International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 10 September 2023 **Revised:** 20 October 2023 **Accepted:** 30 October 2023*

A Dense Network Model for Outlier Prediction Using Learning Approaches

Boddu L V Siva Rama Krishna[1*], V Mahalakshmi[2], Gopala Krishna Murthy Nookala[3]

[1*]*Research Scholar, Department of Computer Science and Engineering , Annamalai University, Annamalai Nagar,Chidambaram, Tamil Nadu 608002-India* Email:-krishna2928@gmail.com
[2]*Assistant Professor, Department of Computer Science and Engineering , Annamalai University, Annamalai Nagar Chidambaram, Tamil Nadu 608002-India.* Email:- mahaa80@gmail.com
[3]*Professor, Department of Information Technology ,SRKR Engineering College, Bhimavaram, Andhra Pradesh 534202-India.*Email: gopinukala@gmail.com

***Corresponding author**: Boddu L V Siva Rama Krishna
[1*]*Research Scholar, Department of Computer Science and Engineering , Annamalai University, Annamalai Nagar,Chidambaram, Tamil Nadu 608002-India* Email:-krishna2928@gmail.com

***Abstract –** There are various sub-categories in outlier prediction and the investigators show less attention to related domains like outliers in audio recognition, video recognition, music recognition, etc. However, this research is specific to medical data analysis. It specifically concentrates on predicting the outliers from the medical database. Here, feature mapping and representation are achieved by adopting stacked LSTM-based CNN. The extracted features are fed as an input to the Linear Support Vector Machine ($l - SVM$) is used for classification purposes. Based on the analysis, it is known that there is a strong correlation between the features related to an individual's emotions. It can be analyzed in both a static and dynamic manner. Adopting both learning approaches is done to boost the drawbacks of one another. The statistical analysis is done with MATLAB 2016a environment where metrics like ROC, MCC, AUC, correlation co-efficiency, and prediction accuracy are evaluated and compared to existing approaches like standard CNN, standard SVM, logistic regression, multi-layer perceptrons, and so on. The anticipated learning model shows superior outcomes, and more concentration is provided to select an emotion recognition dataset connected with all the sub-domains.*

***Keywords**- an outlier, prediction accuracy, LSTM, stacked CNN, linear SVM*

1. Introduction

Huge data has been generated in recent years, particularly over the last ten years, due to the expansion of connectivity and rising internet usage. "Streaming data" is the name given to this sort of big data. It is made up of a potentially infinite number of chronologically arranged data elements [1]. However, every single information element in a stream of data is dynamic and changes over time. As a result, time window models are frequently used to process data streams since they are potentially limitless in size and cannot be fully retained in memory [2]. Time-based Windows are possible, such as every 20 s, or data-driven, such as every 100 instances, where they determine the change gradually throughout overtime or a data window. The most popular timing window types are the damped window, landmark window and sliding window designs [3]. Each window has a fixed dimension w of the duration t, within the sliding window model. The older data points are discarded, and each window only includes the most recent ones. The importance of each region within the active window is similar. Landmark, a timestamp from which the landmark window model analyses data to the present while the other monitors the time it takes to pass while keeping any of the model's bounds fixed at a specific time instant.

On the other hand, a damped window model assigns weights to a stream's data, calculating the weight and delivery time of each object. This model also gives current data larger weights than past results, and so this weight then decreases over an exponential period by just some aging function. Due to the vast quantity of data involved, anomalous data points, also known as outliers, are prone to appear in streaming data. Hawkins [4] defines an outlier as "an observation that differs so significantly from other records as to arouse suspicions that a different process produced it." In data mining and statistics, outliers are also referred to as abnormalities, deviants or anomalies [5].
The detection of outlier data is frequently used to identify new opportunities or to give early warning of peril. Finding data items that don't match expected or typical behavior is the process of outlier detection. Because it has so many uses, among other things, fault prevention in the industrial category, fraud prevention in the finance and banking sectors, abnormal vital signs, and early detection of diseases in the medical and healthcare sector, outlier detection over streaming data is becoming more and more popular [6]. Nevertheless, outlier identification in streaming data presents a major challenge because the fundamental pattern of data distribution may not yet have been identified. As a result, this procedure for streaming data can be carried out online or in

an automated manner. Batch processing, which refers to the continuous processing of massive data, automatically identifies outliers in historical data [7]. On the other hand, real-time processing refers to identifying outliers as new data points are added in the online format. The execution of data is ensured in real-time mode shortly after it has been received. This research uses the real-time streaming processing technique to implement the sliding window setting [8].

Recent developments in many fields, including network security, natural language processing (NLP), computer vision, etc., have made deep neural networks (DNNs) the go-to method for solving various issues. Their deep design, which allows them to offer a variety of representations for learning complex features for effective prediction, has allowed them to thrive in these fields. Additionally, the manual feature engineering needed by traditional ML methods, generalization to new data modifications, and scalability are all addressed by the deep architecture of DNN. As a result, DNN is a good method for detecting outliers thanks to these qualities [9] – [12]. The model is trained over numerous layers using a deep neural network, a supervised learning method. "Deep" means the number of layers to retrieve the relevant data. An output layer, an input layer and several deep layers are all included in the general structure of a DNN shown. A layer is a collection of neurons layered in various ways, such as recurrent and convolutional, fully connected and pooling layers. Multilayer perceptron networks (MLP), recurrent neural networks (RNN), and convolutional neural networks (CNN) are just a few of the popular neural networks that can be constructed using these layers [13].

By combining intricate sets of either nonlinear or linear functions that a computational graph can express, DNNs create expressive representations. These are called "activation functions" because they predict the outcome of neurons in neural networks or computational graph nodes [14], given specific inputs. ReLU (Rectified Linear Unit), sigmoid, tanh, and linear are typical activation functions [15].

This paper describes a novel deep learning-based model that utilizes the stacked Long Short Term Memory with Convolutional Neural Networks ($S-LST$M) to identify outliers in the context of data streams. The suggested model employs the slide window method to handle streaming data and offers an online $S-LST$M -based architecture for instantaneous outlier instance detection. In the suggested model, $S-LST$M is constructed with an output layer, an input layer and three hidden layers, with two neurons for documenting instances as outliers or inliers. The sigmoid function activates the output layer, while the ReLU function activates the three concealed layers. The following are the study's most important contributions:

1. A novel online outlier detection method based on the $S-LST$M technique is suggested for streaming medical data.
2. The suggested model uses a real-time processing mode and feature representation concept to handle streaming data flow.
3. The use of deep learning techniques to identify outliers in data stream situations is reviewed in the research.
4. Thorough experimental investigation of numerous deep learning-based techniques for outlier detection methods were carried out on benchmark datasets for outlier detection.
5. The suggested model outperforms state-of-the-art approaches and gets the best trade-off by getting a high detection rate, although achieving a low miss rate.

This document is organized for the remaining portions: A literature summary is provided in the "Related Work" part. The proposed model used in this study is described in "The proposed $S-LST$M model" section. The experimental outcomes and evaluations section contains a review of the results, a full description of the datasets used, and an experimental setup. Lastly, the Recommendations and Future Efforts Section concludes the proposed task by outlining a course for the future.

2. Related works

Rule-based systems, data clustering, non-parametric and parametric statistical models, SVMs (Support Vector Machines), and mixture models, among other methods, have all been used to identify anomalies. The curious reader can consult comprehensive surveys [16]. These conventional models frequently need to capture complicated data structures accurately. Furthermore, finding outliers at scale using traditional techniques may become more challenging as data volume rises. Therefore, for developing real-world use cases, the efficacy of the algorithms mentioned above in detecting outliers may be sub-optimal. Deep learning-based algorithms for outlier detection have grown in popularity recently and are used for various jobs [17]. Deep learning-based unsupervised outlier detection has primarily taken a hybrid form. The deep neural network first picks up on the intricate data trends. The traditional outlier detection methods are then fed the hidden layer representations out of this trained network. For hybrid outlier detection using deep learning, there are two common types. The first category includes techniques that look at reconstruction flaws in an auto-encoder constructed from common data.

A defect in a test point's reconstruction suggests an outlier [18]. The second set of methods either creates a low-dimensional embedding using an auto-encoder trained over its normal class or employs a neural network to generate predictions. To find anomalies in the embedding or forecasts, traditional techniques are used, such as an OC-SVM (One Class-SVM) [19], a parametric distribution assumption [20], etc.

Despite the fact that the widely used hybrid deep learning-based outlier detection methods has proven to be successful in a variety of tasks, these neural networks are not specifically created for outlier detection. The hybrid models unaffected representational learning in the hidden layers because they retrieve features using a neural network and then pass that information to a different outlier detection technique. The detection and encoding steps are combined in an improved version of this approach by employing a suitable objective function to create a single neural model that is capable of performing both tasks [21].

The authors of a related study [22] employ geometrical transformations to carry out end-to-end deep learning-based outlier detection using CNNs (Convolutional Neural Networks). A feed-forward neural network with an OCSVM objective is used for deep outlier identification [23]. In image data sets, outlier detection is the main emphasis of the literature, as mentioned above, for the outlier detection methods designed for pictures to work well with time sequences. Consequently, the goal of this research is to develop an end-to-end outlier detection system via a neural network designed for sequential data, the LSTM (Long Short-Term Memory) network [24]. We create an end-to-end LSTM-based anomalous detection model by drawing knowledge from EVT (Extreme Value Theory) [25]. An LSTM-based end-to-end deep outlier identification model for transportation data has not yet been thoroughly researched in the literature.

Additionally, the findings from EVT are used to inform our objective function and network weight update rules. A neural network model for conducting outlier detection has yet to be trained using Extreme Value Theory. These qualities distinguish our study from earlier research works [26] – [28]. A comprehensive deep outlier detection algorithm is proposed. It is compared to several baseline models, including the hybrid LSTM outlier detection models, the non-parametric SVM model and the parametric GARCH (Generalized Auto Regressive Conditional Heteroskedasticity) model, with various detection rules [29]. The hybrid deep outlier detection models' rules are based on Tukey's technique, EVT, and Gaussian distribution assumptions [30]. According to an analysis of seven different data sets, our suggested $S-LSTM$ model beats the hybrid deep learning baseline and conventional statistical, machine learning models. This research emphasizes the need for a tailored neural network model based on deep learning in an outlier detection setting.

3. Methodology

Three distinct phases that are coupled to one another make up the expected model for anticipating the medical data: 1) pre-processing, 2) $S-LSTM$ with CNN, and 3) detection phase. Fig. 1 provides the projected model's flow. The methodology is applied over the provided dataset and thoroughly examined in successive phases. Here, an online available dataset (https://physionet.org/content/adfecgdb/1.0.0/) is used for evaluation. The dataset includes recorded signals, ECG recorded from Fetal head, electrode position, bandwidth, sampling rate, resolution and input range. Some essential pre-processing is performed to analyze the outliers over the provided data.

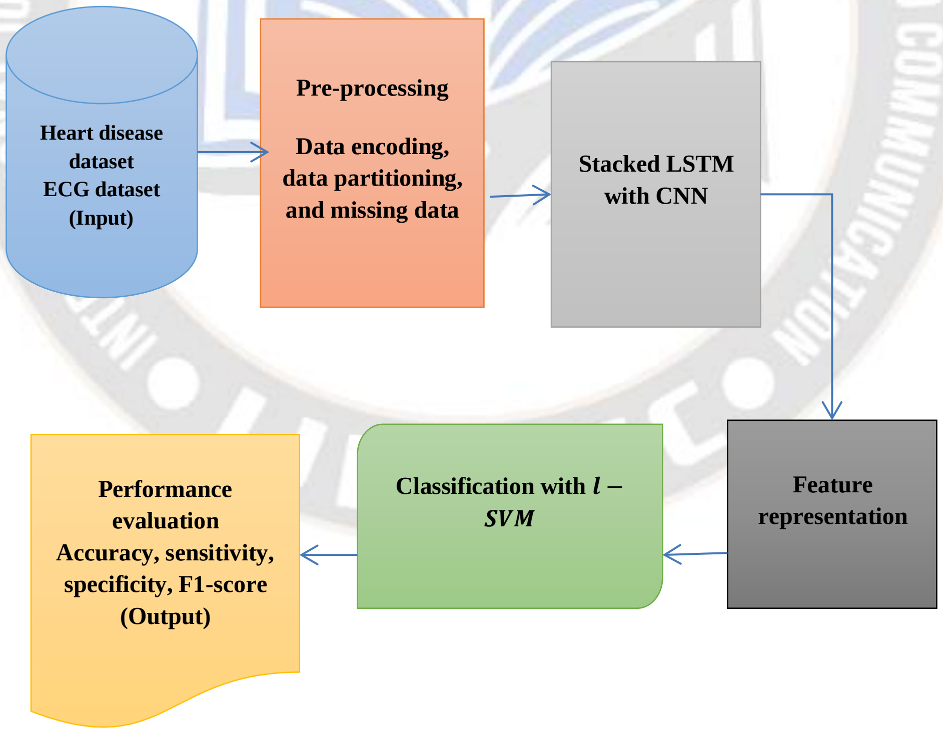
Heart disease dataset ECG dataset (Input)
Pre-processing
Data encoding, data partitioning, and missing data
Stacked LSTM with CNN
Feature representation
Classification with $l-SVM$
Performance evaluation Accuracy, sensitivity, specificity, F1-score (Output)

Fig 1 Workflow of the outlier prediction model

a) Pre-processing

Promoting the relevant features from the provided data is essential to enhance the data quality. Thus, it is the preliminary phase in outlier prediction for clean-up and raw data processing to train and generate the $S-LSTM$ model.

1) Missing data: It is highly solicited to deal with the missing data. The variables with missing values can be replaced with the variable-based data type, which can be categorical or numerical. When the missing data is numerical, the variable fills in the blanks. When it is missing data is categorical, the variable mode is adopted. It can eliminate the erroneous bias produced due to the missing values.

2) Data encoding: Here, the categorical data is transformed into numerical data using a one-hot encoding technique.

3) Data partitioning: For data preparation, the dataset is divided into data for testing and training. The suggested S-LSTM model is trained using the training set. However, the testing set is employed to test the model and measures the unknown test set. The data is partitioned into 70% training and 30% testing set in this work.

4) Feature scaling: It is an approach for independent variable standardization over a certain range. However, feature scaling can reduce the number of variables to a manageable manner. Every feature is standardized except for the target label in training data as in Eq. (1):

$$x' = \frac{x - mean\,(x)}{\sigma} \qquad (1)$$

Here, x' specifies the new value of data points, x refers original data point, and σ specifies SD.

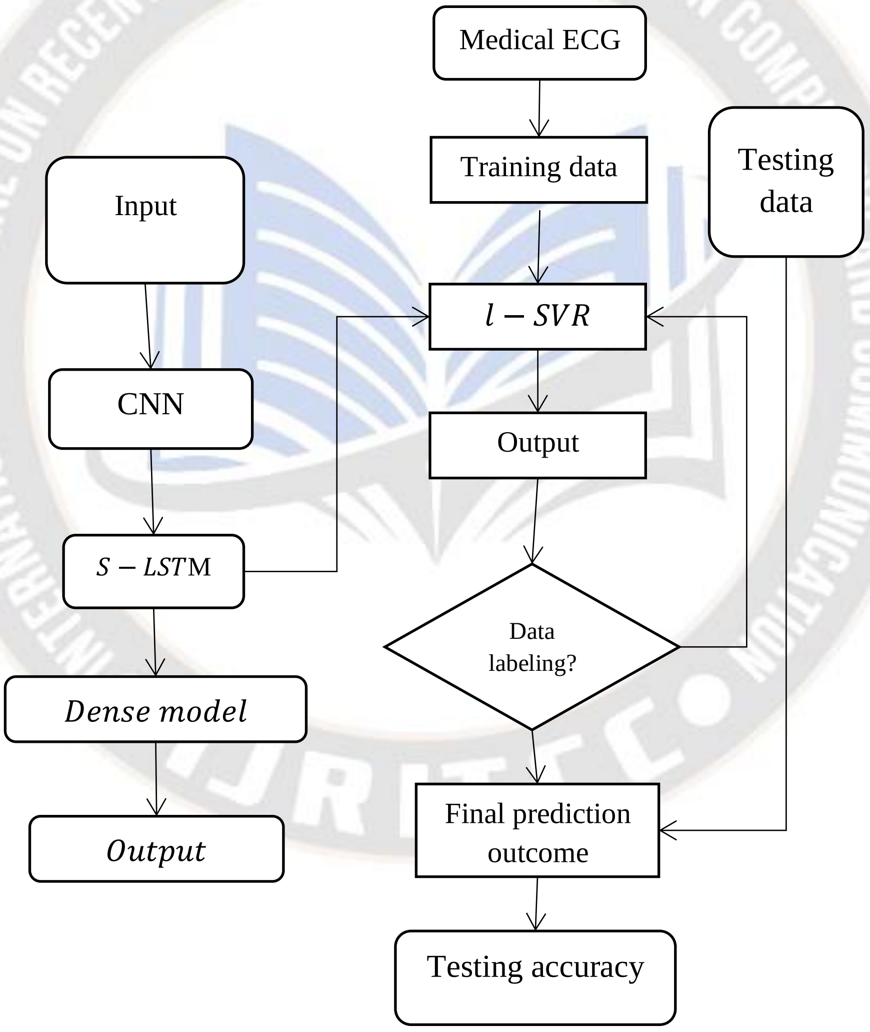

Fig 2 Proposed flowchart

b) Stacked LSTM with CNN

In addition to the number of filters and layers in every layer, LSTM and CNN also have a variety of other variables. You can choose an early indicator of the data being used to help you achieve your desired results. The height of each filter layer on CNN is crucial. Because the $5 * 5$ *and* $3 * 3$ filters are frequently used in image analysis, this article suggested that now the size of each filter be consistent with earlier work in image processing. $S - LST$M, a general medical data forecasting framework built on LSTM and CNN, is

introduced here. This paper's framework is divided into three major stages: input data representation, continuous feature extraction, and final prediction.
Input data analysis: It gathers data from various medicals to forecast the future of these medical data. This research tries to identify a common model that transforms previous medical data into expected fluctuations. This article uses a general model that can be used in a variety of medical fields. Many real mapping functions from the past to the present are accurate. To accomplish this objective, this study needs to create a single model that, using analysis of the past data, can predict the future of medical data. However, this framework must be trained using examples from different medicals to derive the necessary mapping function. However, it also employs eleven other pieces of information that are identical to the data variables of the day, in addition to employing medical history modeling and a wide range of other factors (such options and futures) as data input. A generated framework receives a two-dimensional tensor from this algorithm's combination of these signals.

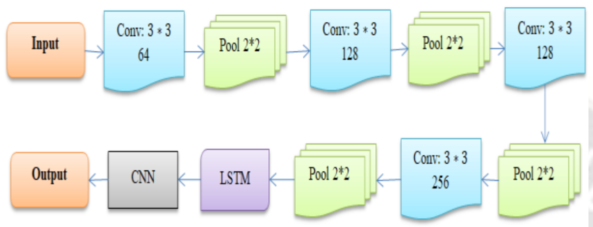

Fig 3 Stacked LSTM-CNN model

Feature representation: Several variables, such as the opening price, ending price, highest price, volume, and lowest price, illustrate the historical data for each day. By analyzing these variables, such as candlesticks, the conventional medical forecasting technique can forecast the medical's future trend by building high-level features. CNN image identification technology inspired the first layer design of $S-LST$M. A convolutional layer's job in the first stage of $S-LST$M can create higher-level features from daily variables to define each day, as seen in Fig 3. Medical trends over a period can provide some insightful information that can help forecast future medical behavior. With the help of this data, we can identify patterns and learn more about current medical behavior trends, which we can then use to forecast upcoming trends. To gather high-level features that reflect patterns or impact medical behavior over a specific period, it is crucial to combine 30 consecutive days of data variables into an image. Within a set amount of time, CNN's convolutional and pooling layers produce more intricate features that describe the data.
Final prediction: Advanced features created in the preceding layer are further processed by the LSTM unit into a one-dimensional vector, which is then employed to map features to predicted fully connected layers.The convolutional and pooling layers' advanced features are sent into the LSTM.
The following part of the paper covers the general design of $S-LST$M and the way it is utilized to the particular experiments in this paper using the dataset. For our experiment, this research utilized data from 10 stocks from 2 locations. Besides its options, futures, and historical data, each stock acquired ten bits of data that were similar to its data in order to better predict the outcome.

Table 1 Parameter settings

Parameters	**Levels**
Epochs	200, 300,…,800
Learning rate	0.1, 0.01, 0.001, 0.00001
Activation functions	ReLU
LSTM layers	1, 2, 3
No. of the hidden layer (neurons)	64, 128…512
No. of hidden layer	3, 4, 5

Algorithm: $S-LST$**M**

```
Input: Training data $'b'$, testing data $'c,'$ no. of iterations $I$, batch size $'B'$, and optimizer $adam$
Process: Training model, evaluation outcomes
1: Initialization;
2: $b \rightarrow$ initialization;
3: $P: \rightarrow$ data partitioning.'
4:  for every round $t = 1,2, \ldots, z$ do
5: $\{training, verification\} \rightarrow \{P_t, P - P_t\}$
6: $(t_f, v_f) \rightarrow generate\ training\ and\ verification\ process$;
7: $n_t \rightarrow\ Fit.(adam, t_f)$
8: $r_t \rightarrow Evaluate.(n_t. v_f)$
9: end for
10: $n \rightarrow best\ model$
11: $c \rightarrow n$
12: $accuracy \rightarrow Evaluate.(test, n)$
```

c) Model functionality

The following describes the precise development process of the $S - LST$M.

Input data: A two-dimensional matrix functions as the S-LSTM's input, as was previously explained. The number of variables and the duration of the retracing history of estimates determine the dimensions of the matrix. The dimension of the input tensor is $g \times f$ if the input employed for prediction is g days and the f variable denotes every g day.

Feature representation: In $S - LST$M, an initial variable filter extracts the 30-day shift feature. The 3×3 filter that CNN uses most frequently for images is chosen, and you can use these filters to merge images into a matrix with even more sophisticated features. It can build various combos of host variables using this layer. By changing the weight of the appropriate variable to 0 in the filter, the network also can remove useless variables. The layer therefore acts as a feature selection module. The different convolution and pooling effects listed below integrate lower-level features from their inputs to higher-level inputs, aggregate the information that is available over a certain period of time, and create features at a higher level. In the second layer, 64 filters are applied. Every filter runs nonstop for three days. The best-known candlestick designs for distinctive patterns over three days. The said setting is used in this study's data collection as a sign to identify any possibly helpful messages within a time window of three succeeding time units. In the third layer, the pooling layer performs up to 2×2 of pooling. $S - LST$M employs another convolutional layer with 128 filters to create more complex features and aggregate data over extended periods. This layer is comparable to the first pooling layer and another pooling layer. This article used the 2 layers of convolution and pooling, each with 256 filters, to acquire more accurate information. The pooling layer's size is the same as the size of the one before it.

Prediction: Before tiling them into the final feature vector to realize the final prediction, the LSTM unit takes the features produced by the final pooling layer and uses them to recover deeper features. According to $S - LST$M medical forecast, prices may increase on the following prediction process. In contrast, this article discretizes the output in our experiment, which is close to the anticipated value. $S - LST$M configuration for days, as was previously stated, is included in the input of this article for every prediction, and a variety of variables depicts every 30 days. $30 * 30$ variable number array serves as the input for 2D-CNN. The first convolutional layer employs a $3 * 3$ filter, each filter follows by a 2x2 max-pooling layer: 128, 256, and 256. The LSTM unit is then used to produce the final output. The visualization of the graphical procedure is shown in Fig 3.

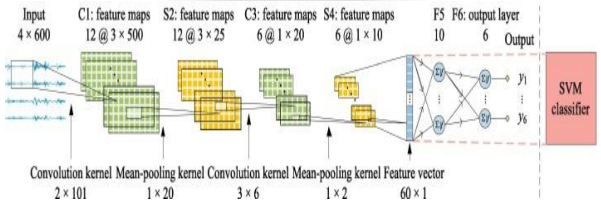

Fig 4: $\boldsymbol{l} - SVM$ model

d) Network parameters

MATLAB 2020a is used to create CNN and LSTM as deep learning software and packages continue to advance. Relu serves as each layer's activation function in the CNN architecture. Each convolutional layer has 64, 128, 256, and 256 filters. Additionally, the network was trained using LSTM and Adam.

e) Linear SVM

A hybrid $l - SVM$ model is suggested here to classify the outliers in the medical dataset. The suggested method combines the best elements of CNN and SVM classifiers. Multiple completely connected layers comprise a convolutional neural network (CNN) with a supervised learning process. CNN can acquire invariant feature points quite effectively and functions similarly to humans. From unprocessed digit images, it can obtain the most discriminating information. In the suggested method, the most recognizable characteristics from the original data images are extracted using a $5x5$ kernel/filter. An output of magnitude $(n - m + 1) * (n - m + 1)$ is produced by convolving $n \times n$ input neurons from the input layer with $an\, m \times m$ filter in the convolutional layer. Each layer's output serves as the stimulus for the layer below it. Based on the raw digit picture, CNN's receptive field feature determines the effective sub-regions. In a domain where data elements from different classes are split by a hyper-plane, Support Vector Machine (SVM) intends to represent multi-dimensional datasets. The generalization error on unknown data can be reduced using the SVM algorithm. An alternative name for the separating hyperplane is an optimum hyperplane. SVM is deemed to be poor for noisy data and found to be excellent for binary classification. Understanding deep features can be challenging due to SVM's shallow design. The hybrid $l - SVM$ model is presented in the current work, where SVM is used as a binary classifier and takes the position of CNN's softmax layer. SVM functions as a binary predictor and CNN as a feature extractor. Fig 4 describes the suggested hybrid $l - SVM$ model's architecture.

The model comprises an SVM classifier and a straightforward CNN architecture. A medical dataset $28 * 28$ array of cantered, normalized handwritten digits is used as the CNN input layer's input. The convolutional layers employ a stride of size $2\, and\, 55$ convolutional filtration. The $N1\, and\, N2$ feature map layers extract values from the input image's distinguishing characteristics. After executing several epochs and waiting for the training phase to merge, the CNN system has been trained. SVM classifier is used instead of the final layer of CNN. The $N3$ layer features of the handwritten input number serve as a source of input for the SVM classifier. These newly automatically created training image features are used to teach the SVM classifier at first. To identify the test-related numbers, the trained SVM classifier is utilized.

4. Numerical results

In this section, we go over the data sets that were taken into account, the evaluation criteria that were used, and the process for choosing the parameters for every outlier detection model. We take into account three assessment measures to contrast our models: The three terms are recall (R), precision (P), and the harmonic sum of recall and precision, or the F1-score. Each data set is min-max normalized prior to modeling and evaluation.

$$P = \frac{TP}{TP + FP}$$
$$R = \frac{TP}{TP + FN}$$
$$F = 2 * \frac{P * R}{P + R}$$

The strange occurrences in the model are accurately identified as outliers, i.e., true positives. The instances properly classified as non-anomalous data are known as true negatives. False negatives are the anomalies incorrectly recognized, and Non-anomalies that are wrongly labeled as abnormal are considered false positives. Because it summarizes Precision and Recall, we consider the model with the greatest F1 score to be the most effective outlier detection technique. We choose the test and validation sets for each data set to maintain the seasonality in the sequences while including both anomalous and non-anomalous cases. The division between training and validation tests is roughly 60-10-30. For each model, the proper hyper-parameters and outlier thresholds must be specified to perform outlier detection effectively. Different parameters and limits are appropriate depending on the use case under consideration. The methods used to select the parameters for every outlier detection model are quickly discussed below.

Table 1 Parameter evaluation

Type	**Model**	**Evaluation parameter**			
		Accuracy	**Sensitivity**	**Specificity**	**F1-score**
Linear	DenseNet	0.57 ± 0.10	0.57 ± 0.10	0.87 ± 0.04	0.57 ± 0.13
	CNN	0.70 ± 0.09	0.70 ± 0.09	0.90 ± 0.05	0.70 ± 0.12
	CNN+LSTM	0.79 ± 0.05	0.79 ± 0.05	0.90 ± 0.03	0.78 ± 0.05

Article Received: *10 September 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *30 October 2023*

	Stacked model	0.91 ± 0.10	0.91 ± 0.03	0.90 ± 0.25	0.90 ± 0.25
	stacked LSTM-based CNN with $l-SVM$	0.95 ± 0.10	0.94 ± 0.03	0.93 ± 0.25	0.93 ± 0.25
Convex	DenseNet	0.53 ± 0.03	0.53 ± 0.03	0.66 ± 0.08	0.51 ± 0.07
	CNN	0.61 ± 0.04	0.61 ± 0.04	0.75 ± 0.09	0.58 ± 0.03
	CNN+LSTM	0.67 ± 0.03	0.67 ± 0.04	0.76 ± 0.14	0.66 ± 0.03
	Stacked model	0.91 ± 0.15	0.91 ± 0.03	0.90 ± 0.25	0.90 ± 0.25
	stacked LSTM-based CNN with $l-SVM$	0.94 ± 0.10	0.93 ± 0.03	0.92 ± 0.25	0.92 ± 0.25

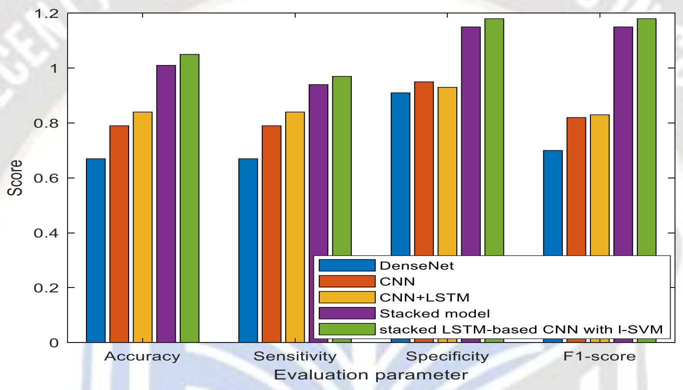

Fig 5 Evaluation parameters (Linear)

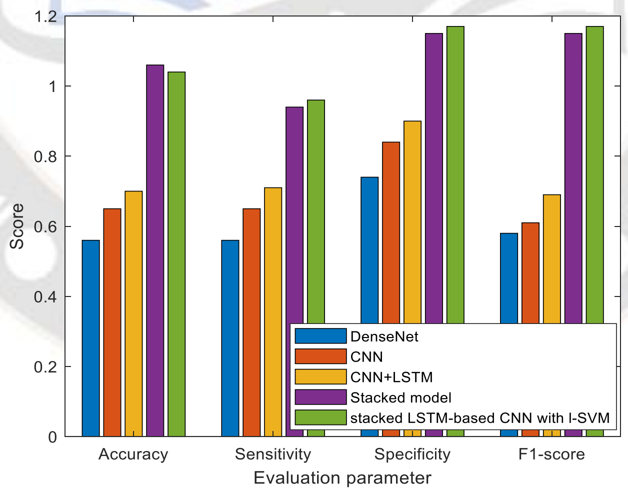

Fig 6 Evaluation parameters (Convex)

Table 2 5-fold CV

Type	**Model**	**Cross-validation**					
		$K = 1$	$K = 2$	$K = 3$	$K = 4$	$K = 5$	**Average**
Linea r	Accuracy	94	94.5	95	95	95.2	95 ± 0.75
	Sensitivity	93.8	94.1	94	93.5	94	94 ± 0.46
	Specificity	93	92.5	93	93	93.2	93 ± 0.78
	F1-score	93	92.5	93	93	93.2	93 ± 0.22

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 10 September 2023 **Revised:** 20 October 2023 **Accepted:** 30 October 2023*

Conv ex	Accuracy	93.8	94.1	94	93.5	94	94 ± 0.98
	Sensitivity	93	92.5	93	93	93.2	93 ± 0.5
	Specificity	91.5	92	92	92.3	92.4	92 ± 0.26
	F1-score	91.5	92	92	92.3	92.4	92 ± 0.72

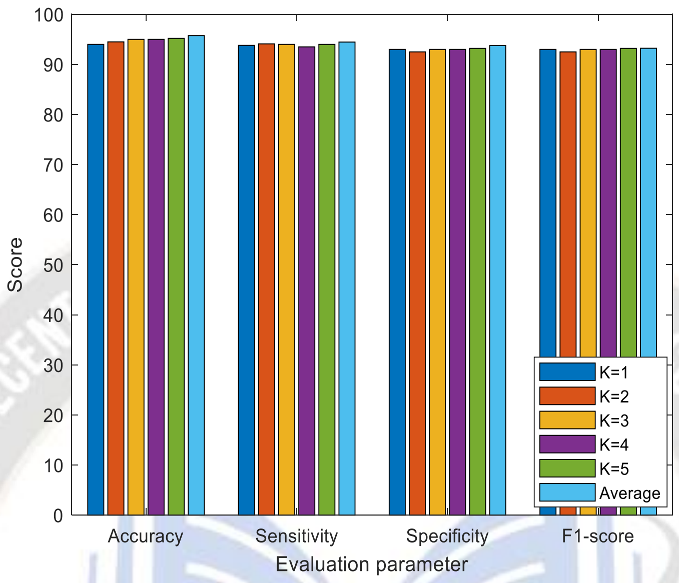

Fig 7 Evaluation parameters based on k-value

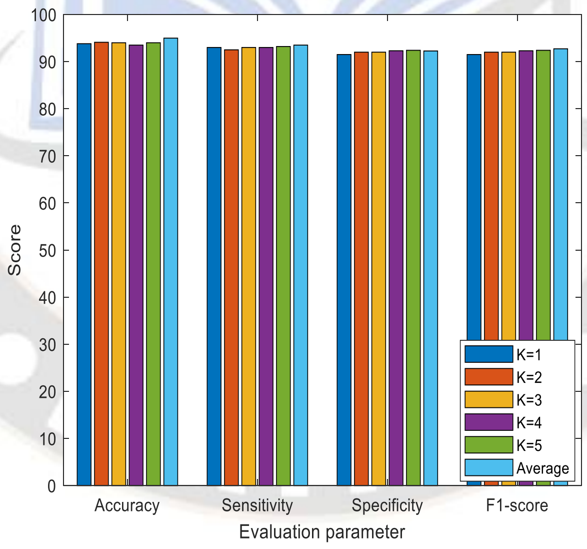

Fig 8 Evaluation parameters (average)

The choice of kernel functions depends on the data sets under consideration and is essential for SVM-based outlier detection performance to be acceptable. This research considers RBF, linear, sigmoid, and polynomial kernels. The kernel coefficient α for the sigmoid, RBF and polynomial kernels is an additional important variable. After changing between [0.0001, 0.1], it is discovered that 0.0001 best fits most of the data sets considered. With various parameters selected from the range of values considered, we ran numerous SVM models on the training data for each use case. The classification precision on a hold-out validation set is then examined to determine the best action. Finally, anomalies on a test set are detected using the best $l - SVM$ model that could be found.

Any neural network model should have the proper hyperparameter values specified before training. Since they relate to the model selection and affect learning speed, these parameters cannot be inferred while the model is being trained. Topology and network size are two hyper-parameters related to the model selection process. Similarly, some hyper-parameters that influence the pace and effectiveness of the learning process include dropout, learning rate, and mini-batch size. We use a Bayesian optimization method called the TPE (Tree-structured Parzen Estimator) algorithm for this exercise because choosing the appropriate hyper-parameters takes work. We define an output layer as a dense, fully linked layer that has RELU activation. The Adam optimizer is applied with the purpose of lowering the Mean Squared Error. All LSTM-based models are learned using a 64-person mini-batch over 100 epochs (See Fig 5 to Fig 8).

The authors have chosen a few appropriate parameters for this data set to analyze the medical data. It is observed that the LSTM learns improved data representation for situations with large l_b, assisting the outlier detection procedure. The false positive regulator is an important variable impacting how accurately the detection systems work. This parameter is the appropriate threshold, $\tau_g \, or \, \tau_t$, for the Gaussian detection criterion. The false positive regulator is the ideal probability q for SVM-based methods. For the Gaussian-based hybrid outlier detection, we choose a threshold τg such that it optimizes the F1-score on the validation set. Based on a straightforward quantile calculation, the threshold τ_t is immediately obtained from the complete set of prediction errors. For both end-to-end and hybrid S-LSTM deep learning models, we employ comparable techniques to determine the parameters for the rule. An initialization data stream is created by joining the prediction errors from the training and validation groups. Based on this data stream, the intended probability q for the $S-LST$M rule is selected. The initial threshold T, usually set to 98% quantile, is chosen using the same procedure. The proposed approach detects all anomalies in the data stream because q, the false positive regulator is chosen. Table 1 lists the selected values for the hybrid LSTM-based methods' false positive regulators.

The mixed LSTM models' selected hyper-parameters and false positive regulators are also applied to the $-LSTM$ model. We adhere to the recommendations when establishing the hyper-parameter for the network weight regularizer. At intervals of $k = 20$ epochs, the threshold is changed. The settings chosen for hybrid deep learning models are equivalent to end-to-end deep learning models in the majority of circumstances taken into account. The Bengaluru taxi demand data set is an exception, where the appropriate number for q is 105. However, the [103, 105] range continues to hold the greatest options for probability.

4.1. Numerical results

To ascertain whether the data distribution adheres, we run the A-D (Anderson-Darling) test, a statistical test. It is suitable for determining whether a selection of data is drawn from a specific probability distribution. The specific distribution is used when determining the critical numbers for this test. The empirical data CDF is used to compute the distance between the hypothesized distribution and the test statistic. In our instance, the data follow a particular distribution according to the null hypothesis. The null hypothesis can or cannot be denied based on the results of the test static and the obtained p −values. A-test, a variation of the KS (Kolmogorov-Smirnov) test, assigns the tails more weight than the KS test. The excesses X-T or prediction mistakes above the empirical threshold T are the subjects of the A-D test. It contains the p −values that were derived from this statistical test. The null hypothesis for each set of data is disregarded if the corresponding p-value is below 0.001. The chart shows that the prediction error distribution for each data set is taken into account. This outcome validates our recommendation to employ a l-SVM-based detection rule.

Based on various F1-score metric, Table 2 show the efficacy of various models in detecting anomalies across various data sets. From the chart, we deduce the following:

• Due to the parametric $l - SVM$ models' poor performance, it is possible to significantly reduce the accuracy of outlier identification by assuming a specific distribution for the prediction errors.

• Over seven different data sets, deep learning-based algorithms for outlier identification outperform statistical and machine learning-based algorithms regarding detection accuracy.

• Among the two types of deep learning-based outlier detection models taken into consideration, an end-to-end detection approach outperforms hybrid detection models on a variety of data sets.

The parametric $l - SVM$ model has a very low Precision but an adequately high Recall for outlier detection. Many non-anomalies on the test set are classified as abnormal by the criterion based on the validation set. Due to the numerous false positives, the total performance of outlier detection is thus negatively impacted, which lowers the F1-score value. This exception is caused by the size of the anomalies in these data sets is significantly larger than that of the non-anomalies. Compared to the statistical model, the $l - SVM$ model gets higher detection accuracy but performs poorly compared to the deep learning alternatives. Additionally, they exhibit low Precision and high Recall numbers. However, a single kernel coefficient value of α (0.0001) demonstrated a satisfactory fit for all the data sets under consideration.

The suggested end-to-end $l - SVM$ model exhibits superior detection accuracy compared to hybrid and deep outlier detection models. For most data sets considered, outlier detection performance needs no post-processing tools. It is consistently at the same level when compared to the hybrid models. This conclusion suggests that, when employing conventional algorithms, a deep learning model developed for outlier detection can generate results with higher accuracy than a deep learning model developed for forecasting. The ECG data collection contains a single exception, explained by the outlier labeling strategy. The labeling method applied to this data set designates an entire ECG signal period as abnormal if even one point during that period is abnormal. So, in this set of data, we work with collective anomalies. As a result, the ECG data set has a larger outlier fraction than other data sets with point anomalies. As a result, the anomalies above the top quartile of ECG data prediction errors span a wide range. $l - SVM$ approach produces accurate outlier identification for the ECG data set because it limits the raw prediction errors depending on the upper quartile. This finding suggests that when the data set's anomalies % is high, a clear threshold determined by the amount of prediction mistakes may be sufficient. The l-SVM model often detects the majority of abnormalities but generates a large number of false positives. This behavior is not desired in the context of outlier detection.

Regarding the difference in false positive regulator values of various approaches, an important observation is made. The false positive regulator values of the Gaussian detection rules exhibit high variability, as shown by the findings from Tables 1 and 2. Different thresholds for $\tau_g \; and \; \tau_e$ can be selected depending on the data collection under consideration. It is discovered that τ_t can assume values between [0.11, 12,961.8] while τ_g varies between $[-15, -25]$. The application of such detection rules is limited by their dependency on the outlier criteria on the time sequence under consideration.

However, the probability q, the only free parameter for l-SVM-based detection, is not overly reliant on the data collection. It is found that this false-positive regulator still lies between [10-3, 10-5].The use of a detection method based on l-SVM is supported by the fact that false positive parameters with little reliance on data sets are highly preferred in real-world scenarios. The increase in accuracy that our proposed model was able to accomplish using data sets from diverse transportation sectors shows how generalizable the end-to-end deep learning-based l-SVM model is. A wide variety of data sets can be combined with the model. It also requires no specific post-processing techniques, which is an advantage over the popular hybrid deep learning-based outlier identification models.

Additionally, our algorithm is unconstrained and doesn't need any labels for anomalies. We concluded that an end-to-end deep learning-based outlier identification system has tremendous promise for detecting instances of aberrant traffic by considering sets of data from several transportation network sectors. Along with data sets from the medical, our suggested $S - LST$M model successfully identified instances of outliers over other data.

5. Conclusion

This study presents $S - LST$M with CNN and $l - SVM$-based outlier detection framework for streaming data settings that can effectively classify data samples as normal or outlier. The results of the experiment showed that the proposed model outperforms the two cutting-edge DL approaches in terms of detection accuracy, with the lowest rate of false alarms ranging from 2.5 to 3.5%. The proposed model is evaluated on numerous real-world datasets. The detection rate or recall of the framework for detecting outliers was also found to range between 98 and 99%. The suggested model also did significantly better than the conventional ML methods. However, the current iteration of the proposed model still needs to be improved due to the time needed to train the model and its exclusive emphasis on detecting global outliers. We plan to build on this work in future work by modifying our methodology to multiclass classification settings and leveraging additional DL techniques in order to more successfully solve the outlier identification issue in the context of data streams. We also plan to discuss the contextual outlier issue.

REFERENCES

1. Momtaz, N. Mohssen, and M. A. Gowayyed, ‘DWOF: A robust density-based outlier detection approach,’’ in Proc. Iberian Conf. Pattern Recognit. Image Anal., 2013, pp. 517–525.
2. Wu, K. Zhang, W. Fan, A. Edwards, and P. S. Yu, ‘‘RS-forest: A rapid density estimator for streaming outlier detection,’’ in Proc. IEEE Int. Conf. Data Mining, Dec. 2014, pp. 600–609.
3. Vázquez, T. Zseby, and A. Zimek, "Outlier detection based on low-density models," Proc. ICDM Workshops, 2018, pp. 970–979.
4. Su, L. Xiao, L. Ruan, F. Gu, S. Li, Z. Wang, and R. Xu, ‘‘An efficient density-based local outlier detection approach for scattered data,’’ IEEE Access, vol. 7, pp. 1006–1020, 2019.
5. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori, ‘‘Statistical outlier detection using direct density ratio estimation,’’ Knowl. Inf. Syst., vol. 26, no. 2, pp. 309–336, 2011.
6. Eskin, ‘‘Outlier detection over noisy data using learned probability distributions,’’ in Proc. 17th Int. Conf. Mach. Learn. (ICML). Jul. 2000, pp. 255–262.
7. X. Tang, R. Yuan, and J. Chen, ‘‘Outlier detection in energy disaggregation using subspace learning and Gaussian mixture model,’’ Int. J. Control Autom., vol. 8, no. 8, pp. 161–170, 2015.
8. Salman, ‘‘A new algorithm for detecting outliers in linear regression,’’ Int. J. Statist.Probably., vol. 2, no. 3, pp. 101–109, Aug. 2013
9. [9] Dalat, A. Fitrianto, and A. Mustapha, "A comparative study of linear and non-linear regression models for outlier detection," Proc. Int. Conf. Soft Comput. Data Mining, 2017, vol. 549, pp. 316–327.
10. Latecki, A. Lazarevic, and D. Pokrajac, "Outlier detection with kernel density functions," Proc. 5th Int. Conf. Mach. Learn. Data Mining Pattern Recognit., 2007, pp. 61–75.
11. Samparthi and H. K. Verma, "Outlier detection of data in wireless sensor networks using kernel density estimation," Int. J. Comput. Appl., vol. 5, no. 7, pp. 28–32, 2010.
12. Boedihardjo, C.-T. Lu, and F. Chen, ‘‘Fast adaptive kernel density estimator for data streams,’’ Knowl. Inf. Syst., vol. 42, no. 2, pp. 285–317, Feb. 2015.
13. Uddin, A. Kuh, and Y. Weng, ‘‘Online bad data detection using kernel density estimation,’’ in Proc. IEEE Power Energy Society General Meeting, Jul. 2015, pp. 1–5.
14. Zhang, J. Lin, and R. Karim, "Adaptive kernel density-based outlier detection for non-linear systems," Knowl.-Based Syst., vol. 139, pp. 50–63, Jan. 2018.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 10 September 2023 **Revised:** 20 October 2023 **Accepted:** 30 October 2023*

15. Rousseeuw and M. Hubert, ‘‘Robust statistics for outlier detection,’’ Data Mining Knowl. Discovery, vol. 1, no. 1, pp. 73–79, 2011.
16. Angiulli and C. Pizzuti, ‘‘Fast outlier detection in high dimensional spaces,’’ in Proc. Eur. Conf. Princ. Data Mining Knowl. Discovery, Sep. 2002, pp. 15–26
17. Dang, H. Y. T. Ngan, and W. Liu, "Distance-based k-nearest neighbors outlier detection method in large-scale traffic data," Proc. IEEE Int. Conf. Digital Signal Process., Jul. 2015, pp. 507–510.
18. Ghoting, S. Parthasarathy, and M. E. Otey, ‘‘Fast mining of distance-based outliers in high-dimensional datasets,’’ Data Mining Knowl. Discovery, vol. 16, vol. 3, pp. 349–364, Jun. 2008.
19. Bhattacharya, K. Ghosh, and A. S. Chowdhury, "Outlier detection using neighborhood rank difference," Pattern Recognit. Lett., vol. 60, pp. 24–31, Aug. 2015.
20. Radovanović, A. Nanopoulos, and M. Ivanović, "Reverse nearest neighbors in unsupervised distance-based outlier detection," IEEE Trans. Knowl. Data Eng., vol. 27, no. 5, pp. 1369–1382, May 2015.
21. Ha, S. Seok, and J.-S. Lee, ‘‘A precise ranking method for outlier detection,’’ Inf. Sci., vol. 324, pp. 88–107, Dec. 2015
22. Angiulli and F. Fassett, ‘‘Very efficient mining of distance-based outliers,’’ Proc. 16th ACM Conf. Inf. Knowl. Manage., Nov. 2007, pp. 791–800
23. Vu and V. Gopalakrishnan, ‘‘Efficient pruning schemes for distance-based outlier detection,’’ in Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases, 2009, pp. 160–175.
24. Angiulli and F. Fassett, "Detecting distance-based outliers in data streams," in Proc. 16th ACM Conf. Inf. Knowl. Manage., Nov. 2007, pp. 811–820.
25. Bhaduri, B. L. Mathews, and C. R. Giannella, ‘‘Algorithms for speeding up distance-based outlier detection,’’ in Proc. ACM KDD Conf., Aug. 2011, pp. 859–867.
26. Salehi, C. A. Leckie, M. Moshtaghi, and T. Vaithianathan, "A relevance weighted ensemble model for outlier detection in switching data streams," Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining, 2014, pp. 461–473.
27. Yin and J. Wang, ‘‘A model-based approach for text clustering with outlier detection,’’ in Proc. 32nd Int. Conf. Data Eng. (ICDE), May 2016, pp. 625–636
28. Pasillas-Díaz and S. Ratté, ‘‘Bagged subspaces for unsupervised outlier detection,’’ Int. J. Comput. Intell., vol. 33, no. 3, pp. 507–523, Aug. 2017.
29. Aggarwal and S. Sathe, ‘‘Theoretical foundations and algorithms for outlier ensembles,’’ ACM SIGKDD Explore. Newslett., vol. 17, no. 1, pp. 24–47, Jun. 2015.
30. Rayana and L. Akoglu, ‘‘Less is more: Building selective outlier ensembles,’’ ACM Trans. Knowl. Discovery Data, vol. 10, no. 4, pp. 1–33, Jul. 2016.
31. Balasubramanian, S., Devarajan, H. R., Raparthi, M., Dodda, S. B., Maruthi, S., & Adnyana, I. M. D. M. (2023). Ethical Considerations in AI-assisted Decision Making for End-of-Life Care in Healthcare. PowerTech Journal, 47(4), 168. https://doi.org/10.52783/pst.168
32. Boukerche, A.; Zheng, L.; Alfandi, O. Outlier Detection: Methods, Models, and Classification. ACM Comput.Surv. 2020, 53, 1–37.
33. Ramírez-Gallego, S.; Krawczyk, B.; García, S.; Wo ´zniak, M.; Herrera, F. A survey on data preprocessing for data stream mining: Current status and future directions. Neurocomputing 2017, 239, 39–57