Computer Science and Information Technology

David C. Wyld,
Dhinaharan Nagamalai (Eds

Computer Science & Information Technology

- 3rd International Conference on Natural Language Processing & Applications (NLPA 2022)
- 8th International Conference on Artificial Intelligence and Soft Computing (AIS 2022)
- 3rd International Conference on Big Data and Applications (BDAP 2022)
- 8th International Conference on Software Engineering (SOENG 2022)
- 8th International Conference on Image Processing and Pattern Recognition (IPPR 2022)
- 9th International Conference on Computer Science and Information Technology (CSIT 2022)

Published By

AIRCC Publishing Corporation

Volume Editors

David C. Wyld,
Southeastern Louisiana University, USA
E-mail: David.Wyld@selu.edu

Dhinaharan Nagamalai (Eds),
Wireilla Net Solutions, Australia
E-mail: dhinthia@yahoo.com

ISSN: 2231 - 5403
ISBN: 978-1-925953-73-2
DOI: 10.5121/csit.2022.121401 - 10.5121/csit.2022.121410

Typesetting: Camera-ready by author, data conversion by NnN Net Solutions Private Ltd., Chennai, India

Preface

3rd International Conference on Natural Language Processing & Applications (NLPA 2022),August 21~22, 2022, Chennai, India, 8th International Conference on Artificial Intelligence and Soft Computing (AIS 2022), 3rd International Conference on Big Data and Applications (BDAP 2022) ,8th International Conference on Software Engineering (SOENG 2022), 8th International Conference on Image Processing and Pattern Recognition (IPPR 2022), 9th International Conference on Computer Science and Information Technology (CSIT 2022) was collocated with 9th International Conference on Computer Science and Information Technology (CSIT 2022). The conferences attracted many local and international delegates, presenting a balanced mixture of intellect from the East and from the West.

The goal of this conference series is to bring together researchers and practitioners from academia and industry to focus on understanding computer science and information technology and to establish new collaborations in these areas. Authors are invited to contribute to the conference by submitting articles that illustrate research results, projects, survey work and industrial experiences describing significant advances in all areas of computer science and information technology.

The NLPA 2022, AIS 2022, BDAP 2022, SOENG 2022, IPPR 2022 and CSIT 2022 Committees rigorously invited submissions for many months from researchers, scientists, engineers, students and practitioners related to the relevant themes and tracks of the workshop. This effort guaranteed submissions from an unparalleled number of internationally recognized top-level researchers. All the submissions underwent a strenuous peer review process which comprised expert reviewers. These reviewers were selected from a talented pool of Technical Committee members and external reviewers on the basis of their expertise. The papers were then reviewed based on their contributions, technical content, originality and clarity. The entire process, which includes the submission, review and acceptance processes, was done electronically.

In closing, NLPA 2022, AIS 2022, BDAP 2022, SOENG 2022, IPPR 2022 and CSIT 2022 brought together researchers, scientists, engineers, students and practitioners to exchange and share their experiences, new ideas and research results in all aspects of the main workshop themes and tracks, and to discuss the practical challenges encountered and the solutions adopted. The book is organized as a collection of papers from the NLPA 2022, AIS 2022, BDAP 2022, SOENG 2022, IPPR 2022 and CSIT 2022

We would like to thank the General and Program Chairs, organization staff, the members of the Technical Program Committees and external reviewers for their excellent and tireless work. We sincerely wish that all attendees benefited scientifically from the conference and wish them every success in their research. It is the humble wish of the conference organizers that the professional dialogue among the researchers, scientists, engineers, students and educators continues beyond the event and that the friendships and collaborations forged will linger and prosper for many years to come.

David C. Wyld,
Dhinaharan Nagamalai (Eds)

Demian Antony D'Mello,	Canara Engineering College, India
Dimitris Kanellopoulos,	University of Patras, Greece
Dinesh Reddy,	SRM university, India
Diptiranjan Behera,	The University of the West Indies, Jamaica
Dongping Tian,	Baoji University of Arts and Sciences, China
Douglas Alexandre Gomes Vieira,	Enacom, Brazil
ElzbietaMacioszek,	Silesian University of Technology, Poland
Enrique Chirivella Perez,	University West of Scotland, UK
Everton Flemmings,	iValley Nutraceuticals-President, Canada
Faeq A.A.Radwan,	Near East University, Turkey
Fahmi El-Ssayed,	American university of the Middle East, Kuwait
Fang Wang,	Wilfrid Laurier University, Canada
Felix J. Garcia Clemente,	University of Murcia, Spain
Fernando Zacarias Flores,	Universidad Autonoma de Puebla, Mexico
Firas Ali Al Laban,	Sultan Moulay Slimane University, Morocco
Francesco Zirilli,	Sapienza Universita Roma, Italy
Garcia Clemente,	University of Murcia, Spain
Giuseppe Carbone,	University of Calabria, Italy
Gordana Jovanovic	Dolecek, Institute INAOE Puebla, Mexico
Grigorios N. Beligiannis,	University of Patras, Greece
Grzegorz Sierpiński,	Silesian University of Technology, Poland
H.V.Ramakrishnan,	DRMGR Educational and Research Institute, India
Habil. Gabor Kiss,	Obuda University, Hungary
Hamed Taherdoost,	University Canada West, Canada
Hamid Ali Abed AL-Asadi,	Iraq University College, Iraq
Hector Migallon,	Miguel Hernandez University, Spain
Hedayat Omidvar,	National Iranian Gas Company, Iran
Henok Yared Agizew,	Mettu University, Ethiopia
Hlaing Htake Khaung Tin,	University of Information Technology, Myanmar
Holger Kyas,	University of Applied Sciences Berne, Switzerland
Hyun-A Park,	Honam University, South Korea
Hyunsung Kim,	Kyungil University, Korea
Ibrahim Hamzane,	Hassan II University of Casablanca, Morocco
Ikvinderpal Singh,	Trai Shatabdi GGS Khalsa College, India
Isa Maleki,	Science and Research Branch, Iran
Islam Tharwat Abdel Halim,	Nile University, Egypt
Israa Shaker Tawfic,	Ministry of Science and Technology, Iraq
Janusz Kacprzyk,	Polish Academy of Sciences, Poland
Jawad K. Ali,	University of Technology, Iraq
Jinwei Liu,	Florida A&M University, United States of America
Kamal Khalilpour,	Islamic Azad University, Iran
Kamran Iqbal,	University of Arkansas at Little Rock, United States
Karim Mansour,	University Salah Boubenider, Algeria
Ke-Lin Du,	Concordia University, Canada
Klenilmar Lopes Dias,	Federal Institute of Amapa, Brazil
L. Iliev,	FON University, Macedonia
Loc Nguyen,	Loc Nguyen's Academic Network, Vietnam
Luis Gomez,	University of Las Palmas de Gran Canaria, Spain
Luisa Maria Arvide Cambra,	University of Almeria, Spain
Malka N. Halgamuge,	The University of Melbourne, Australia
Mallikharjuna Rao K,	International Institute of Information Technology, India
Mario Versaci,	DICEAM - Univ, Italy

Marius Cioca,	Lucian Blaga University of Sibiu, Romania
Masoomeh Mirrashid,	Semnan University, Iran
Maumita Bhattacharya,	Charles Sturt University, Australia
Michail Kalogiannakis,	University of Crete, Greec
Ming-An Ching,	National Taipei University of Technology, Taiwan
Mohamed Fakir,	Sultan Moulay Slimane University, Morocco
Mohamed Hassiba,	Benbouali University Chlef, Algeria
Mohammad Shahid Husain,	University of Technology & Applied Sciences, Oman
Mohd Ashraf Ahmad,	Universiti Malaysia Pahang, Malaysia
Mostafa Rashdan,	American university of the middle east, Kuwait
Mu-Song Chen,	Da-Yeh University, Taiwan
Mu-Yen Chen,	National Cheng Kung University, Taiwan
Nameer N. EL-Emam,	Philadelphia University, Jordan
Ngoc Hong Tran,	Vietnamese-German University, Vietnam
Nikola Ivković,	University of Zagreb, Croatia
Nur Eiliyah Wong,	Researcher, Malaysia
Oleksii K. Tyshchenko,	University of Ostrava, Ostrava
Oliver L. Iliev,	Fon University, Republic of Macedonia
Omar Khadir,	Hassan II University of Casablanca, Morocco
Omid Mahdi Ebadati E,	Kharazmi University, Tehran
Osman Toker,	Yildiz Technical University, Turkey
Palanivelrajan,	M.Kumarasamy College of Engineering, India
Pascal Lorenz,	University of Haute Alsace, France
Paul Bogdan,	University of Southern California, USA
Petra Perner,	Futurelab Artificial Intelligence IBaI-2, Germany
Pietro Guccione,	Politecnico di Bari, Italy
Piotr Kulczycki,	Systems Research Institute, Poland
Pr. Pascal Lorenz,	University of Haute Alsace, France
Priyanka Surendran,	University of Technology Kingdom, Bahrain
PrzemyslawFalkowski-Gilski,	Gdansk University of Technology, Poland
Quang Hung Do,	University of Transport Technology, Vietnam
Rafael Socas,	University of Technology, Spain
Rajeev Kanth,	University of Turku, Finland
Rajesh Bose,	Brainware University, India
Ramadan Elaiess,	University of Benghazi, Libya
Ravikumar CV,	VIT University Vellore, India
Reena Malik,	Chitkara University, India
Richa Purohit,	DY Patil International University, India
Robert Ssali Balagadde,	Kampala international University, Uganda
Rodrigo Pérez Fernández,	Universidad Politécnica de Madrid, Spain
Ruhaidah Samsudin,	Universiti Teknologi Malaysia, Malaysia
Rusudan Makhachashvili,	Borys Grinchenko Kyiv University, Ukraine
S.Sridhar,	Srm Easwari Engineering College, India
Saad Al- Janabi,	Al-hikma college university, Iraq
Sabyasachi Pramanik,	Haldia Institute of Technology, India
Sachin Umrao,	University of California San Francisco, USA
Safawi Abdul Rahman,	Universiti Teknologi MARA, Malaysia
Sahar Saoud,	Ibn Zohr University, Morocco
Saif aldeen Saad Obayes Alkadhim,	Shiite Endowment Diwan, Iraq
Saikumar Tara,	CMR Technical Campus, India
Samir Kumar Bandyopadhyay,	University of Calcutta, India
Santosh Kumar Bharti,	Pandit Deendayal Energy University, India

Technically Sponsored by

Computer Science & Information Technology Community (CSITC)

Artificial Intelligence Community (AIC)

Soft Computing Community (SCC)

Digital Signal & Image Processing Community (DSIPC)

9th International Conference on Computer Science and Information Technology (CSIT 2022)

Giant components in texts generated by a stream

Achraf Lassoued

University of Paris II and IRIF-CNRS

Abstract. Given a text stream, we associate a stream of edges in a graph G and study its large clusters by analysing the giant components of random subgraphs, obtained by sampling some edges with different distributions. For a stream of Tweets, we show that the large giant components of *uniform sampled* edges of the Twitter graph reflect the large clusters of G. For a stream of text, the *uniform sampling* is inefficient but the *weighted sampling* where the weight is proportional to the Word2vec similarity provides good results. Nodes of high degree of the giant components define the *central words* and *central sentences* of the text.

Keywords: NLP, Streaming algorithms, Clusterin, Dynamic graphs.

1 Introduction

We observe streams of high volume of incoming text and study the classification and the sentiment analysis online, without storing the entire data. We consider messages generated by social network or by text servers. Twitter, for example, generates streams of tweets which are transformed into streams of graph edges where the nodes are the tags and edges link the author of the tweet with the tags present in the text. Text servers generate text sentences which are transformed into graph edges where the nodes are the words and the edges link two words of the same sentence.

We study the large clusters of these graphs in sliding windows by sampling a fixed number of edges with two different distributions, the *uniform distribution* or a *weighted distribution*. The analysis is based on the study of the *giant components* of these random subgraphs.

For Twitter applications, we typically receive 10^3 tweets per minute, approximately 3.10^3 edges per minute. We analyse random subgraphs online in sliding windows of length $\tau = 10$ minutes. We sample the edges *uniformly* for each sliding window using a *Reservoir sampling* [23], and analyse the giant components of the Reservoir of constant size K. We only keep the giant components of each window. We interpret the giant components as *topics* and follow the various topics in time. We can also correlate different streams, based on the comparison of their giant components.

For text applications, an RSS-stream of news articles may generate a large stream of sentences. The nodes of the generated graph are the words and an edge is a pair of words in the same sentence. If we sample uniformly the edges, the giant components are however not stable. We observed that if we sample the edges proportionally to the similarity of the words, given by Word2vec [14], the giant components become stable. We use the classical k-means algorithm to classify the text, with the Jaccard distance between giant components. Our main contributions are:

- an analysis of streaming texts by giant components of random graphs: *the uniform sampling* is efficient for Social media such as Twitter, whereas the *weighted sampling* where the weight is the Word2vec similarity between words is efficient for streaming texts,
- a classification technique, based on a natural distance between components, useful for topicalization and an analysis of the nodes of high degree of the giant components which define the *central words* and *central sentences* of the text.

– the definition of a *sentiment index* associated with a text within a text reference L, the basis for a sentiment analysis.

In section 2, we introduce the main concepts. In section 3 we define the giant components in random graphs and the sampling methods. In section 4, we define a distance between giant components and use the k-means for classification. In section 5, we describe how detect offensive messages which have a high impact. In section 6, we describe experiments for analysing Twitter streams and for classifying and analysing standard texts.

2 Preliminaries

We first review Streaming algorithms, Social Networks and some of the classical statistical methods used in Natural language processing.

2.1 Streaming algorithms

These algorithms read data step by step and maintain a small memory, if possible constant, *poly*(log) or at least sublinear in the size of the stream. Classical algorithms may consider a stream of numerical values $x_i \in \{1, 2, ...n\}$, of words on an alphabet Σ, or of edges $e_i = (v_j, v_k)$ of a graph $G = (V, E)$ where $v_j, v_k \in V$.

An important technique called the *Reservoir sampling* [23] keeps K elements of the stream with a *uniform distribution*. In a stream of length m each element has probability K/m to be chosen in the Reservoir. The *weighted Reservoir sampling* keeps K elements of the stream with a *weighted distribution*, detailed in the appendix C. Each element e_i of weight w_i of the stream has probability $K \cdot w_i / \sum_i w_i$ to be chosen in the Reservoir. If K is sublinear, for example $O(\sqrt{m})$, we obtain a sublinear space algorithm.

2.2 Social Networks

In social networks and crowdsourcing we observe large streams of data online, mostly edges $e_1, ..., e_m$ of a graph. Given a set of tags such as {#Ethereum, #Bitcoin}, or {#Amazon}, Twitter provides a stream of tweets represented as Json trees whose content C (the text of the tweet) contains at least one of these tags. The *Twitter Graph* of the stream is the graph $G = (V, E)$ with multiple edges E where V is the set of tags $\#x$ or $@y$ and for each tweet sent by $@y$ which contains tags $\#x$,$@z$ we add the edges $(@y, \#x)$ and $(@y, @z)$ in E. In our approach, we consider the hypergraph where we add the content C to each edge. We have then the hyperedges $(@y, \#x, C)$ and $(@y, @z, C)$. The URL's which appear in the tweet can also be considered as nodes but we ignore them for simplicity. A stream of tweets is then transformed into a stream of edges $e_1, ...e_m, ...$, although each edge is an hyperedge, which also stores a timestamp.

Social networks such as Twitter evolve dynamically, and dense subgraphs appear and disappear over time as interest in particular events grows and disappears.

2.3 Large dense subgraphs

There are several appoaches to density in graphs with n nodes and m edges, described in the Appendix A. We are mainly interested in large clusters S and assume that $|S| > \delta\sqrt{n}$ for some parameter δ. The (γ, δ)-*large dense subgraph* problem, where $\gamma \leq 1$, takes as input a graph $G = (V, E)$ and decides whether there exists an induced subgraph $S \subseteq V$ such that $|S| > \delta\sqrt{n}$ and $|E[S]| > \gamma|S|(|S| - 1)/2$. Social graphs defined by a stream of edges

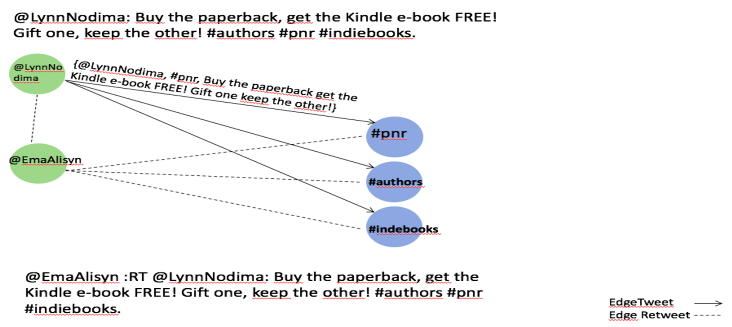

@LynnNodima: Buy the paperback, get the Kindle e-book FREE! Gift one, keep the other! #authors #pnr #indiebooks.
@LynnNodima
{@LynnNodima, #pnr, Buy the paperback get the Kindle e-book FREE! Gift one keep the other!}
@EmaAlisyn
#pnr
#authors
#indebooks
@EmaAlisyn :RT @LynnNodima: Buy the paperback, get the Kindle e-book FREE! Gift one, keep the other! #authors #pnr #indiebooks.
EdgeTweet
Edge Retweet

Fig. 1. Twitter graph for a tweet and a retweet

$e_1, ..., e_m, ...$ follow a specific regime for which a Reservoir of the size $K = O(\sqrt{n}.\log n)$ can detect (γ, δ)-large dense subgraphs with high probability [21]. Appendix A details this approach.

2.4 Analysis of Natural languages

Classification methods in Natural Languages also consider the analysis of clusters. The IRaMuTeQ [20] method enables hierarchical classifications based on PCA (Principle Components Analysis). The Word2vec method [14] associates with the words, vectors of v small dimension. Other embeddings [10, 19] are also possible. Extensions to sentences are considered in Word2Sense [18].

The $LDA(k)$ (Latent Dirichlet Allocation) [4] is a probabilistic method to analyse the matrix A where the lines are the documents and the columns are the principal words, and the value $A(i, j)$ is the number of occurrences of the word j in the document i. It constructs k classes among n documents. LDA, Dynamic Topic Models [5] and IRaMuTeQ techniques require to access the entire data.

Attention mechanisms [11, 22] provide, for a word w_i in a sentence, the distribution of the other most correlated words. The correlation of a word w_j with w_i is approximately the value $v(w_i).v(w_j)$. We can then compute $\{v(w_i).v(w_j) : j \neq i\}$ for a fixed w_i and normalize the values to obtain a distribution. It is mostly used in Transformers for machine translation. We show that the weighted Reservoir can reconstruct the most significant attention distributions.

3 Giant components in dynamic graphs

A giant component is a large connected components in a random graph. For the uniform sampling, the probability that an edge is selected is K/m, i.e. uniform for each edge. It follows the Erdös-Renyi model [9] for which the analysis of giant components is well understood.

A social graph follows a power law degree distribution and can be generated by the configuration model, see section B of the appendix. The existence of giant components is studied in [15].

Our model takes a social graph which follows a power law degree distribution, and sample it uniformly. It is a combination of the configuration model and the Erdös-Renyi model. The analysis for the existence of giant components is given in the section B of the appendix.

For each large connected component C, we approximate a cluster by 2-core(C), defined in the appendix B. The component C alone may not be a good approximation of the cluster. We then store these simplified components for different windows.

3.1 Stability

We can experimentally verify the existence of the giant components by performing independent experiments and measuring the stability. Consider two experiments on the same stream (or document D) with two *independent* Reservoirs. Let V_1 (resp. V_2) be the set of vertices of the giant component C_1 (resp. V_2). Let the *stability* of the stream D and Reservoir size K, be the random variable $\rho(D,k)$ defined as:

$$\rho(D,k) = \frac{|V_1 \cap V_2|}{|V_1|}$$

The random variable depends on two experiments and we can also take its expectation $I\!\!E(\rho(D,k))$ for n experiments. For Twitter streams, the uniform sampling provides a good stability, as suggested by the theoretical analysis of giant components. On the contrary, for texts the stability of the uniform sampling is close to 0.

4 Classification

There are several possible distances between giant components which can be generalized to sets of giant components. For simplicity, we first use the *Jaccard distance* $\mathsf{dist}_J(C_1, C_2) = 1 - J(V_1, V_2)$ where $J(V_1, V_2)$[1] is the Jaccard similarity between the domains V_1 and V_2. Other distances such as the Edit distance would take edges into account.

A sequence of sliding windows generates the giant components $C_1, ...C_n$ with the distances between pairs. We can group them in k-classes, with the classical k-means algorithm. Each class i has a representative C_i. For a new giant component C, we just check the Min_i $\mathsf{dist}(C_i, C)$ to classify C. The classification can be extended to sliding windows if we consider sets of giant components.

5 Measures of sentiments

The classical Word2Vec analysis takes texts L as inputs and construct vectors v of small dimension such that for two words w_i, w_j of the texts, the relative frequency of w_i, w_j in a sentence is proportional to the scalar $v^t(w_i).v(w_j)$. We take Twitter messages [24] as a benchmark and observe streams of Tweets, which are transformed in streams of edges of a Twitter graph. We sample edges uniformy in a Reservoir of size k, or with a weighted distribution which depends on the vectors v. The Reservoir is random graph which contains giant components. The edges of the nodes of high degree of the giant components define the tweets t_i of High Impact.

For the text analysis we take the tweets t_i and compute the weight $\rho(t_i, L)$ for a reference L, defined as follows:

$$\rho(t_i, L) = \sum_{(w_j, w_k) \in t_i} v^t(w_j).v(w_k)$$

We interpret $(w_j, w_k) \in t_i$, as the pair of words w_j, w_k appears in the same sentence of the tweet t_i. For a natural language such as English, L is the basic text reference, for

[1] The Jaccard similarity or Index between two sets A and B is $J(A,B) = |A \cap B|/|A \cup B|$.

example Wikipedia, and $v's$ are the associated vectors. We then construct a list L_o of *offensive* texts, which we want to detect and construct the associated vectors v_o. If only a few examples are available, we only modify the vectors $v's$ associated with the offensive words, and use the same vector $v's$ for the standard words.

These vectors can be very different from the $v's$. We then compare $\rho(t_i, L)$ and $\rho(t_i, L_o)$. A tweet t_i is *abusive* if $\rho(t_i, L_o) > c \cdot \rho(t_i, L)$ for some constant c.This approach generalizes to different natural languages L_i and various offensive texts $L_{o,i}$.

5.1 The detection of offensive High-Impact tweets

A *High-Impact* tweet is associated with an edge connected to a node of high degree. In Figure 2, the node 1 is of high degree and the edges e_1, e_4, e_5, e_8 correspond to High-Impact tweets.

Consider the Twitter stream associated with the tag *#Metoo*. A High Impact tweet is t_i: *@robinmonotti2: Forced or coerced vaccination is medical rape: a crime. We need a medical #MeToo movement now.* There are 7 words (@robinmonotti2, Forced, coerced, vaccination, medical, rape, crime) in the first sentence hence $42 = \binom{7}{2}$ pairs. In the second sentence, there are 5 words hence 10 pairs. In this case, $\rho(t_i, L_o) = 0.18$ and $\rho(t_i, L = 0.78)$, using the publicly available datasets that cover different hate speech-related categories[2]. We conclude that this tweet is not offensive.

5.2 Standard texts

We can apply our method to a standard text corpus. For each sentence, we first apply the *lemmatization* and the *Entity recognition* [16] steps and delete the stop words and the least frequent words. We generate either the bigrams (contiguous pairs of words) [17] or all the possible pairs of a given sentence, as potential edges. For the uniform sampling the weights of the edges are constant (for example equal to 1). For the weighted sampling, the weight of an edge is the Word2vec similarity of two words. In both cases, we process the text as a stream *without storing the entire text.*

6 Experiments

We propose a tool with two versions: the first version analyses Twitter streams on specific tags and the second version analyses texts[3]. We set the parameters of the windows (length τ and step λ) and the size K of the Reservoir and proceed online.

6.1 Twitter streams

The Twitter version[4] is useful to detect trends and correlate different streams. A stream, determined by some keywords, generates edges such as $(u, v, text)$[5] where u is the author and v one of the selected tags in the *text*, the original tweet.

With a keyword such as *CNN*, we obtain: 3.10^3 edges and 10^3 sentences per minute. For a window of length $\tau = 10$ minutes, we have approximately $m = 3.10^4$ and $n = 8000$,

[2] https://github.com/alassou/t/blob/master/labeled_data.csv
[3] https://github.com/alassou/t/blob/master/wr.ipynb
[4] https://github.com/alassou/t/blob/master/topic.ipynb
[5] (*@thomasjacksonjr*, *#PrimeVideo*, "Watched *#ABCMurders* from beginning to end and it was pure #mystery and delight. Excellent show and a must watch for *#mysterylovers*. Great show on *#PrimeVideo*.")

as $m = O(n. \log n)$. The size $K = 400$ of the Reservoir is of the order $O(\sqrt{n}. \log n) \simeq 90.4 = 360$. The stability for the *uniform sampling* of the twitter stream is 0.9. Figure 2 gives the 2-core(C) of a giant component.

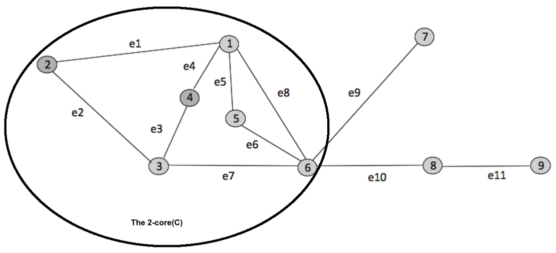


Fig. 2. A large connected component C in the Reservoir and its 2-core(C) (in the circle)

We captured 4 twitter streams on the tags #CNN, #FoxNews, #Bitcoin, and #Ripple during 24 hours with a window size of $\tau = 1h$ and a time interval $\lambda = 30$mins. Figure 3 indicates the number of edges in a window, approximately $m = 30.10^3$ per stream, for each stream, hence 10^6 edges in 24 hours. For the #Bitcoin and $k = 400$, the size of the giant component is approximately 100.

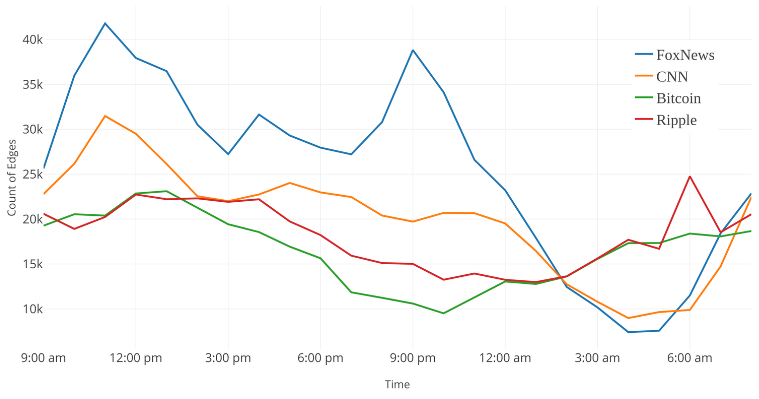


Fig. 3. Number of edges/per hour for 4 streams during 24h

6.2 Classical texts

We evaluate our methods on the NIPS dataset, (Googleplaystore_user_reviews) dataset and the Stanford Natural Language Inference (SNLI) corpus [7] which consists in 570k human-written English pairs: sentence and annotations.

We apply the uniform sampling and the weighted sampling where the weight is the absolute value of the Word2Vec similarity between two words. The stability of the two methods as a function of K is given in Figure 4.

The weighted sampling gives much better result as the uniform sampling stability is less then 0.5.

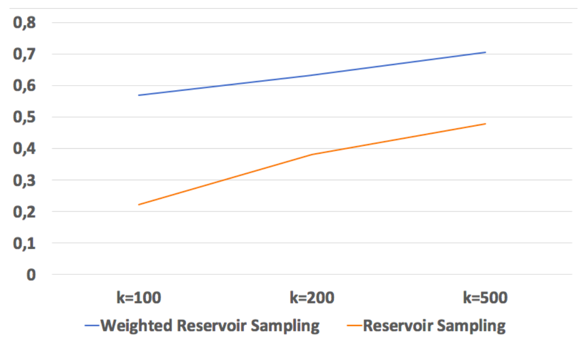


Fig. 4. Stability of the uniform and weighted sampling

6.2.1 Central sentences and Attention mechanisms Each giant component can be analysed, starting from the node of maximal degree, the *center*, and their adjacent edges, the *central edges*. For the 2-core of Figure 2, we start with the center node 1 of maximum degree and its adjacent central edges e_1, e_4, e_5, e_8. Each edge is of the form $(u, v, text)$ and we can analyse the attention distribution[6] [22] of the words u and v in each "text" sentence.

The e_8 edge is *(skate, jump, "A boy is jumping on skateboard in the middle of a red bridge.")* and the e_4 edge is *(skate, sidewalk, "The boy skates down the sidewalk.")*. The attention analysis of the first sentence for the words *skate* and *jump* is given in the Figure 5. A giant component provides a *central node* and *central sentences:* the 4 sentences

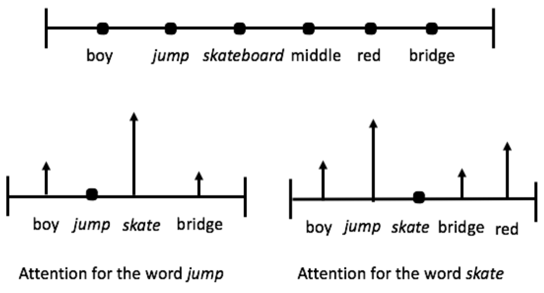


Fig. 5. The analysis of the sentence: *A boy is jumping on skateboard in the middle of a red bridge.*

associated with the edges of the central node *skate*, and then recursively along the tree decomposition of the component. At the next stage, the node 6 would be explored with three new edges.

If we classify the giant components into k-classes, viewed as topics, each component C would have a natural distribution over the topics as in [4].

[6] For a word u, its *attention in a sentence* is the distribution over the other words with a weight proportional to its Word2Vec similarity.

7 Conclusion

We propose sampling techniques to analyse streams of tweets and standard texts in a streaming mode. We observe the giant components of the Reservoirs in sliding windows and introduce a distance between components. The uniform sampling is stable for Twitter, and only the weighted sampling is stable for classical texts. We can classify these components with the k-means algorithm and detect offensive tweets. Each giant component defines central words and central sentences.

References

1. Aggarwal, C.C.: An introduction to cluster analysis. In: Data Clustering: Algorithms and Applications, pp. 1–28. CRC Press (2013)
2. Babcock, B., Datar, M., Motwani, R.: Sampling from a moving window over streaming data. In: Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 633–634 (2002)
3. Bahmani, B., Kumar, R., Vassilvitskii, S.: Densest subgraph in streaming and mapreduce. Proc. VLDB Endow. **5**(5), 454–465 (Jan 2012)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Lattent dirichlet allocation. Journal of Machine Learning Research **3:993-1022** (2003)
5. Blei, D., Lafferty, J.: Dynamic topic models. vol. 2006, pp. 113–120 (2006). https://doi.org/10.1145/1143844.1143859
6. Bollobas, B.: Random Graphs. Cambridge University Press (2001)
7. Bowman, S., Angeli, G., Potts, C., Manning, C.: A large annotated corpus for learning natural language inference (08 2015). https://doi.org/10.18653/v1/D15-1075
8. Braverman, V., Ostrovsky, R., Zaniolo, C.: Optimal sampling from sliding windows. In: Proceedings of the Twenty-eighth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. pp. 147–156 (2009)
9. Erdös, P., Renyi, A.: On the evolution of random graphs. In: Publication of the mathematical institute of the Hungarian Academy of Sciences. pp. 17–61 (1960)
10. Li, X.L., Eisner, J.: Specializing word embeddings (for parsing) by information bottleneck. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. pp. 2744–2754 (2019)
11. Lin, Z., Feng, M., dos Santos, C.N., Yu, M., Xiang, B., Zhou, B., Bengio, Y.: A structured self-attentive sentence embedding. CoRR **abs/1703.03130** (2017)
12. Mathieu, C., de Rougemont, M.: Large very dense subgraphs in a stream of edges. CoRR **abs/2010.07794** (2020), `https://arxiv.org/abs/2010.07794`
13. McGregor, A., Tench, D., Vorotnikova, S., Vu, H.T.: Densest subgraph in dynamic graph streams. CoRR **abs/1506.04417** (2015)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR **abs/1301.3781** (2013)
15. Molloy, M., Reed, B.: The size of the giant component of a random graph with a given degree sequence. Comb. Probab. Comput. **7**(3), 295–305 (1998)
16. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1), 3–26 (Jan 2007)
17. Nawab, R.M.A., Stevenson, M., Clough, P.: Comparing Medline citations using modified N-grams. Journal of the American Medical Informatics Association **21**, 105–110 (2013)
18. Panigrahi, A., Simhadri, H.V., Bhattacharyya, C.: Word2Sense: Sparse interpretable word embeddings. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 5692–5705. Association for Computational Linguistics (2019)
19. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Proc. of NAACL (2018)
20. Ratinaud: Iramuteq: Interface de r pour les analyses multidimensionnelles de textes et de questionnaires [computer software]. http://www.iramuteq.org (2009)
21. de Rougemont, M., Vimont, G.: The content correlation of streaming edges. In: IEEE International Conference on Big Data. pp. 1101–1106 (2018)
22. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. CoRR **abs/1706.03762** (2017)
23. Vitter, J.S.: Random sampling with a reservoir. ACM Trans. Math. Softw. **11**(1), 37–57 (1985)

24. Warner, W., Hirschberg, J.: Detecting hate speech on the world wide web. In: Proceedings of the Second Workshop on Language in Social Media. p. 19–26. LSM '12, Association for Computational Linguistics, USA (2012)

Appendix

A Large dense subgraphs

Let $S \subseteq V$ and $E(S)$ be the set of internal edges i.e. edges $e = (u, v)$ where $u, v \in S$. The classical density of S is the ratio $\rho = |E[S]|/|S|$. One may want to find subgraphs with S nodes which maximize ρ. In the case of a stream of edges, the approximation of dense subgraphs is well studied in [13] and an $\Omega(n)$ space lower bound is known [3]. In [1], another different objective is considered: a γ-*cluster* of domain S is a *subgraph* which depends on a parameter $\gamma \leq 1$ such that $|E(S)| \geq \gamma.|S|.|S-1|/2$. This problem is also hard to approximate when S is large. We consider $|S| > \delta\sqrt{n}$ for some other parameter δ, hence the (γ, δ)-*large dense subgraph* problem.

A family of graphs G_n has a (γ, δ)-cluster if for n large enough there exists S such that S is a (γ, δ)-cluster for G_n. Classical community detection algorithms take the entire graph as input and apply spectral techniques. We consider a specific regime of graphs defined by a stream of edges $e_1, ..., e_m, ...$ which follow a power-law degree distribution μ and use a sublinear space algorithm. A Reservoir sampling [23] with $K = O(\sqrt{n}. \log n/\gamma \cdot \delta)$ edges can detect (γ, δ)-large dense subgraphs (clusters) with high probability [12], on the specific class of graphs taken from μ. We use a *one-sided stochastic randomized* algorithm A to detect the existence of a cluster:

- If G has a cluster , $Prob_{\Omega}[A(x)\ accepts] \geq 1 - \epsilon$
- If G is a random graph drawn from μ with no cluster, $Prob_{\mu\times\Omega}[A(x)\ rejects] \geq 1 - \epsilon$

B Giant components and sampling

In choosing K edges *uniformly*, each edge has a *uniform* probability K/m to be chosen in the Reservoir. It is an Erdös-Renyi model [9], written $G(n, p)$ with $p = K/m$.

In the classical Erdös-Renyi model, we start with the complete graph, whereas we start with a social graph G_n. A fundamental question is the existence of giant components[7] and phase transitions. In the Erdös-Renyi model $G(n, p)$, a giant component occurs if $p > 1/n$ i.e. when each edge of a clique is selected with probability p. If we generalize to a γ-cluster S, a giant component occurs if $p > 1/\gamma.|S|$.

Social graphs with m edges and n nodes follow a power law degree distribution μ, i.e. a Zipfian distribution where $Prob[d = j] = c/j^2$. The maximum degree is $\sqrt{c.n}$ and m is $O(cn \ln(n))$. The configuration model [6] studies random graphs with fixed degree distributions, such as the Zipfian law. The conditions to observe a giant component are given in [15].

We therefore combine the two models: we start with a social graph G with a power law degree distribution and then sample it uniformly: it is the configuration model for the Zipfian law followed by the Erdös-Renyi model.

If the size of a cluster S is larger than $\delta.\sqrt{n}$ and the Reservoir size $K > \frac{c.\sqrt{n}. \log n}{\gamma.\delta}$, then:

$$\frac{K}{m} \geq \frac{c.\sqrt{n}. \log n}{\gamma.\delta.c.n \ln(n)} = \frac{1}{\gamma.\delta.\sqrt{n}} \geq \frac{1}{\gamma.|S|}$$

In this case we observe a giant component C in the Reservoir. Studies in [15] show that if we take a random graph with a power law degree distribution, there is no giant

[7] A giant component is a connected component larger than a constant fraction of n, the number of nodes.

component in the Reservoir with high probability. It is the basis of the approach of [12] to detect clusters on a stream of edges without storing the entire graph but only $K > \frac{c.\sqrt{n}.\log n}{\gamma.\delta}$ edges. The detection algorithm detects γ-clusters of size larger than $\delta.\sqrt{n}$ with high probability as giant components of the Reservoir. This approach can be generalized to dynamic windows as presented in [2, 8].

Let 2-core(C) be the graph obtained from the connected component C by removing the nodes of degree 1 repeatedly[8]. It can be shown that 2-core(C) is a good approximation of a γ-cluster of size $\delta.\sqrt{n}$. We store the 2-core(C) for each large C, as the witness of the clusters.

C Weighted Reservoir sampling

We read a stream of edges $e_1, e_2, ..., e_m, ...$ where each edge e_i has a weight w_i and keep K edges. We keep the K first edges in the Reservoir R. For each new edge e_i, where $i > K$, we decide to select e_i with probability:

$$\frac{K \cdot w_i}{\sum_{j \leq i} w_j}$$

If we select e_i we insert it in the Reservoir in a random position j, replacing the current element: we select $1 \leq j \leq K$ with probability $1/K$.

[8] The k-core of a graph G is obtained by removing repeatedly all the notes of degree less than $k - 1$

AUTHOR IDENTIFICATION USING TRADITIONAL MACHINE LEARNING MODELS

Ojaswi Binnani

Language Technologies Research Centre,
International Institute of Information Technology-Hyderabad, India

ABSTRACT

The Internet has many useful resources with bountiful information at our fingertips. However, there are nefarious uses to this resource, and can be misused in cybercrime, fake emails, stealing content, plagiarism etc. In many cases, the text is anonymously written, and it is important to accurately find the author to bring the criminal to justice. The topic of author identification helps with this task, where from a set of suspect authors, the writer of a given text will be determined. We aim to create a computationally non-complex model that works to find the author of a given text. The model will not require as much data as deep learning methods. This paper focuses on the use of various stylometric and word-based features as well as different machine learning models to create a classifier that gives the best accuracy. We find that the XGBoosting algorithm performs this task with a good accuracy.

KEYWORDS

Author Identification, Forensic Linguistics, Machine Learning.

1. INTRODUCTION

The Internet has a vast and large knowledge base that has many advantages in connecting reliable content and real people. However, the Internet has also paved the way for cybercrime such as plagiarism from other legitimate content-creators. In this situation, when it is difficult to detect where the content or text originated from, this task can prove to be useful.

Another scenario is within Forensic Linguistics. A concept called Linguistic Fingerprinting or Write Print theorizes that each person has a unique style of using language. According to Olsson [24], “A linguistic fingerprint is a concept put forward by some scholars that each human being uses language differently, and that this difference between people involves a collection of markers which stamps a speaker/writer as unique; similar to a fingerprint. Under this view, it is assumed that every individual uses languages differently and this difference can be observed as a fingerprint.” This concept is very attractive to the law enforcement, but there is hardly any evidence to prove this. It has however been used in cases such as the case against the Unabomber. If the author identification model with high accuracy can say that a piece of work is writ-ten by a specific suspect author, then this can be used as evidence (with other additional evidence) to bring justice.

Author Identification has also been used to credit people with works which were anonymously published. One such example is the disputed Federalist Papers. The Federalist papers were released to the public supporting the constitution of America written by Alexander Hamilton, James Madison and John Jay. It was a series of 85 articles published anonymously. Most of the

DOI: 10.5121/csit.2022.121402

writings were credited without dispute, but 12 of the articles were either written by Hamilton or Madison. Through Author Identification task done by Mosteller and Wallace [23], they concluded that it was Madison who wrote all twelve of the disputed Federalist articles. Other studies [3; 10] also supported this conclusion.

We divide the author identification task into two sub-tasks viz. extracting features that can accurately represent the text and testing various traditional machine learning models to see which works best for this task.

To choose features that represent a given text, we explore the different types of features that exist and examples of each type and choose a subset that works for classifying the author.

For our second task of testing various traditional machine learning models, we set the task as a single-label multi-class classification problem where each author is a single class, and each text can be written by only one author. Then we test different types of algorithms explained further in subsection 6.2.

In section 2 we look at related work that has been done in this area; in section 3 we look at the dataset used. In section 4 we discuss the different types of features that exist. We go into depth of these features in subsection 5.1 and subsection 6.1. Machine Learning models are discussed in subsection 5.2 and subsection 6.2.

2. RELATED WORK

Khan et al. [16] discussed the differences between the three tasks in author analysis - one of which is author identification. The paper discusses different types of features and types of models used in author identification and proposes a method to identify an author of an email.

Zheng et al. [32] looked at authorship identification in online messaging. One of the languages they looked at was English and came up with 270 features categorized into five types: lexical features, word-based features (not similar to our word-based feature classification in section 5), syntactic features, structural features and content specific features. Using these 270 features, Li et al. [19] found that a Support Vector Machine (SVM) outperformed the other models. This proves that a traditional model with sufficient meaningful features can be used in author identification. Mohsen et al. [22]; De Vel [6] also used SVMs for author identification.

Zhou and Wang [33]; Qian et al. [27] worked with three stylometric features: average sentence length, average word length and Hapax Legomenon Ratio and applied Glove Vector Embeddings to news articles. They worked with RNN and LSTM respectively. Their models suffered from an overfitting problem due to the computational complexity of their models and their insufficient dataset.

Iqbal et al. [13] has a list of features, and for any two authors has a subset of features that are applicable to one author and not the other. They call this the write print between two authors, and use these features to classify the author of a malicious email.

3. DATASET

In the paper, we use the Reuter 50 50 dataset [27]. This dataset was developed by ZhiLiu in 2011 and has been used in author identification experiments since then.

The dataset consists of 50 authors' works. Each author has 50 articles in the training set and 50 articles in the testing set. There is no overlap of articles between the training set and testing set. Each of the articles is in the news genre.

For each author, the number of words per article ranges from 468 to 569 words, averaging around 512 words per article. The sizes of the articles range from 2.7 kB to 3.6kB, averaging at 3.13 kB per article.

In this paper, the training and testing sets are combined and then a 90-10 split is done for training and testing.

4. FEATURE TYPES

There are three different types of features that we will use to represent each article in the training set.

4.1. Word-Based Features

These features are based off the words or phrases an author commonly uses. This is very effective when the author commonly uses specific adjectives or writes with a distinctive vocabulary. However, the problem with these types of features is that content words such as words that are only used in the case of a specific topic. An author may be writing on a topic that uses certain jargon, but it is not indicative of the author's personal style. The classification may fail when another author is writing that topic and uses the same jargon.

Some examples of this type of feature include Bag of Words (BOW)[31; 20], Doc2Vec [18], Glove Vector Embeddings [26] and n-grams [14].

4.2. Stylometric-Based Features

Since the previous feature may not be accurate on its own, we use stylometric features as well. These features aim to be similar for a single author despite the different topics of the author's texts. Vocabulary richness scale can be one such feature since the variety of vocabulary has little correlation with the topic of the text. Another feature can be the average sentence length or average word length. Both features refer to the author's choice in sentence structure or vocabulary but not necessarily to the topic of the text.

Stylometric features can be

- article-level e.g. number of paragraphs, number of sentences, number of words
- paragraph-level e.g. number of sentences in a paragraph, average length of sentence, number of words in a paragraph
- word-level e.g. number of small words (less than four characters), average length of words, frequency of each alphabet
- vocabulary richness e.g. Hapax Legomenon Ratio

4.3. Syntax-Based Features (Punctuation)

Syntax features can refer to the usage of function words such as determiners and auxiliary verbs. In this paper the features we focus on are punctuation features. Punctuation features can be used for author identification because certain punctuation can be a differentiator between authors. For example, some authors may use semicolons in between sentences and others may not. The number of commas in a sentence can be another distinguishing feature.

Examples of syntax-based features are the count of specific function words e.g. ‘between’, ‘a’, ‘nor’. Zheng et al. [32] found 150 function words that can be used as features. Another type of syntax-based features is the count of different punctuation e.g. exclamation points, commas, semi colons. Zheng et al. [32] found 8 features.

5. EXPERIMENTS

Zhou and Wang [33]; Qian et al. [27] implemented three stylometric features (average word length, average sentence length, and Hapax Legomenon Ratio) through traditional machine learning models such as Support Vector Machine, Gradient Boosting etc. gets a maximum accuracy of 12%.

To further increase the accuracy of authorship identification, rather than using computationally complex machine learning models such as RNNs and LSTMs (the methods implemented by Zhou and Wang [33]; Qian et al. [27]), more features were included to enhance the traditional models.

5.1. Feature Extraction

In Section 5, we discussed the various types of features that exist. To train our models, we need to select a viable subset from each type of feature for the task of author identification.

5.1.1. Word-Based Features

Doc2Vec [18] is the primary word-based feature in this paper. Doc2Vec is a modification to Word2Vec [21] and assigns a vector to each article in the dataset. The vectors represent the words in the article, the ordering of the words as well as their location in paragraphs. These vectors are numerical values that can be fed into a machine learning model.

5.1.2. Stylometric-Based Features

We use the features used by Zhou and Wang [33]; Qian et al. [27] (average word length, average sentence length and Hapax Legomenon Ratio) and the length of the article in words and sentences. The Hapax Legomenon Ratio is the ratio of the number of words used only once in the text to the total number of words in the text.

5.1.3. Syntax-Based Features

We use punctuation features in this paper. The usage of quotation marks, commas, dashes and exclamation marks in an article are counted and used as features in the model.

5.2. Machine Learning Models

Various machine learning models were tested using the above-mentioned features. They are vaguely classified into four types: Linear, Trees, Neighbours and Boosting.

5.2.1. Linear Machine Learning Models

Naive Bayes [29; 15] is a classification technique that often doesn't result in highly accurate outputs However, this is one of the simplest models to run, perfect for a baseline. Both Linear Regression [17; 2] and Support Vector Machine (SVMs) [25; 7; 30] using a linear kernel attempt to linearly separate the classes. Other kernels are possible with the SVM such as polynomials and sigmoids, however, in this task, the linear kernel works the best.

5.2.2. Neighbour-Based Classification Models

K-Nearest Neighbours (KNN) [5] classification is used in this paper. This classification technique is based on the Nearest Neighbour Algorithm. The main downfall of this method is that the results may be inaccurate if the dataset is skewed, i.e. one class has more data points than the other. The dataset we are using is balanced, and hence this method has the potential to decently classify the works.

5.2.3. Tree-Based Classification Models

Tree Based Machine Learning models are found to work decently with discrete models (non-quantifiable results) such as this task. They are also found to be good at capturing complex, non-linear relationships between the classes, hence if the linear models do not produce a desirable output, it is likely that the tree models will. Random Forest [4], Extra Trees [11] and Decision Trees [28] are used.

5.2.4. Boosting Models

Boosting has proven to be a useful machine learning method due to its speed and less complexity. It is a model based on using several weak models working in tandem to become a strong learner. Boosting, like tree-based models, are good at capturing complex, non-linear relationships and are good at discrete classification. AdaBoost (Adaptive Boosting) [8] and XGBoost (a variation on Gradient Boosting) [9; 1] are used in this paper.

6. RESULTS

6.1. Features

Table 1. Feature Importance – Ranking of Features from most to least important excluding Doc2Vec Positions

Average Sentence Length
Average Word Length
Number of Commas
Number of Dashes
Number of Quotation Marks
Hapax Legomenon Ratio
Number of Words
Number of Sentences
Number of Exclamation Marks

Table 1 represents all the features and their importance in the XGBoost Model. The least important features are the number of exclamation points and certain positions in the Doc2Vec vectors, while the more important features are the stylometric and syntax-based features as well as certain Doc2Vec vector positions.

We can see that the stylometric features contributed heavily to a good classification. The average sentence length proved to the most important feature within the whole group of features.
We can also see that the punctuation features (syntax features) are important with the usage of commas being the most important feature within this set.

6.2. Machine Learning Models

Table 2. Results of Various Machine Learning Models

Model	Training Accuracy	Testing Accuracy	F1 Score	Recall	Precision
Naïve Bayes	0.23	0.16	0.114	0.092	0.164
Liner Regression	0.42	0.36	0.330	0.333	0.344
SVM	0.59	0.48	0.463	0.464	0.486
K-Nearest Neighbour	1.00	0.26	0.246	0.247	0.267
Random Forest	1.00	0.71	0.716	0.731	0.735
Extra Trees	1.00	0.68	0.694	0.705	0.697
Decision Trees	1.00	0.57	0.577	0.578	0.588
AdaBoost	0.90	0.76	0.770	0.780	0.773
XGBoost	1.00	0.83	0.826	0.830	0.842

Table 2 shows the various scores for the model using the same feature set. As predicted, Naive Bayes performs the worst with a F1 score of 0.114. We can see all the Linear Models and the K-nearest Neighbour model performed with less than 50% accuracy which makes them worse than a weak model.

The Tree models performed with higher than 50% accuracy with Random Forest being the best within the Tree algorithm group with a F1 score of 0.716.

The Boosting models performed better than all other groups of models, XGBoost performing the best with an F1 score of 0.826. AdaBoost performed second best amongst all the algorithms with a F1 score of 0.770.

From these results we can see that the relationship between each of the classes is non-linear. All Linear classification models failed to classify the testing articles, and even failed to classify the training articles since they had an abysmal training accuracy. The models that are good at classifying non-linear data points such as Tree-Based Models and Boosting Models clearly perform well.

7. CONCLUSIONS

In this paper, we aimed to use traditional machine learning models to do the task of author identification using three types of features: word-based, stylometric and syntactic. We prove that all three of these features are useful in author identification by seeing their importance in the XGBoost model. We also see that Boosting Models perform the best amongst the traditional machine learning models since both Boosting models outperform the remaining models. XGBoost performs the best with 0.826 F1 score and 82.5% testing accuracy. This result is found due to the non-linear relationship between the classes. Hence, this is consistent with our hypothesis that traditional machine learning models can be used in author identification, which reduces the complexity to perform. It can work with an equal or better accuracy than that of complex machine learning models.

REFERENCES

[1] Bradly Boehmke and Brandon Greenwall. 2020. Chapter 12 Gradient Boosting. CRC Press.
[2] Bradly Boehmke and Brandon Greenwall. 2020. Chapter 4 Linear Regression. CRC Press.
[3] Robert A. Bosch and Jason A. Smith. 1998. Separating hyperplanes and the authorship of the disputed federalist papers. The American Mathematical Monthly, 105(7):601–608.
[4] L Breiman. 2001. Random forests. Machine Learning, 45:5–32.
[5] Padraig Cunningham and Sarah Delany. 2007. k-nearest neighbour classifiers. Mult Classif Syst.
[6] Olivier De Vel. 2000. Mining e-mail authorship.
[7] Theodoros Evgeniou and Massimiliano Pontil. 2001. Support vector machines: Theory and applications. volume 2049, pages 249–257.
[8] Yova Freund and Robert E. Schapire. 1995. A Decision Theoretic Generalization of On-Line Learning and an Application to Boosting. In Second European Conference on Computational Learning Theory (EuroCOLT-95), pages 23–37.
[9] Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. The Annals of Statistics, 29(5):1189 – 1232.
[10] Glenn Fung. 2003. The disputed federalist papers: Svm feature selection via concave minimization. pages 42–46.
[11] Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. Machine Learning, 63:3–42.
[12] John Houvardas and Efstathios Stamatatos. 2006. N-gram feature selection for authorship identification. volume 4183, pages 77–86.
[13] Farkhund Iqbal, Rachid Hadjidj, Benjamin C.M. Fung, and Mourad Debbabi. 2008. A novel approach of mining write-prints for authorship attribution in e-mail forensics. Digital Investigation, 5:S42–S51. The Proceedings of the Eighth Annual DFRWS Conference.
[14] D. Jurafsky and J.H. Martin. 2014. Speech and Language Processing. Always learning. Pearson.
[15] Pouria Kaviani and Sunita Dhotre. 2017. Short survey on naive bayes algorithm. International Journal of Advance Research in Computer Science and Management, 04.
[16] Sobiya Khan, Smita Nirkhi, and Rajiv Dharaskar. 2012. Author identification for e-mail forensic. Proceedings of National Conference On Recent Trends In Computing NCRTC.
[17] Khushbu Kumari and Suniti Yadav. 2018. Linear regression analysis study. Journal of the Practice of Cardiovascular Sciences, 4:33.
[18] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. 31st International Conference on Machine Learning, ICML 2014, 4.
[19] Jiexun Li, Rong Zheng, and Hsiu-chin Chen. 2006. From fingerprint to writeprint. Commun. ACM, 49:76–82.
[20] Shuming Ma, Xu Sun, Yizhong Wang, and Junyang Lin. 2018. Bag-of-words as target for neural machine translation. pages 332–338.
[21] Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. pages 1–12.
[22] Ahmed M. Mohsen, Nagwa M. El-Makky, and Nagia Ghanem. 2016. Author identification using deep learning. In 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), pages 898–903, Los Alamitos, CA, USA. IEEE Computer Society.
[23] Frederick Mosteller and David L. Wallace. 2012. Applied Bayesian and Classical Inference: The Case of The Federalist Papers. Springer Series in Statistics. Springer New York.
[24] John Olsson. 2008. Forensic Linguistics: An Introduction to Language, Crime and the Law. Bloomsbury Publishing.

[25] Edgar Osuna, Robert Freund, and Federico Girosi. 1970. Support vector machines: Training and applications. Tech Rep A.I. Memo No. 1602.
[26] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543.
[27] Chen Qian, Ting He, and Rao Zhang. 2017. Deep learning based authorship identification.
[28] J. R. Quinlan. 1986. Induction of decision trees. Mach. Learn., 1(1):81–106.
[29] Irina Rish. 2001. An empirical study of the naïve bayes classifier. IJCAI 2001 Work Empir Methods Artif Intell, 3.
[30] Konstantinos Veropoulos, N. Cristianini, and C. Campbell. 1999. The application of support vector machines to medical decision support: A case study. Adv Course Artif Intell.
[31] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. 2010. Understanding bag-of-words model: A statistical framework. International Journal of Machine Learning and Cybernetics, 1:43–52.
[32] Rong Zheng, Jiexun Li, Hsiu-chin Chen, and Zan Huang. 2006. A framework for authorship identification of online messages: Writing-style features and classification techniques. JASIST, 57:378–393.
[33] Liuyu Zhou and Huafei Wang. 2016. News authorship identification with deep learning.

BOOKSHELF – A DOCUMENT CATEGORIZATION FOR LIBRARY USING TEXT MINING

Carlo Petalver, Roderick Bandalan and Gregg Victor Gabison

Graduate School of Computer Studies,
University of San Jose – Recoletos, Cebu City, Philippines

ABSTRACT

Categorizing books and other archaic paper sources to a course reference or syllabus is a challenge in library science. The traditional way of categorization is manually done by professionals and the process of seeking and retrieving information can be frustrating. It needs intellectual tasks and conceptual analysis of a human effort to recognize similarities of items in determining the subject to the correct category. Unlike the traditional categorization process, the author implemented the concept of automatic document categorization for libraries using text mining. The project involves the creation of a web app and mobile app. This can be accomplished through the use of a supervised machine learning classification model using the Support Vector Machine algorithm that can predict the given category of data from the book or other archaic paper sources to the course syllabus they belong to.

KEYWORDS

Text Mining, Document Categorization, Classification algorithm, Support Vector Machine, Library.

1. INTRODUCTION

Categorizing and associating books and other archaic paper sources with a course reference or syllabus needs a lot of parameters to be considered: what is it all about? With what course reference or syllabus is associated? Early studies of document categorization in libraries need intellectual tasks and conceptual analysis of an item. In this, there are a lot of parameters to be considered in predicting a document.

Books that are related to the syllabus will serve as the basis of the Librarians in categorization tasks. Librarians may classify books that they may think will fit the course reference or syllabus based on titles, contents, and chapter headings where the human assignment was used. Considering these parameters in categorizing documents needs a serious conceptual analysis of human ability. On the other hand, some parameters are not sufficient to justify the target label of the documents. Some titles and subject headings can be present across categories or an absence from all categories that may lead to confusion and misclassification.

To address such gaps, the study aims to:

- Build a classification model using Support Vector Machine

- Develop a system that can provide automatic categorization of documents to course reference or syllabus using mobile and web technologies
- Generate a report of classified books

The study focused on the table of contents and index pages only since it carries the significant components of the document (topics, chapters, and sections) relevant for text categorization. This can provide a cost-effective solution to human classifiers in classifying the books to what course syllabus they are assigned for easy reference, efficiency, and improved accuracy.

2. RELATED STUDIES

There are several studies regarding document categorization. Text analysis is an emerging field of study. Fields such as Academia, Marketing, and Governance are already leveraging the process of analyzing and extracting information from textual data.

A notable study of Text Classification is the 'Automatic Categorization of Tagalog Documents Using Support Vector Machines' by April Dae C. Bation, Erlyn Q. Manguilimotan, and Aileen Joan O. Vicente. In this study, the authors created an automated machine learning document classifier suitable for Tagalog documents. The document used was a news article from the Tagalog News Portal. These documents were manually categorized and later subjected to preprocessing techniques such as stem and stopword removal. They used a variety of document representations to find out which of the classifiers performed the best. An SVM classifier using a master dataset represented by a TF-IDF value provided an F-score of 91.99% and overall accuracy of 92%. This surpassed all other combinations of document representations and classifiers.

In the study of Soumick Chatterjee, Pramod George Jose, and Debabrata Datta, SVMs have been used in linear kernels that use the OneVsRest strategy for text classification with multithreaded and CUDA-enhanced SVMs. SVMs are trained on different datasets collected from different sources. Certain words that were less common about 56 years ago may now be widely used due to new trends. Similarly, discoveries can lead to the coining of new words. This technique can also be applied to text blogs that can be crawled and analyzed. This technique should theoretically be able to classify blogs, tweets, or other documents with high accuracy. The pre-processing phase (cleanup, stemming, lemming, etc.) is the longest in any text classification process. Therefore, the author took a multithreaded approach to speed up the process. SVMs have been used in linear kernels that use the OneVs Rest strategy for text classification with multithreaded and CUDA-enhanced SVMs.

Bernhard Scholkopf (the "support vector machine" for IEEE intelligent systems and their applications), the implementation overview, SVM's specific advantages over other learning algorithms to be theoretically analyzed using the concepts of computational learning theory when applied to actual or real problems. Examples of these real applications are provided by Sue Dumais, who explains the text classification problem above and offers the best results ever in the Reuters collection, and Edgar Osuna, who shows powerful results when applied to facial recognition. The fourth author, John Platt, provides practical guides and new techniques for efficiently implementing algorithms.

Lipo Wang, states that SVMs have a solid mathematical structure in statistical learning theory and has a good performance in many real-world applications such as bioinformatics, text mining, facial recognition, and image processing. We have established SVM as one of the most advanced tools. Machine learning and data mining, and other soft computing technologies (eg B. Neural networks and fuzzy systems)

Roy T. Fielding in his doctoral dissertation, typical Representational State Transfer (REST) is described as an important architectural principle of the World Wide Web, which has received a lot of attention. The majority use REST as an application programming interface (API) for communicating between mobile and the web, as an approach to web service development commonly known as RESTful web services, and as an alternative to other distributed computing specifications.

It is in the light of these theories and related literature that inspires the researcher to develop a 'BookShelf: A Document Categorization for Library using Text Mining' to ease the work of librarians in categorizing or classifying books to the course syllabus they are assigned for easy reference. The above-related studies gave the researcher the idea that the proposed project could be possibly done using the Support Vector Machine (SVM) Algorithm, RESTful Web Services, and different tools and innovations.

3. Methodology

The systematic software development process of 'BookShelf: A Document Categorization for Library using Text Mining' is described in this section.

Below is the conceptual diagram of BookShelf to explain the process.

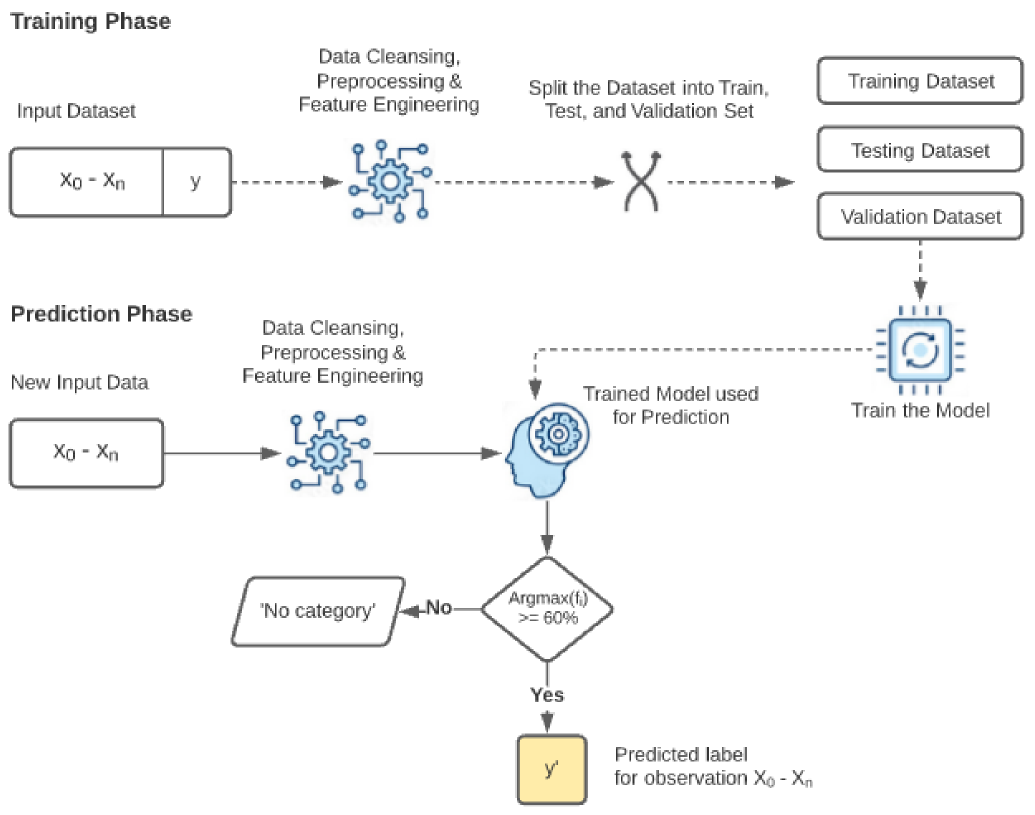

Training Phase
Input Dataset
$X_0 - X_n$
y
Data Cleansing, Preprocessing & Feature Engineering
Split the Dataset into Train, Test, and Validation Set
Training Dataset
Testing Dataset
Validation Dataset
Train the Model
Prediction Phase
New Input Data
$X_0 - X_n$
Data Cleansing, Preprocessing & Feature Engineering
Trained Model used for Prediction
Argmax(f_i) >= 60%
No
'No category'
Yes
y'
Predicted label for observation $X_0 - X_n$

Figure 1. Conceptual diagram for *BookShelf – A document Categorization for Library using Text Mining*

The conceptual diagram shows the detailed process starting from the input, processes, and output of the system. The first step of this process is to create the model that will serve as the classifier in predicting the input data. Input Dataset records, which are text extracted from the table of contents and index pages of specific course references or related books in the curriculum are stored in a database that undergoes data cleansing, pre-processing, and feature engineering techniques. The model is created using the OnevsRest approach of the SVM algorithm.

The *New Input Data* is the texts extracted from the book's table of contents using OCR from the mobile application. The input data will undergo data cleansing, text preprocessing, and feature engineering phase as well. The input data is predicted by the classifier or model created. The predicted document can be assigned to a class or will be predicted as no category. This can be achieved by using a likelihood threshold such as 60%. Maximum values above the threshold are assigned to a class or predicted to be "no category".

3.1. Training phase

This part of the section explains how the model is created using the SVM (Support Vector Machine) algorithm and the components of training, evaluating, and building a model on a supervised approach and using it to classify the input documents. The process being used is through gathering content or text (table of contents and index pages) from the books which is used in training and building the model. The data that was gathered for the creation of the dataset came from the book's table of contents and index pages relevant to the specific course reference or syllabus. We considered five (5) relevant books per course reference or syllabus of different authors for each category in training the model in this project. The texts from the books table of contents and index pages are extracted or pulled out using OCR technology from a mobile app and saved to the database.

During the creation of the model and the prediction of the input data, pre-processing of text and feature engineering techniques were applied as discussed in the next subsections.

3.1.1. Text Processing and Feature Engineering

This includes text cleaning, tokenization, removal of stop words, lemmatization, N-gram, and TF-IDF (term frequency-inverse document frequency). Below are pre-processing and feature engineering techniques that were applied during the creation of the model and the prediction of input data.

3.1.2. Text Cleaning

This phase applies the removal of unwanted characters, some special characters, numbers, and punctuations present in the dataset, and input data using the regular expression. Converting all text to lowercase is also applied in this process.

3.1.3. Tokenization

In the pre-processing phase, it is the next step wherein the text or word sequences are converted into tokens. This process removes the whitespaces and other special characters or punctuations such as {([\t{}():;.])} from the text or document. Each word sequence is converted into tokens. You need to distinguish between the words that make up the string of characters. This is important because you can easily interpret the meaning of the text by analyzing the words that are present in the text. This is also very important in the next step of the process which is the lemmatization.

3.1.4. Lemmatization

The next step is the lemmatization process. This process groups together the inflected forms of a word so that they can be analyzed as a single element (e.g. ‘pointers’ to ‘pointer’, ‘lists’ to ‘list’, etc.). It takes the morphological analysis of the words. Lemmatization can be done after the tokenization process.

3.1.5. Stop Words Removal

The next step is the stop words removal. This process removes auxiliary verbs and prepositions. These words are considered to be unnecessary words that are less informative and are filtered out before processing the dataset and input data. Stop words include *to, at, as, an, a what, where, on, that,* and many others. Words such as (Bibliography, Chapter, Introduction, Contents, Exam, Quiz, Preface, and many others) were also removed since they carry less or no meaning to a document by adding these words to the existing stop word library.

3.1.6. N-gram

The next step is the N-gram process. The basic point of the N-gram is to understand the structure of the language from a statistical point of view, like what word is likely to follow a particular one. It helps capture important differences between the two documents. In this project, we set the parameters of *N-gram* to (1,2) unigram and bigram. It helps determine which n-grams can be chunked into single entities that have a significant impact on mining (such as a "binary tree" chunked as a single word, a "bubble sort" chunked as a single word, etc.). If the n-gram is too short, you may not be able to capture the significant differences. On the other hand, if it is too long, you may not get "general knowledge" and just stick to each case.

3.1.7. TF-IDF (Term Frequency – Inverse Document Frequency)

Before we can start training the dataset, it should be represented in the form of a vector. This is with the help of *TF-IDF* for *Term Frequency-Inverse Document Frequency* as the feature engineering algorithm. In this project, *TfidfVectorizer* has been used to transform text to feature vectors that can be used as input to the estimator.

Words which are present in all the documents for those words will be minimum and hence the total *TF-IDF* values will also come low for those documents. That is how *TF-IDF* tries to balance the unique words in the document that should highlight and come up. Those highlighted words are emphasized and given more weights during the prediction against the input data through which the algorithm/model learns from the pattern by looking at the features. Both training data and input data will undergo this process.

Here is a table with the list of some categories and the highlighted features or keywords from the dataset.

Table 1. *TF-IDF* result of some categories

Data Structures and Algorithms	**Networking**	**Systems Analysis and Design**	**Probability and Statistics**
TF-IDF	TF-IDF	TF-IDF	TF-IDF
queue	basic architecture	language leader	population
0.125196	0.102553	0.152392	0.104746
abstract	backbone	concept language	normal curve
0.118645	0.097059	0.148374	0.099014
abstract data	performance improving	leader	mean
0.116436		0.136394	0.094354
priority queue		flow diagram	test
0.112221	0.086984	0.101193	0.093292
tree	improving	use case	curve
0.106971	0.085592	0.091861	0.091680
... ...	reducing network	... ...	... ...
formal system	0.083072	frame margin	formula random
0.000000	... ...	0.000000	
formal standard	Fragment	frame friendly	0.000000
0.000000	0.000000	0.000000	formula probability
formal argument	fractional frame	frame free	
0.000000	0.000000	0.000000	0.000000
formal approach	Fractional	frame frame	formula lazy
0.000000	0.000000	0.000000	0.000000
zoned decimal	fra digital	zoned decimal	formula grouped
0.000000	0.000000	0.000000	
	zoned decimal		0.000000
25868 rows × 1 column	0.000000	25868 rows × 1 column	zoned decimal
			0.000000
	25868 rows × 1 column		
…		…	25868 rows × 1 column
	…		…

The table above shows the list of *TF-IDF* results of a few highlighted keywords for some categories.

3.1.8. SVM

All of the above steps are pre-processing steps that will lead us to use the dataset results for the *SVM* to process. In this project, we used *SVM* as the classifier with a linear kernel which means the data is linearly separable by a straight line. SVM works well even with a small size dataset which is suitable for this project. The *probability estimate* feature of SVM also is used in deciding to assign a class label. This can be achieved by using a probability threshold such as 60%, where the maximum argument value result is equal to or greater than the threshold is mapped to one class or otherwise predicted as 'no category' since some other archaic documents from the library have no relationship to the existing course reference. SVM outperforms the other classifiers in terms of prediction accuracy. This is explained in the next subsection.

3.1.9. Training, Evaluation, and Model Creation

When the dataset is ready, it will be evaluated and the process begins by splitting the dataset using the *train_test_split* of *sklearn* into training and test sets. 67% went to the training set while 33% went to the testing set. We train the model on the training set and evaluate the model on the test set until the best result is obtained and save the model.

Classifier Performance

Table 2. SVM OneVsRestClassifier Performance

Category	Precision	Recall	F1-Score
Fundamentals of Database Systems	0.75	1.00	0.86
Data Communications and Networking	1.00	1.00	1.00
Data Structures and Algorithms	1.00	0.75	0.86
Systems Analysis and Design	1.00	0.60	0.75
Probability and Statistics	0.33	1.00	0.50

Accuracy:	0.82		
Macro avg:	0.82	0.87	0.79
Weighted avg:	0.92	0.82	0.84

In the above information, the performance of the model has met 0.823529411765 accuracy. It is expected for this case since we only have a small dataset during the training. Another factor for this is the fine-tuning of SVM parameters such as setting the random_state value during the *train_test_split* process. Linear SVM performs well in text classification tasks.

The classifier performance yields a good result with a high F1-Score after testing it with our dataset with a small number of samples per category. Here is the confusion matrix that depicts the accuracy of each category during prediction.

Confusion Matrix

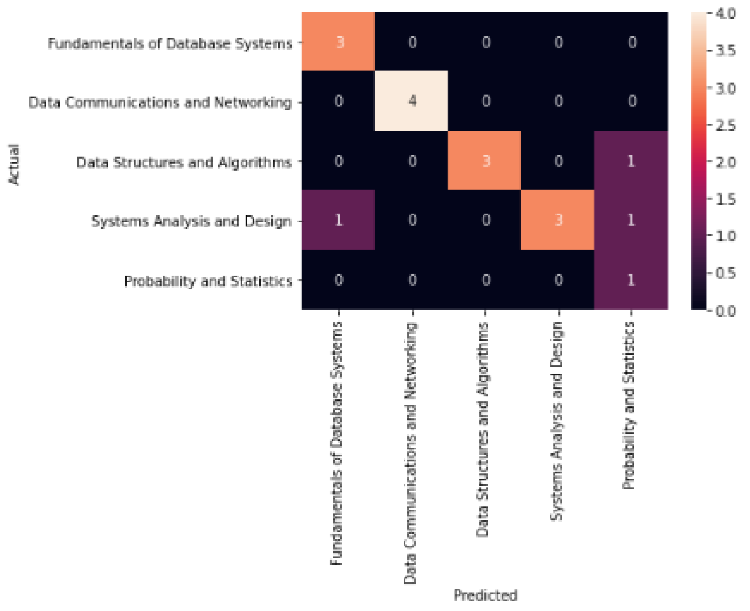


Figure 2. Confusion Matrix result for BookShelf

The above figure depicts some misclassification during the training and testing process and it is only minimal. In some cases, it is expected since both categories might have shared most of the common features.

3.1.10. Cross-Validation

Here is the accuracy result of SVM using the OneVsRestClassifier compared to other classifiers after evaluating them using the cross-validation test with our dataset. Here we achieved a *mean accuracy score* of 0.86 after evaluating the *cross_val_score* with 20% of the dataset in each fold, which is a decent classifier.

Table 3. Model performance using 5-fold cross-validation

Model name	**Accuracy**
MultinomiallNB	0.80
SVM OneVsRestClassifier	0.86
RandomForestClassifier	0.72

As you can see in the above data, the OneVsRestClassifier of SVM has achieved the highest score and outperforms other classifiers during the evaluation using the 5-fold cross-validation method.

Cross-validation summary

Table 4. 5-fold cross-validation summary

Model name	**Fold_idx**	**accuracy**
SVM OneVsRestClassifier	0	1.0
SVM OneVsRestClassifier	1	1.0
SVM OneVsRestClassifier	2	1.0
SVM OneVsRestClassifier	3	0.8
SVM OneVsRestClassifier	4	0.5
RandomForestClassifier	0	1.0
RandomForestClassifier	1	1.0
RandomForestClassifier	2	1.0
RandomForestClassifier	3	0.4
RandomForestClassifier	4	0.2
MultinomialNB	0	1.0
MultinomialNB	1	1.0
MultinomialNB	2	1.0
MultinomialNB	3	0.7
MultinomialNB	4	0.3

The validation technique using the 5-fold cross-validation was used to validate the performance of the SVM OneVsRestClassifier. Training and testing were repeated 5 times on stratified folds for the whole dataset.

3.2. Prediction Phase

The figure below describes the process of predicting input data.

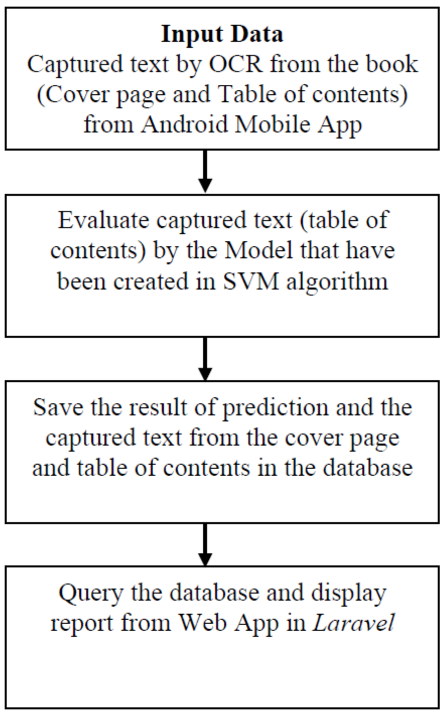

Input Data
Captured text by OCR from the book (Cover page and Table of contents) from Android Mobile App
Evaluate captured text (table of contents) by the Model that have been created in SVM algorithm
Save the result of prediction and the captured text from the cover page and table of contents in the database
Query the database and display report from Web App in *Laravel*

Figure 3. Shows the process of evaluating test files and generating the report

The input data is predicted by the model that has been created using the SVM algorithm. Each document can be assigned to a class or no category. This can be achieved by using a probability threshold, such as 60%, where the highest value equal to or greater than the threshold is mapped to one class or otherwise predicted as ‘no category’. The result of the prediction and the text extracted from the cover page, table of contents, or index pages are saved in the database. Data from the database are queried and displayed as the summary or report of classified books in the Web app in *Laravel*.

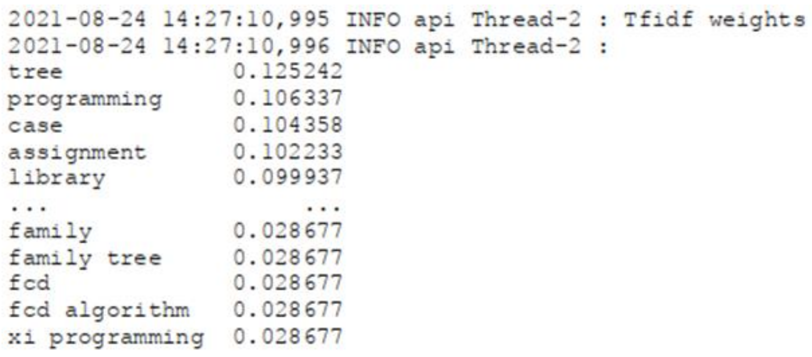
Here is the prediction result of the input data with probability estimates of each class.

```
2021-08-24 14:27:11,014 INFO api Thread-2 : Prediction result
2021-08-24 14:27:11,014 INFO api Thread-2 : 4
2021-08-24 14:27:11,015 INFO api Thread-2 : Probability estimate
2021-08-24 14:27:11,015 INFO api Thread-2 : 0.89
2021-08-24 14:27:11,015 INFO api Thread-2 : Probability estimate of each
class
2021-08-24 14:27:11,015 INFO api Thread-2 : [0.00685022 0.01537133
0.89467403 0.03967454 0.04342988]
2021-08-24 14:27:11,076 INFO werkzeug Thread-2 : 192.168.43.1 - -
[24/Aug/2021 14:27:11] "#[37mPOST /endpointSVM HTTP/1.1#[0m" 200 -
```

Figure 4. Prediction result of input data

In the above figure, the input data or input document achieved a probability estimate score of 0.89 and is correctly assigned to a category which is Data Structures and Algorithms.

Here is the TF-IDF result of the unseen test input data coming from the *Data Structures and Algorithms* book.



```
2021-08-24 14:27:10,995 INFO api Thread-2 : Tfidf weights
2021-08-24 14:27:10,996 INFO api Thread-2 :
tree             0.125242
programming      0.106337
case             0.104358
assignment       0.102233
library          0.099937
...                   ...
family           0.028677
family tree      0.028677
fcd              0.028677
fcd algorithm    0.028677
xi programming   0.028677
```

Figure 5. TF-IDF result of test input data

Figure 6 is the result of the TF-IDF of the test input data. Since most of the highlighted keywords from the trained dataset are also present in the input data, then the input data is assigned with the label of that category by the classifier. Notice the probability estimate of other classes which is very low for other categories. This is where the probability estimate feature of SVM comes effective in making the decision and classification.

4. Conclusion and Recommendations

4.1. Conclusion

The categorization of books or other archaic documents to a specific target or class is affected by many factors. Such include the collection of relevant datasets or corpus, the feature engineering techniques, and applying regular expressions such as removing unnecessary words.

The selection of the algorithm classifier type suitable for the requirements of the project has a contribution also in the categorization task. The use of Support Vector Machine as the algorithm is very well suited for this project and related projects that do not need a large corpus or dataset. The observation and experimental results show that SVM achieved good performance on text categorization tasks compared to other existing classifiers significantly with the given dataset. The SVM classifier yielded an accuracy of 0.86% during the 5-fold cross-validation technique that outperforms other classifiers.

This mobile and web-based application software are intended to solve the problems of librarians in categorizing books and other archaic documents to the course reference or syllabus. It provides automatic categorization for efficient, effective, and improved accuracy.

4.2. Recommendations

The researcher highly recommends the use of Support Vector Machine (SVM) for the requirement of this project and other document categorization for multi-class classification tasks. A keen and careful approach to removing unnecessary words from the extracted text in the table of contents and index pages may improve the performance of the model. Words or other languages aside from English may be considered as an account as well.

Lastly, future research should also be able to build a classifier that can classify or categorize the books or other archaic documents with more than one category (multi-label) instead of multi-class classification. Future research should also build a model using incremental online training that can incorporate updates from the new unseen data to the existing model's precision or learning accuracy if the dataset is too large to fit in the memory.

REFERENCES

[1] Victor Diogho Heuer De Carvalho, Ana Paula Cabral Seixas Costa (2022). Exploring Text Mining and Analytics for Applications In Public Security: An In-Depth Dive Into A Systematic Literature Review.Exploring Text Mining and Analytics for Applications in Public Security: An in-depth dive into a systematic literature review | Request PDF (researchgate.net)

[2] Billie S. Anderson (2021). Using Text Mining to glean insights from COVID-19 literature. Using text mining to glean insights from COVID-19 literature - Billie S Anderson, 2021 (sagepub.com)

[3] Rafael Ferreira-Mello, Máverick André, Anderson Pinheiro, Evandro Costa, Cristobal Romer (2019). Text Mining in education. Text mining in education - Ferreira-Mello - 2019 - WIREs Data Mining and Knowledge Discovery - Wiley Online Library

[4] April Dae Bation, Aileen Joan Vicente, Erlyn Manguilimotan (2017). Automatic Categorization of Tagalog Documents Using Support Vector Machines. https://aclanthology.org/Y17-1046.pdf

[5] Sunita Gopal, S. Raghav (2017). Automatic document retrieval using SVM machine learning. https://ieeexplore.ieee.org/document/8358501/authors#authors

[6] Vladimer B. Kobayashi, Stefan T. Mol, Hannah A. Berkers (2017). Text Mining in Organizational Research.https://journals.sagepub.com/doi/full/10.1177/1094428117722619

[7] Graciela H. Gonzalez, Tasnia Tahsin, Britton C. Goodale, Anna C. Greene, Casey S. Greene (2016). Recent Advances and Emerging Applications in Text and Data Mining for Biomedical Discovery.https://doi.org/10.1093/bib/bbv087

[8] Y. Zhang, M. Chen and L. Liu (2015). A review on text mining. https://ieeexplore.ieee.org/document/7339149

[9] Pooja S. Ikka (2015). Data Mining of Social Networks using Clustering Based-SVM. [PDF] DATA MINING OF SOCIAL NETWORKS USING CLUSTERING BASED-SVM | Semantic Scholar

[10] Mrs. Manisha Pravin Mali, Mohammad Atique (2014). Applications of Text Classification using Text Mining. https://www.researchgate.net/publication/287531225_Applications_of_Text_Classification_using_Text_Mining

[11] Mladeni D., Brank J., Grobelnik M. (2011) Document Classification. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_230

[12] [Neelima Guduru (2006). Text Mining with Support Vector Machines and Non-Negative Matrix Factorization Algorithms.Microsoft Word - Thesis-Neelima-Guduru.doc (uri.edu)

[13] M. Ikonomakis, S. Kotsiantis, V. Tampakas (2005). Text Classification Using Machine Learning techniques.(PDF) Text Classification Using Machine Learning Techniques (researchgate.net)

[14] A. Basu, C. Watters, and M. Shepherd (2002). Support Vector Machines for Text Categorization. Support vector machines for text categorization - System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on (dal.ca)

[15] Anuradha K. Bodile , Manali Kshirsagar. Text Mining in Radiology Reports by SVM Classifier. CiteSeerX — Text Mining in Radiology Reports by SVM Classifier (psu.edu)

AUTHOR

Petalver, Carlo D, graduate of Bachelor of Science in Information Computer Science and currently pursuing my Master in Information Technology at University of San Jose-Recoletos. I have an experience with computer languages, including PHP, Laravel, MySQL, Android, Java, and Python. I have had experience also with small projects both in industry and in the academe and at the same time, I have been teaching for several years in one of the private schools in Cebu City in the field of Information Technology.

STOCK MARKET PREDICTION USING NATURAL LANGUAGE PROCESSING -A SURVEY

Om mane and Sarvanakumar kandasamy

Department of Computer Science and Engineering,
Vellore Institute of Technology, Vellore, India

ABSTRACT

The stock market is a network which provides a platform for almost all major economic transactions. While investing in the stock market is a good idea, investing in individual stocks may not be, especially for the casual investor. Smart stock-picking requires in-depth research and plenty of dedication. Predicting this stock value offers enormous arbitrage profit opportunities. This attractiveness of finding a solution has prompted researchers to find a way past problems like volatility, seasonality, and dependence on time. This paper surveys recent literature in the domain of natural language processing and machine learning techniques used to predict stock market movements. The main contributions of this paper include the sophisticated categorizations of many recent articles and the illustration of the recent trends of research in stock market prediction and its related areas.

KEYWORDS

Stock market Prediction, Sentiment Analysis, Opinion Mining, Natural Language Processing, Deep Learning.

1. INTRODUCTION

Stock movement prediction is a central task in computational and quantitative finance. With recent advances in deep learning and natural language processing technology, event-driven stock prediction has received increasing research attention [12, 13]. With the increasing influence of the stock market on economic trends, forecasting the trend of stocks has become a hot topic in research. Many researchers have conducted scientific and meticulous research on the stock market, trying to formulate rules for the operation of the stock market. However, the results of the research have found that the changes in the stock market seem to be unrelated [14, 15] The efficient market hypothesis theory proposed by Eugene Fame is a more authoritative explanation in the current financial circles to study the law of stock market changes. In this theory, the stock price is mainly affected by future information, namely news, rather than being driven by current or past prices [16-18]. For example, Lin et al. proposed an end-to-end hybrid neural network, which uses convolutional neural networks (CNNs) to extract data features and uses long- and short-term memory recurrent neural networks to capture the long-term dependence in the historical trend sequence of the time series to learn. Contextual features predict the trend of stock market prices [19]. Hu et al. designed a hybrid attention network (HAN) to predict stock trends based on related news sequences [20]. Li et al. proposed a multitask recurrent neural network (RNN) and a high-order Markov random domain to predict the movement direction of stock prices Hindawi Security and Communication [21]. The most popular technique used in financial time series analysis to detect the trend and seasonality is Autoregressive integrated moving average. ARIMA. The current advancement in technology has improved modern techniques in forecasting financial time series data. Deep LSTM, Shallow LSTM, 1-CNN and machine learning models were used to predict stock market data. An attention LSTM model is also used in prediction of financial time series data. ARIMA model is used in prediction of the closing prices of time series data. Va Yassine, Khadija and Faddoul in formulated classification model designed based on LSTM network to predict the probability of investing or not. For each of the selected papers the extracted data and information about (a) research characteristics, as authors, year of publication, languages covered, methodology, corpus characteristics; (b) sentiment level and categorization (binary, ternary, or fine-grained, e.g., rates $0 - 5$); (c) deep learning architectures and techniques, and (d) results and effectiveness of the proposal against baselines or state-of-the-art models.

This survey can be useful for newcomer researchers in this field as it covers the most famous Sentiment Analysis (SA) techniques and applications in one research paper. This survey uniquely gives a refined categorization to the various SA techniques which is not found in other surveys. Existing work has investigated news representation using deep learning [4, 8], neural fuzzy networks [2], heterogeneous graphs [7], long short-term memory and the random forest framework [10].

2. Related Works

2.1. Noisy Recurrent State Transition

Heyan Huang, Xiao Liu, discussed the prediction of the stock market through news events with the help of advances in deep learning techniques and machine learning technology. They proposed a novel future event prediction module to factor in likely next events according to natural events consequences. The future event module was trained over "future" data over historical events. They were able to visualize past events over a large time window, and their model is also more explainable. To their knowledge, they are the first to explicitly model both events and noise over a fundamental stock value state for news-driven stock movement prediction.

2.2. Deep Learning

Jiake Li, tried to establish a stock index forecast and network security model based on time series and deep learning. Based on the time series model, the author proposed to use CNN to extract in depth emotional information to replace the basic emotional features of the emotional extraction level. At the data source level, other information sources, such as basic features, are introduced to further improve the predictive performance of the model. The prediction model constructed in this paper is also based on the model in the specific direction of deep learning in data mining. Agüero, M.M. Salas, proposed the method of Multilingual Sentiment Analysis (MSA) which is an attempt to address the sentiment analysis issue through several strategies. They improved over previous reviews with wider coverage from 2017 to 2020 as well as a study focused on the underlying ideas and commonalities behind the different solutions to achieve multilingual sentiment analysis.

2.3. Constrained Learning

Chen, X., Rajan, D., & Quek, H.C, proposed an efficient and interpretable neuro-fuzzy system for stock price prediction using multiple technical indicators with focus on interpretability–accuracy trade-off. Neuro-fuzzy systems offer to represent complex solutions in a natural language like representation in the form of interpretable fuzzy rules which are easier to comprehend by users.

These systems are well suited to uncertain and complex real-world problems like stock market environments as these systems can deal with incomplete and uncertain data conditions and do not require large numbers of observations as other AI-based prediction models. Various studies have integrated neuro-fuzzy systems with other techniques to improve the prediction accuracy. Atsalakis et al. (2011) introduced a novel stock trading system based on neuro-fuzzy modeling method. The trading system augmented the neuro-fuzzy system with the concepts from Elliot Wave Theory to enhance forecasting accuracy. The study demonstrated high performance of this system in predicting the trends in stock prices. Tan et al. (2011) presented an ANFIS-based model for stock trading. Authors supplemented the model using reinforcement learning (RL).

2.4. Graph based Prediction

Junran Wua, Ke Xua discussed a method to develop a novel framework based on the structural information embedded in price graphs by conducting numerous experiments on real-world market data. In this study, they propose a novel framework to address both issues. Specifically, in terms of transforming time series into complex networks, they convert market price series into graphs. Then, structural information, referring to associations among temporal points and the node weights, is extracted from the mapped graphs to resolve the problems regarding long-term dependencies and the chaotic property. They take graph embeddings to represent the associations among temporal points as the prediction model inputs. Node weights are used as a priori knowledge to enhance the learning of temporal attention. The effectiveness of their proposed framework is validated using real-world stock data, and their approach obtains the best performance among several state-of-the-art benchmarks. Feature based methods only use simple lexical features such as bags-of-words, noun phrases and named entities (Kogan et al., 2009; Schumaker and Chen, 2009) extracted from financial text to predict the movement of stock market. To better predict the movement of stock market, Ding et al. (2015, 2019) propose to utilize the event level information extracted from titles or abstracts. Moreover, to capture contextual information within document, many document representation learning based methods (Augenstein et al., 2016; Chang et al., 2016; Tang et al., 2015; Yang et al., 2016) are proposed to learn a distributed representation of the whole document or abstract, and build prediction model based on the representation vector, so that information from overall document can be utilized for making predictions.

2.5. Liquidity Prediction for Learning Models

As an evaluative consideration in investment decisions, stock liquidity is of critical importance for all stakeholders in the market. It also has implications for the stock market’s growth. Liquidity sanctions investors and issuers to meet their quotas regarding investment, financing or hedging, reducing investment costs and the cost of capital experimental results, it can be concluded that the LSTM model allows for prediction characterized by lowest value of mean square error (MSE). The aim of [6] is to develop the machine learning models for liquidity prediction. The subject of research is the Vietnamese stock market, focusing on the recent years - from 2011 to 2019. Vietnamese stock market differs from developed markets and emerging markets. It is characterized by a limited number of transactions, which are also relatively small. The Multilayer Perceptron, Long-Short Term Memory and Linear Regression models have been developed. Data was prepared for further analysis before data was preliminarily analyzed and relationships were found that enable illiquidity prediction. After preparing and selecting data, a deep learning model was developed. The study was conducted on the Ho Chi Minh City Stock Exchange in Vietnam. A sample of daily data of 220 companies representing different sectors was used for analysis. The main disadvantage of the proposed approach is the performing prediction only for one day.

3. ARCHITECTURE

This section gives a detailed explanation about the methods used in analyzing and evaluating the trend of S&P500 stocks, where a System Architecture of stock market price prediction is designed which explains its associated process flow. Also, there has been a great interest in novelty architectures for Sentiment Analysis (SA). One approach that has received substantial attention when working at the basic document level, is the design of hierarchical models which learn a representation for sentences from its words, on top of this level, another model can learn representations for documents. Other substitutes such as Convolutional Neural Networks (CNN) or Long Short-Term Memory (LSTM) can be used at each level. Works in [25,26] and [27] are some examples of this approach.

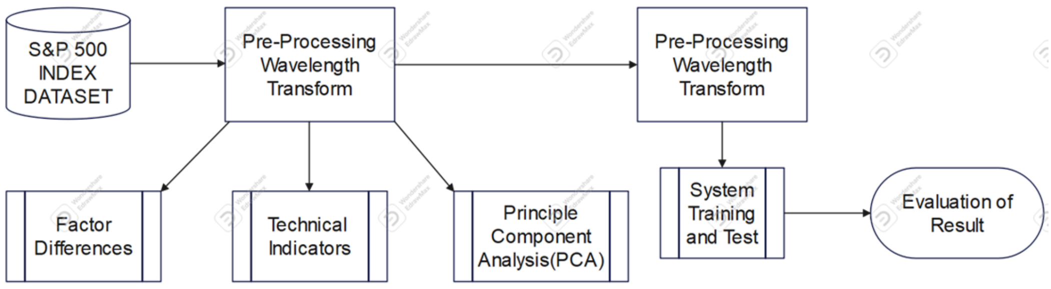

S&P 500 INDEX DATASET
Pre-Processing Wavelength Transform
Pre-Processing Wavelength Transform
Factor Differences
Technical Indicators
Principle Component Analysis(PCA)
System Training and Test
Evaluation of Result

Figure 1. System Architecture [3]

In [1] the authors explicitly model both noise and events over a recurrent stock value state, which is modelled using LSTM. For each day of trading, they consider the news events happened in that day as well as the past news events using neural attention [28]. Assessing the impacts of insider trading, they also pre-suppose future news in the training method and procedure. To model the high stochasticity of stock market, they illustrate a supplemented noise using a neural module. Their model is named attention-based noisy recurrent states transition (ANRES) as can be seen from figure 2.

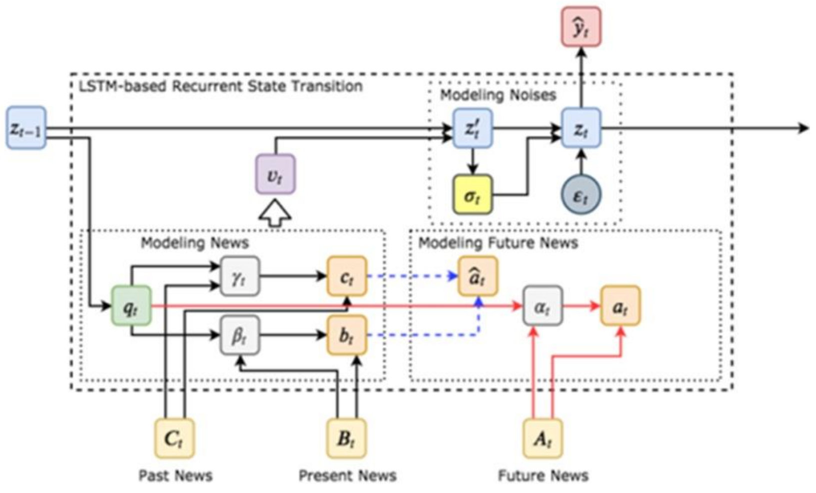


Figure 2. The ANRES model framework for trading day t in a trading sequence. The black solid elbows are used both in the training and the evaluating procedures. The red solid elbows are only used in the training procedure, while the blue dotted elbows in the evaluating procedure [1]

3.1. Time Series Model

The object of the stock model based on time series is the historical data of stocks. The core step is to divide the historical data of stocks to facilitate the subsequent stock market forecasts. In this model, the first and most important step is to collect and process time series data. When predicting a time series, it is mainly by observing the trend changes of the time series first and predicting future time series changes by learning the law of past changes. Time series data often have a large amount of data and are difficult to process directly. This requires dividing it and dividing the time series by finding the key trend points. Through this division method, the originally complex data can be compressed while also removing some noise in the stock sequence. Some points that are not helpful for prediction, so that the retained information is more effective for the model to learn the changes in the time series data, and the time series rules can be found more clearly [4].

3.2. Deep Learning Model

First, for traditional classifiers (such as SVM and KNN) to deal with the general problem of time series data classification, with the help of the recurrent neural network to facilitate the modelling of time series data, a depth-based stock prediction model learned, and on the basis of this model, the sentiment analysis results of stock-related data in the social media text are added to construct a trend prediction model that integrates basic emotional features. Among the deep learning technologies that have emerged in recent years, convolutional neural networks are the most widely used. Figure 3 shows the index prediction process based on deep learning [4].

3.3. Heterogeneous Graphs

Given one or several news documents of a corporation, [7]’s goal is to predict the future stock reaction of this corporation based on corresponding financial text. Rather than predicting the specific price reaction range, we formulate the prediction as fluctuation polarity. The prediction will be “Positive” for the rise in stock price or “Negative” for the decline in stock price. The edges in the heterogeneous graphs are undirected.

Table 1. Example of sentences in Financial Text and stock prices of corporations

Sentence	*After three consecutive days rise on stock prices, H & M was pushed to the forefront of public opinion due to the Xinjiang cotton incident.*
Stock Price Reaction	*H&M Negative*

4. METHODOLOGY

4.1. Dataset Description

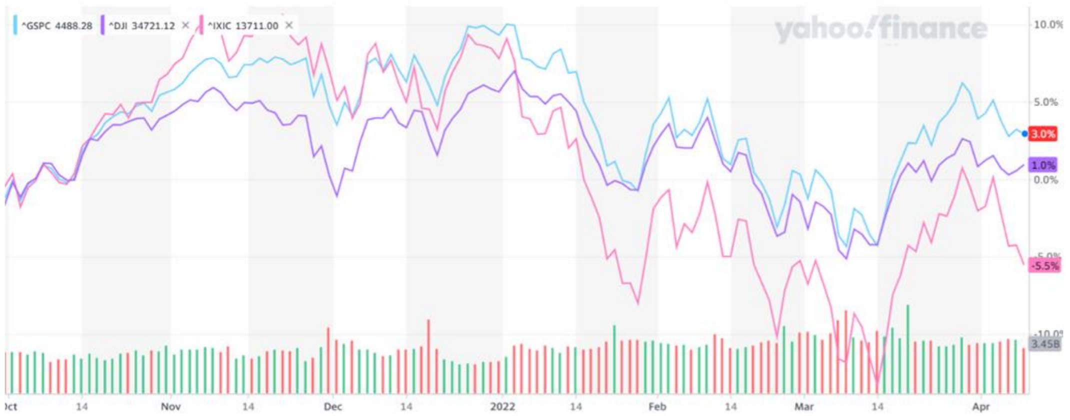
To study the proposed algorithm, Hadi Rezaei, Hamidreza Faaljou extract the historical daily financial time series from January 2010 to September 2019 from the Yahoo Finance Website. The daily data includes the close price of S&P 500, Dow Jones, DAX, and Nikkei225. The authors use the dataset released by Duan et al. in their experiments. The dataset includes more than 100k financial articles from Reuters. They have used an open source openIE tool for information extraction. Training is stopped if the performance doesn't improve for 10 epochs.

Use of heuristic rules to build connections among words, event triples and sentences is done. Employment of multi-grained encoders to encode the nodes within HGk into embedding is done. Following baselines are used in this model: Sentiment, Event Tensor, Event Tensor-CS, TGT-CTX-LSTM, Conditional Encoding, TGT-HN, TGT-HAN, GCN, GAT. The paper [6] covers the sample period from January 2011 to December 2019, which contains 2,242 trading days. The eligible stocks in the sample consist of 378 companies in two stock exchanges, which included 179 stocks on the HOSE and 199 stocks on the HNX. Stock market liquidity is represented by seven liquidity measures.

[6] Covers the sample period from January 2011 to December 2019, which contains 2,242 trading days. The eligible stocks in the sample consist of 378 companies in two stock exchanges, which included 179 stocks on the HOSE and 199 stocks on the HNX. In [2] Daily stock data of three indices, viz. BSE, CNX Nifty and S&P 500, comprising of five fundamental stock quantities, namely maximum price, open price, minimum price, close price, and trading volume, are used to evaluate the proposed system. The dataset of BSE index is from January 30, 2005, to December 30, 2015. For Nifty index, dataset from January3, 2005, to December 24, 2015, is used. In case of S&P 500, daily stock data of 2865 records from February 9, 2005, to June 28, 2016, have been used.

4.2. Dataset Pre-Processing

To work with this dataset, there is a need to pre-process it, wherein visualization followed by illustration is foremost importance. This results in us being able to identify trends, missing values, noise, and outliers. To study the proposed algorithm, data was extracted from the historical daily financial time series from October 2021 to April 2022 from the Yahoo Finance Website. The daily data includes the close price of S&P500, Dow Jones and DAX. To better visualize these indices, The data is described in graph 1.



Graph 1. Price Index for S&P 500, Dow Jones, DAX

Due to the complexity of financial stock exchange environment, stock prices contain a lot of noise that make it very difficult to achieve a good model accuracy when trying to forecast it trend movement. Since is the interest of investors to achieve good forecasting model, then there is a need to reduce the noise in the dataset because stock environment is containing noise from the news articles and other source media information. A wavelength transform, a mathematical function introduced in the pre-processing stage by de noise S&P500 dataset and present it trend and structure of the dataset. The authors transform the dataset using wavelength mathematical transform function:

$$X_\omega(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t)\psi\left(\frac{t-b}{a}\right) dt$$

After transformation we drop the co-efficient with more standard deviation away from the co-efficient and inversely transform the new co-efficient to get the denoise S&P 500 dataset [3].

4.3. Equations

From [1] the ANRES model uses LSTM with peephole connections. The underlying stock value trends are represented as a recurrent state z transited over time, which can reflect the fundamentals of a stock. In each trading day, they consider the impact of corresponding news events and a random noise.

$$z_t' = \overrightarrow{\mathbf{LSTM}}(v_t, z_t - 1)$$

$$z_t = f(z_t')$$

where v_t is the news events impact vector on the trading day t and f is a function in which random noise will be integrated. In deep learning models, loss error is usually used to evaluate the prediction, which refers to the difference between the actual observed value and the predicted value. In order to evaluate the loss error, different tools can be used. The standard metrics in this type of models include root-mean square error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE) [8]. The formula for their calculation is as follows:

$$MSE = \frac{1}{n}\sum_{i}^{n}(y_i - \hat{y}_i)^2$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

In order to select key information and get the stock price fluctuation of a specific corporation. Firstly, we use the corporation to softly select relevant information from sentences:

$$\alpha_i = H_c H_s',$$

$$A_i = \frac{\alpha_i}{\Sigma_{j\epsilon S_k}\alpha_j}$$

$$u = AH_s$$

where $H_c \epsilon \mathbb{R}^d$ is the corporation representation and $H_s' \epsilon \mathbb{R}^{d*r}$ is the embedding of sentences, $A \epsilon \mathbb{R}^r$ is the weight matrix and $U \epsilon \mathbb{R}^d$ is the weight sum of sentences representation.

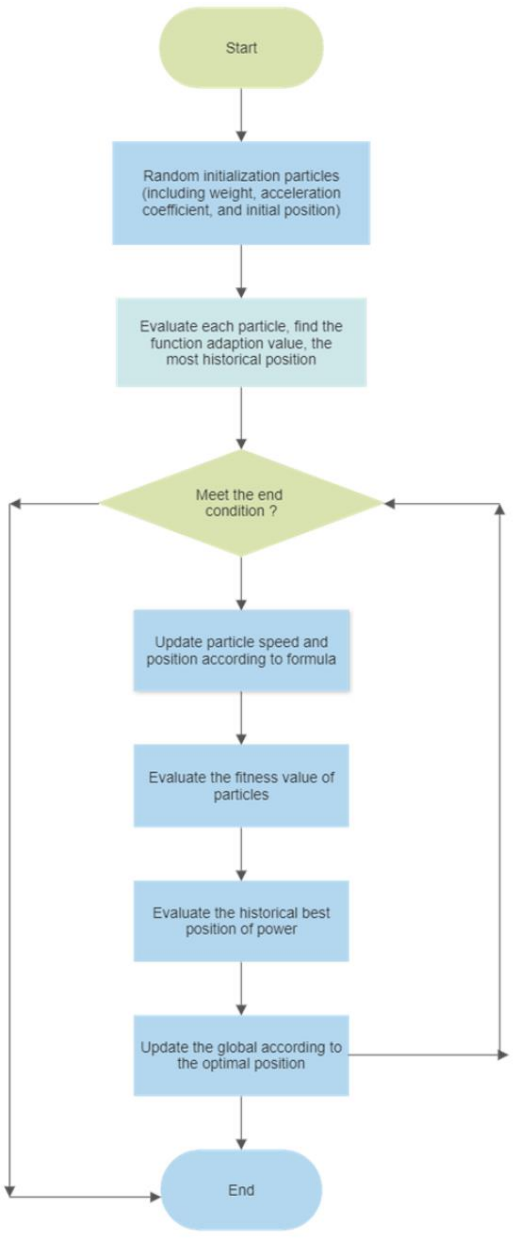

Start
Random initialization particles (including weight, acceleration coefficient, and initial position)
Evaluate each particle, find the function adaption value, the most historical position
Meet the end condition ?
Update particle speed and position according to formula
Evaluate the fitness value of particles
Evaluate the historical best position of power
Update the global according to the optimal position
End

Figure 3. Index prediction process based on deep learning [4]

Therefore, based on U, the authors have used a linear function to predict future stock market fluctuation of this corporation:

$$X_{pred} = Concate[H_c, \mathrm{U}],$$

$$Probabilities = Softmax(W_{pred}\, X_{pred}),$$

$$Prediction = ArgMax(Probabilities),$$

where $X_{pred} \epsilon \mathbb{R}^{2*d}$ is the concatenation of corporation and sentence representation and $W_{pred} \epsilon \mathbb{R}^{d}$ is a trainable parameter. Probabilities ϵ $\mathbb{R}^{2}$ denotes the final distribution of Positive and Negative. Predictions are the final prediction on stock market fluctuation, 0 and 1 denotes Negative and Positive, respectively.

5. Experiments and Evaluation

5.1. Evaluation Metrics

In [2] in order to evaluate the performance of the proposed model, root- mean-squared error (RMSE) and mean average percentage (MAPE) have been used to measure forecasting accuracy. RMSE and MAPE measure the deviation between the actual and the forecasted values. Smaller the value of these metrics larger is the forecasting performance of the model. Also, directional accuracy (DA) has been used which gives the correctness of a system for the predicted trend of a stock price index. For [3] MACD Moving average convergence divergence displays the trend following the characteristic of market prices. CCI Consumer channel index identifies price reversals, price extreme and the strength of rise prices. ROC Rate of change is a momentum indicator that measures the percentage change in prices from one period to the next. For comparison, the baseline approach is chosen based on classical machine learning algorithms. The machine learning algorithm is a traditional method that is widely used but less complex than the model used in this work. Using the same input variables trained and tested their system architecture predictions on S&P 500 new process data. They chose the Logistic regression and Random Forest model. They implemented the two baseline models using Scikit -Learn library. List of supporting algorithms used in this paper are RSI, Momentum, OBV, MA, EMA, ADX. In [7] the authors use the dataset released by Duan et al. (2018) in their experiments. The dataset includes more than 100k financial articles from Reuters2, which also includes stock prices of different corporations. The time interval of the articles starts from October 2006 to December 2015. Only articles that mentioned at least one firm are selected by them, and the dataset is balanced on positive and negative classes. The statistics of this dataset are listed in Table 2. Evaluation Metrics Follow the metrics used in prior works (Chang et al., 2016; Duan et al., 2018), they adopt the area under the precision-recall curve (AUC) as our evaluation metrics. To construct the heterogeneous graph, they use an open source openIE tool (Schmitz et al., 2012) for information extraction. Following the settings of Wang et al. (2020), they initialize the word nodes with 300-dimension GloVe embeddings (Pennington et al., 2014), and the dimension of every node feature is d = 128. The heads of multi-head attention is set as 8. For the training process, the learning rate is set as 5e-4. Moreover, they apply an early stopping mechanism, training will be stopped if the performance on the development set doesn’t improve for 10 epochs.

Table 2. The statistics of [7] dataset.

	Train	**Development**	**Test**
Positive	9,376	477	973
Negative	9,394	486	981
Total	18,770	963	1,954
Docs	27,657	1,419	2,989

For trend forecasting in financial time series with indicator system, the evaluation metric formula is as follows:

$$y_{prob} = \begin{cases} 0, \hat{y}_{\text{model}} < 0.5 \\ 1, \quad otherwise \end{cases}$$

They further move on to evaluate the performance of our prediction model, the selected performance metrics: Recall, Precision and F1 is chosen. They define the chosen metrics as positive class (true positive), negative class (true negative).

$$Precision = \frac{truepositive}{truepositive + falsepositive}$$

$$Recall = \frac{truepositive}{truepositive + falsenegative}$$

For ANRES Model in [1] the authors use the public financial news dataset released by [13], which is crawled from Reuters and Bloomberg over the period from October 2006 to November 2013. They conduct their experiments on predicting the Standard & Poor's 500 stock (S&P 500) index and its selected individual stocks, obtaining indices and prices from Yahoo Finance. Detailed statistics of the training, development and test sets are shown in Table 3.

Table 3. Statistics of ANRES Model Dataset

	Training	**Development**	**Test**
#documents	358,122	96,299	99,030
#samples	1,425	169	191
time span	10/20/2006-06/18/2012	06/19/2012-02/21/2013	02/22/2013-11/21/2013

Following previous work [20, 3, 21], they adopt the standard measure of accuracy and Matthews Correlation Coefficient (MCC) to evaluate S&P 500 index prediction and selected individual stock prediction. MCC is applied because it avoids bias due to data skew. Given the confusion matrix which contains true positive(tp), false positive(fp), true negative(tn) and false negative(fn) values, MCC is calculated as:

$$MCC = \frac{\mathrm{tp} \times \mathrm{tn} - \mathrm{fp} \times \mathrm{fn}}{\sqrt{(\mathrm{tp}+\mathrm{fp})(\mathrm{tp}+\mathrm{fn})(\mathrm{tn}+\mathrm{fp})(\mathrm{tn}+\mathrm{fn})}}$$

In summary, the following four baselines are designed:

• ANRES_Sing_R: randomly initializing the states for each single trading day.
• ANRES_Sing_Z: initializing the states as zeros for each single trading day.
•ANRES_Seq_R: randomly initializing the first states for each trading sequence only.
• ANRES_Seq_Z: initializing the first states as zeros for each trading sequence only.

For Deep Learning Model, the evaluation is done by Simulation Environment and Data. Compared with individual stocks, the volatility of stock indexes is generally smaller because stock indexes are composed of many stocks in different industries and can better reflect the overall economic momentum and overall condition. Index Forecasting Effect Analysis. Using the 1219-day data samples of the Shanghai Composite Index for 5 years from 2015 to 2019, the stock data of 10 consecutive days and 20 days were used as input samples to establish a prediction model for closing price prediction. Using the 731-day data sample of the Shanghai Composite Index for 3 years from 2017 to 2019, 5 consecutive days and 10 days of stock data were used as input samples to establish a prediction model for closing price prediction.

5.2. Baselines

- **Sentiment** (Mayew and Venkatachalam, 2012) is a lexicon-based sentiment analysis method. We use the sentiment lexicons and method released by Loughran and McDonald (2011).
- **Event Tensor** (Ding et al., 2015) is a neural tensor network based event representation learning method. They extract event triples from titles or abstracts. The event triples from the abstract will be averaged for prediction
- **Event Tensor-CS** (Ding et al., 2019) is an external common sense knowledge enhanced event representation learning method. Similar events are trained to be close to each other in vector space by predicting their sentiment polarities and contextual events.
- **TGT-CTX-LSTM** (Chang et al., 2016) uses dependency parse tree to learn a target-related representation of abstract for stock market prediction.
- **LSTM:** a basic LSTM network used to predict the future trends of stock prices based on historical price data [29].
- **DARNN:** a dual-stage attention-based RNN that employs input attention and temporal attention in the encoder and decoder stages, respectively [30].
- **DARNN-SA:** an extension of the DARNN that employs a self-attention layer between the output of the DARNN and the prediction layer [9]
- **MFNN:** a multifilter deep learning model that integrates convolutional and recurrent neurons for feature extraction with respect to financial time series and stock prediction [31].
- **CA-SFCN:** a fully convolutional network (FCN) incorporating cross attention (CA), in which CA is also used as dual-stage attention for the variable and temporal dimensions (with temporal attention first) [32].

6. RESULTS

Given below are the results which were obtained after reviewing several research papers:

Table 4. Result of Deep Learning (LSTM) and Baseline models by indicator system

Metrics	**Deep Learning (LSTM model) (%)**	**Baseline (random forest model)**	**Baseline (logistic regression model)**
Accuracy	0.687%	0.56%	0.551%
Precision	0.538	0.514	0.594
Recall	0.697	0.412	0.236

In heterogeneous graphs the overall results are shown in Table 5, from which we can make the following observations:

Table 5. Results of heterogenous graph method [7]

Methods	**AUC**
Sentiment	0.533
Event Tensor + Title	0.544
Event Tensor + Abstract	0.549
Event Tensor-CS + Title	0.564
Event Tensor-CS + Abstract	0.570
Conditional Encoding	0.603
TGT-CTX-LSTM	0.632
TGT-HN	0.615
TGT-HAN	0.633
GCN	0.621
GAT	0.615
HGM-GIF	**0.638**

1. Semantic knowledge acquisition methods (Event Tensor, Conditional Encoding, TGT-CTX-LSTM, TGT-HN, TGTHAN, HGM-GIF) achieve better results than feature-based method. The reason is that learning semantics within financial text is more effective than learning linguistic features for stock market prediction.

2. Comparison between Event Tensor –CS, Conditional Encoding, TGT-CTX-LSTM, TGT-HN, TGT-HAN, HGM-GIF and Event Tensor shows that external knowledge or sentences within documents can supplement contextual information and offer more useful information for prediction. This confirms that predictions need rich contextual information. With the sequential heterogeneous graph layer, we can capture rich contextual information within financial text.

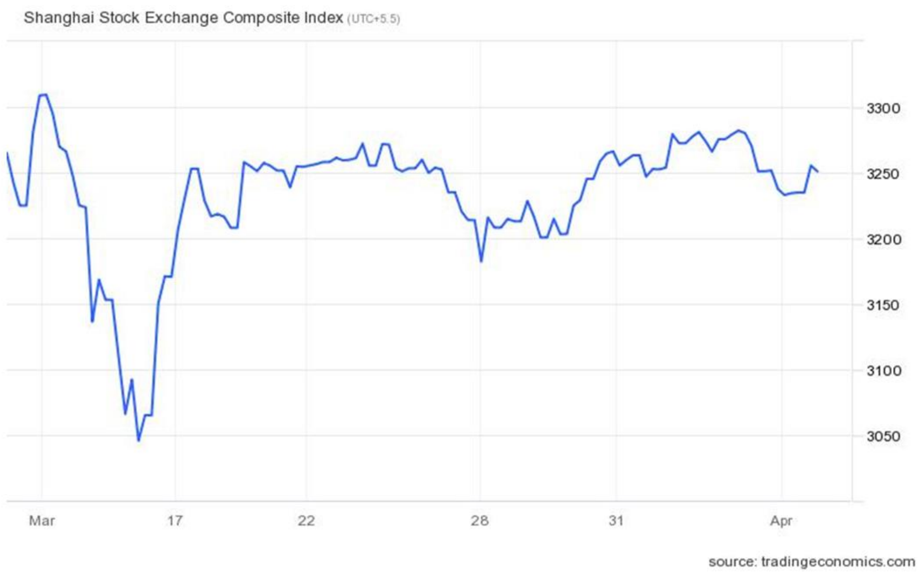


Figure 4. The prediction results of the Shanghai Composite Index at 30-days.

Using the 1219-day data samples of the Shanghai Composite Index for 5 years from 2018 to 2022, the stock data of 7 consecutive days and 30 days were used as input samples to establish a prediction model for closing price prediction. +ese two models are called SHYSD10 and SHYSD20, respectively. Figures 4 and 5 show their prediction results. Figure 4 shows the prediction results of the Shanghai Composite Index at 30-day intervals. Figure 5 shows the forecast results of the Shanghai Composite Index at 70 consecutive days.

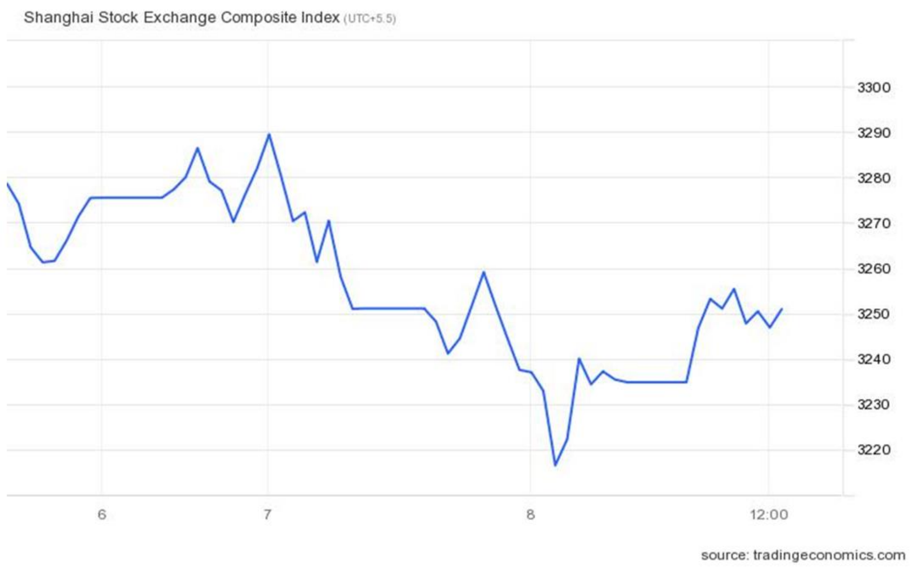
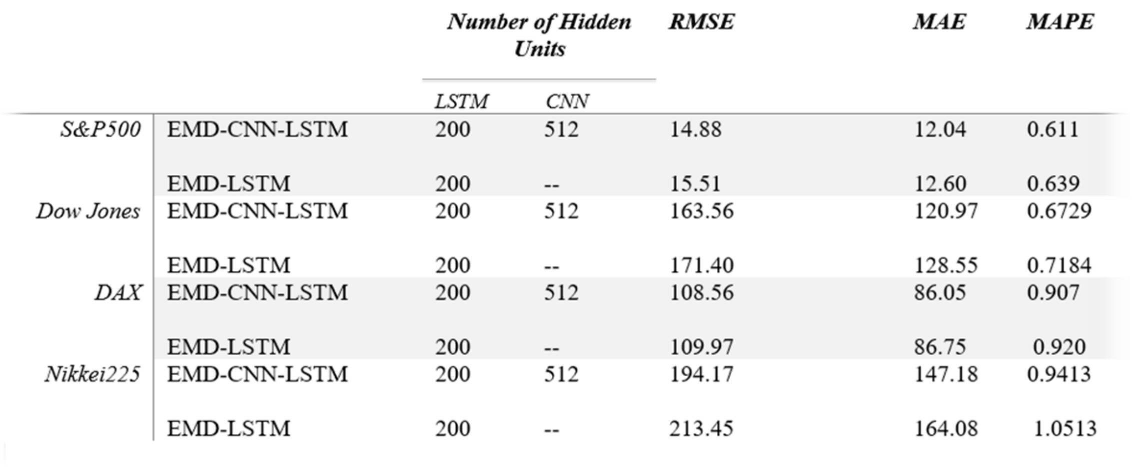


Figure 5. The prediction results of the Shanghai Composite Index at 7 consecutive days.



		Number of Hidden Units		*RMSE*	*MAE*	*MAPE*
		LSTM	*CNN*			
S&P500	EMD-CNN-LSTM	200	512	14.88	12.04	0.611
	EMD-LSTM	200	--	15.51	12.60	0.639
Dow Jones	EMD-CNN-LSTM	200	512	163.56	120.97	0.6729
	EMD-LSTM	200	--	171.40	128.55	0.7184
DAX	EMD-CNN-LSTM	200	512	108.56	86.05	0.907
	EMD-LSTM	200	--	109.97	86.75	0.920
Nikkei225	EMD-CNN-LSTM	200	512	194.17	147.18	0.9413
	EMD-LSTM	200	--	213.45	164.08	1.0513

Figure 6. Prediction based on EMD based algorithm

7. CONCLUSION

Concerning the significance of stock market prediction and its challenges, researchers always try to introduce modern methods in the analysis of these markets. LSTM as a state-of-the-art model and CNN which are deep learning models yield good results in the analysis of stock market. EMD and CEEMD are among the effective algorithms that have recently been considered in this research area. Thus, this article sought to introduce the proposed algorithm of CEEMD-CNN-LSTM by studying each of these models and evaluate them based on data of different stock price indices. The major concept of the suggested algorithm was to create a collaboration between CEEMD, CNN, and LSTM models by joining them together, which could extract deep features and time sequences. Then, we applied the trained algorithm for one-step-ahead stock price forecasting. In this regard, first, we decomposed the financial time series to different IMFs and the residual by CEEMD and EMD algorithms. The IMFs and the residual were separately analysed by CNN-LSTM model. Facilitation of their analysis with CEEMD and EMD, as well as extracting features and patterns in data with CNN and further analysis in the context of the time plus analysis of the dependencies with LSTM enhanced the predictive capabilities of this model. The practical results of this article also support this claim. Note that the assessment metrics used in this study were RMSE, MAE, and MAPE [3].

8. FUTURE WORKS

ANRES model tends to pay more attention to a new event when it first occurs, which offers us a potential improving direction in the future. In future, the model can be applied to other stock indexes like NSE, TAIEX. For further improving the system accuracy, an underlying TSK-based fuzzy system with a constrained gradient-based technique can be employed. Also, a constrained gradient-based technique can be integrated with a Mamdani-type fuzzy system so that interpretability is not compromised which may happen in case of a TSK system. For reducing the rule base size, techniques like similar rule merging can also be used. Some area that will be needed to achieved better accuracy in our future work, some scholars are concerning the used of sentiment analysis on twitter to predict stock market trend movement, Beside the social media, other qualitative indicators like news, internet and domestic policy changes can also be used as input to predict the trend of stocks. Another important concept is Elliot waves principles which can perform better on stock market trend prediction.

ACKNOWLEDGEMENTS

The authors would like to thank Vellore Institute of Technology for providing them with the resources necessary to complete this paper.

REFERENCES

[1] Heyan Huang, Xiao Liu, Yue Zhang, Chong Feng, (2022, Volume 470), News-driven stock prediction via noisy equity state representation, Neurocomputing.

[2] Chen, X., Rajan, D., & Quek, H.C. (2021). An interpretable Neural Fuzzy Hammerstein-Wiener network for stock price prediction. Inf. Sci., 577, 324-335.

[3] Gyamerah, S., & Awuah, A. (2020). Trend Forecasting in Financial Time Series with Indicator System.

[4] Jiake Li, "Research on Market Stock Index Prediction Based on Network Security and Deep Learning", *Security and Communication Networks*, vol. 2021, Article ID 5522375, 8 pages, 2021.

[5] Agüero-Torales, M.M., Salas, J.I., & López-Herrera, A.G. (2021). Deep learning and multilingual sentiment analysis on social media data: An overview. *Appl. Soft Comput.*, *107*, 107373

[6] Khang, P.Q., Hernes, M., Kuziak, K., Rot, A., & Gryncewicz, W. (2020). Liquidity prediction on Vietnamese stock market using deep learning. *KES*.

[7] Xiong, K., Ding, X., Du, L., Liu, T., & Qin, B. (2021). Heterogeneous graph knowledge enhanced stock market prediction. *AI Open, 2*, 168-174.

[8] Hadi Rezaei, Hamidreza Faaljou, Gholamreza Mansourfar, Stock price prediction using deep learning and frequency decomposition, Expert Systems with Applications, Volume 169,2021,114332,

[9] Junran Wu & Ke Xu & Xueyuan Chen & Shangzhe Li & Jichang Zhao, 2021. "Price graphs: Utilizing the structural information of financial time series for stock prediction," Papers 2106.02522, arXiv.org, revised Nov 2021.

[10] Hyun Jun Park, Youngjun Kim, Ha Young Kim, Stock market forecasting using a multi-task approach integrating long short-term memory and the random forest framework, Applied Soft Computing, Volume 114,2022,

[11] Ronen Feldman. 2013. Techniques and applications for sentiment analysis. Commun. ACM 56, 4 (April 2013), 82–89.

[12] Boyi Xie, Rebecca J. Passonneau, Leon Wu, and Germán Creamer. Semantic frames to predict stock price movement. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 873–883, 2013.

[13] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Deep learning for event-driven stock prediction. In Proceedings of the 24th International Joint Conference on Artificial Intelligence, pages 2327–2333, 2015.

[14] J.-F. Fournier, C. Bouix-Peter, D. Duvert, A.-P. Luzy, and G. Ouvry, "Intrinsic property forecast index (iPFI) as a rule of thumb for medicinal chemists to remove a phototoxicity liability," Journal of Medicinal Chemistry, vol. 61, no. 7, pp. 3231–3236, 2018

[15] J. F. Fournier, C. Bouix-Peter, D. Duvert et al., "Intrinsic property forecast index (iPFI) as a rule of thumb for medicinal chemist to remove a phototoxicity liability," Journal of Medicinal Chemistry, vol. 61, no. 7, pp. 3231–3236, 2018.

[16] A. Yoshimura and Y. Matsuno, "Dynamic material flow analysis of copper and copper alloy in global scale: —forecast of in-use stock and inputs and the estimation of scrap recovery potential—," Journal of the Japan Institute of Metals, vol. 82, no. 1, 2017.

[17] X. Zhang, Y. Zhang, S. Wang et al., "Improving stock market prediction via heterogeneous information fusion," Knowledge-Based Systems, vol. 11, 2017.

[18] X. Pang, Y. Zhou, P. Wang, W. Lin, and V. Chang, "An innovative neural network approach for stock market prediction," 7e Journal of Supercomputing, vol. 76, no. 3, pp. 2098–2118, 2020.

[19] P. F. Bestwick, "A forecast monitoring and revision system for top management," Journal of the Operational Research Society, vol. 26, no. 2, pp. 419–429, 1975.

[20] M. G. Jacox, M. A. Alexander, C. A. Stock et al., "On the skill of seasonal sea surface temperature forecasts in the California Current System and its connection to ENSO variability," Climate Dynamics, vol. 53, pp. 7519–7533, 2017

[21] B. D. Williams and M. A. Waller, "Estimating a retailer's base stock level: an optimal distribution centre order forecast policy," Journal of the Operational Research Society, vol. 62, no. 4, pp. 662–666, 2017

[22] Derbentsev, Vasily, Semerikov, Serhiy, Serdyuk, Olexander, Solovieva, Victoria, and Soloviev, Vladimir (2020) "Recurrence based entropies for sustainability indices" in EasyChair Preprint 3454.

[23] Hwang, Jungsik (2020) "Modeling Financial Time Series using LSTM with Trainable Initial Hidden States", Samsung Electronics Co., Ltd. Seoul, South Korea

[24] Xu, Wenquan, Peng, Hui, Zeng, Xiaoyong., Zhou, Feng, Tian, Xiaoying, and Peng, Xiaoyan (2019) "A hybrid modelling method for time series forecasting based on a linear regression model and deep learning." Applied Intelligence 49: 3002–3015.

[25] S. Ruder, P. Ghaffari, J.G. Breslin, A hierarchical model of reviews for aspect-based sentiment analysis, in: Proc. of the 2016 Conf. on Empirical Methods in NLP, 2016, pp. 999–1005.

[26] D. Tang, B. Qin, T. Liu, Document modelling with gated recurrent neural network for sentiment classification, in: Proc. of the 2015 Conf. on Empirical Methods in NLP, 2015, pp. 1422–1432.

[27] G. Rao, W. Huang, Z. Feng, Q. Cong, LSTM with sentence representations for document-level sentiment classification, Neurocomputing 308 (2018) 49–57.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, pages 5998–6008, 2017.

[29] Nelson, D.M., Pereira, A.C., de Oliveira, R.A. Stock market's price movement prediction with lstm neural networks. In: IJCNN. IEEE; 2017. p. 1419–1426.

[30] Qin, Y., Song, D., Cheng, H., Cheng, W., Jiang, G., Cottrell, G.W. A dual-stage attention-based recurrent neural network for time series prediction. In: Proceedings of the 26th IJCAI. 2017. p. 2627–2633.

[31] Long, W., Lu, Z., Cui, L. Deep learning-based feature engineering for stock price movement prediction. Knowledge-Based Systems 2019; 164:163– 173.

[32] Hao, Y., Cao, H. A new attention mechanism to classify multivariate time series. In: Proceedings of the 29th IJCAI. 2020.

AUTHOR

Om Mane is a sophomore at Vellore Institute of Technology, Vellore in Computer Science and Engineering.

AN OPTIMIZED METHOD FOR MASSIVE SENSITIVE DATA CLASSIFICATION IN AN INDUSTRY ENVIRONMENT

Qi Zhong, Shichang Gao and Bo Yi

Department of Computer Science and Engineering,
Northeastern University, Shenyang, Liaoning, China

ABSTRACT

In the era of big data, data is endowed with higher potential value. However, new challenges are also brought to data security, especially for the sensitive data in an industrial environment. Nowadays, with the development of industrial internet, enterprises connect each other, under which a slight carelessness may lead to the leakage of sensitive data, which will bring inestimable losses to enterprises. Hence, sensitive data classification is required as a secure way to avoid such situation. This paper presents a sensitive data classification method based on an improved ID3 decision algorithm. Firstly, we introduce the idea of attribute weighting to optimize the basic structure of traditional ID3. Secondly, we use the weighted information gain to select nodes during tree construction, which improves multi-value bias defect compared with the traditional algorithm. Experimental results show that we can achieve branching accuracy up to 97.38%.

KEYWORDS

Sensitive data, Data classification, ID3 decision tree, Industrial environment.

1. INTRODUCTION

Since entering the 21st century, the importance of data has become increasingly apparent and the amount of data has grown exponentially. Such large-scale data promotes the rapid development of the Internet, artificial intelligence and other emerging technologies. With the continuous progress of science and technology, the era of big data has arrived. Although big data brings a lot of convenience, the huge information base also brings some difficulties for accurate screening of information. Therefore, effective classification of big data is an important mean to improve the efficiency of data processing in the network big data platform, as well as an available way to improve users' query experience. Data classification can improve data governance and facilitate data management.

The era of Industry 4.0 is known as "Fourth Industrial Revolution", commonly understood as the transformation from traditional manufacturing to intelligent manufacturing [1]. Industrial Internet is a technology that uses Internet of Things for large-scale industrial manufacturing. The idea behind industrial internet is based on big data, which can process and transmit information more accurately, saving costs for large-scale manufacturing enterprises and achieving optimal output efficiency. Large-scale manufacturing industry chain is a complex network chain consisting of a series of different types of enterprises. Each enterprise, as a data source, continuously generates massive and heterogeneous data at each stage of industrial chain, which is of great value in industrial chain management. There are many enterprises and departments involved in industrial chain network, and both cooperative and competitive relations are required among them, which brings challenges to data sharing among enterprises.

On the one hand, all stages of the industrial chain urgently need transparent access and sharing of key information, such as production information, to solve the problem of information asymmetry between upstream and downstream enterprises, improve enterprise cooperation efficiency and reduce costs. On the other hand, there is competition among core enterprises of the industrial chain, such as suppliers at the same level. For the protection of their own privacy, enterprises do not want private data such as trade secrets and financial information to be obtained by others. How to ensure transparent access and sharing of necessary data among enterprises and realize protection of sensitive data is an urgent problem to be solved.

Generally, the data classification needs labels and categories for a given data type. These types will be used to set sensitivity and confidentiality levels. Sensitive data refers to data that is unknown to public, has actual or potential use value, and will cause harm to society, enterprises or individuals if lost, improperly used or unauthorized access [2]. With the development of big data, Data security faces new challenges. After years of infomationization construction, large amounts of enterprise-level data has been accumulated by large-scale manufacturing industry chain, including customer data, marketing data, production data and other important sensitive data.

However, with the large-scale use of network data, especially in the application of big data, there are loopholes in the big data platform itself, and a slight carelessness may lead to data leakage. Once these sensitive data are leaked, it will inevitably bring incalculable losses to enterprises. If all data are set to high sensitivity level instead of data classification, it will lead to high cost, high operational complexity and high cost in data security protection, and also greatly reduce enterprise cooperation efficiency. However, if all data are set to low sensitive level, due to leakage of important private data, these data will be obtained by attackers or used by malicious people to engage in business transactions or steal personal privacy, thus causing unpredictable losses to interests of enterprise and employees. Therefore, sensitive data should be properly graded and classified as needed.

Existing sensitive data classification methods focus on classification of imbalanced data (e.g. [3]-[5]) and feature selection (e.g. [6]-[8]). At present, existing imbalanced data classification technology can be roughly divided into data-level method and algorithm-level method [3]. Datalevel strategy is more general because it does not rely on classifier model; However, when creating samples for minority classes [4] are added, valuable information may be eliminated due to the loss of majority class samples, leading to over fitting. Algorithm-level strategy is not widely used because it is limited by specific classifiers or datasets [3]. The existing imbalanced data classification models mainly focus on the classification performance of majority class samples, which does not correspond to the fact that the sensitivity of sensitive data is inversely proportional to the scale of data. Techniques such as [5] balance different classes by enlarging samples with minority classes (oversampling) or discarding samples from majority classes (under sampling), achieving improved data-level sampling. Although undersampling technique equalizes different categories of samples and reduces time cost, the loss of important information in sensitive data with imbalanced scale will appear.

Feature selection methods ([6]-[8]) can effectively project high-dimensional data into relatively low-dimensional space. Feature selection is mostly based on filtering and wrappering approaches that suffer from poor classification accuracy, high computational cost, irrelevant selection and redundant features [6]. This is due to the limitations of adopted objective functions leading to overestimation of feature significance. On the contrary, hybrid feature selection methods based on information theory and nature-inspired meta-heuristic algorithms have advantages of high computational efficiency, good scalability, avoidance of redundancy and less informative features, and independence from classifier. However, these methods have three common shortcomings on sensitive data classification: *(1) poor trade-off between exploration and exploitation phase; (2) trapping in optimal local solution; (3) avoiding irrelevance and redundancy of selected features.*

To address the above challenge, we propose an improved ID3 decision tree based efficient sensitive data classification method. Specifically, we divide sensitive data into four sensitive levels according to actual situation, use data subset with sensitivity label to train the model, and then classify sensitive level of data by integrating characteristics with this model. Based on traditional ID3 algorithm, we introduces the idea of attribute weighting and uses the weighted information gain to select nodes during tree construction, which improves multi-value bias defect in traditional algorithm. We use different labels and training set size, and add error sensitive data of manual intervention for training to prevent the occurrence of over-fitting phenomenon. Our research makes full use of the information in sensitive data and minimizes the loss of information while preventing irretrievable loss caused by the leakage of important privacy data, which has great practical significance. Using ID3 decision tree can achieve our goal well. The model can not only keep concise structure, but also synthesize various data characteristics to decide the final classification of data, whose accuracy meets industrial requirements. The experimental results indicate that the proposed method can ensure maximum accuracy while avoiding the problem of low model efficiency caused by multi-value bias phenomenon compared with existing methods.

The rest of this work are as follows: Section 2 summarizes the related work about the sensitive data classification. Section 3 introduces the proposed improved ID3 decision tree algorithm for sensitive data classification. Section 4 evaluates the performance of proposed decision tree model. And section 5 concludes this work.

2. RELATED WORK

There are three main kinds of technologies used for sensitive data classification, which are differential privacy protection, quantum machine learning and neural network. We present them as follows.

2.1. Differential Privacy Protection based Sensitive Classification

Dwork et al. [9] put forward "differential privacy" protection framework in 2017, injecting an appropriate amount of random noise into original data sets, processed and deformed data sets, query results and query functions of statistical functions to ensure that whether the target data is stored in database does not affect search results of functions. At the expense of certain prediction accuracy, data privacy is not compromised.

Private Aggregation of Teacher Ensembles (PATE) strategy proposed by Nicolas Papernot et al. [10], combined with differential privacy protection framework, proposes a solution to the problem of data privacy leakage in classification task of image data sets with few categories, and protects privacy security of sensitive attribute values of specific data in classification task. However, this method is not satisfactory for classification task of image data sets with a large number of categories, and can not achieve the function of balancing model accuracy and data privacy security.

Uri Stemmer [11] proposes to apply the method of local differential privacy in k-means clustering algorithm to adapt to privacy needs of different users, reduce concerns about exposure of specific sensitive attribute values. Although local privacy budget is considered in this method, the overall differential privacy budget of the algorithm is not taken into account, which will lead to lower accuracy of new data partition results of the trained model. Literature [12] proposes a clustering scheme LDPK-modes (Local Differential Privacy K-modes) that can support localized differential privacy technology. Compared with traditional privacy protection algorithm based on centralized differential privacy clustering, it solves the problem that it is difficult to find a trusted third party to collect and process sensitive data in practical application scenarios, reduces the possibility of the third party stealing user sensitive data, and considers the possibility of user sensitive data leakage from another perspective. However, this method does not guarantee the accuracy of exploring laws and properties of data because clustering is sensitive to initial center point.

2.2. Quantum Machine Learning based Sensitive Classification

Existing quantum machine learning privacy protection mainly focuses on the following three types of privacy protection technologies: privacy protection technology based on differential privacy, privacy protection technology based on homomorphic encryption and privacy protection technology based on secure multi-party computing.

Quantum differential privacy is a quantum scheme of differential privacy, which has been used to protect the privacy of quantum machine learning models such as classification, regression and neural network. Du et al. [13] proposes a differential privacy protection scheme for quantum classifiers by using depolarization noise of quantum channels, and gives robustness bounds of anti-disturbance. Watkins et al. [14] proposes a quantum machine learning algorithm based on differential privacy framework to realize privacy protection through differential privacy optimization algorithm based on Gaussian noise.

Homomorphic encryption is widely used in quantum computing to realize privacy-protecting quantum computing. Gong et al. [15] proposes a quantum homomorphic encryption method based on improved T-gate update, and realized a quantum K-means algorithm based on cloud server for privacy protection. The increase in the number of H-gate and T-gate in algorithm leads to an increase in algorithm complexity and a decrease in accuracy.

Quantum secure multi-party computing is a quantum solution for secure multi-party computing, which has been used in privacy protection research of quantum machine learning. Li et al. [16] proposes a private single-party delegation training protocol for variable component sub classifiers based on blind quantum computing, and extends this protocol to multi-party private distributed learning with differential privacy. Xia et al. [17] proposes QuantumFed, a quantum federated learning framework, which enables multiple quantum nodes with local quantum data to train model together. Chehimi et al. [18] proposes a complete quantum federated learning framework for clients with quantum computing capabilities.

2.3. Neural Network based Sensitive Classification

Deep learning prediction model based on privacy protection was first proposed by Dowlin et al. [19]. This model implements CryptoNets, a protocol for privacy protection of convolutional neural network. Mohassel et al. [20] adopts a hybrid protocol framework to realize training of neural networks and evaluate prediction classification of both parties. Liu et al. [21] proposes MiniONN, which uses single-instruction multi-data technology (SIMD) to transform existing neural network into observable neural network.

Some studies have also focused on privacy of data in training stage of neural networks, mainly aiming at back propagation algorithms. Bu et al. [22] proposes a back propagation algorithm for privacy protection based on BGV encryption mechanism in the cloud, which uploads all computing load to the cloud to reduce computing overhead, and also uses BGV [23] to protect data privacy during processing. Zhang et al. [24] proposes to use BGV encryption scheme to effectively support secure computing of high-order back propagation algorithm, so as to conduct model training in the cloud. In order to reduce the depth of multiplication, after each iteration, the updated weights are sent back to all parties for decryption and re-encryption, so the communication cost of this scheme is very high.

Most of the existing work above focus on privacy protection of image data, classifier data or sensitive data classification in training process, while not suitable for enterprise-level sensitive data generated in practical large-scale manufacturing industry chain. In addition, compared with above methods, proposed sensitive data classification method maintains a very concise algorithm structure while ensuring accuracy in line with industry requirements, and is still equipped with the ability to classify data by integrating various features, which has great practical significance.

3. IMPROVED ID3 DECISION TREE FOR SENSITIVE DATA CLASSIFICATION

Decision tree algorithm is one of the most basic and effective algorithms in artificial intelligence and machine learning. Decision tree algorithm constructs mathematical analysis model and obtains basic classification rules through training and mining rule of irregular and disordered sample data, and then makes prediction classification of target data set. On the basis of the preprocessing sensitive data, the ID3 decision tree is constructed and optimized in this paper to classify the sensitive industrial data. Due to the simple ideas and strong learning ability of ID3 for data sample features, we first adopt it as the basis for data classification, which is then improved by using the attribute weighting. After that, the improved ID3 algorithm is used to achieve more accurate sensitive data classification.

3.1. Basic ID3 Description

The basis of ID3 algorithm looks for the attribute with the largest amount of information in input data set as a node of decision tree, and then classifies sample data according to different attribute values under this attribute. The core of the algorithm is information entropy, which is defined as follows:

$$Ent(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{1}$$

where p_i is the probability of the target attribute appearing in sample data set D.

Information entropy is used to calculate the purity of a sample set, which represents the number of sample types contained in this node. It can be seen from Formula (1) that higher information entropy represents lower sample purity and more sample categories of this node. On the contrary, if information entropy is lower, the purity of this node sample will be higher, and sample data of this node will have fewer categories.

Information gain is the change value of information entropy of sample data before and after attribute division, which can be used to measure difference of information entropy in probability distribution. The information gain divided according to attribute A is defined as:

$$Gain(A) = \mathrm{Ent}(D) - Ent_A(D) \tag{2}$$

ID3 algorithm calculates information gain of each attribute through information entropy and takes it as the dividing criterion to compare which attribute is selected for node splitting. The attribute with the highest information gain value is selected as the standard for each partition, and the process is repeated until information entropy of the sample set is zero. ID3 algorithm is simple and feasible with intuitive and clear process. It can train classification model well for most discrete data.

3.2. Improved ID3 Algorithm based on Attribute Weighting

In ID3 algorithm, each non-leaf node selects the attribute with the maximum information gain value as split attribute, which often ignores the attribute that is very important for decision making but has few values, namely the defect of multi-value bias in ID3 decision tree. Aiming at this shortcoming, this paper proposes an improved ID3 algorithm based on attribute weighting. In this algorithm, attribute weights are introduced in the calculation of information gain, and correlation coefficients are used as attribute weight coefficients to ensure rationality of weighted coefficients.

Correlation coefficient reflects closeness of the relationship between random variables, and its value is between -1 and 1. In the same sample space, the closer the absolute value of correlation coefficient between attributes is to 1, the higher the correlation degree between attributes is, so the correlation weight between attributes can be calculated by using the meaning of correlation coefficient. Set A_c as a conditional attribute and A_d as a decision attribute, and the correlation coefficient between attribute A_c and attribute A_d can be calculated according to Formula (3).

$$\rho(A_c, A_d) = \frac{\mathrm{cov}(A_c, A_d)}{\sqrt{D(A_c)D(A_d)}} \tag{3}$$

where $cov(A_c, A_d)$ is the covariance of A_c and A_d, $D[A_c]$ is the variance of A_c, and $D[A_d]$ is the variance of A_d.

Thus, the correlation weight of attribute A_c to decision attribute A_d can be obtained:

$$\lambda = |\rho(A_c, A_d)| = \left| \frac{\mathrm{cov}(A_c, A_d)}{\sqrt{D(A_c)D(A_d)}} \right| \tag{4}$$

Therefore, according to the characteristics of ID3 algorithm that selects the maximum information gain as splitting basis and the significance of correlation coefficient, Formula (5) is obtained after the improvement of attribute weighting from Formula (2) :

$$Gain(A) = \mathrm{Ent}(D) - (1-\lambda)Ent_A(D) \tag{5}$$

3.3. Sensitive Data Classification Based on Improved ID3 Algorithm

The sensitive data classification based on improved ID3 algorithm proposed in this paper is mainly composed of three parts: data preprocessing, feature selection of decision tree nodes and decision tree construction.

3.3.1. Data Preprocessing

There are many non-numeric attribute values in sensitive data. In order to make data more suitable for the model to be established, and improve computing efficiency of ID3 decision tree, the specific operation of data conversion in this paper is to transform non-numerical characteristic parameters in original sensitive data into numerical characteristic parameters. For example, in original sensitive data sample set, the characteristic value of *safety* feature in the data is text type, with four levels of "*Low*", "*Med*", "*High*" and "*Vhigh*". Convert them correspondingly to numeric types, that is, "*Low*"—1, "*Med*"—2, "*High*"—3, and "*Vhigh*"—4 respectively, which then achieves the operation of data type conversion.

3.3.2. Feature Selection

The feature selection criterion of each decision node in ID3 decision tree algorithm is based on information gain theory. According to the improved ID3 algorithm proposed in this paper, feature with the maximum information gain after improvement is selected as the feature of current node by calculating information gain of each feature based on attribute weighting.

According to Formula (1), (3) and (5), taking unsegmented training sample set of sensitive data as an example, information entropy, weighting coefficient and improved information gain of six conditional features of sensitive data are calculated as shown in Table 1.

Table 1. Key calculation results of unsegmented training sample set.

Feature	Information Entropy	Weighting Coefficient	Information Gain
buying	1.10	0.28	0.40
maint	1.13	0.24	0.35
doors	1.20	0.05	0.06
persons	0.99	0.34	0.55
lug-boot	1.17	0.04	0.07
safety	0.94	0.41	0.64

According to data in Table 1, safety feature has the maximum information gain after attribute weighting, so safety feature is selected as the root node feature. The selection of other node features is similar to the above process.

3.3.3. Decision Tree Construction

The core stage of decision tree algorithm is tree construction process. Suppose training set D have M attributes A_i, where i = 1, 2, ..., M. For the improved ID3 algorithm proposed in this paper, its tree-construction process is as follows:

Step 1. Construct a node. If all samples are on this node, stop the algorithm, change the node into a leaf node, and mark it with this class.

Step 2. If all samples are not on the same node, work out information entropy of each attribute, figure out improved information gain of each attribute according to Formula (5), and then select the attribute with maximum information gain as classification basis. That is, *best_attribute* = A_j where A_j satisfies Formula (6):

$$Gain(A_j) = \max_{1 \le i \le M} (\mathrm{Ent}(D) - (1-\lambda) Ent_{A_i}(D)) \tag{6}$$

Step 3. Assume that the attribute value of A_j has V different discrete values, create a branch for each value, and divide sample set D into V subsets $\{D_1, D_2, ..., D_v\}$ according to attribute A_j.
Step 4. Repeat ***Step 1-Step 3*** until all attributes are used up.

In the process of tree construction, improved ID3 algorithm excludes the special case of ***Step 1*** and judges all attribute values in attribute set. If there are multiple values for multiple attributes in the attribute set, the optimal attribute is selected, and sub-nodes are divided according to different values of sample data on the optimal attribute. After dividing data of the current node, the features in data set that have been divided in the previous step are removed to generate data subset, and a new round of division is continued. Until data set cannot be divided, the decision tree stops growing, and finally a complete classification decision tree is returned.

Table 2. Pseudo code of improved ID3 decision tree construction.

```
Algorithm Improved ID3 Decision Tree Construction

Input Training Data Set; Attribute List
Initialize a node $N$
if samples are on this node
  return $N$ as a leaf node and mark $N$ with this class else
    for $i \in \{1,2,...,M\}$ do
        figure out information gain with attribute weight of each
        attribute $A_j \in \{A_1,A_2,...,A_{M-i}\}$ in Attribute List select
        best_attribute with maximum information gain remove
        best_attribute from Attribute List mark $N$ as
        best_attribute
        for attribute value $a_k \in \{a_1,...,a_v\}$ in best_attribute do
            create a branch and divide the samples. end
    for
     end for end if
Output A Decision Tree
```

3.3.4. Sensitive Data Classification Based On ID3 Improved Algorithm

The improved ID3 algorithm proposed in this paper introduces attribute weights in calculation of information gain, considering the correlation between attributes, which not only makes up for multi-value bias defect of ID3 algorithm to a certain extent, but also applies to data analysis in more cases. The specific process of sensitive data classification based on the improved ID3 algorithm proposed in this paper is described as follows:

Step 1. Divide sensitive data set into training set and test set in a ratio of 7:3.

Step 2. Calculate the information entropy and attribute weight of each condition attribute of sensitive data in training set, and introduce weights into the calculation of information gain according to Formula (7) :

$$Gain(A_i) = \mathrm{Ent}(D) - \left(1 - \left|\frac{\mathrm{cov}(A_i, A_d)}{\sqrt{D(A_i)D(A_d)}}\right|\right)Ent_{A_i}(D) \qquad (7)$$

where $i = 1, 2, \ldots, M-1$.

Step 3. Select the attribute with the maximum information gain after improvement as the split attribute to construct a decision tree.

Step 4. Test the constructed decision tree with test set.
The specific flow chart of sensitive data classification based on improved ID3 algorithm is shown in Figure 1.

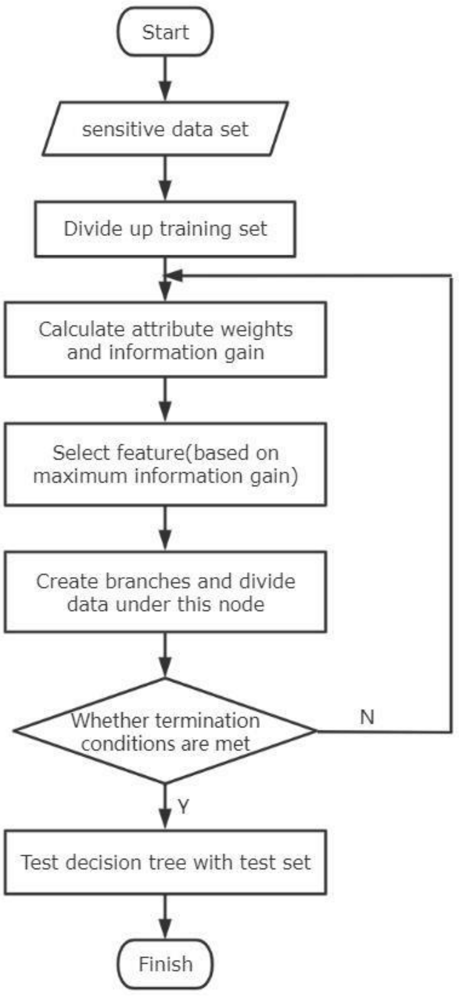

Start
sensitive data set
Divide up training set
Calculate attribute weights and information gain
Select feature(based on maximum information gain)
Create branches and divide data under this node
Whether termination conditions are met
N
Y
Test decision tree with test set
Finish

Figure 1. Algorithm flow chart

4. Performance Evaluation

4.1. Setup

A total of 3387 sensitive data in data set are used in the experiment, and each piece of data has 7 features, including 6 conditional features and 1 decision feature (sensitivity level). Sensitive data are divided into four levels: public data, external sensitive data, internal inter-departmental sensitive data and internal intra-departmental sensitive data. Each level has 2353,680,156 and 198 pieces of data respectively.

The sensitive data set consists of training data set and test data set. Training sample set constructs ID3 decision tree classification model while test sample set evaluates the performance of ID3 decision tree model. Sample selection should be representative, typical and complete. In order to improve classification accuracy, the data set is divided into training set and test set at the ratio of 9:1, 8:2, 7:3 and 6:4, respectively. The experimental results are shown in Figure 1. After experiments on training sets of different scales, it is finally found that the optimal classification effect of decision tree model can be achieved by dividing training data set according to the ratio of 7:3. After dividing by 7:3, there are 2370 sensitive data pieces in training set and 1017 sensitive data pieces in test set.

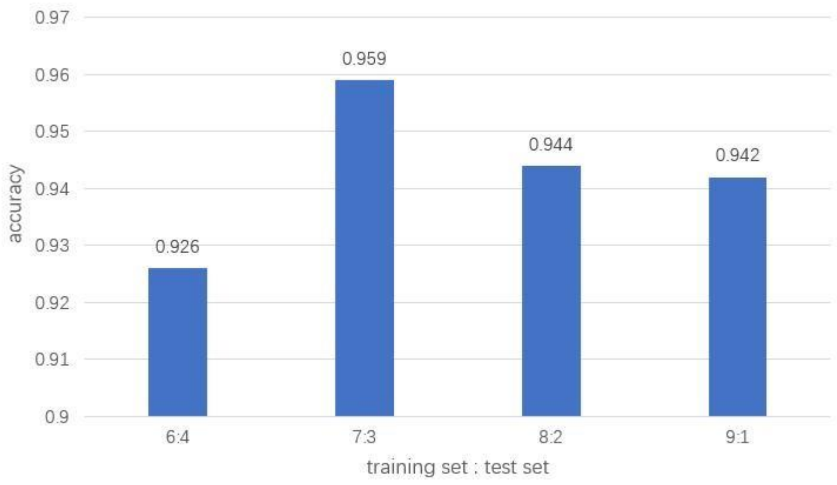


Figure 2. Branching accuracy under different training set scale

4.2. Metric

Generally, the evaluation of decision tree classification model is mainly carried out from the following two aspects:

(1) Accuracy: Accuracy refers to the ability of decision tree classification model to accurately predict category of new or unknown target data. The accuracy of prediction is related to the number of samples in training set, the number of decision attributes, the number of attribute values and the distribution of samples to be predicted.

(2) Robustness: Robustness refers to the ability of the classification model to accurately classify data sets in the case of incorrect data and incomplete data. Data collected in real life will inevitably have incomplete data, data errors, data redundancy and other phenomena, so the constructed decision tree classification model should have the ability to

deal with these data well, so as to avoid the existence of such data affecting performance of classifier.

4.3. Result Analysis

4.3.1. Accuracy Analysis

Table 3 shows the accuracy of data classification of each sensitivity level. As can be seen from Table 3, the classification results of data of all sensitivity levels are highly accurate, which can meet the needs of industry.

Table 3. Accuracy of different sensitivity level data.

Sensitive level	public	external	Inter-departmental	Intra-departmental
Sample number	2353	680	156	198
Error number	47	40	9	14
Accuracy	98.0%	94.1%	94.2%	92.9%

Figure 3 shows the classification accuracy of traditional ID3 decision tree algorithm and improved ID3 decision tree algorithm in this paper under different training sample set sizes. As can be seen from Figure 3, the accuracy of ID3 decision tree algorithm after attribute weighting is significantly improved. And the smaller the training set size, the better the performance of improved ID3 algorithm compared with traditional one.

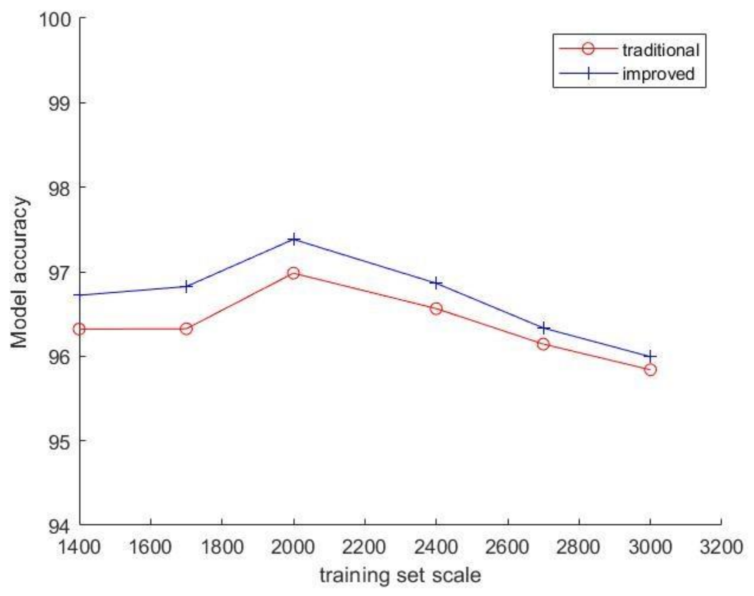


Figure 3. Accuracy in different training set scale

4.3.2. Robustness Analysis

In order to verify the robustness of the improved ID3 decision tree algorithm in this paper, noise data of manual intervention is added for training under the same data set size. The experimental results are shown in Figure 4. As can be seen from Figure 4, when the proportion of noise data is less than 8%, the robustness of improved ID3 decision tree algorithm is significantly higher than that of traditional ID3 decision tree algorithm. However, when the proportion of error sensitive data is 9% and 10%, the robustness of traditional ID3 algorithm is better than that of improved ID3 algorithm. The reason for this phenomenon may be that the calculation of attribute weights is deviated due to excessively high noise data, which affects the selection of feature in each node and worsens classification results.

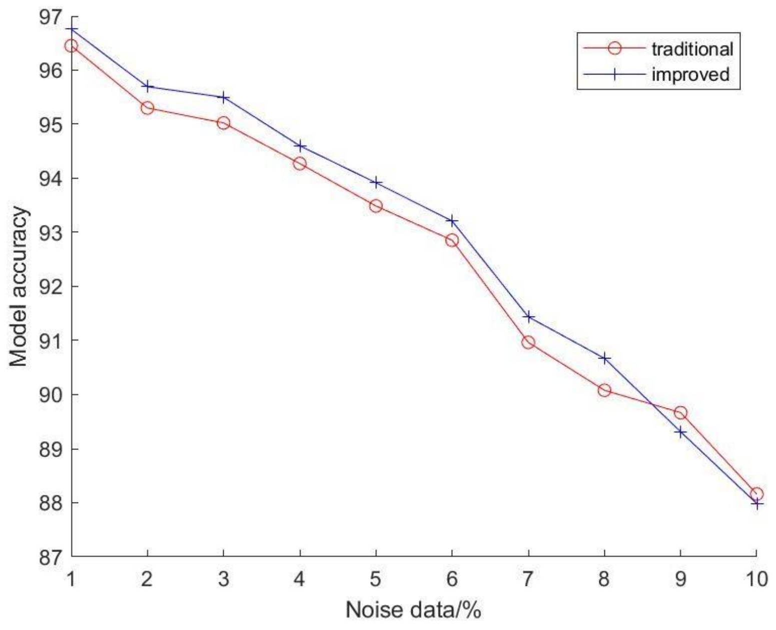


Figure 4. Accuracy under different noise data ratio

In addition, we also verify robustness of the algorithm from the perspective of data sources. The Vote, DNA-data, Weather and Soybean data sets are selected for training. The experimental results are shown in Figure 5. As can be seen from Figure 5, under four different data sets, the improved ID3 algorithm proposed in this paper has stable classification performance, that is, good robustness. Compared with the traditional method, the improved algorithm also has higher accuracy, which further verifies the performance optimization effect of this algorithm.

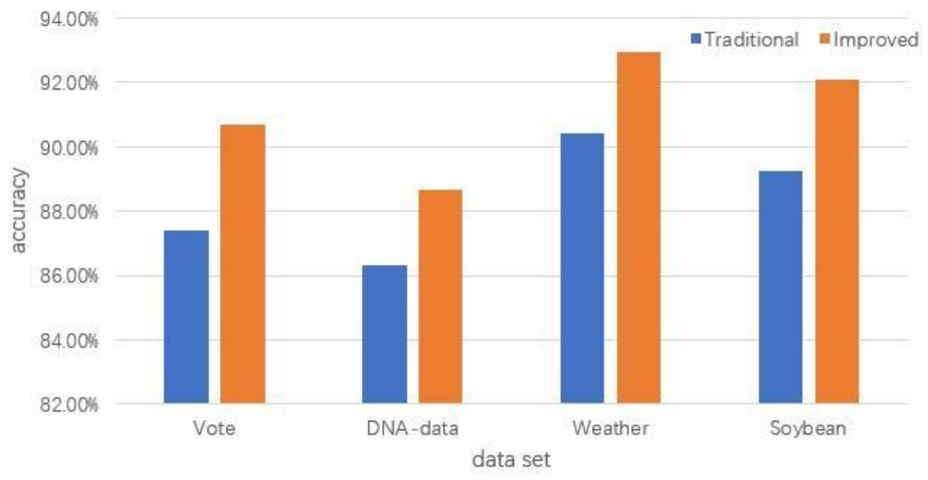


Figure 5. Accuracy under different data set

5. CONCLUSION

The sensitive data classification based on improved ID3 decision tree algorithm by this invention realizes the maximum utilization of sensitive data generated by large-scale manufacturing industry chain in big data environment. While making full use of information in sensitive data, it prevents irreparable loss caused by the leakage of important sensitive data, which is of great practical significance. According to experimental results, the branch accuracy of improved ID3 decision tree algorithm for sensitive data can reach 98%, which is significantly higher compared with traditional ID3 decision tree. In addition, it can still meet industrial needs with good test results even in the case of artificial error data added.

ACKNOWLEDGEMENTS

This work is supported by the National Key R&D Program of China under Grant No. 2021YFB3300900.

REFERENCES

[1] Miguel Oliveira & Daniel Afonso, (2019) "Industry Focused in Data Collection How Industry 4.0 is Handled by Big Data[C]", *Proceedings of 2019 International Conference on Data Science and Information Technology（DSIT 2019）*, DOI: 10.26914/c.cnkihy.2019.092721.

[2] Ji-sung Park, Gun-woo Kim & Dong-ho Lee, (2020) "Sensitive Data Identification in Structured Data through GenNER Model based on Text Generation and NER[C]", *Proceedings of 2020 2nd International Conference on Computing, Networks and Internet of Things（CNIOT 2020）*, DOI: 10.26914/c.cnkihy.2020.033027.

[3] Xia Shuyin & et al, (2021) "Granular Ball Sampling for Noisy Label Classification or Imbalanced Classification[J]", *IEEE transactions on neural networks and learning systems*, pp112

[4] Sebastián Maldonado & Julio López, (2018) "Dealing with high-dimensional class-imbalanced datasets: Embedded feature selection for SVM classification[J]", *Applied Soft Computing*, 67: pp94-105.

[5] Funa Zhou & et al, (2020) "Deep learning fault diagnosis method based on global optimization GAN for unbalanced data[J]", *Knowledge-Based Systems*, 187(C): pp104837-104837.

[6] Andrea Bommert & et al, (2020) "Benchmark for filter methods for feature selection in highdimensional classification data[J]", *Computational Statistics and Data Analysis*, 2020, 143(C): pp106839-106839.

[7] Li Gui & et al, (2020) "DLEA: A dynamic learning evolution algorithm for many-objective optimization[J]", *Information Sciences*, 574: pp567-589.

[8] Sadeghian Zohre, Akbari Ebrahim & Nematzadeh Hossein, (2021) "A hybrid feature selection method based on information theory and binary butterfly optimization algorithm[J]", *Engineering Applications of Artificial Intelligence*, p97

[9] Cynthia Dwork & et al, (2017) "Calibrating Noise to Sensitivity in Private Data Analysis[J]", *Journal of Privacy and Confidentiality*, 7(3): pp17-51.

[10] Nicolas Papernot, Shuang Song, Ilya Mironov & et al, (2018) "Scalable private learning with PATE[C]", *The proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada: ICLR: pp1-15.

[11] Uri Stemmer, (2019) "Locally Private k-Means Clustering[J]", *CoRR*, abs/1907.02513.

[12] Peng Chunchun, Chen Yanli & Xun Yanmei, (2021) "k-modes clustering guaranteeing local differential privacy[J]", *Computer Science*, 48(2): pp105-113.

[13] Du Yuxuan & et al, (2021) "Quantum noise protects quantum classifiers against adversaries[J]", *PHYSICAL REVIEW RESEARCH*, 3(2).

[14] Watkins WM, Chen SYC & Yoo S, (2021) "Quantum machine learning with differential privacy [J]", *arXiv preprint*, arXiv:2103.06232.

[15] Gong C, Dong Z, Gani A & et al, (2021) "Quantum k-means algorithm based on trusted server in quantum cloud computing", *Quantum Inf Process* 20, 130.

[16] Li Weikang, Lu Sirui & Deng Dongling, (2021) “Quantum private distributed learning through blind quantum computing [J]”, *arXiv preprint,* arXiv: 2103.08403.

[17] Xia Qun, Li Qun & QuantumFed, (2021) “A federated learning framework for collaborative quantum training [J]”, *arXiv preprint,* arXiv: 2106.09109.

[18] Chehimi M & Saad W, (2021) “Quantum federated learning with quantum data [J]”, *arXiv preprint,* arXiv, 2106.00005.

[19] Dowlin N, Gilad-Bachrach R, Laine K & et al, (2016) “Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy [C]”, *Proceedings of the 33nd In-ternational Conference on Machine Learning,* JMLR. org, pp201-210.

[20] Mohassel P & Zhang YP, (2017) “SecureML: A system for scalable privacy-preserving machine learning [C]”, *IEEE Sym-posium on Security and Privacy (SP),* IEEE, pp19-38.

[21] Liu J, Juuti M, Lu Y & et al, (2017) “Oblivious neural network predictions via MiniONN transformations [C]”, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Com-munications Security,* ACM, pp619-631.

[22] Bu FY, Ma Y, Chen Z K & et al, (2015) “Privacy preserving back-propagation based on BCV on cloud[C]”, I*EEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12thInternational Conference on Embedded Software and Systems, IEEE,* pp1791 - 1795.

[23] Gentry C, Halevi S, Peikert C & et al, (2013) “Ring switching in BGV-style homomorphic encryption [J]”, *Journal of ComputerSecurity*, 21(5): pp663 -684.

[24] Zhang QC, Yang LT & Chen ZK, (2016) “Privacy preserving deep computation model on cloud for big data feature learning [J]”, *IEEE Transactions on Computers*, 65(5): pp1351 -1362.

AUTHORS

Qi Zhong is studying for a bachelor's degree in college of Computer Science and Technology at North eastern University in Shenyang, China. Her research interests include neural network, AI-enable network and cognitive radio network.

Gaoshichang, male, 21 years old, is studying at North eastern University for a Bachelor of engineering degree in communication engineering. During the undergraduate period, he devoted himself to scientific research competitions and won many awards. During his undergraduate years, his main research direction was big data classification.

Bo Yi received the BS and MS degrees in computer science from the South-Central University for Nationalities, Wuhan, China, in 2012 and 2015, respectively, and the PhD degree in computer science from North eastern University, Shenyang, China, in 2019. He is currently a lecturer with the College of Computer Science and Engineering, North eastern University, Shenyang, China. His research interests include routing and service function chain in SDN, NFV, deterministic networking, and cloud computing.

PERFORMANCE STUDY OF TIME SERIES DATABASES

Bonil Shah, P. M. Jat and Kalyan Sasidhar

DAIICT, Gandhinagar, India

ABSTRACT

The growth of big-data sectors such as the Internet of Things (IoT) generates enormous volumes of data. As IoT devices generate a vast volume of time-series data, the Time Series Database (TSDB) popularity has grown alongside the rise of IoT. Time series databases are developed to manage and analyze huge amounts of time series data. However, it is not easy to choose the best one from them. The most popular benchmarks compare the performance of different databases to each other but use random or synthetic data that applies to only one domain. As a result, these benchmarks may not always accurately represent real-world performance. It is required to comprehensively compare the performance of time series databases with real datasets. The experiment shows significant performance differences for data injection time and query execution time when comparing real and synthetic datasets. The results are reported and analyzed.

KEYWORDS

Timeseries database, benchmark, real-world application.

1. INTRODUCTION

The importance of sensors has been growing in recent years as the fourth industrial revolution has begun. The increased use of sensors results in a rise in the volume of sensor data that must be stored and processed. The Database Management System must be scaled appropriately to handle that volume of data. There are differing opinions on whether typical Relational Database Management Systems can handle such a large volume of data. Knowing that sensor data has a timestamp associated with it most of the time and can thus be utilized as an index, a trade-off between RDBMS and NoSQL DBMS arose: Time Series Databases.

Time series data are utilized in the finance business (e.g., historical stock performance research), the IoT industry (e.g., capturing the reading from sensors), and the analytics industry (e.g., tracking performance over time). Time series data is used for sensors, networks, the stock market, and other applications [4]. Time series databases make it easy to manage events that are tracked, monitored, downsampled, and aggregated over time [12, 14]. A few years ago, Time Series Databases were not so popular. It was only used by some trading applications or any other applications where you required data monitoring. Everyone else was using relational databases or NoSQL back then. But in the last three-four years, TSDBs like InfluxDB, OpenTSDB, and TimescaleDB have gained popularity.

There are numerous time series benchmarks available, which are used to compare database performance, such as IoTBenchmark, TSDBBench, TS-Benchmark, etc. However, all of these benchmarks measure the performance of time series databases using random or synthetic

DOI: 10.5121/csit.2022.121406

datasets. The biggest issue with using pre-captured or pre-calculated data is that it only applies to one circumstance. The queries in these benchmarks are also based on the datasets. It is required to study how the time series databases perform on real datasets of different domains like finance, analytics, and IoT. It is also important to consider various types of queries, such as queries with aggregate functions, exact point queries, queries with time range filters, etc. These are frequently used queries in industries to access historical data.

2. LITERATURE REVIEW

This section represents the literature survey and is structured as follows: Section 2.1 summarises all the types of databases, like relational and non-relational databases. Section 2.2 presents detailed information about time series databases and their use cases. Section 2.3 has information about some existing time series benchmarks.

2.1. Databases

A database is a structured collection of data that can be accessed and managed efficiently. Databases are used to store, maintain, and access any type of data. They gather data on people, places, or things. That data is collected in one location so that it can be observed and analyzed.

2.1.1. Database Management System

A database management system (or DBMS) is the application used to manage databases. The system provides users with the ability to perform various operations such as create, read, write, and delete. MySQL, Oracle, etc., are some popular examples of DBMS used in different applications.

2.1.2. Relational Databases

Relational databases are based on the relational model, a simple and obvious manner of representing data in tables. Each row in the table is a record with a unique ID known as the key in a relational database. The table's columns carry data attributes, and each record typically includes a value for each attribute, making it simple to construct links between data points. SQL is the query language used to manipulate and retrieve the data from the relational database.

2.1.3. Non-Relational Databases

Non-relational databases are often known as "NoSQL," which stands for Not Only SQL. It is a database that does not use the tabular schema of rows and columns that is common in most traditional database systems. Non-relational databases, on the other hand, use a storage model that is optimized for the specific needs of the data being stored. Following are some characteristics of non-relational databases: The scalability of the non-relational database is very high, which makes it suited for massive amounts of data. NoSQL databases, in comparison to SQL databases, are not complex. They store data in an unstructured or semi-structured format that does not require any relational or tabular structure. NoSQL databases are highly durable as they can accommodate data ranging from heterogeneous to homogeneous.

2.2. Time Series Databases

Time series databases are used to manage time series data. A time series is an ordered sequence of variable values over a specified time interval. Time Series is a popular format for representing and analyzing data that changes over time [13]. Formally, a time series is a description of a stochastic process consisting of a collection of pairs [(p_1, t_1), (p_2, t_2}, ... (p_n, tn})] where p_i is the information collected in time t_i. A DBMS that can (i) store a record consisting of a timestamp, value, and optional tags, (ii) store multiple records aggregated, (iii) query for records, and (iv) contain time ranges in a query is called time series databases.

2.2.1. Properties

S. N. Z. Naqvi, S. Yfantidou, and E. Zimányi have mentioned some characteristics of time series data in their paper called “Time Series Database and InfluxDB”[2]. Following are some characteristics of time series data:

• **Scalability:** The volume of time-series data is rapidly increasing. Every hour, for example, the linked car will send 25 GB of data to the cloud. Regular databases are not built to handle this level of scalability. On the other hand, Time series databases are designed to account for scale by introducing functions that are only possible when time is prioritized. This can improve performance, such as higher insertion rates, faster queries at scale, and better data compression.

•**Usability:** TSDBs often incorporate time series data analysis capabilities and processes. They use data retention policies, continuous queries, flexible temporal aggregations, range queries, etc. As a result, this improves usability by improving the user experience while dealing with time-related analysis.

• **Data Compression:** Time Series data is mostly recorded per second or even less time, so there will be a lot of data to handle. A good data compression technique is required to handle this kind of data.

• **Data Location:** The queries on data are slow when the related data is not together. Time series databases co-relate the chunks of data of the same device and time range to access the data more efficiently.

• **Fast and Easy Range Queries:** As data location is there in TSDB, the range queries are very fast compared to traditional databases. The query language in TSDB is also like SQL, so it is easier for users to write queries.

• **High Write Performance:** In peak load, most databases cannot quickly serve the read and write request. TSDBs should ensure high availability and performance for both read and write operations during peak loads because they are usually designed to stay available in these conditions.

2.2.2. Uses and Benefits

•**Time Series Analysis:** Time series analysis means to analyze the variable which changes over time. Ex. The temperature of different cities.

•**Time Series Forecasting:** Time Series Forecasting means predicting future activity by using information regarding associated patterns and historical values. For example, Weather Forecasting, Earthquake Forecasting, Sales Forecasting, etc.

•**Regression Analysis:** Regression analysis means how the changes of any one variable affect another variable, like in the stock market, how one stock price changes affect any other stock.

2.2.3. Industry Use Cases

The Internet of Things revolution has enabled gadgets that were previously only available as offline systems to be connected to the internet. These devices are classified into two types: actuators, which operate on commands, and sensors, which sense the current environment and convert physical quantities into digital values. The values transmitted by these sensors and the standard analyses performed on them are a natural fit for time series databases [4]. A timestamp is assigned to each data point generated by a sensor. The application domain determines the frequency at which data is generated. The intervals used by sensors are usually every minute, every 10 minutes, or every hour. Getting the most recent data points, averaging data points over time periods, and flexible visualization are all common operations on sensor data. IoT data sets are typically regular and low in volume for small numbers of sensors.

Time series can be used in the analytics domain to track website visits, advertisement clicks, or E-commerce orders. The data volume may be affected by a variety of factors, such as time (e.g., orders on Tuesday versus orders on Sunday), weather (e.g., the number of air conditioners sold in a specific store), or other arbitrary circumstances (e.g., the number of cars per hour on a day with a taxi strike).

In the financial domain, time series data is commonly used. Time series can be used to illustrate stock prices, currency rates, and portfolio valuations. As a result, storing financial data points in a time series database is a sensible choice. For example, Kx Systems' kdb+, a time series database, is frequently used in high-frequency trading. Other financial use cases specifically presented by kdb+ include algorithmic trading, FX trading, and regulatory management. Financial time series are regular, although the volume varies dramatically. Data points can be generated every day, every minute (for example, stock closing prices), or every few seconds. (e.g., high-frequency trading).

Physical systems are frequently monitored using time series databases. Most time series databases feature spatial data storage and querying capabilities. It is possible to link location data in this manner. Asset tracking (e.g., storing the current position of all taxis) and geographical filtering (e.g., a number of taxis within a range) are examples of use cases. Asset tracking use cases produce data points with geospatial information. Time series produced can be regular (e.g., location is sent every minute) but is often irregular.

2.2.4. Some Databases and their Designs

TimescaleDB is an open-source time series database that is entirely based on PostgreSQL[15]. It inherits PostgreSQL's functionality and provides adaptive temporal chunks to reduce system resource usage. TimescaleDB offers full SQL support.

Druid is a columnar data storage that is open source and designed for OLAP queries [7]. It consists of many nodes that support a specific set of functionalities. Interactions between nodes have been reduced to a bare minimum.

InfluxDB is the most commonly used time series database. InfluxDB employs two protocols for data intake: a text-based protocol known as "Line Protocol" and a deprecated JSON protocol [9]. It uses its own query language, Flux. InfluxDB's storage engine is built on a time-structured merge tree (TSM) derived from a log-structured merge tree [13]. TSM significantly improves InfluxDB's performance when writing and reading data.

Cassandra is a NoSQL database that provides the always-on availability, fast read-write speed, and limitless linear scalability required by modern applications. Thousands of businesses trust Cassandra for its scalability and high availability without sacrificing performance. It is the ideal platform for mission-critical data because of its linear scalability and verified fault-tolerance on commodity hardware or cloud infrastructure [3].

2.3. Time Series Benchmarks

There are some existing time series benchmarks available in the market, which are examined in more detail in this section. The one common thing about these benchmarks is that they all use any random or generated dataset.

IoTDB Benchmark is built exclusively for TSDBs and IoT application scenarios. It focuses on specific data ingestion scenarios [11]. IotDB-benchmark has 10 types of queries, ranging from the exact point queries to time range queries with value filters. The performance metrics measured by the IoTDB benchmark are query latency, space consumption, system resource consumption, and throughput. The IoTDB benchmark's data generator generates random values within the provided range. Therefore, it is not using any real-world data. Many data generating factors, such as the data type of fields, the number of tags per device, and many other things, can be configured using the IoTDB-benchmark. The IoTDB benchmark currently supports IoTDB, InfluxDB, KariosDB, OpenTSDB, QuestDB, Sqlite, and TimescaleDB. This benchmark focuses solely on IoTDB, and not all functions are supported by databases other than IoTDB. For example, only IoTDB supports the generation and insertion of customized time series.

TSDBBench is a benchmark designed for comparing different TSDBs. In a project called YCSB-TS, it enhances the Yahoo! Cloud Serving Benchmark (YCSB) for use with time series databases [2]. In practice, the benchmark seems unmaintained. The documentation is outdated, and the required files are hosted on an inactive domain. The databases version used in this benchmark is also ancient. It only measures two metrics query latency and space consumption.

The TS-Benchmark is another TSDB benchmark centered on the requirements of managing huge time series data. It uses DCGAN based model to generate the high-quality synthesized data after being trained with real time series data [8]. For the benchmark, they used data from wind turbines. The number of wind farms, devices, and sensors utilized to generate data is viewed as a benchmark scale factor. TS-Benchmark can compare the performance of different TSDBs under various workloads such as data injection, data loading, and data fetching. It uses 6 queries with a different parameter to test the query latency of various TSDBs. The TS-Benchmark currently supports four types of TSDBs: InfluxDB, Druid, TimescaleDB, and OpenTSDB.

3. EXPERIMENTATION

Time series datasets can be categorized as synthetic data and real data. In the previous section, we have seen that all the benchmarks use synthetic data for comparison. It may represent the problem for the generalization of their results. This section describes a study focusing on real-world data. It also includes synthetic data to examine the performance differences. This section is structured as follows: Section 3.1 has the three performance metrics used to compare the databases. Section 3.2 represents the experimental setup of the experiment. Section 3.3 has details about the datasets we have used in the experiment. Section 3.4 represents all the types of quires used in the experiment.

3.1. Performance Metrics

Benchmarking requires defining what should be measured and how to evaluate the outcomes in order to create a ranking and distinguish between different results. A performance metric must be specified in order to do so. First, we used throughput to measure the data loading performance of individual databases. The time required to load the local dataset is called throughput.

Second, the space used to store a particular dataset is measured for every database. It will be interesting to see if there is any difference in space efficiency between TSDBs when storing the same data. If the space consumption of any database is low, it means that the database compresses the data at a higher rate.

Third, we used query latency to measure the time required to run the particular query. It measures how long a query takes to run from the time it is sent to the database until a result is delivered and received by the client. A query must be successfully and thoroughly completed. All eight types of queries will be executed and measured for each database.

3.2. Experimental Setup

When comparing different TSDBs, the results must be comparable if developed in the same environment. It is also essential that each TSDB executes the task under similar conditions, implying that the benchmark utilized does not give any TSDB an advantage or disadvantage. It is not expected that results from different situations (for example, measuring TSDB 1 on system A and TSDB 2 on system B) will be comparable. The databases are tested on Intel(R) Core(TM) i5-9500 CPU @ 3.00GHz, 8GB of RAM, 500GB of hard disk, and the operating system of Ubuntu 20.04.1 64-bit.

InfluxDB v2.1, TimescaleDB v2.6, Druid v0.22.1, and Casandra v3.11.12 were evaluated and compared in this experiment. InfluxDB's cluster version is not free, and TimescaleDB has no cluster version. We considered single-node versions to compare them fairly.

3.3. Datasets

All the TSDB benchmarks available in the market are using only one particular dataset. In the experiment, four different datasets of different domains are considered. Of course, four distinct datasets are insufficient to cover every industry or use case. However, analyzing the results of benchmarks using these workload datasets will allow comparisons to show whether the examined use case has an impact on performance. Out of these four datasets, three are real datasets from different industry domains like Financial, IoT, and Analysis. We have also taken one synthetic dataset into consideration to compare the performance of synthetic data and real data. All the real datasets are taken from Kaggle [10, 6, 5], and for synthetic data, we used the timeseries-generator library in python to generate the random time series data. Timeseries-Generator package is an exciting and excellent way to generate time-series data. A generator is a linear function with several factors and a noise function in this case. It is the data of some fast food item sales in different countries.

3.4. Queries

In this experiment, we have considered total eight kinds of queries. There are almost all kinds of the queries like exact point queries, queries with aggregate functions(min, max, sum, avg, count), group by queries, and queries with a time range.

3.4.1. Queries for Financial Dataset

1. Exact Point Query
    - Get the price of some stock on some particular day
2. Aggregation Queries
    - Get the total volume of a particular stock
    - Get the minimum price of the particular stock
    - Get the maximum price of the particular stock
    - Count the number of times when the particular stock volume is greater than 5000
    - Get the avg value of the stock price of a particular stock
3. Group By Query
    - Get the maximum volume group by stock
4. Time Range Query
    - Get the lowest and highest price of a particular stock from 2017 to 2019

3.4.2. Queries for Analytics Dataset

1. Exact Point Query
    - Get the amount paid on some particular taxi
2. Aggregation Queries
    - Get the minimum distance
    - Get the maximum distance
    - Get total distance travel
    - Get the total number of passengers who travels alone
    - Get the average tip amount
3. Group By Query
    - Get the total toll amount paid group by vendor
4. Time Range Query
    - Get the total amounts from 1 Jan 2015 to 25 Jan 2015

3.4.3. Queries for IoT Dataset

1. Exact Point Query
    - Get the turbidity on some particular day and time
2. Aggregation Queries
    - Get the date and time when the temperature is lowest at Ohio street beach
    - Get maximum turbidity at calumet beach
    - Get the total turbidity of montrose beach
    - Count when the temperature at calumet beach went below 19
    - Get the average wave height of Montrose Beach
3. Group By Query
    - Get the average battery life group by beach
4. Time Range Query
    - Get top 50 days in 2018 when the temperature is highest

3.4.4. Queries for Synthetic Dataset

1. Exact Point Query
    - Get the sale value of some particular day and time
2. Aggregation Queries
    - Get the minimum sale value of pizza
    - Get country name with maximum gdp factor

- Get the total value of sandwich
- Get the number of count when value is above 15000
- Get the average value of Netherlands

3. Group By Query
   - Get the average value group by product
4. Time Range Query
   - Get the value data from 2005 to 2015

4. Results

This section represents the results of the experiment. It is structured as follows: Section 4.1 represents the results of data loading, Section 4.2 represents the results of space consumption, and Section 4.3 represents the results of queries.

4.1. Data Loading

The local data files are imported using the different database's built-in import tool. Before data import, the databases are restarted to ensure that no cache is used. Table 1 shows the data loading performance for every dataset.

Table 1. Data loading (in seconds).

Database	IoT	Financial	Analytics	Synthetic
TimescaleDB	8	949	604	44
Druid	6	695	303	106
InfluxDB	2	576	283	19
Cassandra	2.23	1068	388	24.993

Based on the experiment results, we can say that InfluxDB beats all of the datasets in terms of load performance. After InfluxDB, Druid is the best option. Druid requires permanent backup storage for the distributed file system, known as deep storage. Because each segment is repeated in deep storage, the time required to load the data increases. It outperforms TimescaleDB and Cassandra in real datasets, but it is the worst performer in synthetic datasets. So, here you can see the performance difference of Druid in real and synthetic data. Cassandra beats TimescaleDB in all datasets except the financial dataset. We can say from the results that InfluxDB is the best choice, followed by Druid, Cassandra, and TimescaleDB. If we compare the best and the worst performer, InfluxDB outperforms TimescaleDB in the IoT dataset by 4 times, the financial dataset by 1.5 times, the analysis dataset by nearly 3 times, and the synthetic dataset by almost 2 times.

4.2. Space Consumption

All four datasets are inserted in particular databases in CSV format. Afterward, the space occupied by the data for the specific database is measured. Table 2 shows the data loading performance for every dataset. In terms of space consumption, Druid outperforms all other databases by a wide margin. It took only 25\% of the original data, which is exceptional. It compresses data to a high degree using data granularity. The level of detail in a data structure is measured by data granularity. The granularity of measurement of time-series data, for example, could be based on intervals of years, months, weeks, days, or hours.

Table 2. Space consumption (in MB).

Database	IoT	Financial	Analytics	Synthetic
CSV	3.9	3844.83	1985.96	130.9
TimescaleDB	9.7	5222.4	2355.2	179.4
Druid	6.14	946.43	489.76	56.15
InfluxDB	3.6	3276.8	1536	126
Cassandra	3.9	4526.08	1433.6	94.51

InfluxDB is the next option after Druid. InfluxDB uses many compression techniques for various data types. It also takes up less space than the original file, but it is more than Druid. Cassandra also takes less space than the original in all the datasets. However, Cassandra takes up more space in the financial dataset than the original. So, as data size grows, Cassandra's performance decreases. In terms of space utilization, TimescaleDB is the worst performer. Because TimescaleDB is built on Postgres, storing data in tabular format takes much space. If we compare the best and the worst performer, Druid outperforms TimescaleDB in the IoT dataset by 1.5 times, the financial dataset by 5 times, the analysis dataset by nearly 3 times, and the synthetic dataset by almost 5 times.

4.3. Query Latency

The results of queries are divided into four groups. It is structured as follows: Section 4.3.1 represents exact point query, Section 4.3.2 represents aggregation queries, Section 4.3.3 represents group by query, and Section 4.3.4. represents time range query.

4.3.1. Exact Point Query

When we performed the exact point query on all databases, TimescaleDB performed better than all other databases. This is because TimescaleDB is based on a relational database, PostgreSQL, and uses time as the primary key, which is specially indexed. The results are shown in Figure 1. In the financial domain, Druid beats TimescaleDB because that dataset is the largest. After TimescaleDB, Druid is the second-best option, followed by InfluxDB and Cassandra. TimescaleDB outperforms Cassandra in the IoT query workload by 3 times, the financial query workload by 14 times, the analysis query workload by nearly 9 times, and the synthetic query workload by nearly 4 times. Clearly, the query workload has a significant impact on performance.

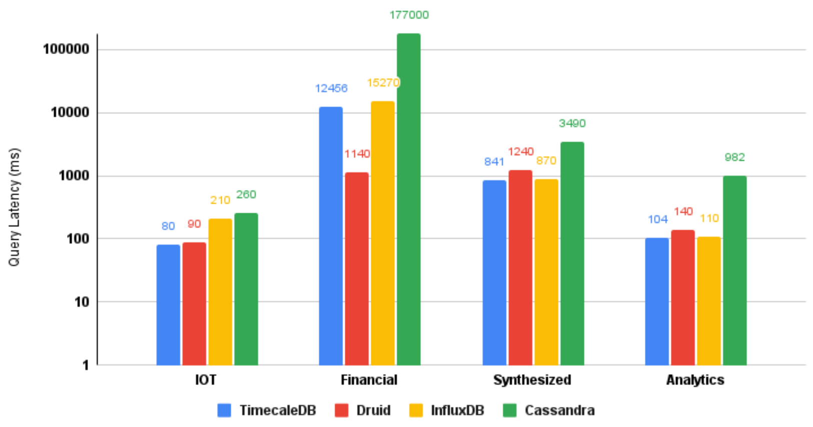


Figure 1. Exact point query

4.3.2. Aggregation Queries

We have used the aggregation functions MIN, MAX, COUNT, SUM, and AVG in aggregate queries in the experiment. The results are pretty similar for all aggregate functions. The results of MIN, MAX, COUNT, SUM, and AVG are shown in Figure 2, 3, 4, 5, and 6, respectively. Except for the financial dataset, the results show that InfluxDB outperforms all other databases in every aggregate function query. InfluxDB beats the weakest performer, TimescaleDB, by around 300 times, which is impressive. In terms of query latency performance, Druid is pretty close to InfluxDB. Cassandra is a NoSQL database. However, it outperformed TimescaleDb, which is specifically designed to handle time series data. For the IoT query workload, InfluxDB performs 5 times better than TimescaleDB, for the financial query workload 56 times better, for the analysis query workload nearly 4 times better, and for synthetic data almost 18.5 times better.

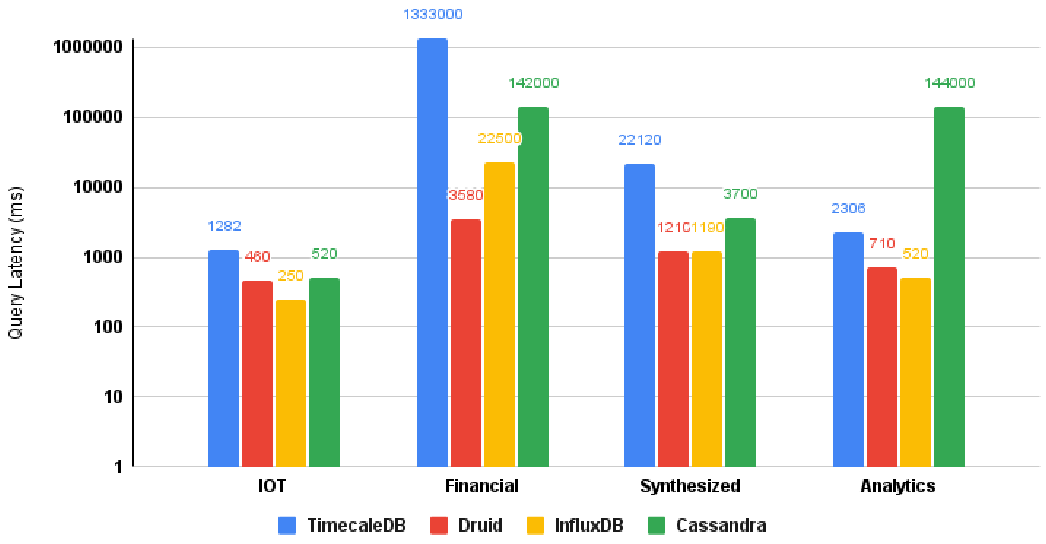


Figure 2. MIN Aggregation query

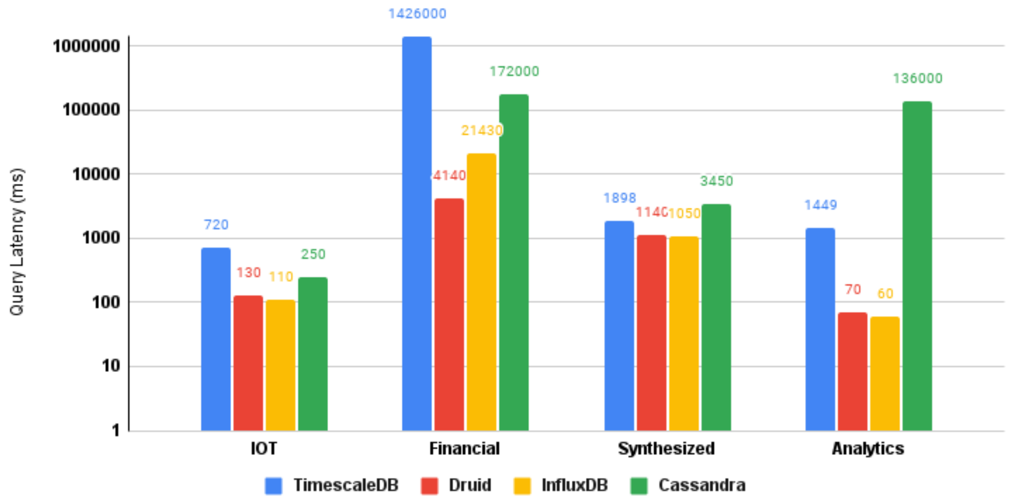


Figure 3. MAX Aggregation query

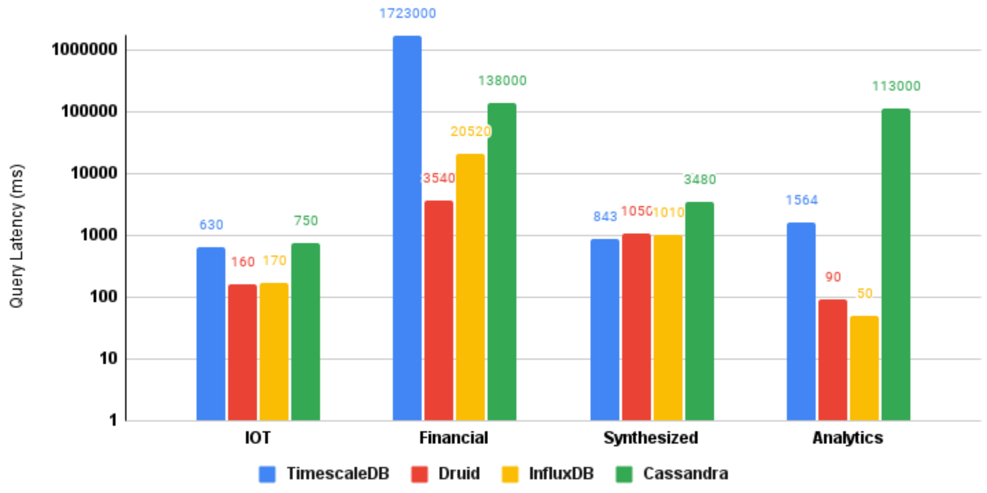


Figure 4. COUNT Aggregation query

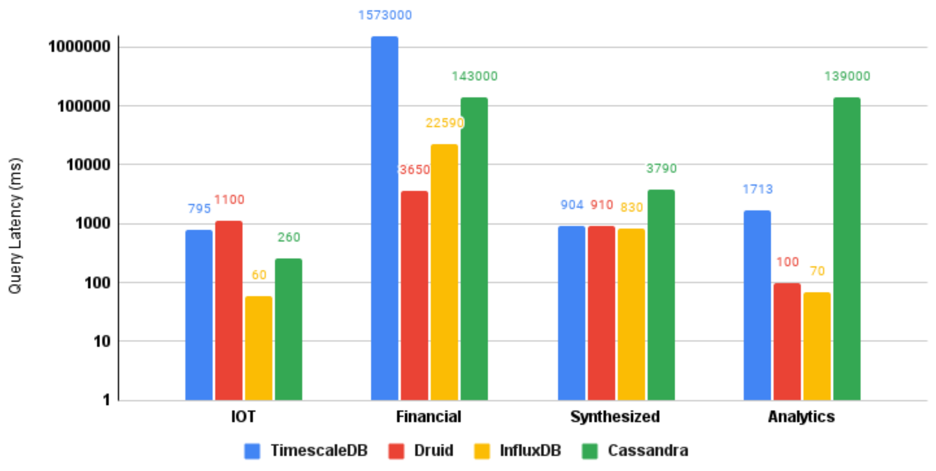


Figure 5. SUM Aggregation query

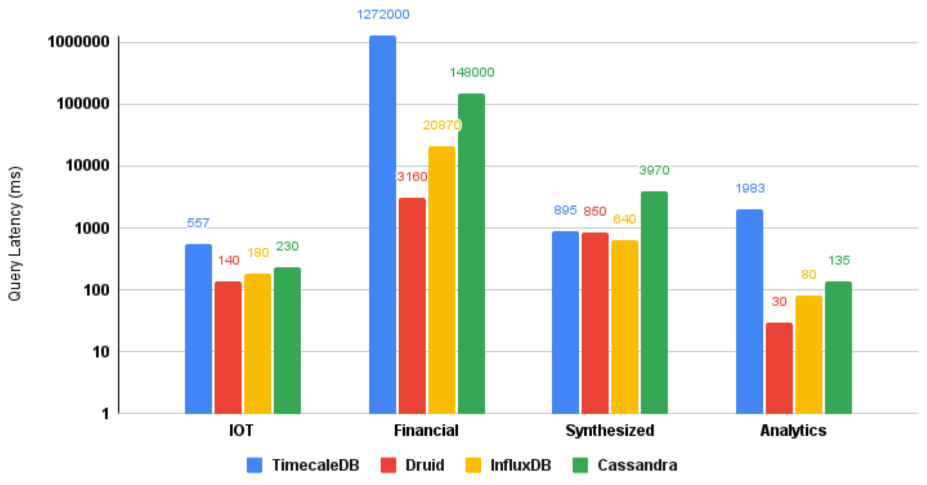


Figure 6. AVG Aggregation query

4.3.3. Group by Query

The most common query type in the industries is a group by queries. we ran group by queries on various datasets across all databases. The results are shown in Figure 7. Druid and InfluxDB performed equally well across all datasets in this query type. Druid outperforms InfluxDB by a hair, but it's a razor's edge. After Druid and InfluxDB, Cassandra outperforms TimescaleDB in the IoT and financial domains, whereas TimescaleDB outperforms Cassandra in the analytics domain.

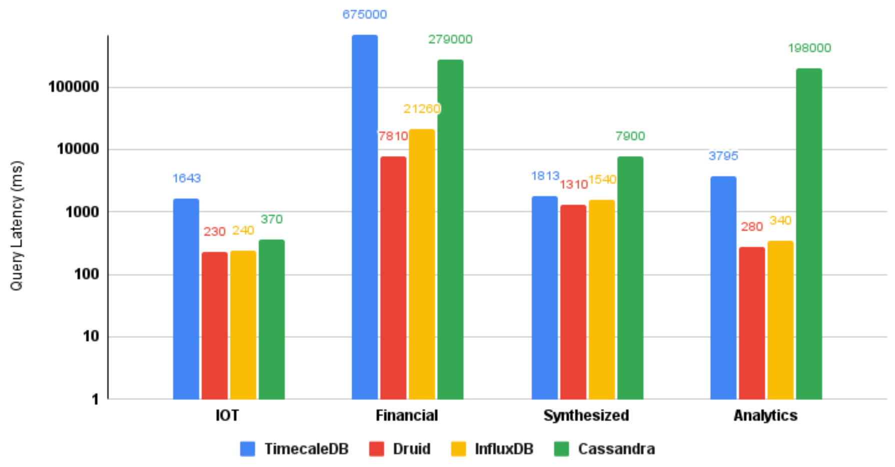


Figure 7. Group by query

4.3.4. Time Range Query

The time range query is the essential query type of time series data. It is used to retrieve data from a specific time range. The results of all the databases are shown in Figure 8. In the time range query workload, we can see that Druid outperforms all other databases. Druid is just ahead of InfluxDB. In this query type also, Cassandra and TimescaleDb are the worst performers. Cassandra performs well in IoT and financial scenarios, and in other cases, TimescalDB performs better.

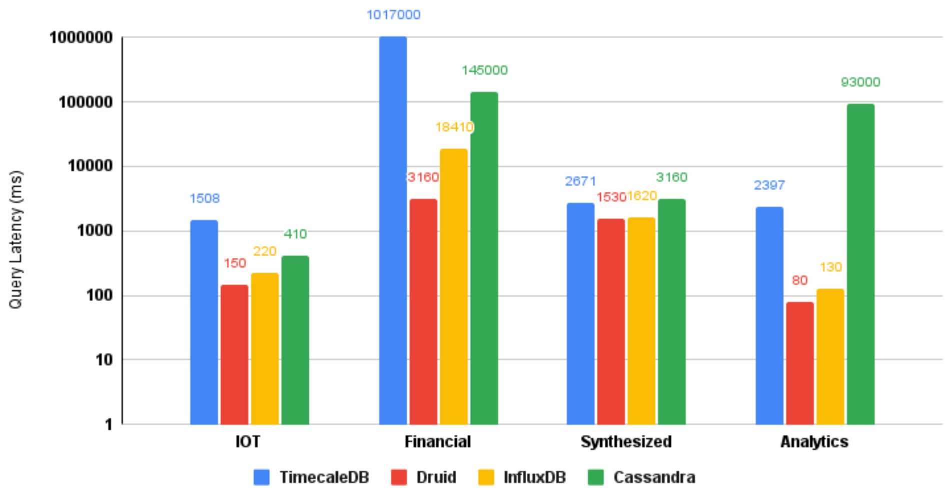


Figure 8. Time range query

5. Conclusion

In this experiment, the performance of TSDBs is evaluated by several real-world scenarios such as the IoT domain, the Financial domain, and the Analytics domain. We have also considered the synthesized data for comparison. We develop performance metrics that are more suited to evaluate the effectiveness of TSDBs. It includes data loading, database space usage, and query performance. We ran experiments and examined the findings on three common TSDBs, TimescaleDB, Druid, InfluxDB, and one NoSQL database, Cassandra. InfluxDB is a native TSDB with a storage engine that uses a Time-Structured Merge Tree (based on log-structured merge tree). TimescaleDB, based on PostgreSQL, uses row-wise storage to represent databases. Druid saves data in columnar format and creates indexes when data is injected. Cassandra is the most popular column-oriented NoSQL database.

From the experiment results, we have obtained some preliminary yet lightening conclusions: I) Because of its unique storage mechanism, namely a time-structured merge tree that supports high-speed writing, InfluxDB has a high injection and query throughput. II) During pre-processing, a granularity component in Druid separates data into segments and generates a bitmap index. Druid stream processing also struggles with excessive concurrency. Columnar storage format and indexes, on the other hand, pay off later. Druid achieves high query throughput, sometimes beating InfluxDB. III) Because TimescaleDB uses a row-wise storage format borrowed from PostgreSQL, it has poor query speed. The reason for low performance is that TimescaleDB does not automatically create indexes. IV) Cassandra is the NoSQL database that uses column format to store data. It gives better read and write performance than TimescaleDB.

We have also used one synthetic dataset with the other real datasets. The reason to use a synthetic dataset is to see if we can find any interesting results. There is a lot of difference between the synthesized and real datasets in data loading. Druid performs better in the real datasets, but it is the worst performer in synthesized data. The financial dataset also has different query performance and space consumption results than the other datasets. The recommendations are based on the performance comparison: I) If the focus lies on load performance, InfluxDB is the best choice, followed by Druid, Cassandra, and TimescaleDB. II) Druid is far better than others if the focus lies on the lowest space consumption. InfluxDB is the second-best choice. III) If there are more aggregate queries, one can use InfluxDB. It performs very well, followed by Druid and Cassandra. If there are queries containing data filters like time range, Druid is the best choice.

References

[1] Engines ranking. https://db-engines.com/en/ranking/time+series+dbms.
[2] A. Bader, O. Kopp, and M. Falkenthal. Survey and comparison of open source time series databases. Datenbanksysteme für Business, Technologie und Web (BTW 2017)-Workshopband, 2017
[3] Cassandra. Open source nosql database. https://cassandra.apache.org/_/index.html.
[4] S. Di Martino, L. Fiadone, A. Peron, A. Riccabone, and V. N. Vitale. Industrial internet of things: persistence for time series with nosql databases. In 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pages 340–345. IEEE, 2019.
[5] C. P. District. Beach water quality - automated sensors: City of chicago: Data portal. https://data.cityofchicago.org/Parks-Recreation/Beach-Water-Quality-Automated-Sensors/qmqz-2xku
[6] Elemento. Nyc yellow taxi trip data. https://www.kaggle.com/elemento/nyc-yellow-taxi-trip-data.
[7] A. S. Foundation. Database for modern analytics applications. https://druid.apache.org/
[8] Y. Hao, X. Qin, Y. Chen, Y. Li, X. Sun, Y. Tao, X. Zhang, and X. Du. Ts-benchmark: A benchmark for time series databases. In 2021 IEEE 37th International Conference on Data Engineering (ICDE), pages 588–599. IEEE, 2021.

[9] InfluxDB. Open source time series database. https://www.influxdata.com/.
[10] H. Kumar. Stock market india. https://www.kaggle.com/hk7797/
[11] R. Liu and J. Yuan. Benchmarking time series databases with iotdb-benchmark for iot scenarios. arXiv preprint arXiv:1901.08304, 2019.
[12] S. N. Z. Naqvi, S. Yfantidou, and E. Zimányi. Time series databases and influxdb. Studienarbeit, Université Libre de Bruxelles, 12, 2017.
[13] P. O'Neil, E. Cheng, D. Gawlick, and E. O'Neil. The log-structured merge-tree (lsm-tree). Acta Informatica, 33(4):351–385, 1996.
[14] D. Ted and F. Ellen. Time series databases: New ways to store and access data.
[15] TimescaleDB. Timeseries database for postgresql. https://docs.timescale.com/

DEVELOPMENT OF COMMUNICATING STREAM X-MACHINE TOOL FOR MODELING AND GENERATING TEST CASES FOR AUTOMATED TELLER MACHINE

Bashir Adewale Sanusi[1], Emmanuel Ogunshile[1], Mehmet Aydin[1],
Stephen Olatunde Olabiyisi[2] and Mayowa Oyedepo Oyediran[3]

[1]Department of Computer Science and Creative Technologies,
University of the West of England, Bristol, United Kingdom
[2]Department of Computer Science,
Ladoke Akintola University of Technology, Ogbomoso, Nigeria
[3]Department of Computer Sciences, Ajayi Crowther University, Oyo, Nigeria

ABSTRACT

The improvement of this paper takes advantage of the existing formal method called Stream X-Machine by optimizing the theory and applying it to practice in a large-scale system. This optimized formal approach called Communicating Stream X-Machine (CSXM) applied in software testing based on its formal specifications to a distributed system as it points out its advantages and limits of the use of the existing formal methods to this level. However, despite the tremendous works that has been done in the software testing research area, the origin of bugs or defects in a software is still cost and takes more time to detect. Therefore, this paper has proven that the current state of art challenge is due to that lack of a formal specification of what exactly a software system is supposed to do. In this paper, CSXM principles was used for the development of Automated Teller Machine (ATM) given formal specification which outputs conforms with the implementation. Moreso, the computational strength of Remote Method Invocation (RMI) network interface in Java programming was used to provide communication between the stand-alone systems i.e., the client (ATM) and server (Bank) in the context of this paper. The results of this paper have been proven and helps software developers and researchers takes early action on bugs or defects discovered by software testing.

KEYWORDS

Formal Method, Software Testing; Stream X-machine; Communicating Stream X-Machine; Software Testing; Distributed System; Formal Specification; Defects; Automated Teller Machine; Remote Method Invocation; Java Programming Language.

1. INTRODUCTION

In computational term, testing attempts to achieve correctness by detecting all the faults that are present in an implementation, for the errors to be removed. Nevertheless, software defect is referred to as a flaw, fault or failure in a computer system or program which gives an unexpected or incorrect result [1]. It gives either an incorrect, or unexpected result, and behaves in unintended ways. The unexpected result is identified during software testing and marked as a defect [1]. Stream X-Machine (SXM) testing methodology is a complete functional testing approach to software and hardware testing that exploits the scalability of the SXM model of computation [2]. SXM testing method provides repeatable and strong guarantees of functional correctness, up to a specification [3]. To model systems composed of communicating agents and introduction of stand-alone SXM models, communication is suitable by exchanging messages between components processing functions [4]. In addition, the design of a Communicating X-machine system (CXM) is referred to as the graph whose nodes are the components and edges are simply the communication channels among all. Sequel to the issue of integrating a society of X-machine into a communicating system for the goal of building large-scale software systems which fulfil their requirements, several classes of CXM models have been presented [5]. However, CXM is a formal model which ease a regimented development of large-scale systems. Nevertheless, the optimization of the existing formal method in use called SXM has leads to theory of the Communicating Stream X-Machine (CSXM) which has been proven and best suited the specification of a distributed system. With the ever-expanding areas of applications today including embedded and real-time systems, safety-critical systems, service-oriented architectures and so on, it is often easy to lose sight of the essential similarities that exist among all of these systems using formal methods. But the critical issue is the relationship between the proposed solution and the understanding of the problem's originator as to whether the proposed solution does, in fact provide the desired answer. Therefore, in this paper the strength of the CSXM was used to specify a distributed system called Automated Teller Machine (ATM) which was correctly implemented and satisfies the challenge of software testing.

2. RELATED WORK

[6], transformed X-machine specification written in X-Machine Language (XML) and which is automatically converted into an executable java code. This research was unable to write test cases, and it is limited to the theory of standalone X-machine components. [7], reports on the development and formal verification of CompCert, i.e., a compiler from Clight which is referred to as the subset of the C programming language to PowerPC assembly code. This research used the Coq proof assistant both for programming the compiler and for proving its correctness. Also, it stated that the verified compiler is useful in the context of the critical software and its formal verification.

[8], center the research on providing tool support for business-level, example-based specifications which are mapped to the browser level for automatic verification. This research allows refactoring support for the evolution of existing browser-level tests into business-level specifications, and it provides feedback on coverage as the resulting business rule tables may be incomplete, contradictory, or redundant. [9], stated formal verification are used for exhaustive investigation of the system space which ensures that undetected failures in the behavior are excluded. The research constructs the system incrementally from subcomponents, based on the software architecture. According to this study, developing a safe multi-agent robotic system to ensure the correctness properties of safety and liveness was a challenge. In conclusion, the development approach allows for formal verification during the specification definition.

According to [10], the software testing may consume 35% to 40% of a software development budget. However, the manual and automated testing methods seems complementary to each other. In this study, it is stated that the purpose of testing is to find out defects or bugs, the causes, and approaches in which this fault can be fixed as early as possible. It is therefore stated that the testing requires more project effort and time than any other software development activity which needs a suitable strategy to make the testing successful.

[11], analyzes three states of the art formally verified implementations of distributed systems which includes Iron-Fleet, Verdi, and Chapar. This research sees through the code review, testing and found a total of sixteen bugs in which some produces serious consequences, including

crashing servers, returning incorrect results to clients, and invalidating verification guarantees. Therefore, this research develops a testing toolkit called PK, which focuses on testing these parts and is able to automate the detection of thirteen out of sixteen bugs.

[12], share their experience and discuss the open problems or challenges that the software testing, verification, and compiler development presents. Although, the aim was to deliberate on new ideas on how to approach these problems. It is concluded that there is no universal appropriate name for this field. However, the compiler explains most of the work in this field and is not difficult to explain to people but has the barrier of excluding some subjects such as debuggers. Due to this reason, the research decided to keep the name compiler testing and verification for future research works.

[13], developed an automatic Java X-Machine testing tool for software development. This research focuses on addressing the software complexity and changing the software developers' expectations with the motive of reducing the amount or cost in detecting defects in software systems. It was concluded that this research could not generate test cases automatically.

[14], stated that for a compiler correctness theorem to assure complete trust, then such theorem must reflect the reality of how the compiler will be used. Although, the variation of theorems, stated in remarkably different ways, develops questions about what researchers meant by a compiler is correct. Therefore, this study developed a framework with the idea to understand compiler correctness theorems in the presence of linking and applying it to understanding and comparing this diversity of results. Hence, this research did not only focus on the strengths and weaknesses but also gain insight into what should be expected from compiler correctness theorems of the future research.

[1], conducted experiments on publicly available bug prediction dataset that is a repository for most open-source software. This research used Genetic algorithm to extract relevant data features from the acquired dataset and machine learning algorithms was used to predict defect in software system. [15], proposed an enhanced fault-detection W method for increasing software reliability in safety-critical embedded systems. The research stated the testing time of the proposed method takes much time than the W method during the software testing.

[16], stated software defect prediction is one of the most encouraging exercises of the testing phase of the software development life cycle. In this case this research created a framework to anticipate the modules that deformity inclined the software quality. Nonetheless, GA was used to extract the relevant features from publicly available data sets to eliminate the possibility of overfitting and the relevant features were classified into defective or non-defective. In conclusion, the outcome indicated that ECLIPSE JDT CORE, ECLIPSE PDE UI, EQUINOX FRAMEWORK and LUCENE has the accuracy, precision, recall and the f-score of 86.93, 53.49, 79.31 and 63.89% respectively, 83.28, 31.91, 45.45 and 37.50% respectively, 83.43, 57.69, 45.45 and 50.84% respectively and 91.30, 33.33, 50.00 and 40.00% respectively.

[17], study and evaluate a narrative verification approach based on Bounded Model Checking (BMC) and Satisfiability Modulo Theories (SMT) to verify C++ source codes. This research verification approach analyses bounded C++ source codes by encoding into SMT various sophisticated features that the C++ programming language offers which includes inheritance, templates, polymorphism, exception handling, and the standard template libraries. Nevertheless, the research compares Efficient SMT BMC (ESBMC) to The Low-Level BMC (LLBMC) and DIVINE, which are the state-of-the-art verifiers to check C++ source code directly from the LLVM bitcode. It is concluded in this research that ESBMC can handle a wide range of C++ source codes, presenting a higher number of correct verification results.

Therefore, in this paper, having reviewed some existing papers on the previous methods that researchers have applied in the software testing stage of the software development lifecycle, but this area of research remains researchable and continuous research area. Formal methods as being proven lately by researchers in specifying how a system should work. Hence, the SXM was improved on by using the Communicating Stream X-Machine (CSXM) theory to prove that the specification conforms with the implementation of the said case study and will be used in future research to assure the correctness, testing, and verification of a compiler design by integrating the strength and computational use of machine learning algorithms so as to construct test programs which will be used to determine whether a compiler behaves correctly.

3. METHODOLOGY

As the Software Development Life Cycle (SDLC) is said to be the most used and oldest formalized framework for constructing distributed systems [18]. The strength of this methodology framework was used to build the large-scale distributed system (ATM) in this paper. There are different models in SDLC but the choice of waterfall model in this paper was its strength of step-by-step flow in respect to its importance or an organized process that makes sure every stage is followed carefully before moving on to the next steps. As a result of this, the waterfall model was adopted in this paper to divide the software development work into sequential steps and smaller process to optimize the design and specification of the SXM. The strength of this methodology has helped this paper to build and deliver a correct and stable system. It also creates a clear understanding of the task ahead as stated in the last paragraph of section 2 of this paper which enables a better estimate and identify errors earlier i.e., the specifications conforming with the implementation. Nevertheless, advancement in computing capabilities puts higher demand on the software system and the developers. This raises the issues of cost, delivery of faster software, and conforming to the needs of the end users. Therefore, SDLC strength was used in this paper to measure the correctness, testing which identify inefficiencies and verifying them in the development process which helps in fixing the errors and run smoothly. Therefore, the case study ATM was designed and implemented using the CSXM formal theory written in Java programming language to test and proof the correctness of the developed system.

3.1. Steps followed in this Paper

The following are the stages that was adopted in this case study. Figure 1 represents the diagrammatic design of the developed system that was used in this paper.

- Requirement Gathering and Analysis: this phase is the definition of the requirements considered in planning which established what the software system is supposed to do as specified and its requirements. In this paper, ATM was the software constructed which required the ability to make transactions such as deposit, withdrawal e.t.c. However, this requirement includes the definition of the resources used to build this software which is use of CSXM specification to develop the software system. Hence, CSXM theory is the requirement used in this process.
- System Design: This phase models the functioning of the software system i.e., how the system is working. Also, the CSXM specification was written in Java Programming language which defined the way end users interacted with the software system and how the application responded to the end users' input. In this paper, the system was developed to run on MS Windows platform due to the methods used in solving the problems in the case study specified. With respect to the strength of Java programming language, the

Remote Method Invocation (RMI) was the method defined in the software system which enables the communication between the SXMs.

- Software Development: This phase is the system coding which is used as the access control in the software system. In this paper, this stage was used to track changes to the code, helps to verify the system by ensuring the specification conforms with this implementation. Nevertheless, finding, and fixing errors is an area generating interest in the field of software engineering which cannot be left out in the development process such as generating test cases or compilation of the code to ensure the software system runs as specified. This paper utilizes the computational power of the formal method (CSXM) which leads to the next stage of this life cycle.
- Software Testing: In SDLC, testing stage is a critical phase of the development process of a software system. Nonetheless, the choice of the formal method (CSXM) used in this paper was to explore one of the strengths which involves testing the correctness of the software system. In this paper, as the entire system was tested for any defects or bugs and failures so also verified by ensuring the specification conforms with the implementation.
- Maintenance: Over the decades, the cost and time of software testing continues to be a researchable area among researchers and everyday concern among software developers. At this phase, the software system is completed and being used by the end users. However, the end users may or may not discover defects that was not found during the testing stage which makes this stage an important phase in the SDLC. Therefore, the bugs found during this process are resolved and generate the new development cycles.

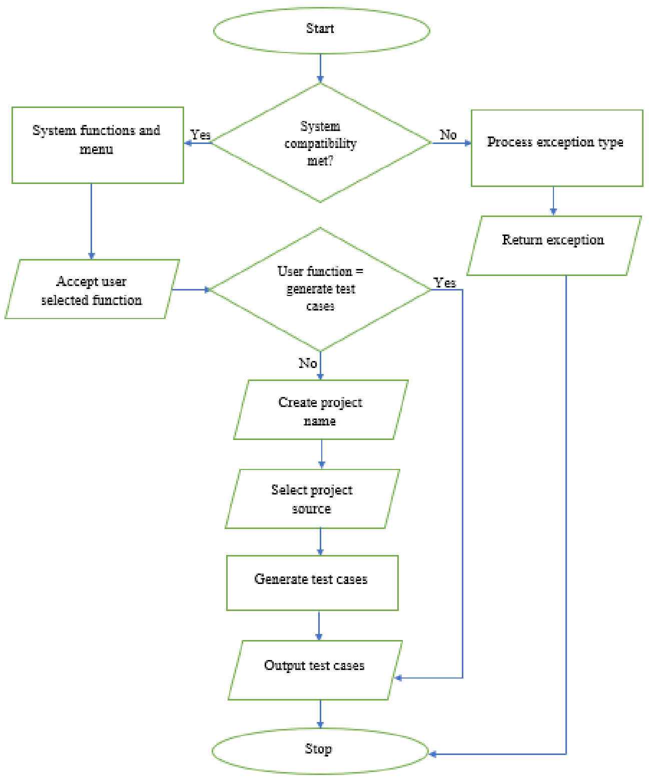


Figure 1. The Flowchart of the Developed System

3.2. Stream X-machine

In particular, the benefit of this model is that it permits a system to be driven, with extra care, through its states and transitions while noticing the results at each progression. These are observer esteems that ensure that specific capacities were executed on each progression [4]. Because of this explanation, complex software systems might be deteriorated into a pecking order of SXM, planned in a hierarchical way and tried in a base up manner. However, SXM is an eight tuple which are as follows;

$$Machine = (Q, \sum, \Gamma, M, \Phi, q_o, F, m_o) \quad (1)$$

Where;
Q: is the finite set of states
$\sum$: is the finite set of input symbols
Γ: is the finite set of output symbols
M: is a (possibly infinite) set called memory
Φ: is the finite set of distinct processing functions; a processing function is a non-empty (partial) function of type $\mathrm{M} \times \sum \rightarrow \Gamma \times \mathrm{M}$
$q_o \in Q$: is the Initial state
F: is the (partial) next-state function, $\mathrm{F}: \mathrm{Q} \times \Phi \rightarrow Q$
$m_o \in M$: is the initial memory

Beginning from the initial state q_o alongside the initial memory m_o, an input symbol $\sigma \in \sum$ activate a function $\varphi \in \Phi$ which in turn generate a transition to a new state $q \in Q$ and a new memory state $m \in M$. However, sequence of transitions generated by the stream of input symbols is called the computation. Therefore, the output of a computation is the sequence of results generated by the sequence of transitions.

3.3. Communicating Stream X-machine

In this paper, the CSXM systems model was adopted, and it is further reviewed below. Therefore, CSXM with n x-machine components are a triplet tuple [5];

$$Machine_n = (R, MAT, C^0) \quad (2)$$

Where;

i. R is the set of $n = |R|$ x-machine components of the system of the form $V_i = (\Lambda_i, IN_i, OUT_i, in_i^0, out_i^0) \forall\, 1 \leq i \leq n$. Such x-machine components of the system are called the Communicating X-Machine (CXM). Λ_i in the definition stated above refers to a SXM with memory M_i. The IN_i and OUT_i directly correspond to the values that can be communicated by input and output ports of the ith CSXM such that $IN_i, OUT_i \subseteq M_i \cup \{\lambda\}$ and $\lambda \notin M_i$. The symbol λ is simply used to indicate that a port is empty and the initial values of the x-machine ports are set to in_i^0 and out_i^0.

ii. MAT is simply referred to as the set of matrices of order $n \times n$ to form the values of the matrix variable which supposed to be used for creating communication amongst the x-machine components. Therefore, for any $C \in MAT$ and any pair of x-machine such that i, j are the data value stored in $C|i,j|$ shows at most one message that is being processed from the memory M_i of x-machine $V_i \in R$ to the memory M_j of x-machine $V_j \in R$. Thus, each element of the matrix $C|i,j|$ can be considered as a temporary buffer variable where the property $IN_i \subseteq M_i \subseteq OUT_j$ holds.

iii. Basically, all messages that are sent from the CXM V_i that is x-machine Λ_i and V_j which is x-machine Λ_j are data values from the memory M_i and M_j respectively. The λ symbol in the matrices is used to specify that there is no message, where the @ symbol is simply used for specifying a channel that is not going to be used (an x-machine communicating with itself disallowed). The individual elements of the matrices are derived from the machine memory $M \cup \{\lambda, @\}$, where;

$$M = \bigcup_{i=1}^{machine} M_i \; and \; \lambda, @ \notin M \tag{3}$$

iv. C^0 simply defines the initial communication matrix as $C^0|i,j| = \lambda$ assuming a valid communication between the x-machine V_i that is x-machine Λ_i and V_j which is x-machine Λ_j is allowed; otherwise, the initial matrix is defined as $C^0|i,j| = @$ so as to indicate that the communication between the two x-machines $i \; and \; j$ are disallowed. Nevertheless, the matrix $C^0|i,j| = @$ shows that an x-machine communicating with itself is definitely not allowed.

v. Generally, the ith CXM component can only read from the ith column and then write to the ith row of the communication matrix.

Also, for any $C \in MAT$, any value $x \in M$ and any pair of indices $1 \leq i,j \leq n$, with $i \neq j$.

i. If $C|i,j| = \lambda$ an output variant of C, denoted by $C_{ij} \Leftarrow x$ is defined as;

$$(C_{ij} \Leftarrow x)[i,j] = x \tag{4}$$
$$(C_{ij} \Leftarrow x)[k,m] = C[k,m] \; \forall \; (k,m) \neq (i,j) \tag{5}$$

ii. If $C|i,j| = x$ an input variant of C, denoted by $\Leftarrow C_{ij}$ is defined as;

$$(\Leftarrow C_{ij})[i,j] = \lambda \tag{6}$$
$$(\Leftarrow C_{ij})[k,m] = C[k,m] \; \forall \; (k,m) \neq (i,j) \tag{7}$$

As stated in equation (4), (5), (6), and (7) above, simply denotes several acceptable transitions from one matrix to another. Generally, CXM is a five tuple;

$$V = (\Lambda, IN, OUT, in^0, out^0) \tag{8}$$

Recall, SXM in equation (1) $Machine = (Q, \sum, \Gamma, M, \Phi, q_o, F, m_o)$ which is equivalent to Λ and stated below;

$$\Lambda = (Q, \sum, \Gamma, M, \Phi, q_o, F, m_o) \tag{9}$$

Where;

i. IN and OUT has been defined within equation (2) above
ii. $\sum$ and Γ are the finite set of input and output symbols respectively
iii. Q is the finite set of states of each x-machine component gathered into a communicating system must be partitioned as $Q = Q^{I} \cup Q^{II}$ where Q^{I} corresponds to the processing states in each x-machine component in the communicating system and Q^{II} is the set of

communicating states corresponding to the central medium where all the n x-machine components have been integrated and where $Q' \cap Q'' = \emptyset$ holds. Therefore, this implies that for each $q' \in Q'$ in each x-machine component, the functions emerging from q' are processing functions. For instance, in the state q' different functions can be triggered, in this scenario one of them is arbitrarily chosen or if no function can be applied the entire CXM system halts. However, if the machine is in state $q'' \in Q''$ then all the functions emerging from state q'' are communicating functions. Otherwise, while the machine is in state q'', if different functions can be applied then one of them is arbitrarily chosen, else if this is not the case then the machine simply does not change it current state and would have to wait until one of such functions can be applied.

iv. M a set or possibly infinite set called the memory

v. The type of the machine is defined as a set $\Phi = \Phi' \cup \Phi''$ where Φ' is called the set of processing function and Φ'' is the set of communicating functions and $\Phi' \cap \Phi'' = \emptyset$. Notably, each element $\phi' \in \Phi'$ is a relation (partial function) of the type below;

$$\phi' : IN \times M \times OUT \times \Sigma^* \rightarrow \Gamma^* \times IN \times M \times OUT \qquad (10)$$

4. Results and Discussion

In this paper, to assure the correctness, testing, and verification of the formal method used called the CSXM, a software system was developed called Automated Teller Machine (ATM) as it is one of the good and reliable examples of a distributed system. The ATM called client in this in this paper is the combination of computer terminal which communicates with one another to a bank’s central computers also referred to as server in the context of this paper. The software system specifications in this paper enables end users to perform transactions such as login (security check), deposit, withdraw, inquiry, statement, and account details.

4.1. Developed System

The understanding of SXM and CSXM from theory to practice was used in a real-world scenario to proof the correctness and to test whether the ATM specification conforms with the developed implementation as constructed in this paper. However, as stated in equation (8) which includes the SXM, input, and output port. The SXM specified in this paper represents the overall specification of the said case study (ATM) as shown in figure 2 below. The figure 2 represents the state diagram of the overall case study (ATM). In formal methods, the said theory can be represented either by mathematical notation as stated in section 3.1 above or by state diagram which is the diagrammatical representation of the specified mathematical notations. From figure 2, the states include start, ATM_OutofSource, Insert_Card, and Continue_Transaction, where start is the initial state. Also, the transitions are ATM_InUse, Card_Insert, ATM_Not_Available, and Transaction_Aborted. The transition functions enable the states to move from one to another. Nevertheless, the strength of memory included in this model enables each customer stored information to be assessed when their information is being verified from the server. In this paper, the distributed banking system consist of some ATM referred to as client and a server called the bank which communicates via Java Remote Method Invocation (RMI). The server controls all users account information where an end user can use the following operations at an ATM.

i. void deposit (int acnt, int amt): this simply referred to when the operation increases the balance of end user account acnt by amt and returns nothing.

ii. withdraw (int acnt, int amt): this simply referred to when the operation decreases the balance of end user account acnt by amt and returns nothing.

iii. Float inquiry (int acnt): this referred to when the operation returns the balance of the end user account acnt.

iv. GetStatement (Date from, Date to): this also referred to when the operation returns a statement object of the transaction history of the said account acnt.

Nevertheless, the client (ATM) was used to initiate an end user operation by calling a remote method on the bank server to execute the specified procedure such as withdraw with its parameters as presented in Fig. 3 and Fig. 4 represents server (bank) authentication by validating the end users' card and pin before a transaction can be performed. Figures 3 and 4 are specified as two different SXM which are communicated with the RMI and give the representation of the communicating stream x machine (CSXM) as shown in Figure 5 below. In these figures the states are represented in oval shapes while the transitions are represented in arrows.

Furthermore, the concept of SXM and CSXM are discussed in section 3 of this paper. However, this paper is incomplete without giving details about the RMI. Java programming language has the strength which enables software developers write large scale object-oriented system of this manner where objects on different computers i.e., SXMs can interact in a distributed network. Regardless, RMI was used in this paper because it has the ability to pass one or more objects along with the request which suites the aim of this paper. This object includes information that change the service that is performed in the remote computer.

Moreso, in this paper when a user at the client (ATM) fills out an expense account, the SXM specification interacting with the user was communicated using RMI with a SXM specification in the server (Bank) that had the latest policy about the expense reporting. As a result of this, the program send back an object and associated method information that enables the client (ATM) program to screen the end users expense account data in a way that was consistent with the bank policy.

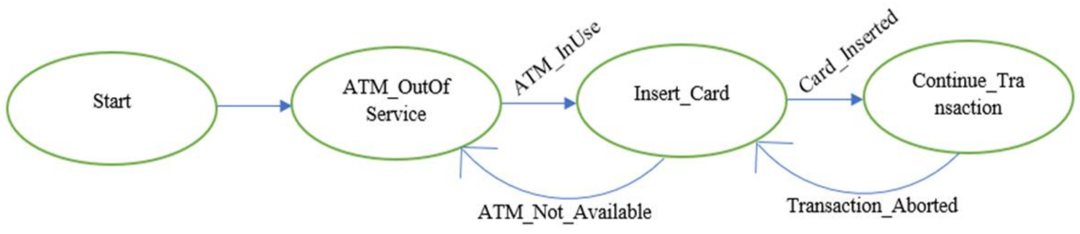


Figure 2. State Diagram of the Overall Case Study (ATM)

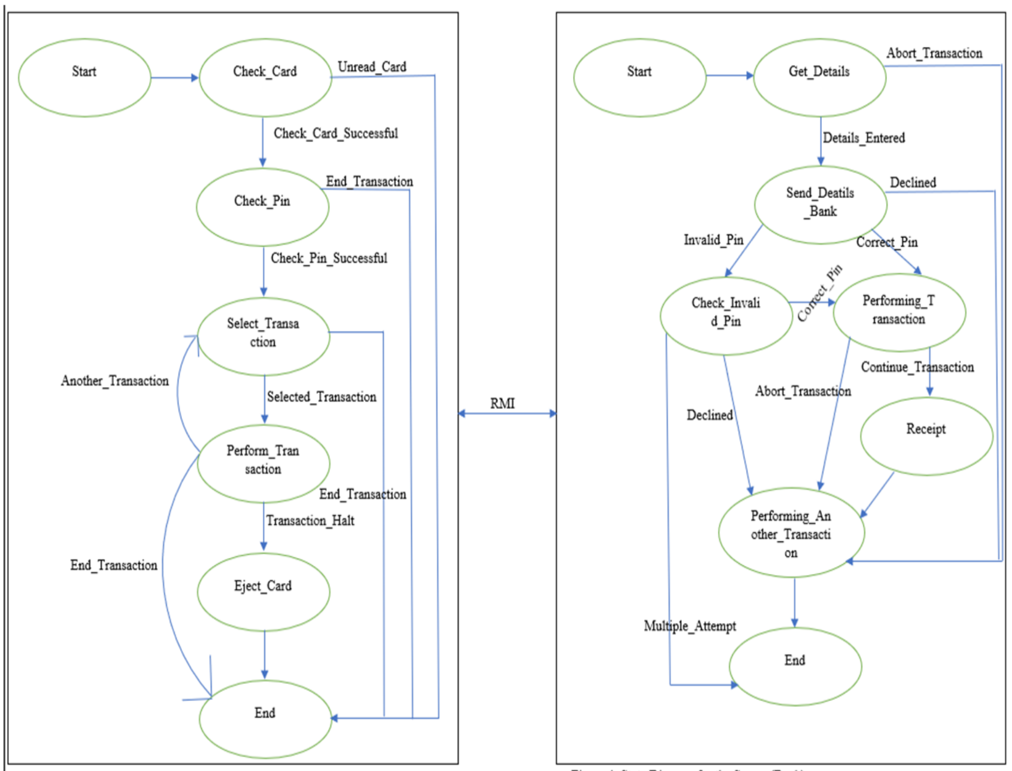


Figure 3: State Diagram for the Client (ATM)

Figure 4: State Diagram for the Server (Bank)

Figure 5. State Diagram of the CSXM.

The formal method (CSXM) used in this paper enables the end user and the bank save time by locating mistakes early which gives the advantage of correct system i.e., the specification meeting the implementations. However, whenever the bank policy changed, it would require a change to the SXM specification in only one computer. The RMI used in this paper was implemented as three layers when communicating with the SXMs i.e., client (ATM) and the server (Bank) as represented in Figure 6. These layers are as follows;

i. Stud: the stud used here is called the proxy that appears to the calling program which is then the program that is later called for the service.
ii. Remote Reference Later (RRL): the RRL was used to determine whether the request is to call a single remote service or more when considering a multicast.
iii. Transport Later: the transport connection layer was used to set up and manage the end user's request.

Hence, a single request was transmitted down through the layers on one machine and up through the layers at the other end. Also, the output of the generated test cases is presented in Fig. 7 below.

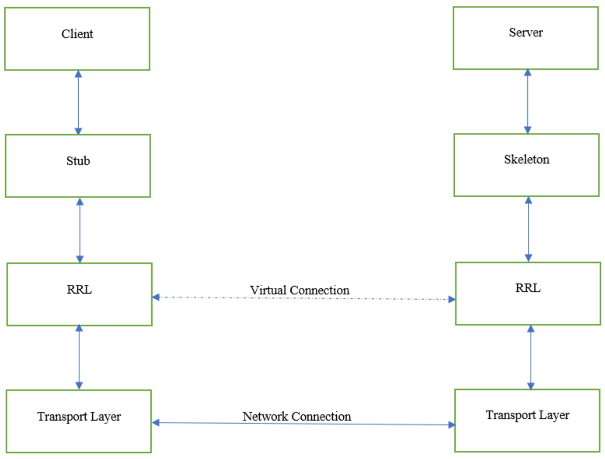
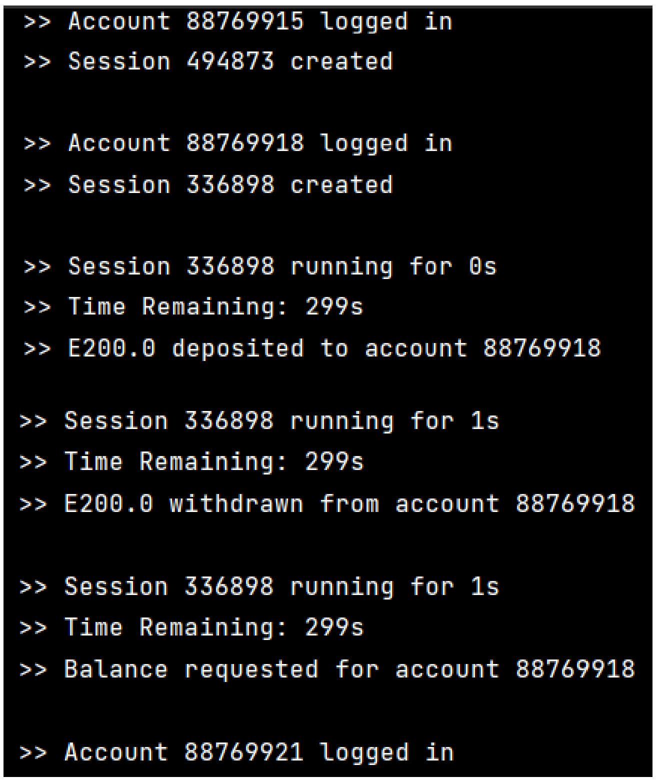


Figure 6. Architecture of the RMI.



```
>> Account 88769915 logged in
>> Session 494873 created

>> Account 88769918 logged in
>> Session 336898 created

>> Session 336898 running for 0s
>> Time Remaining: 299s
>> E200.0 deposited to account 88769918

>> Session 336898 running for 1s
>> Time Remaining: 299s
>> E200.0 withdrawn from account 88769918

>> Session 336898 running for 1s
>> Time Remaining: 299s
>> Balance requested for account 88769918

>> Account 88769921 logged in
```

Figure 7. Output of the Generated Test Cases.

4.2. Evaluation of the Developed Tool

In this paper, the developed tool was evaluated check for the correctness of the system by checking the specification if it conforms with the implementation. Furthermore, during this process some parts of the code are indicated which needs refactoring and the generated report are acted upon to assure the correctness of the software system. Moreso, the source code changes are monitored, and the functionality of this tool checks up to 500,000 lines of code. Therefore, the choice of the black box testing method was selected simply because it evaluates the functionality of an application or tool without having investigated its internal structure. Software testers and programmers were considered in the testing phase without disclosing the structure and design of the source code.

5. CONCLUSION

In conclusion, software testing in SDLC still continues to be a great research interest among researchers and software developers. However, many works have been done but still removing defects or bugs during and after development process is crucial and cannot be left out. First version of this paper was also based on using the formal method (SXM) strengths to ensure all specified in the software was implemented correctly. Optimization and improvement in the SXM extend the interest of using formal methods specification for future software testing. CSXM theory was used and practiced in this paper because of its advantage over the previous techniques of solving large-scale system problems. Nevertheless, this is not the end of this great research area as more work are proposed for future problems. Notably, the trending research as established the area compiler design where bugs not only cause unintended behavior with possibly severe consequence but also make software debugging more difficult because it is not easy to detect whether the bugs actually come from the compiler used to compile the source code or the source code itself. This as also been observed in SDLC process to avoid spending more time in checking for errors in the source code. One of the challenges is the lack of a formal specification of what exactly a compiler is supposed to do. Therefore, the future research will focus on reviewing the computational power of machine learning algorithms and integrating it with the strengths of CSXM to assure the correctness, testing, and verification of a compiler design.

ACKNOWLEDGMENT

First, we would like to give our sincere gratitude to Almighty God for His supports in every aspect of our lives and for His mercy, protection, and provision during this research. Our unalloyed appreciation goes to all the staffs in the Department of Computer Science and Creative Technologies and the entire staff of the Faculty of Environment and Technology for their continuous supports, guidance, and invaluable suggestions throughout the duration of this research. God's wisdom and understanding to continue guiding students and colleagues will not depart from you all.

REFERENCES

[1] Sanusi, B. A., Olabiyisi, S. O., Olowoye, A. O. and Olatunji, B. L. (2019). Software Defect Prediction System using Machine Learning based Algorithms. *Journal of Advances in Computational Intelligence Theory*, 1(3), 1–9. http://doi.org/10.5281/zenodo.35908 41

[2] X-machine. (2021). *In Wikipedia*. https://en.wikipedia.org/wiki/X-machine.

[3] Simons, A.J.H. and Leftticaru, R. (2020). A Verified and Optimized Stream X-Machine Testing Method, with Application to Cloud Service Certification. *Software Testing, Verification and Reliability*. 30(3): e1729. https://doi.org/10.1002/stvr.1729

[4] Kefalas, P., Stamatopoulou, I, Sakellariou, I. and Eleftherakis, G. (2008). Transforming Communicating X-Machines into P Systems. *Nat Comput*. 8:817–832 DOI 10.1007/s11047-008-9103-y

[5] Ogunshile E. (2011). A Machine with Class: A Framework for Object Generation, Intergration and Language Authentication (FROGILA). Ph.D. Thesis. The University of Sheffield. United Kingdom.

[6] Ogunshile, E.K.A. (2005). Automatic Generation of Java Code from Communicating X-machine Specifications. MSc (Eng) Thesis. Department of Computer Science. The University of Sheffield, England, United Kingdom.

[7] Leroy, X. (2009). Formal Verification of a Realistic Compiler. *Communications of the ACM*. Vol. 52. No. 7. Pages 107-115. DOI: 10.1145/1538788.1538814

[8] Mugridge, R., Utting, M. and Streader, D. (2011). Evolving Web-Based Test Automation into Agile Business Specifications. *Future Internet*. Vol. 3. Page 159-174.

[9] Akhtar, N. and Missen, M.M (2014). Contribution to the Formal Specification and Verification of a Multi-Agent Robotic System. *European Journal of Scientific Research*. ISSN 1450-216X / 1450-202X Vol.117 No.1. Page 35-55.

[10] Sharma, R. and Sangwan, M. (2016). To Improve Correctness, Quality and Reducing Testing Time. International Journal of Computer Science and Mobile Computing. (IJCSMC). Vol. 5. Issue 6. Page 211-217.

[11] Fonseca, P., Zhang, K., Wang, X. and Krishnamurthy, A. (2017). An Empirical Study on the Correctness of Formally Verified Distributed Systems. *EuroSys '17. ACM*. ISBN 978-1-4503-4938-3/17/04. DOI: http://dx.doi.org/10.1145/3064176.3064183.

[12] Chen, J., Donaldson, A. F., Zeller, A., and Zhang, H. (2017). Testing and Verification of Compilers. *Dagstuhl Reports*. Vol. 7. Issue 12. Page 50-65.

[13] Ogunshile, E.K.A. (2018). CompleX-Machine: An Automated Testing Tool using X-Machine Theory. International Journal of Computer and Systems Engineering 12 (3).

[14] Patterson, D. and Ahmed, A. (2019). The Next 700 Compiler Correctness Theorems (Function Pearl). Proc. ACM Program. Lang. 3, ICFP, Article 85. Page 1-29. https://doi.org/10.1145/3341689

[15] Koo, B., Bae, J., Kim, S., Park, K. and Kim, H. (2020). Test Case Generation Method for Increasing Software Reliability in Safety-Critical Embedded Systems. *Electronics* 2020, *9*(5), 797; https://doi.org/10.3390/electronics9050797.

[16] Olatunji, B. L., Olabiyisi, S. O., Oyeleye, C. A., Sanusi, B. A., Olowoye, A. O. and Ofem, O. A. (2020). Development of Software Defect Prediction System using Artificial Neural Network. *International Journal of Advances in Applied Sciences*. Vol. 9. No. 4. Page 284-293. ISSN:2252-8814. DOI: 10.11591/ijaas.v9.i4.pp284-293.

[17] Monteiro, F. R., Gadelha, M. R. and Cordeiro, L. C. (2021). Model Checking C++ Programs. Software Testing, Verification and Reliability. 32:e1793. Page 1-30. https://doi.org/10.1002/stvr.1793

[18] Elliott, G. (2004). Global Business Information Technology. An Integrated Systems Approach. Pearson Education. Page 87.

AUTHORS

Bashir Adewale Sanusi is currently an Associate Lecturer in the Department of Computer Science and Creative Technologies, University of the West of England (UWE), Bristol, United Kingdom. He received B.Tech. and M.Tech. in Computer Engineering and Computer Science from Ladoke Akintola University (LAUTECH), Ogbomoso, Nigeria in 2015 and 2021 respectively. Also undergoing his Ph.D. degree in Computer Science from University of the West of England (UWE). He began his career in LAUTECH as a Graduate Assistant in 2018. He has published 6 peer-reviewed Journal articles and Conference paper.

Dr Emmanuel Ogunshile, Ph.D. is a Senior Lecturer in Computer Science and Chair Athena SWAN process at the University of the West of England (UWE), Bristoll, UK-conducting highly innovative Teaching, Research, Scholarship and Administration in Computer Science and Software Engineering. Previously, he graduated with an MSc(Eng) in Advanced Software Engineering (2005) following a BEng(Hons) in Software Engineering (2003) and a Ph.D. in the subject of Computer Science (2011) all at The University of Sheffield, England, UK.

Mehmet Aydin joined CSCT department at the end of January 2015 as Senior Lecturer in Computer Science. Prior to this post, I have worked in academic and research positions for various universities including University of Bedfordshire, London South Bank University and University of Aberdeen. I am editorial board member of a number of international peer-reviewed journals, and have been serving as committee member of various international conferences. I am also member of EPSRC Review College and fellow of Higher Education Academy.

Stephen Olatunde Olabiyisi is Professor of Computer Science in the Department of Computer Science, Ladoke Akintola University of Technology (LAUTECH), Ogbomoso, Nigeria. He received B.Tech., M.Tech and Ph.D. degrees in Applied Mathematics from LAUTECH, Nigeria, in 1999, 2001 and 2006 respectively. He also received M.Sc. degree in Computer Science from University of Ibadan, Ibadan, Nigeria in 2003. He began his career in LAUTECH as a Graduate Assistant in 2000 and rose through the ranks to become a Professor in 2013. He has supervised 51 Doctoral (Ph.D) students and 58 Master's students and published over 200 peer-reviewed Journal articles and Conference papers. He is currently the Director, LAUTECH ICT Center.

Oyediran Mayowa Oyedepo is a senior lecturer in the department of Computer Engineering in Ajayi Crowther University, Oyo town in Nigeria, he has a PhD in Computer Engineering from Ladoke Akintola University of Technology, Ogbomoso. Nigeria. He is specialized in Distributed Systems and Applications with focused on Cloud computing, fog computing, MANETs and the likes.

SYSTEM FOR ASSISTANCE IN DIAGNOSIS OF DISEASES PULMONARY

Gustavo Chichanoski and Maria Bernadete de Morais França

Department of Electronic Engineering,
State University of Londrina, Londrina, Paraná, Brazil

ABSTRACT

Covid-19 is caused by the SARS-COV2 virus, where most people experience a mild to moderate respiratory crisis. To assist in diagnosing and triaging patients, this work developed a Covid-19 classification system through chest radiology images. For this purpose, the neural network models ResNet50V2, ResNet101V2, DenseNet121, DenseNet169, DenseNet201, InceptionResnetV2, VGG-16, and VGG-19 were used, comparing their precision, accuracy, recall, and specificity. For this, the images were segmented by a U-Net network, and packets of the lung image were generated, which served as input for the different classification models. Finally, the probabilistic Grad-CAM was generated to assist in the interpretation of the results of the neural networks. The segmentation obtained a Jaccard similarity of 94.30%, while for the classification the parameters of precision, specificity, accuracy, and revocation were evaluated, compared with the reference literature. Where DenseNet121 obtained an accuracy of 99.28%, while ResNet50V2 presented a specificity of 99.72%, both for Covid-19.

KEYWORDS

Deep Learning, U-Net, Covid-19, Segmentation & Machine Learning

1. INTRODUCTION

Terms related to artificial intelligence have gained evidence in the media due to their presence in devices such as smartphones, virtual assistants, and autonomous cars, where as a general concept, researchers seek to automate decisions executed by humans. As with everyday tools, medicine, and more specifically the area of radiology, has undergone major technological changes in recent years.

As described in [3] and [4], typical symptoms of COVID-19 are characterized by fever, cough, shortness of breath, muscle aches, mental confusion, headaches, sore throat, rhinorrhea, pain in the chest, diarrhea, nausea, vomiting, pneumonia, decrease in white cell count or decrease in lymphocytes.

The symptoms that can be visualized in radiology are presented in three stages, initial, progression, and strong. In the early stage of the disease, the lungs show multiple small irregular shadows and interstitial changes, most apparent in peripheral zones of the lungs. In the progression stage, there are multiple opaque ground glass and infiltration in both lungs, and in the strong stage, there is a consolidation of the lungs.

Among the works carried out on the diagnosis of Covid-19 using CT images of the chest, the work [11] stands out, whose objective was to produce a design of deep convolutional neural networks called COVID-Net, obtaining an accuracy of 93,3%, using 11.75 million parameters and with a sensitivity to COVID-19 of 91%, cited by [7] as one of the most successful methods in diagnosis. [11] used the dataset COVIDx, divided into three categories, No Disease, Pneumonia, and COVID-19, the work was made available at https://github.com/lindawangg/COVID-Net.

According to [12], in the 101 x-ray images collected at 4 institutions in Hunan, China, of patients between 21 and 50 years old, divided into non-emergency (87), mild or common cases, and emergencies (14), severe or fatal cases.

Most patients had GGO (Ground Glass Opacity), 87 (86.1% of the total cases), or a mixture of GGO with lung consolidation, 65 (64.4% of the total cases). However, when looking exclusively at emergency patients (14 cases), the number of patients with GGO reached 100% (14 cases), consolidation rose from 41.4% in non-emergency cases to 57.1% in emergency cases, while the mixture of both remained constant in 64.4% of both types of cases.

The article [7] proposes a new diagnostic approach, using an (FC)-DenseNet103 to perform lung segmentation and the Resnet-18 network to classify thorax radiology, available at https://github.com/jongcye/Deep-Learning-COVID-19-on-CXR-using-Limited-Training-Data-Sets.

The purpose of [7] is to train the weights of the ResNet-18 network using only a random real image package selected from the thorax radiology image, and this package cannot be completely zero or black. While the test is done by removing non-zero K slices from the image, passing each package through the trained network image, and defining the winner based on each class's highest number of votes. Resulting in a sensitivity of 100% for COVID-19 and an accuracy of 76.9%.

However, in [13], using the method suggested in [7] and in [14], he can create a new manual method to auxilian in the diagnosis of Covid-19, through RALE the lung was classified in a grade between 0 and 4 to each lung, where 0 is normal, 1 to 2 is wild, 3 to 5 is moderate and 6 to 8 is severe.

This article aims to improve the performance of [7] in segmentation by changing the error function to Log-Cosh Dice-Loss and using other models of neural networks. To make it possible was used a U-Net to separate lungs of background in thorax radiology, evaluating the model using F1 Score and Jaccard Score parameters. On other hand, for classification was used the models ResNet50V2, ResNet101V2, DenseNet121, DenseNet169, DenseNet201, VGG-16, VGG-19, and InceptionResnetV2, using as parameters, precision, accuracy, recall and specify.

2. REVIEW OF LITERATURE

2.1. Segmentation

To reduce the complexity of the chest x-ray image, the segmentation models isolate the lungs from the background. One of these models is U-Net, suggested in [8], shown in Figure 1.

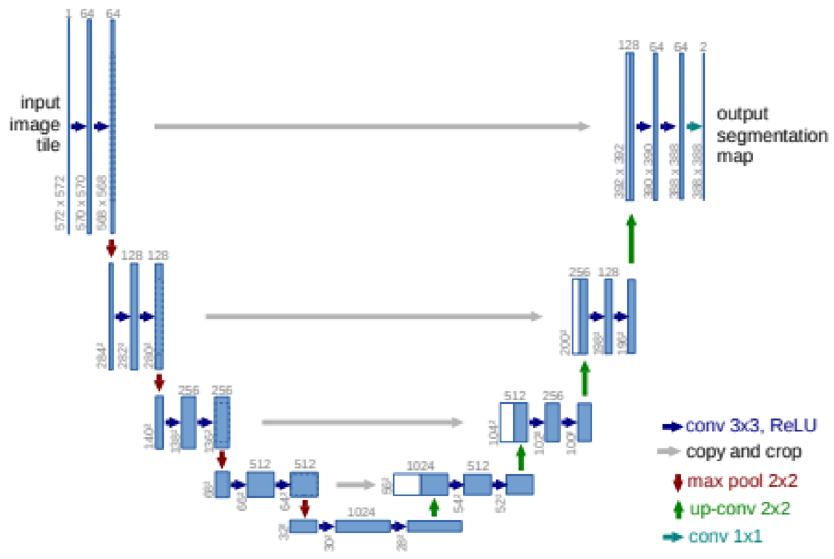


Figure 1. Architecture of U-Net model described in [8].

To reduce the complexity of the chest x-ray image, the segmentation models isolate the lungs from the background.

The U-Net model work in two steps: compression and expansion. Each level is composed of 3x3 convolution layers, ReLU activation, and a max-pooling layer in the compression step.

In the last level, the expansion step begins, consisting of up-sampling layers, concatenation layers, joining the output of the up-sampling layer with the inputs of the same level in the compression step, and then going through another convolution layer and activation. But at the last level of the expansion, add a convolution layer with a 1x1 filter and a Sigmoid activation layer.

To increase the performance of the segmentation model, the Sorensen-Dice coefficient was used to calculate the correlation between two images, according to the equation:

$$D = \frac{2 \cdot TP}{2 \cdot TP + FP + TN}, \tag{1}$$

whose abbreviations can be described as TP (true negatives), FP (false positives), FN (False negatives), and D (Sorensen-Dice coefficient). However, this equation would not serve to be applied as a cost function, for that [7] describes a new equation, shown in:

$$L(y, \hat{p}) = 1 - \frac{2y\hat{p} + 1}{y + \hat{p} + 1}, \tag{2}$$

where y represents the real values and $\hat{p}$ the values predicted by the U-Net model, the addition of 1 in the numerator and denominator makes the function finite when y and $\hat{p}$ are 0. Thus, the closer the prediction to the true value, the smaller the value of L. However, this function does not behave like a convex function, decreasing the learning efficiency of the model, to correct this [8] suggests the correction for (2) adding the Lovsz extension:

$$LD(y,\hat{p}) = \log\left(\cosh\left(1 - \frac{2y\hat{p}+1}{y+\hat{p}+1}\right)\right), \tag{3}$$

whose characteristic curve is shown in Figure 2, transforming the function into convex, keeping the finite function in the range from -1 to 1.

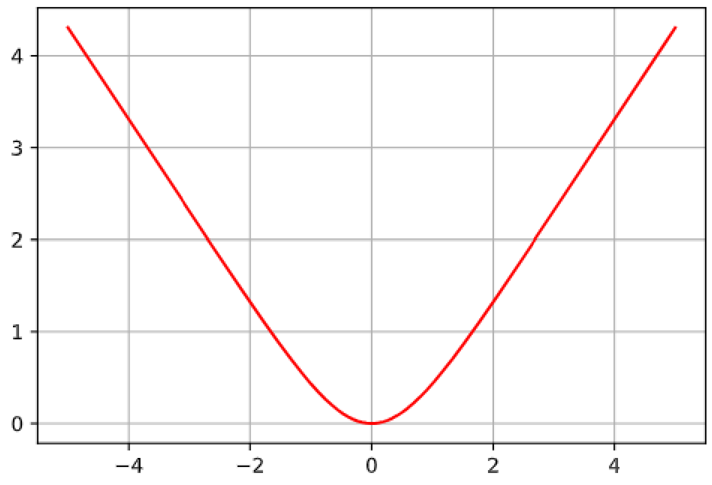


Figure 2. Characteristic curve of the Log-Cosh Dice error function.

2.2. Grad-CAM Probabilistic

To assist visualization in the interpretation of results of this new method of classification a new Grad-CAM was proposed, the Probabilistic Grad-CAM. It is a variation of the Grad-CAM, based on the equation proposed in [8], the Grad-CAM is calculated for only a single slice:

$$L^c = \mathrm{Re}LU\left(\sum_k \alpha_k^c \cdot \mathrm{A}^k\right), \tag{4}$$

Where α_k^c represents a partial linearization of the deep network through A, the importance of feature map k of target class c, calculated with the equation:

$$\alpha_k^c = \frac{1}{\mathrm{Z}}\sum_i^u \sum_j^v \frac{\partial y^c}{\partial \mathrm{A}_{i,j}^k}, \tag{5}$$

so, to calculate the Probabilistic Grad-CAM of the original input image, use the equation:

$$\left[l_{prob}^c\right] = \frac{1}{K_i}\left[\sum_{k=1}^{K} r^c(x_k) Q_k\left(l^c(x_k)\right)\right]_i, \tag{6}$$

where c is the class to be analyzed, x_k is the input slice of the model, l^c is the heatmap generated by Grad-CAM, Q_k is the operator that places the slice in the original position of the image, r^c is the class probability for the slice, K is the number of slices that use pixel, *i* is the pixel to be analyzed.

3. METHODOLOGY

3.1. Segmentation

The segmentation used two public datasets, made available by [10]. They are provided by the Department of Health and Human Services of Montgomery County (MC), Maryland, the United States, and the Shenzhen People's Hospital 3 of Guangdong Medical University in Shenzhen (Shenzhen), China. Containing a total of 800 chest radiographs. For the masks, it used the dataset provided by [10], which was generated based on [6], described in Table 1.

Table 1. Scattering of the segmentation dataset between Normal and Tuberculosis.

Dataset	No Disease	Tuberculosis	Total
Montgomery County	80	58	138
Shenzhen	326	336	662
Total	406	394	800

Both datasets form a set of X-ray images and their masks, which were also divided into training, validation, and testing. The proportion of division was 72%, 18%, and 10% respectively, whose objective was to reduce the bias. To reduce the impact of the reduced number of images in the dataset, the segmentation training dataset increases the number of artificial images to 10,000 using the data augmentation, described in [1], applying gamma correction, gaussian noise, elastic transformation, grid distortion and flip horizontal.

The images are of different sizes and types, making it necessary to normalize them to improve network performance. In this way, transform the image from uint8 to float32, resize it to 256 x 256 pixels, and normalized it from 0 to 1. In addition, a Gamma correction of 0.5 was applied. The weights were initialized using the Glorot distribution, being trained using the Log-Cosh Dice Loss error function, with the Adamax optimizer and a learning rate of 0.001, reducing according to the validation error and with a patience of 10 epochs.

3.2. Classification

For the classification of X-ray images, the dataset [8] was used, available free of charge, containing 7,103 images, of which 853 are X-ray images of patients with Covid-19, 1,887 of patients No Disease, and 4,363 of patients with Pneumonia.

The classification dataset used was publicly available in [8]. More than 7,103 images, 853 infected with Covid-19, 1,887 No Diseases, and 4,363 with Pneumonia were separated into training, validation, and testing, 61%, 15%, and 24%, shown in Table 2.

Table 2. Dispersion of images from the classification dataset between COVID-19, Normal, and Pneumonia, and the separation between training, validation, and testing in the number of images.

Labels	Train	Validation	Test	Total
Covid-19	457	114	282	853
No Disease	1,069	267	551	1,887
Pneumonia	2,790	698	875	4,363
Total	4,316	1,079	1,708	7,103

The pre-processing of the images consists of normalizing the values from 0 to 1, resized to 1024x1024 pixels, applying the Gamma correction of 0.55, and performing the image segmentation with the mask generated by the segmentation model. Next, the image is cut into several random packets of size 224x224 pixels, avoiding null areas, as shown in Figure 3. This process aims to produce the inputs for the local classification model, from which the predictions for each packet are made.

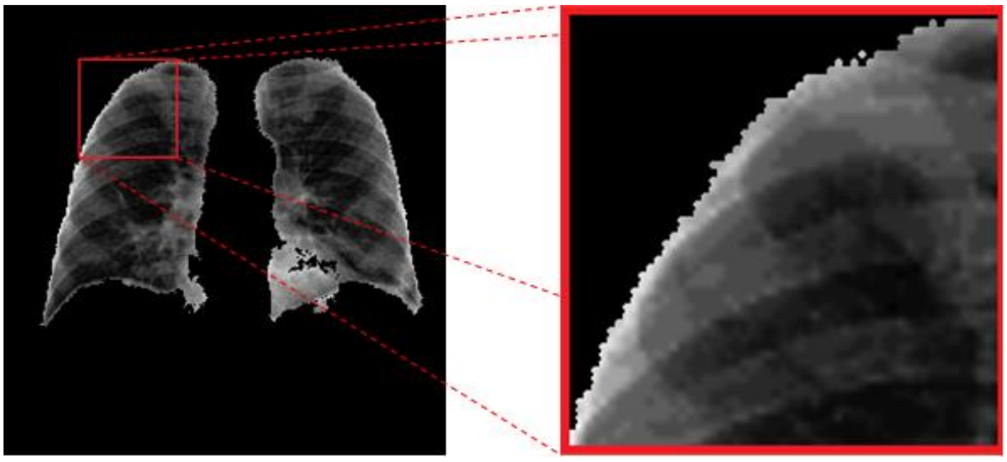


Figure 3. Slice generated by the classification model.

The model for the local approach is composed of the input layer, a convolution layer, a convolution layer, a base layer, and a dense layer. This architecture allows the exchange between the models to be tested, without interfering with the results. The input layer receives the image and converts it to TensorFlow tensors. The convolution layer converts the single-channel image to a three-channel image, with three 3x3 size filters, with a ReLU activation layer. The base layer varies depending on the model type selected: ResNet50V2, InceptionResnetV2, DenseNet121, and VGG-19. The dense layer calculates the predictions for the image, and the SoftMax activation layer filters the results.

3.3. Grad-CAM Probabilistic

Using the heatmaps generated by each package in the classification model, removing the negative parts, the heatmap of the original image is created. Finally, the value of each pixel is divided by the number of packets that contained its position. The number of packets can vary as needed, generating sharper images as the number of packets increases. In the analysis of the results, 400 packages were used.

4. RESULTS

The U-Net segmentation model was evaluated using a test dataset with 80 images through the parameters of the F1 Score and Jaccard Score, to compare the segmentation results with the reference mask. Using the F1 Score criterion, the five worst similarity results in the entire dataset were respectively 88.41%, 89.23%, 89.50%, and 89.65%, while the five best results were 98.48%, 98.51%, 98.53% 98.56%, and 98.57%. The average F1 Score performance was 96.37% similar, with a standard deviation of 2.26%.

The five worst results obtained by the Jaccard Score criterion were 79.23%, 80.56%, 80.99%, 81.25%, and 82.48%, on the other hand, the five best results were 97.01%, 97.06%, 97.11%, 97.16%. The average performance obtained by the Jaccard Score was 93.09% with a standard deviation of 4. 07%. Figure 4 shows the histogram for the F1 Score criterion from the segmentation test dataset and Figure 5 shows the histogram for the Jaccard Score criterion.

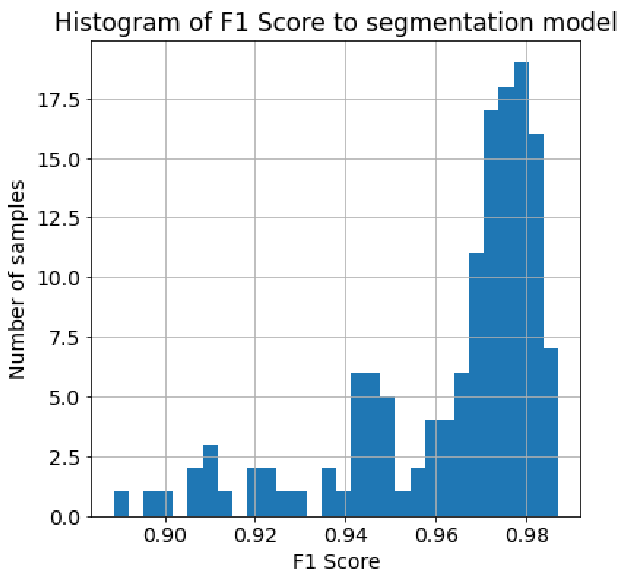


Figure 4. Histogram of segmentation model to F1 Score.

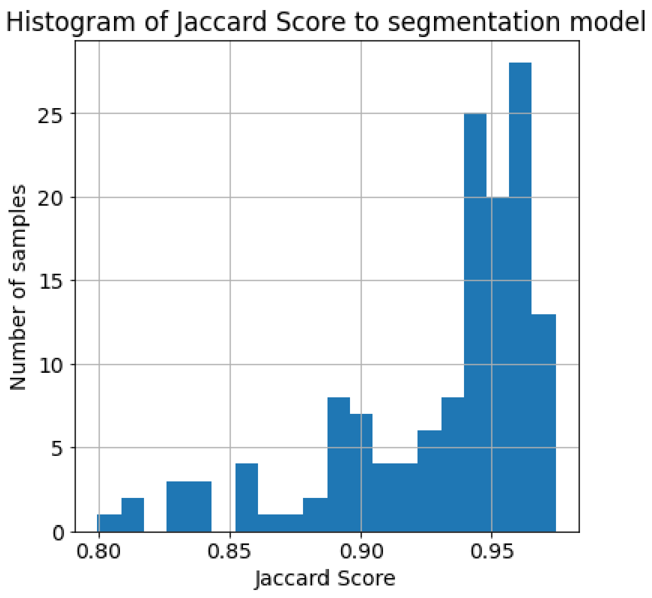


Figure 5. Histogram of segmentation model to Jaccard Score.

Figure 6 (a) shows a pre-processed image of a chest X-ray contained in the test dataset. Figure 6 (b) shows the reference mask provided by the dataset and Figure 6 (c) shows the mask generated by the segmentation model.

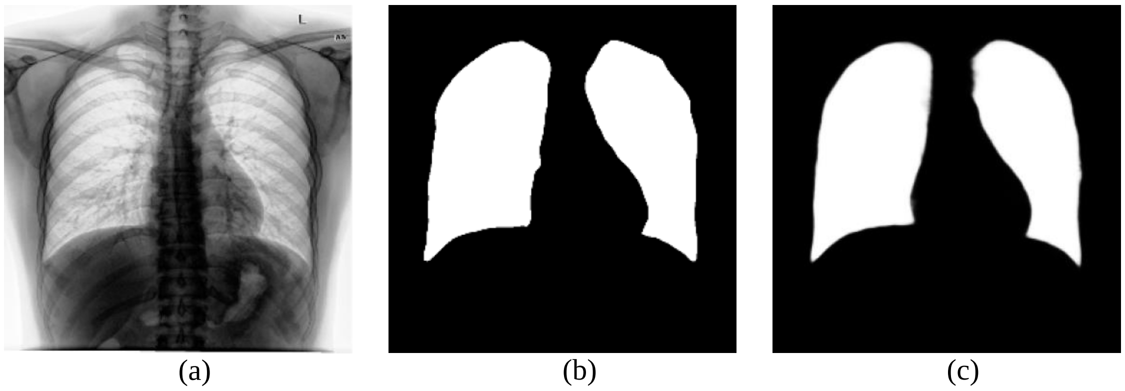


Figure 6. Example of mask generated by the model.
(a) Original image. (b) Original mask. (c) Predict mask.

The classification, in global approximation, was evaluated for precision, accuracy, specificity, and recall. Table 3 shows the results, in which the best performance for each of the criteria is highlighted.

Table 3. Results for Covid-19 using 100 slices of each image to test the dataset.

Models	Precision	Accuracy	Recall	Specify	Weights
ResNet50V2	98.23	**99.47**	**98.58**	**99.72**	23,570,980
DenseNet121	**99.28**	99.41	97.16	99.44	**7,040,613**
InceptionResNetV2	98.57	99.41	97.87	99.58	54,341,381
VGG-19	98.86	98.54	92.20	98.48	20,025,957
DenseNet169	99.21	98.13	89.36	97.94	12,647,909
DenseNet201	96.60	97.95	90.78	98.20	18,327,781
ResNet101V2	98.17	98.89	95.04	99.02	42,632,741
VGG-16	97.55	97.13	84.75	97.06	14,716,261

As the objective of this work is to develop a Medical Diagnostic System, the most important parameters of the networks are precision and specificity. The neural network with better precision was DenseNet121 and the network with better specificity was ResNet50V2. Analyzing the other parameters, we observed that, in general, ResNet50V2 presented the best performance.

Next, the performance of the networks was compared with the literature 7. Table 4 compares the precision results and Table 5 compares the specificity results. It is noted that all models presented a more accurate result for the classification of Covid19 and Pneumonia, however, for cases of No disease, the literature was better. The literature obtained a result with a better specificity for the classification of Covid-19, however, the results of the neural networks used in this work reached better values of specificity in the classification of Non-disease and Pneumonia.

Table 4. Precision of classification models for classes.

Models	Covid-19	No Disease	Pneumonia
[7]	90,30	95,70	90,30
ResNet50V2	98,23	94,32	95,90
DenseNet121	99,28	95,51	94,65
InceptionResNetV2	98,57	94,98	95,05
VGG-19	98,86	94,55	94,10
DenseNet169	99.21	88.05	93.08
DenseNet201	96.60	91.65	92.03
ResNet101v2	98.17	95.55	94.88
VGG-16	97.55	91.65	89.82

However, when looking at specificity, shown in Table 5, the literature [7] obtained the best result, 2.81% higher than the article, while for the other labels, higher values were obtained in all models, reaching 8.13% greater than [7] in best model.

Table 5. Specificity of classification models for classes.

Models	Covid-19	No Disease	Pneumonia
[7]	**100**	90.00	93.00
ResNet50V2	99.72	96.98	96.02
DenseNet121	99.30	96.59	**96.91**
InceptionResNetV2	99.58	96.58	96.45
VGG-19	98.48	**97.32**	96.19
DenseNet169	97.94	96.88	92.02
DenseNet201	98.20	95.12	94.65
ResNet101v2	99.02	96.88	95.17
VGG-16	97.06	95.12	94.26

Finally, through the Probabilistic Grad-CAM method, the heatmaps of the models in the article were generated using 400 image packages, to obtain a clearer image, as shown in Figure 7. In red are the locations that most influenced the decision and in blue the least influential, where you can see that the regions of interest are in the left lung. DenseNet121 shows less dispersed results.

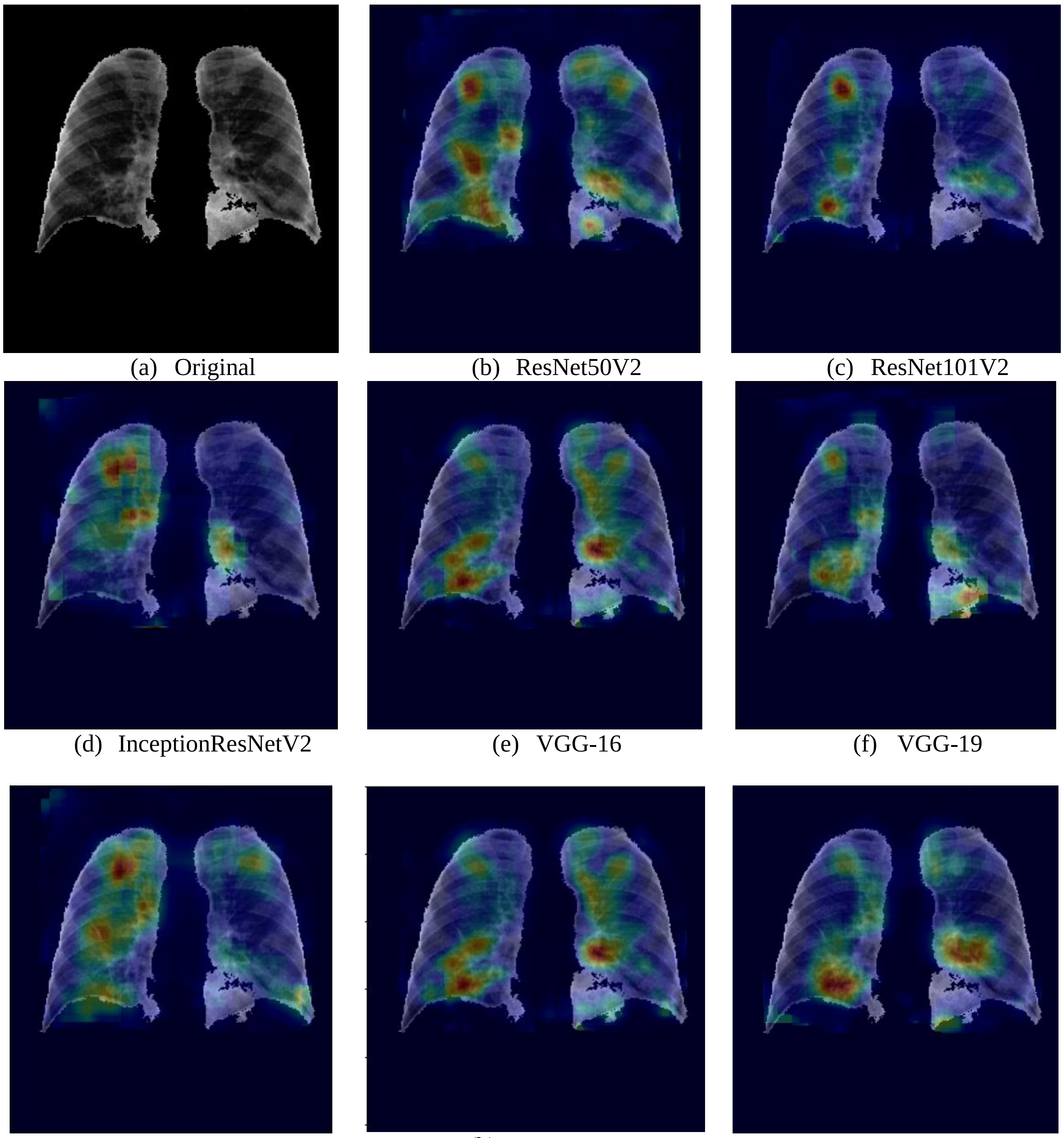


Figure 7. Grad-CAM Probabilistic generated by classifications models to Covid-19 using 400 slices.

5. Conclusions

With the current pandemic situation in the world, new diagnostic methods for these new diseases are increasingly necessary. This article improves the performance of the model suggested by [7]. The DenseNet121 network obtained an improvement in the precision of 9.94 when compared to the literature, however, in the specificity the performance was 0.7 worse. ResNet50V2 obtained an 8.78 higher precision than the literature and a 0.28 worse performance in specificity. Despite the worsening of specificity seen in ResNet50V2, this value was small compared to the high gain that this network achieved in precision.

Finally, when analyzing the images generated by the Probabilistic Grad-CAM, it can be seen that the regions of interest are in similar locations, however, the DesNet121 models obtained less diffuse regions.

ACKNOWLEDGMENTS

I would like thanks to the team of professors and colleagues from the Automation and Instrumentation Laboratory of the Electrical Engineering Department of the State University of Londrina (UEL) and other sectors involved.

REFERENCES

[1] Buslaev, A., Iglovikov, V.I., Khvedchenya, E., Parinov, A., Druzhinin, M., Kalinin, A.A.: Albumentations: Fast and flexible image augmentations. Information 11(2) (2020). https://doi.org/10.3390/info11020125, https://www.mdpi.com/2078-2489/11/2/125

[2] Cohen, J.P., Morrison, P., Dao, L.: Covid-19 image data collection. arXiv 2003.11597 (2020), https://github.com/ieee8023/covid-chestxray-dataset.

[3] De Oliveira Lima, C.M.A.: Information about the new coronavirus disease (COVID-19). Radiologia Brasileira 53(2), v–vi (2020). https://doi.org/10.1590/0100-3984.2020.53.2e1.

[4] Dong, D., Tang, Z., Wang, S., Hui, H., Gong, L., Lu, Y., Xue, Z., Liao, H., Chen, F., Yang, F., Jin, R., Wang, K., Liu, Z., Wei, J., Mu, W., Zhang, H., Jiang, J., Tian, J., Li, H.: The Role of Imaging in the Detection and Management of COVID-19: A Review. IEEE Reviews in Biomedical Engineering 14(c), 16–29 (2021). https://doi.org/10.1109/RBME.2020.2990959.

[5] Jadon, S.: A survey of loss functions for semantic segmentation. 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology, CIBCB 2020 (2020). https://doi.org/10.1109/CIBCB48159.2020.9277638.

[6] Jaeger, S., Candemir, S., Antani, S., Ẃang, Y.X.J., Lu, P.X., Thoma, G.: Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. Quantitative imaging in medicine and surgery 4(6), 475–477 (2014). https://doi.org/10.3978/j.issn.2223-4292.2014.11.20.

[7] Oh, Y., Park, S., Ye, J.C.: Deep learning covid-19 features on cxr using limited training data asets. IEEE Transactions on Medical Imaging 39(8), 2688–2700 (2020).https://doi.org/10.1109/TMI.2020.2993291.

[8] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9351, 234–241 (May 2015). https://doi.org/10.1007/978-3-319-24574-428, https ://arxiv.org/pdf /1505.04597.pdf.

[9] Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision 128(2), 336–359 (2020). https://doi.org/10.1007/s11263-019-01228-7, https://arxiv.org/pdf/1610.02391.pdf.

[10] Stirenko, S., Kochura, Y., Alienin, O., Rokovyi, O., Gordienko, Y., Gang, P., Zeng, W.: Chest x-ray analysis of tuberculosis by deep learning with segmentation and augmentation. arXiv (2018).

[11] Wang, L., Lin, Z.Q., Wong, A.: Covid-net: a tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. Scientific Reports 10(1), 19549 (Nov 2020). https://doi.org/10.1038/s41598-020-76550-z, https://doi.org/10.1038/s41598-020-76550-z.

[12] Zhao, W., Zhong, Z., Xie, X., Yu, Q., Liu, J.: Relation between chest CT findings and clinical conditions of coronavirus disease (covid-19) pneumonia: A multicenter study. American Journal of Roentgenology 214(5), 1072–1077 (2020). https://doi.org/10.2214/AJR.20.22976.

[13] TABIK, S. et al. COVIDGR Dataset and COVID-SDNet Methodology for Predicting COVID-19 Based on Chest X-Ray Images. IEEE Journal of Biomedical and Health Informatics, v. 24, n. 12, p. 3595–3605, 2020. ISSN 21682208.

[14] ZIMATORE, C. et al. Accuracy of the Radiographic Assessment of Lung Edema Score for the Diagnosis of ARDS. Frontiers in Physiology, v. 12, n. June 2021, 2021. ISSN 1664042X.

AUTHORS

Gustavo Chichanoski, Bachelor in Electronic Engineering, work in Department of Electronic Engineering, State University of Londrina, Londrina, Paraná

Maria Bernadete de Morais França, Doctor in Electronic Engineering work in Department of Electronic Engineering, State University of Londrina, Londrina, Paraná

PROGNOSIS OF INDIAN STOCK PRICE THROUGH MACHINE LEARNING ALGORITHM AND SENTIMENT ANALYSIS

Harshwardhan Patil and Rahul Patil

Department of Computer Engineering,
Pimpri Chinchwad College of Engineering, Nigdi, Pune, India

ABSTRACT

Unpredictable stock price forecasting is a difficult task due to the markets' flexible and unconditionally volatile nature. views into the machine learning meadow with the impending emotive and quantitative strategy. Increasing computational capabilities, software-based statistical medium of prognosis, and inventive method of prognosticating the model are all combined. In this study, the next due day-closing prices of Indian equities SBIN and Tatamotors are used to compare the long short term memory, Random Forest, and linear regression algorithms. Utilizing the RMSE standard layout indication, the prototypes are assessed. The low value of this indicator results in the most accurate closing price prediction model when compared to others.

KEYWORDS

Random Forest Regression; Long-Short Term Memory; Stock Price Prognosis.

1. INTRODUCTION

The character of the National Stock Exchange may be distinguished as unpredictable, fluid, and non-linear. However, the two main approaches have been proposed for predicting the stock[1] momentum of [1]particular organization. Technical analysis uses[1] histrocial price of[1] the stocks like lables such as close and open price of the stock, volume traded, adjacent close[1] value of the particular stock for predicting the future price of the stock.[1] The second type is qualitative analysis, this analysis method is permormed on the basis of external factors[1] such as portfolio of company, current situation trending in market, economical and political circumstances, text based information in the form of stock based and generalized news and social media articiles even blogs by economist , CA , financial advisiors, researchers. Modern clever tactics based on either technical or fundamental analysis are [1] utilised to forecast the movement of stock prices in these days of technical trends. In particular, the data amount is enormous and fictitious for stock price forecasting. An effective prototype that can spot hazy patterns and complex linkages in a vast data collection is required to handle varied data. In comparison to the prior approaches, machine learning techniques have shown to increase competency by 55–87% in this field. The majority of tasks in this sector make use of algorithms including linear regression, Random Forest(RF)[6], MAC/ MACD, as well as certain models for anticipating NSE goods prices, such as Autoregressive Moving Average(ARMA)[4] and Autoregressive Integrated Moving Average(ARIMA).[4] Recent studies have shown that the application of machine learning techniques[5] like Support vector machines and Random forests may improve stock price predictions (10). Some neural network-based techniques, such as Artificial Neural

Networks (ANN)[1], Convolutional Neural[1] Networks (CNN)[1], Recurrent Neural Networks[1] (RNN), and Deep Neural Networks (DNN) as Long Short Term Memory (LSTM), have also produced encouraging results[1].Moreover, sentiment prognosis prototypes records from sentiment dimensions. The sentiment of news articiles have been studied in finance sphere. Neiderhoffer (2nd pap-27) analyses NYT headlines over the past 20 years and divides them into 19 predefined semantic categories, ranging from from negative to extremely positive. He also examines how exchanges respond to various sets of events and discovers that the market has a propensity to overreact to negative news. The impact of optimistic and pessimistic language used in news on future performance of businesses is examined by Davis et al. (2nd Pap). They suggested two things: (1) there is a conflict between readers' expectations and writers' intentions; and (2) readers significantly respond to both the emotional and content aspects of reports that deviate from their expectations. According to research, investors first forecast event articles and convert them into market feelings. Investors then base their decisions on the market sentiment interpretations. Finally, market prices aggregate all of the investor activities and represent them in the ultimate price movements. As a result, it could be harder to incorporate sentiment prognosis into the forecasting blend. With this the quantative strategy like LSTM is capable for finding through long term dependability in data. This is gained through module of recurring model which has a blending of 4 layered interacting with each other.RF regressiom a supervised machine learning algorithm uses ensable method of learning in regression has also used by researchers' for forecasting the stock price. Linear regression another supervised machine learning algorithm is also used widely for forecasting the stock prices. In this work, we have divided our experimentation in two medhodologies on broad level of predicting the stock price momentum. The first method is prognosis of day end price of National Stock Exchange (NSE) listed stock price by LSTM , Random Forest and linear regression. The model uses group of new variable construct by the live National Stock Exchange(NSE) dataset with Low, Close, Open and High of selected NSE listed firm. In terms of the models' improvised forecast, which is assessed by the RMSE performance metric, the models trained using these three techniques will play a key role.The influence of sentiment that has been identified by news flashing in the market and categorised as good, negative, or neutral is being examined as the second technique in this paper.

The following is a full summary of this: Section 2 describes the methodology used. The experimentation is covered in Section 3, and the task experimentation is covered in Section 4.

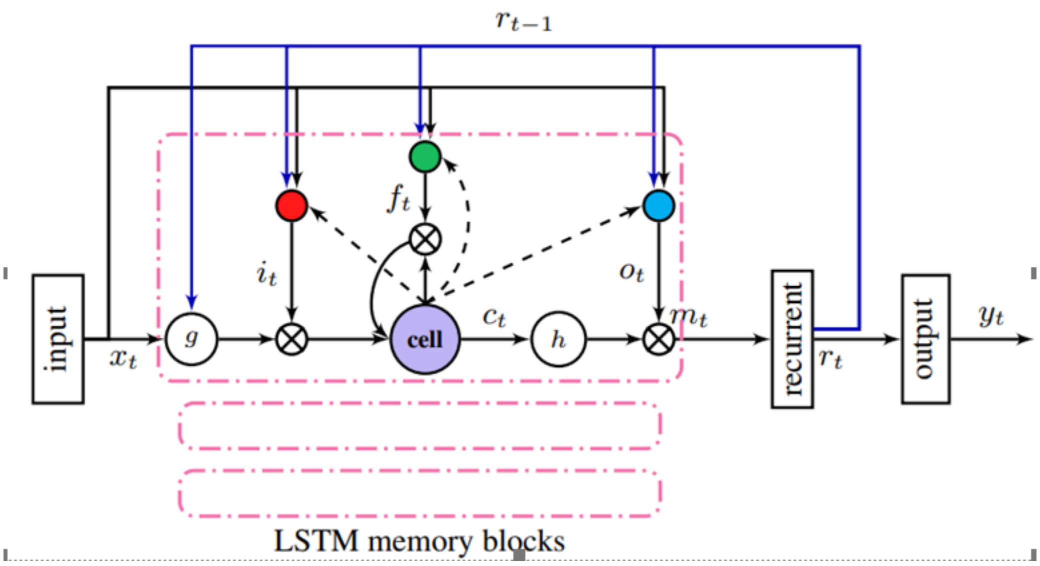
2. METHODLOGY

2.1. Explanation of Information

Information data for two firm has gathered from YFG Finance. The dataset consists of 46 days' worth of quantitative data for Tata Motors and State Bank of India from 19 May 2022 to 21 July 2022. The information includes exchange-related price points such as Low, Close, High, Open, AdjacentClose, and Volume. For predicting, the price of the exchange at day's end, taken in 24-hour intervals, has been taken into consideration. The following seven-day closing price is predicted.

2.2. LSTM

Improvised factor after RNN is LongShort-Term Memory.RNN mimicazise the learning like humans. As human, start our thinking from scratch each second. LSTMs are a type of RNN that remember information over long periods of time. This mentioned behaviour of LSTM make them better suited for predicting stock prices. The internal gates of LSTM have the ability to control the information flow. The gates are capable of learning which data in a sequence should be kept or ignored. In order to create predictions, it can convey pertinent information down the extensive chain of sequences by doing this.



2.3. Random Forest

RF that is Random Forest it is machine ensamble learnable strategy. It is bearable of computizing regressable & classifiable work. Instead of depending solely on one decision tree, it is proposed that many decision trees be combined in order to identify the desired conclusion, hence reducing model variation. All decision trees that show the choice made at the nodes of the tree may be trained using newly produced variables. The lags in exchange point data usually is presize high reasonably of its large spaced & reasoned that trees to evolve in a finished variant manner comparable to expectable extent . It focus at lowering forecasting error

2.4. The Linear Regresson

Machine Learning's supervised another type is linear regression algorithm use in work .LR carries task to prognose a variable that depends variable indicator (y) relaive on given free variable x.Regression strategy of this kind explores a L-relation between x,y In this experiment x test datagroup & y prognostable day end price for next seven days.

2.5. Sentiment Analysis

In this work for predicting the sentiment intensity of the stock releated news we have used sentiment intensity analyzer method from Vader sentiment. To fetech the news and article releated to the stock we made use of Google News and Newspaper package of python. The news articles are current day and previous day mentioned as now and yesterday.

The news is categorized in keywords to intensify positive, negative and neutral sentiments. Positive sentiement will increase if negative count is less than positive count on the other side Negative sentiment increase by +1 if the positive count in news is less than negative count. In the condition where positive count is equal to the negative count then it will be categorized to neutral sentiment.

We implement module by sentiment PO[polarity] , indicator for distinguish the article range of sentiment

ght = positive when (f(p)-f(n))/f(n) ≥th
ght = negative when (f(n)-f(p))/f(p) ≥th

3. Concluding and Experimented Result

To digout effect trained systems, comparision is done with RMSE performance score of LSTM, Random forest and linear regression. With the data date starting from 19-05-2022 till 21-07-2022. The stock like SBI, Tata Motors and Cipla are choosen for experimentation purpose. Prognosted next end day prices permormed through (RMSE).
The RMSE equat by below formation.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(O_i - F_i)^2}{n}}$$

here 'The Oi' point to the original day end price, 'fi' point to the prognosted day-end price ,n point overall observation space.

Price action chart below shows the trend in SBI stock price for last 46 days. The trend line for LSTM , Linear regression and Random forest shows the positive momentum in SBI . Moreover calculating the RMSE results in choosing best predictive model among LSTM, Random Forest and Linear Regression.

3.1. State Bank of India Prognosis

The figure 2 shows the forecasting of LSTM against SBI stock price from 19 May 2022 to 21 July 2022.
The figure 3 shows the forecasting of Random Forest against SBI stock price from 19 May 2022 to 21 July 2022.
The figure 3 shows the forecasting of Linear Regression against SBI stock price from 19 May 2022 to 21 July 2022.

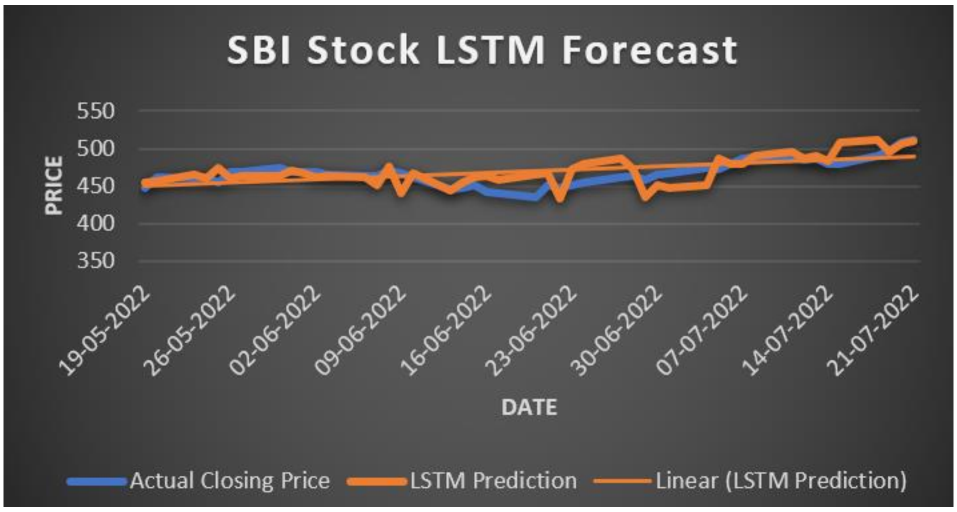


Figure 2. SBI Actual Price Vs LSTM Forecast Price

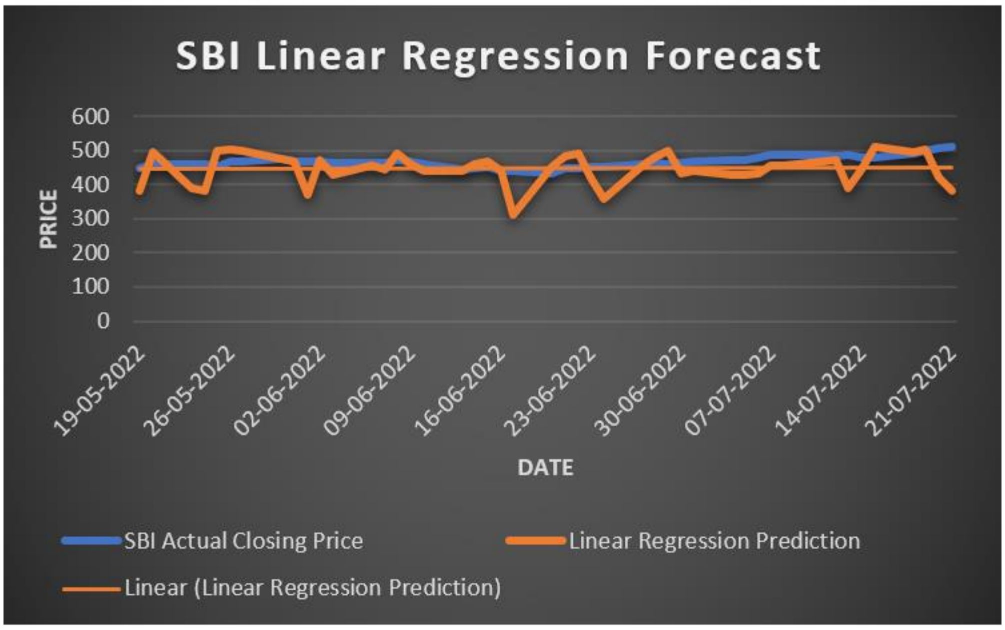


Figure 2. SBI Actual Price Vs Linear Regression Forecast Price

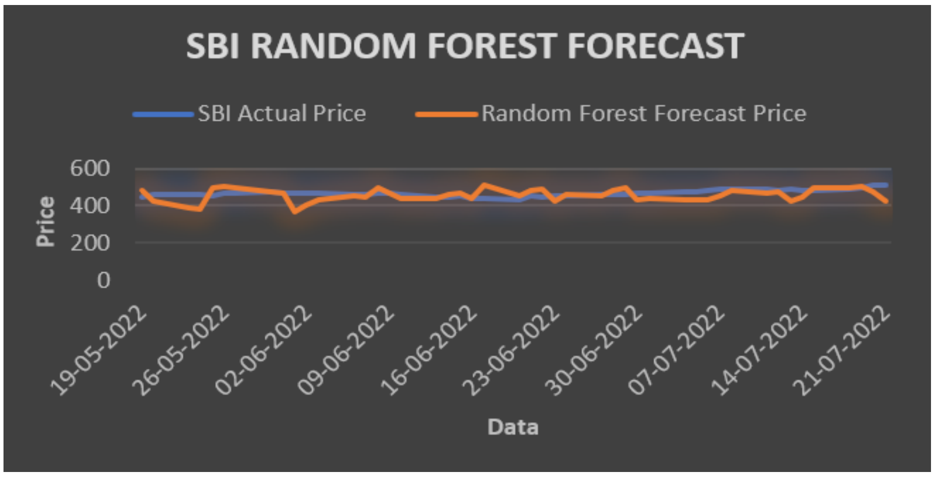


Figure 3. SBI Actual Price Vs Random Forest Forecast Price

Table 1. Root Mean Squared Error of SBI Stock for 46 days.

Algorithm	**Root Mean Squared Error**
Long Short Term Memory	14.2
Linear Regression	36.5
Random Forest	37.2

The sentiment analysis shows the positive sentiments are 85% whereas negative sentiments has coverage of 10% and Neutral sentiment has coverage of 5%.

Stock releated news article believed to have the impact on stock releated news article with the momentum of trend. Here SBI stock 46 days average polarity for positive sentiment is 85%. That has witness the positive trend. Similarly tatamotors polarity in figure 6

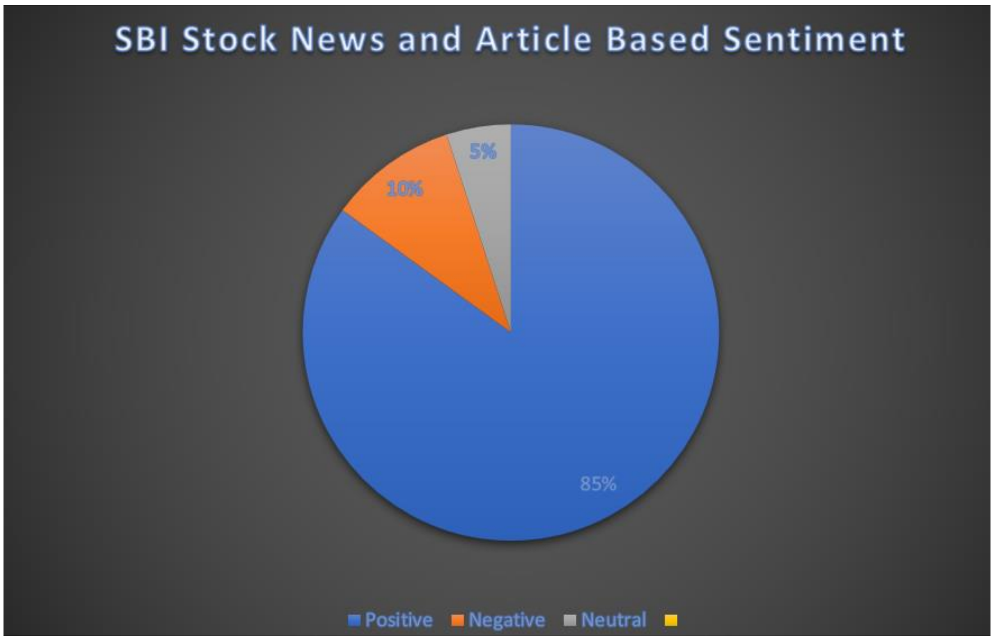


Figure 5. Sentiment Analysis of SBI News

3.2. Tata Motors Prognosis

Figure 6 shows the forecasting of LSTM against SBI stock price from 19 May 2022 to 21 July 2022.

Figure 7 shows the forecasting of Linear Regression against SBI stock price from 19 May 2022 to 21 July 2022.

Figure 8 shows the forecasting of Random Forest against SBI stock price from 19 May 2022 to 21 July 2022.

Comparative trend in tata motors has predicted the momentum in the same direction as per actual price. Research proved that for all three algorithms Price is exactly accurate on each day of the market. There observed the variation in the actual and predicted prices. But the significant conclusion here can be statemented that the trend momentum of stock price can be predicted i.e in upcoming days price action will be at up-trend or down-trend. Tata motors here shows the positive momentum in futuristic cycle.

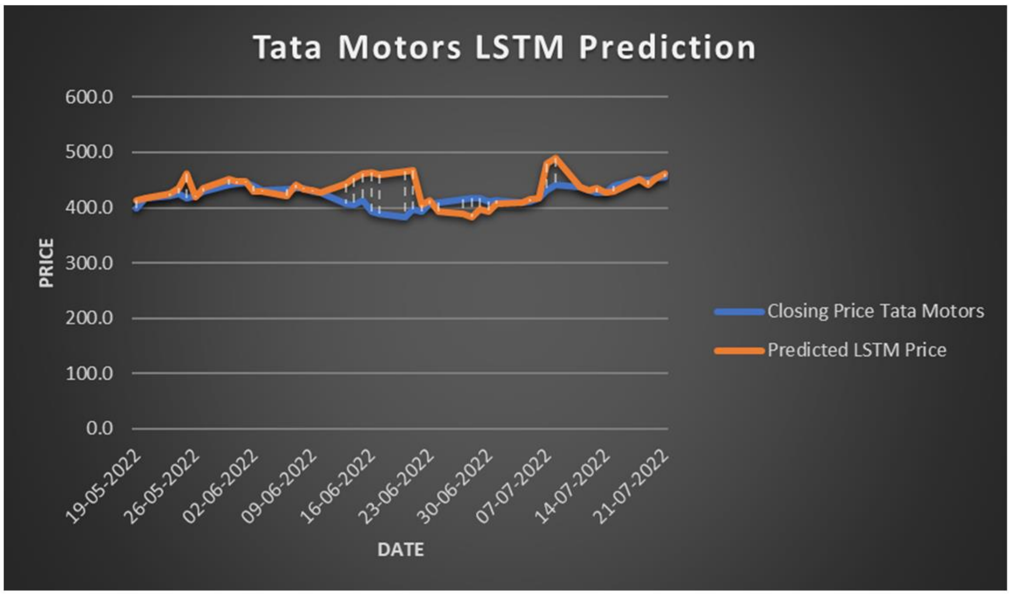


Figure 5. Tata Motors Actual Price vs LSTM forecasted price

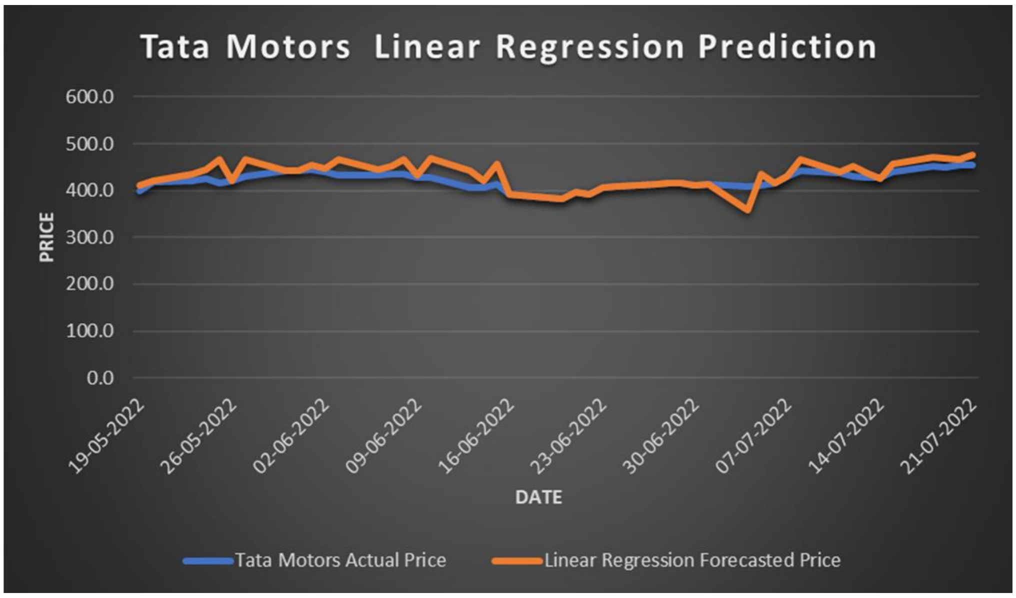


Figure 6. Tata Motors Actual Price vs Linear Regression forecasted price

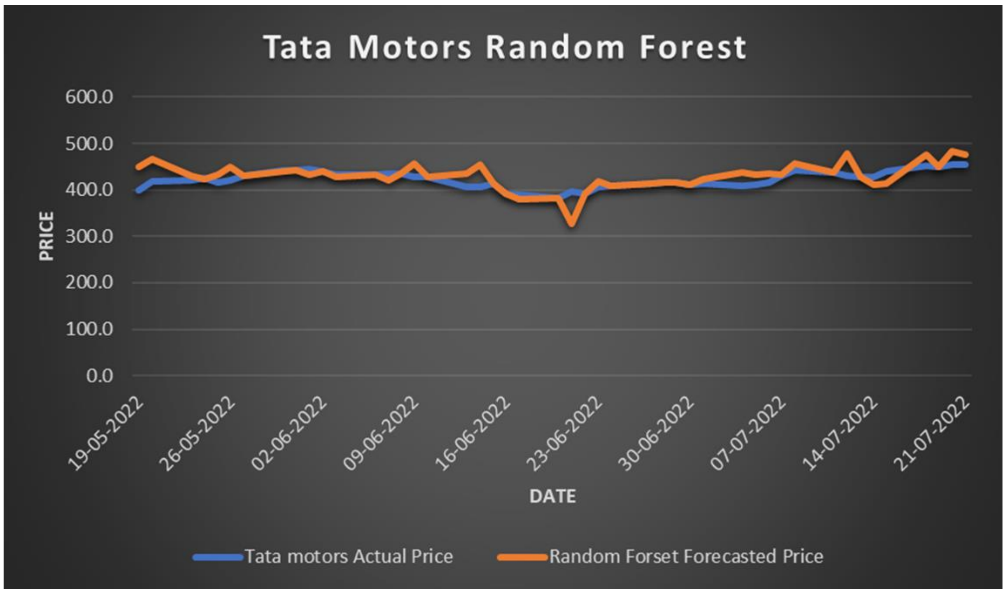


Figure 7. Tata Motors Actual Price vs Linear Regression forecasted price

Table 2. Root Mean Squared Error Tata Motors Stock for 46 days.

Algorithm	**Root Mean Squared Error**
Long Short Term Memory	11.94
Linear Regression	19.94
Random Forest	22.44

Table 2 shows the root mean squared error value for Tata Motors stock of LSTM, Random Forest and Linear Regression. Form table two the result shows tata motors prediction price performance through RMSE value Long-Short Term memory has least root mean squared error with 11.94 whereas it is followed by Linear regression with 19.94 and Linear regression is followed by Random forest with 22.44 RMSE respectively. The 46 days average of tata motors sentiment analysis results the positive sentiments as 78% whereas negative sentiments has coverage of 18% and Neutral sentiment has coverage of 4%. Based on 96 news article in past 46 days the polarity score has been generated. The table below informs the total number of news article that has used for polarity prediction.

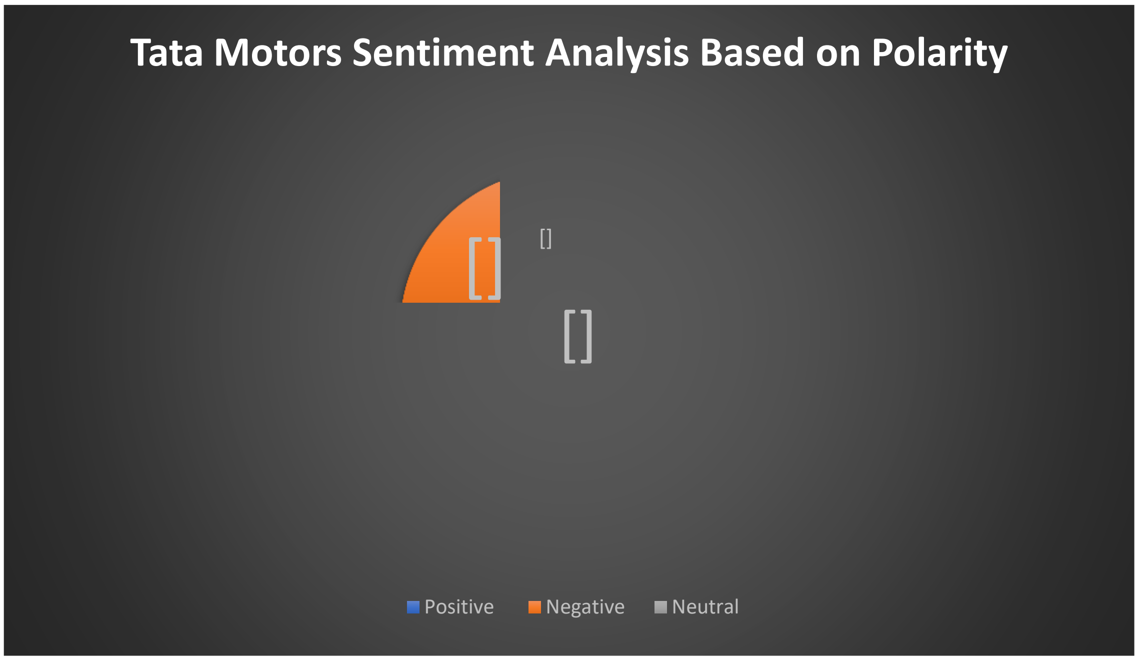


Figure 8. Sentiment Analysis of SBI News

Table 4. Comparitive analysis of Tata Motors and SBI RMSE values obtained using LSTM, Random Forest and Linear Regression

Company	LSTM	Linear Regression	Random Forest
Tata Motors	11.94	19.94	22.44
SBI	14.2	36.5	37.2

The values above T4. depict, for SBI & Tatamotors company, LSTM results to be finer technique, giving least rmseRootMeanSquaredError, as shown in the Table 3.

The result of sentiment analysis about SBI and tatamotors predict the positive polarity about both the stock with positive polarity factor of SBI as 85% and that of tatamotors is 78%. This shows bull side will be stronger for these stocks in upcoming days.

4. Conclusions

Forecasting exchange price outcomes a stimulating work anticipated through routinely fluctuating Indian National exchange price that depend on multiple facts which form compound patterns. Results show that the best values obtained by LSTM model gives RMSE (11.94) for past 46 days of tata mototrs and 14.2(RMSE) for SBI.

The result of sentiment analysis about SBI and tatamotors predict the positive polarity about both the stock with positive polarity factor of SBI as 85% and that of tatamotors is 78%. This shows bull side will be stronger for these stocks in upcoming days

REFERENCES

[1] Mehar Vijh, Deeksha Chandola, Vinay Anand Tikkiwal, Arun Kumar. "Stock Closing Price Prediction using Machine Learning Techniques", Procedia Computer Science, 2020

[2] https://www.mdpi.com/ Stock Market

[3] pure.iiasa.ac.at

[4] https://www.science.gov/

[5] Stock Market Forecasting using Machine Learning: Today and Tomorrow 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)

[6] Indian stock market prediction using artificial neural networks on tick data.DharmarajaSelvamuthu,etlhttps://jfinswufe.springeropen.com/articles/10.1186/s40854-019-0131-7

[7] Stock Prediction with Random and Long Short-term Memory. Shangxua

[8] Research on the text sentiment classification about the social hot events on Weibo Fulian Yin etl

[9] Analysis of various machine learning algorithm and hybrid model for stock market prediction using python. Sahil Vazirani; Abhishek Sharma; Pavika Sharma

[10] Predicting Stock Movement Using Sentiment Analysis of Twitter Feed with Neural Networks. Sai Vikram Kolasani1

[11] Stock Market Analysis using Supervised Machine Learning.Kunal Pahwa; Neha Agarwal

[12] Word Embeddings and Their Use In Sentence Classification Tasks

[13] Adi Shalev etl Stock Market Forecasting using Machine Learning: Today and Tomorrow

[14] Sukhman Singh; Tarun KumarMadan; Jitendra Kumar; Ashutosh Kumar Singh

AUTHORS

Harshwardhan Patil, MTech Student at Department Of Computer Engineering, Pimpri Chinchwad College Of Engineering For Year 2021-22.

Prof Rahul Patil, Professor at Department Of Computer Engineering, Pimpri Chinchwad College of Engineering

A PROGRESS ON PROTEIN STRUCTURE PREDICTION USING VARIOUS SOFT COMPUTING TECHNIQUES

Niharika Chaudhary and Sanjay Saini

Department of Physics and Computer Science, Faculty of Science,
Dayalbagh Educational Institute (Deemed to be University),
Dayalbagh, Agra, India

ABSTRACT

In molecular and computational biology, predicting the three-dimensional structure of a protein from its amino acid sequence has long been an outstanding goal. Soft computing techniques for solving protein structure prediction problems have been gaining the attention of researchers because of their capacity to accommodate imprecision and uncertainty in vast and complicated search spaces. This paper provides a comprehensive overview of recent protein structure prediction efforts and progress using various soft computing techniques. This paper summarises key research in the field of protein structure prediction that has been published in the recent decade. Despite significant research efforts in recent decades, there is still a lot of room for improvement in this field.

KEYWORDS

Nature Inspired Computing, Swarm Intelligence, Deep Learning, Protein folding, Protein structure prediction.

1. INTRODUCTION

Proteins play a vital role in all living organisms. Proteins are complex macromolecules comprised of one or more long chains of amino acid residues connected by peptide bonds. Proteins perform various functions within organisms, including catalysing metabolic reactions, DNA replication, responding to stimuli, providing structure to cells, and transporting molecules from one place to another [1]. The function of a protein is directly linked with its native structure. The protein sequence is composed of 20 different amino acids that are aligned in linear chains via peptide bonds [2]. The classification techniques of popular protein sequences involve the extraction of particular characteristics of the sequences; these characteristics depend on the structural and functional properties of the amino acids [3]. During synthesis, an individual protein is folded into a three-dimensional structure under the influence of various chemical and physical factors, providing essential biological functions and properties that play an essential role in biological science, medicine, drug design, and disease prediction [4]. Sometimes, proteins fold into an incorrect structure (known as misfolding); a single missing or incorrect amino acid could cause such a misfold, which leads to a protein with different properties which can be harmful to the organisms [5]. Therefore, knowledge of protein structure is very crucial in protein research.

Biologists describe a hierarchy of protein structures with four levels to better characterize the structural properties. The levels are organised in steps, with each lower level influencing the construction of the next. A linear sequence of amino acid residues is linked together in long chains by peptide bonds to form a primary structure of a protein. The secondary structure refers to the periodic structure fragment of the polypeptide chain. The impact of hydrogen bonding between amide hydrogen and carbonyl oxygen in the peptide chain produces it. The fundamental elements of the secondary structure of the proteins are the α-helix and β-sheets [6]. The tertiary structure is a whole polypeptide chain generated by further combination and folding of multiple secondary structures in 3-D space. It could already represent the primary biological function of those proteins that only have one polypeptide chain [7]. The quaternary structure is a protein complex consisting of several polypeptide chains with multiple tertiary structures, and it could completely represent the characteristics of its biological function [8]. As a result, protein structure prediction (PSP) is the technique of determining a protein's three-dimensional structure from its amino acid sequence.

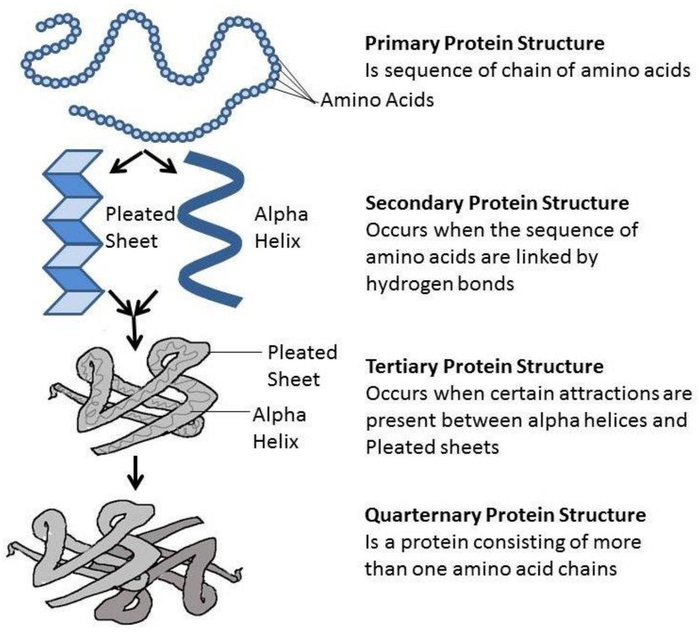

Primary Protein Structure
Is sequence of chain of amino acids
Amino Acids
Pleated Sheet
Alpha Helix
Secondary Protein Structure
Occurs when the sequence of amino acids are linked by hydrogen bonds
Pleated Sheet
Alpha Helix
Tertiary Protein Structure
Occurs when certain attractions are present between alpha helices and Pleated sheets
Quarternary Protein Structure
Is a protein consisting of more than one amino acid chains

Fig.1. Levels of Protein Structure

Protein structure prediction (PSP) is a global optimization problem and today's most important and stimulating problem of structural bioinformatics, due to the fact that it determines the biological functions and the properties of a protein. PSP is a multimodal optimization problem classified as NP-hard [9]. Anfinsen's thermodynamic hypothesis [10] states that the native structure of a protein corresponds to the global minimum of the free energy surface of the protein, therefore, the protein structure prediction (PSP) problem can be translated into a global optimization problem. Hence, predicting the native structure of a protein is an important and challenging task in computational biology [11].

In general, experimental techniques like X-ray crystallography (XRC), Nuclear Magnetic Resonance (NMR), and cryo-electron microscopy (cryoEM) have been used to determine the native structure of protein, that are not always feasible and are very expensive and time-consuming[12]. More recently, Cryo-EM has fostered an acceleration of the protein structure determination process. This method's secret is in using photographs of frozen molecules to ascertain their structure. Despite this, the method typically produces structures with lesser resolution than those measured by other experimental techniques [13]. Though there has been a continuous improvement in the experimental techniques, the gap between the number of protein sequences and the known structures is increasing rapidly. As of June 2021, the

UniProtKB/TrEMBL [14] database contains 21,91,74,961 protein sequences, whereas in the Protein Data Bank (PDB) there are 1,80,419 protein structures stored.

To bridge this sequence-structure gap, computational approaches are preferred to overcome the difficulties associated with the experimental approaches [15]. The three basic computational approaches for PSP are homology or comparative modelling, threading or fold recognition, and ab-initio or denovo technique. Homology modelling is based on comparing the sequence's similarity, but the threading approach is a method for connecting probably brief sub-conformations of the relevant sub-sequence [16]. On the other hand, ab-initio methods try to find a 3D model of the protein exclusively using the amino acid sequence, according to the laws of physics and chemistry [17]. Critical Assessment of protein Structure Prediction (CASP), a well-known biennial competition of protein structure prediction adopted this classification. Results from this competition serve as a benchmark of computational biology's progress [18].

2. Overview of Techniques used in PSP

Although PSP is NP-hard problem, no precise and effective method can provide the best solution in a polynomial amount of time. For decades, a lot of researchers have offered several approaches to solve the PSP. But several methods have limitations when the complexity of a problem increases. In contrast to other available methods, Nature Inspired Computing (NIC) based techniques can be employed to tackle PSP and provide solutions that is close to optimal in terms of complexity [19]. Emerging technology called Nature Inspired Techniques (NIC) intends to create new computational methods by drawing inspiration from how nature behaves in diverse scenarios while tackling challenging issues. A subset of Nature Inspired Computing (NIC) methods is Swarm Intelligence (SI). Swarm intelligence (SI) algorithms are primarily stochastic search and optimization methods that are inspired by the collective behaviour and self-organization of insect swarms. They are effective, flexible, and reliable research methodologies with several latent parallelisms that yield nearly ideal results [20].

SI is a promising stream to develop powerful solutions for optimization problems. Also, Metaheuristics have shown promising outcomes in the solution of difficult optimization problems [21]. For high-dimensional issues, they become constrained in terms of effectiveness and runtime. GPU-based parallel metaheuristics have been implemented in a variety of ways, and they appear to be a viable option for reducing execution time and improving solution quality. GPUs are now an effective tool for high-performance computing. They utilise the single-instruction, multiple-data (SIMD) architecture, which enables the concurrent parallel execution of hundreds of threads [22]. NIC techniques have successfully been experimented with and applied to machine learning and advanced artificial intelligence. Most of the current research is based on methods encouraged by these concepts [23].

Artificial Intelligence (AI) has been a thriving field, with a lot of applications. In recent times AI shows great promise in protein structure prediction. AI can accurately and efficiently predict thousands of possible protein structures [24]. In order to compare the prediction models' output to the known crystal structures and evaluate the quality of the models, artificial intelligence programmes are trained using a variety of numerically represented atomic features from the models (such as bond lengths, bond angles, residue-residue interactions, physio-chemical properties, and potential energy properties) [25].

Machine learning is a technology that allows computers to learn from their experiences and do tasks that are comparable to those performed by humans. ML algorithms formulate rules that link inputs to predicted results. Later, these rules are then applied to unknown data to provide physiochemically plausible accurate solutions. Machine learning techniques in AI have been successfully applied to automate routine labour, understand speech or images perform medical diagnoses, and predict protein structures [26]. Recently, researchers have also proposed machine learning algorithms that are different from the previously dominating deep learning approach, such as the deep forest [27]. Even though machine learning has been seen to be very successful in a lot of fields, there is still a significant gap between methodology and practical applications.

Deep Learning is a sub-field of Machine Learning based on artificial neural networks, usually used to learn the patterns from the input data, and in turn, make accurate predictions from it [28]. DL emphasizes the use of multiple layers to better represent data in different abstractions and has drastically improved the state-of-the-art methods in PSP. The use of DL approaches in structural bioinformatics improves the prediction model performance to a new level [29]. Deep learning has already been applied in a variety of protein processing tasks, including protein structure prediction [30], protein interaction prediction [31], protein secondary structure prediction [32], and protein sub cellular localization prediction [33]. Since the third generation of predictors, a variety of Deep Learning algorithms, as well as more traditional Machine Learning methods like k-Nearest Neighbours, Linear Regression, Hidden Markov Models, Support Vector Machines (SVM), and Support Vector Regression, have been routinely used for PSA prediction.

Every year, a significant gathering known as the Critical Assessment of Protein Structure Prediction (CASP) compares models for predictions and prediction analysis. Since 1994, the CASP has been a community-wide, worldwide project for protein structure prediction that takes place biannually [34]. CASP aims to benchmark the protein-structure prediction methods and stimulate the advancement of the field. Fully blinded testing of structure prediction algorithms is the core principle of CASP [35].

3. SOLVING PSP USING VARIOUS TECHNIQUES (LITERATURE SURVEY)

In this section, we present a literature survey on relevant methods of protein structure prediction based on soft computing techniques. Soft computing is built on both organic and artificial concepts. It is referred to as computational intelligence. The major concepts related to the various successful soft computing techniques are discussed in the following section.

3.1. Computational Approaches to PSP

These can broadly be categorized into Template-Based Modelling (TBM) and Template-Free Modelling (FM). TBM makes use of a template to predict 3D models. The proteins with the same sequences tend to bend in the same structures form the basis of this method. It can be seen that even proteins with a 30% sequence identity bend in the same structures [36].

Free-modelling, ab initio modelling, and de novo modelling are used interchangeably to discuss template-free modelling approaches. Ab initio protein structure prediction or Free modelling (FM) attempts to build a 3D structure without using homologous proteins as templates [37]. FM methods rely on creating algorithms that can quickly discover global energy minima and a scoring system that can choose the best conformation from among the several created models. FM aims to predict the most stable protein spatial arrangement with the lowest free energy [38]. Template-based approaches have a greater prediction accuracy than other methods, especially when the target protein and template are relatively homogeneous, making them useful in practical applications.

SWISS-MODEL [39], developed by Torsten Schwede's structural bioinformatics group, is a well-known online tool for homology modelling of protein structures. In the last two decades, it has become one of the most extensively used homology modelling tools. Template identification, target-template alignment, model construction, and model evaluation are all part of the prediction process. HHblits [40] is employed for template recognition and target-template alignment. Modeller [41] is another homologous modelling tool developed, that uses the target-template alignment to implement structure modelling by satisfying spatial restraints. I-TASSER [42] [43] is a comprehensive structure prediction method based on the threading method developed by the Yang Zhang Lab.

The Yang Zhang Lab has created another good fragment-assembly approach, QUARK [44]. QUARK employs replica-exchange Monte Carlo simulations guided by a knowledge-based force field to build structural fragments ranging from 1 to 20 residues. Bhageerath [45] is one such ab initio-homology hybrid method. It is accessible in the form of a web server called Bhageerath-H [46]. The main focus of the protocol is on the reduction of conformational search space. It is based on a template-based modelling and energy-based function that generates several structures according to a systematic sampling of loop dihedral conformational space. The performance of this software was tested by using CASP10 targets with promising prediction results. After the assessment of its shortcomings, an updated version was introduced in the CASP12 meeting as BhageerathH+ [47]. The David Baker Lab created Rosetta [48], a template-free approach that assembles a full-length structure from fragments of 3-9 residues from PDB Structures. To predict globally minimized protein models it follows a Monte Carlo simulation-based strategy. Its greatest success was recorded in CASP11, when it accurately predicted the structure of a sequence of 256 amino acids [49].

CASP (Critical Assessment of Protein Structure Prediction) determines the current state of the art of protein structure prediction. The first success in protein tertiary structure prediction was observed in CASP4, but primarily for small proteins ($\leq$ 120 residues). In later years, better contact prediction approaches were introduced in CASP11 [50] competing pipelines with promising improvements in prediction accuracy. A similar pattern was observed in CASP12 with the incorporation of alignment-based contact prediction methods [51]. The use of deep learning techniques for structure prediction in CASP13 demonstrated considerable improvement above the average GDT_TS score. That gives an encouragement to delve further into deep learning-based approaches to solve the protein folding problem [52]. Experimental results on CASP11, CASP12 refinement targets, and blind tests in CASP 13 turn out to be promising. Residue-residue interactions, disorder region prediction, model quality assessment (MQA) approaches, tertiary structure refinement, and other structure-related modelling categories are all evaluated by CASP.

For protein sequences with 150 residues or fewer, free-modelling has so far proven to be a good fit. Few instances have been reported when algorithms went too far and attempted to predict the structure of longer proteins. CASP11 witnessed major success in ab initio protein structure prediction for a length of 256 amino acids [53]. Template-Based modelling approaches still have few limitations whereas ab initio approaches can move a step ahead and might help to understand the basic principles of protein folding [54].

3.2. Deep Learning-Based Approaches in PSP

In recent years, deep learning has recently proven extremely effective in tackling challenging problems in several problem domains of protein structure prediction. Their strength comes from their capacity to learn features of data at multiple levels of abstraction, beginning from relatively simple feature sets. Meanwhile, it has become a very powerful artificial intelligence tool in bioinformatics [55]. It is detailed in [55] how the prediction method DeepCov uses fully convolutional neural networks to operate on frequency or covariance data for amino-acid pairs that are directly obtained from sequence alignments. The newly developed DeepMetaPSICOV(DMP) [56] deep learning-based contact prediction tool combines the input feature sets utilised by MetaPSICOV and DeepCov as input to a deep fully convolutional residual neural network. Similarly, MapPred [57] is also a deep learning-based contact prediction method. MapPred consists of two-component methods, DeepMSA and DeepMeta, both trained with the residual neural networks.

Al Quraishi [58] builds a pure deep learning-based prediction approach. It is a one-step algorithm for the prediction of protein structure relying on an end-to-end deferential deep learning strategy. Thus, the RGN (Recurrent Geometric Networks) end-to-end differentiable protein structure predictor was created, allowing for the direct prediction of torsion angles to build the protein backbone. MULTICOM [59] is another protein structure prediction pipeline that has also incorporated recent advances in DL-based approaches to improve the protein structure prediction system. Torrisi et al. [60] recently reviewed DL-based approaches for the evolution of predictive methods for one-dimensional and two-dimensional Protein Structure Annotations, from simple statistical methods. Also looked at how these algorithms database growth has affected our ability to learn about evolution and co-evolution and how that has affected our ability to make better predictions. Moreover, Gao et al. [61] reviewed the recent advance in DL-based approaches for the protein sequence–structure–function paradigm. They analysed the emerging deep learning techniques for protein structural modelling and discussed advances and challenges that must be addressed.

Li et al. [62] proposed a new Rosetta based method trRosetta (transform-restrained Rosetta) for protein structure prediction given a protein sequence developed by Yang's and David Baker's group. In order to predict the inter-residue geometry and orientations, three dihedral and two planar angles between the residues are used to indicate the orientations between two residues. For the first time, RaptorX-Contact [63], a residual neural network (ResNet)-based model, used a deep neural network to predict protein contacts, considerably increasing accuracy on difficult targets with novel folds. Hanson et al. [64] proposed the method called SPOT-Contact (Sequence-based Prediction Online Tools for Contact map prediction) that captures the contextual information from the whole protein 'image' at each layer. SPOT-contact is highly accurate for predicting contacts at all sequence-position separations, and improves over RaptorX-Contact at all Neff values, with the largest improvement for low medium range Neff sequences. TripletRes [65] developed by Zhang's group, is a contact map prediction that ranked first in the Contact Predictions category of CASP 13 for PSP. Co-evolutionary feature extraction, deep neural network modelling, and deep multiple-sequence alignment generation are the components of TripletRes. TripletRes training needed four GPUs operating at the same time using Adam, with an 80% dropout rate. It also correctly predicted both globally and locally multi-domain proteins.

DeepMind's AlphaFold [66] is another tool for protein structure prediction method developed by DeepMind, has achieved the best performance in the template-free modelling domain of CASP13 and has evolved to AlphaFold2 at CASP14 [67]. It uses a two-step process for protein structure determination and also involves the use of co evolutionary profiles to guide model building. This methodology constructed high-accuracy models for 24 out of 43 test proteins achieving a TM-score of 0.7 [68].The protein structure prediction field has seen a lot of progress due to Deep Learning (DL)-based techniques as verified by the success of AlphaFold2 in the most recent CASP14. AlphaFold works on the idea that given a protein sequence, it is feasible to build a learnt, protein-specific potential by training a Deep Neural Network to produce accurate structure predictions and then reducing the potential via gradient descent to predict the structure itself [69].

DL-based Cryo-electron microscopy (Cryo-EM) is a useful method for studying protein structures and determining their dynamics [70]. Cryo-EM is a Nobel Prize-winning technique for creating three-dimensional maps of protein structures. The basic computational step in this method is the analysis of EM data to obtain protein structural information. Esquivel-Rodrguez et al. [71] reviewed recent improvements in computational approaches for modelling protein three-dimensional structures using a 3D EM density map built from 2D maps. For Cryo-EM based protein structure determination, there have been some recent DL-based breakthroughs in several processes such as single-particle picking, backbone prediction, secondary structure prediction, EM density map refining, and all-atom prediction for protein complexes [72]. The authors reviewed some of the recent approaches for single-particle picking, backbone structure prediction, and secondary structure prediction for the construction of high-resolution 3D Cryo-EM maps [73].

The molecular dynamics (MD) simulation is one of the most popular optimization techniques for fine-tuning protein structures. In order to speed up the extraction of the final structure from comparable fragments in the PDB and to improve the convergence of MD-based structural refinement simulations, Raval et al. [74] introduced contact-based restraints into MD simulation. David Shaw's group utilized different sets of restraints to reduce the molecular dynamics simulation runs and prevent the model from getting trapped in a non-native energy state. MELD (Modelling Employing Limited Data) is a newly developed physics-based protein structure prediction approach that uses Bayesian law to use atomic molecular dynamics of proteins for structural modelling. It has proven to be productive in determining high-resolution structures of proteins up to 260 residues long [75]. Replica exchange molecular dynamics (REMD) and the residue-specific force field (RSFF1) in explicit solvent have been demonstrated by another group to shorten simulation times and control energy landscapes [76].

3.3. Support Vector Machines

SVM is a machine learning method used to predict protein structure from its primary sequence. SVM's fundamental approach entails transforming the samples into a high-dimension Hilbert space and searching for a separation plane there [77]. The optimal separation plateau (OSH), also known as the separation hyperplane, is selected to maximize its distance from the nearby training samples. The nine SVM-based contact predictors that Wu et al. [78] created are combined with the sparse template contact restrictions in [79]. They make use of the I-TASSER's energy function as well as contact predictions produced by SVMSEQ's enhanced versions.

Wang et al. [80] proposed a protein secondary structure prediction method based on SVM with position-specific scoring matrix (PSSM) profiles. The PSSM profiles are obtained from commonly used protein CB513 datasets and conclude that accuracy increases by 11.3%. Xie et al. [81] developed a new method for the PSSP based on an improved fuzzy support vector machine (FSVM), in which an approximate optimal separating hyperplane is constructed by iterating the class centres in the feature space, and sample points close to this hyperplane are assigned with large membership values, while outliers with small membership values are assigned with low membership values based on the K-nearest neighbour algorithm. SVMQA [82] is one of the best SVM based single-model quality assessment methods. Using 19 features, including 8 potential energy based and 11 comparing projected and actual models, this method generates a TM Score and GDT_TS score.

Uziela et al. [83] introduced two unique approaches, ProQRosFA and ProQRosCen, modelled after the cutting-edge of ProQ2 technique. ProQ2 [84] is a model quality evaluation algorithm that uses SVM to forecast both the local and overall quality of protein models. While the new predictors are based on Rosetta energies, ProQ2 employs contacts and other features that were computed from a model. ProQ3 inherits all of ProQ2's capabilities and add two new ones based on Rosetta energy: full-atom Rosetta energy terms and coarse-grained centroid Rosetta energy terms. On both the CAMEO and CASP11 datasets, ProQ3 exceeds ProQ2 in correlation and achieves the highest average GDT_TS score [83]. Another secondary structure prediction method, conSSert [85] based on support vector machines (SVM) gives remarkable accuracy for beta-strand prediction, with a QE accuracy of over 0.82 and a Q2-EH of 0.86. Finally, Zhou et al. [86] applied a support vector regression technique that detects the native structures of a protein among decoy sets. This method makes use of the fast Fourier transform, amino acid network score, and contact energy-based score.

3.4. Neural Network Techniques

Since Qian and Sejnowski first introduced one of the initial NN approaches for PSSP in 1988, NN has steadily grown to be the most popular model in this area and has accomplished great feats. Various NN architectures have been developed throughout time to produce the best results in certain application areas. Deep learning, recurrent neural networks, BP neural networks, radial basis function neural networks, and complex-valued neural networks have been the most often utilised NN models in the recent decade [87].

Heffernan et al. [88] developed an integrated sequence-based prediction that predicts four different sets of structural properties of proteins by iterative learning in a parallel scheme based on a deep neural network. Local backbone structures that are represented by secondary structure, backbone torsion and Cα angles, and dihedral angles are the structural characteristics. SPIDER2 expands on this technique by predicting solvent accessibility and training an independent set of weights for each iteration. Spencer et al. [89] proposed the first deep learning-based PSSP method, called DNSS, where restricted Boltzmann machine (RBM) based deep belief network (DBN) model was trained by contrastive divergence46 in an unsupervised manner. The method used a position-specific scoring matrix generated by PSI-BLAST to train a deep learning network. GPU and CUDA software optimizes the deep network architecture and efficiently trains the deep networks.

Wang et al. [90] presented Deep Convolutional Neural Fields (DeepCNF) for protein SS prediction which is a conditional neural field deep learning extension for PSSP including Q3 and Q8. The DeepCNF integrated the advantages of both CNF and DCNN and include information about longer-range dependencies as well as the complicated link between neighbouring secondary structure labels and the sequence structure. Also, JPred4 [91] and Porter 5 [92] are the most accurate predictors used in the prediction of 3-class protein secondary structures.

Yaseen et al. [93] proposed a novel method for improving secondary structure prediction accuracy by employing statistical context-based scores as encoded features in neural network training. This approach is being used by a server known as SCORPION (secondary structure prediction). The three common secondary structure states are predicted by SCORPION by grouping (G, H, I) into helices (E, B) into sheets, and (T, S, C) into coils, in accordance with the majority of secondary structure prediction algorithms.

Multiple architectures are interlaced in a few modern PSSP approaches to enhance overall network prediction. Li and Yu [94] employed a deep convolutional recurrent NN architecture (DCRNN) to extract multi-scale local contextual information using CNN with varied kernel sizes. To predict secondary structure given amino acid sequence information, a typical feed-forward NN with one hidden layer is used. Their goal was to identify the ideal parameter configuration for producing the greatest prediction outcomes.

Zhang, B et al. [95] proposed a convolutional residual recurrent neural network (CRRNN) for both Q8 and Q3 secondary structure prediction. The model's parameters are evaluated, and dimensionality reduction is accomplished using a 1D convolutional filter with a single kernel. For PSSP, a recurrent neural network (RNN) is extensively used to solve sequence-based issues. Porter 4.0 [96] and SPIDER3 [97] are successful RNN methods for protein secondary structure prediction where physicochemical parameters are effectively included in the NN model to improve SS prediction and obtain up to 84 per cent accuracy.

The Deep inception-inside-inception (Deep3I) network is presented as a novel deep neural network design for protein secondary structure prediction and developed as a software application. MUFOLD-SS. It is made up of a series of layered inception modules that transfer the input matrix to either eight or three secondary structure states [98]. Zhang and Shen [99] proposed a method called ThreaderAI that applies a deep residual neural network for prediction. By integrating sequence profile, predicted sequential structural features, and predicted residue-residue contacts, ThreaderAI first uses deep learning to predict residue-residue aligning probability matrix. Then, by applying a dynamic programming algorithm to the probability matrix, it builds template-query alignment.

3.5. Evolutionary Computation (EC) Techniques

Based on the Darwinian concepts of evolution, Evolutionary Algorithms (EAs) and Genetic algorithms (GAs) represent two subtypes of Evolutionary Computation (EC). Different protein structural representations, such as dihedral or torsion angles and lattice models, may be used in EC-based methods (direction vector representation and hydrophobic–polar model). The 2D-HP triangular model is addressed in [100], where Evolutionary Programming is used as a search algorithm by applying a hybrid algorithm combining genetic algorithm, tabu search strategy, and local search procedure.

The hybridization of global and local search techniques is commonly known as Memetic Algorithms (MAs) or Hybrid Genetic Algorithms. The method utilizes a structured population using a local search strategy based on the Simulated Annealing (SA) method, ad-hoc crossover, and mutation operators to deal with the problem. By utilising an Angle Probability List (APL), which helps in reducing the search space and providing guidance for the search strategy, it retrieves the structural knowledge recorded in the PDB [101].

Dorn et al. [102] proposed a knowledge-based Genetic Algorithm (GA), aiming to reduce the size of the conformational search space considering the previous occurrences of amino acid residues in experimentally determined proteins. Similar to the APL concept, the technique used appropriate torsion angle intervals for the amino acid targets. The NIAS server was made available by Borguesan et al. [103] to compute ad-hoc APLs to benefit from prediction techniques or in any other application that can use the conformational preferences of amino acids. A hybrid algorithm that combined GA and tabu search (TS) algorithms to solve PSP problems were used, where authors utilized TS instead of the mutation operator in GA for preventing premature convergence and providing a faster convergence rate [104].

Garza-Fabre et al. [105] applied a memetic algorithm (MA), based on fragment assembly technique identified as a search heuristic of the Rosetta ab initio protocol. The phenotypic crowding technique is used as a similarity criterion for the selection of individuals in the multi-objective GA developed in [106]. Based on this approach, two solutions with the most similar ones are selected according to their structural differences, which inferred the template-based delay of the population convergence. Gao et al. [107] attempted a multi-objective evolutionary algorithm for protein structure prediction. They divide the force fields of the Chemistry at

Harvard Macromolecular Mechanics (CHARMM) protein-energy function into bond and non-bond energies as their first and second objectives. Considering the effect of solvent innovatively acquires a solvent-accessible surface area as the third objective, and sixty-six benchmark proteins are used to verify the proposed method.

3.6. Swarm Intelligence (SI) Techniques

The collective behaviour of decentralised, self-organized systems, which can be either natural or artificial, is known as swarm intelligence (SI). Generally, SI systems contain a population of simple agents interacting locally with one another and with the environment. The inspiration usually comes from nature, especially biological systems. Recently, SI techniques have been applied to solve various PSP problems.

In order to establish the protein structure, Cheng-yuan et al. [108] created a quantum-behaved PSO (QPSO), in which the population is split into an elite sub-population, an exploitation sub-population, and an exploration sub-population. Fine-tuning and exploration strategies are used to facilitate faster convergence, bond angles are used to represent the protein conformation, and the AB off-lattice model reveals the protein-energy function. Zhou et al. [109] proposed an improved PSO for protein folding prediction. Levy probability distribution was utilised to update locations, and the velocities of the chosen worst particles were used to accelerate convergence and break out of local optima. For protein structure optimization, an enhanced Artificial Bee Colony (ABC) [110] algorithm was put forth and tested on comparatively less synthetic and natural protein sequences. The performance of the fundamental ABC algorithm is improved by an internal feedback method, which also produces results that are marginally superior to those of other algorithms.

Another ABC variant for 3D PSP is presented in [111], where structure quality was improved by using the convergence information of the ABC algorithm during the execution process. The new algorithm performed better than previous cutting-edge methods on some cases of artificial and actual proteins, according to the authors' demonstrations. By applying the same technique as in [103] to thirteen genuine protein sequences, Li et al. [112] expanded their previous work and compared the outcomes with those found in the literature. Additionally, the chaotic ABC method has a high computing cost. Swarm intelligence systems were compared in a study of protein folding issues based on the 3D AB off-lattice model. Only a small number of artificial protein sequences with short lengths were used by the authors to analyse the performance of the standard versions of the algorithms [113]. When comparing actual protein sequences to artificial protein sequences, the comparison might be more accurate.

The Particle Swarm Optimization search algorithm is used to discover the ideal pair of dihedral angles of a structure utilising a profile-level knowledge-based force field. Detail the population-based harmony search as the optimizer as well as the 2D-AB off-lattice model [114]. In an effort to improve the artificial intelligence-based protein structure refinement method known as AIR, the authors of Wang et al. [115] developed a unique multi-objective particle swarm optimization (PSO) structure refinement protocol. The basic objective is to use multiple energy functions as multi-objective for correcting the potential bias problem caused by minimizing only a single energy function. When large conformation spaces are required, it can achieve optimal search with swarm intelligence and the information-sharing method amongst the particles in PSO.

In order to propose a neural network model for protein secondary structure prediction (PSSP), PSO has been investigated [116]. Six common datasets, including PSS504, RS126, EVA6, CB396 and Manesh, have been used for the neural network's training and testing. The suggested model is validated using cross-validation with 10, 20, 30, and 40 folds, as well as sensitivity analysis. Self-organizing map (SOM) based PSO approach with the efficient classification of secondary and tertiary proteins explored in [117]. They grouped 2D and 3D proteins, where 2D proteins contain fewer hydro-carbons than 3D proteins. The angles of the proteins are considered by evaluating the SOMs with the Bounding Box technique for a quicker analysis. An effective protein structure prediction method that combines template-based and template-free techniques was used by Gao et al. [118]. The initial protein conformations, in particular, can be constructed using a non-redundant protein database and a random selection approach with secondary structure limitations. To update the protein structures while keeping the secondary structures the same, three alternative structure evolution approaches are used: enhanced particle swarm optimization (PSO) algorithm, random perturbation, and fragment substitution.

4. SUMMARY

Protein structure prediction (PSP) is an important field of research in computational biology and protein science. Due to growing demand in the last decade, a large number of PSP methods have been proposed, but still, there is considerable scope for further improvement. This paper tried to cover the recent works published from 2012. In this review, we have outlined the most cutting-edge soft computing methods used to solve the protein structure prediction problem.

This paper first provides an introduction and the related knowledge of PSP; then, the overview of most relevant methods used in PSP are reported; finally, the literature review on various soft computing techniques like TBM (template-based modelling), TFM (template-free modelling), deep learning, support vector machines, neural networks, evolutionary computations, and swarm intelligence for solving PSP problem is provided. Various soft computing techniques and their tools that have been applied to solve the PSP problem are summarized in Table1.

Table.1. Summary of protein structure prediction methods

Method/Tools	Dataset Size	Architecture	Year	Reference
ProQ2	CASP7-9	Support Vector Machine (SVM)	2012	[84]
Porter 4.0	TS115, PDB	2 stages with Bidrectional Recurrent Neural Network (BRNN)	2013	[96]
Bhageerath-H	CASP10	Template-Free based modelling	2014	[46]
SCORPION	Cull7987	3 stages with Feed-forward Neural Network (FNN)	2014	[93]
PSI-BLAST	CASP9-10	Position-specific scoring matrix (PSSM)	2014	[89]
SPIDER 2	TR4590,CASP11	Deep artificial neural network (DeepANN)	2015	[88]
Rosetta	CASP11	Template-Free modelling	2016	[48]
ProQ3	CAMEO, CASP8-10	Support Vector Machine (SVM)	2016	[83]
conSSert	PDBSelect25, PISCES	Support Vector Machine (SVM)	2016	[85]
DeepCNF	CASP,CB513	Deep Neural Network with 5 hidden layers	2016	[90]
SVMQA	CASP8-12	Support Vector Machine (SVM)	2017	[82]
SPIDER 3	TR4590, TS115	Long Short Term Memory (LSTM) Bidirectional Recurrent Neural Networks (BRNNs)	2017	[97]
RaptorX-Contact	PDB25	Residual network (ResNet)	2017	[63]
Porter 5	PDB	Two-stage ensemble of BRNN and CNN	2018	[92]
MUFOLD-SS	CullPDB, CASP10-12	Deep Neural Network (inception-inside-inception)	2018	[98]
CRRNN	CB513, CASP10-12	Deep Neural Network with 5 hidden layers	2018	[95]
DeepCov	PDB, CASP12	2D Convolutional Neural Network (CNN)	2018	[55]
SPOT	PDB, UniProt	Residual-bidirectional-LSTM	2018	[64]
QUARK	CASP13	Deep Residual CNN	2019	[44]
C-I-TASSER	CASP13	2D Convolutional Neural Network (CNN)	2019	[42]
RGN	ProteinNet12	bi-Long Short Term Memory (LSTM)	2019	[58]
DeepMetaPSICOV	PDB, CASP13	Residual network (ResNet)	2019	[56]
AlphaFold	PDB, CASP13	Deep Neural Network	2020	[66]
trRosetta	CAMEO, CASP13	Residual network (ResNet)	2020	[62]
MapPred	PISCES	Residual network (ResNet)	2020	[57]
ThreaderAI	SCOPe40, CASP13	Deep Residual Neural Network (RNN)	2020	[99]
AIR	CASP11-12	Artificial intelligence-based multi-objective optimization protocol (MOOP)	2020	[115]
TripletRes	CASP11-13	Deep Neural Network	2021	[65]
MULTICOM	CASP8-11	Deep Convolutional Neural Network (CNN)	2021	[59]

REFERENCES

[1] Lesk, A. (2019). Introduction to bioinformatics. Oxford university press.

[2] Wang, Y., Mao, H., & Yi, Z. (2017). Protein secondary structure prediction by using deep learning method. Knowledge-Based Systems, 118, 115-123.

[3] Hendy, H., Khalifa, W., Roushdy, M., & Salem, A. B. (2015). A study of intelligent techniques for protein secondary structure prediction. International Journal of Information Models and Analyses, 4(1).

[4] Neudecker, P., Robustelli, P., Cavalli, A., Walsh, P., Lundström, P., Zarrine-Afsar, A & Kay, L. E. (2012). Structure of an intermediate state in protein folding and aggregation. Science, 336(6079), 362-366.

[5] Argyrou, A. (2020). The Misfolding of Proteins. In GeNeDis 2018 (pp. 249-254). Springer, Cham.

[6] Jiang, Q., Jin, X., Lee, S. J., & Yao, S. (2017). Protein secondary structure prediction: A survey of the state of the art. Journal of Molecular Graphics and Modelling, 76, 379-402.

[7] Voet, D., Voet, J. G., & Pratt, C. W. (2016). Fundamentals of biochemistry: life at the molecular level. John Wiley & Sons.

[8] Jana, N. D., Das, S., & Sil, J. (2018). A Metaheuristic Approach to Protein Structure Prediction: Algorithms and Insights from Fitness Landscape Analysis (Vol. 31). Springer.

[9] Kuhlman, B., & Bradley, P. (2019). Advances in protein structure prediction and design. Nature Reviews Molecular Cell Biology, 20(11), 681-697.

[10] Anfinsen, C. B. (1973). Principles that govern the folding of protein chains. Science, 181(4096), 223-230.

[11] Zhou, C., Hou, C., Wei, X., & Zhang, Q. (2014). Improved hybrid optimization algorithm for 3D protein structure prediction. Journal of molecular modelling, 20(7), 2289.

[12] Boiani, M., &Parpinelli, R. S. (2020). A GPU-based hybrid jDE algorithm applied to the 3D-AB protein structure prediction. Swarm and Evolutionary Computation, 58, 100711.

[13] Lawson, C. L., Berman, H. M., & Chiu, W. (2020). Evolving data standards for cryo-EM structures. Structural Dynamics, 7(1), 014701.

[14] UniProt: the universal protein knowledge base in 2021." Nucleic Acids Research 49, no. D1 (2021): D480-D489.

[15] Hanson, J., Paliwal, K. K., Litfin, T., Yang, Y., & Zhou, Y. (2020). Getting to know your neighbor: protein structure prediction comes of age with contextual machine learning. Journal of Computational Biology, 27(5), 796-814.

[16] Feig, M. (2017). Computational protein structure refinement: almost there, yet still so far to go. Wiley Interdisciplinary Reviews: Computational Molecular Science, 7(3), e1307.

[17] Márquez-Chamorro, A. E., Asencio-Cortés, G., Santiesteban-Toca, C. E., & Aguilar-Ruiz, J. S. (2015). Soft computing methods for the prediction of protein tertiary structures: A survey. Applied Soft Computing, 35, 398-410.

[18] Kinch, L. N., Li, W., Schaeffer, R. D., Dunbrack, R. L., Monastyrskyy, B., Kryshtafovych, A., & Grishin, N. V. (2016). CASP 11 target classification. Proteins: Structure, Function, and Bioinformatics, 84, 20-33.

[19] Krishnaveni, A., Shankar, R., &Duraisamy, S. (2019). A Survey on Nature Inspired Computing (NIC): Algorithms and Challenges. Global journal of computer science and technology: D Neural & Artificial Intelligence, 19(3).

[20] Al Kawam, A., Sen, A., Datta, A., & Dickey, N. (2017). Understanding the bioinformatics challenges of integrating genomics into healthcare. IEEE journal of biomedical and health informatics, 22(5), 1672-1683.

[21] Rajakumar, R., Dhavachelvan, P., &Vengattaraman, T. (2016, October). A survey on nature inspired meta-heuristic algorithms with its domain specifications. In 2016 international conference on communication and electronics systems (ICCES) (pp. 1-6). IEEE.

[22] Essaid, M., Idoumghar, L., Lepagnot, J., &Brévilliers, M. (2019). GPU parallelization strategies for metaheuristics: a survey. International Journal of Parallel, Emergent and Distributed Systems, 34(5), 497-522.

[23] Shandilya, S. K., Shandilya, S., & Nagar, A. K. (Eds.). (2019). Advances in nature-inspired computing and applications (Vol. 1). Switzerland: Springer International Publishing.

[24] Hessler, G., &Baringhaus, K. H. (2018). Artificial intelligence in drug design. Molecules, 23(10), 2520.

[25] Xu, J., & Wang, S. (2019). Analysis of distance-based protein structure prediction by deep learning in CASP13. Proteins: Structure, Function, and Bioinformatics, 87(12), 1069-1081.

[26] Gawehn, E., Hiss, J. A., & Schneider, G. (2016). Deep learning in drug discovery. Molecular informatics, 35(1), 3-14.

[27] Dorn, M., e Silva, M. B., Buriol, L. S., & Lamb, L. C. (2014). Three-dimensional protein structure prediction: Methods and computational strategies. Computational biology and chemistry, 53, 251-276.

[28] Torrisi, M., Kaleel, M., &Pollastri, G. (2019). Deeper profiles and cascaded recurrent and convolutional neural networks for state-of-the-art protein secondary structure prediction. Scientific reports, 9(1), 1-12.

[29] Goodfellow, I.,`Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[30] Mirabello, C., &Wallner, B. (2019). RAWMSA: End-to-end deep learning using raw multiple sequence alignments. PloS one, 14(8), e0220182.

[31] Xu, J. (2019). Distance-based protein folding powered by deep learning. Proceedings of the National Academy of Sciences, 116(34), 16856-16865.

[32] Smolarczyk, T., Roterman-Konieczna, I., &Stapor, K. (2020). Protein secondary structure prediction: a review of progress and directions. Current Bioinformatics, 15(2), 90-107.

[33] Barberis, E., Marengo, E., & Manfredi, M. (2021). Protein Subcellular Localization Prediction. In Proteomics Data Analysis (pp. 197-212). Humana, New York, NY.

[34] Moult, J., Pedersen, J. T., Judson, R., & Fidelis, K. (1995). A large-scale experiment to assess protein structure prediction methods.

[35] Fleishman, S. J., Whitehead, T. A., Strauch, E. M., Corn, J. E., Qin, S., Zhou, H. X., ...& Baker, D. (2011). Community-wide assessment of protein-interface modelling suggests improvements to design methodology. Journal of molecular biology, 414(2), 289-302.

[36] Yang, J., Zhang, W., He, B., Walker, S. E., Zhang, H., Govindarajoo, B., ...& Zhang, Y. (2016). Template-based protein structure prediction in CASP11 and retrospect of I-TASSER in the last decade. Proteins: Structure, Function, and Bioinformatics, 84, 233-246.

[37] Dhingra, S., Sowdhamini, R., Cadet, F., &Offmann, B. (2020). A glance into the evolution of template-free protein structure prediction methodologies. Biochimie, 175, 85-92.

[38] Bhattacharya, D., Cao, R., & Cheng, J. (2016). UniCon3D: de novo protein structure prediction using united-residue conformational search via stepwise, probabilistic sampling. Bioinformatics, 32(18), 2791-2799.

[39] Bienert, S., Waterhouse, A., de Beer, T. A., Tauriello, G., Studer, G., Bordoli, L., &Schwede, T. (2017). The SWISS-MODEL Repository—new features and functionality. Nucleic acids research, 45(D1), D313-D319.

[40] Remmert, M., Biegert, A., Hauser, A., &Söding, J. (2012). HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. Nature methods, 9(2), 173-175.

[41] Webb, B., &Sali, A. (2016). Comparative protein structure modelling using MODELLER. Current protocols in bioinformatics, 54(1), 5-6.

[42] Zheng, W., Zhang, C., Bell, E. W., & Zhang, Y. (2019). I-TASSER gateway: A protein structure and function prediction server powered by XSEDE. Future Generation Computer Systems, 99, 73-85.

[43] Yang, J., Yan, R., Roy, A., Xu, D., Poisson, J., & Zhang, Y. (2015). The I-TASSER Suite: protein structure and function prediction. Nature methods, 12(1), 7-8.

[44] Zheng, W., Li, Y., Zhang, C., Pearce, R., Mortuza, S. M., & Zhang, Y. (2019). Deep-learning contact-map guided protein structure prediction in CASP13. Proteins: Structure, Function, and Bioinformatics, 87(12), 1149-1164.

[45] Jayaram, B., Dhingra, P., Lakhani, B., & Shekhar, S. (2012). Bhageerath—Targeting the near impossible: Pushing the frontiers of atomic models for protein tertiary structure prediction. Journal of Chemical Sciences, 124(1), 83-91.

[46] Jayaram, B., Dhingra, P., Mishra, A., Kaushik, R., Mukherjee, G., Singh, A., & Shekhar, S. (2014). Bhageerath-H: a homology/ab initio hybrid server for predicting tertiary structures of monomeric soluble proteins. BMC bioinformatics, 15(16), 1-12.

[47] Rahul, K., Ankita, S., Debarati, D., Amita, P., Shashank, S., and B. Jayaram. (2016). Bhageerath H+ : A hybrid methodology based software suite for protein tertiary structure prediction. In CASP12 Proceedings, pages 25–26, 2016.

[48] Park, H., DiMaio, F., & Baker, D. (2016). CASP 11 refinement experiments with ROSETTA. Proteins: Structure, Function, and Bioinformatics, 84, 314-322.

[49] Kinch, L. N., Li, W., Monastyrskyy, B., Kryshtafovych, A., & Grishin, N. V. (2016). Evaluation of free modelling targets in CASP11 and ROLL. Proteins: Structure, Function, and Bioinformatics, 84, 51-66.

[50] Moult, J., Fidelis, K., Kryshtafovych, A., Schwede, T., &Tramontano, A. (2016). Critical assessment of methods of protein structure prediction: Progress and new directions in round XI. Proteins: Structure, Function, and Bioinformatics, 84, 4-14.

[51] Abriata, L. A., Tamò, G. E., Monastyrskyy, B., Kryshtafovych, A., & Dal Peraro, M. (2018). Assessment of hard target modelling in CASP12 reveals an emerging role of alignment-based contact prediction methods. Proteins: Structure, Function, and Bioinformatics, 86, 97-112.

[52] Hou, J., Wu, T., Cao, R., & Cheng, J. (2019). Protein tertiary structure modelling driven by deep learning and contact distance prediction in CASP13. Proteins: Structure, Function, and Bioinformatics, 87(12), 1165-1178.

[53] Lee, J., Freddolino, P. L., & Zhang, Y. (2017). Ab initio protein structure prediction. In From protein structure to function with bioinformatics (pp. 3-35). Springer, Dordrecht.

[54] Cao, R., Bhattacharya, D., Adhikari, B., Li, J., & Cheng, J. (2016). Massive integration of diverse protein quality assessment methods to improve template-based modelling in CASP11. Proteins: Structure, Function, and Bioinformatics, 84, 247-259.

[55] Jones, D. T., &Kandathil, S. M. (2018). High precision in protein contact prediction using fully convolutional neural networks and minimal sequence features. Bioinformatics, 34(19), 3308-3315.

[56] Kandathil, S. M., Greener, J. G., & Jones, D. T. (2019). Prediction of interresidue contacts with DeepMetaPSICOV in CASP13. Proteins: Structure, Function, and Bioinformatics, 87(12), 1092-1099.

[57] Wu, Q., Peng, Z., Anishchenko, I., Cong, Q., Baker, D., & Yang, J. (2020). Protein contact prediction using metagenome sequence data and residual neural networks. Bioinformatics, 36(1), 41-48.

[58] AlQuraishi, M. (2019). End-to-end differentiable learning of protein structure. Cell systems, 8(4), 292-301.

[59] Liu, J., Wu, T., Guo, Z., Hou, J., & Cheng, J. (2021). Improving protein tertiary structure prediction by deep learning and distance prediction in CASP14. bioRxiv.

[60] Torrisi, M., Pollastri, G., & Le, Q. (2020). Deep learning methods in protein structure prediction. Computational and Structural Biotechnology Journal, 18, 1301-1310.

[61] Gao, W., Mahajan, S. P., Sulam, J., & Gray, J. J. (2020). Deep learning in protein structural modelling and design. Patterns, 100142.

[62] Li, Y., Hu, J., Zhang, C., Yu, D. J., & Zhang, Y. (2019). ResPRE: high-accuracy protein contact prediction by coupling precision matrix with deep residual neural networks. Bioinformatics, 35(22), 4647-4655.

[63] Wang, S., Sun, S., Li, Z., Zhang, R., & Xu, J. (2017). Accurate de novo prediction of protein contact map by ultra-deep learning model. PLoS computational biology, 13(1), e1005324.

[64] Hanson, J., Paliwal, K., Litfin, T., Yang, Y., & Zhou, Y. (2018). Accurate prediction of protein contact maps by coupling residual two-dimensional bidirectional long short-term memory with convolutional neural networks. Bioinformatics, 34(23), 4039-4045.

[65] Chen, C., Wu, T., Guo, Z., & Cheng, J. (2021). Combination of deep neural network with attention mechanism enhances the explainability of protein contact prediction. Proteins: Structure, Function, and Bioinformatics, 89(6), 697-707.

[66] Senior, A. W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., ...& Hassabis, D. (2020). Improved protein structure prediction using potentials from deep learning. Nature, 577(7792), 706-710.

[67] Kinch, L. N., Schaeffer, D. R., Kryshtafovych, A., & Grishin, N. V. (2021). Target Classification in the 14th Round of the Critical Assessment of Protein Structure Prediction (CASP14). Proteins: Structure, Function, and Bioinformatics

[68] Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., & Moult, J. (2019). Critical assessment of methods of protein structure prediction (CASP)—Round XIII. Proteins: Structure, Function, and Bioinformatics, 87(12), 1011-1020.

[69] Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., ...& Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. Nature, 1-11.

[70] Murata, K., & Wolf, M. (2018). Cryo-electron microscopy for structural analysis of dynamic biological macromolecules. BiochimicaetBiophysicaActa (BBA)-General Subjects, 1862(2), 324-334.

[71] Esquivel-Rodríguez, J., &Kihara, D. (2013). Computational methods for constructing protein structure models from 3D electron microscopy maps. Journal of structural biology, 184(1), 93-102.

[72] Pakhrin, S. C., Shrestha, B., Adhikari, B., & Kc, D. B. (2021). Deep Learning-Based Advances in Protein Structure Prediction. International Journal of Molecular Sciences, 22(11), 5553

[73] Alnabati, E., &Kihara, D. (2020). Advances in structure modelling methods for cryo-electron microscopy maps. Molecules, 25(1), 82.

[74] Raval, A., Piana, S., Eastwood, M. P., & Shaw, D. E. (2016). Assessment of the utility of contact-based restraints in accelerating the prediction of protein structure using molecular dynamics simulations. Protein Science, 25(1), 19-29.

[75] Morrone, J. A., Perez, A., MacCallum, J., & Dill, K. A. (2017). Computed binding of peptides to proteins with MELD-accelerated molecular dynamics. Journal of chemical theory and computation, 13(2), 870-876.

[76] Jiang, F., & Wu, Y. D. (2014). Folding of fourteen small proteins with a residue-specific force field and replica-exchange molecular dynamics. Journal of the American Chemical Society, 136(27), 9536-9539.

[77] Li, D., Ju, Y., & Zou, Q. (2016). Protein folds prediction with hierarchical structured SVM. Current Proteomics, 13(2), 79-85.

[78] Wu, S., Szilagyi, A., & Zhang, Y. (2011). Improving protein structure prediction using multiple sequence-based contact predictions. Structure, 19(8), 1182-1191.

[79] Savojardo, C., Fariselli, P., Martelli, P. L., &Casadio, R. (2013). Prediction of disulfide connectivity in proteins with machine-learning methods and correlated mutations. BMC bioinformatics, 14(1), 1-8.

[80] Wang, Y., Cheng, J., Liu, Y., & Chen, Y. (2016, May). Prediction of protein secondary structure using support vector machine with PSSM profiles. In 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference (pp. 502-505). IEEE.

[81] Xie, S., Li, Z., & Hu, H. (2018). Protein secondary structure prediction based on the fuzzy support vector machine with the hyperplane optimization. Gene, 642, 74-83.

[82] Manavalan, B., & Lee, J. (2017). SVMQA: support–vector-machine-based protein single-model quality assessment. Bioinformatics, 33(16), 2496-2503.

[83] Uziela, K., Shu, N., Wallner, B., &Elofsson, A. (2016). ProQ3: Improved model quality assessments using Rosetta energy terms. Scientific reports, 6(1), 1-10.

[84] Ray, A., Lindahl, E., &Wallner, B. (2012). Improved model quality assessment using ProQ2. BMC bioinformatics, 13(1), 1-12.

[85] Kieslich, C. A., Smadbeck, J., Khoury, G. A., &Floudas, C. A. (2016). conSSert: consensus SVM model for accurate prediction of ordered secondary structure.

[86] Zhou, J., Yan, W., Hu, G., & Shen, B. (2014). SVR_CAF: an integrated score function for detecting native protein structures among decoys. Proteins: Structure, Function, and Bioinformatics, 82(4), 556-564.

[87] Wardah, W., Khan, M. G., Sharma, A., & Rashid, M. A. (2019). Protein secondary structure prediction using neural networks and deep learning: A review. Computational biology and chemistry, 81, 1-8.

[88] Heffernan, R., Paliwal, K., Lyons, J., Dehzangi, A., Sharma, A., Wang, J., Sattar, A., Yang,Y., Zhou, Y., (2015). Improving prediction of secondary structure, local backbone angles, and solvent accessible surface area of proteins by iterative deep learning. Sci. Rep. 5, 11476.

[89] Spencer, M., Eickholt, J., & Cheng, J. (2014). A deep learning network approach to ab initio protein secondary structure prediction. IEEE/ACM transactions on computational biology and bioinformatics, 12(1), 103-112.

[90] Wang, S., Peng, J., Ma, J., & Xu, J. (2016). Protein secondary structure prediction using deep convolutional neural fields. Scientific reports, 6(1), 1-11.

[91] Drozdetskiy, A., Cole, C., Procter, J., & Barton, G. J. (2015). JPred4: a protein secondary structure prediction server. Nucleic acids research, 43(W1), W389-W394.

[92] Torrisi, M., Kaleel, M., &Pollastri, G. (2018). Porter 5: fast, state-of-the-art ab initio prediction of protein secondary structure in 3 and 8 classes. bioRxiv, 289033.

[93] Yaseen, A., & Li, Y. (2014). Context-based features enhance protein secondary structure prediction accuracy. Journal of chemical information and modelling, 54(3), 992-1002.

[94] Li, Z., & Yu, Y. (2016). Protein secondary structure prediction using cascaded convolutional and recurrent neural networks. arXiv preprint arXiv:1604.07176.

[95] Zhang, B., Li, J., & Lü, Q. (2018). Prediction of 8-state protein secondary structures by a novel deep learning architecture. BMC bioinformatics, 19(1), 1-13.

[96] Mirabello, C., & Pollastri, G. (2013). Porter, PaleAle 4.0: high-accuracy prediction of protein secondary structure and relative solvent accessibility. Bioinformatics, 29(16), 2056-2058.

[97] Heffernan, R., Yang, Y., Paliwal, K., Zhou, Y., (2017). Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers, and solvent accessibility. Bioinformatics 33 (18), 2842–2849.

[98] Fang, C., Shang, Y., & Xu, D. (2018). MUFOLD-SS: New deep inception-inside-inception networks for protein secondary structure prediction. Proteins: Structure, Function, and Bioinformatics, 86(5), 592-598.

[99] Zhang, H., & Shen, Y. (2020). Template-based prediction of protein structure with deep learning. BMC genomics, 21(11), 1-9.

[100] Boumedine, N., & Bouroubi, S. (2019). A new hybrid genetic algorithm for protein structure prediction on the 2D triangular lattice. arXiv preprint arXiv:1907.04190.

[101] Correa, L., Borguesan, B., Farfan, C., Inostroza-Ponta, M., & Dorn, M. (2016). A memetic algorithm for 3D protein structure prediction problem. IEEE/ACM transactions on computational biology and bioinformatics, 15(3), 690-704.

[102] Dorn, M., Inostroza-Ponta, M., Buriol, L. S., &Verli, H. (2013, June). A knowledge-based genetic algorithm to predict three-dimensional structures of polypeptides. In 2013 IEEE Congress on Evolutionary Computation (pp. 1233-1240). IEEE.

[103] Borguesan, B., e Silva, M. B., Grisci, B., Inostroza-Ponta, M., & Dorn, M. (2015). APL: An angle probability list to improve knowledge-based metaheuristics for the three-dimensional protein structure prediction. Computational biology and chemistry, 59, 142-157.

[104] Borguesan, B., Inostroza-Ponta, M., & Dorn, M. (2017). Nias-server: Neighbors influence of amino acids and secondary structures in proteins. Journal of Computational Biology, 24(3), 255-265.

[105] Garza-Fabre, M., Kandathil, S. M., Handl, J., Knowles, J., & Lovell, S. C. (2016). Generating, maintaining, and exploiting diversity in a memetic algorithm for protein structure prediction. Evolutionary computation, 24(4), 577-607.

[106] Rocha, G. K., Custódio, F. L., Barbosa, H. J., & Dardenne, L. E. (2016, July). Using crowding-distance in a multiobjective genetic algorithm for protein structure prediction. In Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion (pp. 1285-1292).

[107] Gao, S., Song, S., Cheng, J., Todo, Y., & Zhou, M. (2017). Incorporation of solvent effect into multi-objective evolutionary algorithm for improved protein structure prediction. IEEE/ACM transactions on computational biology and bioinformatics, 15(4), 1365-1378.

[108] Cheng-yuan, L., Yan-rui, D., & Wen-bo, X. (2010, August). Multiple-layer quantum-behaved particle swarm optimization and toy model for protein structure prediction. In 2010 Ninth International Symposium on Distributed Computing and Applications to Business, Engineering and Science (pp. 92-96). IEEE.

[109] Zhou, C., Hou, C., Wei, X., & Zhang, Q. (2014). Improved hybrid optimization algorithm for 3D protein structure prediction. Journal of molecular modelling, 20(7), 1-12.

[110] 110. Li, B., Li, Y., & Gong, L. (2014). Protein secondary structure optimization using an improved artificial bee colony algorithm based on AB off-lattice model. Engineering Applications of Artificial Intelligence, 27, 70-79.

[111] Li, B., Chiong, R., & Lin, M. (2015). A balance-evolution artificial bee colony algorithm for protein structure optimization based on a three-dimensional AB off-lattice model. Computational biology and chemistry, 54, 1-12.

[112] Li, B., Lin, M., Liu, Q., Li, Y., & Zhou, C. (2015). Protein folding optimization based on 3D off-lattice model via an improved artificial bee colony algorithm. Journal of molecular modelling, 21(10), 1-15.

[113] Parpinelli, R. S., Benitiez, C. M., Cordeiro, J., & Lopes, H. S. (2014). Performance Analysis of Swarm Intelligence Algorithms for the 3D-AB off-lattice Protein Folding Problem. J. Multiple Valued Log. Soft Comput., 22(3), 267-286.

[114] Khakzad, H., Karami, Y., & Arab, S. S. (2015). Accelerating protein structure prediction using particle swarm optimization on GPU. BioRxiv, 022434.

[115] Wang, D., Geng, L., Zhao, Y. J., Yang, Y., Huang, Y., Zhang, Y., & Shen, H. B. (2020). Artificial intelligence-based multi-objective optimization protocol for protein structure refinement. Bioinformatics, 36(2), 437-448.

[116] Akbar, S., Pardasani, K. R., & Khan, F. (2021). Swarm optimization-based neural network model for secondary structure prediction of proteins. Network Modelling Analysis in Health Informatics and Bioinformatics, 10(1), 1-9.

[117] Kamal, M. S., Chowdhury, L., Khan, M. I., Ashour, A. S., Tavares, J. M. R., & Dey, N. (2017). Hidden Markov model and Chapman Kolmogrov for protein structures prediction from images. Computational biology and chemistry, 68, 231-244.

[118] Gao, P., Wang, S., Lv, J., Wang, Y., & Ma, Y. (2017). A database-assisted protein structure prediction method via a swarm intelligence algorithm. RSC advances, 7(63), 39869-39876.

AUTHOR INDEX