22]. Variety deals with the different types of sources from where data is taken such as structured, semi-structured, and unstructured as shown in Fig. 4. Velocity deals with time-limited and time-variant processes that can be streaming data, real-time data, near real-time data, and batch processes [23, 24]. The volume deals with the size of data that can be

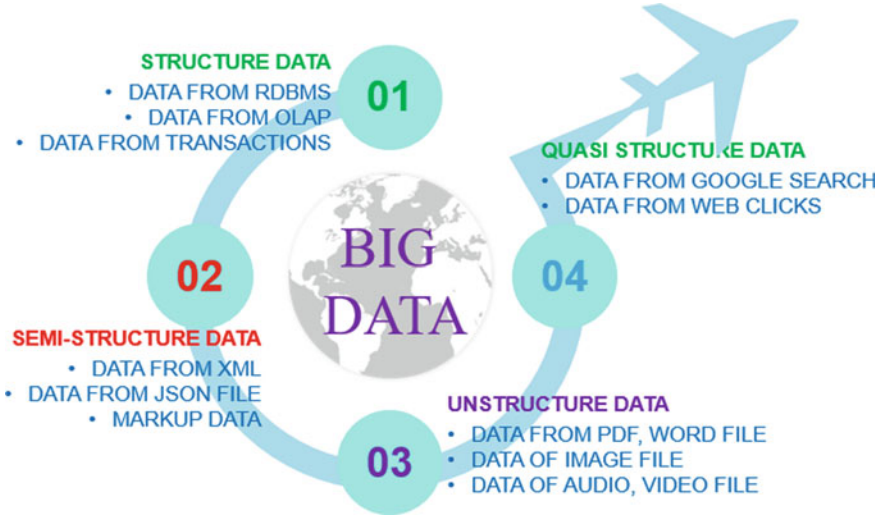


Fig. 4 Variety of data

in terabytes, petabytes, exabytes, and zettabytes. These three main components are known as three Vs of big data. In the late first decade of the twentieth century, some additional Vs add in the definition of big data. Figure 5 shows eight Vs of big data.

Big data has the potential to deal with new data as well as preexisting data for transforming business aspects by using MapReduce. MapReduce is a programming framework that uses the divide and conquer method to break down the big data complexity into smaller ones and process them together. In MapReduce, the map breaks the problem in subproblems and processes these into subsets which is stored

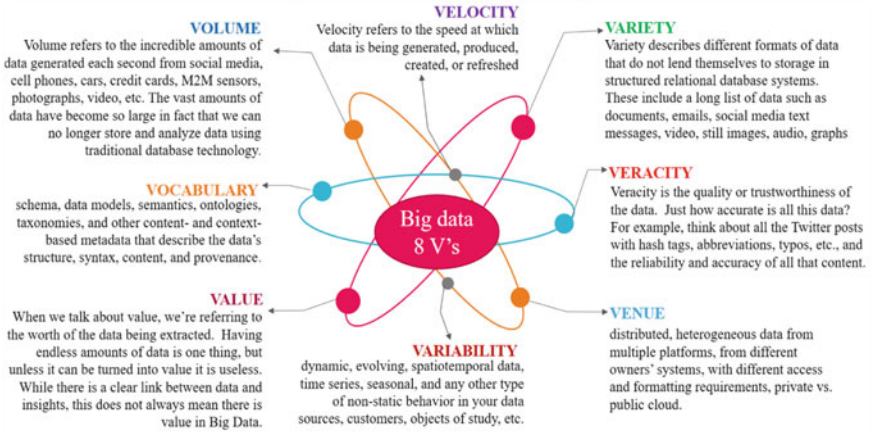


Fig. 5 Eight Vs of big data

in a local file system by job tracker, the reduce step picks up the subsets from the file system and applies many reduce task for parallelization and aggregation.

Big data stores data differently from traditional warehouses, the data that is to be stored in big data is first needs to properly transform and cleanse into well-defined structures [25–27]. Big data also deals with data that is to be stored as well as the data that is not to be stored in warehouses. Big data stores data in the form of sensor data and log file storage.

By 2020, it is expected that the digital universe will reach by 44 trillion gigabytes. Such data cannot be processed by traditional tools. To harness such amount of data is a challenging task. The two main challenges are how to store such a voluminous amount of data and how to process the data quickly. Therefore, data lake comes to the rescue.

Data lake is a central repository that stores the unstructured or semi-structured data in the granular or raw format [28–30]. The term is coined by James Dixon, and data lake means the ad hoc nature of the data in comparison with clean and processed data in a data warehouse unit. Data lake can be configured on the cluster of commodity hardware, and these clusters can either exist on the premises or in the cloud [31, 32]. Data lake works on the principle of "schema-on-read" that is data needs not to be in a pre-defined schema before storing it in a data lake [33].

For the analytics, expert's availability of such a voluminous amount of data in the non-traditional format rises the opportunity to access, prepare, and analyze the data faster and with better accuracy [34, 35]. Properties such as storing the large chunks of data, highly scalable environment, and accepting data in its original format make the data lake next-generation data management solution to real-time analytics.

Once the data is collected together, the data can be processed through various big data techniques. In a data lake, security can be assigned in such a way that users of the data lake have access to certain information that does not have access to the original content source. The data in the data lake can be normalized and enriched which is extraction of metadata, indexing, format conversion, extraction of an entity, and cross-linking. Data lake data can be prepared as required, therefore it provides flexible access to the analysts across the different globe [36].

4.1 Big Data Life Cycle

Now we have a brief view of big data. The author is describing how to the treat with big data. What is a big data life cycle? How data is treated when dealing with a big data system? What are the steps to analyze big data? Though tactics to execution vary, there are some harmonies in the policies and software we use mostly. The steps given below and shown in Fig. 6 may not be true for all but broadly used. Figure 6 shows the complete life cycle of big data.

- Clustered computing: As the author defined big data, personal computers are inadequate for dealing with the data at the utmost phases. Big data needs high computational power and high storage, to address such kind of issues, computer clusters are the best fit.
- Ingestion data: Data ingestion is the procedure of capturing fresh data and accumulates it into the system [37]. The complication of the process depends deeply on the quality and format of the data sources. There is a dedicated ingestion tool in a big data system. Hadoop is a big data system and supports many tools and technologies for data ingestion, for example, Apache Sqoop, Apache Flume, Apache Kafka, and Apache Chukwa [38]. In the upcoming section, the author gives a table for detail tools that are supported by Hadoop and helpful to manage the big data life cycle.
- Persevering the data in storage: The consumption procedures classically collect data from different resources and send them to the system storage, so that it can consistently persevere to medium. Although this appears like it would be an easy process, the requirements for availability, the volume of incoming data, and the system become more complex with distributed computing. Tools like HDFS are used to preserve or store big data in a big data system.
- Analyzing data and computing data: As soon as the data is accessible, the system can start treating the data to outward real information. The most varied part is the computation layer, of the system as the necessities and best tactic can differ significantly, dependent on what kind of visions desired. Data is regularly processed, either iteratively by a solo tool or by using several tools to outward different types of information. Tools like MapReduce, Apache Storm, and Apache Spark are used to analyze data in a big data system [36–38].
- Results visualization: Due to the different types of data is being treated in big data systems to get different types of insights, identifying trends or variations in data over time is often more vital than the values themselves. Visualizing data is one of the utmost useful traditions to show trends and make the wisdom of various data points [39]. Tools like Tableau, Elastic Stack, Excel, and Power View are used to visualize the result in the big data system.

4.2 Tools and Technology to Process and Analysis Big Data

Apache Hadoop software is used to manage the life cycle of big data. Hadoop is a software which is used for distributed processing and distributed storage of huge amount of data sets on computer clusters. Apache Hadoop offers many services like storage of data, data processing, data access, data governance, data security, data visualization, and operations. Hadoop has a different tool for every kind of operation on big data. Table 1 shows all kinds of tools which are used to do different operation on big data.

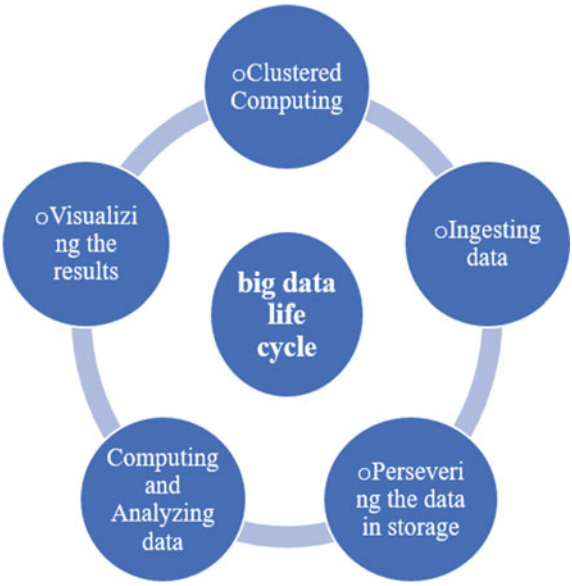


Fig. 6 Big data life cycle

5 Data Lake

We are witnessing a time of extraordinary revolution in the area of big data storage as data progresses toward great diversity which means more data types, more data schemas, more data sources, and more latencies. As a user of data, organizations also diversify the way how they use the data for business requirements, to get business insights and values via advanced tools for analytics.

To scale up growing old-style data, to manage new big data, and to fully leverage both, organizations need to enhance or to modernize the collection of tools and techniques, must improve skill sets, need best practices within the organization, and needs a new platform like a data lake. The data lake is just a landing area of data in its naïve. The basic concept of the data lake is shown in Fig. 7. It is a good practice to make a data lake, but the organization takes a risk to make data lake because many times they started with a data lake and it turns into a data swamp. Data lake and data swamp both are the data repository but data lake is well organized, saturated and provides data governance, whereas data swamp is disorganized, oversaturated. Data lake offer gives insights and business value but data swamp cannot offer this. It is very difficult to distinguish between a data lake and data swamp because both store the data in its raw format and both are data landing areas. During the creation of data lake enterprises need to focus on some point which is given below, otherwise you ended with a data swamp.

- **Data cleaning policy**. Not a single organization wants to make a data swamp but if they did not follow stick strategies for data cleaning regularly data lake turns in data swamp because incorrect, duplicate, or uncleaned data gives a poor decision.

Table 1 Tools to manage big data life cycle

Tool category	Tool name	Descriptions	Tool features
Security	Apache Knox	It offers security for entrance in Hadoop clusters. The Knox Gateway delivers edge-level security so that the organization can confidently spread Hadoop access to new users, by maintaining agreement with enterprise security strategies	• Deliver security for services like HTTP and REST • Offer REST API services for Falcon, Ranger, and Ambari • It simplifies configuration management with the incorporation of Ambari
	Apache Ranger	Ranger delivers a wide-ranging tactic to secure a cluster. It offers a centralized policy to describe, manage, and administer security	• Integrate API with HDFS • Sustenance for different permissions within cluster modules • Whole system audit • Global tag-based policies
	HDFS encryption	HDFS implements transparent, throughout encryption. Once it is constructed, data written to or read from HDFS file system directories is encrypted and decrypted transparently, and it is not required to change application code for users. This encryption process is end to end	• Inside an encryption block, it offers file system-based client decryption and encryption • HDFS user works with KMS using API and recovers encryption keys • Ranger KMS is designed by Apache for key retrieval using a proxy

(continued)

Table 1 (continued)

Tool category	Tool name	Descriptions	Tool features
Operations (provisioning, managing, and monitoring)	Apache Ambari	The objective of the Ambari venture is making Hadoop management simple by evolving new software for monitoring systems, for node management, and provisioning clusters. It offers easy-to-use and intuitive Hadoop management with the help of a web-based user interface with REST API	• Ambari used to check the status and health of clusters with the help of the dashboard • It offers system alerting and sends message or email for your attention, system at risk in a situation like low disk space, node down, etc.
	Apache Zookeeper	Zookeeper is a centralized provision or service for synchronization, maintaining configuration information, deliver distributed and offer group services, and naming services	• Fast • Simple • Ordered
Operations (scheduling)	Apache Oozie	It is a Java-based web application that offers workflow scheduler for Hadoop jobs. Oozie makes e logical unit of work with the combination of several serial jobs. It is combined and works with the Hadoop stack. It supports jobs for Hive, Pig, MapReduce, and Sqoop	• Oozie offers a workflow scheduler to maintain and manage Hadoop works • Oozie workflow works are directed acyclic graphs (DAGs) of actions • Oozie is a scalable, reliable, and extensible system

(continued)

Table 1 (continued)

Tool category	Tool name	Descriptions	Tool features
Data access	MapReduce	MapReduce is a framework for simply building applications that process huge size of data in-parallel on big clusters of commodity hardware in a fault-tolerant, reliable manner. This framework contains a master job tracker and one slave task tracker per cluster node	• Flexible access • Scale-out architecture • Fault tolerance • High availability • POSIX-based file permissions load balancing • High security
	Apache Hive	Hive software simplifies querying and managing huge data sets present in distributed storage. Hive offers a tool to query the data using a SQL-like language called HiveQL [1]	• Indexing to provide acceleration • Different storage types • Metadata storage in an RDBMS • SQL-like queries (HiveQL)
	Apache Spark	It is a framework for data analytics useful for cluster computing. Spark offers data analytics for in-memory cluster computing [2]	• Established scalability to 100 nodes in the research laboratory • Capability to cache data sets in memory for interactive data analysis • Offer command–line interface
Data management	Avro	Avro delivers a suitable approach to denote composite data structures within a Hadoop MapReduce. Avro data is an input as well as output for a MapReduce job	• Near real-time indexing • Flexible and adaptable with XML • Extensible plugin architecture

(continued)

Table 1 (continued)

Tool category	Tool name	Descriptions	Tool features
	HDFS	It is initially designed to accumulate very big data sets dependably and to transfer those data sets at very high bandwidth to applications and users. In a huge cluster, hundreds of hundred servers directly attached storage	• Rack awareness tools diagnose file system health • Rollback • Standby name node • Highly operable
	NoSQL	It is new age databases generally addressing few points: being distributed, horizontally scalable, open-source, and non-relational. The initial intention has been up-to-date web databases	• Simple data model • Simple programming model with ACID transactions • Application security • Geo-distributed data • High availability • Scalable throughput
Data governance (data life cycle and governance)	Apache Atlas	Atlas delivers scalable governance for organizations Hadoop that is determined by metadata. Atlas, at its essence, is planned to simply model fresh business processes	• Data lineage • Agile data modeling • REST API • Metadata exchange
	Apache Falcon	Falcon permits an organization to process a solo huge data set stored in HDFS in several ways—for interactive, batch, and streaming applications	• Centralized definition • Ensure disaster readiness • Out of the box policies • End to end monitoring • Visualize data pipeline • Track data pipeline audit • Tag data with business metadata
Data governance (data workflow)	Sqoop	It is a tool that is designed to transfer data between Hadoop and databases. You can use Sqoop to ingress data from an RDBMS such as MS SQL server into the HDFS and vice versa	• Connection to database server • Import data to HBase • Controlling parallelism • Import data to hive • Controlling the import process

(continued)

Table 1 (continued)

Tool category	Tool name	Descriptions	Tool features
	WebHDFS	Hadoop offers a high-performance built-in protocol for reading HDFS. It is a great tool for Hadoop applications that are executing within cluster, but sometimes users need data from the outside cluster. To resolve this, researchers find an additional protocol to read, ed WebHDFS	• HTTP REST API • Wire compatibility • Secure authentication • Data locality
Machine learning	Mahout	It is a library of machine learning algorithms, implemented on topmost of Hadoop and using the MapReduce prototype	• Collaborative filtering • Clustering • Classification and many other techniques for machine learning
Operating system	YARN	YARN was the initial component of the Hadoop project, and it is the architectural center of Hadoop which permits several data processing engines, for example, real-time streaming, interactive SQL, batch processing, and data science to handle data stored in a solo platform, revealing an utterly novel tactic to analytics [40]	• Multi-tenancy: YARN allows multiple access engines • Cluster utilization: YARN's dynamic allocation of cluster • Scalability: data center processing power continues • Compatibility

Data source Hortonworks

Fig. 7 Data lake concept

- **Automated process**. The data lake has a huge amount of data, and if the automation process of data management is not applying on data lake, it becomes a data swamp because it is impossible to handle such huge data manually.
- **No data governance**. Data governance sets the rules for data treatment, for data users, and for data flow. A good data lake has a good data governance system if it lacks data lake became a data swamp.
- **Lack of metadata**. Metadata has information that defines data in the system. It helps the user to search for data in a data lake. A bad metadata design leads to a data swamp.

5.1 Data Lake Versus Data Swamp Versus Data Warehouse

There are primarily two repositories to store big data, one is data warehouse and another is data lake. In the last section, the author describes a bad version of the data

Table 2 Data warehouse versus data lake versus data swamp [39]

Attributes	Data lake	Data warehouse	Data swamp
Schema	Schema-on-write	Schema-on-read	Schema-on-write
Scale	Scales to large volumes at moderate cost [39]	Scales to huge volumes at low cost	Scales to large volumes at moderate cost
Saturation level	Saturated	Undersaturated	Oversaturated
Access methods	Through standard SQL and BI tools	Through a SQL-like system, programs created by the developer and another tool [39]	Through standard SQL and BI tools
Workload	Support batch processing, as well as thousands of concurrent users performing interactive analytics [39]	Support batch processing, plus an improved capability over EDWs to support interactive queries by users	Support batch processing, as well as thousands of concurrent users performing interactive analytics
Data	Cleansed	Raw	Not cleansed
Metadata	Well-designed centralized	Fair designed	Multiple metadata storage
Complexity	Complex integrations	Complex processing	Complex integrations
Cost/efficiency	Efficiently uses of CPU/IO	Efficiently uses storage and processing at very low cost	Efficiently uses of CPU/IO
Data governance	Offer	Partial offer	Lacks of data governance
Configuration	Fixed configuration	Reconfigurable	Fixed configuration
Automation process	Highly automated	Partially automated	No automation process
Users	Businesspersons	Data scientists	Only storage

lake known as a data swamp. In this section, we compare all three data repositories. Table 2 shows some common differences.

5.2 The Need for Data Consumer and Producer

A data lake is a growing approach to placing and extracting all kinds of data, from different kinds of data sources, for data analytics in a single storage. All kinds of data mean internal from the organization and external from the market, as well as big and small data, as well as structured, semi-structured, and unstructured data to it.

Table 3 Need for data consumer and data producer

The need for the data consumer Need flexibility		The need for the data producer Want control
• Need rapid and real-time data: There are several cases when reports need to be delivered in real-time so need real-time data	Mismatch in their need	• Focus on privacy and access control: It is important for data producers who are accessing data. Producer control data access and information privacy
• Want to access historical data: We need to access historical data to find insights and compare with time		• Manage costs for the system: Data producer is always worried about the cost of implementing a data lake
• Focus on a variety of analytics: Data engineers and scientists perform machine learning or data mining algorithms so the consumer needs a variety of analytics		• Need minimal IT overheads: The data producer is worried about IT overhead. Overhead due to infrastructure and resources to manage complex and large data lakes
• Always need validated data set: If data sets are not validated, then generated insights lead to the wrong direction so it is good to provide a validated data set		• Compliance: While data is accessing and storing in a data lake, producer of data guarantees that business strategies and instructions are followed properly
• Quickly find relevant data: They should be capable of finding and discovering data sets in data lake according to their research		• Focus on data and usage visibility: It is very important to know for data producer that who and when used the data

Data is used in the alternative of a data warehouse or it is an advanced version of the data warehouse to set a termination to data silos. Before presenting an architecture of data lake, it is necessary to know the main actor of data in an organization and their need. There are two types of actor, one is data consumers who want flexibility in data to solve business problems with new insights and the other is data producers who want proper control on data. Table 3 presents a brief overview of the need for data consumer and data producer and shows a mismatch between the need of two.

5.3 Application and Use Cases of the Data Lake

Implementing a Hadoop project as a part of data storage architecture in any industry is an important decision for any organization. The momentum of Hadoop is unstoppable for the current scenario; its growth is a drive from solo data instance application to a well-developed data lake for multiple distributed data instances. Now data lake is

Table 4 Application area of the data lake

Application or industry	Application area or use case	Data type used
Financial services	Trading risk, fraud detection	Server logs, text
	Account risk screens	Server logs, machine-generated
	Insurance underwriting	Sensor, text, geographic
Telecom industry	Call detail recording	Geographic, machine-generated
	Infrastructure investment	machine-generated, server logs
	Bandwidth allocation, etc.	Sensor, text, social
Retail	360 views of customer	Text, clickstream
	Localization, promotions	Text, clickstream, geographic
	Web optimization	clickstream, and geographic
Healthcare	Genomic data in medical trial	Sensor, structure
	Real-time patient monitor	Sensor, machine, structure, and text
Pharmaceuticals	Improve prescription observance	Text, social, geographic, unstructured
Oil and gas	Exploration and production data	Sensor, geographic, and unstructured
	Monitor safety in real-time	Sensor, unstructured
Government	Sentiment analysis for govt programs	Social, text, and unstructured
Manufacturing	Supply chain	Sensor, machine, and geographic
	Quality assurance	Sensor, structured, and unstructured

a part of every industry. Some example of industry and their use case is given in Table 4.

6 The Architecture of Data Lake

Building a data lake is a very tough job in comparison with building a system that supports single use of data after processing such as data warehouse. All the data lake application is Hadoop-based. Data lake must support both schema-on-read and schema-on-write approaches because it is not possible to model and anticipate all of an organization's data in advance. That is why a new approach and a new architecture are required, an architecture which does not have the limitation of the data warehouse or a limitation of Hadoop installation. New architecture supports an extensive range of tools and techniques for data processing and analysis.

A data lake must support four different needs: data acquisition, data management, data access, and data insight [41, 42]. Rather than adopting a monolithic style, we

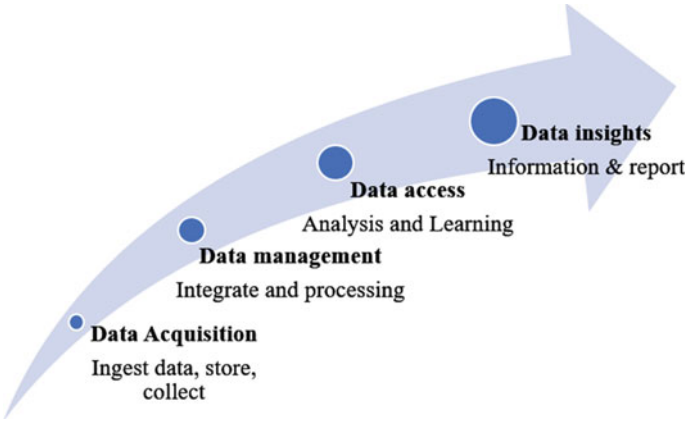


Fig. 8 Data lake architectural component

must take each of these four different needs as an autonomous design built on a shared platform as shown in Fig. 8, which shows architectural component of data lake. Data acquisition must be independent of data management because you want to make data collection, data ingestion, and data storing as easy as possible so that data is made available quickly for the next stage. Separating both from each other allows the user to collect data at one time and can process data at another, as well as it permits the user to process and to use the data [43, 44]. This approach suggests that the user should make the data acquisition system with an undisputable data store for raw data. By adopting this approach, user retains all the raw data and stores separately all the changed or derived data which support easy reprocessing of any derived or changed data.

In a similar way, data access should be separate from both data acquisition and data management. Humans are not the lone users of data in data lake, its also support independent application for data access and control processing of data for automation. Some applications and users want raw data access and should not be prevented. Some of them want structured data, while others want to access their own data. This kind of separation of data access from data acquisition and data management can only be done with the help of decoupling the part of the data lake that supports data access from the part of a data lake which supports data collection and data management. On the other hand, the data warehouse is monolithically designed around one workload. It combines data access and data management into one global data model.

A data lake design with separation of functions means repeatable use of data when this repeatability is needed, instead of forcing on everyone upfront [45, 46]. The separation of functions adds flexibility in the system. Always design a data lake to keep a secret in mind and the secret is knowing when to relax data and when to manage control over data. As discussed above, some data needs to be stored in a form only while others should be structured or standardized. To apply this kind of

approach, it requires two kinds of architecture, one is data architecture, and another is technology service architecture, both together define a data lake.

6.1 Technology Service Architecture

Data lake needs a technology architecture that defines the required services or capabilities. As discussed above, every specific job determines which service you require and which are not. Table 5 gives some services and their descriptions.

6.2 Data Architecture of Data Lake

Data design is the heart of the data lake. Data architecture defines the arrangement of data for easy retrieval, easy capture, easy reuse, easy store, and easy use to avoid and deadlock processing and user efforts. Data lake needs an architecture for data processing that is not limiting to a limited warehouse schema. Data lake wants to deal with a change in data requirement more easily on a big scale [45, 47, 48]. Data lake should not be imposed by some models and requirements in starting that prevent raw data collection. It also should not be limited to any kind of data format or structure. It must support read-write schema rather than read-only and must provide shared data location and private data location. Data architecture is divided into four zones as shown in Fig. 9.

- **Raw data storage zone**. This data zone is the primary landing area of data. Whenever a data request is made by the user, it should be done very quickly so that data is available on time. At this stage, there is no need to do data modeling or processing. As discussed above, data lake is schema less and it means nobody defines or models data in starting before data is available. To record data in the original structure and format, the data collection problem is decoupled from the data management problem. Sometimes data scientists or data analyst wants to know that it is useful or not by the data access process. If it is useful or repeatable, then data cleaning and data enhancement are worth otherwise resources and time are wasted. This zone of data collection must be an immutable, data store as it is with no attempt of data cleaning or enhancement.
- **Formatted or enhanced data zone**. Some data is very much valuable than other data. Some data is accessed more frequently than others. When this kind of data is needed, the data can be cleaned or enhanced, and it should be made available in the different zone of the data lake. The aim behind a separate zone for this kind of data is to make data more shareable and accessible for users.
- **Usage-specific data zone**. In some cases, data is mutual crosswise large part of the enterprise, and this kind of data is placed into this zone of data. For example, telecommunication data is core data that should be a common format, a single

Table 5 Basic technology service supported by a data lake

Service name	Service descriptions
Data persistence	Here, we use keyword persistence because data lake is more than simple data storage. Applications, tools, and developers require access to high-level services in which data can persist for different duration, from short-lived output caching to long-term raw data storage
Data movement	Data lake should support different types of data movement because data can be pulled from a data lake and can be pushed to a data lake. Data can be collected from different data sources, services, files or databases, and from streams. The movement of data is bidirectional
Data access	As discussed in the last section that data access is distinct from the data movement. Data lake allows outside tools and applications to access the stored data in any format or type of persistence. Data lake supports APIs, SQL queries, and search all are the possible tactics for accessing stored data
Processing engine	Data lake should support different types of processing engines because different types of workloads need different types of processing to solve problems. Some need batch processing, some are pipelining, or some need stream
Data flow tools	As discussed, data lake stores data in raw format but is not the only way to use data. Different application needs data in different formats that is why data lake must support data flow tools which used to reformat, filter, clean, integrate, and compute data
Workflow management and scheduling	Workflow and scheduling tools provide the facility to the developer to define and solve scheduling priorities of jobs and manage dependencies in jobs, between jobs, and on data sets
Metadata	Every recorded data has metadata. Data lake also supports metadata services to record metadata with it, at the source-level metadata and every element-level metadata. It is very crucial to know about when data is stored, from which data source it comes, and in what format is stored
Data curation	Self-service analysis is not a reality; it works under a curated atmosphere. As discussed above, data lake has many data sets from a different data source like a data warehouse. To analyze these data sets, data lake supports data curation services to provide an environment

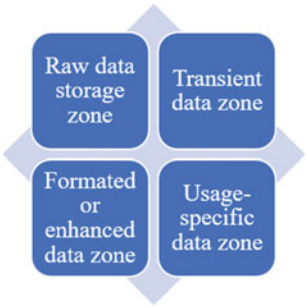


Fig. 9 Data architecture of data lake

with standard names. In general, it is a small percentage of total data available, but it uses. Data that is structured for a specific goal and also stores in this zone.

- **Transient data zone**. One common challenge which user or data analyst is facing is to find a location to play with data. Users need an area that let them work on data without disturbing others. Sometimes user's need is very complex such as prepare a data model for machine learning, sometimes it is very simple such as loading an external data file, but one thing is common that is a place to work with data. This problem is solved by providing a special zone for this kind of data known as transient data zone [49].

7 Conclusion

This chapter gives an overview of different types of data repository used in the past to store big data and why they are out of the market now. The authors show why data engineers want a new repository for storing data to fulfill market demand. Data engineers want a new repository to store big data as data warehouse works on the concept of schema-on-write state that transforms the data before storage but engineers want data in raw format and later on according to business need they can transform the data to get the different values from data. To overcome the challenges which were faced by data warehouse, research comes with a new concept known as data lake, a technologically advanced version of a data warehouse. Data lake works on the concept of schema-on-read. This chapter gives a detailed view of the big data lake and gives the difference between a data lake and data warehouse and data swamp. Author presents data lake architecture from a different perspective.

8 Future Direction

In the upcoming paper about data lake, author will discuss cloud-based data lake and link data lake with IoT. If any organization is moving data to the cloud, first

thought comes in mind of every stakeholder is a security of data. So the security of cloud-based data lake is a good research area for the future. As the hype of bitcoin, every organization is talking about the blockchain for different application areas other than finance. Many organizations explore the potential of blockchain in supply chain management and some in health care. The reader can make a cloud-based data lake for healthcare data and implement security with the help of blockchain. Readers can try to store data in blockchain to make data temper proof. There are many security problems in cloud-based data storage readers can work on these problems and try to overcome with the help of blockchain.

References

1. Inmon W, Linstedt D, Levins M (2015) Data architecture: a primer for the data scientist
2. Klettke M, Awolin H, Storl U, Muller D, Scherzinger S (2017) Uncovering the evolution history of data lakes. In: Proceedings—2017 IEEE international conference on big data, big data 2017, vol 2018, Jan, pp 2462–2471
3. Costa C, Santos MY (2017) The SusCity big data warehousing approach for smart cities. In: ACM international conference proceeding series, vol Part F1294, pp 264–273
4. Auer S et al (2017) The BigDataEurope platform—supporting the variety dimension of big data. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 10360 LNCS, pp 41–59
5. Foran DJ et al (2017) Roadmap to a comprehensive clinical data warehouse for precision medicine applications in oncology. Cancer Inform 16
6. Jarke M, Quix C (2017) On warehouses, lakes, and spaces: the changing role of conceptual modeling for data integration. In: Conceptual modeling perspectives. Springer International Publishing, pp 231–245
7. Alili H, Belhajjame K, Grigori D, Drira R, Ben Ghezala HH (2017) On enriching user-centered data integration schemas in service lakes. Lect Notes Bus Inf Process 288:3–15
8. Wibowo M, Sulaiman S, Shamsuddin SM (2017) Machine learning in data lake for combining data silos. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 10387 LNCS, pp 294–306
9. Yamada T, Kato Y, Maekawa Y, Tomiyama T (2017) Interactive service for visualizing data association using a self-organizing structure of schemas. In: Proceedings—2017 IEEE 10th international conference on service-oriented computing and applications, SOCA 2017, vol 2017, Jan, pp 230–233
10. Martínez-Prieto MA, Bregon A, García-Miranda I, Álvarez-Esteban PC, Díaz F, Scarlatti D (2017) Integrating flight-related information into a (big) data lake. In: AIAA/IEEE digital avionics systems conference—proceedings, vol 2017, Sept
11. Madera C, Laurent A, Libourel T, Miralles A (2017) How can the data lake concept influence information system design for agriculture? In: EFITA CONGRESS
12. Stefanowski J, Krawiec K, Wrembel R (2017) Exploring complex and big data. Int J Appl Math Comput Sci 27(4):669–679
13. Jarke M (2017) Data spaces: combining goal-driven and data-driven approaches in community decision and negotiation support. Lect Notes Bus Inf Process 293:3–14
14. Mathis C (2017) Data Lakes. Datenbank-Spektrum 17(3):289–293
15. Spendla L, Kebisek M, Tanuska P, Hrcka L (2017) Concept of predictive maintenance of production systems in accordance with industry 4.0. In: SAMI 2017—IEEE 15th international symposium on applied machine intelligence and informatics, proceedings, pp 405–410
16. Golov N, Rönnbäck L (2017) Big data normalization for massively parallel processing databases. Comput Stand Interfaces 54:86–93

17. Mari F, Masini P (2017) Big data at work: the practitioners' point of view. IEEE Instrum Meas Mag 20(5):13–20
18. Ramakrishnan R et al (2017) Azure data lake store: a hyperscale distributed file service for big data analytics. In: Proceedings of the ACM SIGMOD international conference on management of data, vol Part F1277, pp 51–63
19. Rudnicki R, Donohue B, Cox AP, Jensen M (2018) Towards a methodology for lossless data exchange between NoSQL data structures. In: spiedigitallibrary.org, p 25
20. Hai R, Quix C, Zhou C (2018) Query rewriting for heterogeneous data lakes. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 11019 LNCS, pp 35–49
21. Shepherd A, Kesa C, Cooper J, Onema J, Kovacs P (2018) Opportunities and challenges associated with implementing data lakes for enterprise decision-making. Issues Inf Syst 19(1):48–57
22. Miller RJ (2018) Open data integration. PVLDB 11(12):2130–2139
23. Nogueira ID, Romdhane M, Darmont J (2018) Modeling data lake metadata with a data vault. In: ACM international conference proceeding series, pp 253–261
24. Beheshti A et al (2018) Iprocess: enabling IoT platforms in data-driven knowledge-intensive processes. Lect Notes Bus Inf Process 329:108–126
25. Cha BR, Park S, Kim JW, Pan SB, Shin JH (2018) International network performance and security testing based on distributed Abyss storage cluster and draft of data lake framework. Secur Commun Netw 2018
26. Cha B, Park S, Kim J (2018) Design and interface testing of connected data architecture of data lake. In: 9th international conference on information and communication technology convergence: ICT convergence powered by smart intelligence, ICTC 2018, pp 780–782
27. Vermeulen AF (2018) Data science technology stack. In: Practical data science. Apress, pp 1–13
28. Llave MR (2018) Data lakes in business intelligence: reporting from the trenches. Procedia Comput Sci 138:516–524
29. Maini E, Venkateswarlu B, Gupta A (2018) Data lake—an optimum solution for storage and analytics of big data in cardiovascular disease prediction system
30. Phyu KP, Shun WZ (2018) Data lake: a new ideology in big data era. In: ITM web of conferences 17, 03025 (2018) WCSN 2017, vol 03025, pp 1–11
31. Schuetz CG, Schausberger S, Schrefl M (2018) Building an active semantic data warehouse for precision dairy farming. J Organ Comput Electron Commer 28(2):122–141
32. Villegas-Ch W, Luján-Mora S, Buenaño-Fernandez D, Palacios-Pacheco X (2018) Big data, the next step in the evolution of educational data analysis. Adv Intell Syst Comput 721:138–147
33. Quinto B (2018) Big data warehousing. In: Next-generation big data. Apress, pp 375–406
34. Jain A, Bhatnagar V (2016) Concoction of ambient intelligence and big data for better patient ministration services. Int J Ambient Comput Intell (IJACI) 08(04):19–30. IGI Global. ISSN: 1941-6237 (Web of science)
35. Sharma A, Bhatnagar V, Bansal A (2016) SENSEX price fluctuation forecasting comparison between global indices and companies making it. J Glob Inf Manag (JGIM) 26(03):90–104. IGI Global, July–Sept. ISSN: 1062-7375 (Web of science: 0.517)
36. Sinha S, Bhatnagar V, Bansal A (2016) Multi-label Naïve Bayes classifier for identification of top destination and issues to accost by tourism sector. J Glob Inf Manag (JGIM) 26(03):37–53. IGI Global, July–Sept. ISSN: 1062-7375 (Web of science: 0.517)
37. Sinha S, Bhatnagar V, Bansal A (2017). A framework for effective data analytics in tourism sector: big data approach. Int J Grid High Perform Comput (IJGHPC) 09(03):92–104. IGI Global. ISSN: 1938-0259 (Web of science: 0.517)
38. Sangwan N, Bhatnagar V (2019) Comprehensive contemplation of probabilistic aspects in intelligent analytics. Int J Serv Sci Manag Eng Technol (IJSSMET) 11(01):116–141. IGI Global. ISSN: 1947-959X (Scopus)
39. Panwar A, Bhatnagar V (2019). Data lake architecture: a new repository for data engineer. Int J Org Collect Intell (IJOCI) 10(01):63–75. IGI Global. ISSN: 1947-9344 (ACM digital Library)

40. Quinto B (2018) Big data governance and management. In: Next-generation big data. Apress, pp 495–506
41. Dobson S, Golfarelli M, Graziani S, Rizzi S (2018) A reference architecture and model for sensor data warehousing. IEEE Sens J 18(18):7659–7670
42. Diamantini C, Lo Giudice P, Musarella L, Potena D, Storti E, Ursino D (2018) A new metadata model to uniformly handle heterogeneous data lake sources. Commun Comput Inf Sci 909:165–177
43. Li Y, Zhang AM, Zhang X, Wu Z (2018) A data lake architecture for monitoring and diagnosis system of power grid. In: ACM international conference proceeding series, pp 192–198
44. Mrozek D, Dabek T, Małysiak-Mrozek B (2019) Scalable extraction of big macromolecular data in azure data lake environment. Molecules
45. Chen TY, Yang CT, Kristiani E, Cheng CT (2019) On construction of a power data lake platform using spark. Lect Notes Electr Eng 542:99–108
46. Sawadogo PN, Scholly É, Favre C, Ferey É, Loudcher S, Darmont J (2019) Metadata systems for data lakes: models and features. Commun Comput Inf Sci 1064:440–451
47. Rajadnye A (2019) Datawarehouse versus datalake. SSRN Electron J
48. Ravat F, Zhao Y (2019) Metadata management for data lakes. Commun Comput Inf Sci 1064:37–44
49. Singh A (2019) Architecture of data lake. Int J Sci Res Comput Sci Eng Inf Technol 5(2):411–414

Author Index

P. Johri et al. (eds.), *Applications of Machine Learning*,
Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-3357-0