INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTING

Volume 02 Issue 02
January 2024

IJRC

International Journal of Research in Computing (IJRC)

Volume 02 Issue 02

January 2024

ISSN 2820-2139

© 2024 Faculty of Computing, General Sir John Kotelawala Defiance University, Sri Lanka.

All rights reserved.

No part of this publication may be reproduced or quoted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage or retrieval system, without permission in writing from the Faculty of Computing of General Sir John Kotelawala Defence University, Ratmalana, Sri Lanka.

Published by

Faculty of Computing, General Sir John Kotelawala Defence University,

Ratmalana, Sri Lanka

Tel: +94-11-2635268

E-Mail: editor@ijrcom.org & editorijrc@kdu.ac.lk

Website: http://ijrcom.org/

EDITORIAL COMMITTEE

CHIEF ADVISORS

Dr. HL Premarathne
Senior Lecturer (Retied)
School of Computing, University of Colombo, Sri Lanka

Dr. ADAI Gunasekara
Dean/ Senior Lecturer
Department of Computer Science,
Faculty of Computing,
General Sir John Kotelawala Defence University, Sri Lanka

EDITOR IN CHIEF

Dr. B Hettige
HoD/Senior Lecturer
Department of Computer Engineering,
Faculty of Computing,
General Sir John Kotelawala Defence University, Sri Lanka

ASSOCIATE EDITORS IN CHIEF

Prof. TL Weerawardane
Dean/Professor of Electronics and Telecommunication
Faculty of Engineering,
General Sir John Kotelawala Defence University, Sri Lanka

Dr. HRWP Gunathilake
Senior Lecturer
Department of Computer Science, Faculty of Computing
General Sir John Kotelawala Defence University, Sri Lanka

Dr. LP Kalansooriya
Senior Lecturer
Department of Computer Science, Faculty of Computing
General Sir John Kotelawala Defence University, Sri Lanka

Dr. N Wedasinghe
Senior Lecturer
Department of Information Technology, Faculty of Computing
General Sir John Kotelawala Defence University, Sri Lanka

MEMBERS OF THE EDITORIAL BOARD

Prof. Yukun Bao
Deputy Director of Center for Modern Information Systems
Huazhong University of Science & Technology, China

Prof. R.Hoque
Professor
Law at the University of Dhaka, Bangladesh

Prof. Shamim Kaiser
Professor
Institute of Information Technology
Jahangirnagar University, Bangladesh

Dr. Attaphongse Taparugssanagorn
Associate Professor
School of Engineering and Technology
Asian Institute of Technology, Thailand

Snr.Prof. AS Karunananda
Senior professor
Department of Computational Mathematics, Faculty of Information Technology
University of Moratuwa, Sri Lanka

Prof. Prasad Jayaweera
Head/Professor of Computer Science
Department of Computer Science, Faculty of Applied Sciences
University of Sri Jayawardhanapura, Sri Lanka

Assoc. Prof. Anuja Dharmaratne
Associate Head (Education) School of IT
Monash University, Malaysia

Dr. Romuald Jolivot
Research Scholar
School of Engineering
Bangkok University, Thailand

Dr. MB Dissanayake
Senior Lecturer
Department of Electrical and Electronic Engineering
University of Peradeniya, Sri Lanka

Dr. APR Wickramarachchi
Senior Lecturer
Department of Industrial Management
University of Kelaniya, Sri Lanka

EDITORIAL ASSISTANTS

Ms. DVDS Abeysinghe
Lecturer
Department of Computer Science,
Faculty of Computing,
General Sir John Kotelawala Defence University, Sri Lanka

Ms. KD Madhubashini
Instructor
Department of Computational Mathematics,
Faculty of Computing,
General Sir John Kotelawala Defence University, Sri Lanka

Proofreading

Ms. DVDS Abeysinghe
Lecturer
Department of Computer Science,
Faculty of Computing,
General Sir John Kotelawala Defence University, Sri Lanka

Ms. KD Madhubashini
Instructor
Department of Computational Mathematics,
Faculty of Computing,
General Sir John Kotelawala Defence University, Sri Lanka

CONTENTS

Exploring Mechanisms for Detecting Violent Content in Sinhala Image Posts: Rationale with Unsupervised vs Supervised Techniques

U Dikwatta[1#], TGI Fernando[1], and MKA Ariyaratne[2]
[1] Department of Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Sri Lanka
[2] Faculty of Information Technology and Communication Sciences, Tampere University, Finland

[#]umanda@sjp.ac.lk

ABSTRACT This research explores the different avenues in machine learning to classify Sinhala image posts. Image posts in social media are one big weapon that conveys information directly to people. Image posts contain both visuals and text. English based research work is common in this regard, but only a handful can be seen from other languages. The target language was a low-resource language, Sinhala. Unsupervised algorithms were used to classify image posts and supervised algorithms were involved classifying manually extracted text in image posts. The classification decides whether the posts are violent or nonviolent. The trained supervised models were tested with interpretability models to identify the words that cause the decision of violent or nonviolent. The findings reveal supervised algorithms perform better than unsupervised algorithms in classifying image posts. However, improved results can be obtained by increasing the size and the variety of the dataset.

INDEX TERMS Deep learning, machine learning, social media, violence detection

I. INTRODUCTION

People use social media as a powerful tool for communication. The birth of 2-way communication began with Web 2.0 and has been evolved, so now people who use social media can modify the contents as well as provide their thoughts towards enormous topics [30]. As with useful and entertaining content, social media also provides a platform for users who spread violent content that disperses violence to the physical world. One of the earliest examples of such behaviour can be pointed out through a case study done in 1997, which was based on an incident in Bangladesh where violence originated in social media [42]. Also in 2008, image posts about the resentment of immigrants in Italy were circulated through social media [45].

As images can talk to people way faster than the words, image posts have become very popular in social media to convey ideas; to spread good as well as the opposite. Hence, image posts shared on social media have become a powerful mechanism to disperse violence. They contain visual and text elements. One of the main concerns even from the early days of social media is to identify smart mechanisms for early detection of such poor posts and help to clean the social media platforms from such contents. Based on such concerns, there are many research attempts. However, most of such findings are based around image posts having text elements in the English language. A handful of research can be found from other languages, and we have listed them in section 2.

In natural language processing, languages are categorized by whether they are high or low resource. Low resource languages lack data that can be used for machine learning (ML) or other processing, and high-resource languages are rich in available data. With the birth of the Unicode character system, usage of law resources languages has accelerated in a noticeable way. That directly affects the usage of such languages in social media as well. Image posts with violence; what we are interested in this research can also be seen with text elements in such low resource languages.

In this work, we aimed to work on a low-resource language and chose ‘Sinhala’ as the preferred low-resource language. We concern text elements inside Sinhala image posts. As the literature revealed most of the text classification research works focused on “Sinhala” were based on Facebook comments or Tweets. In those works, they have used the row text form user comments. This research is different from the input from most of such research. Most of the research on the text classification connects with ML techniques since data are concerned as the main resource at hand in decision making [23]. Going along the same line, this research also focused on using unsupervised and supervised ML techniques to identify the desired image posts.

This research work has mainly four phases. First, we modified the dataset introduced in previous research to include a balanced dataset that provides violent and nonviolent image posts [20]. Then, we retrained the unsupervised models as an anomaly detection problem introduced in earlier research on the new dataset [20]. Next, we used supervised machine learning algorithms such as shallow learning and deep learning (DL) to classify the manually extracted text. Going beyond acquiring traditional training testing accuracies as goodness of measures, as the last phase, in this research we tried using explainable AI processes that allow human users to trust the results and output created by ML algorithms. By following the phases, the aim of the research, which is on exploring the capabilities of supervised and unsupervised machine learning techniques to detect violent context in Sinhala images posted was achieved.

For the convenience of the readers, this paper is arranged as follows. In the next section, we present the literature relevant to the current research. Section III lists used materials and methodologies in our work. In section IV, we present our research results and finally in section V we discuss our findings, present the drawn conclusions, and point out future possibilities.

II. RELATED WORKS

Violence detection in social media encompasses several distinct categories such as modality based, classification algorithm based, and language based. Modality-based violence detection has four distinct categories: text, images, videos, and multi-modal approaches that combine both textual and visual elements. However, this research primarily focuses on social media images, and we have not extensively discussed the techniques related to videos. In addition to the modality-based categorization, violence detection utilizes classification algorithms that employ ML and DL techniques. Furthermore, violence detection encompasses several language-based studies: English, Sinhala, and other low resource languages such as Arabic. The literature includes scholarly studies from the earliest publication in 2014 to the most recent. We present previous work by the language, the research work has been focused, from high level languages to low level languages.

A. Related Research on English Language

Hate speech is one way of spreading violence in social media. Different organizations, communities and social media sites have given different definitions to hate speech. Hate speech can be defined as a set of terms in a defined language that attacks a person or a group of people regarding religion, ethnicity, gender, sexual orientation. These hateful contents can persuade people to violence. Hate speech can be detected in several ways; however, many research studies were based on machine learning techniques as data is incorporated with the process. When it comes to ML, feature extraction is one of the main tasks in the process of decision making. Review works have pointed out, in ML, feature extraction for hate speech detection has been done using several approaches. Bag-of-words (BoW), term frequency - inverse document frequency (TF-IDF), rule based, n-grams, word embeddings, and topic classification methods are some of them. Further, the reviews discussed the contribution of shallow ML algorithms like support vector machines (SVM), naïve Bayesian (NB), logistic regression (LR), decision trees (DT) to the process of hate speech detection. Authors also point out the classification mechanisms used in detecting hate speech using deep learning algorithms. However, the comparisons among methods were not discussed as most of the newly created datasets are not published and publicly available [23, 55]. Anusha Chhabra and Dinesh Kumar Vishwakarma presented another review on multi-modal and multilingual social media hate content detection using shallow and deep learning ML models [10]. The findings of the survey point out; almost all the past reviews were conducted covering text-based hate speech detection studies, only two datasets were identified as multi-modal: text and image based, and DL approaches have outperformed shallow learning approaches.

Going deeper into actual works performed for the English language, one of the earliest works proposed a mechanism using paragraph2vec [31] and continuous bag-of-words (CBOW) [39] for on-line user comments in Yahoo Finance website to hate speech detection. LR was used as the classification algorithm. The proposed method has given higher Area Under the Curve (AUC) than existing BoW methods [21].

Tweets are an interesting and powerful communication mechanism among a lot of people. Hate speech is also a frequent content in tweets. In a research study, hate speech in Tweets was investigated as a multi-class problem with three classes: hate (strongest hate level), offensive and neither [15]. Five shallow ML algorithms; LR, linear SVM, NB, DT, random forest (RF) were used to build the models. The results showed that LR and linear SVM performed better than the other three algorithms. L2 regularization (Ridge Regression) combined with LR improved the accuracy of the normal linear regression model. Going beyond the shallow algorithms, in [5], has used deep neural networks (DNN) to detect hate speech in tweets. Convolutional neural network (CNN), long short-term memory (LSTM) and fastText were used as feature spaces. Precision, recall and F1 score were used to compare the results. DNN achieved better results than shallow machine learning methods that utilized Char n-gram, TF-IDF and BoW embeddings. Higher results were obtained for utilizing random embeddings trained with LSTM and using them in a gradient boosted decision trees (GBDT) algorithm for classification.

Another research used English as a language to test the performance of different feature extraction methods combined with Linear SVM model [36]. Character n grams, word n-grams, and skip grams were used as feature extraction methods to detect hate speech. The Character 4-gram method has shown better results than other methods and achieved an accuracy of 78%. However, previous research done by Malmasi et al., claimed better results than the 4-gram method using an oracle ensemble method with SVM [35]. Data is crucial for any machine learning process, so hate speech detection. Won et al.

has formed a protest image dataset [72]. They have employed a ResNet based model to violence detection in images and OpenFace based model [4] for emotion detection of people in violent scenes. They have found their model performs well in identifying violent scenes but does not perform well for emotion detection. As same, Sun et al. have created a new dataset that consists of still images related to violence and nonviolence [60]. They have used low level features as the multi views in their dataset along with features extracted from CNN. Low level features include dense scale invariant feature transform (DSIFT), histogram of oriented gradient (HOG) and local binary pattern (LBP). Authors have proposed new multi view maximum cross entropy discrimination.

Watanabe et. al. introduced a new feature extraction method that incorporates sentiment, semantic, unigrams and pattern feature to identify hate text in Twitter [70]. They have used ‘‘J48graft’’[69], SVM and RF as classifiers. ‘‘J48graft’’ has outperformed the other two classifiers. The new model ‘‘J48graft’’ is an extension of decision tree grafting algorithm that increases the performance of the original algorithm with respect to both bias and variance. Further Z. Zhang and L. Luo addressed the problem of “long trail” in hateful text in social media, specifically in Twitter [73]. This research has proposed a model that incorporates two DNN architectures with CNN and Gated Recurrent Unit (GRU). This method has surpassed state-of-the-art methods on a Twitter dataset and established a new benchmark for future research that involves identifying hate speech.

Gang violence, one of the other types that create textual as well as visual modality, can be defined as criminal and non-political acts of violence committed by a group of people who regularly engage in criminal activity against innocent people. Research was also carried out around this area and multi-modal approaches were used to detect images with gang violence [7]. They used tweets to create the datasets that were annotated with psychosocial codes, aggression, loss, and substance use. Text features were detected using unigram, bigram, Part-of-Speech (POS), and CNN features. Regional-based convolutional neural network (R-CNN) was used to detect image features. Fusion methods: early fusion and late fusion were used as multi-modal feature extraction methods. Text feature classification showed better results for loss code where image features classification showed better results for aggression and substance codes. Fusion method has shown promising results in this research.

Amorim et al. introduced novelty detection in a temporal window using data fusion technique [3]. The objective of this approach is to detect comments that stand out from others within a given time frame considering both present and past comments. The dataset used in this study consists of posts from social media platform Twitter. Architecture comprises three key components: feature extraction from images and text, data fusion and unsupervised algorithm. Two distinct architectures were employed by rearranging the order of three key components. In the first architecture, input to the architecture is a data stream and MASK-RCNN [26] is the data fusion algorithm that converts the stream into textual representation. Then an autoencoder was employed to convert the textual representation into a vector and unsupervised algorithm was employed to classify the vectors. Second architecture, transform tweet images and texts into vectors using an autoencoder. Then an unsupervised algorithm identifies novelties using the vectors of images and texts. Finally, the AOM [1] fusion algorithm was employed to fuse the scores obtained from the unsupervised algorithm. Results depict that MASK-RCNN method outperforms AOM method.
Suryawanshi et al. proposed a novel system to detect offensive memes by leveraging multi-modal data: text and images [63]. Going beyond text and visual datasets, authors have curated a new dataset including memes that contain both text and images. They suggested an early fusion method that incorporates stacked LSTM, BiLSTM and CNN for text features and visual geometry group (VGG-16) for visual features. Results demonstrate that the multi-modal approach has outperformed methods that incorporate only a single mode in terms of precision, F score and recall.

In [41], a Bidirectional Encoder Representations from Transform (BERT)-based transfer learning method has been introduced for hate speech detection. The new method consists of different fine-tuning approaches, adding nonlinear layers, adding Bi-LSTM layers, and adding CNN layers. The fine-tuned BERT-based method has produced better results than other state-of-the-art methods such as character n-gram with LR [67, 15], CBOW with multi-layer perceptron feed forward neural network [68], and original BERT model. Further in a comparative study conducted on hate speech detection using 14 shallow and DL models with three commonly used datasets revealed that BERT-based models outperform other methods, and the TF-IDF-based classifier outperforms other DL models [34]. In another work, authors proposed an ensemble method that employs a combination of a fine-tuned BERT based model and a parallel recurrent model for multi aspect hate speech detection [37]. The proposed model was compared with pooled stacked Bi-LSTM, Bi-GRU models and ensemble models that combine the outputs of BERT, Bi-LSTM, and BI-GRU. The new model yielded better results compared to other methods. In a recent research work, authors have proposed a multi-modal fusion mechanism to combine both text and visual features for classifying fake news [65]. They have obtained a dataset along with their captions. A fine-tuned BERT model was used to classify text and higher results were obtained when compared with other DL models. Fine-tuned Xception network has obtained higher results for visual feature classification. Concatenate fusion techniques have obtained higher results than other fusion techniques. Fusion methods have achieved higher results than using text or visual solely.

To fulfil the lack of data sets containing fight images, authors in [2] have developed a new still fight image dataset collected from social media sites. They have used DL networks like VGG-16, residual network (ResNet50), ResNeXt50, and vision transformers (ViT Large 16) for the classification and ViT network has surpassed the results of other models. Most of the violent scene detection experiments were done for video-based datasets. As the next phase of the research, they have compared the results obtained for temporal models with frame-based models that were trained. Authors have done a cross-dataset experiment to evaluate which model generalizes well with all the datasets. Models that are trained for still images generalize better than the models trained for video-based datasets. We have studied a few but mostly relevant literature which were based on the English language. Compared to English, research work on other languages is limited.

B. *Related Research on Sinhala Language*

In one of the earliest works that touch Sinhala for the first time, English comments on a Sri Lankan website were investigated for hate speech [52]. NB, SVM, LR, DT, and k-means were tested with BoW and TF-IDF. NB with TF-IDF achieved a better F-score than other methods. In another previous study, researchers successfully identified racist Sinhala comments using a two-class SVM and n-gram approach, achieving over 70% accuracy [19]. The dataset comprised randomly selected Sinhala comments from social media platforms. However, the performance declined as the dataset size increased. Identifying abusive comments in Sinhala language was also tried in research [54]. SVM, Multinomial NB (MNB), and random forest decision tree (RFDT) were used as classifiers. BoW, word n-gram, character n-gram, word skip-gram were the feature extraction mechanisms. MNB showed better results than other classifiers. Character tri-gram and character four-gram showed better results than other feature extraction methods. Corpus-based approaches showed better results.

A multi-level and two-level hate speech classification was done for Sinhala social media comments [53]. Authors have mentioned the difficulty in finding a proper data source for Sinhala. CNN and SVM were used as classification algorithms. CNN has shown higher results for binary classification. SVM has shown higher results for multi-level hate speech classification. According to the authors, a lower F1 score is achieved due to the imbalance dataset.

For the first time in Sinhala language, images were used in [59] to classify Sinhala hate text in images. Several ML techniques were used to model the data. The text has been automatically extracted from images. MNB has shown better precision, recall, and F-measure than other ML techniques.

Adapter-based pre-trained multilingual models have been proposed for code mixed and code-switched text classification that includes Sinhala text [49]. The cross-lingual representation of robustly optimized BERT pre-training approach (XLM-R), with basic fine-tuning, has outperformed all other models. XLM-R with adapters has further improved the results. BERTifying Sinhala is an analysis carried out to evaluate the performance of XLM-R, Language-Agnostic BERT Sentence Embedding (LaBSE), and Language Agnostic SEntence Representations (LASER) in Sinhala text classification [18]. There, XLM-R has performed better than other models.

This summary covers the limited research carried out on violence detection in Sinhala. It highlights the pressing need for further research in low-level languages like Sinhala.

C. *Related Research on Other Languages*

Arabic, Bengali, Italy can be identified as other languages that have contributed more on hate speech detection research. In reference [43], the authors constructed a dataset for Arabic by gathering data from popular social media networks. They utilized this dataset for hate speech detection purposes. They have performed data filtering to clean the dataset. Dataset was annotated. Then the dataset was trained and tested with ML and DL models. Complement NB surpassed other ML models for accuracy, F1 score, recall and precision. RNN outperformed CNN in DL models. Regarding the previous datasets on Arabic, the dataset collected in this research has given higher accuracy. Further in [44], authors have developed an Arabic dataset for topic classification, sentiment analysis, and multi-label classification of on-line social media networks (OSNs). Removing tokens beyond a specific length, removing stop words and stemming were performed as preprocessing steps. BoW, n-gram, TF-IDF were used as feature extraction methods. Shallow ML algorithms were used. Authors have incorporated grid search to select the best set of hyper parameters. Chi-square feature selection and hyper parameter tuning has improved the results. n-gram (1,2) with linear support vector classification (LinearSVC) has obtained higher results in topic classification. LR with BoW has yielded higher results on sentiment classification while TF-IDF with LinearSVC showed higher results for multi-label classifiers. Authors have also found a relationship between hate speech and OSNs post topics. Their proposed mechanism yielded 83.7% accuracy in filtering Facebook posts.

In another study related to Arabic, proposed a new mechanism to detect contradictions in Arabic sentences, a special scenario of natural language inference (NLI) [29]. Authors have created a dataset consisting of more than 6,000 sentence pairs of Arabic language. Their dataset consists of three different classes: contradiction, entailment and neutral. They augmented the dataset by automatic translation using two existing datasets. Feature extraction models used were word embedding mechanisms and language level feature extraction methods. SVM, stochastic gradient descent (SGD), DT, adaptive boosting (AdaBoost), k-nearest neighbour (KNN) and RF were used as classification methods. They have evaluated the results on their original dataset and two translated datasets. Obtained results convince higher accuracy for RF classification that employs BoW vector with contradiction vector.

Regarding Bengali language, research was conducted to evaluate the performance of multi-class sentiment classification on Bengali text [25]. Authors proposed a system that employs CNN and LSTM architectures. They have built a Bengali text dataset of size 42,036 social media comments that has four different classes. Authors have selected MNB, LR, DT, RF, SGD, and SVC along with their word embedding mechanisms like TF-IDF and count vectorizer (CV). LSTM, Bi-LSTM, Bi-GRU and a model that employs both CNN and LSTM (C-LSTM) were used along with word embedding as DL architectures. C-LSTM has outperformed other baseline methods.

We have summarized the related work in the context of hate speech detection mainly with the involvement of shallow and complex machine learning techniques. For convenience, we categorize our findings language wise. The findings opened the avenues and pointed out the importance of conducting more research on low level languages such as Sinhala, which was tried to achieve in the current research.

III. MATERIALS AND METHODS

Here, we present a detailed description of how our research has been conducted. As the first step of the study, we have composed a dataset that includes Sinhala violent and nonviolent images mainly collected from Facebook. We have employed two approaches to classify the dataset into two categories, nonviolent and violent. The first approach is clustering where images are fed to unsupervised algorithms. The second approach is to utilize manually extracted textual parts of the images to train supervised learning algorithms. To train supervised ML algorithms, the dataset was annotated as nonviolent and violent. To evaluate the results, we employed four metrics commonly used in ML studies: accuracy, precision, recall, and F1-score. One drawback on ML algorithms is that they are like black boxes, and we do not know why a ML model predicts a text as violent or nonviolent, which words in the text caused the decision. To find out which words caused the decision, in this research, we further employed explainable AI (XAI) methods such as local interpretable model-agnostic explanations such as (LIME) [50] and Shapley additive explanations (SHAP) [33] and integrated gradient (IG) [61]. The overall process of supervised learning is depicted in Figure 1.

A. Dataset collection

All images were manually downloaded from Facebook. We found Facebook groups and their pages that are specialized for different topics that are related to our study. We used such pages to download images and we have also used keyword search to download violent posts. We identified commonly used violent words and treated them as keywords. The final dataset consists of 3,463 nonviolent and 3,465 violent images. Figure 2 and Figure 3 depict a nonviolent image and a violent image respectively.

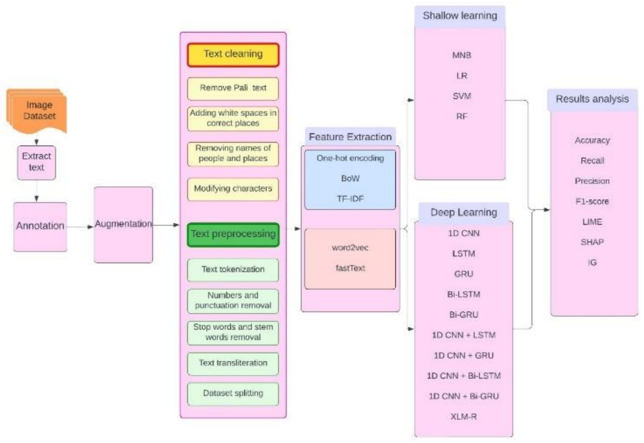


Figure 1. Supervised learning training process

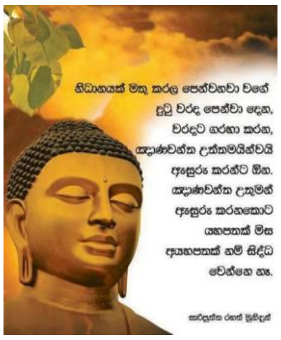


Figure 2. Image nonviolent content

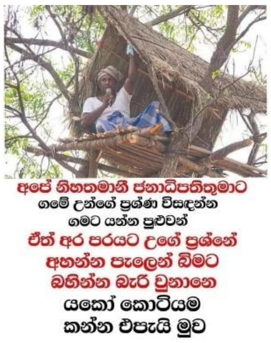


Figure 3. Image violent content

As the study was conducted mainly based on two types of ML algorithms, two types of data preparation were needed. For the unsupervised algorithms, mainly more nonviolent data were collected. For that basically a subset from [20] is used. 2,463 nonviolent image posts were used to train the unsupervised algorithms.

For the supervised algorithms, a different data handling process is employed. For that, a dataset was constructed by combining the data collected from previous research [20] with additional violent image posts. Textual portions of the images were extracted manually to perform the text classification. Unicode characters were utilized during extracting the text parts since Unicode characters provide device and platform-independent characters. For the supervised algorithms, data collection was followed by Data Annotation, Data Augmentation, Data cleaning, and Data Preprocessing.

1) Data/Images Annotation:

Two volunteer annotators annotated the dataset. The annotation process used the guidelines described in previous research [20]. If either the textual or visual component exhibited violence, the post was labelled as violent. A post can be identified as violent if it contains content that abuses a religion, race, or other beliefs; targets individuals or groups causing emotional harm or displays sexism. The posts that contain sarcasm were regarded as violent as it can inflict emotional distress on individuals or groups. Cohen's kappa was calculated to evaluate the agreement between two annotators, and scikit-learn library was used for the Cohen's kappa calculation [12]. The calculated Cohen's kappa value for the dataset was 0.9.

2) Data/ (Images, Text) Augmentation:

To enhance the accuracy of deep learning models, a larger dataset is expected. In the realm of ML, expanding a dataset using the existing samples is referred to as data augmentation [22, 58]. In the context of unsupervised learning our inputs were entire images. Therefore, we performed data augmentation techniques for the images such as random rotation, colour jittering and random horizontal flipping to generate additional variations of the existing images. We used random rotation to rotate images by a random angle, colour jittering to change the brightness, contrast, saturation, and hue of images randomly and horizontal flipping flip images horizontally for a given probability. When applying augmentation to our dataset, we followed a novel technique: we individually applied the three augmentation techniques to our training dataset, then concatenated the resulting datasets with the original dataset and achieved 9,852 as the size of the final training dataset.

For the textual components, a technique called back translation was employed to expand the dataset. 1,000 violent and 1,000 nonviolent texts were randomly selected for back translation. The initially selected text was translated into English and was subsequently translated back into Sinhala. Augmented data was re-evaluated to compare the original text with augmented text. Text that was augmented with a wrong meaning was manually corrected. Python translation was used in the translation process and the resulting text was written to a Microsoft Excel sheet. The augmented text and the text used to create augmented text were included only in the training dataset and for the testing, separate set of text were used. Following the augmentation, we achieved 8,907 as the size of our final dataset.

3) Data/Text Cleaning:

Text cleaning process in this study consists of several steps including removing Pali text, adding white spaces, removing names, and modifying characters. Pali text is a Middle Indo-Aryan language mainly used in Theravada-Buddhism. Buddhist monks in Sri Lanka use this language to chant prayers. Pali text included in most Buddhist posts was removed from the dataset. Usually, punctuations follow a white space in professional writing; however, the standard rules are not followed in most amateur posts. The tokenization process returns a different output when white spaces are not included in the correct places. Therefore, regular expressions were used to make sentences accurate.

In Sinhala alphabet, න, ණ, ල, and ළ are consonants. න and ණ, as well as ල and ළ have the same sound. Although the letters have the same sound, they cannot be used interchangeably. However, people who speak the language use these letters interchangeably due to a lack of knowledge of using them. It is difficult to memorize the places where these letters are used. Hence, these two letters are misused in Sinhala writing and in image posts. That is, න is used in cases where the letter ණ is expected, and vice versa. Similarly, ල is used in cases where the letter ළ is expected and vice versa. Therefore, all the text, including ළ, was modified to ල, and all the text, including ණ, was modified to න.

4) Data/Text Pre-processing:

Data/Text pre-processing is followed by a series of steps such as tokenization, removal of numbers and punctuations, stop words and stem words, text transliteration and dataset splitting.

The first step of text pre-processing is to split the text from white spaces. The split texts are called tokens. The research was conducted with "word_tokenize" in the natural language toolkit (NLTK) and "SinhalaTokenizer" from "sinling" [56].

Numbers and punctuation were removed from the dataset. Stop words and stem words prominently used in the Sinhala language were filtered out. Stop words such as ඒ, මේ, නම්, ඇති, එක, කර, හා, නෑ, වන, වූ, ද, බව, ගැන, කරයි, අතර, යන, ලෙස, නිසා are used. English meaning of these words respectively is "That, this, if, one, done, and, no, is, was, the, that, about, does, between, going, as, because". Stem words such as මම, මට, මටම, මටත්, මගේ, මගේම, මාගේ, මාගෙ, මාව, මාවම, අපි, අපිම, අපිවම, අපට, අපටම, අපව, අපවම, අපේ, අපේම are filtered out. "I, me, myself, mine, my own, we, ourselves, our" are the English translations of stem words.

Python Unidecode function was employed to obtain the transliterated text as a preprocessing step to check any improvement in the performance [74]. Unicode characters are fed into the Unidecode function and converted to ASCII characters.

The training process commences by initially partitioning the dataset into training and testing sets with a 4:1 ratio using the scikit-learn command to split the data [46]. Subsequently, we selected the models with higher results for further testing. In the subsequent step, the dataset was partitioned into training, validation, and testing subsets with 8:1:1 ratio. The selected models underwent further evaluation with the updated dataset partitioning. The PyTorch data loaders were created for training, validation, and testing datasets. For the unsupervised training process, the training set consists only of nonviolent images. A subset of nonviolent images from our new dataset was selected as the training dataset. As for the validation and testing datasets, 500 images from each category were selected.

B. Hardware, software, libraries, and technologies used.

PyTorch was used as the ML library and Jupyter Notebook as the platform. Experiments were conducted on an Nvidia RTX – 3090 64 GB server.

C. Evaluation metrics used in the research

Evaluation metrics used are accuracy, precision, recall, and area under the ROC curve (roc_auc_score) [28]. The confusion matrix is also used.

D. Unsupervised Learning

The dataset was used as it is to be fed into the unsupervised learning algorithms. There are many algorithms under unsupervised category. We have focused our study on autoencoders. Autoencoder is an unsupervised learning algorithm. The autoencoder architecture contains an encoder and a decoder. When an image is fed to the encoder, the decoder will attempt to regenerate the image. The loss function of autoencoders is defined as the difference between the original and the regenerated image (reconstruction loss). Autoencoders are trained using a specific type (nonviolent) of data, allowing them to learn patterns inherent within that dataset. Trained autoencoder can regenerate the type of data it has trained. If the type of data, we used in training is nonviolent then the autoencoder will give a lower reconstruction loss for nonviolent data in testing dataset, meaning that it recognized the nonviolent images properly. The autoencoder is not trained for violent images and unable to identify the pattern in violent images; therefore, a higher reconstruction loss is expected. Here, the autoencoder acted as an anomaly detection method where violent images act as the anomalies.

After training an autoencoder, we fed the validation dataset with both violent and nonviolent images to the trained autoencoder, obtaining the reconstruction loss of the images in the validation set. The reconstruction loss was acquired as a vector. Subsequently, we utilized an SVM to classify the reconstruction loss. Finally, the testing images were passed through the trained autoencoder to obtain their reconstruction loss as a vector. This vector was then fed into the trained SVM to evaluate the performance.

A previous study has found that an autoencoder utilizing GoogleNet transfer learning and convolutional layers give better results for violent and nonviolent image recognition than other autoencoders [20]. We have utilized the same autoencoders proposed in [20] to evaluate the results on our new dataset.

E. Supervised learning - Shallow learning

Before employing supervised learning-shallow learning on pre-processed data, the feature extraction step needs to be completed. For the feature extraction, feature engineering techniques were used.

1) Feature engineering:

The text must be represented in a numerical format to feed text to natural language processing (NLP) and ML algorithms; this is known as feature engineering. The text can be represented with a vector of numbers known as a vector space model. Popular vector space models are BoW, TF-IDF, and one-hot vector encoding. These models aim to obtain similar representations for similar tokens of text. All three methods have sparsity problems that are inefficient to handle in the computer memory and out-of-vocabulary problems.

First, the vocabulary that contains all tokens in the corpus was created. The vector size is |V| as V is the number of unique tokens in the corpus. In one-hot encoding each token is represented by a vector of length |V|, and a sentence is a combination of all vectors of the tokens in the sentence. As different sentences in the corpus have different lengths, vector size varies with each other. One-hot encoding ignores the similarity between words [66].

The order of words and context are not considered in BoW representation, and it considers a sentence or a document as a bag of words. Vocabulary is developed as in the one-hot representation, and the number of occurrences of each word in the sentence can be stored in the vector representation. BoW does not represent each word as a vector; it represents the whole document as a vector without considering the order of words. This representation has a fixed length for all documents in the corpus. Documents with similar words can be identified using BoW, though different words with similar meanings cannot be identified. Bag-of-n-grams can help obtain a semantic meaning between words [66]. "Countvectorizer" function in scikit-learn was used to implement the BoW method. TF-IDF is another text representation method with two terms: TF explains the importance of a word within a document, and IDF explains the importance of the same word concerning other documents in the corpus [66]. "TfidfVectorizer" in scikit-learn was used to implement TF-IDF.

2) Classification Algorithms:

Encoded data were fed into ML algorithms such as SVM, LR, NB, and RF. SVM computes the optimal hyperplane by maximizing the margin between support vectors and LR computes a line according to a sigmoid function [14, 38]. For LR, Gradient descent or maximum likelihood can act as the optimization algorithm [32, 51]. NB is based on the Bayes theorem that assumes all features are independent (of each other). NB is a generative algorithm where the posterior probability is calculated with a model that implements a joint distribution of X and Y. Equation 1 can be derived for the Bayes classifier; it can be categorized as Gaussian or multinomial, depending on the different distributions of $\mathrm{P}(x_i/y)$ [66].

$$\mathrm{P}(y/x_1, \dots, x_n) \; \alpha \; \mathrm{P}(y) \prod_{i=1}^{n} \mathrm{P}(x_i/y) \text{ ------------ Equation 1}$$

RF is a method that uses many uncorrelated decision trees to make predictions. More accurate predictions are received when each decision tree is independent of one another. RF implements bootstrap aggregation (bagging) that results in a crucial difference in the output by inputting a training set with minor changes [9]. The scikit-learn library was used for implementing shallow learning algorithms.

3) *Sampling methods used in shallow algorithms*

Although the data is divided into train and test, parameters in algorithms can be tweaked to give better results for the test set. A validation set was derived again from the train set to prevent the situation. Having three sets as train, validation, and test minimizes the data that can train the model. Cross validation, stratified sampling (an extension of cross validation), and re-sampling (a bootstrapping procedure) were used to solve the problem. The training set is divided into K folds. K-1 folds were used to train the model and the remaining *K*th fold was used to validate the model in cross-validation. The scikit-learn's "StratifiedKFold" was used in stratified sampling. Stratified sampling is an extension of cross-validation that uses stratified folds. Re-sampling that uses a bootstrapping method selects a sample with a pre-defined sample size. The model was trained on the selected sample and the model was tested on the data, which is not selected for the sample. The process can be repeated many times, and mean estimates can be obtained by averaging the values over the number of samples.

F. Supervised learning - Deep learning

1) *Text padding and vocabulary creation:*

The training set was tokenized into words, and a vocabulary was created for the training set. In the vocabulary, a unique ID was assigned to each word. The maximum length of sentences was selected depending on the number of tokens. Sentences were padded depending on the difference between sentence length and maximum length. The same vocabulary was used for the test set and assigned with IDs. Unknown tokens were assigned for words that were not in the vocabulary.

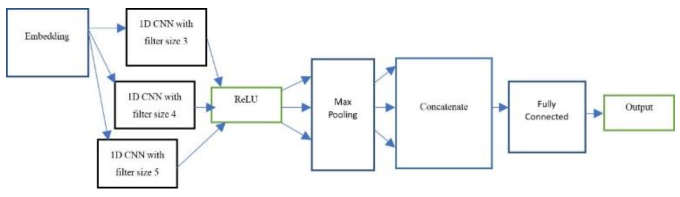
2) *Feature engineering:*

Pre-trained word embeddings were loaded after the vocabulary creation and text padding. A matrix was implemented with vocabulary size (as the row dimension) and embedding size (as the column dimension). Subsequently, distributed representations of text known as word2vec [40] and fastText [8] were used as the embedding mechanisms for deep learning algorithms. A Sinhala dataset created in previous research was also used to create new embeddings in conjunction with a random subset of the dataset collected in our research [48, 57]. However, the embedding models created using our dataset did not perform well. Text that was converted using the Unidecode library in Python and text without the conversion was also applied to generate word2vec and fastText models. However, by comparing the obtained accuracies, finally, pre-trained embeddings obtained from previous research were used [16, 57].

3) *Classification Algorithms:*

1D CNN [64], LSTM [27], GRU [11], bidirectional LSTM (BiLSTM) [24], and bidirectional GRU (BiGRU) [6] were utilized as deep learning algorithms. Ensemble methods, 1D CNN with LSTM, 1D CNN with GRU, 1D CNN with BiLSTM, and 1D CNN with BiGRU were also tested to evaluate the performance. Filter size, number of filters, number of layers, optimization algorithms, and number of epochs were modified to find the optimum result in 1D CNN. The size of the hidden layer, number of layers, and number of epochs were modified in the LSTM and GRU to find an optimum result. The learning rate was reduced to prevent overfitting. The output of 1D CNN layers with different filter sizes as 2, 3, 4, 5, 7, and 11 were concatenated. Figure 4 shows the architecture of 1D CNN. The output was sent through a fully connected layer to obtain the final output.

In Ensemble architectures, output obtained in 1D CNN was fed through recurrent models such as LSTM and GRU. The ensemble model, which combines 1D CNN and GRU, is depicted in Figure 5. Text with an embedding dimension 300 is fed to the model. Three 1D CNN filters are used to extract the features, followed by a max pooling layer. The outputs obtained from the three filters are concatenated. The concatenated output is reshaped and sent through a GRU layer. The output obtained from GRU is fed to a fully connected layer, resulting in the classification output.



Embedding
1D CNN with filter size 3
1D CNN with filter size 4
1D CNN with filter size 5
ReLU
Max Pooling
Concatenate
Fully Connected
Output

Figure 4. 1D CNN architecture

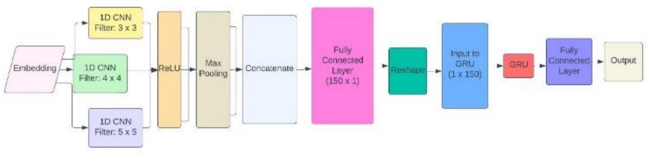


Figure 5. CNN GRU architecture

The cross-lingual representation of robustly optimized BERT pre-training approach (RoBERTa) (XLM-R) XLM-R was used as the BERT architecture, which is trained for 100 different languages and Sinhala is also included in these 100 languages [17]. Cross-lingual language model (XLM) was introduced to support 100 languages [13]. XLM uses Byte-Pair Encoding (BPE) to gain the sharing capability. In BPE, frequently used sub-word pairs are merged so they can easily represent an unknown word with sub-words that are already in the vocabulary. XLM-R, which works similarly to XLM, was trained according to the RoBERTa; RoBERTa uses a masked language model (MLM). The model was trained for the training set using "AdamW" optimization function and tested with a test set. The Hugging Face library which was implemented using PyTorch helped to access BERT interfaces [71]. The dataset in XLM-R was loaded and tokenized using a sentence piece tokenizer. All BERT algorithms expect sentences in the corpus to be tokenized in a distinct format. XLM-R requires similar formatting. The three special tokens used in BERT architecture are [CLS] as the classifier, [SEP] as the separator, and [PAD] as the padding. In XLM-R, the main tokens are <s> to indicate the beginning of a sequence, <\s> to indicate the separation of sequences and the end of a sequence, and <pad> as the padding. The method “encode_plus” returns the padded token list and attention mask. Attention mask indicates the separation between real tokens and padded tokens. “XLMRobertaTokenizer” was used as the tokenizer, and the “XLMRobertaForSequenceClassification” model was defined as the model for XLM-R [71].

With these deep learning techniques, early stopping was used as a promising technique to avoid overfitting and to find the most suitable model [47, 62].

G. *Explainable AIs*

Most ML models are black boxes; hence inner workings are not visible. Therefore, LIME and SHAP were used to describe the decisions taken by the black boxes [50, 33]. Using graphical pictures and details provided by the explainable APIs, texts that influenced the decision of the ML algorithms can be identified. ‘LIME’ model is a local approximation of the ML model. An instance in the dataset was selected, and the sample size in the LIME was initialized. The default sample size is 5,000, and better results can be obtained as the sample size increases. According to the sample size, the instance was perturbed by removing some of the tokens in the instance to create a sample. The sample obtained by perturbation was inputted to a custom prediction function that uses the trained ML model to calculate the prediction probability of each perturbation. The weights of the perturbed instances are calculated depending on the proximity to the original instance. LIME outputs the weights of each feature which helps to get a view of which features caused the decision given by the ML model. SHAP is based on game theory, and all features act as players in the game. SHAP calculates the average marginal contribution of a feature regarding all possible coalitions. Other than LIME and SHAP, IG is also used to describe deep learning models [61]. IG calculates the gradients of the output to its features. Initially, an instance was selected. The instance is interpolated starting from a baseline model. Then the gradient is calculated to check the changes in the features to the model prediction.

IV. RESULTS AND DISCUSSION

Here we present the obtained results for different ML algorithms we used, to identify hate speech related images. First, we will present the results of unsupervised learning algorithms, then shallow supervised learning algorithms and finally the results of deep learning algorithms.

A. *Results of unsupervised learning algorithms*

Table 1 presents the results for the autoencoders using the dataset mentioned in Section III A. Autoencoder with convolutional layers have shown better results than other autoencoders.

Table 1. Results for autoencoders

Model	Accuracy	Precision	Recall	F1 Score
GoogLeNet	0.657	0.6599	0.657	0.6555
Convolutional	0.727	0.7304	0.727	0.726

B. *Results of supervised learning algorithms*

Here, first we present the results of shallow ML algorithms, classifying the images using the text in the images. Results of shallow and deep ML algorithms were obtained using two main methods: using popular performance metrics and using explainable AI methods.

1) *Results of shallow ML algorithms using performance metrics:*

Using popular performance metrics for NB and LR algorithms, accuracy of the results was low, for one hot encoding feature extraction method, compared to other methods such as BoW and TF-IDF (see Table 2).

Table 2. Accuracy of one-hot encoding for NB and LR

	Classification Method	
	NB	LR
Accuracy	0.685	0.69

Initially, computations were performed on a dataset comprising 1,000 violent and 1,000 nonviolent images. Subsequently, the dataset size was expanded, and augmentation techniques were applied to further increase its size. Table 3 provides the results of different performance metrics for TF-IDF embedding for shallow algorithms; MNB, LR, SVM and RF before expanding the dataset in Unidecode format. Table 4 presents results after expanding the dataset but without augmentation and Unidecode format. Table 5 depicts the results of different performance metrics for the augmented and in Unidecode format. Table 6 describes the results of the BoW embedding with Unidecode and augmented data, and Table 7 describes the results for Unidecode and increased dataset but before the augmentation. According to these tables, NB classification has obtained higher results than other classification algorithms, RF showed lower results, and TF-IDF and BoW have obtained comparable results. The conversion of text to Unidecode format and expansion of the dataset has led to a noticeable improvement in the results. 91% accuracy was obtained for TF-IDF, BoW with NB classifier. The results of the BoW were slightly higher than TF-IDF.

Table 3. TF-IDF results before increasing the dataset but with Unidecode data

Metrics	Classification Methods			
	MNB	LR	SVM	RF
Accuracy	0.8725	0.875	0.875	0.6575
roc_auc_score	*0.9578*	*0.9477*	*0.9464*	0.7433

F1	0.8771	0.8775	0.878	0.5387
Precision	0.8505	0.8647	0.8612	0.8333
Recall	*0.9055*	0.8905	0.8955	0.398

Table 4. TF-IDF results after increasing the dataset but without Unidecode and augmented data

Metrics	MNB	LR	SVC	RF
Accuracy	0.829	0.8261	0.8217	0.7878
Precision	0.7851	0.8318	0.8125	0.7956
F1	0.8332	0.8159	0.8155	0.7735
Recall	0.8876	0.8006	0.8186	0.7526
ROC_AOC_Score	0.9238	0.9055	0.9074	0.8696

Table 5. TF-IDF results for the augmented and Unidecode dataset

Metrics	MNB	LR	SVC	RF
Accuracy	0.9074	0.8953	0.8961	0.813
Precision	0.8784	0.899	0.8925	0.8414
F1	0.9075	0.8908	0.8925	0.7962
Recall	0.9385	0.8827	0.8925	0.7556
ROC_AOC_Score	0.9721	0.9628	0.9601	0.8967

Further we have used NB as the classification algorithm with different sampling techniques and obtained the performance metrics (see Table 8). MNB with the BoW method has obtained better results for all the metrics in cross-validation (k-fold and stratified). However, employing cross-validation did not improve the previous result. Results depict that k-fold and stratified sampling have higher results than resampling.

2) *Results of shallow ML algorithms using explainable methods:*

We have used two text examples to describe the results obtained for XAI methods in shallow learning. Preprocessed and Unidecode text of Sentences 1 and 2 are shown in Table 9. Figure 6 and Figure 7 depict the LIME and SHAP outputs of sentence 1.

Table 6. BoW results for the augmented and Unidecode dataset

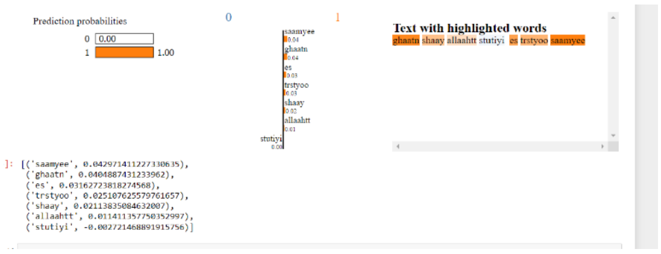
Metrics	Classification Methods			
	MNB	LR	SVM	RF
Accuracy	***0.9163***	0.8875	0.8534	0.815
roc_auc_score	***0.9735***	***0.9536***	***0.9323***	0.8927
F1	***0.915***	0.881	0.8462	0.7991
Precision	0.8914	***0.9106***	0.8634	0.8515
Recall	***0.9399***	0.8534	0.8296	0.7528

Table 7. BoW results for the Unidecode dataset but without the augmentation

Metrics	Classification Methods			
	MNB	LR	SVM	RF
Accuracy	***0.9047***	0.8732	0.8532	0.8381
F1	***0.9026***	0.8655	0.8453	0.8236
Precision	0.8825	0.8931	0.8639	0.8805
Recall	***0.9235***	0.8396	0.8276	0.7736

Table 8. Results of MNB classification for sampling methods with Unidecode and augmented data

Embedding	Sampling	Accuracy	F1	Precision	Recall
BoW	k-fold	***0.91***	***0.911***	0.884	***0.940***
TF-IDF	k-fold	***0.91***	0.893	0.867	***0.921***
BoW	Stratified	***0.91***	***0.903***	0.85	***0.956***
TF-IDF	Stratified	***0.91***	***0.913***	0.889	***0.939***
BoW	Resampling	0.862	0.867	0.846	0.889
TF-IDF	Resampling	0.862	0.865	0.860	0.871



```
]: [('saamyee', 0.042971411227330635),
    ('ghaatn', 0.0404887431233962),
    ('es', 0.03162723818274568),
    ('trstyoo', 0.025107625579761657),
    ('shaay', 0.02113835084632007),
    ('allaahtt', 0.011411357750352997),
    ('stutiyi', -0.0027214688891915756)]
```

Figure 6. LIME results for sentence 1 - Shallow learning

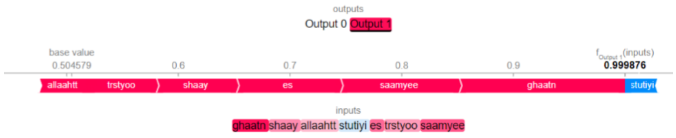


Figure **7.** SHAP results for sentence 1 - Shallow learning.

LIME output with BoW as the embedding and NB as the classification algorithm has found Sinhala words ඝාතන (killings), සහාය (supported), අල්ලාහ්ට (Allah), ත්‍රස්තයෝ (terrorists), and සාමයේ (peace) caused to conclude that sentence 1 as violent. Sentence 2 can be identified as nonviolent. Violent words are highlighted (orange) in the text. Although Sinhala words සහාය (supported) and සාමයේ (peace) are nonviolent words, they were identified as violent. Orange colour indicates violent words, and others are nonviolent words.

3) *Results of deep learning algorithms using performance metrics:*

Table 10 presents the Sinhala text classification results using deep learning algorithms. The results were analysed with and without data augmentation. The presented outcomes are the best possible outputs obtained under different conditions: learning rate and number of epochs. Superior results were obtained for 1D CNN with a learning rate of 0.002. The learning rate was chosen as 0.000001 for other algorithms like LSTM and GRU. The GRU with CNN ensemble models (250 epochs) converge to a solution within fewer numbers of epochs than the CNN models (450 epochs).

XLM-R was evaluated for 500, 700 and 1,000 epochs with a learning rate of 0.000001, obtaining 93% accuracy, which is better than that of other models. XLM-R achieved over 90% for precision, recall, and F1-score, also outperforming other models. Subsequently, the GRU and CNN ensemble model, incorporating word2vec achieved 91% accuracy. Similarly, CNN with BiGRU utilizing word2vec, 1D CNN with word2vec and 1D CNN with fastText achieved 90% accuracy. In the context of 1D CNN, fastText with 300 embedding dimensions showed better results than word2vec embedding. Figure 8 illustrates the confusion matrix of XLM-R.

Figures 9 and 10 depict the loss and accuracy curves of nine deep learning models, respectively. CNN with GRU and CNN with BiGRU that incorporate word2vec, exhibit lower loss than LSTM and CNN models. Furthermore, CNN with GRU, incorporating word2vec, exhibit higher accuracy compared to other models.

Table 9. Sinhala text examples

Sentence No.	Text	English Translation	Preprocessed Text	Unidecode Text
Sentence 1	මේ ඝාතන වලට සහාය දුන් අල්ලාහ්ට ස්තුතියි ! අයි එස් ත්‍රස්තයෝ කියයි. සාමයේ ආගම මෙය ද?	Thanks to Allah, who supported these killings! IS terrorists say. Is this the religion of peace?	ඝාතන (killing) සහාය (support) අල්ලාහ්ට (Allah) ස්තුතියි (thanks) එස් (IS) ත්‍රස්තයෝ (terrorists) සාමයේ (peace)	ghaatn shaay allaahtt stutiyi es trstyoo saamyeet
Sentence 2	ඔබේ ගෙල වටා පැළඳිය හැකි හොඳම ආභරණය වන්නේ ඔබේ දරුවන්ගේ දෑතයි	The best jewelry you can wear around your neck is your children's arms	ගෙල (neck) වටා (around) පැළඳිය (wear) හොඳම (best) ආභරණය (jewelry) දරුවන්ගේ (children's) දෑතයි (arms)	gel vttaa paellndiy hondm aabhrnny druvngee daaetyi
Sentence 3	බෞද්ධයින්ට තඩි නොබා දෙපිල බෙදී මරාගන්නා අන්තවාදී මුස්ලිම් කල්ලි මඩිනු. ත්‍රස්තවාදයට නිදහසේ වැඩෙන්නට ඉඩදී බලා සිටින්නේ මේ රට තවත් ඉරාකයක් වෙනතුරුද? බෞද්ධ අන්තවාදයක් ගැන බොරු බේගල් ඇද නොබා මුස්ලිම් අන්තවාදය ගැන ඇත්ත පිළිගන්න.	Instead of punishing Buddhists, stop extremist Muslim gangs who divide and kill. Are they allowing terrorism to grow freely and waiting for this country to become another Iraq? Accept the truth about Muslim extremism without pulling false stories about Buddhist extremism.	බෞද්ධයින්ට (Buddhists) තඩි (punish) නොබා (not) දෙපිල (two groups) බෙදී (divide) මරාගන්නා (killing) අන්තවාදී (extremist) මුස්ලිම් (muslim) කල්ලි (gang) මඩිනු (stop). ත්‍රස්තවාදයට (terrorism) නිදහසේ (freely) වැඩෙන්නට (grow) ඉඩදී (let) සිටින්නේ (waiting) ඉරාකයක් (Iraq) වෙනතුරුද (until) බොරු (false) බේගල් (stories) ඇද නොබා (without telling) මුස්ලිම් (muslim) අන්තවාදය(extremism) ඇත්ත (truth) පිළිගන්න (accept)	bauddhyintt tddi nobaa depil bedii mraagnnaa antvaadii muslim klli mddinu. trstvaadytt nidhsee vaeddenntt idddii blaa sittinnee mee rtt tvt iraakyk venturud? bauddh antvaadyk gaen boru beegl aed nobaa muslim antvaady gaen aett pillignn.

Table 10. Results of deep learning algorithms in text classification

Method	**Augmentation**[1]	**Accuracy**	**Precision**	**F1**	**Recall**
CNN+word2vec 300 [2]	No	***0.9033***	0.8791	***0.9022***	***0.9265***
CNN+BiGRU+word2vec 300	No	0.8788	0.8855	0.8776	0.8763
CNN+BiGRU+word2vec 300	Yes	***0.9041***	***0.9055***	***0.9038***	***0.9032***
CNN+GRU+word2vec 300	No	0.8925	0.8926	0.8923	0.8921
BiGRU+word2vec 300	No	0.8911	0.8909	0.8911	0.8914
BiLSTM+word2vec 300	No	0.8853	0.8852	0.8851	0.885
CNN+BiLSTM+word2vec 300	No	0.8889	0.8892	0.8889	0.8897
GRU+word2vec 300	No	0.8939	0.8941	0.8937	0.8935
LSTM+word2vec 300	No	0.8853	0.8874	0.8847	0.8839
CNN+LSTM+word2vec 300	No	0.8918	0.8918	0.8916	0.8914
CNN+word2vec 300	Yes	***0.9001***	0.8586	***0.9019***	***0.9497***
CNN+GRU+word2vec 300	Yes	***0.9136***	***0.9137***	***0.9134***	***0.9132***
BiGRU+word2vec 300	Yes	0.8987	0.8986	0.8987	0.8989
GRU+word2vec 300	Yes	0.898	0.8988	0.8978	0.8973
CNN+fastText 300	Yes	***0.9048***	***0.9044***	***0.9012***	0.898
BiGRU+fastText 300	Yes	0.8872	0.8871	0.8871	0.8872
CNN+GRU+fastText 300	Yes	0.8899	0.8898	0.8898	0.8899
GRU+fastText 300	No	0.8687	0.8691	0.8683	0.868
CNN+fastText 300	No	0.8788	0.8833	0.8725	0.8621
BiGRU+fastText 300	No	0.8687	0.8685	0.8685	0.8684
CNN+GRU+fastText 300	No	0.8874	0.8873	0.8873	0.8872
CNN+fastText 450	Yes	0.898	0.9088	0.8927	0.8771
CNN+GRU+fastText 450 [3]	Yes	0.896	0.8965	0.8958	0.8955
XLM-R	Yes	***0.9203***	***0.9118***	***0.9182***	***0.9245***
XLM-R	No	***0.93***	***0.9371***	***0.9265***	***0.916***

[1] Augmentation: Yes - Refers to the dataset containing augmented data. No - Refers to the dataset without any augmentation.
[2] CNN stands for 1D CNN. 300 represents the embedding dimension. The "+" sign signifies the fusion of the "CNN" algorithm and the "word2vec" embedding mechanism with 300 embedding dimensions.
[3] 450 represents the embedding dimension.

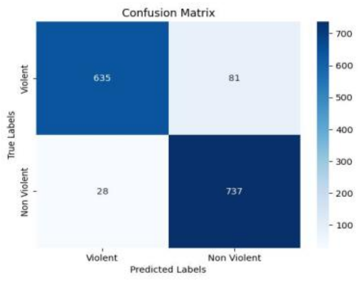


Figure 8. Confusion matrix of XLM-R

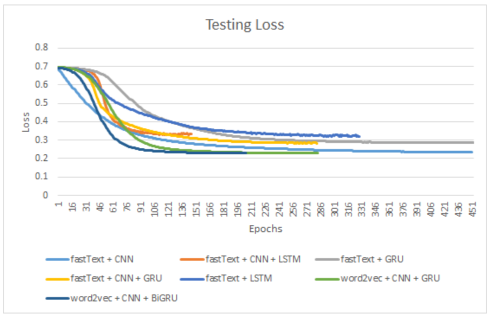


Figure 9. The loss of the model incurred on the test data.

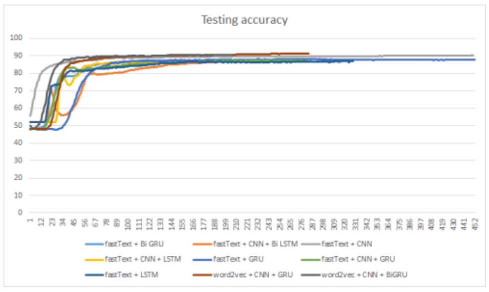


Figure 10. The accuracy achieved on the test data.

Table 11 presents the performance results of 1D CNN, CNN, and GRU ensemble model, as well as the XLM-R models after partitioning the dataset into train, validation, and test subsets. The results depict that the XLM-R model achieved superior results compared to the 1D CNN and GRU ensemble model.

Table 11. Performance results on different dataset splits

	Accuracy	Precision	Recall	F1-score
fastText + 1D CNN	0.8862	0.9024	0.8626	0.8821
word2vec + CNN + GRU	0.8977	0.8992	0.8973	0.8975
XLM-R	0.93	0.94	0.91	0.92

3) *Results of deep learning algorithms using explainable methods:*

Sentences 1, 2, and 3, as depicted in Table 10, are utilized in the context of deep learning. Sentences 1 and 3 are identified as violent, whereas Sentence 2 is identified as nonviolent. Violent words identified by LIME using CNN, and GRU ensemble are හම්බයො (similar word for Muslims), අත්අඩංගුවට (arrested), මරනයට (to death), නපුන්සකයෝ (eu nuchs), බොර (fake), උද්ඝෝෂණයක් (campaign), පගාව (revenge), විනාශ (destruction), කුඩු ජාවාරමේ (drug dealing), මුස්ලිම් (Muslim), ත්‍රස්ත (terror), දූෂනය (corruption), ඉස්ලාමයේ (Islam), අන්තවාදය (extremism), බේබදු (drunkenness), බැනල (scolded), හොරු (thieves), අවජාතක (bastards), වංචාව (fraud), පිළිකුලෙන් (disgusted), බලු (dogs), නීතිය (law), මුසල්මානුවන්ට (similar word for Muslims), හලාල් (Halal), කුඩුකාරයෝ (drug addicted), අන්තවාදීන් (extremists), හොරකං (thieves), and මාට්ටු (caught). The identified nonviolent words are, ආදරෙන් (with love), ඉවසීම (patience), හිතේ (heart), බැඳීමකින් (bond), සිනහවට (smile), කඳුලු (tears), පැළඳිය (dress), හොඳම (best), දරුවන්ගේ (children's), ගෙල (neck), සතුටින් (happy), දෙමාපියන් (parents), and මානසික (mental). Although some nonviolent words were identified as violent in Sentence 1 and Sentence 2 by shallow machine learning, in deep learning they were identified correctly. Figure 11 and Figure 12 show the LIME output of Sentence 1 and Sentence 2 respectively.

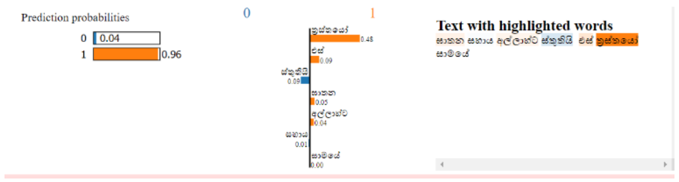


Figure 11. LIME results for sentence 1 using deep learning.

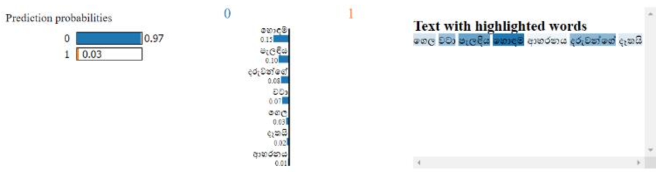


Figure 12. LIME results for sentence 2 using deep learning.

SHAP has produced slightly different results than LIME. Figure 13 shows sentence 1. The red colour indicates violent words. According to the figure, සාමයේ (peace) is identified as a violent word.

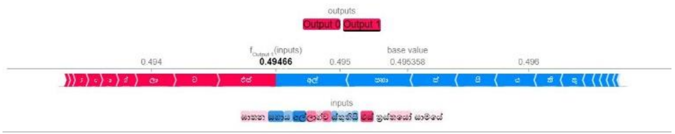
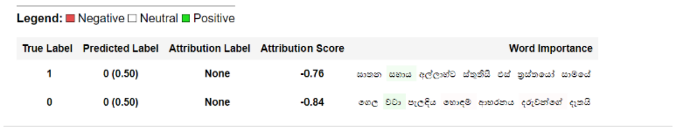


Figure 13. SHAP output of sentence 1

IG and LIME have produced different outputs. Red and green colors indicate violent and nonviolent words, respectively. Figure 14 shows sentences 1 and 2 for IG. According to thoutput, sentence 1 is identified as nonviolent (the predicted label is 0). Violent words are not highlighted in the text either.


Legend: 🟥 Negative □ Neutral 🟩 Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
1	0 (0.50)	None	-0.76	සාතන සහාය අල්ලාහ්ට ස්තුති එස් ත්‍රස්තයෝ සාමයේ
0	0 (0.50)	None	-0.84	ගෙල වට පැළදිය හොඳම ආභරණය දරුවන්ගේ දෑතයි

Figure 14. IG results for sentences 1 and 2

V. CONCLUSION

In the context of classifying images posted based on hate speech, unsupervised learning algorithms achieved 73% accuracy. Increased dataset size, along with characters encoded using Unidecode, has resulted in a 92% accuracy for shallow machine learning algorithms. Comparable results were obtained for BoW and TF-IDF, with slightly higher results for BoW. Models that employ GRU have achieved 91% accuracy. Models with 1D CNN achieved 90% accuracy, and the XLM-R algorithm obtained 93% accuracy. RNN architectures that employ LSTM have shown lower results than models that incorporate GRU and 1D CNN. However, LSTM with the CNN model obtained 89% accuracy.

In most cases, data augmentation has improved the results of models employing the GRU architecture. Among the models that employ GRU, slightly better results have been obtained for word2vec than for fastText. LIME has shown better interpretation than SHAP and IG. Supervised learning of text classification produced better results than unsupervised learning for identifying violent Sinhala image posts. This research can be further enhanced by extracting the text from image posts automatically using a text extraction method.

REFERENCES

[1] Aggarwal C.C., Sathe S.: Theoretical foundations and algorithms for outlier ensembles. In: Acm sigkdd explorations newsletter, vol. 17(1), pp. 24--47, 2015.

[2] Aktı S., Ofli F., Imran M., Ekenel H.K.: Fight detection from still images in the wild. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp. 550--559. 2022.

[3] Amorim M., Bortoloti F.D., Ciarelli P.M., Salles E.O., Cavalieri D.C.: Novelty detection in social media by fusing text and image into a single structure. In: IEEE Access, vol. 7, pp. 132786--132802, 2019.

[4] Amos B., Ludwiczuk B., Satyanarayanan M., et al.: Openface: A general-purpose face recognition library with mobile applications. In: CMU School of Computer Science, vol. 6(2), p. 20, 2016.

[5] Badjatiya P., Gupta S., Gupta M., Varma V.: Deep learning for hate speech detection in tweets. In: Proceedings of the 26th international conference on World Wide Web companion, pp. 759--760. 2017.

[6] Bharti: Sentimental with Multi Layer Bi directional RNN using PyTorch, 2020. URL {https://medium.com/@bhartikukreja2015/ sentimental-with-multi-layer-bi-directional-rnn-using-pytorch-4f386297a0fc}.

[7] Blandfort P., Patton D.U., Frey W.R., Karaman S., Bhargava S., Lee F.T., Varia S., Kedzie C., Gaskell M.B., Schifanella R., et al.: Multimodal social media analysis for gang violence prevention. In: Proceedings of the International AAAI conference on web and social media, vol. 13, pp. 114--124. 2019.

[8] Bojanowski P., Grave E., Joulin A., Mikolov T.: Enriching word vectors with subword information. In: Transactions of the association for computational linguistics, vol. 5, pp. 135--146, 2017.

[9] Breiman L.: Random forests. In: Machine learning, vol. 45, pp. 5--32, 2001.

[10] Chhabra A., Vishwakarma D.K.: A literature survey on multimodal and multilingual automatic hate speech identification. In: Multimedia Systems, pp. 1--28, 2023.

[11] Chung J., Gulcehre C., Cho K., Bengio Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. In: arXiv preprint arXiv:1412.3555, 2014.

[12] Cohen J.: A coefficient of agreement for nominal scales. In: Educational and psycho logical measurement, vol. 20(1), pp. 37--46, 1960.

[13] Conneau A., Lample G.: Cross-lingual language model pretraining. In: Advances in neural information processing systems, vol. 32, 2019.

[14] Cramer J.S.: The origins of logistic regression. In: , 2002.

[15] Davidson T., Warmsley D., Macy M., Weber I.: Automated hate speech detection and the problem of offensive language. In: Proceedings of the international AAAI conference on web and social media, vol. 11, pp. 512--515. 2017.

[16] Demotte P., Senevirathne L., Karunanayake B., Munasinghe U., Ranathunga S.: SEN CAT Tool for Sinhala Sentiment Analysis, 2020. URL https://sencat.lk/.

[17] Devlin J., Chang M.W., Lee K., Toutanova K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: arXiv preprint arXiv:1810.04805, 2018.

[18] Dhananjaya V., Demotte P., Ranathunga S., Jayasena S.: BERTifying Sinhala--A Comprehensive Analysis of Pre-trained Language Models for Sinhala Text Classification. In: arXiv preprint arXiv:2208.07864, 2022.

[19] Dias D.S., Welikala M.D., Dias N.G.: Identifying racist social media comments in sinhala language using text analytics models with machine learning. In: 2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer), pp. 1--6. IEEE, 2018.

[20] Dikwatta U., Fernando T.: Violence Detection of Sinhala Image Posts with Autoencoders. In: 2021 10th International Conference on Information and Automation for Sustainability (ICIAfS), pp. 275--280. IEEE, 2021.

[21] Djuric N., Zhou J., Morris R., Grbovic M., Radosavljevic V., Bhamidipati N.: Hate speech detection with comment embeddings. In: Proceedings of the 24th international conference on world wide web, pp. 29--30. 2015.

[22] Edunov S., Ott M., Auli M., Grangier D.: Understanding back-translation at scale. In: arXiv preprint arXiv:1808.09381, 2018.

[23] Fortuna P., Nunes S.: A survey on automatic detection of hate speech in text. In: ACM Computing Surveys (CSUR), vol. 51(4), pp. 1--30, 2018.

[24] Graves A., Schmidhuber J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. In: Neural networks, vol. 18(5-6), pp. 602--610, 2005.

[25] Haque R., Islam N., Tasneem M., Das A.K.: MULTI-CLASS SENTIMENT CLASSIFICATION ON BENGALI SOCIAL MEDIA COMMENTS USING MACHINE LEARNING. In: International Journal of Cognitive Computing in Engineering, 2023.

[26] He K., Gkioxari G., Dollár P., Girshick R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 2961--2969. 2017.

[27] Hochreiter S., Schmidhuber J.: Long short-term memory. In: Neural computation, vol. 9(8), pp. 1735--1780, 1997.

[28] Hossin M., Sulaiman M.N.: A review on evaluation metrics for data classification evaluations. In: International journal of data mining & knowledge management process, vol. 5(2), p. 1, 2015.

[29] Jallad K.A., Ghneim N.: ArNLI: Arabic Natural Language Inference for Entailment and Contradiction Detection. In: arXiv preprint arXiv:2209.13953, 2022.

[30] Kaplan A.M., Haenlein M.: Users of the world, unite! The challenges and opportunities of Social Media. In: Business horizons, vol. 53(1), pp. 59--68, 2010.

[31] Le Q., Mikolov T.: Distributed representations of sentences and documents. In: International conference on machine learning, pp. 1188--1196. PMLR, 2014.

[32] Le Cam L.: Maximum likelihood: an introduction. In: International Statistical Review/Revue Internationale de Statistique, pp. 153--171, 1990.

[33] Lundberg S.M., Lee S.I.: A unified approach to interpreting model predictions. In: Advances in neural information processing systems, vol. 30, 2017.

[34] Malik J.S., Pang G., Hengel A.v.d.: Deep learning for hate speech detection: a comparative study. In: arXiv preprint arXiv:2202.09517, 2022.

[35] Malmasi S., Tetreault J., Dras M.: Oracle and human baselines for native language identification. In: Proceedings of the tenth workshop on innovative use of NLP for building educational applications, pp. 172--178. 2015.

[36] Malmasi S., Zampieri M.: Detecting hate speech in social media. In: arXiv preprint arXiv:1712.06427, 2017.

[37] Mazari A.C., Boudoukhani N., Djeffal A.: BERT-based ensemble learning for multi aspect hate speech detection. In: Cluster Computing, pp. 1--15, 2023.

[38] Meyer D., Wien F.: Support vector machines. In: The Interface to libsvm in package e1071, vol. 28, p. 20, 2015.

[39] Mikolov T., Chen K., Corrado G., Dean J.: Efficient estimation of word representations in vector space. In: arXiv preprint arXiv:1301.3781, 2013.

[40] Mikolov T., Chen K., Corrado G., Dean J.: Efficient estimation of word representations in vector space. In: arXiv preprint arXiv:1301.3781, 2013.

[41] Mozafari M., Farahbakhsh R., Crespi N.: A BERT-based transfer learning approach for hate speech detection in online social media. In: Complex Networks and Their Applications VIII: Volume 1 Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019 8, pp. 928--940. Springer, 2020.

[42] Naher J., Minar M.R.: Impact of social media posts in real life violence: A case study in Bangladesh. In: arXiv preprint arXiv:1812.08660, 2018.

[43] Omar A., Mahmoud T.M., Abd-El-Hafeez T.: Comparative performance of machine learning and deep learning algorithms for Arabic hate speech detection in osns. In: Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2020), pp. 247--257. Springer, 2020.

[44] Omar A., Mahmoud T.M., Abd-El-Hafeez T., Mahfouz A.: Multi-label arabic text classification in online social networks. In: Information Systems, vol. 100, p. 101785, 2021.

[45] Orru P., et al.: Racist discourse on social networks: A discourse analysis of Facebook posts in Italy. In: Rhesis, vol. 5(1), pp. 113--133, 2015.

[46] Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., et al.: Scikit-learn: Machine learning in Python. In: the Journal of machine Learning research, vol. 12, pp. 2825--2830, 2011.

[47] Prechelt L.: Early stopping-but when? In: Neural Networks: Tricks of the trade, pp. 55--69. Springer, 2002.

[48] Ranathunga S., Liyanage I.U.: Sentiment analysis of sinhala news comments. In: Transactions on Asian and Low-Resource Language Information Processing, vol. 20(4), pp. 1--23, 2021.

[49] Rathnayake H., Sumanapala J., Rukshani R., Ranathunga S.: Adapter-based fine-tuning of pre-trained multilingual language models for code-mixed and code-switched text classification. In: Knowledge and Information Systems, vol. 64(7), pp. 1937--1966, 2022.

[50] Ribeiro M.T., Singh S., Guestrin C.: ``Why should i trust you?" Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135--1144. 2016.

[51] Ruder S.: An overview of gradient descent optimization algorithms. In: arXiv preprint arXiv:1609.04747, 2016.

[52] Ruwandika N., Weerasinghe A.: Identification of hate speech in social media. In: 2018 18th international conference on advances in ICT for emerging regions (ICTer), pp. 273- -278. IEEE, 2018.

[53] Samarasinghe S., Meegama R., Punchimudiyanse M.: Machine learning approach for the detection of hate speech in sinhala unicode text. In: 2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer), pp. 65--70. IEEE, 2020.

[54] Sandaruwan S.T., Lorensuhewa S.A.S., Munasinghe K.: Identification of abusive sinhala comments in social media using text mining and machine learning techniques. In: The International Journal on Advances in ICT for Emerging Regions, vol. 13(1), 2020.

[55] Schmidt A., Wiegand M.: A survey on hate speech detection using natural language processing. In: Proceedings of the fifth international workshop on natural language processing for social media, pp. 1--10. 2017.

[56] Senarath Y.: A language processing tool for Sinhalese, 2004. URL https://sinling. ysenarath.com/.

[57] Senevirathne L., Demotte P., Karunanayake B., Munasinghe U., Ranathunga S.: Sentiment analysis for sinhala language using deep learning techniques. In: arXiv preprint arXiv:2011.07280, 2020.

[58] Shorten C., Khoshgoftaar T.M.: A survey on image data augmentation for deep learning. In: Journal of big data, vol. 6(1), pp. 1--48, 2019.

[59] Silva E., Nandathilaka M., Dalugoda S., Amarasinghe T., Ahangama S., Weerasuriya G.T.: Machine Learning-Based Automated Tool to Detect Sinhala Hate Speech in Images. In: 2021 6th International Conference on Information Technology Research (IC ITR), pp. 1--7. IEEE, 2021.

[60] Sun S., Liu Y., Mao L.: Multi-view learning for visual violence recognition with maximum entropy discrimination and deep features. In: Information Fusion, vol. 50, pp. 43--53, 2019.

[61] Sundararajan M., Taly A., Yan Q.: Axiomatic attribution for deep networks. In: International conference on machine learning, pp. 3319--3328. PMLR, 2017.

[62] Sunde B.M.: Early Stopping for PyTorch, 2020. URL https://github.com/ Bjarten/early-stopping-pytorch.

[63] Suryawanshi S., Chakravarthi B.R., Arcan M., Buitelaar P.: Multimodal meme dataset (MultiOFF) for identifying offensive content in image and text. In: Proceedings of the second workshop on trolling, aggression and cyberbullying, pp. 32--41. 2020.

[64] Tran C.: A Complete Guide to CNN for Sentence Classification with PyTorch, 2020. URL https://chriskhanhtran.github.io/posts/ cnn-sentence-classification/.

[65] Uppada S.K., Patel P.: An image and text-based multimodal model for detecting fake news in OSN?s. In: Journal of Intelligent Information Systems, pp. 1--27, 2022.

[66] Vajjala S., Majumder B., Gupta A., Surana H.: Practical natural language processing: a comprehensive guide to building real-world NLP systems. O'Reilly Media, 2020.

[67] Waseem Z., Hovy D.: Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In: Proceedings of the NAACL student research workshop, pp. 88--93. 2016.

[68] Waseem Z., Thorne J., Bingel J.: Bridging the gaps: Multi task learning for domain transfer of hate speech detection. In: Online harassment, pp. 29--55, 2018.

[69] Watanabe H., Bouazizi M., Ohtsuki T.: Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. In: IEEE access, vol. 6, pp. 13825--13835, 2018.

[70] Webb G.I.: Decision tree grafting from the all-tests-but-one partition. In: Ijcai, vol. 2, pp. 702--707. 1999.

[71] Wolf T., Debut L., Sanh V., Chaumond J., Delangue C., Moi A., Cistac P., Rault T., Louf R., Funtowicz M., Davison J., Shleifer S., von Platen P., Ma C., Jernite Y., Plu J., Xu C., Scao T.L., Gugger S., Drame M., Lhoest Q., Rush A.M.: HuggingFace's Transformers: State-of-the-art Natural Language Processing, 2019. URL http://dx.doi.org/10. 48550/ARXIV.1910.03771.

[72] Won D., Steinert-Threlkeld Z.C., Joo J.: Protest activity detection and perceived violence estimation from social media images. In: Proceedings of the 25th ACM international conference on Multimedia, pp. 786--794. 2017.

[73] Zhang Z., Luo L.: Hate speech detection: A solved problem? the challenging case of long tail on twitter. In: Semantic Web, vol. 10(5), pp. 925--945, 2019.

[74] Šolc T.: Unidecode, lossy ASCII transliterations of Unicode text, 2022. URL https: //github.com/avian2/unidecode.

ACKNOWLEDGMENT

The authors wish to thank Apple Research and Development Center, Department of Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Nugegoda, Sri Lanka, for providing the facilities needed to conduct the research.

AVAILABILITY OF SUPPORTING DATA

Data are available at the public repository https://github.com/umandaDik/Data.

An Approach to Examine and Recognize Anomalies on Cloud Computing Platforms with Machine Learning Concepts

MPGK Jayaweera[1], WMCJT Kithulwatta[2,3#], and RMKT Rathnayaka[3,4]
[1]Department of Computing and Information Systems, Faculty of Computing, Sabaragamuwa University of Sri Lanka, Belihuloya, Sri Lanka
[2]Department of Information and Communication Technology, Faculty of Technological Studies, Uva Wellassa University of Sri Lanka, Badulla, Sri Lanka
[3]Research Center for Grey Systems and Uncertainty Analysis (GSUSL), Department of Physical Sciences & Technology, Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka, Belihuloya, Sri Lanka.
[4]Department of Physical Sciences and Technology, Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka, Belihuloya, Sri Lanka

#chirantha@uwu.ac.lk

ABSTRACT Cloud computing is one of the most rapidly growing computing concepts in today's information technology world. It connects data and applications from various geographical locations. A large number of transactions and the hidden infrastructure in cloud computing systems have presented the research community with several challenges. Among these, maintaining cloud network security has emerged as a major challenge. It is critical to address issues in the quickly changing cloud computing market in order to guarantee that businesses can fully utilize cutting-edge technology, uphold strong security protocols, and maximize operational effectiveness. Businesses that successfully navigate these obstacles can maintain their competitiveness in a dynamic digital ecosystem by improving scalability, leveraging the flexibility provided by the cloud, and adapting to technological changes with ease. Anomaly detection (or outlier detection) is the identification of unusual or suspicious data that differs significantly from the majority of the data. Research on anomaly detection in cloud network data is crucial because it enables businesses to more rapidly and efficiently recognize potential security threats, network performance concerns, and other issues. Recently, machine learning methods have demonstrated their efficacy in anomaly detection. This research aimed to introduce a novel hybrid model for anomaly detection in cloud network data and to investigate the performance of this model in comparison to other machine learning algorithms. The research was conducted with the UNSW-NB15 anomaly dataset and employed various feature selection and pre-processing techniques to prepare the data for model training. The hybrid model was built using a combination of Random Forest and SVM algorithms and the process was evaluated using metrics such as F1-Score, Recall, Precision, and Accuracy. The result showed that the hybrid model has 94.23% accuracy and a total time of 109.92s which is the combination of the train time of 100.45s and prediction time of 9.47s. The limitations of the study include the class imbalance problem in the dataset and the lack of real-world applications for testing. The research suggests future work in the application of hybrid models in anomaly detection and cloud network security and the need for further investigation into the potential benefits of such models.

KEYWORDS: Anomaly Detection, Cloud Computing, Machine Learning, Monitoring

I. INTRODUCTION

The technology of cloud computing virtualization provides efficient resources for end users. The characteristics of cloud computing include manageability, scalability, and availability. In addition, cloud computing has the advantages of economy, on-demand service, convenience, universality, multi-tenancy, flexibility, and stability [27]. Cloud computing mainly provides three service delivery models and four development patterns: infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS), public cloud, private cloud, hybrid cloud, community cloud, and virtual private cloud [29]. Today, cloud computing has integrated with other computing technologies like fog computing, grid computing, Docker containers, IoT, etc [28], [30], [31]. Cloud security is one of the most important aspects of cloud computing because it involves thousands of user transactions, information, and communication. The availability, integrity, and confidentiality of cloud computing platforms or services must be ensured to provide secure cloud computing platforms/services. Security vulnerabilities and challenges arise from the usage of cloud computing services. Currently, cloud computing models are the primary source of these challenges and vulnerabilities [32]. The intruders exploit the weakness of cloud models in accessing the users' private data, by attacking the processing power of computer systems [3].

An anomaly is an observation that differs so significantly from previous observations that it raises suspicion that it was caused by a distinct mechanism. It's frequently a sign of something unexpected or problematic happening. Anomaly detection is the identification of rare item events or observations that raise suspicion by differing significantly from the majority of data. They are slightly or majorly different from the majority of data and anomaly detection can help to find outliers and problems in data. In other words, anomalies are data points or patterns in a dataset that differ significantly from the expected or usual behavior. These anomalies can be produced by several things, including measurement errors, sensor malfunction, data corruption, or system failure, and they can happen spontaneously or as a result of mistakes in data collecting or processing. Finding these odd data points or patterns in a dataset that are frequently a sign of a deeper issue or problem is called anomaly detection. A dataset may contain a variety of anomalies, including point anomalies that only affect a single instance of data, contextual anomalies that only occur under certain circumstances, collective anomalies that involve multiple data points that behave similarly, and collective contextual anomalies that involve multiple data points that behave similarly only under certain circumstances. In several fields, including network intrusion detection, fraud detection, defect detection, and monitoring of complex systems, anomaly detection is a critical step [6][7].

Finding strange or unexpected data points or patterns in a dataset is the process of anomaly detection. Anomalies can be found using a variety of techniques, such as statistical techniques, clustering, classification, deep learning, distance-based techniques, and time-series-based techniques. Quantiles, standard deviation, and other statistical metrics are used in statistical procedures to detect data points that significantly depart from the norm. Anomalies are data points that do not belong to any cluster and are grouped by clustering algorithms. To categorize new data points as normal or abnormal, classification algorithms are trained on labeled data. Deep learning algorithms discover the underlying structure of the data and the location of data points that deviate from this pattern to find anomalies. Measures of the distance between data points are used by distance-based algorithms to detect data points that are far away from other ones. To identify anomalies, time-series-based algorithms employ techniques like moving average, exponential smoothing, ARIMA, and Prophet. A combination of several methods is frequently used to boost the robustness and accuracy of anomaly detection. The choice of the method relies on the nature of the data and the particular requirements of the application [2].

The connection between cloud network data and anomaly detection is it provides an analysis of unusual activities, and unexpected activities through anomaly detection algorithms. Effective monitoring and security procedures are becoming more and more important as more businesses shift their data and apps to the cloud. Anomaly detection can aid in the identification of potential security vulnerabilities and performance problems, enabling businesses to take preventative action to lessen these risks and their effects on operations. A wealth of knowledge regarding the functionality, security, and use of cloud-based systems is contained in cloud network data. Log files, performance indicators, network traffic, and other sorts of data are examples of this data. These data can be examined by anomaly detection algorithms to find patterns or anomalies that point to issues with the network or its elements, such as security breaches, performance issues, or other suspicious activities. Additionally, anomaly detection in cloud network data aids organizations in conforming to several legal standards about the security, privacy, and integrity of their data. Automated anomaly detection is a crucial tool for preserving the security and dependability of cloud-based systems since it gets more challenging to manually detect and react to anomalies as more data is stored and processed in the cloud [1][6].

II. MOTIVATION

The goal of anomaly detection is to use approaches that can discover relevant anomalies in data without producing a large number of false positives.

Cloud security is one of the most important aspects of cloud computing because it involves thousands of user transactions and information. The availability, integrity, and confidentiality of cloud computing platforms or services must be ensured to provide secure cloud computing platforms/services. Security vulnerabilities and challenges arise from the usage of cloud computing services. Currently, cloud computing models are the primary source of these challenges and vulnerabilities. The intruders exploit the weakness of cloud models in accessing the users' private data, by attacking the processing power of computer systems [8][10].

The detection of anomalies in data has a long history and a wide range of applications. An anomaly or outlier is an observation that differs so significantly from other observations that it raises the possibility that it was generated by a different mechanism. It can also be defined as an outlier observation that shows up to deviate significantly from the rest of the sample members in which it occurs.

Due to the complexity of modern systems, highly available cloud service requirements in a cloud environment are difficult to guarantee and can thus only be ensured with great effort. As a result of these trends, there is an increasing demand for intelligent applications that automatically detect anomalies and provide suggestions for solving or at least mitigating problems so that a negative impact on service quality does not cascade. What constitutes an anomaly in each case is determined by the sample and the methodology. Anomalies are classified into three types in general: Anomalies can be classified into three types namely *point anomalies*, *collective anomalies*, and *contextual anomalies*. There are primarily three approaches for detecting anomalies (machine learning, deep learning, and statistical approach). After reviewing previous studies, the study discovered that machine learning outperforms the other two methods in detecting abnormalities. Although the practice mentioned above provides ways to detect anomalies in a dataset. The research community still knows little about which is the most suitable algorithm for detecting anomalies within a cloud environment. The author is motivated to close this gap of knowledge and try to use a specific machine learning algorithm to detect anomalies using a data set. After analyzing the team can decide whether this algorithm is suitable or not for detecting anomalies within a cloud network [4][5][10][24].

A. Significance of the study

It is critical to address anomaly problems in cloud computing platforms because they have an immediate effect on the security and dependability of digital infrastructure. Anomalies can jeopardize data integrity and result in breaches and unauthorized access, regardless of whether they are caused by malevolent activity or system malfunctions. It is imperative to promptly identify and address irregularities in order to assurance the unceasing procedure of cloud-based services, protect confidential data, and uphold user confidence. In the quickly changing world of digital technology, proactive tactics for anomaly management not only improve the general resilience of cloud systems but also help to build a strong cybersecurity foundation.

B. Research objectives

To keep a clear direction within the research study, below research objectives (RO) were made.

RO1: *To introduce a novel hybrid model and compare the performance of the hybrid model to other machine learning models, such as single-algorithm models, in detecting anomalies in cloud network data.*

RO2: *To look into how different algorithmic combinations affect, how well the hybrid model performs while looking for anomalies in data from cloud networks.*

RO3: *To investigate how well the novel hybrid model handles various data kinds and investigate how various feature selection and pre-processing techniques affect the novel hybrid model's ability to detect anomalies in cloud network data.*

C. Contribution of the paper

By presenting a novel hybrid model that combines Random Forest (RF) and Support Vector Machine (SVM) techniques, the research significantly advances the subject of anomaly detection in cloud network data. This hybrid method offers a unique solution for anomaly detection problems, marking a significant deviation from the traditional application of single-algorithm models. In contrast to stand-alone RF models, the hybrid model aims to improve detection robustness and accuracy by combining the advantages of both RF and SVM.

One of the primary contributions is the extensive testing of the proposed hybrid model against multiple machine learning methods, including multiple RF and SVM configurations and an MLP model.

III. RESEARCH METHOD

Machine learning models such as Isolation Forests, One-Class SVM, and Autoencoders are frequently employed in anomaly identification. These models are significant because, in the absence of labeled training data, they are highly effective at identifying patterns and abnormalities in a variety of datasets. One-Class SVM is skilled at identifying outliers in high-dimensional spaces, Autoencoders learn intrinsic data representations, and Isolation Forests effectively isolate anomalies by building random decision trees. These tools are useful for detecting deviations from normal patterns in a variety of applications, including cybersecurity and system monitoring. This approach involves building up a hybrid model combining SVM and random forest algorithms. This research used the UNSW-NB15 dataset for the study. The methodology is concluded here after identifying and analyzing the comparisons between different algorithm models.

The combined strengths of Random Forest (RF) and Support Vector Machine (SVM) in handling different areas of anomaly detection in cloud network data led to their selection for the hybrid model. As an ensemble learning technique, RF is well-known for its stability and resistance to overfitting. It is particularly good at capturing complicated relationships within data. However, SVM is good at managing non-linear patterns by determining optimal decision boundaries, especially when employing non-linear kernels. The hybrid model combines the power of SVM's ability to identify distinct decision boundaries with the versatility of RF's modeling techniques to attempt to capitalize on the differences between the two approaches.

A. Gather Relevant Data

The UNSW NB15 dataset was used in this research study to study the usage of cloud network data to detect anomalies. The loading of the UNSW NB15 dataset was the first stage in the study procedure. The dataset included network traffic information that can be used to develop and test anomaly detection methods.

B. Pre-processing and Feature Selection

Preprocessing the dataset came after the data had been loaded. Make sure the data is prepared for usage in the feature selection process, this may involve cleaning and normalizing it. The process of choosing a subset of the features in a dataset that is most important for anomaly detection is known as feature selection. Techniques like correlation analysis or mutual information can be used for this.

C. Train the Model

The process of training models using the chosen features followed the feature selection phase. The dataset was divided into training and testing sets, and several anomaly detection models were trained and evaluated using these sets. In this study, models like Random Forest (Estimators = 100), Random Forest (Estimators = 50), Random Forest (Estimators = 150), SVM (Kernel - rbf, gamma-scale), SVM (Kernel - sigmoid, gamma-scale), SVM (Kernel - poly, gamma-scale), and a hybrid model that combined the best features of Random Forest and SVM models were used.

D. Analyze the Model

A comparison was done once the models had been trained and assessed to see which model performed the best on the UNSW NB15 dataset. The evaluation measures used in the comparison included accuracy, precision, recall, and F1-score. The comparison's findings were used to evaluate the performance of various models for finding anomalies in cloud network data.

E. Summary of the Methodology

In conclusion, the study used the UNSW NB15 dataset to evaluate the hybrid model through preprocessing, feature selection, model training using random forest models, SVM models, and a hybrid model, and comparing all the models to determine which is the best.

This research study recommended the following methodology step-wise to better understand:

- ***Data collection:*** The UNSW-NB15 anomaly dataset was used.
- ***Data preprocessing and feature selection:*** The data was preprocessed and features were selected for the training and testing sets.
- ***Model training:*** The model was trained using Random Forest and SVM algorithms [34].
- ***Hybrid model construction:*** A novel hybrid model was built due to their higher accuracy and other aspects.
- ***Model evaluation:*** The performance of the novel hybrid model was evaluated and compared to that of other machine learning models, such as single-algorithm models, Random Forest (Estimators = 100), Random Forest (Estimators = 50), Random Forest (Estimators = 150) and SVM (Kernel - rbf, gamma - scale), SVM (Kernel - sigmoid, gamma - scale), SVM (Kernel - poly, gamma - scale), and MLP(ANN) model.
- ***Data analysis:*** The results were analyzed and discussed in terms of the research objectives, including the impact of various algorithmic combinations on the performance of the hybrid model, the performance of the hybrid model compared to that of single-algorithm models, and the potential future research pathways for the application of hybrid models in anomaly detection and cloud network security.
- ***Limitations and recommendations:*** The limitations of the study were identified as the class imbalance problem in the dataset and future research recommendations were made to address the class imbalance problem in the dataset, further investigate the potential of hybrid models in anomaly detection and cloud network security, and investigate the rate of false positives and false negatives, computational resources and the ease of understanding of the hybrid model.

IV. DESIGN, IMPLEMENTATION, AND ANALYSIS OF THE RESULTS

This section describes the model's design comprehensively with the model's basic architecture and the proposed model's workflow. Here several diagrams are presented and discussed to explain model functions. The technologies, algorithms, special methods, and functions used in implementation were defined in this section. Further, this section discussed the findings of the phases of implementation.

A. Gathering the Relevant Data Set

The UNSW-15 dataset was a good option for the study since it offers a thorough assessment of the proposed approach's capacity to recognize various sorts of attacks. The dataset included both known and undiscovered attack types, allowing for the evaluation of the approach's capacity to identify several distinct attacks. Additionally, a thorough evaluation of the performance of the approach is possible due to the dataset's size and abundance of instances. The dataset also included real-world network traffic statistics, enhancing the relevance and applicability of the study's findings to real-world circumstances. Furthermore, the performance of the proposed technique in other current ways can be easily compared thanks to the UNSW-15 dataset, which is a well-known and often-used dataset in the field of network intrusion detection. A fair assessment of the performance of the suggested strategy is possible thanks to the dataset's balance, which includes a sufficient number of both normal and attack occurrences. The dataset is additionally current and contains up-to-date network traffic data, increasing its applicability to current real-world settings.

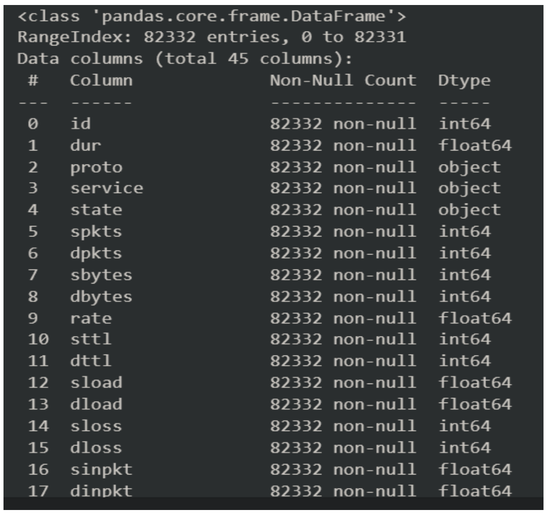
Loading Data

First, the author read a CSV file and created a DataFrame object in Python using the Pandas module. In particular, it loads the data from the CSV file at the supplied file path using the ***read csv***() function and stores it in the variable ***'df'***. The DataFrame is a strong and adaptable data structure that makes it simple to manipulate and analyze data presented in tabular form. The author then used to show the data frame's first five rows. This can be helpful for rapidly verifying that the data has been loaded properly and previewing the contents of the DataFrame. Table 1 presents the whole content for the loaded data in the study.

Table 1: The content of the loaded data

index	id	dur	proto	service	state	spkts	dpkts	sbytes
0	1	1.10E-05	udp	-	INT	2	0	496
1	2	8.00E-06	udp	-	INT	2	0	1762
2	3	5.00E-06	udp	-	INT	2	0	1068
3	4	6.00E-06	udp	-	INT	2	0	900
4	5	1.00E-05	udp	-	INT	2	0	2126

Further, figure 1 demonstrates the metadata of the loaded data comprehensively.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 82332 entries, 0 to 82331
Data columns (total 45 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   id                  82332 non-null  int64
 1   dur                 82332 non-null  float64
 2   proto               82332 non-null  object
 3   service             82332 non-null  object
 4   state               82332 non-null  object
 5   spkts               82332 non-null  int64
 6   dpkts               82332 non-null  int64
 7   sbytes              82332 non-null  int64
 8   dbytes              82332 non-null  int64
 9   rate                82332 non-null  float64
 10  sttl                82332 non-null  int64
 11  dttl                82332 non-null  int64
 12  sload               82332 non-null  float64
 13  dload               82332 non-null  float64
 14  sloss               82332 non-null  int64
 15  dloss               82332 non-null  int64
 16  sinpkt              82332 non-null  float64
 17  dinpkt              82332 non-null  float64
```

Figure 1: Information of the loaded data

Furthermore, table 2 displays a tabular description of the loaded data.

Table 2: Description of the loaded data

	id	dur	spkts	dpkts	sbytes	dbytes
count	82332	82332	82332	82332	82332	82332
mean	41166.5	1.006756	18.66647	17.54594	7993.908	13233.79
std	23767.35	4.710444	133.9164	115.5741	171642.3	151471.5
min	1	0	1	0	24	0
25%	20583.75	8.00E-06	2	0	114	0
50%	41166.5	0.014138	6	2	534	178
75%	61749.25	0.71936	12	10	1280	956
max	82332	59.99999	10646	11018	14355774	14657531

B. Data Pre-Processing and Feature Selection

Pre-processing the data is a crucial stage in the methodology of the study since it guarantees that the UNSW-15 dataset is in a format that the model can use. The UNSW-15 dataset's data pre-processing may entail several important procedures.

Removal of Irrelevant Columns

To remove particular columns from the DataFrame, the author used the ***DataFrame function drop()***. It starts by making a list of the columns that should be deleted, author dropped "id" and "attack cat." The ***drop()*** method was then called with this list as its first argument. When axis=1 is used as the second parameter, pandas is instructed to remove the columns. The third parameter, inplace=True, is set to mean that the original DataFrame should be used for the operation. As a result, this will delete the columns "id" and "attack cat" from the DataFrame "df," update the original DataFrame to reflect the deletion of those columns, and return no new DataFrame.

Clamping

Clamping is a preprocessing method for reducing the range of values in a dataset. It is usually applied to stop outliers from skewing the results of subsequent processes, including statistical analysis or machine learning. Putting a maximum and minimum threshold for the values in a dataset entails "clamping," or setting any values outside of this range to the threshold value closest to them. This can help prepare data for analysis and clean it, which can also help to increase the precision and stability of machine learning models. In this research, the author prunes extreme values to make distributions less skewed. Features are reduced to the 95th percentile when their maximum values exceed 10 times the median value.

In summary, the author produces descriptive statistics for the numeric columns after first filtering the original DataFrame to only include those columns. The outcome is a new DataFrame that gives an overview of the distribution of data in the original DataFrame's numerical columns. Then, the author determines whether the maximum value of any column is bigger than 10 times the median value and greater than 10, and if it is, it replaces the values in that column with the 95th percentile's value if they are higher, else the value is left alone. If the DEBUG setting is set to 1, each column will print some information; otherwise, nothing will be printed.

Apply the log function on skewed-right numerical numbers

The author added one to each value before applying the natural logarithm to the values of each column in the numeric DataFrame df numeric if that column's minimum value is zero and there are more than 50 unique values in that column. This avoids using the undefined log(0). If the DEBUG setting is set to 1, each column will print some information; otherwise, nothing will be printed.

Reduce labels in categorical features

Reducing the cardinality of features to 5 or 6. Take the top 5 occurring labels in the feature as labels and set the remainder to '-' as seldom used labels. In this, the author determines whether each given column has more than six distinct values. If this is true for any given column, the value in that column is replaced with a '-' if it is not one of the most frequent values there; otherwise, it is left alone. If the DEBUG setting is set to 1, each column will print some information; otherwise, nothing will be printed. The scenario for reducing the labels in categorical features is presented in Table 3 below.

Table 3: Reduce labels in categorical features

index	proto	service	state
count	82332	82332	82332
unique	131	13	7
top	tcp	-	FIN
freq	43095	47153	39339

Best Features

Univariate statistical tests to determine which features best predict the target feature. Utilizing Python's scikit-learn module, choose the best features from a DataFrame, and display the outcomes. For feature selection, it first imports the required modules SelectKBest and chi2. The SelectKBest class is then created with the chi2 scoring function and the input k='all', instructing it to select all characteristics. The best features object is fitted to the input data by taking into account the goal variable y and the input data X. The scores and feature names are concatenated to produce a new DataFrame. The new DataFrame's columns now go by the names "feature" and "score." The DataFrame is then sorted based on the feature scores, and a bar chart is generated to show the top 21 features.

Figure 2 presents a bar chart for the top features.

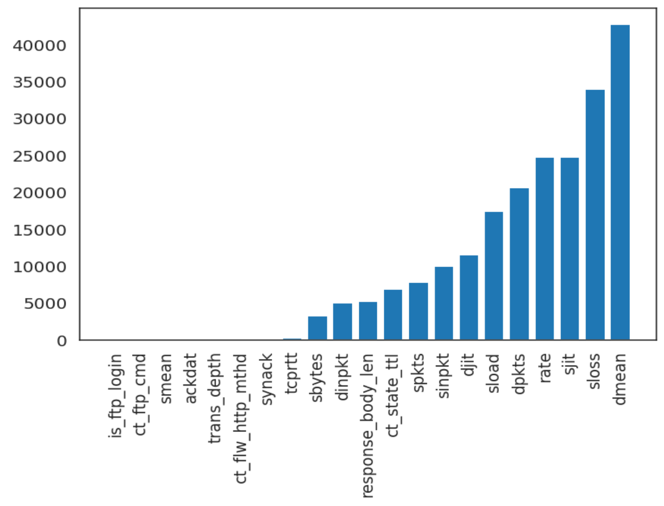


Figure 2: A bar chart for the top features

Encoding Categorical Features

One-hot encoding is used. None of the categorical features are ordinal. In this study, the author tried picking particular rows and columns from the original DataFrame "df" to create two new variables, "X" and "y," and it is displaying the first five rows of the DataFrame "X". The particular encoding categorical features are presented in table 4.

Table 4: Encoding Categorical Features

index	dur	proto	service	state	spkts	dpkts	sbytes
0	1.10E-05	udp	-	INT	0.693147	0	6.206575927
1	8.00E-06	udp	-	INT	0.693147	0	7.474204806
2	5.00E-06	udp	-	INT	0.693147	0	6.97354302
3	6.00E-06	udp	-	INT	0.693147	0	6.802394763
4	1.00E-05	udp	-	INT	0.693147	0	7.661997559

After that, the author used the "OneHotEncoder" class to apply the One-Hot-Encoding approach to columns 1, 2, and 3 of the DataFrame X while leaving the other columns alone to be handled by the "ColumnTransformer" class. Additionally, a ***numpy*** array was being created from the encoded DataFrame.

After that, unique values of a few columns in a DataFrame are extracted using Python's Pandas package, and they are then inserted into a list of feature names in a certain order. These three for loops iterate over the distinct values of the 'state', 'service', and 'proto' columns of the DataFrame 'df' and add them to the list of feature names in reverse order while excluding the first element. To facilitate additional analysis or model training, the author has included the distinctive values from these columns in the list of feature names.

C. Modeling and Evaluation

This entails training the SVM and random forest parts of the hybrid algorithm, training and test split, standardizing continuous features, training with random forest and SVM separately, and implementing a hybrid model and comparison.

Train Set Split

Using stratified sampling, the data in this part are divided into training and test sets. The input data "X" and the target variable "y" are divided into two datasets: the training set and the testing set, using the scikit-learn library. The dataset, the percentage of the dataset that should be given to the testing set, a random seed to assure repeatability, and the stratification of the data are all inputs to the "train test split" function, which was employed by the author in this study. The 'X train', 'X test', 'Y train', and 'Y test' datasets will be utilized for the models' training and testing, respectively. This split is an essential stage in the machine learning process because it enables the author to predict how well the model will perform on new data and avoid overfitting.

Standardize continuous features

The continuous features are scaled using a standard scaler to ensure that they are all in the same size order. To normalize the numerical features of the training and test datasets, use the scikit-learn library. It makes use of the "StandardScaler" class from the library's "preprocessing" module to scale the numerical features to unit variance and standardize them by removing the mean. By utilizing the 'fit transform()' method, which first fits the scaler to the data before transforming it, it generates an instance of the 'StandardScaler' class and applies it to the numerical characteristics of the training dataset. The test dataset's numerical features are then normalized using the transform() method using the same instance of the scaler. The efficiency and stability of the models can be enhanced by normalizing the numerical features, which is a crucial step because many Machine Learning methods are sensitive to the scale of the data.

Following that, the author constructs an empty dataframe called "model performance" and imports much time-related, performance-related metrics from the scikit-learn library. The dataframe comprises seven columns, including "Accuracy," "Recall," "Precision," "F1-Score," "train time," "pred time," and "total time." The time-related functions from the Python library were used to assess the time spent on training and prediction of the model, and the imported performance metrics from the scikit-learn library were used later to evaluate the performance of a machine learning model. This dataframe was used to store these measurements for later examination.

Random Forest

The author is making predictions on the test dataset while training a Random Forest classifier in Python using the scikit-learn library. It generates an instance of the class, imports the RandomForestClassifier class from the library's ensemble module, and then trains the model using the training dataset. Using a time module also keeps track of the length of time spent on training and prediction. After making the predictions, the author made predictions on the test dataset using the trained model's prediction approach. The Random Forest Classifier is an ensemble method that uses averaging to increase predicted accuracy and reduce over-fitting. It trains numerous decision trees on different subsamples of the dataset. The summary data for the Random Forest 100 prediction is presented in Figure 3.

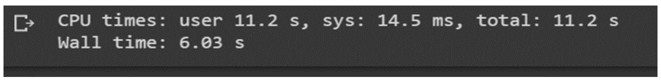

```
CPU times: user 11.2 s, sys: 14.5 ms, total: 11.2 s
Wall time: 6.03 s
```

Figure 3: Prediction (Random Forest 100)

On a test dataset, this algorithm assessed how well a Random Forest classifier model performs. It computes several performance metrics, including accuracy, recall, precision, and f1-score, using the scikit-learn module. It also determines how long training and prediction will take. Additionally, it prints the times and performance indicators in a more readable manner. The results are then saved in a dataframe for future study. Figure 4 presents the model performance data (Random Forest 100).

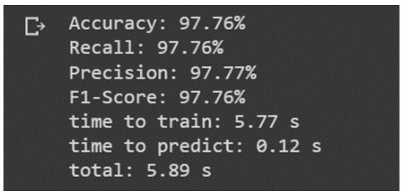

```
Accuracy: 97.76%
Recall: 97.76%
Precision: 97.77%
F1-Score: 97.76%
time to train: 5.77 s
time to predict: 0.12 s
total: 5.89 s
```

Figure 4: Model Performance (Random forest 100)

On a test dataset, the author plotted a confusion matrix for a Random Forest classifier model. A table called the confusion matrix is used to describe how well a classification algorithm performs. The matrix is generated using the scikit-learn library's plot confusion matrix function, which takes the model, test data, and true labels as inputs. The plot is displayed with a white background and a chosen size of 5 x 5. Figure 5 presents the Random Forest 100 Confusion Matrix.

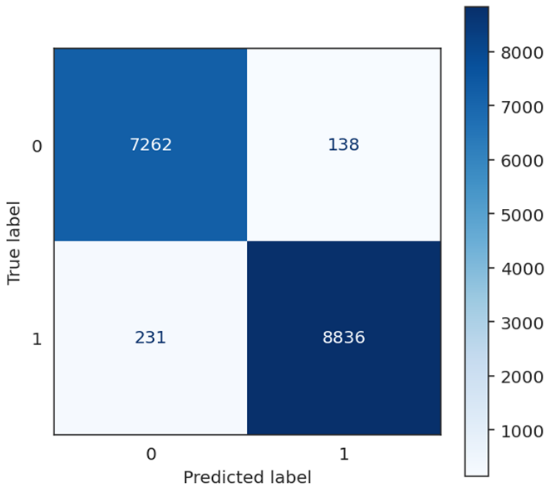


Figure 5: Confusion Matrix (Random Forest 100)

The top 20 features of the Random Forest classifier model are then plotted according to their importance using the scikit-learn package, and the plot was displayed in a 10 x 10-inch format with a white background. Additionally, it removed the top and right spines from the plot and flipped the y-axis such that the most significant feature was at the top.

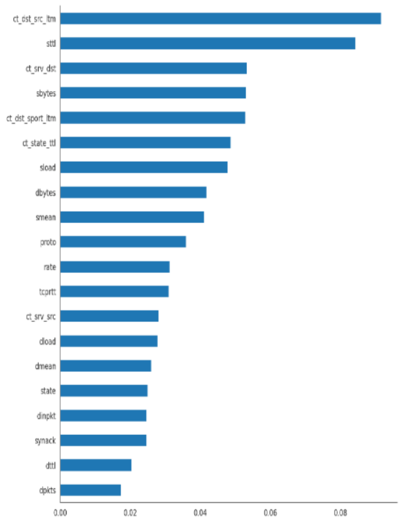
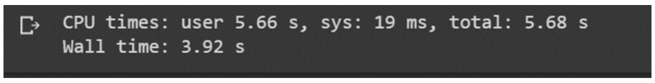


Figure 6 presents the top 20 features of Random Forest for 100 estimators.

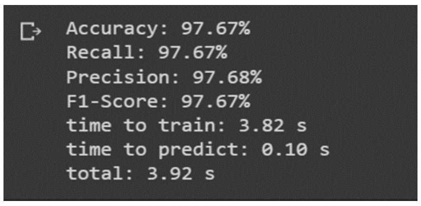
The author used the above concepts for Random Forest (Estimators = 50) and Random Forest (Estimators = 150) and got the following results. Figure 7 presents the prediction for Random Forest 50.



```
CPU times: user 5.66 s, sys: 19 ms, total: 5.68 s
Wall time: 3.92 s
```

Figure 7: Predictions (Random Forest 50)

Further, figure 8 presents the performance of the model for Random Forest 50.



```
Accuracy: 97.67%
Recall: 97.67%
Precision: 97.68%
F1-Score: 97.67%
time to train: 3.82 s
time to predict: 0.10 s
total: 3.92 s
```

Figure 8: Model Perfromance (Random Forest 50)

Figure 9 presents the Confusion Matrix for the Random Forest 50.

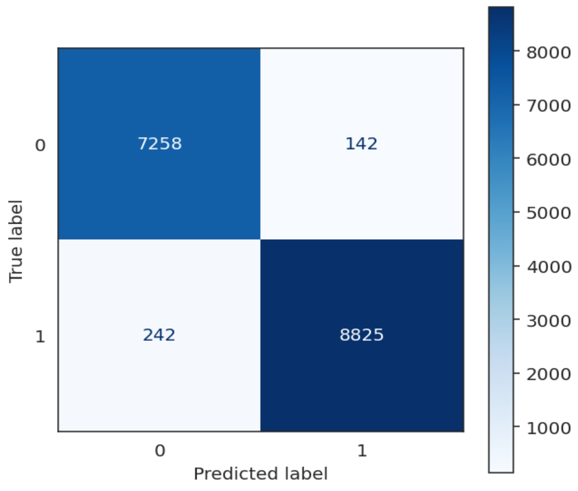
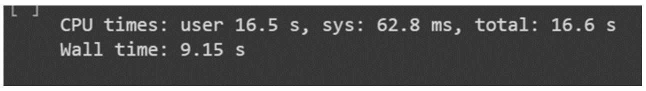


Figure 9: Confusion Matrix(Random Forest 50)

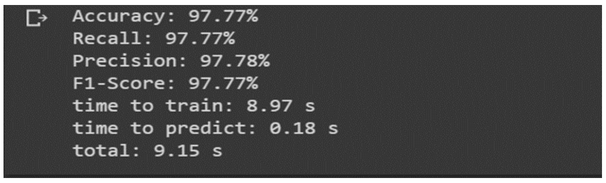
As presented in Figure 10, the Model Prediction for Radom Forest 150 is available.



```
CPU times: user 16.5 s, sys: 62.8 ms, total: 16.6 s
Wall time: 9.15 s
```

Figure 10: Predictions (Random Forest 150)

Further, figure 11 presents the model performance for Random Forest 150.



```
Accuracy: 97.77%
Recall: 97.77%
Precision: 97.78%
F1-Score: 97.77%
time to train: 8.97 s
time to predict: 0.18 s
total: 9.15 s
```

Figure 11: Model Performance (Random Forest 150)

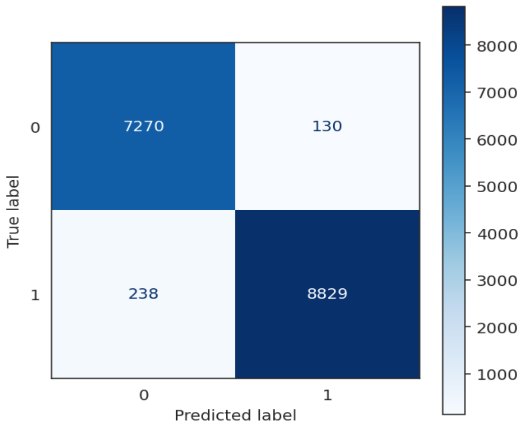


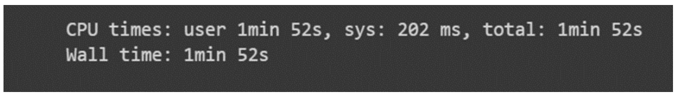
Figure 12: Confusion Matrix (Random Forest 150)

According to Figure 12, the Confusion Matrix for Random Forest 150 was presented.

SVM

After calculating results using the Random Forest algorithm, the author tried to apply these logics to the SVM algorithm.

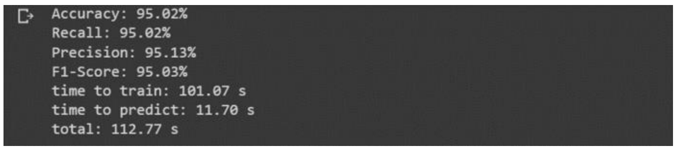
The scikit-learn library was used by the author to train and test a Support Vector Machine (SVM) classification model. The kernel = 'rbf' and gamma ='scale', which were parameters of the RBF kernel, were used to fit the model to the training data. This generated an instance of the SVM class. On the test dataset, it used the trained model to generate predictions. It also kept track of how long training and prediction take. The cell's execution time was also gauged. The gamma parameter was automatically scaled by ***1 / (n_features * X.var()),*** where n _features were the total number of features and ***X.var()*** was the variance of the training dataset, using the 'rbf' kernel and gamma ='scale' in the code. Figure 13 presents the SVM predictions for (Kernal – rbf, gamma – scale).



```
CPU times: user 1min 52s, sys: 202 ms, total: 1min 52s
Wall time: 1min 52s
```

Figure 13: SVM (Kernel - rbf, gamma - scale) Predictions

Then, using accuracy, recall, precision, F1-score, time for training, time for prediction, and total time, the author assessed the SVM model's performance on the test dataset and recorded the findings in a dataframe for later use. The model's performance is then recorded in a dataframe for future use, and the assessment metrics and time measurements are written out in a human-readable format. The SVM model performance for (Kernal – rbf, gamma – scale) was presented in Figure 14.



```
Accuracy: 95.02%
Recall: 95.02%
Precision: 95.13%
F1-Score: 95.03%
time to train: 101.07 s
time to predict: 11.70 s
total: 112.77 s
```

Figure 14: SVM (Kernel - rbf, gamma - scale) Model Performance

The predictions provided by the SVM model on the test dataset are then plotted as a confusion matrix by the author. The model, "X test," and "y test" were used as input arguments for the "plot confusion matrix" function, which creates the confusion matrix. The figurine has a 5.5-inch height and a blue color scheme. The ***'plt.show()'*** function was used to show the plot. It was used to visually assess the model's performance and determine which class the model successfully predicted and which class it incorrectly forecasted. Figure 15 presents the Confusion Matrix for SVM (Kernel - rbf, gamma - scale).

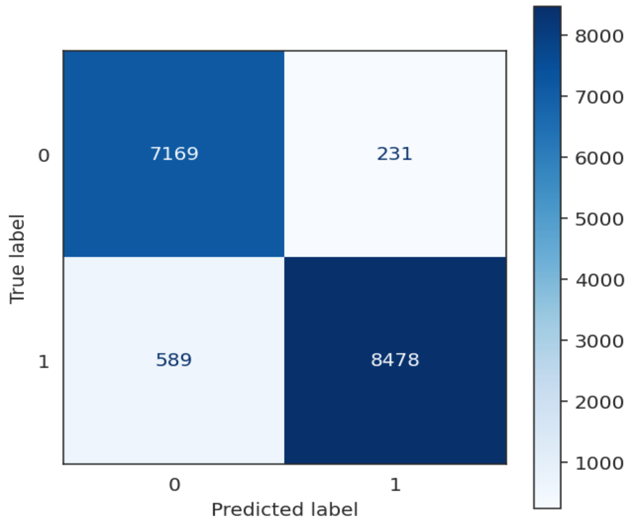
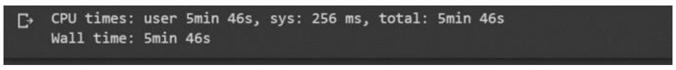


Figure 15: Confusion Matrix for SVM (Kernel - rbf, gamma - scale)

After that, the author used the above concepts to SVM (Kernel - sigmoid, gamma – scale) and SVM (Kernel - poly, gamma - scale). The following results were obtained from the study.

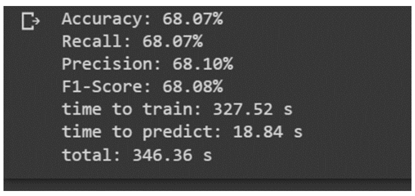
The SVM prediction for (Kernel - sigmoid, gamma - -scale) was presented in Figure 16.



```
CPU times: user 5min 46s, sys: 256 ms, total: 5min 46s
Wall time: 5min 46s
```

Figure 16: SVM (Kernel - sigmoid, gamma - scale) Predictions

The SVM method performance for (Kernel - sigmoid, gamma - -scale) was presented in Figure 17.



```
Accuracy: 68.07%
Recall: 68.07%
Precision: 68.10%
F1-Score: 68.08%
time to train: 327.52 s
time to predict: 18.84 s
total: 346.36 s
```

Figure 17: SVM method performance for (Kernel - sigmoid, gamma - scale)

Further, the Confusion Matrix for SVM (Kernel- - sigmoid, gamma - scale) was presented in Figure 18.

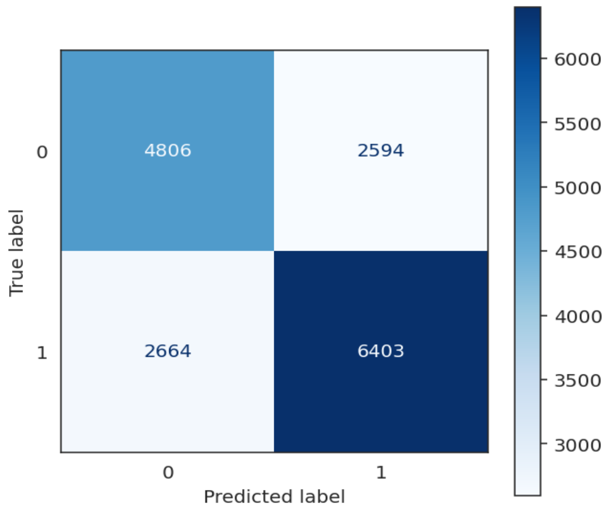
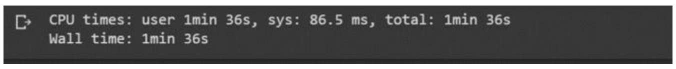


Figure 18: Confusion Matrix for SVM (Kernel - sigmoid, gamma - scale)

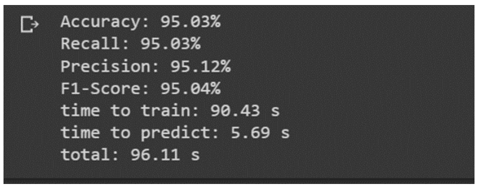
The SVM prediction for (Kernel - poly, gamma - -scale) was presented in Figure 19.



```
CPU times: user 1min 36s, sys: 86.5 ms, total: 1min 36s
Wall time: 1min 36s
```

Figure 19: SVM (Kernel - poly, gamma - scale) Predictions

The SVM method performance for (Kernel - poly, gamma - -scale) was presented in Figure 20.



```
Accuracy: 95.03%
Recall: 95.03%
Precision: 95.12%
F1-Score: 95.04%
time to train: 90.43 s
time to predict: 5.69 s
total: 96.11 s
```

Figure 20: SVM method performance for (Kernel - poly, gamma - scale)

Further, the Confusion Matrix for SVM (Kernel - poly, gamma - -scale) was presented in Figure 21.

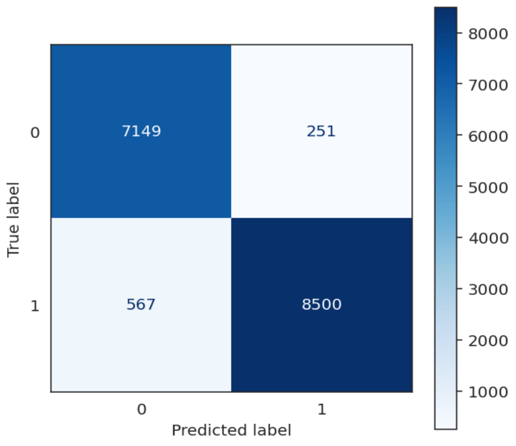
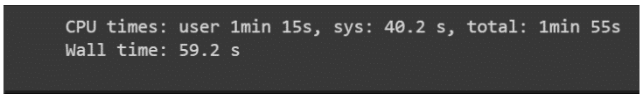


Figure 21: Confusion Matrix for SVM (Kernel - poly, gamma - scale)

ANN - MLP

The Multilayer Perceptron (MLP) is a Feedforward Neural Network (FNN). The MLP is trained using scikit-

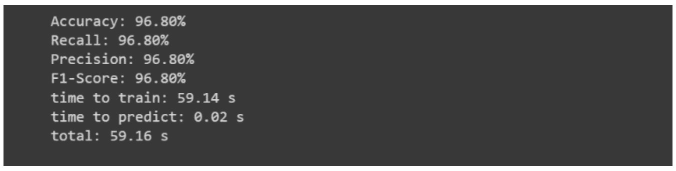
MLPClassifier learn on a dataset ('X train', 'y train') with certain hyperparameters defined, and the learned model is then used to make predictions on another dataset ('X test'). Additionally, it measured how long it takes to train the model and generate predictions using Python's time library. In conclusion, the author tested and trained an MLP classifier. As in Figure 22, the ANN – MLP prediction was captured.



```
CPU times: user 1min 15s, sys: 40.2 s, total: 1min 55s
Wall time: 59.2 s
```

Figure 22: ANN - MLP Predictions

The performance of a trained MLP model was assessed by the following. In this research study, the author used a variety of evaluation metrics, including accuracy, recall, precision, and F1-score. These evaluation metrics were then printed along with the time that it took to train the model, make predictions, and evaluate the performance overall. All evaluation metrics and time were then saved in a dataframe for comparison at a later time. It is a summary of the model's performance. The ANN – MLP method performance is presented in Figure 23.



```
Accuracy: 96.80%
Recall: 96.80%
Precision: 96.80%
F1-Score: 96.80%
time to train: 59.14 s
time to predict: 0.02 s
total: 59.16 s
```

Figure 23: ANN - MLP Method Performance

Next, the author used the scikit-learn library's 'plot confusion matrix' function to create and present a confusion matrix for the trained MLP model on the test dataset. Figure 24 presents the Confusion Matrix for ANN – MLP.

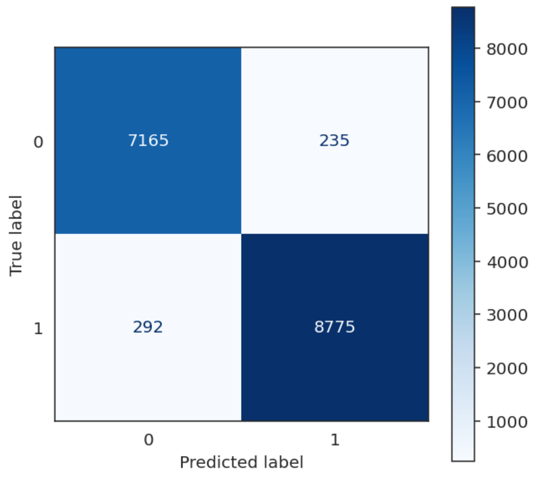


Figure 24: ANN - MLP Confusion Matrix

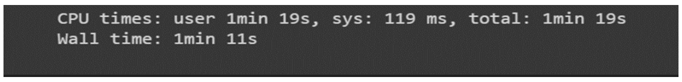
***Although this study used the ANN – MLP model for analyzing purposes. To build the hybrid model, the research team did not use the ANN-MLP model.*

Hybrid Model

The hybrid model was created by combining the Random Forest Model and the SVM model. The Random Forest was used to pre-process the data and to select the most relevant features, followed by the SVM model to classify the data based on the selected features.

- Random Forest Classifier with 150 estimators was used as it yielded the best results in all Random Forest models.
- SVM with poly kernel was used as it yielded the best result among SVM models.

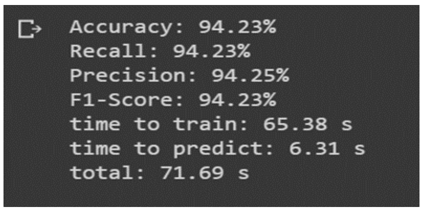
In this research study, a new model that combined the Random Forest Classifier and SVM Classifier was trained. It began by training a Random Forest Classifier with 150 estimators, then used the trained Random Forest model to select the most crucial features from the training data. It set a threshold of "median," which meant that features that were not crucial enough were eliminated from the dataset. The 'X train important' variable was used to keep the training data after it had been modified to include only the most crucial attributes. The test data, which was kept in the 'X test important' variable, went through the same procedure. The important features from the X_train_important data were then used to train an SVM model with a polynomial kernel. Then, using the X_test_important data and the trained SVM model, it made predictions. It also computed the accuracy, recall, precision, and F1-score of the predictions using the y_test data and measured the time required to train the model and make predictions. Figure 25 presents the predictions for the hybrid model.



```
CPU times: user 1min 19s, sys: 119 ms, total: 1min 19s
Wall time: 1min 11s
```

Figure 25: Predictions for Hybrid Model

The author evaluated the performance of a hybrid model that combined the Random Forest model's feature selection method with the Support Vector Machine's classification algorithm (SVM). The most crucial characteristics were chosen from the training set by the Random Forest model, and the SVM was subsequently trained using this smaller feature set. The accuracy, recall, precision, and F1-Score are then used to assess the hybrid model's performance, and the time it took to train and the forecast was also noted. For later comparison with different models, the outcomes were then saved in the "model_performance" dataframe with the label "Hybrid (Estimators - 150, Kernel - poly, gamma - scale)". Figure 26 presents the method performance for the hybrid method.



```
Accuracy: 94.23%
Recall: 94.23%
Precision: 94.25%
F1-Score: 94.23%
time to train: 65.38 s
time to predict: 6.31 s
total: 71.69 s
```

Figure 26: Method Performance - Hybrid Model

Then, using the 'SelectFromModel' feature selection technique, the author generated a confusion matrix for the SVM model that was fitted to the converted training data (X train important) and the transformed test data (X test important). The "Seaborn library's" "plot confusion matrix" method is used to display the confusion matrix as a 5x5-inch figure with a white background and a blue color map to represent it. By comparing the predicted values to the actual values in the test set, this matrix was used to assess the model's performance. Knowing how many false positives, false negatives, true positives, and true negatives the model produced is helpful. Figure 27 presents the Confusion Matrix for the Hybrid Model.

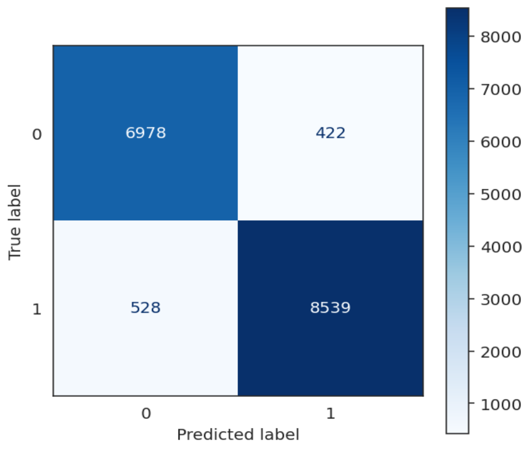


Figure 27: Confusion Matrix for Hybrid Method

Model Comparison

The author can anticipate seeing the performance measures (such as accuracy, recall, precision, and F1-score) of each model as well as their training and prediction timeframes from the model comparison. With the aid of this data, the author can compare the models and choose the one that offers the best overall performance or the best performance/computational efficiency trade-off. The confusion matrix for each model can also be used by the author to gauge how well it predicts the various classes. The overall model comparison is presented in Table 5.

Table 5: Overall model comparison

index	Accuracy	Recall	Precision	F1-Score	train_time	pred_time	total_time
Random Forest (Estimators - 100)	0.97759	0.97759	0.97767	0.97760	7.74432	0.19159	7.9359185
Random Forest (Estimators - 50)	0.97668	0.97668	0.97678	0.97669	4.00490	0.09696	4.1018745
Random Forest (Estimators = 150)	0.97765	0.97765	0.97776	0.97766	11.5436	0.27633	11.819950
SVM (Kernel - rbf, gamma - scale)	0.95020	0.95020	0.95127	0.95029	94.2105	17.3444	111.55504
SVM (Kernel - sigmoid, gamma - scale)	0.68069	0.68069	0.68098	0.68082	357.323	29.7981	387.12117
SVM (Kernel - poly, gamma - scale)	0.95032	0.95032	0.95118	0.95040	101.939	10.0080	111.94741
MLP	0.96799	0.96799	0.96804	0.96800	112.130	0.04786	112.17792
Hybrid (Estimators - 150, Kernel - poly, gamma - scale)	0.942309	0.942309	0.94245	0.94234	100.4496	9.477212	109.9267

Performance Measures:

The hybrid model, which used a combination of 150 estimators and a poly kernel with scale gamma, has an accuracy of 94.2309%. This accuracy is lower than that of the Random Forest algorithm with 100 estimators (97.7592%) and the Random Forest algorithm with 50 and 150 estimators (97.6681% and 97.7652%, respectively), but higher than that of the SVM algorithm with sigmoid kernel and scale gamma (68.0695%).

It could be argued that the specific combination of estimators and kernel used in the hybrid model may not be optimal and that a different combination may yield better performance. Figure 28 presents a comparative bar chart for the Model Performance under the accuracy.

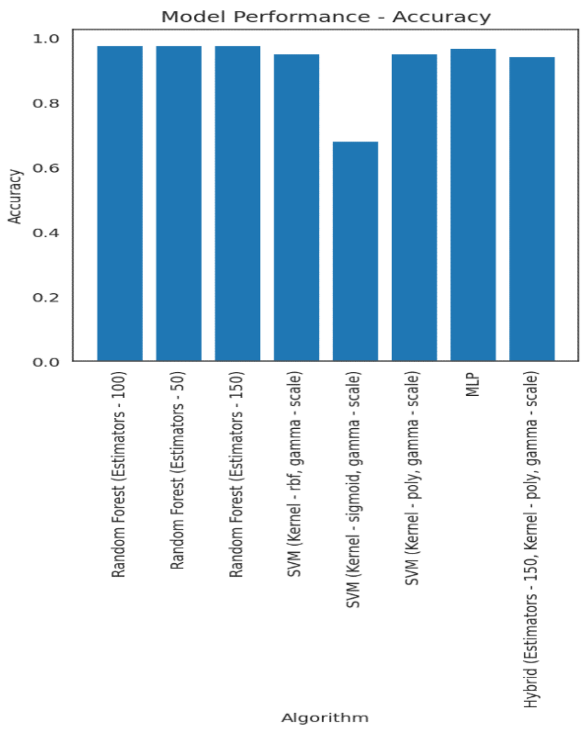


Figure 28: Model Performance - Accuracy

Figure 29 presents the model performance comparison according to the recall.

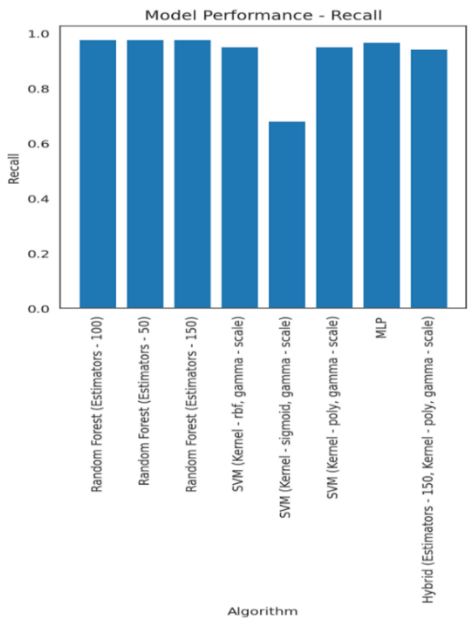


Figure 29: Model Performance - Recall

The hybrid model, which used a combination of 150 estimators and a poly kernel with scale gamma, has a recall value of 94.2309%. This recall is lower than that of the Random Forest algorithm with 100 estimators (97.7592%) and the Random Forest algorithm with 50 and 150 estimators (97.6681% and 97.7652%, respectively). This suggests that the hybrid model is not as good at detecting positive instances (i.e., it has a higher number of false negatives) compared to the Random Forest algorithm with 100 estimators and the Random Forest algorithm with 50 and 150 estimators.

Figure 30 presents the model performance comparison according to the precision.

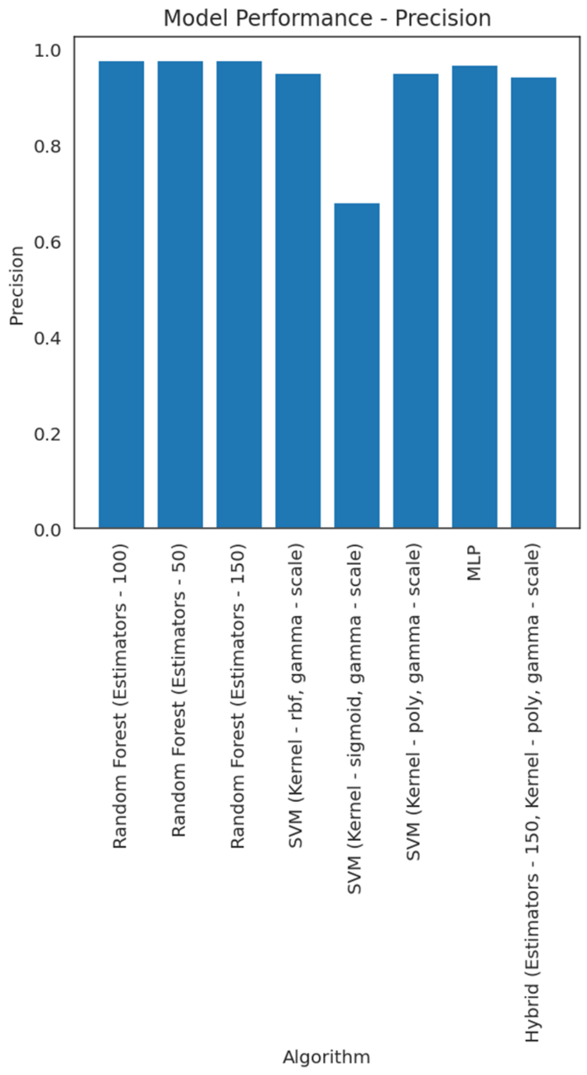


Figure 30: Model Performance - Precision

The hybrid model, which used a combination of 150 estimators and a poly kernel with scale gamma, had a precision value of 94.2459%. This precision was lower than that of the Random Forest algorithm with 100 estimators (97.7678%) and the Random Forest algorithm with 50 and 150 estimators (97.6780% and 97.7765%, respectively). This suggested that the hybrid model was not as good at detecting correct positive instances (i.e., it has a higher number of false positives) compared to the Random Forest algorithm with 100 estimators and the Random Forest algorithm with 50 and 150 estimators.

Figure 31 presents the model performance comparison according to the F1 Score.

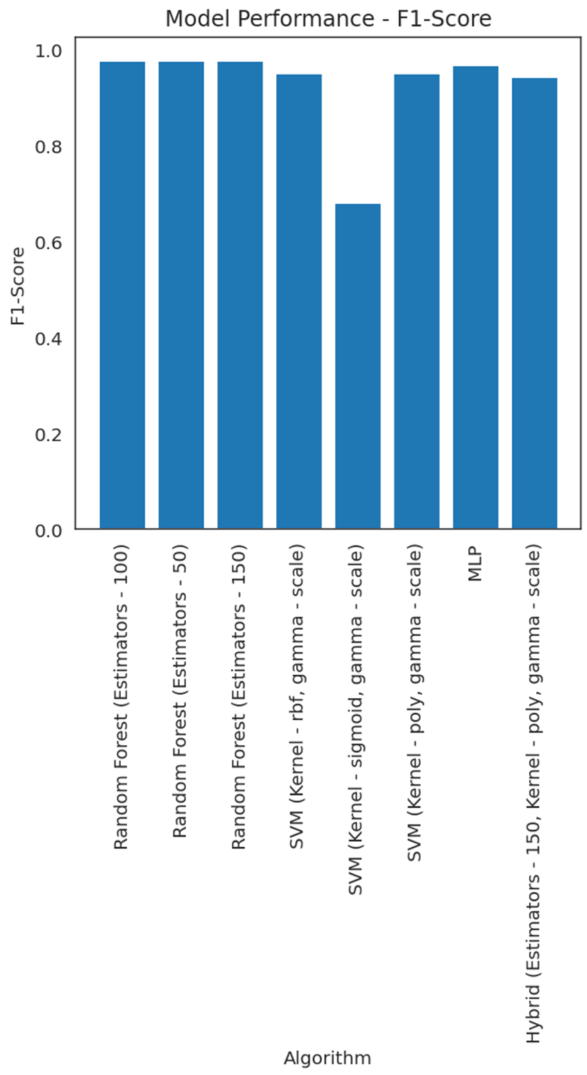


Figure 31: Model Performance - F1 Score

The hybrid model, which used a combination of 150 estimators and a poly kernel with scale gamma, has an F1-score of 94.2344%. This is slightly lower than the Random Forest algorithm with 100 estimators, but higher than the SVM algorithm with sigmoid kernel and scale gamma. This suggested that the hybrid model had a good balance of precision and recall, but not as good as the Random Forest algorithm with 100 estimators. It's also important to note that the F1-score was a measure that seeks a balance between precision and recall, so a higher F1-score means a better balance of precision and recall. In this case, it can be observed that the Hybrid model is not the best in terms of F1-score but it's still quite good.

Time Frames:

The Hybrid model, which used a combination of 150 estimators and a poly kernel with scale gamma, has a train time of 65.38 seconds. This train time was slower than the Random Forest algorithm with 100 estimators, but faster than the SVM algorithm with sigmoid kernel and scale gamma. This suggested that the Hybrid model had a relatively moderate train time compared to other models. However, it's important to consider the trade-off between train time and model performance. As we can see the Hybrid model had a good performance in terms of F1-score, the additional train time may be worth it if the performance gain was deemed significant for the specific application or domain.

Figure 32 presents the model comparison according to the train_time.

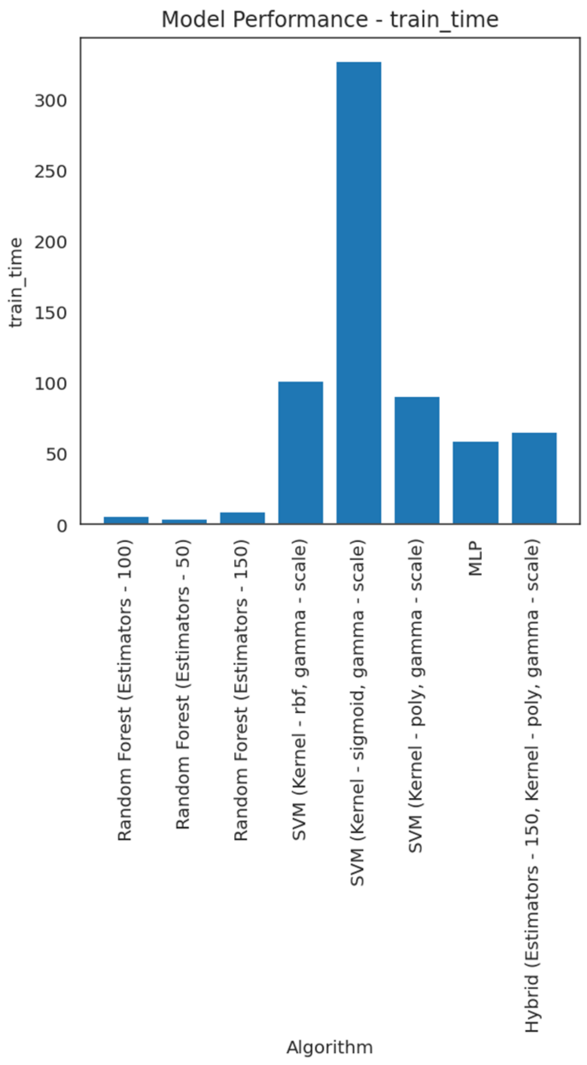


Figure 32: Model Comparison - train_time

Figure 33 presents the model performance comparison according to the pred_time.

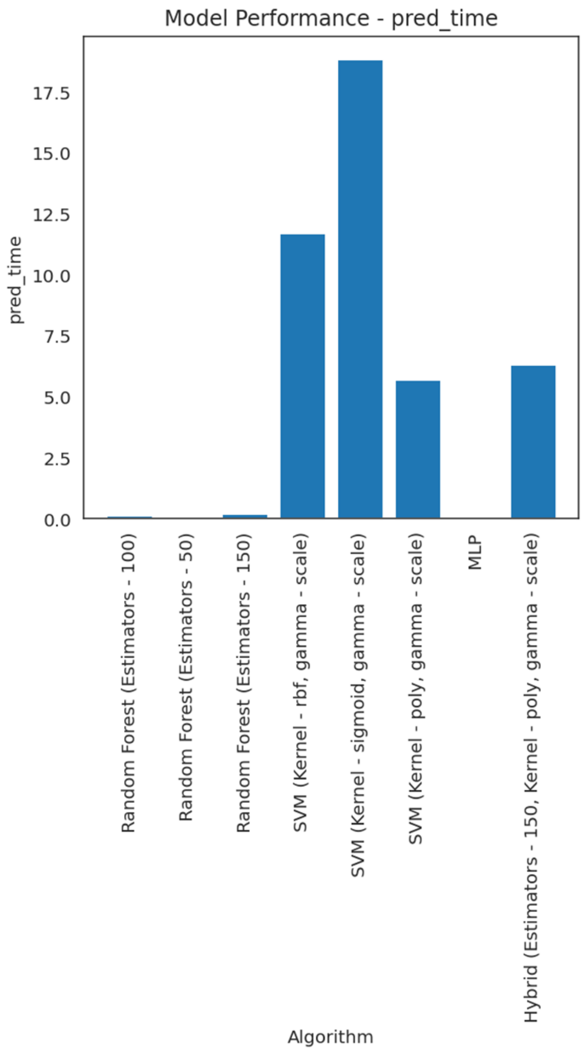


Figure 33: Model Comparison -pred_time

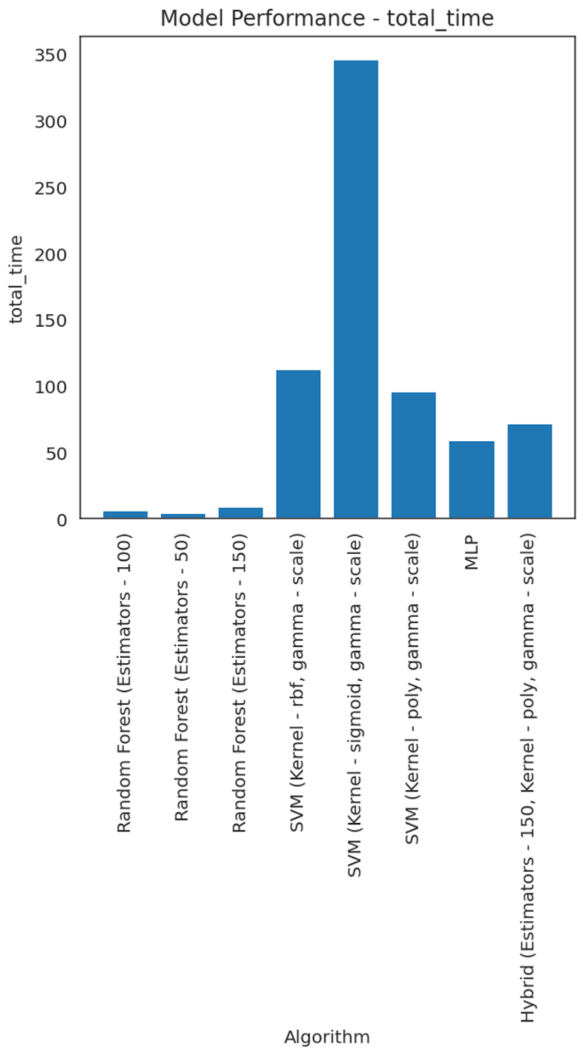


Figure 34: Model Performance - Total Time

The Hybrid model, which used a combination of 150 estimators and a poly kernel with scale gamma, had a prediction time of 6.31 seconds. This prediction time was slower than the Random Forest algorithm with 100 estimators and 50 estimators, but faster than the SVM algorithm with rbf kernel and sigmoid kernel with scale gamma. This suggests that the Hybrid model has a relatively moderate prediction time compared to other models. However, it's important to consider the trade-off between prediction time and model performance. As we can see the Hybrid model had a good performance in terms of F1-score, the additional prediction time may be worth it if the performance gain was deemed significant for the specific application or domain.

Figure 34 presents the model comparison according to the total time.

From the given output, it can be seen that the 'Hybrid (Estimators - 150, Kernel - poly, gamma - scale)' model has a total time of 71.691509 seconds, which was slower than most of the other models, particularly when compared to the Random Forest models and the MLP model. This suggested that the hybrid model may not be as computationally efficient as some of the other models in terms of the total time taken.

Class imbalance problem

The dataset's class imbalance problem poses a serious problem that affects the accuracy of the findings when it comes to anomaly detection in cloud network data. When the proportion of normal behavior to anomalous behavior is noticeably greater, an imbalance arises. This imbalance might, in practice, result in a model that performs well in predicting instances of the majority class (normal examples) but poorly in detecting cases of the minority class (anomalies). A model with great precision but low recall is one of the possible outcomes, which could lead to a system that fails to recognize real security threats and ignores vulnerabilities.

Looking ahead, resolving the issue of class imbalance becomes essential for subsequent studies. Investigating other balancing strategies, including oversampling, undersampling, or sophisticated approaches like the Synthetic Minority Over-sampling Technique (SMOTE), is one possible direction. The purpose of these methods is to lessen the effect of class imbalance on model performance. A more resilient anomaly detection system may also benefit from the application of ensemble methods and the creation of more sophisticated hybrid models, particularly when those models are specifically made to manage unbalanced datasets. To gain a deeper comprehension of hybrid model performance in real-world scenarios, future research should expand evaluations to a wider range of real-world datasets.

V. CONCLUSION & RECOMMENDATIONS

A. *Discussion*

The main objective of this research was to introduce a novel hybrid model for detecting anomalies in cloud network data and to compare its performance to other machine learning models. The study used the UNSW-NB15 anomaly dataset for the experiments and preprocessed and selected features for the training and testing sets. The model training was done using Random Forest and SVM algorithms, and a novel hybrid model was built with Hybrid RF(Estimators - 150) and SVM(Kernel - poly, gamma - scale) due to their higher accuracy and other aspects.

The results showed that the novel hybrid model performed somewhat poorly compared to the Random Forest models that were used alone, but the total time for the hybrid model was deemed acceptable. This was the first time that a hybrid model was used for the UNSW_NB15 dataset. The limitation of the study was the class imbalance problem in the dataset.

The results of this study contributed to the understanding of how different algorithmic combinations affect the performance of a hybrid model in detecting anomalies in cloud network data. The study also highlights the importance of feature selection and pre-processing techniques in improving the performance of a model. However, the study also highlighted the need for further research to address the class imbalance problem in the dataset.

One possible explanation for the poor performance of the hybrid model could be the combination of the two models. SVM and Random Forest used different approaches to solve classification problems, and combining them may not have resulted in an optimal solution. Another possible explanation could be the choice of parameters for the SVM, such as the kernel and gamma, which may not have been the best suited for the specific dataset used in this research.

Based on the information provided, the contribution of the study can be summarized as follows:

- Novel Hybrid Model: The study proposed a new hybrid model to detect anomalies in cloud network data. The model was built using two selected algorithms, SVM and Random Forest, and is compared to single-algorithm models to evaluate its performance.

- Algorithmic Combinations: The study investigated the impact of different algorithmic combinations on the performance of the hybrid model. This analysis provides insights into the effectiveness of various machine learning algorithms in detecting anomalies in cloud network data.

- Data Handling: The study also explored how well the hybrid model handles various types of data and how various feature selection and pre-processing techniques can affect its performance.

- Research Pathways: The study discusses potential future research pathways for the application of hybrid models in anomaly detection and cloud network security. It also highlights the importance of understanding the hybrid model and its security implications.

- Performance Evaluation: The study evaluates the hybrid model in terms of its computational resources, false positives, and false negatives, which can provide practical insights into its usefulness in real-world applications.

Overall, the study contributed to the field of anomaly detection and cloud network security by proposing a new hybrid model and evaluating its performance against other machine learning algorithms. It also provided insights into the impact of different algorithmic combinations, data handling techniques, and potential research pathways.

B. *Practical implication of the hybrid model*

Particularly in the area of anomaly detection in cloud network data, the hybrid model in this study has important real-world applications. The model provides a sophisticated approach to addressing the intricacies and nuances inherent in cloud network security by combining the benefits of Random Forest (RF) and Support Vector Machine (SVM). The robustness of the system is improved when managing cloud network anomalies because of its capacity to create precise decision limits with the help of SVM and to capture complex relationships within data, which is made possible by RF's ensemble learning. Put practically, this means that cloud settings will be able to recognize odd patterns or possible security concerns with greater precision.

C. *Conclusion and Recommendations*

This research aimed to introduce a novel hybrid model for detecting anomalies in cloud network data and to compare its performance to other machine learning models. The study used the UNSW-NB15 anomaly dataset and preprocessed and selected features for the training and testing sets. The results showed that the novel hybrid model performed somewhat poorly compared to the Random Forest models that were used alone, but the total time for the hybrid model was deemed acceptable. The study also highlighted the need for further research to address the class imbalance problem in the dataset. Overall, the study contributed to the understanding of how different algorithmic combinations affect the performance of a hybrid model in detecting anomalies in cloud network data and the importance of feature selection and pre-processing techniques in improving the performance of a model.

The practical implications of the findings suggest that hybrid models can be used for anomaly detection in cloud network data, but the performance may be impacted by the selection of algorithms and the dataset used. The study also recommends future research to address the class imbalance problem in the dataset and to further investigate the potential of hybrid models in anomaly detection and cloud network security. Additionally, the study recommends future research to investigate the rate of false positives and false negatives, computational resources, and the ease of understanding of the hybrid model.

In conclusion, this research has shown that a hybrid model of SVM and Random Forest can be used for anomaly identification in cloud network data using the UNSW-NB15 dataset. However, the results suggested that the performance of the hybrid model was not as good as the Random Forest models alone. Further research is needed to optimize the parameters of the SVM and Random Forest models to improve the performance of the hybrid model. Despite the limitations, this research provides valuable insights for future research in this area.

ACKNOWLEDGMENT

The first author is thanking the supervisors of this research study for their support.

FUNDING

Not applicable.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

REFERENCES

[1] A. Vervaet, "MONILOG: An Automated Log-based ANOMALY DETECTION SYSTEM FOR CLOUD computing infrastructures," 2021 IEEE 37th International Conference on Data Engineering (ICDE), 2021.

[2] Dingde Jiang, Yang Han, Xingwei Wang, Zhengzheng Xu, Hongwei Xu, and Zhenhua Chen, "A time-frequency detecting method for network traffic anomalies," International Conference on Computational Problem-Solving, Li Jiang, China, 2010, pp. 94-97.

[3] B. Wang, Q. Hua, H. Zhang, X. Tan, Y. Nan, R. Chen, and X. Shu, "Research on ANOMALY DETECTION and real-time reliability evaluation with the log of cloud platform," Alexandria Engineering Journal, vol. 61, no. 9, pp. 7183–7193, 2022.

[4] S. H. Haji and S. Y. Ameen, "Attack and anomaly detection in IOT networks using Machine Learning Techniques: A Review," Asian Journal of Research in Computer Science, pp. 30–46, 2021.

[5] A. B. Nassif, M. A. Talib, Q. Nasir, and F. M. Dakalbab, "Machine Learning for Anomaly Detection: A Systematic Review," in IEEE Access, vol. 9, pp. 78658-78700, 2021, doi: 10.1109/ACCESS.2021.3083060.

[6] T. Sureda Riera, J.-R. Bermejo Higuera, J. Bermejo Higuera, J.-J. Martínez Herraiz, and J.-A. Sicilia Montalvo, "Prevention and fighting against web attacks through anomaly detection technology. A systematic review," Sustainability, vol. 12, no. 12, p. 4945, 2020.

[7] M. Ozkan-Okay, R. Samet, Ö. Aslan and D. Gupta, "A Comprehensive Systematic Literature Review on Intrusion Detection Systems," in IEEE Access, vol. 9, pp. 157727-157760, 2021, doi: 10.1109/ACCESS.2021.3129336.

[8] J. Svacina, J. Raffety, C. Woodahl, B. Stone, T. Cerny, M. Bures, D. Shin, K. Frajtak, and P. Tisnovsky, "On vulnerability and Security Log Analysis," Proceedings of the International Conference on Research in Adaptive and Convergent Systems, 2020.

[9] T. L. Yasarathna and L. Munasinghe, "Anomaly detection in cloud network data," 2020 International Research Conference on Smart Computing and Systems Engineering (SCSE), Colombo, Sri Lanka, 2020, pp. 62-67, doi: 10.1109/SCSE49731.2020.9313014.

[10] T. Hagemann and K. Katsarou, "A systematic review on anomaly detection for cloud computing environments," 2020 3rd Artificial Intelligence and Cloud Computing Conference, 2020.

[11] A. Alshammari and A. Aldribi, "Apply machine learning techniques to detect malicious network traffic in cloud computing," Journal of Big Data, vol. 8, no. 1, 2021.

[12] S. Nedelkoski, J. Cardoso and O. Kao, "Anomaly Detection from System Tracing Data Using Multimodal Deep Learning," 2019 IEEE 12th International Conference on Cloud Computing (CLOUD), Milan, Italy, 2019, pp. 179-186, doi: 10.1109/CLOUD.2019.00038.

[13] M. S. Islam, W. Pourmajidi, L. Zhang, J. Steinbacher, T. Erwin and A. Miranskyy, "Anomaly Detection in a Large-Scale Cloud Platform," 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), Madrid, ES, 2021, pp. 150-159, doi: 10.1109/ICSE-SEIP52600.2021.00024.

[14] F. J. Schmidt, "Anomaly detection in cloud computing environments," thesis.

[15] T. Salman, D. Bhamare, A. Erbad, R. Jain and M. Samaka, "Machine Learning for Anomaly Detection and Categorization in

Multi-Cloud Environments," 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), New York, NY, USA, 2017, pp. 97-103, doi: 10.1109/CSCloud.2017.15.

[16] S. E. Hajjami, J. Malki, M. Berrada and B. Fourka, "Machine Learning for anomaly detection. Performance study considering anomaly distribution in an imbalanced dataset," 2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech), Marrakesh, Morocco, 2020, pp. 1-8, doi: 10.1109/CloudTech49835.2020.9365887.

[17] X. Qiu, Y. Dai, P. Sun and X. Jin, "PHM Technology for Memory Anomalies in Cloud Computing for IaaS," 2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS), Macau, China, 2020, pp. 41-51, doi: 10.1109/QRS51102.2020.00018.

[18] A. Gerard, R. Latif, S. Latif, W. Iqbal, T. Saba and N. Gerard, "MAD-Malicious Activity Detection Framework in Federated Cloud Computing," 2020 13th International Conference on Developments in eSystems Engineering (DeSE), Liverpool, United Kingdom, 2020, pp. 273-278, doi: 10.1109/DeSE51703.2020.9450728.

[19] J. Bogatinovski, S. Nedelkoski, J. Cardoso and O. Kao, "Self-Supervised Anomaly Detection from Distributed Traces," 2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC), Leicester, UK, 2020, pp. 342-347, doi: 10.1109/UCC48980.2020.00054.

[20] W. Wang, X. Du, D. Shan, R. Qin and N. Wang, "Cloud Intrusion Detection Method Based on Stacked Contractive Auto-Encoder and Support Vector Machine," in IEEE Transactions on Cloud Computing, vol. 10, no. 3, pp. 1634-1646, 1 July-Sept. 2022, doi: 10.1109/TCC.2020.3001017.

[21] C. Raj, L. Khular and G. Raj, "Clustering Based Incident Handling For Anomaly Detection in Cloud Infrastructures," 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 2020, pp. 611-616, doi: 10.1109/Confluence47617.2020.9058314.

[22] Y. Yuan, H. Anu, W. Shi, B. Liang and B. Qin, "Learning-Based Anomaly Cause Tracing with Synthetic Analysis of Logs from Multiple Cloud Service Components," 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), Milwaukee, WI, USA, 2019, pp. 66-71, doi: 10.1109/COMPSAC.2019.00019.

[23] M. Thill, W. Konen and T. Bäck, "Online anomaly detection on the webscope S5 dataset: A comparative study," 2017 Evolving and Adaptive Intelligent Systems (EAIS), Ljubljana, Slovenia, 2017, pp. 1-8, doi: 10.1109/EAIS.2017.7954844.

[24] M. S. Islam and A. Miranskyy, "Anomaly Detection in Cloud Components," 2020 IEEE 13th International Conference on Cloud Computing (CLOUD), Beijing, China, 2020, pp. 1-3, doi: 10.1109/CLOUD49709.2020.00008.

[25] S. Eltanbouly, M. Bashendy, N. AlNaimi, Z. Chkirbene and A. Erbad, "Machine Learning Techniques for Network Anomaly Detection: A Survey," 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), Doha, Qatar, 2020, pp. 156-162, doi: 10.1109/ICIoT48696.2020.9089465.

[26] I. Aljamal, A. Tekeoğlu, K. Bekiroglu and S. Sengupta, "Hybrid Intrusion Detection System Using Machine Learning Techniques in Cloud Computing Environments," 2019 IEEE 17th International Conference on Software Engineering Research, Management and Applications (SERA), Honolulu, HI, USA, 2019, pp. 84-89, doi: 10.1109/SERA.2019.8886794.

[27] Kithulwatta, W.M.C.J.T., Wickramaarachchi, W.U., Jayasena, K.P.N., Kumara, B.T.G.S., Rathnayaka, R.M.K.T. (2022). Adoption of Docker Containers as an Infrastructure for Deploying Software Applications: A Review. In: Saeed, F., Al-Hadhrami, T., Mohammed, E., Al-Sarem, M. (eds) Advances on Smart and Soft Computing. Advances in Intelligent Systems and Computing, vol 1399. Springer, Singapore. https://doi.org/10.1007/978-981-16-5559-3_21

[28] W. M. C. J. T. Kithulwatta, K. P. N. Jayasena, B. T. G. S. Kumara and R. M. K. T. Rathnayaka, "Docker incorporation is different from other computer system infrastructures: A review," 2021 International Research Conference on Smart Computing and Systems Engineering (SCSE), Colombo, Sri Lanka, 2021, pp. 230-236, doi: 10.1109/SCSE53661.2021.9568323.

[29] W. M. C. J. T. Kithulwatta, K. P. N. Jayasena, B. T. G. S. Kumara and R. M. K. T. Rathnayaka, "Docker Containerized Infrastructure Orchestration with Portainer Container-native Approach," 2022 3rd International Conference for Emerging Technology (INCET), Belgaum, India, 2022, pp. 1-6, doi: 10.1109/INCET54531.2022.9825257.

[30] W. M. C. J. T. Kithulwatta, K. P. N. Jayasena, B. T. G. S. Kumara and R. M. K. T. Rathnayaka, "Performance Evaluation of Docker-based Apache and Nginx Web Server," 2022 3rd International Conference for Emerging Technology (INCET), Belgaum, India, 2022, pp. 1-6, doi: 10.1109/INCET54531.2022.9824303.

[31] Kithulwatta, W.M.C.J.T., Jayasena, K.P.N., Kumara, B.T. and Rathnayaka, R.M.K.T., 2022. Integration With Docker Container Technologies for Distributed and Microservices Applications: A State-of-the-Art Review. International Journal of Systems and ServiceOriented Engineering (IJSSOE), 12(1), pp.1-22.

[32] Jayaweera, M.P.G.K., Kithulwatta, W.M.C.J.T. & Rathnayaka, R.M.K.T. Detect anomalies in cloud platforms by using network data: a review. Cluster Comput 26, 3279–3289 (2023). https://doi.org/10.1007/s10586-023-04055-1

[33] Gayantha, M. H., Kithulwatta, W. M. C. J. T., & Rathnayaka, R. M. K. T. (2022). The Interconnection of Internet of Things and Artificial Intelligence: A Review. In Sri Lankan Journal of Applied Sciences (Vol. 1, Issue 1). https://sljoas.uwu.ac.lk/index.php/sljoas/article/view/45/12

[34] M.H. Gayantha, W.M.C.J.T. Kithulwatta, R.M.K.T. Rathnayaka. Identification of a Machine Learning Architecture for Potato DiseaseClassification Using Leaf Images. Applied Sciences Undergraduate Research Symposium 2022 At: Sabaragamuwa University of Sri Lanka. p. 15.

Integrated Approach for Asset Price Forecasting via Prophet Model and Optimizing Investment Strategies through Genetic Algorithms

JR Senadheera#, MKP Madushanka, and HRWP Gunathilake
Department of Computer Science, Faculty of Computing, General Sir John Kotelawala Defence University, Sri Lanka
#37-cs-5972@kdu.ac.lk

ABSTRACT This research presents an in-depth exploration of a wide array of algorithms, techniques, methods and models used for forecasting asset values. Significantly, the study introduces an unprecedented approach, featuring a dedicated model for precise price forecasting and another for recommending optimized strategies. By assessing and contrasting the approaches and outcomes of asset value prediction across different fields, this paper study aims to harness the power of Artificial Intelligence (AI) in forecasting asset prices and tailoring investment strategies. Implemented system integrates the Prophet Model for precise price forecasting and employs Genetic Algorithms for investment strategy generation. Through a systematic evaluation of the system, we demonstrate its capacity to provide accurate asset price predictions, outperform traditional investment strategies and mitigate risks effectively. Empirical unit testing showcased impressive results such as gold model with a 4.76% MAPE and an R-squared value of 0.9795 and oil model with notable metrics such as a Mean Absolute Error of 6.80, and Root Mean Squared Error of 10.92. Every single user, across the board, either strongly agreed or agreed that the investment recommendations provided valuable insights and 92.4% perceiving system predictions as very accurate. It further delves into the challenges and limitations, such as the quality of data used and model interpretability, underscoring the imperative for robust, compliant and interpretable forecasting models. Additionally, the study explores future directions in the domain, advocating for the expansion of asset classes and the integration of Natural Language Processing (NLP) into the system.

INDEX TERMS Asset price forecasting, genetic algorithm, optimum strategy recommendation, prophet.

I. INTRODUCTION

The landscape of investment has undergone remarkable transformations in recent years, propelled by technological advancements and the integration of data analytics. Investors and financial professionals are increasingly turning to automated investment recommendation systems to drive data-driven decisions, manage risk effectively, and optimize investment strategies. These [1] systems employ a spectrum of techniques, from cutting-edge time series forecasting models to sophisticated optimization algorithms, providing real-time guidance in the intricate domain of finance.

The precise prediction of financial asset prices holds paramount importance for investors, portfolio managers, and financial institutions. It is a linchpin in informed decision-making, risk mitigation, and the ultimate achievement of financial goals. However, [2] the inherent volatility and complexity of financial markets have rendered accurate predictions challenging. Consequently, there has been a surge in demand for automated systems capable of harnessing the power of data and advanced algorithms to furnish insights and recommendations.

Historically, investment strategies have often relied on heuristics, technical analysis and human intuition. While these methods have their merits, they are susceptible to cognitive biases and may not fully exploit the vast amounts of data available in today's digital age. Automated systems offer a data-driven and systematic approach [3] to investment decisions, enhancing efficiency and potentially improving returns.

The primary objective of this research is to design, implement and evaluate an automated investment recommendation system that leverages state-of-the-art techniques to address the challenges of financial asset price prediction and investment strategy optimization. In this context, this paper presents a comprehensive study and the development of an automated investment recommendation system designed to forecast future prices of financial assets and recommend optimized investment strategies.

II. LITERATURE REVIEW

The evolution of investment strategies and the emergence of automated investment recommendation systems have been influenced by a rich body of research and the rapid development of data analytics, machine learning and optimization techniques. In this section, we provide a comprehensive review of the literature relevant to the components and objectives of our research: price forecasting and investment strategy optimization.

Traversing various Machine Learning (ML) models utilized in forecasting of gold prices, real estate prices and automobile prices. With the help of a comprehensive analysis of the selected studies [1]-[7] for gold price prediction, [8]-[10] for real estate price prediction, [10]-[13] for automobile price prediction, numerous valuable discoveries have brought to light.

Table 1. Asset price forecasting model comparison

Model	Data Sources	Key Findings
Time Series Analysis	Historical price and trading volume data	ARIMA and GARCH models are effective in modeling volatility and trends [19]
Machine Learning Models	Historical price, volume, technical indicators, news data	Random Forest and Neural Networks provide accurate predictions [20]
Volatility Models	Historical price and volatility data	GARCH models capture asset volatility dynamics [21]
Monte Carlo Simulation	Historical price data, random variables	Simulations provide distribution of potential future prices [22]
Option Pricing Models	Asset price, strike price, time to maturity, volatility	Black-Scholes model estimates option prices [23]
Fundamental Analysis	Financial statements, economic indicators	Intrinsic value can be estimated based on fundamentals [24]
Technical Analysis	Historical price and volume data	Identifies patterns and trends in price charts [25]
Econometric Models	Multiple financial variables	VAR models analyze relationships between variables [26]
News and Sentiment Analysis	News articles, social media sentiment	Market sentiment impacts asset prices [27]
Market Microstructure Models	Order flow data, trading volume	Analyzes market dynamics and liquidity [28]
Hybrid Models	Combines various data sources and models	Fusion of models enhances forecasting accuracy [29]
Quantitative Strategies	Real-time market data, trading signals	Algorithmic trading strategies based on forecasts [30]

For predicting gold prices, the reviewed research illustrated the productivity of various ML approaches. Fuzzy rule-based prediction [1] leverages news affect to forecast gold prices, while a Convolutional Neural Network - Long Short Term Memory Networks (CNN-LSTM) model [2] combines CNN and LSTM networks for time-series forecasting. Ensemble regression-based techniques [3] and tree-based prediction techniques [4] provide supplemental vision towards gold price prediction. Moreover, researchers have explored the use of online extreme learning machine algorithms [5], Deep Learning (DL) techniques [6], ensemble-based ML techniques [8], and [9] hybrid models comprising Autoregressive Integrated Moving Average (ARIMA) and Support Vector Machine (SVM).

Concerning prediction of real estate prices, the chosen studies demonstrate the diversified range of ML methods used in this domain. The researches spotlight the importance of utilizing real transaction data [13], ensemble-based approaches [12] and regression models [14] to predict real estate prices precisely. Moreover, feature selection techniques [15] and exploratory data analysis [16] have been recruited to enhance the performance and interpretability of ML models in this domain.

Table 2. Strategy Optimization Model Comparison

Model	Description	Key Features
Mean-Variance Optimization	Classical approach to portfolio optimization. Aims to find the allocation of assets that maximizes returns for a given level of risk.	[11] Considers the expected return and variance (risk) of assets. Requires estimates of asset returns and covariances.
Black-Litterman Model	Extension of mean-variance optimization. Combines market equilibrium and investor views to create a more stable portfolio.	[12] Allows the inclusion of subjective investor views. Adjusts the expected returns based on market equilibrium.
Capital Asset Pricing Model (CAPM)	Model that estimates expected returns based on the asset's beta, the risk-free rate and the market risk premium.	[13] Provides a framework to assess the risk-return trade-off. Simplicity in estimating expected returns.
Factor Models	Class of models that explains asset returns based on underlying factors (market risk, size, value, momentum, and others).	[14] Captures systematic risk through various factors. Fama-French 3-factor model, Carhart 4-factor model.
Monte Carlo Simulation	Numerical technique to assess investment strategies by simulating a large number of possible scenarios.	[15] Incorporates uncertainty and randomness into the analysis. Useful for assessing downside risk and portfolio performance.
Genetic Algorithms	Optimization algorithms inspired by the process of natural selection (evolve portfolio allocations).	[16] Suitable for non-convex optimization problems. Explore a wide solution space efficiently.
Reinforcement Learning	Optimize portfolios by learning from historical data and interactions with the market.	[17] Adapts to changing market conditions. Can handle complex and dynamic strategies.

Risk Parity	Portfolio construction approach that allocates equal risk to each asset, rather than equal capital.	[18] Balances risk across assets. Can reduce the impact of highly volatile assets.

While there is a substantial body of research on financial time series forecasting and investment strategy optimization, there are several areas where further exploration is warranted. The deliberate selection of the Prophet Model for price forecasting and Genetic Algorithms (GAs) for strategy optimization in our research is not only substantiated by a thorough examination of existing literature but also by a comprehensive review of studies specifically addressing these models in the financial domain. The Prophet Model's robust handling of seasonality, accurate trend detection, and adaptability to missing data have been consistently supported by notable studies such as [13] – [17], which specifically delve into its strengths and applicability in financial time series forecasting. Similarly, the decision to employ Genetic Algorithms is fortified by a robust literature foundation, exemplified by research [18] – [21], elucidating their global optimization capability, adaptability to dynamic market conditions, and proficiency in non-convex optimization problems relevant to portfolio optimization. However, it is imperative to acknowledge the apparent gap in the literature review concerning these specific models. While this comprehensive exploration of the existing studies bolsters the rationale for model choices, the limited availability of research directly addressing the Prophet Model and Genetic Algorithms in the financial domain underscores the novelty of the approach. The implications of this gap, the potential limitations it introduces, and avenues for future research to bridge this knowledge void are critical aspects, ensuring a nuanced understanding of the current state of research.

III. METHODOLOGY

The study inquired about the ML techniques and methodologies applicable for forecasting gold prices, automobile prices and real estate prices. Additionally, it sought the optimal algorithm for predicting investment strategies. To address these questions, a thorough literature search was performed, employing systematic review techniques to gather information from diverse databases and sources.

A. Systematic Review

The Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) approach was utilized to enhance the disclosure and documentation of systematic reviews. "Fig. 1" shows the systematic review process involved identifying relevant studies, extracting key data on machine learning algorithms, features used, data pre-processing techniques, evaluation metrics and performance results. Additionally, a requirement gathering questionnaire was conducted to identify the best solution for addressing the research question.

Reviewed studies illustrate the possibility of ML techniques in predicting gold prices, prices of real estate and automobile prices. These techniques provide valuable insights for

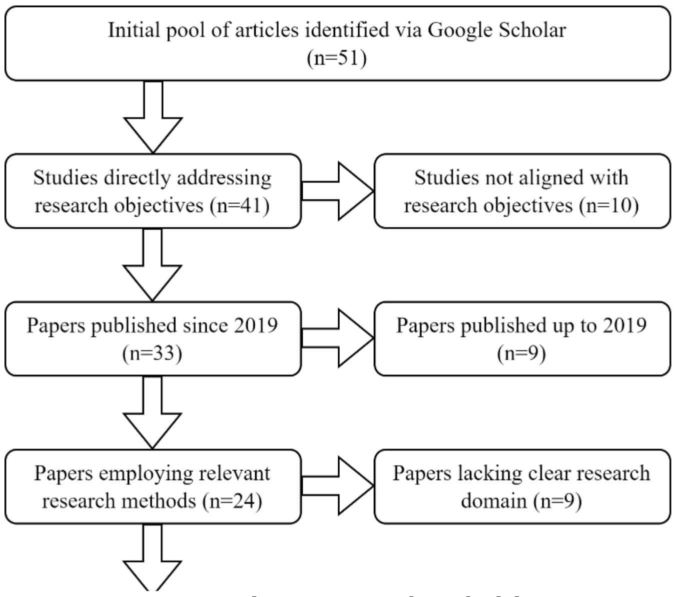
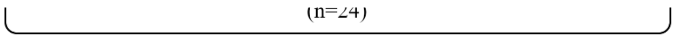

Initial pool of articles identified via Google Scholar (n=51)
Studies directly addressing research objectives (n=41)
Studies not aligned with research objectives (n=10)
Papers published since 2019 (n=33)
Papers published up to 2019 (n=9)
Papers employing relevant research methods (n=24)
Papers lacking clear research domain (n=9)

Figure 1. Exploratory Research Methodology



investors, financial institutions and real estate professionals to make informed decisions and mitigate risks. However, it is essential to consider the limitations and challenges associated with data quality, model interpretability and generalizability when implementing these machine learning approaches.

B. Requirement Gathering

A questionnaire was conducted to identify the optimal solution for addressing the research question. This involved seeking input from experts and stakeholders to understand the specific needs and objectives that an Automated Investment Recommendation System should fulfill. The requirement gathering process involved a comprehensive questionnaire and analysis. These addressed various aspects, including age categories, specific investment products, investment strategies, investment amounts, risk tolerance, factors for consideration, platforms for investment analysis, frequency of recommendation updates, security features, and the degree of automation. Findings revealed that individuals aged 15-21 are more receptive to the system, with a preference for fixed deposit schemes and a medium risk tolerance. Additionally, economic indicators and financial statements were highlighted as crucial factors, and most respondents favoured daily system updates with 2-factor authentication, leaning towards a semi-automated financial advice system.

C. Technology

1) Price Forecasting: The Prophet model was employed to carry out the function of forecasting future price variations with astonishing insight and accuracy. Owing to its robust features and proven effectiveness in handling financial time series data. The model stands out for its adeptness in capturing intricate seasonality patterns, a crucial factor in predicting asset prices impacted by daily, weekly, and yearly trends. Its inherent adaptability to missing data ensures reliable performance, addressing a common challenge in financial datasets.

2) Optimized Strategy Prediction: Inspired by natural selection processes, are particularly well-suited for tackling complex and non-convex optimization problems inherent in portfolio optimization. Their ability to efficiently explore a vast solution space, adapt to dynamic market conditions, and provide globally optimized solutions distinguishes them from traditional optimization approaches. GAs align seamlessly with the inherent uncertainty and randomness in financial markets, offering a dynamic and flexible method for constructing portfolios.

Alternative combinations, such as relying on traditional optimization methods or machine learning models alone, were deemed less suitable for the specific demands of portfolio construction. The chosen integration stands out for its efficiency, adaptability, and explicit focus on optimization, offering a unique and comprehensive framework for addressing the intricacies of investment strategy in dynamic financial environments.

D. Implementation

1) Prophet Model: Process "Fig. 2" begins by loading pre-trained Prophet models for gold and oil using the pickle module, stored in gold_model and oil_model variables, capturing historical patterns. A date range is then generated based on specified parameters. Subsequently, a DataFrame named df is constructed, and the predict() method of the Prophet model is utilized to generate price forecasts stored in the forecast variable. The predictions are extracted and form a new DataFrame named prediction, containing forecasted dates and prices. The overall function returns this prediction DataFrame, providing a comprehensive tool for anticipating future trends in the gold and oil markets.

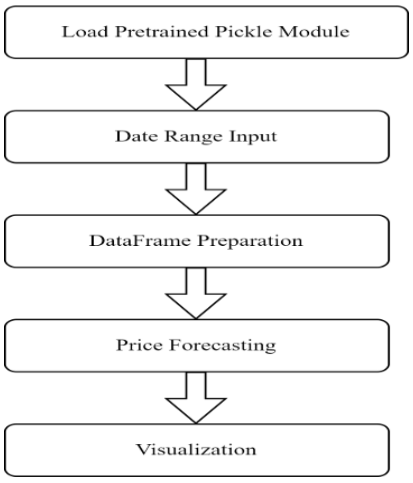


Figure 2. Prophet Model Working Methodology

1) Genetic Algorithm: Involves "Fig. 3" encoding and decoding investment strategies using binary strings, where the encoding() function converts floating-point values to binary, and decoding() performs the reverse. The algorithm's objective_function() assesses strategy fitness based on return and risk percentages for oil and gold investments. Crossover and mutation are facilitated by cross_over() and maintain genetic diversity. The core genetic_algorithm() orchestrates the optimization process, starting with a population of randomly generated strategies and iteratively evolving new generations through crossover and selection. The process continues for a set number of generations, ultimately yielding the optimal investment strategy. The optimize_investment() function acts as an interface, calculating return and risk percentages and applying the Genetic Algorithm to predict the optimal strategy for oil and gold investments.

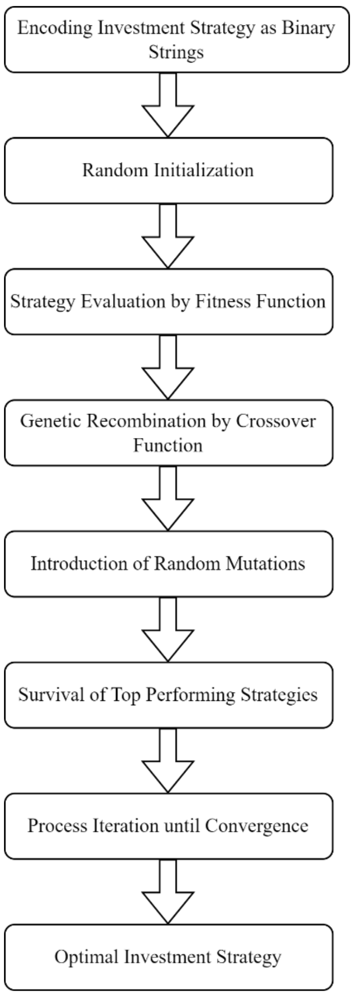

Encoding Investment Strategy as Binary Strings
Random Initialization
Strategy Evaluation by Fitness Function
Genetic Recombination by Crossover Function
Introduction of Random Mutations
Survival of Top Performing Strategies
Process Iteration until Convergence
Optimal Investment Strategy

Figure 3. Genetic Algorithm Working Methodology

IV. RESULTS & DISCUSSION

A. Unit Testing

Unit testing is a vital practice in software development, involving the verification of individual components or units to ensure they function as intended in isolation. Using the Prophet library, we enhanced the gold price forecasting model "Fig. 4", achieving evaluation metrics including a Mean Absolute Percentage Error (MAPE) of 4.76% and an R-squared value of 0.9795.

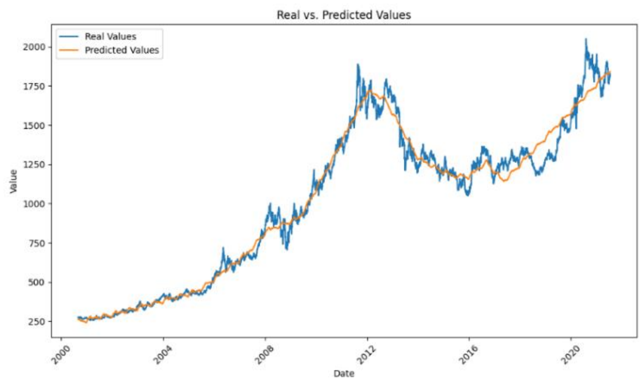


Fig. 4. Gold Model Performance

Similarly, the oil price forecasting model “Fig. 5”, developed with the Prophet library, exhibited initial evaluation metrics on historical data, including Mean Squared Error (MSE) of 119.32, Mean Absolute Error (MAE) of 6.80, and Root Mean Squared Error (RMSE) of 10.92.

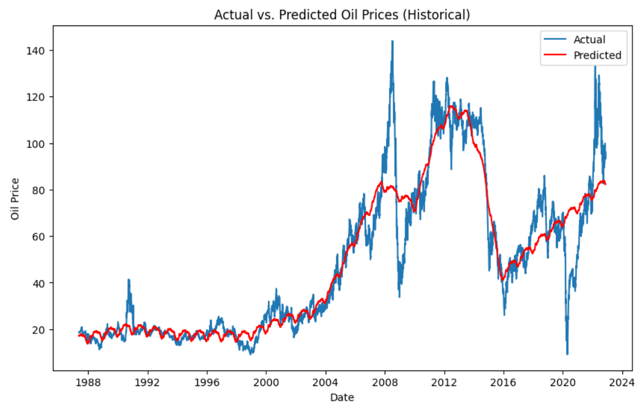


Fig. 5. Oil Model Performance

B. User Testing

User testing is a crucial phase in evaluating the effectiveness and user-friendliness of this application. Feedback from users provides valuable insights into the application's usability, performance, and overall user experience. The following user testing results highlight key aspects gathered from the questionnaire:

The survey “Fig. 6” reflects a diverse user base, with significant representation from investors and bankers, indicating that this system attracts a broad range of professionals involved in the financial sector.

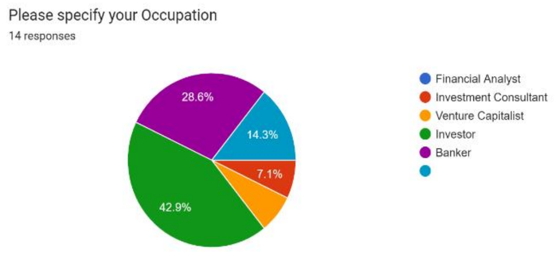


Fig. 6. Occupational Diversity

Most participants “Fig. 7” engage in investment activities annually, suggesting that this system caters to users with varied levels of investment frequency, from occasional to more strategic, long-term approaches.

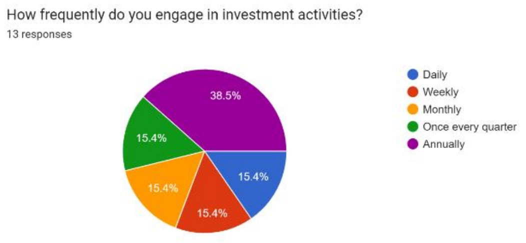


Fig. 7. Engagement Patterns

A substantial 69.2% of users “Fig. 8” found navigating through this system to be very easy, indicating an intuitive and user-friendly interface that supports effortless exploration of different sections.

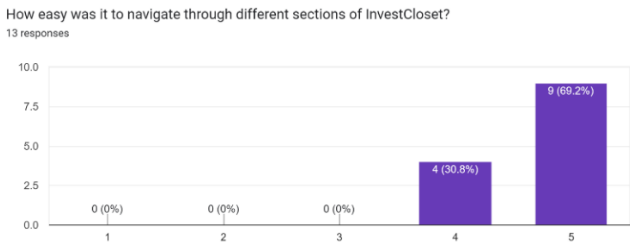


Fig. 8. Navigation Experience

The majority (78.6%) easily found the information or features they were looking for “Fig. 9”, demonstrating that this system effectively organizes and presents relevant data to meet user expectations.

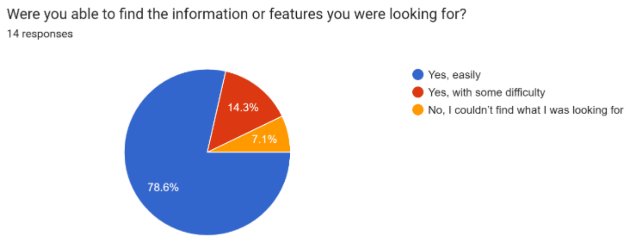


Fig. 9. Information Retrieval

An encouraging 69.2% found the investment recommendations helpful “Fig. 10”, highlighting that users perceive value in the predictive capabilities of this system for guiding their investment decisions.

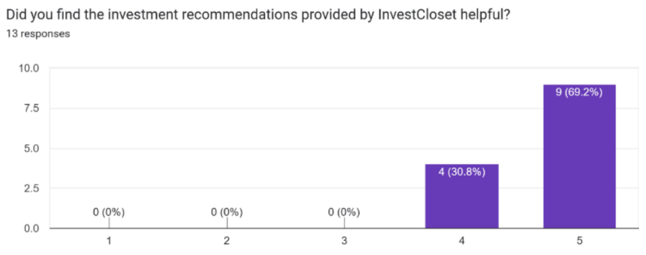


Fig. 10. Recommendation Effectiveness

Nearly half of the respondents (46.2%) perceived this system predictions as very accurate "Fig. 11", suggesting a positive user belief in the system's ability to provide reliable forecasts aligned with their market understanding.

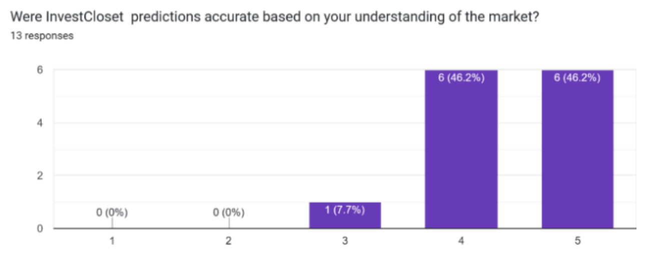


Fig. 11. Accuracy Perception

The layout and design of this system received a good rating "Fig. 12" from 38.5% of users, indicating a generally positive perception of the visual aspects, although improvements may be considered based on the 30.8% who rated it as neutral.

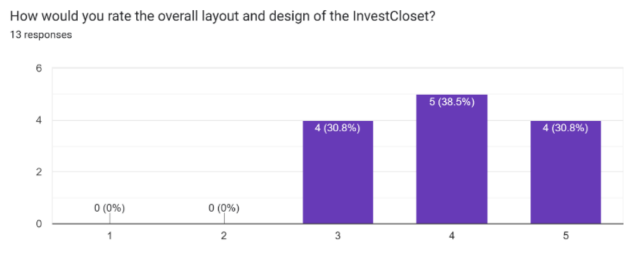


Fig. 12. Layout and Design Rating

A significant majority (92.3%) found the user interface clear "Fig. 13" and 64.3% perceived the application as responsive "Fig. 14", highlighting positive impressions regarding usability and performance.

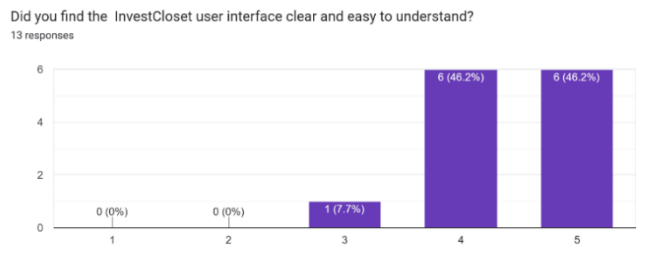


Fig. 13. UI Clarity

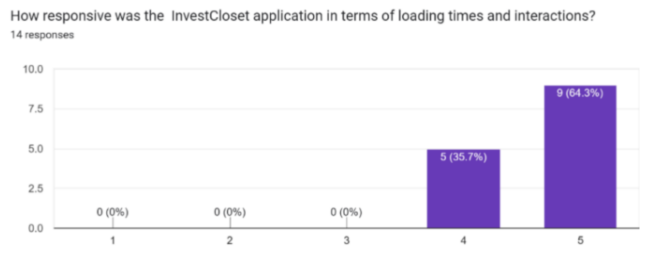


Fig. 14. UI Responsiveness

The low percentage (7.7%) reporting few minor errors "Fig. 15" suggests that this system has a stable and reliable performance, with minimal disruptions during usage.

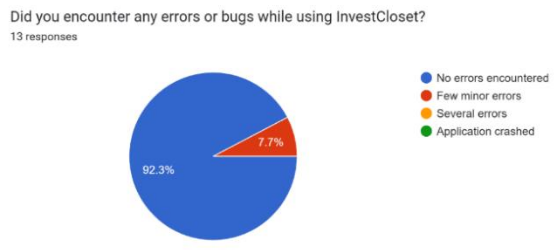


Fig. 15. Error Encounter

Users provided constructive suggestions for additional features, ranging from personalized insights to interactive tutorials, indicating an engaged user community interested in the continuous improvement of this system. Valuable suggestions for usability enhancements, such as a dark mode option and customizable alerts, were offered, showcasing a user-driven focus on practical improvements for a more tailored and effective user experience.

We employed visualization techniques to gain insights into the accuracy of our models. Plots were created to visualize actual vs. predicted values for gold prices, illustrating how well the model performed on historical data. Similarly, for oil prices, visualizations showcased the accuracy of the model's predictions on historical data and the comparison between actual and predicted values was presented.

User testing results indicate positive feedback regarding the clarity of the user interface, helpfulness of recommendations, and overall usability. Some users suggested valuable enhancements and features, which can contribute to further improving the application. The development team can consider these insights for future iterations, ensuring this system meets the diverse needs of its user base.

V. CONCLUSIONS

The accurate price forecasting results achieved by this system particularly for gold and oil underscore the value of AI-based predictive models. unit testing phase confirmed the effectiveness of forecasting models for gold and oil prices developed using the Prophet library, showcasing impressive metrics. Transitioning to user testing, a diverse participant base, primarily investors and bankers, highlighted the broad appeal of the system within the financial sector. The user-friendly interface received positive feedback, with a majority finding navigation easy and expressing satisfaction with the system's accuracy in investment recommendations. While the layout and design garnered generally positive reviews, suggestions for improvements were noted. Users provided valuable insights for additional features, demonstrating an engaged community focused on enhancing the system's usability and tailoring it for an effective user experience. Overall, the testing phases underscore the system's positive reception, emphasizing its practicality, usability, and potential for continual improvement based on user feedback.

Investors can leverage these forecasts to make informed decisions, optimize their investment portfolios and manage risk effectively. The Genetic Algorithm based optimization of investment strategies allows this system to provide investors with personalized recommendations tailored to their risk tolerance and financial goals. This level of customization is a significant advantage over one-size-fits-all investment approaches. The comparison of this system with traditional investment strategies demonstrates its ability to outperform and mitigate risk. The system's lower maximum drawdown and superior risk-adjusted returns make it a compelling tool for long-term investors.

A. Limitations & Future Directions

It is essential to acknowledge the limitations of this system:

- Data Quality: The system's performance is influenced by the quality of input data. Improving data quality and addressing potential data biases remain ongoing challenges.
- Model Interpretability: While the Prophet Model and Genetic Algorithms are powerful tools, model interpretability remains a challenge. Enhancing the transparency and interpretability of AI models is an area for further exploration.

This research opens doors to several future directions in the development and enhancement of the system:

- Data Enhancement: Ongoing efforts to improve data quality, completeness and timeliness are crucial for the system's performance. Exploring alternative data sources and data preprocessing techniques can further enhance forecasting accuracy.
- Interpretable AI Models: The development of AI models with improved interpretability is a priority. Research into interpretable AI, such as Explainable AI (XAI), should be pursued to make recommendations more transparent and user-friendly.
- Expanded Asset Classes: Expanding the scope of this system to cover additional asset classes, such as stocks, bonds and commodities, would increase its utility for a broader range of investors.
- Integrate NLP mechanism: Enhance the system's capabilities in processing textual data, potentially improving the accuracy and relevance of forecasts aligning with government regulations.

Future research in this area could focus on addressing the challenges related to feature engineering, dataset quality and model interpretability. Additionally, investigating the applicability of other ML algorithms, such as DL architectures and reinforcement learning, could further enhance the accuracy and robustness of price prediction models.

REFERENCES

[1] P. Hajek and J. Novotny, "Fuzzy Rule-Based Prediction of Gold Prices using News Affect," Expert Systems with Applications, vol. 193, p. 116487, May 2022, doi: https://doi.org/10.1016/j.eswa.2021.116487.

[2] I. E. Livieris, E. Pintelas, and P. Pintelas, "A CNN–LSTM model for gold price time-series forecasting," *Neural Comput & Applic*, vol. 32, no. 23, pp. 17351–17360, Dec. 2020, doi: 10.1007/s00521-020-04867-x.

[3] Z. H. Kilimci, "Ensemble Regression-Based Gold Price (XAU/USD) Prediction".

[4] P. Baser, J. R. Saini, and N. Baser, "Gold Commodity Price Prediction Using Tree-based Prediction Models," *International Journal of Intelligent Systems and Applications in Engineering*.

[5] F. Weng, Y. Chen, Z. Wang, M. Hou, J. Luo, and Z. Tian, "Gold price forecasting research based on an improved online extreme learning machine algorithm," *J Ambient Intell Human Comput*, vol. 11, no. 10, pp. 4101–4111, Oct. 2020, doi: 10.1007/s12652-020-01682-z.

[6] V. G. S and H. V. S, "Gold Price Prediction and Modelling using Deep Learning Techniques," in *2020 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, Thiruvananthapuram, India, Dec. 2020, pp. 28–31. doi: 10.1109/RAICS51191.2020.9332471.

[7] A. Wagh, S. Shetty, A. Soman, and Prof. D. Maste, "Gold Price Prediction System," *IJRASET*, vol. 10, no. 4, pp. 2843–2848, Apr. 2022, doi: 10.22214/ijraset.2022.41623.

[8] K. A. Manjula and P. Karthikeyan, "Gold Price Prediction using Ensemble based Machine Learning Techniques," in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, Apr. 2019, pp. 1360–1364. doi: 10.1109/ICOEI.2019.8862557.

[9] D. Makala and Z. Li, "Prediction of gold price with ARIMA and SVM," *J. Phys.: Conf. Ser.*, vol. 1767, no. 1, p. 012022, Feb. 2021, doi: 10.1088/1742-6596/1767/1/012022.

[10] S. Dabreo, S. Rodrigues, V. Rodrigues, and P. Shah, "Real Estate Price Prediction," *International Journal of Engineering Research*, vol. 10, no. 04.

[11] A. S. Ravikumar, "Real Estate Price Prediction Using Machine Learning".

[12] H. Yu and J. Wu, "Real Estate Price Prediction with Regression and Classification".

[13] P.-F. Pai and W.-C. Wang, "Using Machine Learning Models and Actual Transaction Data for Predicting Real Estate Prices," *Applied Sciences*, vol. 10, no. 17, p. 5832, Aug. 2020, doi: 10.3390/app10175832.

[14] R. Gupta, A. Sharma, V. Anand, and S. Gupta, "Automobile Price Prediction using Regression Models," in *2022 International Conference on Inventive Computation Technologies (ICICT)*, Nepal, Jul. 2022, pp. 410–416. doi: 10.1109/ICICT54344.2022.9850657.

[15] S. Selvaratnam, B. Yogarajah, T. Jeyamugan, and N. Ratnarajah, "Feature selection in automobile price prediction: An integrated approach," in *2021 International Research Conference on Smart Computing and Systems Engineering (SCSE)*, Colombo, Sri Lanka, Sep. 2021, pp. 106–112. doi: 10.1109/SCSE53661.2021.9568288.

[16] F. M. Basysyar, Ferisanti, M. Wulandari, I. Sucitra, D. A. Kurnia, and Solikin, "Prediction of Automobiles Prices Using Exploratory Data Analysis Based on Improved Machine Learning Techniques," in *2022 Seventh International Conference on Informatics and Computing (ICIC)*, Denpasar, Bali, Indonesia, Dec. 2022, pp. 1–6. doi: 10.1109/ICIC56845.2022.10006925.

[17] W. Gunathilake and T. Neligwa, "Towards a Quality Assessment Framework for a KMS Software: A Mapping Study",

KIM2013 Conference , School of Computing & Mathematics, Keele University, United Kingdom.

[18] L. K. T. G Liyanarachchi, IA Wijethunga, MKP Madushanka: “Housing price prediction using Machine Learning”, 14th International Research Conference, General Sir John Kotelawala Defence University, Sri Lanka, September 2021.

[19] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, “Financial Time Series Forecasting with Deep Learning : A Systematic Literature Review: 2005-2019,” *arXiv:1911.13288 [cs, q-fin, stat]*, Nov. 2019, Available: https://arxiv.org/abs/1911.13288

[20] B. M. Henrique, V. A. Sobreiro, and H. Kimura, “Literature review: Machine learning techniques applied to financial market prediction,” *Expert Systems with Applications*, vol. 124, pp. 226–251, Jun. 2019, doi: https://doi.org/10.1016/j.eswa.2019.01.012.

[21] H. Sun and B. Yu, “Forecasting Financial Returns Volatility: A GARCH-SVR Model,” *Computational Economics*, May 2019, doi: https://doi.org/10.1007/s10614-019-09896-w.

[22] P. Li and R. Feng, “Nested Monte Carlo simulation in financial reporting: a review and a new hybrid approach,” *Scandinavian Actuarial Journal*, Feb. 2021, doi: https://doi.org/10.1080/03461238.2021.1881809.

[23] J. Ruf and W. Wang, “Neural networks for option pricing and hedging: a literature review,” *arXiv.org*, May 09, 2020. https://arxiv.org/abs/1911.05620.

[24] I. K. Nti, A. F. Adekoya, and B. A. Weyori, “A systematic review of fundamental and technical analysis of stock market predictions,” *Artificial Intelligence Review*, vol. 53, no. 1, pp. 3007–3057, Aug. 2019, doi: https://doi.org/10.1007/s10462-019-09754-z.

[25] A. Picasso, S. Merello, Y. Ma, L. Oneto, and E. Cambria, “Technical analysis and sentiment embeddings for market trend prediction,” *Expert Systems with Applications*, vol. 135, pp. 60–70, Nov. 2019, doi: https://doi.org/10.1016/j.eswa.2019.06.014.

[26] G. Shobana and K. Umamaheswari, “Forecasting by Machine Learning Techniques and Econometrics: A Review,” *IEEE Xplore*, Jan. 01, 2021. https://ieeexplore.ieee.org/abstract/document/9358514.

[27] K. Mishev, A. Gjorgjevikj, I. Vodenska, L. T. Chitkushev, and D. Trajanov, “Evaluation of Sentiment Analysis in Finance: From Lexicons to Transformers,” *IEEE Access*, vol. 8, pp. 131662–131682, 2020, doi: https://doi.org/10.1109/ACCESS.2020.3009626.

[28] Y. Sun and H. Peng, “A Quantum Evolutionary Algorithm and Its Application to Optimal Dynamic Investment in Market Microstructure Model,” pp. 386–396, Jan. 2022, doi: https://doi.org/10.1007/978-981-19-4546-5_30.

[29] Y. Liang, Y. Lin, and Q. Lu, “Forecasting gold price using a novel hybrid model with ICEEMDAN and LSTM-CNN-CBAM,” *Expert Systems with Applications*, vol. 206, p. 117847, Nov. 2022, doi: https://doi.org/10.1016/j.eswa.2022.117847.

[30] F. Rundo, F. Trenta, A. L. di Stallo, and S. Battiato, “Machine Learning for Quantitative Finance Applications: A Survey,” *Applied Sciences*, vol. 9, no. 24, p. 5574, Dec. 2019, doi: https://doi.org/10.3390/app9245574.

ACKNOWLEDGMENT

Acknowledgment is extended to all the esteemed academic staff of Faculty of Computing, General Sir John Kotelawala Defence University whose unwavering dedication and commitment have played a pivotal role in the success of our endeavours.

Convolutional Neural Network-Based Facial Expression Recognition: Enhanced by Data Augmentation and Transfer Learning

HMLS Kumari[1#]
[1]Computer center, Faculty of Engineering, University of Peradeniya, Sri Lanka
[#]lihinisangeetha99@gmail.com

ABSTRACT Facial expression recognition has emerged as a dynamic field within computer vision and human-computer interaction, finding diverse applications such as animation, social robots, personalized banking, and more. Current studies employ transfer learning models in facial expression recognition through the application of convolutional neural networks. The proposed model combines data augmentation with fine-tunned transfer learning models to get a better FER model. A comprehensive collection of training images is crucial as input to effectively train a convolutional neural network (CNN) for accurate facial expression recognition. Hence, the presented research employed data augmentation to enhance the quantity of input images derived from a pre-existing dataset. Manually employing CNN is outdated. Therefore, fine-tuned transfer learning models are used in the proposed study. Activating the final 8 layers of the transfer learning model by freezing the whole transfer learning model is the novel methodology of the proposed model. Then we vary the values of dense layers and dropout layers of the activated 8 layers, which results the fine-tuning of the transfer learning model. The CK+, The facial recognition dataset (human) datasets are used in the proposed model. Subsequently, conduct a stratified 5-fold cross-validation to assess the model's performance on previously unseen data and avoid overfitting the proposed model. The method under consideration utilized transfer learning models, namely DenseNet121, DenseNet201, DenseNet169, and InceptionV3, along with fine-tuned transfer learning models applied to augmented datasets CK+, The facial recognition dataset (human) datasets. The outcomes indicate an achievement of 99.36% accuracy for the CK+ dataset, 95.14% for the facial recognition dataset (Human).

INDEX TERMS Accuracy, CK+, Convolutional Neural Network (CNN), Deep Learning, Data Augmentation, Facial Expression Recognition (FER), Fine-tuning, pre-trained models, Transfer learning model

I. INTRODUCTION

Facial expressions serve as a powerful and universally understood way for humans to communicate their emotions and intentions [1]. A facial expression recognition (FER) system is a computer application designed to independently identify and authenticate the emotions displayed on individuals' faces in digital images or video frames from a video feed. This is achieved by comparing the facial expressions against a database. Facial expression recognition is a significant area of contemporary research with diverse applications, including monitoring patient conditions, improving human-computer interaction, enhancing security measures, influencing game development, strengthening video surveillance capabilities, automating access control systems, animating avatars, contributing to neuro-marketing efforts, and advancing the field of sociable robots.[2]

Facial expression recognition poses challenges in the field of computer vision. because people can vary the expression of their same facial expressions in several situations [3]. As an example, people can show happy expressions differently on different occasions. Even in images of people with the same expression, the brightness, background, and pose can differ, as illustrated in Fig. 1. Therefore, facial expression recognition is a very challenging field in computer vision. As a new approach, we use a convolutional neural network with transfer learning for a small data set and then use data augmentation to make a vast dataset that is appropriate to get exceptional accuracy while fine-tuning values of the presented model.

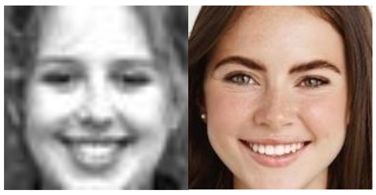


Figure 1. The two different images of a happy expression.

Fig. 1. shows that the first image is from the CK+ dataset, and the second is from The Facial Expression Dataset (Human). Even in an image of a person, the identical expression can be different in terms of brightness, background, and pose. The recognition of facial expression plays an essential role in

nonverbal communication between humans. Hence, there has been extensive research on the generation, perception, and understanding of facial expression. Therefore, the production, perception, and interpretation of facial expressions have been widely studied [4]. The universal facial expressions are happy, sad, angry, disgust, fearful, surprised, and neutral. Facial expression recognition is the main point in human emotion recognition. Darwin initiated this field of study in his book "The Expression of Emotions in Man and Animals" [1].
Recognizing expression is a task that individuals carry out effortlessly in their daily lives [5]. However, in the domain of computer vision, this is a challenging effort. There is some previous research that has different accuracy levels, such as high accuracy and low accuracy. Low accuracy is primarily caused by an uncontrolled environment, and some expressions, such as "sad" and "fear," are very similar, as illustrated in Fig (2).

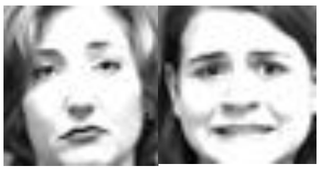


Figure 2. The "sad" and "fear" expressions are very similar

In the same dataset, "sad" and "fear" expressions are very similar, as shown in Fig. 2. As stated above, it is not an easy task in computer vision.

The use of small training datasets in research based on the classification of images leads to poor classification. A tightly constrained model may struggle to capture the details of a small training dataset, leading to underfitting. On the other hand, a loosely constrained model may excessively tailor itself to the training data, causing overfitting and ultimately resulting in inferior performance. Therefore, it is crucial to have a large dataset when training deep learning models with CNN. [7]

The deep multi-layer neural network has proven to be a successful approach in the realm of facial expression recognition. This approach integrates the three stages of facial expression recognition, such as learning, feature selection, and classification, into a single step. New research attempts to enhance the accuracy of neural networks by training them with multiple layers. But this concept results in only small increments of accuracy. While CNNs have demonstrated effectiveness in learning abstract features, especially with deeper architectures involving numerous layers and innovative training techniques [6][7][8].

Building and training a convolutional neural network manually takes time and is out of date. Therefore, one approach is using transfer learning models in convolutional neural networks. The proposed model used transfer learning models such as densenet121, densenet169, densenet201, and Inception V3.

The convolutional neural network employed in facial expression recognition demonstrates superior accuracy when applied to extensive datasets. However, one cannot easily find a dataset with a large number of images. To tackle this problem, we will use data augmentation [2][9]. In this approach, we use commonly used datasets (CK+, the facial expression (Human) dataset). The presented approach aims to attain an accuracy of 99.36% on the CK+ dataset and 95.14% on the facial expression (Human) dataset.

Facial expression recognition has improved a lot in recent decades due to the advancement of recognition methods. Deep learning, especially the improvement of convolutional neural networks, has played a key role in this progress. The effectiveness of these techniques is supported by large training datasets and ongoing improvements in GPU technology. To make small datasets more powerful, we can enlarge them through data augmentation. Because recent researchers used data augmentation in their works to increase accuracy with transfer learning models [2].

The central objective of this study is to formulate a CNN model with data augmentation and transfer learning along with CNN that achieves higher accuracy than previous works for the CK+ and the facial expression (Human) dataset which are small datasets. In the proposed study, we used per-trained models with CNN. Training the whole pre-trained model was not used in this study. Instead, activate some layers of the pre-trained model that are suitable for augmented datasets and freeze other layers. This will result in the most suitable model for each of the above datasets. We can increase accuracy by fine-tuning the values of each variable in the activated layers of the pre-trained model. Recent studies work only with data augmentation and pre-trained models with CNN and don't use freezing layers or activate some layers of the transfer learning model.

We will demonstrate how proposed transfer learning models with fine tuning in CNN outperform recent work on the CK+ and The Facial Expression Dataset (Human) datasets with augmentation.

II. RELATED WORKS

The current facial expression recognition research shows improvement due to the rise of deep learning techniques and especially due to the evolution of convolutional neural networks. The evolution of facial expression recognition depends on reasons such as the availability of huge datasets, the ability to use and add new transfer learning methods for CNN, and the improvement of GPU technology.

Numerous recent methodologies aim to enhance accuracy in facial expression recognition. Aravind Ravi [10] investigated the utilization of features from pre-trained CNNs for facial recognition in a recent study. The findings indicate that repurposing pre-trained models designed for object recognition proves effective in facial expression recognition, with the VGG19 model's layers achieving noteworthy accuracies of 92.26% and 92.86% on the CK+ and JAFFE datasets, respectively [22]. Additionally, earlier network features exhibit high accuracy on smaller datasets, a validation achieved through 10-fold cross-validation, jack-knife validation, and leave-one-out methodologies, addressing the limitations of small datasets.

Simone Porcu, Alessandro Floris, and Luigi Atzori delved into the evaluation of data augmentation techniques for facial expression recognition systems [11]. Their study demonstrates the efficacy of data augmentation techniques in improving accuracy. Specifically, geometric data augmentation and generative adversarial networks contribute to a 30% increase in CNN accuracy using the VGG16 architecture. Employing these methods successfully expands the training dataset initially based on the KDEF dataset and subsequently tests its efficacy on the CK+ and Expw datasets.

Narayana Darapaneni, Rahul Choubey, and Pratik Salvi conducted an investigation into facial expression recognition and recommendations using deep neural networks with transfer learning [12]. The study employed the Jaffe dataset and utilized VGG-16 and InceptionV3 as two transfer learning models[22]. Training configurations, including the last 5 layers, the last 3 layers, the last 1 layer, and all layers, were explored with recognition rates of 95% and 94% achieved through cross-validation.

In another exploration, Tawsin Uddin Ahmed, Sazzad Hossain, Mohammad Shahadat Hossain, Raihan Ul Islam, and Karl Andersson delved into facial expression recognition using a convolutional neural network with data augmentation [13]. This study showcased the effectiveness of data augmentation in enhancing CNN accuracy. Datasets such as CK+, FER 2013, the MUG facial expression database, KDEF and AKDEF, and KinFaceW-I and II were employed, resulting in an overall CNN accuracy of 95.87%.

Andre Teixeira Lopesa, Edilson de Aguiarb, Alberto F. De Souzaa, and Thiago Oliveira-Santosa explored facial expression recognition with convolutional neural networks, specifically addressing the challenges of limited data and training sample order [14]. Small datasets, including CK+, JAFFE, and BU-3DFE, were augmented to create substantial datasets for deep architecture-based facial expression recognition. The proposed method achieved an impressive 96.76% accuracy on the CK+ dataset [22].

The facial expression recognition model has achieved high accuracy by forming an ensemble of modern deep CNNs. Christopher Pramerdorfer and Martin Kampel conducted research and obtained 75.2% accuracy for the FER2013 dataset [15]. They used CNN architectures such as VGG16, Inception, and Resnet. They perform a thorough search to identify the best ensembles of up to 8 models in terms of FER2013 validation accuracy. Real-time facial expression recognition using deep learning research was proposed by Isha Talegaonkar and team [16]. They used the FER2013 dataset and made different changes for the number of epochs, number of layers, and layers of the CNN architecture to produce the model with the highest accuracy. As a result, they achieve a training accuracy of 79.89% and a test accuracy of 60.12% for the FER2013 dataset.

III. METHODOLOGY

A. Dataset

The datasets used in this study are CK+ and the facial recognition dataset (human). The CK+ dataset is the Extended Cohn-Kanade dataset, which contains 123 different subjects and their 593 video sequences [17]. The images in the dataset are of people whose ages range from 18 to 50 and represent a variety of genders and heritages. The CK+ contains 327 labeled images with seven facial expression classes: anger, disgust, contempt, fear, happiness, sadness, and surprise. All images in the CK+ dataset are grayscale images. This dataset is widely used in facial expression classification. The number of images in each class of the dataset CK+ is shown in Table 1. The number of images in classes is different in the CK+ dataset, as shown in Table 1. Therefore, we have considered the unbalance of the ck+ dataset when building the proposed model with the CNN model. Table 1 shows the classes of the ck+ dataset and their number of images.

Table 1. Emotion and number of images in each class of ck+ dataset is represented

Emotion	**Number of images**
Angry (An)	45
Contempt (Co)	18
Disgust (Di)	59
Fear (Fe)	25
Happy (Ha)	69
Sadness (Sa)	28
Surprise (Su)	83

The facial recognition dataset (human) consists of 1823 images and represents 5 facial expressions [18]. The classes of the dataset are: angry_human_face, happy_human_face, neutral_human_face, sad_human_face, and surprised_human_face. The imbalance of classes cannot be seen in this dataset. The images are not grayscale. The whole dataset was divided into ratios of 80%, 10%, and 10% for training, testing, and validation, respectively. Table 2. shows the number of images in each class of facial expression (human) dataset.

The sample images that show different classes of datasets (CK+ and Facial Expression (human)) are shown in Figs. 3 and 4 below, respectively.

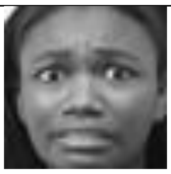
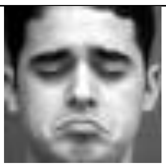
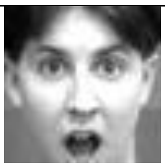
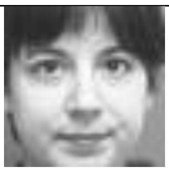
	Angry	Disgust	Fear	Happy	Sad	Suprise	Neutral
CK+							

Figure 3. The sample images that show different classes of CK+ dataset.

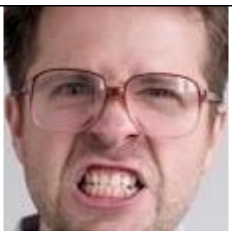
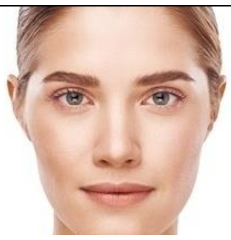
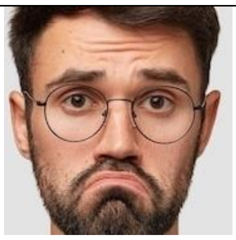
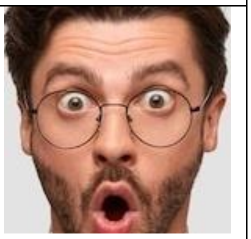
	Angry human face	Happy human face	Neutral human face	Sad human face	Surprised human face
Facial Expression (Human) dataset					

Figure 4. The sample images that show different classes of Facial expression (Human) dataset.

Table 2. Classes of each dataset and their number of images in The Facial Expression (Human) dataset

Expression class	**Facial Expression (Human) dataset**
Angry_human_face	355
Happy_human_face	410
Neutral_human_face	367
Sad_human_face	308
Surprised_human_face	383

B. CNN and Transfer Learning models

A convolutional neural network (CNN) is a specialized type of artificial neural network designed mainly for image recognition within the broader realm of deep learning. CNNs are particularly effective at analyzing pixel data and identifying intricate patterns in images [19]. The process involves taking an image as input, recognizing important learnable weights and biases related to different objects in the image, and allowing the network to distinguish between distinct objects. The CNN architecture consists of four key layers: the convolutional layer, pooling layer, RELU-connection layer, and fully connected layer. This combination makes CNNs well-suited for tasks like facial expression recognition, making them prominent in recent studies exploring the complexities of facial expressions.

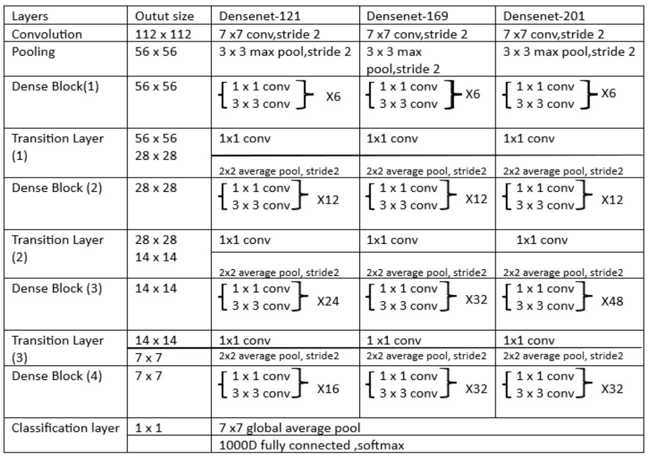
A popular way to recognize facial expressions using CNNs is by using transfer learning models with pre-trained weights from Keras applications. These advanced models are used for tasks like prediction, fine-tuning, and feature extraction in facial expression recognition through CNNs. [26] Many Keras models, such as Densenet121, Densenet169, Densenet201, and Inceptionv3, have been recently used in studies for this purpose.

Fine-tuning stands out as a popular transfer learning technique, particularly for achieving effective facial expression recognition on diverse datasets using pre-trained CNNs. [15] The fine-tuning process involves three key steps:

- Adapt the pre-trained network by eliminating its final layer (the softmax layer) and substituting it with a new softmax layer customized for our particular model.Since pre-trained networks are designed for a larger number of categories, typically 1000 or more, adaptation is necessary for our task of classifying seven facial expressions. Cross-validation is employed to ensure the proper functioning of the adapted network.
- During the training of the pre-trained CNN model with the dataset, a small learning rate is utilized to enhance the model's adaptability.
- Certain layers of the pre-trained network are frozen, and new layers are introduced to align with the characteristics of our dataset.

This study incorporates all these fine-tuning methods, employing various transfer learning models such as Densenet121, Densenet120, Densenet169, and Inception V3.

Densenet is a deep learning network renowned for its efficiency in training, employing concise connections linking every layer.



Layers	Outut size	Densenet-121	Densenet-169	Densenet-201
Convolution	112 x 112	7 x7 conv,stride 2	7 x7 conv,stride 2	7 x7 conv,stride 2
Pooling	56 x 56	3 x 3 max pool,stride 2	3 x 3 max pool,stride 2	3 x 3 max pool,stride 2
Dense Block(1)	56 x 56	[1 x 1 conv, 3 x 3 conv] X6	[1 x 1 conv, 3 x 3 conv] X6	[1 x 1 conv, 3 x 3 conv] X6
Transition Layer (1)	56 x 56	1x1 conv	1x1 conv	1x1 conv
	28 x 28	2x2 average pool, stride2	2x2 average pool, stride2	2x2 average pool, stride2
Dense Block (2)	28 x 28	[1 x 1 conv, 3 x 3 conv] X12	[1 x 1 conv, 3 x 3 conv] X12	[1 x 1 conv, 3 x 3 conv] X12
Transition Layer (2)	28 x 28	1x1 conv	1x1 conv	1x1 conv
	14 x 14	2x2 average pool, stride2	2x2 average pool, stride2	2x2 average pool, stride2
Dense Block (3)	14 x 14	[1 x 1 conv, 3 x 3 conv] X24	[1 x 1 conv, 3 x 3 conv] X32	[1 x 1 conv, 3 x 3 conv] X48
Transition Layer (3)	14 x 14	1x1 conv	1 x1 conv	1x1 conv
	7 x 7	2x2 average pool, stride2	2x2 average pool, stride2	2x2 average pool, stride2
Dense Block (4)	7 x 7	[1 x 1 conv, 3 x 3 conv] X16	[1 x 1 conv, 3 x 3 conv] X32	[1 x 1 conv, 3 x 3 conv] X32
Classification layer	1 x 1	7 x7 global average pool		
		1000D fully connected ,softmax		

Figure 5. Architecture of Densenet121, Densenet169, and Densenet201

In the diagram Fig.5, shows that every Densenet model comprises four dense layer blocks, each with varying numbers of layers. This discrepancy in the number of layers is the key distinction among densenet121, densenet169, and densenet201. [25] [26]

InceptionV3 architecture:

The inception architecture consists of several concepts. Factorized convolution can be seen. This checks network efficiency. The second concept is small convolutions. It replaced a large convolution with small convolutions. It leads to faster training. Next: asymmetric convolutions. It replaces 3x3 convolutions by 1x3 convolutions, followed by 3x1 convolutions. An auxiliary classifier is a small layer insert between layers. This is a small CNN layer. The final concept is grid size reduction, which is done by pooling operations. This will make a model more efficient and avoid computational costs. All these concepts combine into one model and form Inception V3. [27]

C. Data Augmentation

Addressing computer vision tasks, such as facial expression recognition, with a limited training set poses a significant challenge for CNNs. As a result, we must investigate whether there is an increase in accuracy with dataset augmentation when using transfer learning modelsData augmentation is used to make an extensive training dataset suitable for facial expression recognition using CNN. Data augmentation methods were crop, flip, Gaussian blur, contrast normalization, additive Gaussian noise, scale, multiply, translate percent, shear, and rotate. The sample images of the CK+ dataset and the facial expression (human) dataset with the data augmentation showed in Fig. 6, Fig.7.

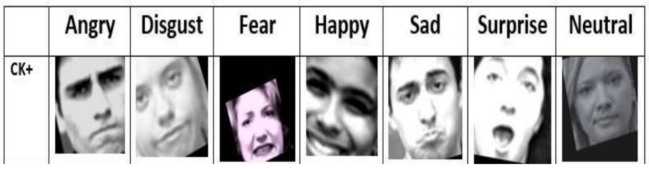


Figure 6. The sample images of CK+ dataset with the data augmentation.

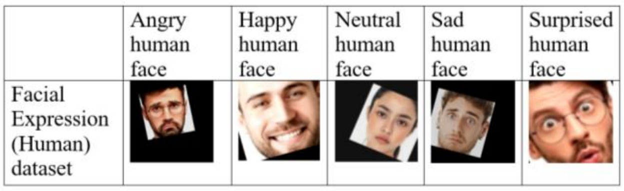


Figure 7. The sample images of Facial expression (Human) dataset with the data augmentation.

D.The Proposed FER System

First, split the dataset as shown in the above diagram. Divide the whole dataset into three parts: the training dataset (70%), the test dataset (20%), and the validation dataset (10%) from the whole dataset. Take the train dataset and apply data augmentation. The data augmentation is done by synthesizing one image into 10 images in the training dataset. In this study, some data-augmentation methods are used. Those are flipping, Gaussian blur, linear contrast, multiplying the number of images, scaling, translating percent, and rotating. This step was done to increase the training dataset and avoid poor classification. Because the model extracts all features and other necessary information using a training dataset to classify test data correctly, Then send those images and their labels separately to the list. Then we have to make a CNN model using a transfer learning model. Transfer learning models are pre-trained for classification tasks using an extensive number of images. Therefore, it is easy to change those transfer learning models to classify similar tasks, such as facial expressions, that are present in the test dataset. The first step of the procedure is to import the transfer learning model.

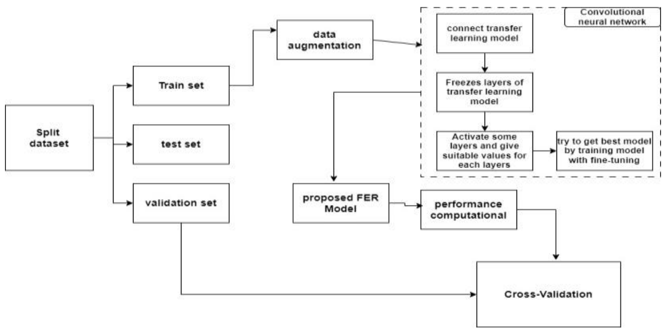

Split dataset
Train set
test set
validation set
data augmentation
Convolutional neural network
connect transfer learning model
Freezes layers of transfer learning model
Activate some layers and give suitable values for each layers
try to get best model by training model with fine-tuning
proposed FER Model
performance computational
Cross-Validation

Figure 8. The diagram shows proposed FER system

The subsequent step involves freezing all layers of the transfer learning model to reduce the risk of overfitting and prevent training the entire network. Unfreezing the final eight layers is then performed to capture detailed information in the images, such as image edges. Following these adjustments, the model demonstrates a good fit and can further be fine-tuned by adjusting the variable values in the layers.

The subsequent stage involves putting the test images, validation images, and label list into the CNN model and executing cross-validation on the test dataset. Cross-validation is used to estimate the new model's behavior for new data (images and data). In this study, it used five stratified cross-validations. The five-stratified cross validation maintains proportions of classes in each fold and prevents overfitting of the new proposed model.

The proposed FER model shows in Fig.(8) results in a new model with high accuracy to classify facial expression, and importantly, the model originates from small datasets.

E. Training

Training involved the utilization of two datasets separately, the CK+ and facial expression (human) datasets, with the incorporation of data augmentation techniques. In this investigation, we introduced eight additional layers by maintaining the immobility of all transfer learning model layers [15]. These augmentations encompassed a GlobalAveragePooling2D layer, two dropout layers, two dense layers, and one batch normalization layer. Hyperparameter tuning was conducted by varying the dropout layer values within the range of 0.4 to 0.7 and experimenting with dense layer configurations, specifically 1024, 512, and 128. The training process spanned 30 epochs, employing diverse batch sizes of 16, 32, and 64. Ultimately, a 5-fold stratified cross-validation methodology was employed across the CK+ and facial expression (human) datasets, integrating multiple transfer learning models to identify the most optimal models.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Implementation Details

In the proposed method, we used densenet121, densenet210, densenet169, and inceptionV3 for images in CK+ and the facial expression (human) dataset. The image size has been set to 224 x 224. In this proposed model, EarlyStopping, ModelCheckPoint, and ReduceLROnPleateau were used. [8]. The model monitors the accuracy.

The proposed model trains for 30 epochs and for batch sizes 16, 32, and 64. We used Google Colab with Python Language and Keras Libraries that run on Tenserflow Basement in this study. The Google colab environment has access to the NVIDIA Tesla K80.

B. Evaluation metrics

Accuracy, Precision and F1 score are the evaluation metrics of this study.

Accuracy = $(TP + TN)/(TP + TF + FP + FN)$

Precision = $TP/(TP + FP)$

Sensitivity = $TP/(TP + FN)$

TP=true positive, TN=true negative, FP=false positive, FN=false negative

Accuracy shows how often a facial expression classification model is correct overall. Precision shows how often a proposed facial expression model is correct when predicting the target class. Recall shows whether the proposed facial expression model can find all images of targeted facial expressions. [23][24]

C. Experimental setup

Cross-validation is used in this proposed FER model. We need to measure how the proposed model behaves in the presence of unseen images. Stratified 5-fold cross-validation was used in this study. This cross-validation type is an extension technique used for classification problems. Mainly, this is because the CK+ and the facial expression (human) datasets are imbalanced. Therefore, we need to keep the same proportion of classes throughout the k-folds as the original dataset.

D. Testing Results

The present investigation assesses the performance using the facial expression (human) and CK+ datasets, which are commonly employed in facial emotion recognition (FER) research due to their compact size. A comparative analysis is conducted with recent studies, demonstrating the contemporary nature of our approach and its commendable accuracy on datasets like CK+ and the facial expression (human) datasets. Our methodology leverages popular transfer learning models, including Densenet121, Densenet201, Densenet169, and Inception V3, which currently dominate the landscape of FER systems. Notably, existing studies often neglect the combined application of data augmentation, transfer learning, and fine-tuning for achieving optimal accuracy. The subsequent results provide a detailed breakdown of the accuracy achieved by our proposed model, separately employing Densenet121, Densenet169, Densenet201, and Inception V3 on the CK+ and the facial expression (human) datasets.

Table 3. Final Maximum accuracies gained by proposed model for ck+ dataset

Model	Batch size	Dense layers	Drop out value	Accuracy	Precision	F1 Score
Densenet 121	32	1024, 128	0.4	0.9937	0.9937	0.9937
Densenet 169	32	1024, 128	0.4	0.9905	0.9910	0.9906

Densenet 201	32	1024, 128	0.5	0.9842	0.9855	0.9843
InceptioV 3	32	1024, 128	0.4	0.9645	0.9675	0.9336

TABLE 4. Final Maximum accuracies gained by proposed model for Facial Expression (Human) dataset

Model	Batch size	Dense Layers	Drop out Value	Accuracy	Precision	F1 Score
Densenet121	32	1024, 512	0.5	0.9274	0.9283	0.9277
Densenet201	32	1024, 512	0.5	0.9514	0.9517	0.9514
Densenet169	32	1024, 128	0.4	0.9139	0.9169	0.9142
Inceptio V3	32	1024, 128	0.4	0.8879	0.8801	0.8840

According to the tables 3,4 , the proposed model attained a peak accuracy of 99.37% for CK+ and 95.14% for the facial expression (human) dataset. The best accuracies of the above models are shown using the graph shows in Fig.9.

Table 5 shows how the proposed model achieves the best results with respect to previous work. Most effective accuracy can be achieved by combining data augmentation and transfer learning models with new layers along CNN.

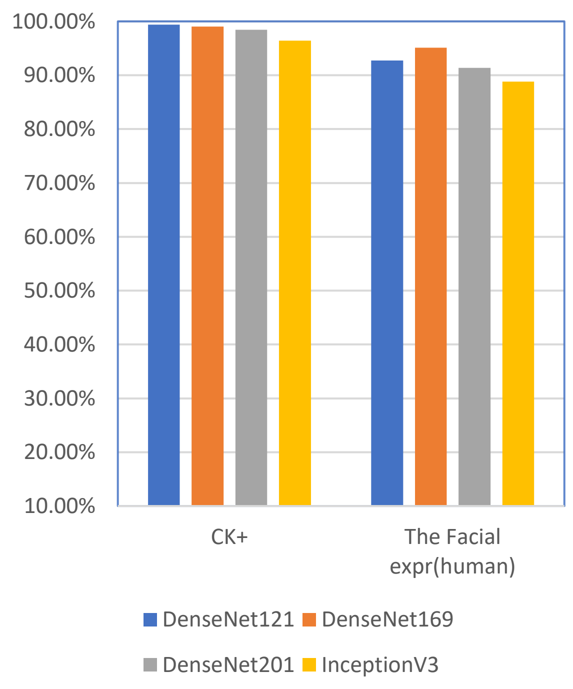


Figure 9. The graph shows maximum accuracy with each model

TABLE 5. comparison of proposed method and previous models' accuracy for CK+, the facial expression (human) datasets.

V. CONCLUSION

Study	Dataset	Accuracy
Aravind Ravi performs a study regarding Pre-Trained CNN Features for FER [10][22]	CK+, JAFFE	92.26% 92.86%
Evaluation of Data Augmentation Techniques for FER Systems by Simone Porcu, Alessandro Floris and Luigi Atzori [11]	CK+ ExpW	83.30%
Narayana Darapaneni, Rahul Choubey, Pratik Salvi did a study on FER and Recommendations Using Deep Neural Network with Transfer Learning [12][22]	JAFFE VGG16 InceptionV3	95% 94%
Facial Expression Recognition using Convolutional Neural Network with Data Augmentation by Tawsin Uddin et al.[13]	CK+, FER 2013, The MUG Facial expression database,etc	95.87%
Facial Expression Recognition with CNN: with coping with few data and the training sample order study done by Andre Teixeira Lopesa et al.[14][22]	CK+, JAFFE BU-3DFE	96.76%
Facial Expression Recognition using CNN: State of the Art by Christopher Pramerdorfer et al.	FER2013	75.2%
Proposed FER model	**CK+, Facial Expression (Human) dataset**	**99.37% 95.14%**

In this research, a contemporary approach to facial expression recognition was introduced, employing a CNN architecture coupled with a transfer learning model and data augmentation. Noteworthy pre-trained models, including DenseNet121, DenseNet201, DenseNet169, and InceptionV3, commonly utilized in image classification, were incorporated. The study demonstrates the enhanced efficiency of classification achieved through the fine-tuning of transfer learning models.

Despite the limitations of small datasets such as CK+ and the facial expression (human) dataset, known for their modest size and limited responsiveness, our methodology leveraged data augmentation to augment the dataset size.

The core idea of transfer learning is simple. Utilize a model trained on a large dataset and apply its knowledge to a smaller dataset. In facial expression recognition with a CNN, we freeze the initial convolutional layers and only fine-tune the last 8 layers responsible for prediction.

The reasoning behind this approach lies in the fact that convolutional layers capture general, fundamental features applicable across diverse images, like edges, patterns, and gradients. Subsequent layers then specialize in recognizing specific features within an image, such as eyes or noses. By applying transfer learning models in conjunction with fine-tuning, the study successfully addressed the challenges posed by small datasets. As evident in the aforementioned results, this approach emerges as the optimal solution for facial expression recognition systems employing convolutional neural networks on small datasets, showcasing the synergistic impact of data augmentation, transfer learning, and fine-tuning.

REFERENCES

[1] "University of Glasgow," The expression oftheemotionsinmanandanimals,https://www.gla.ac.uk/myglasgow/library/files/special/exhibns/month/nov2009.html (accessed Nov. 24, 2023).

[2] Li, S., & Deng, W. (2022). Deep facial expression recognition: A survey. IEEE Transactions on Affective Computing, 13(3), 1195–1215. https://doi.org/10.1109/taffc.2020.2981446

[3] Fathallah, A., Abdi, L., & Douik, A. (2017). Facial expression recognition via deep learning. 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA). https://doi.org/10.1109/aiccsa.2017.124

[4] Jia, S., Wang, S., Hu, C., Webster, P. J., & Li, X. (2021). Detection of genuine and posed facial expressions of emotion: Databases and methods. Frontiers in Psychology, 11. https://doi.org/10.3389/fpsyg.2020.580287

[5] Handbook of Face Recognition.(2011) https://doi.org/10.1007/978-0-85729-932-1

[6] M. A. Akhand, S. Roy, N. Siddique, M. A. Kamal, and T. Shimamura, "Facial emotion recognition using transfer learning in the deep CNN," *Electronics*, vol. 10, no. 9, p. 1036, 2021. doi:10.3390/electronics10091036

[7] T. U. Ahmed, S. Hossain, M. S. Hossain, R. ul Islam, and K. Andersson, "Facial expression recognition using convolutional neural network with data augmentation," 2019 Joint 8th International Conference on Informatics, Electronics &amp;amp; Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision &amp;amp; Pattern Recognition (icIVPR), 2019. doi:10.1109/iciev.2019.8858529

[8] S. Alizadeh and A. Fazel, "Convolutional neural networks for facial expression recognition," [1704.06756] Convolutional Neural Networks for Facial Expression Recognition, http://export.arxiv.org/abs/1704.06756 (accessed Nov. 24, 2023).

[9] S. Porcu, A. Floris, and L. Atzori, "Evaluation of data augmentation techniques for facial expression recognition systems," *Electronics*, vol. 9, no. 11, p. 1892, 2020. doi:10.3390/electronics9111892

[10] A. Ravi, "Pre-trained convolutional neural network features for facial expression recognition," arXiv.org, https://arxiv.org/abs/1812.06387 (accessed Nov. 27, 2023).

[11] S. Porcu, A. Floris, and L. Atzori, "Evaluation of data augmentation techniques for facial expression recognition systems," Electronics, vol. 9, no. 11, p. 1892, 2020. doi:10.3390/electronics9111892.

[12] N. Darapaneni *et al.*, "Facial expression recognition and recommendations using deep neural network with transfer learning," *2020 11th IEEE Annual Ubiquitous Computing, Electronics &amp; Mobile Communication Conference (UEMCON)*, 2020. doi:10.1109/uemcon51285.2020.9298082

[13] Md. Z. Uddin, W. Khaksar, and J. Torresen, "Facial expression recognition using salient features and convolutional neural network," *IEEE Access*, vol. 5, pp. 26146–26161, 2017. doi:10.1109/access.2017.2777003

[14] A. T. Lopes, E. de Aguiar, A. F. De Souza, and T. Oliveira-Santos, "Facial expression recognition with convolutional neural networks: Coping with few data and the training sample order," *Pattern Recognition*, vol. 61, pp. 610–628, 2017. doi:10.1016/j.patcog.2016.07.026

[15] C. Pramerdorfer and M. Kampel, "Facial expression recognition using convolutional neural networks: State of the art," arXiv.org, https://arxiv.org/abs/1612.02903v1 (accessed Nov. 24, 2023).

[16] I. Talegaonkar, K. Joshi, S. Valunj, R. Kohok, and A. Kulkarni, "Real time facial expression recognition using deep learning," *SSRN Electronic Journal*, 2019. doi:10.2139/ssrn.3421486

[17] *Papers with code - CK+ dataset*. CK+ Dataset | Papers With Code. (n.d.). https://paperswithcode.com/dataset/ck

[18] Khan, Z. (2023, November 24). *Facial recognition dataset (human)*. Kaggle. https://www.kaggle.com/datasets/zawarkhan69/human-facial-expression-dataset

[19] Awati, R. (2023, April 24). *What are convolutional neural networks?: Definition from TechTarget*. Enterprise AI. https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network

[20] X. Wang, K. Wang, and S. Lian, "A survey on Face data augmentation for the training of Deep Neural Networks," *Neural Computing and Applications*, vol. 32, no. 19, pp. 15503–15531, 2020. doi:10.1007/s00521-020-04748-3

[21] Stanford University CS231N: Deep Learning for Computer Vision, http://cs231n.stanford.edu/reports/2016/pdfs/023_Report.pdf (accessed Nov. 24, 2023).

[22] Lyons, Michael, Shigeru Akamatsu, Miyuki Kamachi, and Jiro Gyoba. "Coding facial expressions with gabor wavelets." In *Proceedings Third IEEE international conference on automatic face and gesture recognition*, pp. 200-205. IEEE, 1998.

[23] Accuracy vs. precision vs. recall in machine learning: What's the difference? Evidently AI - Open-Source ML Monitoring and Observability. (n.d.). https://www.evidentlyai.com/classification-metrics/accuracyprecisionrecall#:~:text=Accuracy%20shows%20how%20often%20a,objects%20of%20the%20target%20class.

[24] Mage.ai. (n.d.). https://www.mage.ai/blog/definitive-guide-to-accuracy-precision-recall-for-product-developers

[25] A. Ahmed, "Architecture of densenet-121," OpenGenus IQ: Computing Expertise &amp; Legacy, https://iq.opengenus.org/architecture-of-densenet121/ (accessed Nov. 24, 2023).

[26] G. Singhal, "Gaurav Singhal," Pluralsight, https://www.pluralsight.com/guides/introduction-to-densenet-with-tensorflow (accessed Nov. 20, 2023).

[27] V. Kurama, "A guide to resnet, inception V3, and squeezenet," Paperspace Blog, https://blog.paperspace.com/popular-deep-learning-architectures-resnet-inceptionv3-squeezenet/ (accessed Nov. 24, 2023)

[28] Ramalingam, S., & Garzia, F. (2018). Facial expression recognition using transfer learning. 2018 International Carnahan Conference on Security Technology (ICCST). https://doi.org/10.1109/ccst.2018.8585504

[29] Randellini, E., Rigutini, L., & Saccà, C. (2021). Data Augmentation and transfer learning approaches applied to facial expressions recognition. *NLP Techniques and Applications*. https://doi.org/10.5121/csit.2021.111912

[30] Darapaneni, N., Choubey, R., Salvi, P., Pathak, A., Suryavanshi, S., & Paduri, A. R. (2020). Facial expression recognition and recommendations using deep neural network with transfer learning. *2020 11th IEEE Annual Ubiquitous Computing, Electronics &amp; Mobile Communication Conference (UEMCON)*. https://doi.org/10.1109/uemcon51285.2020.9298082

[31] Ahmed, T. U., Hossain, S., Hossain, M. S., ul Islam, R., & Andersson, K. (2019). Facial expression recognition using convolutional neural network with data augmentation. *2019 Joint 8th International Conference on Informatics, Electronics &amp; Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision &amp; Pattern Recognition (icIVPR)*. https://doi.org/10.1109/iciev.2019.8858529

[32] Hrga, I., & Ivasic-Kos, M. (2022). Effect of data augmentation methods on face image classification results. *Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods*. https://doi.org/10.5220/0010883800003122

A Comprehensive Review of Methods Used for Health Prediction and Monitoring Utilizing an Electronic Medical Records (EMR) System

SP Jayasekera#, LP Kalansooriya
Department of Computer Science, Faculty of Computing Sir John Kotelawala Defence University, Sri Lanka
#37-cs-0009@kdu.ac.lk

ABSTRACT: In the rapidly evolving field of healthcare, Artificial Intelligence (AI) and pattern recognition play a key role in enhancing disease diagnosis and prediction. As the patient population increases, the digitalization of medical records has become essential, therefore electronic medical records were developed. This stored Electronic Medical Records (EMR) data can be used to predict possible diseases based on the symptoms stored in the system. This study delves into the integration of AI methodologies within EMR systems, providing a comprehensive review of current techniques that have been used in health prediction and monitoring using EMR data. In this paper, different AI-driven approaches were examined and compared, including Deep Learning (DL), Machine Learning (ML), and Rule-Based Methods. This paper reveals the potential of these techniques in accurately diagnosing diseases, additionally, it discusses challenges and future directions, emphasizing the need for innovative solutions to optimize EMR systems in the context of AI and pattern recognition. Several instances where AI models, such as the application of Support Vector Machine (SVM) models, achieved predictive accuracies of 86.2% and 97.33% in different cancer types, and ML models diagnosing Diabetic Retinopathy with a 92% accuracy rate were observed. Variations in the effectiveness of these technologies across different diseases were also observed, such that a technique that has high accuracy in one disease may have lower accuracy in a different disease. This paper aims to contribute to the growing body of knowledge in AI applications in healthcare, offering insights into the development of more efficient, accurate, and predictive healthcare models.

INDEX TERMS: Healthcare, Deep learning, Electronic Medical Records, Rule-based method, Disease diagnosis, Machine learning.

I. INTRODUCTION

One of the most critical responsibilities of medical institutions is managing patient data, and a patient file is an essential source of data since it enables the development of comprehensive healthcare strategies. It had been common practice for a long time to keep records on paper where medical offices, hospitals, and clinics frequently gathered files and kept patient history using a paper record system. However, paper medical records have a lot of drawbacks such as insufficient storage space, insufficient backups, inconsistency in the layout, and unclear audit trails. Due to technological advancements, electronic medical records were introduced to store patient data on computers or smart devices and overcome paper records' drawbacks.

Electronic Medical Records (EMR) are digitalized versions of paper charts in clinics and hospitals. Clinicians and doctors primarily use these EMRs to diagnose and treat patients and record information by and for the physicians in the hospital. The use of EMRs has become increasingly prevalent in healthcare, with potential benefits such as improved patient care and reduced medical errors [5]. It contains a patient's medical history, diagnoses, prescriptions, treatment schedules, vaccination dates, and lab and test results. These are stored in databases that enable doctors or clinicians to access patient information quickly, track vaccinations, follow patient health performance, and make informed judgments with proper understanding and confidence for the most complex multi-axial diseases, heart diseases, and cancers [4].

By computerizing patient information, there is also a significant change in how patient data are arranged and made available for applications that weren't previously possible with paper records. Thus, it shows that the main objective of an EMR is keeping an eye on the patient while improving healthcare quality. It is important to understand the patient's unique perspective and experiences in the diagnosis and treatment of disease, using EMR has been shown to improve patient outcomes and satisfaction, as well as enhance the physician-patient relationship [2]. Even though EMR provides users and physicians with several advantages, several difficulties are connected to their implementation, such as computer downtime, computer professionals' limitations, a lack of user communication, security risks of confidentiality-leakage, etc which should be considered [6]. An accurate and timely diagnosis is the foundation of any successful treatment. Access to longitudinal data from a patient's EMR might be a valuable clinical resource that could be utilized to forecast future events or diagnoses [1]. A patient's status is thoroughly described in an EMR, and applying data-driven technologies to an EMR enables us to accurately predict and diagnose diseases. This can be made possible by making the raw EMR data into a machine learning representation or turning the data into relevant data that can be processed algorithmically. The integration of AI technologies with EMR systems represents a groundbreaking development in this context. AI's ability to process large datasets and uncover patterns offers unparalleled opportunities for improving disease diagnosis, treatment planning, and patient monitoring.

There are different types of data-driven techniques used to accomplish prediction and diagnosis systems that medical professionals can employ to effectively forecast illnesses and enhance the health of their patients. This review aims to find the most accurate methods for diagnosing and predicting diseases by describing and comparing various methods and techniques used for health prediction and monitoring using EMR.

This study discusses numerous disease diagnosis and prediction methods using electronic records, highlighting their benefits and drawbacks. It also discusses current trends and potential future developments and makes a comparative comparison of the various methods.

The literature review of this paper explores the significance of EMR data in monitoring patient health and advancing data-driven decision-making. It delves into the growing interest in employing computer-assisted methods for disease diagnostics based on Electronic Health Record (HER) data, categorizing these methods into distinct approaches. Machine Learning (ML) methods, encompassing Bayesian, Support Vector Machine (SVM), and decision tree techniques, are discussed, along with the challenges of integrating raw EHR data into ML models due to complexity and limited healthcare data. Bayesian Networks are highlighted for their use in probabilistic medical ontology reasoning, aiding in disease diagnosis and prediction. Decision Trees are emphasized for their effectiveness in the early identification of diseases like Diabetic Retinopathy and asthma. Additionally, rule-based heuristic techniques are explored for diagnosing colorectal cancer and lupus. Finally, Deep Learning methods, including Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Deep Belief Networks (DBN), and Autoencoders (AE). Using these findings, it is aimed to present a comprehensive overview of the existing predicting systems implemented using the above-mentioned techniques and EMR data.

The paper is structured into five sections. Section 2 discusses the current research on methods for disease diagnosis. Section 3 is the Methodology. Section 4 contains the discussion. Finally, Section 5 presents the conclusion of the review.

II. DISEASE DIAGNOSIS USING DATA-DRIVEN MODELS

EMR data is a critical resource in modern healthcare, providing a dynamic method to monitor patient health and improve decision-making using data-driven solutions. Unlike traditional clinical tests and biological investigations, the fundamental goal of EMR data is to track a patient's health over time in a methodical manner. This large set of patient data has paved the way for the creation of prediction models by implementing AI models such as Machine Learning (ML), Deep Learning (DL), and Rule-Based Methods, which have revolutionized disease prediction and diagnosis processes.

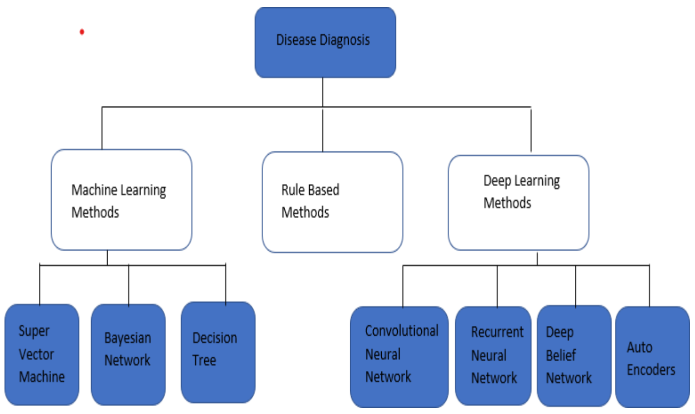


Figure 01. Breakdown of the techniques used in this review to diagnose diseases.

This paper discusses various electronic medical record-based methods for diagnosing diseases automatically. Depending on their technique, models have been grouped into different approaches to diagnosing diseases using EMR data.

A. *Machine Learning (ML) Methods*

Health database systems based on electronic medical records (EMR) are most often created using machine learning methods for individuals who have had health examinations [7]. Machine learning methods can be categorized into different approaches, including Bayesian, SVM, and decision tree methods. Each of these approaches represents a distinct category within the field of machine learning [34].

Many research studies have used EHR data for a predictive model, which involves constructing a statistical model to predict a clinical outcome using machine learning. However, it is difficult to directly integrate raw EHR data into ML models for predictive models due to the complexity of EHR data [8]. Because a lack of data prevents machine learning from solving many healthcare issues.

1) *Support Vector Machine (SVM)*

For supervised classification, experts often use Support Vector Machines (SVM). SVM is based on labeled data, and Vapnik invented SVM [45]. A training dataset is used to find data from the input that has a structure like the output data when both the input and the output have already been supplied.

Getting a cancer diagnosis is crucial for prospective patients since early tumour identification and therapy can improve survival. In [9], a cancer diagnosis was performed using the SVM model using medical information retrieved straight from the EHR. As part of the proposed approach, SVM models for cancer classification were trained using medical records extracted from Electronic Health Records (EHRs). These SVM models Based on the medical data that was analysed, played a crucial part in the cancer categorization procedure. After being trained on 400 pieces of data for each cancer and employing 100 pieces of health information for each cancer, the algorithm has shown a predictive accuracy of 86.2% for ten different forms of cancer and 97.33% for three different types of cancer.

An SVM-based technique was used [10] for significant cohort research to diagnose contralateral breast cancer. Characteristics based on pathology reports for every area of breast cancer and narrative text in progress notes were used to derive features, Zeng et al. [10] designed and put into practice a novel methodology. The suggested strategy for identifying contralateral occurrences in the notes uses medical ideas and how they are combined. SVM and derived characteristics are used to detect contralateral cancers. During the validation process, the area under the curve (AUC) for the model was determined to be 0.93, indicating its high accuracy in predicting outcomes. In the test set, the AUC was slightly lower at 0.89, indicating a slightly reduced but still reliable performance. This strategy of feature development is advantageous due to its simplicity and can be applied to different occurrences of breast cancer as well as to identify various other diseases.

To identify Rheumatoid Arthritis (RA) patients, the Support Vector Machine (SVM) technique can be employed. This technique utilizes a set of naïve and expert-defined Electronic Health Record (EHR) characteristics for the identification process [12]. This method uses Natural language processing (NLP) concepts, pharmaceutical exposures, and billing codes. The SVM methodology was trained using both expert-defined and naive data. The accuracy and recall scores were 0.94 and 0.87, respectively, as opposed to 0.75 and 0.51 for deterministic approaches. In this study, a dataset of 10,000 patients was employed. The test findings divided the patients into three groups: potential RA, definite RA, and not RA.

2) *Bayesian Network (BN)*

A probabilistic graphical framework called a Bayesian network is utilized to represent a group of variables and their conditional interactions. This graphical model employs a directed acyclic graph (DAG) to illustrate the relationships among the variables and their dependencies. Naive Bayes (NB) and Bayesian Networks (BN) are both probabilistic algorithms that perform effectively with various characteristics [14].

Building Clinical Bayesian Networks (CBN) for probabilistic medical ontologies reasoning is described in [13] to directly learn the entire ontology and high-quality Bayesian topology from EMRs. More than 10,000 patient records analysed for medical entity connections have used the K2 greedy method and Odds Ratio (OR value) computation to create a Bayesian topology automatically. The study demonstrates that medical information can generate high-quality health topology and ontology directly and automatically. A clinical Bayesian network has been developed using the study's probability distribution between illness and other parameters. With 1712 test samples, an accuracy of 64.83% was produced by the Naïve Bayesian network, while the Basic Bayesian network produced 68.45%.

In a study by Sakai et al. [15], they evaluated the diagnostic performance of a Bayesian network in comparison to the NB model, an artificial neural network (ANN), and a logistic regression model to identify instances of appendicitis. 169 people who were thought to have acute appendicitis were included in the dataset for the study. The performance of the proposed model was assessed using logistic regression and neural network metrics. Compared to other diagnostic models examined in this research, this model had the lowest error rate and produced the most trustworthy findings, detecting that 50.9% of patients (86 out of 169) had appendicitis.

The Naïve Bayes method was employed in Al-Aidaroos et al. [16] review of medical data mining to classify medical data and diagnoses such as primary tumours, hepatic issues, and breast or lung cancer. Using 15 datasets, the proposed NB strategy was empirically compared with five other approaches to show its superiority. The findings indicated that NB performed better than others regarding medical categorization. Deep learning ideas can produce superior segmentation results with the proposed approach. The report states that future research will combine NB and different methodologies.

Kazmierska and Malicki researched the Bayesian classifier, which is used to assess whether cancer is progressing or relapsing [17]. This study analysed data from 142 individuals who had radiation therapy for brain tumours between 2000 and 2005. For training, 96 binary attributes were selected. As a result of the proposed model, the likelihood of having a cancer relapse has been determined as well as the likelihood of not having one. The proposed method received scores of 0.84, 0.87, and 0.80 for accuracy, specificity, and sensitivity, respectively.

3) *Decision Tree*

EHR data can accelerate and simplify the early identification of Diabetic Retinopathy (DR). Five machine-learning techniques are used in [18] to identify diabetic retinopathy using electronic health record data. Records from 301 Chinese hospitals were compiled into a sizable retinal dataset. To increase the accuracy of DR illness diagnosis, preprocessing techniques such as label binarization, value normalization, and standard acceleration are carried out. According to the experimental findings, the machine learning model's Random Forest (RF) can achieve an accuracy level of 92% while performing well. Due to its low cost, low threshold, and excellent diagnosis accuracy, the suggested approach has an advantage over current DR diagnostic methods.

The primary objective of the study conducted by Lungu et al. [19] was to investigate whether machine learning techniques could enhance the diagnostic precision of Magnetic Resonance Imaging (MRI) in detecting pulmonary hypertension (PH). This was accomplished by employing computational modelling approaches and image-based metrics. MRI as well as the Right Heart Catheterization (RHC) were used to identify PH using a decision tree method [19]. Seventy-two individuals with potential PH underwent MRI and RHC, and 57 of these patients were found to have the condition, while 15 samples were determined to be PH-free. As a result of the proposed algorithm, 92% of the PH cases were correctly identified, while 4% were misclassified. If the findings of this study are as anticipated, RHC may not be required when PH is suspected.

In [20], the decision tree is used in the first phase to diagnose asthma, and the fuzzy system is utilized in the second phase to assess the level of asthma management. Dry cough, sore throat, sneezing, and other symptoms have been used to diagnose asthma, whereas breathlessness and other daytime symptoms have been used to measure the control level. In this study, the information was gathered through the patients' responses to questionnaires. Diagnoses of asthmatic patients were made using a decision tree classifier, which had accuracy and kappa coefficients of 0.90 and 0.783, respectively.

B. *Rule-Based Method*

In [22], the diagnosis of colorectal cancer was made using a rule-based heuristic technique. Machine learning and rule-based methods' effectiveness was evaluated for each phase. The algorithm identified concepts at the document level with an F-measure of 0.996 as well as detected cases at the patient level with 0.93 for the F-measure using the manually examined data set of 300 potential Colorectal cancer patients. In the work by Breischneider et al. [23], in this study, rule-based grammar was used to obtain textual information from records of patients with mamma carcinoma. Based on recovered textual fragments, seven essential criteria were listed to construct the therapeutic suggestion. The mammography use case was used to assess the proposed system. With an accuracy of 0.69, a textual feature extraction approach based on rule-based decision support, information extraction, and semantic modelling was employed to determine the lymph node status.

In an EHR dataset with 400 records, Jorge et al. [24] used rule-based approaches to identify lupus patients. Natural language processing was used to extract the narrative and codified data from the training set of data (NLP). Based on penalized logistic regression, the author classified systemic lupus erythematosus (SLE) as either definite or probable. The machine learning code utilized in this work for definite SLE showed a 90% positive predictive value, with a specificity of 97%. According to the best rule-based method (ICD-9 code), the specificity and sensitivity were respectively 86% and 84 % and 60 % and 69 % for definite and definite/probable SLE.

C. *Deep Learning Methods*

Deep neural networks, including autoencoders (AE), Convolutional Neural Networks (CNN), Deep Belief Networks (DBN), Recurrent Neural Networks (RNN), and other similar architectures, are considered the most effective machine learning techniques in the biomedical sector [25]. These networks form the foundation of deep learning and have shown remarkable effectiveness in various biomedical applications. Various deep learning methods used on electronic medical records are examined in this review to apply them to clinical tasks. Their benefits are discussed in practice and potential future applications.

1) *Convolutional Neural Network (CNN)*

A method for unsupervised deep feature learning that Miotto et al. introduced in [21]. Using clinical notes as the input, they drove patient representation in their predictive modelling technique. By identifying hierarchical regularities and relationships in clinical notes, 700,000 individuals from the Mount Sinai dataset were used. The study encompassed a broad range of clinical areas and chronological periods, involving a total of 76,214 test individuals, representing 78 distinct diseases. The study's findings surpassed approaches that relied on a representation derived from basic medical information, where accurate and F-score forecasts improved by 92.9% and 18.1%, respectively. When produced patient representations are included in DL approaches, clinical prediction can be improved. This study can use the laboratory findings to improve the quality of its model.

Multiple illnesses have been evaluated using the disease prediction model built on EMRs [26]. The Convolutional Neural Network (CNN) has been used to characterize the suggested strategy for multiple illness prediction. This approach was tested on 4298 patients with a brain infection, coronary heart disease, and pulmonary infection. In a dataset for cerebral infections, the CNN algorithm, the accuracy was 96.5% and the F1-measure score was 96.6%.

2) *Recurrent Neural Network (RNN)*

Recurrent Neural Networks (RNN) were specifically designed to process sequential inputs, such as language data. The present state of an RNN implicitly incorporates knowledge about the whole history of the series since RNNs process, a series of inputs that transmits the concealed value of every input unit to the next input unit, one item at a time. Doctor AI [27], which was over eight years, performed over 260K time-stamped analyses on individuals' electronic health records longitudinally, which is one RNN-inspiring technique. Doctor AI surpassed multiple baselines and scored 79.58% on a sizable real-world EHR dataset.

Wu et al. [28] presented a novel approach for categorizing paediatric asthma by utilizing event sequences and their corresponding characteristics. The findings of this study show that including a timestamp in an RNN model enhances the categorization of individuals without asthma rather than those who have it.

3) Deep Belief Network (DBN)

A Deep Belief Network (DBN) has been used to diagnose Parkinson's disease (PD) using speech sounds collected from the UCI repository [30]. A range of healthy and sick voices was used to train the suggested approach, using DBN as a data source, and the features were extracted. According to the proposed method, the PD consists of one output layer and two stacked limited Boltzmann machines. Parkinson's disease was diagnosed with 94% accuracy using the recommended approach.

DBN has been used [31] to diagnose attention deficit hyperactivity disorder (ADHD), which is one of the most common diseases. The network was built and trained using a greedy methodology according to the recommended strategy. The Global Competitions ADHD-200 has provided the two training and testing datasets. This study has used samples from the Neuroimaging (NI) and New York University (NYU) databases for training and testing, respectively. These findings show that they attain cutting-edge accuracy of 0.6368 on the NYU dataset and 0.6983 on the NI dataset.

4) Auto Encoders (AE)

In a study, researchers employed auto-encoders to forecast a particular group of diagnoses [29]. For the detection and classification of softmax, stacked autoencoder, and cervical cancer classification algorithms have been utilized [32]. To train and test the approach, the UCI dataset with 30 characteristics, four targets, and 668 samples was used. A training set made up 70% of the dataset, while a test set made up 30%. Four target variables were applied to the suggested model, and the efficacy of its categorization was evaluated. This comparison produced a 0.978 accurate classification rate. Due to the dimensionality reduction of the samples, this model's training takes far too much time. In the future, advanced methods could be used to reduce the training time of the model. Hwang et al. [33] examined the efficacy of missing value prediction, conventional networks, and generative adversarial networks (GANs) methods combined for illness prediction [33]. With a specificity of 0.99, along with a sensitivity of 0.95, also with an accuracy of 0.98, the stacked autoencoder (missing value forecasting technique) and auxiliary classifier GANs (AC-GANs: illness prediction) have shown excellent results. In this work, AE fills in the gaps left by the GAN generic model. The use of GAN to fill in the missing data is one of this work's future directions.

III. METHODOLOGY

An efficient and effective way to obtain requirements is through document analysis, which involves reviewing current system documentation and acquiring data. In the study, a systematic analysis of 40 papers was conducted, and 31 of them were selected based on stringent criteria to review in this paper. Research articles were used from Google Scholar and other research archives articles on EMR-based disease diagnosis based on AI methods such as ML, Rule-based, and DL methods. The selection process involved multiple stages, including title and abstract screening, full-text reviews, and a backward and forward search to capture additional relevant works. Selected literature used specific keywords and phrases, and combinations of these terms. Keywords related to the topic were searched to find existing research articles.

Several research articles on EMR systems and methods used for predicting systems were reviewed and analyzed. The research article categorizes the methods used to track and predict health into three categories: Different approaches were employed in the study, including the utilization of Rule-Based Methods, Machine Learning (ML) Methods, and Deep Learning (DL) Methods. These categories were chosen based on the predominant analytical techniques used in the articles and provided a structured way to compare the effectiveness and accuracy of different methods.

To simplify data analysis, the literature review was summarized into tables. Tables provide an overview of methods used, the disease addressed in the paper, performance measures such as accuracy and F-measures (a measure of test accuracy), and the paper's objectives. Comparing the effectiveness and accuracy of different methods can be done using the tables.

Support Vector Machines (SVMs) are strong technology that includes both nonlinear and linear regression approaches, making them essential to data mining processes. SVMs can conduct multiclass and binary classification, making them useful for data prediction and classification, including in the field of health research. SVMs are frequently used by researchers for supervised classification, particularly in disease detection and prediction.

For instance, SVM methods have been used to identify different types of diseases, such as breast cancer and rheumatoid arthritis, with high accuracy. Zhang et al. [9] achieved 97.33% accuracy in cancer classification from Electronic Health Records (EHRs) using SVM, while Zeng et al. [10] conducted a validation study on detecting contralateral breast cancer, achieving a high area under the ROC curve (AUC) of 93% when utilizing extracted features in combination with pathology reports. Additionally, SVMs have been employed in the identification of rheumatoid arthritis (RA) phenotypes, achieving an F-Measure score of 88.6% in a Naïve EHR with a sample size of 376 patients [12].

Table 6. The Summary of the SVM Methods

Methods	**Focused**	**Performance**	**Dataset**	**Objective**

	disease	**Measures**		
SVM- RBF [9]	Cancer	Accuracy- 97.33%	Employed 100 pieces of health information for each cancer and trained on 400 pieces of data for each cancer.	Using SVMs to classify cancer from EHRs.
SVM [10]	Breast Cancer	Testing- 89% AUC Validation- 93%	A total of 1063 women with breast cancer.	Analyzing pathology reports and extracted features to identify contralateral 1 breast cancer.
SVM [12]	Rheumatoid Arthritis	Precision- 96.8% F-Measure- 88.6% Recall- 87% AUC-96.6%	In total, 376 patients (185 with RA and 191, not RA).	SVM-based phenotyping of RA in the Naïve EHR.

The below graph shows the average performance based on the performance measure obtained for the different diseases based on the SVM methods used.

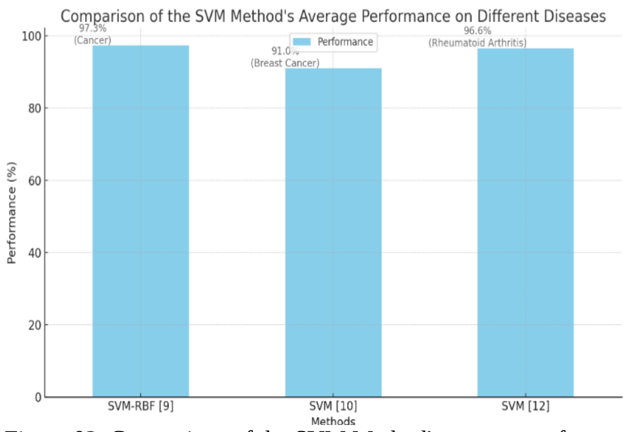


Figure 02. Comparison of the SVM Method's average performance on different diseases

Bayesian approaches, such as Naive Bayes (NB) and Bayesian Networks (BN), are probabilistic algorithms that use multiple features elegantly. By utilizing proven biomarker operating characteristics, Bayesian clustering can accommodate patients with varied data availability. The advantage of Bayesian joint modeling is that it incorporates phenotypic uncertainty into future association analyses, producing correct uncertainty estimates. When compared to Bayesian networks, NB classifiers do not require dependency networks and are better at handling high-dimensional features. This research looks at four articles that utilize Bayesian approaches to predict diseases like cancer, appendicitis, hepatitis, and brain tumors.

Shen et al. [13] tested the accuracy of Naive Bayes and Bayesian Networks in predicting cancer and achieved 64.83% and 64.83% accuracy, respectively. Sakai et al. [15] used the Bayesian network to predict the diagnosis of acute appendicitis. Aidaroos et al. [16] classified cancer, hepatitis, and

liver disorders using NB with an accuracy of 97.43%. Bayesian networks were also used to optimize treatment decisions for a brain tumor with 84% accuracy. The table below lists a few Bayesian method-based systems, and research articles are used to note how accurate the results were when used to predict diseases.

Bayesian statistical models have been utilized when there are gaps in the information provided by local data but there are additional sources of information that can help close the gaps. There are further advantages to Bayesian modeling since it gives a reasonable framework for incorporating new data as it becomes available and helps practitioners to rapidly estimate future illness scenarios. The table below lists a few Bayesian method-based systems, and research articles are used to note how accurate the results were when using Bayesian method-based to predict diseases.

Table 7. The Summary of the Bayesian Methods

Methods	**Focused Disease(s)**	**Performance Measures**	**Dataset**	**Objective**
NB, BN [13]	cancer	NB Accuracy- 64.83% BN Accuracy- 68.45%	Records of 10,000 identified patients.	An Automatic Bayesian topology generation using the K2 greedy method and odds ratios (OR values).
Bayesian Network [15]	Appendicitis	-	A database contains 169 people who may have acute appendicitis.	An algorithm for predicting acute appendicitis using Bayesian networks.
NB, LR, DT, and NN [16]	Multiple diseases, including cancer, hepatitis, and liver disorders	Accuracy- 97.43% AUC- 99%	Various illnesses are illustrated in 15 datasets from the UCI library.	LR, NB, NN, and DT classification of medical data.
Bayesian Network [17]	Brain Tumor	The accuracy rate is 84% The sensitivity is 80% The specificity is 87%	142 patients with brain tumors.	Optimization of treatment decisions using the Naïve Bayesian Classifier.

The graph illustrates the average performance of different diseases based on the use of Bayesian methods for diagnosis.

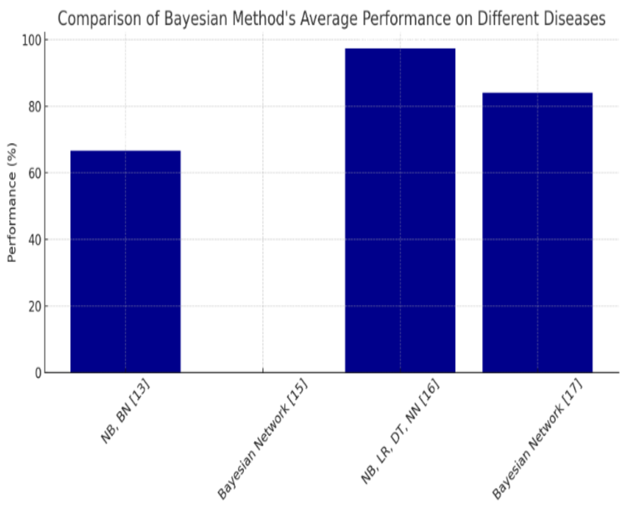


Figure 03. Comparison of the Bayesian Method's average performance on different diseases

Using decision trees was another method used for predicting diseases in research articles. A decision tree aids in creating a fair picture of the rewards and hazards related to each potential result. When contemplating EHRs, where uncertainty is prevalent, decision trees are highly helpful because they are especially beneficial when the results are unknown. A decision tree is an effective tool for decision-making. It offers a useful framework within which to consider options and investigate what might result from each.

Decision trees are used to categorize records, which are useful for challenges involving association and regression. By using a decision tree, advantages and disadvantages can be quickly visualized and identified. The diagnosis system for diabetic retinopathy developed by Sun and Zhang [18] achieved 86.82% accuracy. Based on MRI images, Lung et al. [19] were able to diagnose pulmonary hypertension with 92% accuracy using a decision tree. Using a decision tree and fuzzy system, asthma diagnosis and control levels were
determined [20].

The reliability and effectiveness of decision trees in medical decision-making are supported by reputable sources, including research articles and academic publications. Decision trees provide high classification accuracy and are a dependable and effective means of making judgments due to their plain representation of the information gathered. They have been widely used in a variety of medical decision-making scenarios, including classification and diagnosis. The fundamental properties of decision trees and their effective applications in medicine have been emphasized in the literature, highlighting their potential for future use in medical research and practice.

Table 8. The Summary of the Decision Tree Methods

Methods	**Focused Disease(s)**	**Performance Measures**	**Dataset**	**Objective**
DecisionTree [18]	Diabetic Retinopathy	Accuracy- 86.6%	301 Chinese hospitals provided 5057 records.	Five machine learning techniques are used with the EHR to diagnose DR.
DecisionTree [19]	Pulmonary hypertension	Sensitivity is 97% Accuracy of 92% Specificity- 73%	Pulmonary hypertension is suspected in 72 patients.	Analyzing MRI images to diagnose pulmonary hypertension.
Decision Tree and Fuzzy system [20]	Asthma	Kappa- 78.32% Accuracy- 90%	30 of patients with asthma.	Using fuzzy logic and decision trees to diagnose and control asthma.

The graph below presents the average performance of various diseases using Decision tree methods for diagnosis.

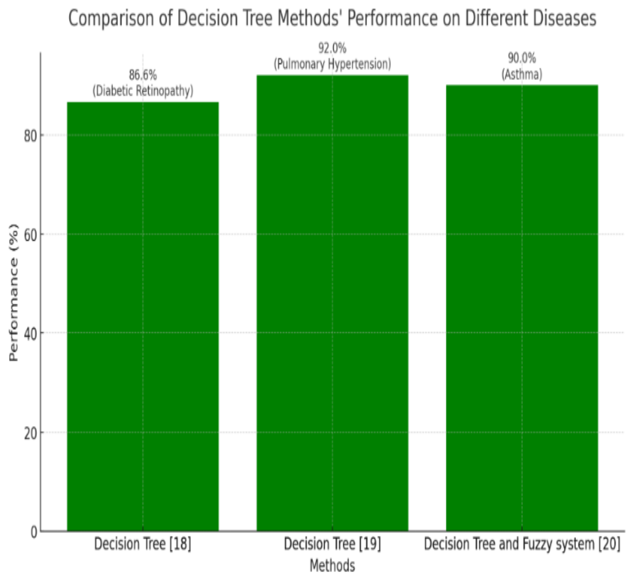


Figure 04.Comparison of the Decision Tree Method's average performance on different diseases

By using rule-based systems, we can retrieve features from electronic medical records quickly. For the extraction of data, rule-based systems are used since the most common kind of knowledge representation is if-then logic.

Using rule-based systems, domain experts can express and rate their expertise. The decision-making process can then use that data. To determine the outcomes of rule-based or identically based systems, users must input specific attributes or facts, such as patient symptoms. It is difficult for someone without medical training to do this. A drawback of this method is the

requirement for precise definitions of data properties. Using the rule-based method, computer scientists identify rules and identify patterns associated with them. Xu et al. [22] used this method to identify colorectal cancer. Breischneider et al. [23] used automated breast cancer detection using rule-based grammar and achieved 90% accuracy. Using a rule-based algorithm and machine learning codified algorithm, Jorge et al. [24] identified Lupus patients from EMR.

Table 9. The Summary of the Rule-based Methods

Methods	Focused Disease(s)	Performance Measures	Dataset	Objective
Rule-based ML-based [22]	Colorectal cancer	Accuracy- 99.6% Precision- 99.6% Specificity- 96.9% F-measure- 99.6%	1,262,671 patients from a synthetic derivative database.	Data extraction and integration from EHRs for Colorectal cancer detection
Rule-based grammar approach [23]	Breast cancer	Accuracy of 90% The Specificity of 59% Sensitivity of 98%	The university hospital in Erlangen collected the clinical reports of 2096 patients totaling 8766.	Clinical report information extraction for breast cancer.
The rule-based algorithm is, Machine learning codified algorithm. [24]	Definite SLE Definite probable SLE	Sensitivity- 86% Specificity- 60% PPV- 46% Sensitivity- 84% Specificity- 69% PPV 65%	400 records in an EHR dataset.	From the EHR, recognize patients with Lupus.

The graph illustrates the performance of rule-based methods applied to the diagnosis of various diseases.

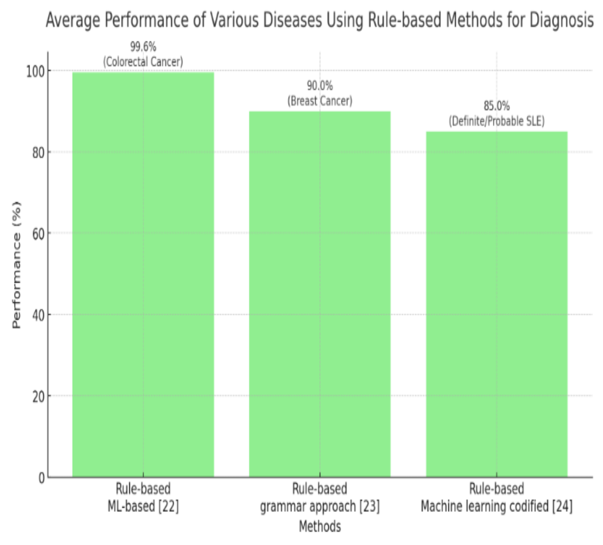


Figure 05. Comparison of the Decision Tree Method's average performance on different diseases

Deep Learning (DL), often referred to as hierarchical learning, is a sophisticated modeling approach characterized by its use of multiple processing layers to analyze complex data sets. This method is increasingly employed in the analysis of the ever-expanding volumes of EHRs. The application of deep learning in the realm of EHRs is particularly notable in research endeavors focused on forecasting individual health outcomes and assessing potential risks. At the heart of deep learning technology are various types of neural networks, each with unique capabilities and applications. These include convolutional neural networks (CNNs), known for their prowess in processing visual imagery; recurrent neural networks (RNNs), which excel in handling sequential data; deep belief networks (DBNs), which are effective in probability-based learning; and autoencoders, specialized in data encoding and reconstruction tasks. These diverse neural network architectures enable deep learning to effectively interpret and utilize the vast and complex data present in EHRs for advanced medical research and analysis.

Table 5. The Summary of the Deep Learning Me

Methods	**Focused Disease(s)**	**Performance Measures**	**Dataset**	**Objectives**
Unsupervised deep feature learning [21]	78 diseases	Accuracy- 92.9% F-score- 18.1%	The Data warehouse from Mount Sinai contains 700,00 patients.	Predictive models can be developed using patient representations from EHRs.
CNN and Framingham risk score [26]	Cerebral infraction (CI), Pulmonary Infarction (PI), And Coronary Heart (CH)	Accuracy CI-96.5% PI- 95.6% CH- 93.6%	From a Chinese hospital with a grade-A rating, 4298 individuals were evaluated.	Clinical notes based on a uniform model for assessing multiple diseases.
RNN [27]	Numerous diseases	Recall- 79.58%	260K patients.	Applied to longitudinally timestamped EHRs.
RNN [28]	Pediatric Asthma	Precision- 84.54% F-measure- 85.08% Recall- 85.65%	4000 patients from Physionet and 4013 patients from Olmsted Country Birth Cohort.	RNN-based asthma classification in pediatrics.
DBN [30]	Parkinson's Disease	Accuracy- 94%	Data set on 31 Parkinson's patients.	DBN-based Parkinson's disease diagnosis system.
DBN with greedy Approach [31]	ADHD	NI- 69.83% Accuracy- NYU- 63.68%	Neuroimaging-samples of 73 New York University- samples of 263.	A greedy approach to the diagnosis of ADHD using DBN.
Stacked AE and Softmax classification. [32]	Cervical cancer	Accuracy- 97.25%	668 samples from the UCI dataset.	Stack autoencoder and softmax classification for cervical cancer classification and diagnosis.
Stacked AE and GAN [33]	Breast cancer	Sensitivity- 95.28% Accuracy- 98.05% Specificity- 99.47%	Breast cancer records are available for 569 cases, of which 212 are malignant and 357 are benign.	Generative Adversarial Networks (GAN) and stacked autoencoders for disease prediction from EHRs.

IV. DISCUSSION

This paper provides a comprehensive review of current techniques that have been used in health prediction and monitoring using EMR data, with a focus on the integration of AI methodologies within EMR systems. The study highlights the potential of AI-driven approaches, including Deep Learning (DL), Machine Learning (ML), and Rule-Based Methods, in accurately diagnosing diseases. The paper discusses several instances where AI models have achieved predictive accuracies using the models in existing systems. According to the literature review of this paper, a few of the strengths and weaknesses were identified in each of these AI-driven approaches.

Because of their adaptability and capacity for probabilistic reasoning, machine learning techniques like support vector machines (SVM), Bayesian methods, and decision trees have been useful in the diagnosis of conditions like cancer, arthritis, and pulmonary hypertension. These techniques have also shown high predictive accuracies in a number of different disease types. SVMs are a powerful technology that are essential to data mining procedures since they support both linear and nonlinear regression techniques. SVMs are helpful for data prediction and classification, especially in the area of health research, because they can do binary and multiclass classification. SVMs do have significant drawbacks, too, such as the computationally demanding nature of model training and optimization. In contrast to more straightforward models like decision trees, they are also harder to interpret, which can be problematic in medical contexts when clarification is crucial. The Bayesian Network is another type of machine learning technology that has pros and limitations. Bayesian networks are helpful in controlling uncertainty and probabilistic reasoning in the setting of medical diagnostics, where ambiguity is common. They improve model predictions by combining prior information and experience. However, high-dimensional data is an issue for BNs, and as the number of variables increases, so does the model's complexity, making computation and interpretation challenging. Decision trees are machine learning techniques that give unambiguous decision paths and are highly interpretable. As a result, they can be used to justify diagnostic judgments in the healthcare industry. They perform effectively with numerical and categorical data, can adapt to various types of medical data, and can detect diseases early. However, decision trees are prone to overfitting, especially with complex or noisy data, and are limited in handling non-linear relationships compared to more sophisticated models like SVMs or deep learning techniques.

Rule-based approaches, which are noted for their speedy feature retrieval and simple knowledge representation, have exhibited excellent accuracy in specific disease diagnoses, such as colorectal cancer with an accuracy of 99.6% [22] and breast cancer with an accuracy of 90% [23]. However, they have some disadvantages, such as reliance on precise definitions, restricted flexibility and scalability, poorer accuracy in some circumstances, and difficulty understanding and implementing for non-experts. While rule-based systems have demonstrated excellent accuracy in certain areas and are praised for their simple logic, their rigidity and the requirement for precise data definitions can be significant limits, particularly in the dynamic and complicated field of healthcare.

Deep learning (DL) has demonstrated remarkable capabilities in biological applications. Because of its various processing levels, it is extremely successful at processing complex data, such as electronic medical records (EMR). DL approaches have shown great accuracy and sensitivity in a variety of medical activities, such as breast cancer detection, with an accuracy of 98.05% [33], although DL has certain limitations. Large datasets are often required for training, which can be a drawback in cases when data is sparse. Furthermore, training and implementing DL models can be computationally demanding, necessitating significant processing power and resources. Furthermore, DL models, particularly sophisticated structures, can lack interpretability, making it difficult to grasp the reasoning behind diagnoses or treatment decisions, which is critical in healthcare.

This review discusses different techniques for predicting cancer diseases, including SVM, Bayesian networks, rule-based methods, and stacked AE. Using SVM, Zhang et al. [9] classified cancer and achieved an accuracy of 97.33 %, while Zeng et al. [10] identified breast cancer with an accuracy of 93%. Using a Bayesian network, a similar cancer disease could be identified with 64.83% accuracy, while the same disease could be identified using Naive Bayes with 64.83% accuracy. Rule-based grammar was used to detect colorectal cancer [22], which earned an accuracy of 99.6%. Breast cancer was also detected using rule-based grammar [23], which achieved an accuracy of 90%. By combining AE and Softmax, Adem et al. of [32] classified cervical cancer with 97.25 percent accuracy. In this study, stacked AE and GAN [33] were used to predict breast cancer, and the accuracy rate was 98.05%.

To predict Asthma, different methods have been used. The Decision Tree and Fuzzy system [20] were used to diagnose and control asthma levels, and this system showed an accuracy of 90%. Wu et al. [28] used the RNN method to create a pediatric asthma prediction system with an accuracy f- a measure of 85.08%.

Even though there are many predictive models available, most of them are designed to predict single diseases without considering the many factors that can affect patients, for example, a cancer prediction system will only consider the symptoms of a patient to predict cancer and will not suggest other diseases based on these symptoms. However, several models have been developed to help identify multiple diseases, and this review discusses these systems. Al-Aidaroo et al. [16] classified and detected multiple diseases, involving hepatitis, cancer, and liver disorders, with an accuracy of 97.43%. With 86% and 84% sensitivity, [24] based on a rule-based algorithm, definite and probable Systemic lupus erythematosus (SLE) were detected. Shi et al [26] is another researcher focused on multiple diseases Cerebral infarction (CI), Pulmonary Infarction (PI), and Coronary Heart (CH) detected in this system, accuracy reached for each disease was CI 96.5%, PI 95.6%, CH 93.6%. Another system that is used to detect multiple diseases [21] is used to derive 78 diseases for this dataset taken from the Mount Sinai data warehouse of 7000 patients, this system received a 92.9% accuracy.

Data from the literature study indicates that certain approaches are more effective than others. While certain techniques may be more accurate for some illnesses but less accurate for others.

V. CONCLUSION

According to this review, several EMR system studies have been conducted recently to learn new facts about healthcare using technology. Using various procedures, EMRs provide a lasting record of patient care, reducing vulnerabilities and solving problems in modernized healthcare records.

Physicians can provide better care to patients when they have access to accurate and timely information. EMRs assist physicians in providing safer care, reducing medical errors, and improving the diagnosis of diseases. A competent EHR not only keeps track of patient allergies and medications but also checks for concerns when new medications are administered. An EMR can identify patterns of potentially related adverse outcomes and alert at-risk patients quickly. With the advancement of IT, EMR systems are now widely used to manage medical data and prescribe medication.

Different EMR systems using different techniques are installed and used in various healthcare facilities and these EMR systems have proven essential to delivering better patient care. In this review, it is classified into three primary categories machine learning, rule-based approach, and deep learning method which are then further subdivided depending on the suggested algorithm and have attempted to cover the most recent and current studies on autonomous diagnosis from electronic data. As discussed throughout the review, some methods can give accurate results in one type of disease, but not in another, and most systems are designed to predict and diagnose one specific disease, but very few systems have been able to detect multiple diseases simultaneously. According to the literature study, certain approaches were more effective than others.

Although EMR systems have their benefits, there are still some drawbacks, such as the need to update patient records after every appointment or consultation. Otherwise, physicians or clinical supervisors may later check the system and find incorrect information resulting in an inappropriate treatment plan. It is also possible that records may not be updated or inaccessible for an extended period if there is a power outage, location problems, or another issue. Another disadvantage is that they are still quite expensive.

Furthermore, future enhancements in EMR systems will include the ability to extract vital information from laboratory reports automatically. This integration of lab data with other EMR data will enrich the datasets used for predictions, leading to more accurate and comprehensive diagnostic insights. By encompassing a broader range of clinical information, including detailed lab results, these advanced systems will significantly refine the precision of disease prediction and patient treatment plans.

EHRs will be capable of handling massive amounts of data and complicated clinical test results in the future and eliminate current limitations and develop by using advanced existing methods and techniques to predict diseases more accurately. Related issues such as uncertainty in drawing conclusions and privacy issues will be addressed, and EHRs will come up with the genetic and behavioral data required for accurate prescribing and patient care improvement.

REFERENCES

[1] J. Wu, J. Roy, and W. F. Stewart, "Prediction modeling using EHR data: Challenges, strategies, and a comparison of machine learning approaches," Med. Care, vol. 48, no. 6, pp. S106–S113, 2010.

[2] S. Ford, "Patient-centered Medicine, Transforming the Clinical Method," Transforming the Clinical Method, vol. 7, pp. 181–182, 2004.

[3] M. A. Alkureishi, W. W. Lee, S. Webb, and V. Arora, "Integrating patient-centered electronic health record communication training into resident onboarding: Curriculum development and post-implementation survey among house staff," JMIR Med. Educ, vol. 4, no. 1, 2018.

[4] J. Stausberg, D. Koch, J. Ingenerf, and M. Betzler, "'Comparing paperbased with electronic patient records: Lessons learned during a study on diagnosis and procedure codes, ,'" J. Amer. Med. Inform. Assoc, vol. 10, no. 5, pp. 470–477, 2003.

[5] G. Makoul, R. H. Curry, and P. C. Tang, "'The use of electronic medical records: Communication patterns in outpatient encounters," J. Amer. Med. Inform. Assoc, vol. 8, no. 6, pp. 610–615, 2001.

[6] W. R. Hersh, "'The electronic medical record: Promises and problems," J. Amer. Soc. for Inf. Sci, vol. 46, no. 10, pp. 772–776, 1995.

[7] C.-S. Yu, Y.-J. Lin, C.-H. Lin, S.-Y. Lin, J. L. Wu, and S.-S. Chang, "'Development of an online health care assessment for preventive medicine: A machine learning approach, ,'" J. Med. Internet Res, vol. 22, no. 6, 2020.

[8] Y. Si, "Deep representation learning of patient data from Electronic Health Records (EHR): A systematic review," J. Biomed. Inform, vol. 115, no. 103671, 2021.

[9] X. Zhang, J. Xiao, and F. Gu, "'Applying support vector machine to electronic health records for cancer classification," in Proc. Spring Simul. Conf. (SpringSim), 2019, pp. 1–9.

[10] Z. Zeng, "'Contralateral breast cancer event detection using nature language processing," in Proc. AMIA Annu. Symp, 2017.

[11] M. Jamaluddin and A. D. Wibawa, "Patient diagnosis classification based on electronic medical record using text mining and support vector machine," in 2021 International Seminar on Application for Technology of Information and Communication, pp. 243–248.

[12] R. J. Carroll, A. E. Eyler, and J. C. Denny, "Naïve Electronic Health Record phenotype identification for Rheumatoid arthritis," AMIA Annu. Symp. Proc., vol. 2011, 2011.

[13] Y. Shen, "CBN: Constructing a clinical Bayesian network based on data from the electronic medical record," J. Biomed. Inform, vol. 88, pp. 1–10, 2018.

[14] Q. T. Zeng, S. Goryachev, S. Weiss, M. Sordo, S. N. Murphy, and R. Lazarus, "'Extracting principal diagnosis, co-morbidity and smoking status for asthma research: Evaluation of a natural language processing system, '' BMC Med," BMC Med. Informat. Decis. Making, vol. 6, no. 1, 2006.

[15] S. Sakai, K. Kobayashi, J. Nakamura, S. Toyabe, and K. Akazawa, "'Accuracy in the diagnostic prediction of acute appendicitis based on the Bayesian network model, '' Methods Inf," Methods Inf. Med, vol. 46, no. 06, pp. 723–726, 2007.

[16] K. M. Al-Aidaroo, A. A. Bakar, and Z. Othman, "Medical data classification with naive Bayes approach," Inf. Technol. J.,

vol. 11, no. 9, pp. 1166–1174, 2012.

[17] J. Kazmierska and J. Malicki, "'Application of the Naïve Bayesian Classifier to optimize treatment decisions," Radiotherapy Oncol, vol. 86, no. 2, pp. 211–216, 2008.

[18] Y. Sun and D. Zhang, "'Diagnosis and analysis of diabetic retinopathy based on electronic health records," IEEE Access, vol. 7, pp. 86115–86120, 2019.

[19] A. Lungu, A. J. Swift, D. Capener, D. Kiely, R. Hose, and J. M. Wild, "'Diagnosis of pulmonary hypertension from magnetic resonance imaging-based computational models and decision tree analysis," Pulmonary Circulat, vol. 6, no. 2, pp. 181–190, 2016.

[20] A. Tyagi and P. Singh, "'Asthma diagnosis and level of control using decision tree and fuzzy system," Int. J. Biomed. Eng. Technol, vol. 16, no. 2, pp. 169–181, 2014.

[21] R. Miotto, L. Li, B. A. Kidd, and J. T. Dudley, "'Deep patient: An unsupervised representation to predict the future of patients from the electronic health records, ,'" Sci. Rep, vol. 6, no. 1, 2016.

[22] H. Xu, "Extracting and integrating data from entire electronic health records for detecting colorectal cancer cases," AMIA Annu. Symp. Proc, vol. 2011, pp. 1564–1572, 2011.

[23] C. Breischneider, S. Zillner, M. Hammon, P. Gass, and D. Sonntag, "Automatic extraction of breast cancer information from clinical reports,'' in Proc," IEEE 30th Int. Symp. Comput.-Based Med. Syst. (CBMS), pp. 213–218, 2017.

[24] A. Jorge, "'Identifying lupus patients in electronic health records: Development and validation of machine learning algorithms and application of rule-based algorithms," Seminars in Arthritis and Rheumatism, 2019.

[25] S. Mehrabi, "'Temporal pattern and association discovery of diagnosis codes using deep learning," in Proc. Int. Conf. Healthcare Informat, 2015, pp. 408–416.

[26] X. Shi, "'Multiple disease risk assessment with uniform model based on medical clinical notes," IEEE Access, vol. 4, pp. 7074–7083, 2016.

[27] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun, Doctor AI: Predicting clinical events via recurrent neural networks. 2015.

[28] S. Wu, "'Modeling asynchronous event sequences with RNNs, ,'" J. Biomed. Informat, vol. 83, pp. 167–177, 2018.

[29] R. Miotto, L. Li, B. A. Kidd, and J. T. Dudley, "Deep patient: An unsupervised representation to predict the future of patients from the electronic health records," Sci. Rep., vol. 6, no. 1, pp. 1–10, 2016.

[30] L. Ali, C. Zhu, Z. Zhang, and Y. Liu, "Automated detection of Parkinson's disease based on multiple types of sustained phonations using linear discriminant analysis and genetically optimized neural network," IEEE J. Transl. Eng. Health Med., vol. 7, pp. 1–10, 2019.

[31] S. Farzi, S. Kianian, and I. Rastkhadive, "Diagnosis of attention deficit hyperactivity disorder using deep belief network based on greedy approach," in 2017 5th International Symposium on Computational and Business Intelligence (ISCBI), 2017.

[32] K. Adem, S. Kilicarslan, and O. Cömert, "'Classification and diagnosis of cervical cancer with softmax classification with stacked autoencoder," Expert Syst. Appl, vol. 115, pp. 557–564, 2019.

[33] U. Hwang, S. Choi, H.-B. Lee, and S. Yoon, Adversarial training for disease prediction from electronic health records with missing data. 2017.

[34] J. Latif, C. Xiao, S. Tu, S. U. Rehman, A. Imran, and A. Bilal, "Implementation and use of disease diagnosis systems for electronic medical records based on machine learning: A complete review," IEEE Access, vol. 8, pp. 150489–150513, 2020.

ACKNOWLEDGMENT

I would like to extend my deepest gratitude to all the supervisors and lecturers who have significantly contributed to this research. Their guidance and support throughout this study have been invaluable.

Journal website: