Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Software Department

Data Stream Mining Using an Evolving Intelligent System

A Dissertation

Submitted to the Council of the College of Information Technology, University of Babylon in Partial Fulfillment of the Requirements for the Degree of Doctorate of Philosophy in Information Technology

By

Hussein AbdulAmeer Abbas Fahad

Supervised by

Prof. Dr. Nabeel Hashem Kaghad Jar-allah

Prof. Dr. Eman Salih Sagban Nasir

2022 A.D. **1444 A.H.**

بسم الله الرحمن الرحيم

﴿هُوَ الَّذِي يُرْسِلُ الرِّيَاحَ بُشْرًا بَيْنَ يَدَيْ رَحْمَتِهِ حَتَّىٰ إِذَا أَقَلَّتْ سَحَابًا ثِقَالًا سُقْنَاهُ لِبَلَدٍ مَّيِّتٍ فَأَنزَلْنَا بِهِ الْمَاءَ فَأَخْرَجْنَا بِهِ مِن كُلِّ الثَّمَرَاتِ كَذَٰلِكَ نُخْرِجُ الْمَوْتَىٰ لَعَلَّكُمْ تَذَكَّرُونَ ﴾

صدق الله العلي العظيم

سورة الأعراف / آية ٥٧

Supervisor Certification

I certify that this dissertation was prepared under my supervision at the Department of Software / Collage of Information Technology / Babylon University, by **Hussein AbdulAmeer Abbas** as a partial fulfillment of the requirements for the degree of **Ph.D. in Information Technology**.

Signature:
Name: **Dr. Nabeel H. Al-A'araji**
Title: **Professor**
Date: / / 2022

Signature:
Name: **Dr. Eman Salih Al-Shamery**
Title: **Professor**
Date: / / 2022

The Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled " **Data Stream Mining Using an Evolving Intelligent System** " for debate by the examination committee.

Signature:
Name: **Dr. Ahmed Saleem Abbas**
Title: **Professor**
Date: / / 2022

Certification of the Examination Committee

We, the undersigned, certify that (**Hussein AbdulAmeer Abbas**) candidate for the degree of Doctor of Philosophy in Information Technology-Software, has presented his dissertation of the following title (**Data Stream Mining Using an Evolving Intelligent System**) as it appears on the title page and front cover of the dissertation that the said dissertation is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on: Oct 27, 2022.

Signature: **Name: Dr. Tawfiq A. Al-Assadi** **Title: Professor** **Date: / / 2022** **(Chairman)**	**Signature:** **Name: Dr. Mohammed Abdullah Naser** **Title: Professor** **Date: / / 2022** **(Member)**
Signature: **Name: Dr. Hasanen S. Abdullah** **Title: Asst. Professor** **Date: / / 2022** **(Member)**	**Signature:** **Name: Dr. Noor Dhia Kadhem** **Title: Asst. Professor** **Date: / / 2022** **(Member)**
Signature: **Name: Dr. Sura Zaki AlRashid** **Title: Asst. Professor** **Date: / / 2022** **(Member)**	**Signature:** **Name: Dr. Nabeel H. Al-A'araji** **Title: Professor** **Date: / / 2022** **(Member / First Supervisor)**
Signature: **Name: Dr. Eman S. Al-Shamery** **Title: Professor** **Date: / / 2022** **(Member / Second Supervisor)**	

Signature:
Name: Prof. Dr. Hussein Atiya Lafta
Title: Professor
Date: / / 2022
(Dean of Collage of Information Technology)

Declaration

I hereby declare that this thesis, **Data Stream Mining Using an Evolving Intelligent System** submitted to University of Babylon in partial fulfillment of requirements for the degree of Doctorate of Philosophy in Information Technology-Software has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for reports and summaries whose sources are appropriately cited in the references.

Signature:

Name: **Hussein AbdulAmeer Abbas Al-Khamees**

Date: / / 2022

Acknowledgements

It is with great honor that I express my thanks and gratitude to the God Almighty, who gave me health and determination to complete this project successfully. The utmost respect to the Prophet of mercy, Muhammad Ibn Abdullah and to his cousin, the hero of Islam, Al-Imam Ali Ibn Abi Talib.

I would like to express my greatest gratitude and major thanks to my supervisors, **Dr. Nabeel H. Al-A'araji** and **Dr. Eman Salih Al-Shamery,** for their guidance and continuous follow-up to work and overcoming all difficulties.

All the words cannot express my gratitude and deepest thanks to my mother, brother, wife and three sons for their patience and constant encouragement during my study years.

I'm extremely grateful to the management of the College of Information Technology, especially the software department, from the head of the department, professors, and all staff for support and assistance.

I would be remiss not to mention my loyal friends and all the people who encouraged me and helped me to complete this work.

Hussein AbdulAmeer Abbas

Dedication

I dedicate this dissertation to my role models, the Prophet Muhammad (may God bless him and his family and grant them peace) and Imam Ali (peace be upon him).

To my Father …

First and foremost, to his soul, who I haven't seen up close a lot. I will never forget you.

To my mother …

The source of my inspiration who loves me infinitely.

To my brother …

Who taught me the value of hard work and he did not skimp to help me.

To my wife …

The source of happiness in my life who encouraged me to pursue my dreams.

To my children …

My roses and hope in my life.

Hussein AbdulAmeer Abbas

Abstract

Most of real world applications are able to generate the data stream. Data streams have unique characteristics and different issues as well as it changes over time. The evolving system is presented as a convenient solution to overcome the issues of stream mining due to the ability to change its structure according to change of data behavior. Therefore, this dissertation presents an evolving intelligent system that based on the clustering technique to solve some problems of the stream mining.

In the proposed system, the evolving Cauchy (e-Cauchy) algorithm is applied. Despite this algorithm is a successful algorithm for data stream clustering, but suffers from the large number of generated clusters therefore, this algorithm is developed to overcome this limitation. Moreover, the general clustering problem is the evaluation, which is solved by using deep learning method, that means it is used as an external evaluator. However, this method increases the accuracy of the network and ensures its stability by training the network with a non-constant (dynamic) learning rate value. Accordingly, the proposed system consists of two models, deep learning model (off line mode) and evolving clustering model (online mode).

This proposed system aims to make a decision based fuzzy rules that changes over time to assign the data stream sample to a cluster accurately. In addition, the proposed system is capable to detect new data samples (that have not been previously trained) and then, create a new cluster.

The number of stream datasets used to test the proposed system is eleven datasets that generated from different applications such as, sensor, monitoring systems, devices and diverse domains like, questionnaire, behavior recognition and human activity recognition. Moreover, these datasets have different environments, complexity, number of features and classes.

The evaluation measurements for this system are, accuracy, precision, recall, F1-score and the silhouette coefficient. The accuracy, precision, recall and F1_score ranged from (88.75%, 88.74%, 88.75%, 88.74%) to (100.0%, 100.0%, 100.0%, 100.0%). While the improvement rate for the silhouette coefficient from 0.28 to 0.99 among all datasets.

Table of Contents

List of Algorithms

List of Figures

List of Tables

List of Abbreviations

Abbreviation	**Meaning**
AI	Artificial Intelligence
ANN	Artificial Neural Network
CM	Confusion Matrix
DL	Deep Learning
DLR	Deep Learning Rate
DNN	Deep Neural Network
e-Cauchy	Evolving Cauchy
EIS	Evolving Intelligent System
ES	Evolving System
Final ev	Final Evolving
FHQCs	Final High Quality Clusters

Abbreviation	**Meaning**
FN	False Negative
FP	False Positive
IS	Intelligent Systems
Iter ev_1	First Iteration of Evolving
Iter ev_2	Second Iteration of Evolving
Iter ev_3	Third Iteration of Evolving
HQCs	High Quality Clusters
LQCs	Low Quality Clusters
ML	Machine Learning
MLP	Multilayer Perceptron
NMI	Normalized Mutual Information
NoC	Number of Cluster
TN	True Negative
TP	True Positive
UV_1	Unit $Value_1$
UV_2	Unit $Value_2$
UV_3	Unit $Value_3$

List of Dissertation Related Publications

I. Published Articles

1- **Paper Title:** Survey: Clustering Techniques of Data Stream

Conference name: 1st. Babylon International Conference on Information Technology and Science (BICITS 2021)- Babil- IRAQ.

Authors: Hussein A. A. Al-Khamees, Nabeel Al-A'araji, Eman S. Al-Shamery

Paper details: p.p:113-119, Year 2021

https://doi.org/10.1109/BICITS51482.2021.9509923

2- **Paper Title**: Data Stream Clustering Using Fuzzy-based Evolving Cauchy Algorithm

Journal name: International Journal of Intelligent Engineering and Systems

Authors: Hussein A. A. Al-Khamees, Nabeel Al-A'araji, Eman S. Al-Shamery

Paper details: vol 14, No 5, p.p: 348-358, Year 2021

Journal details: Scopus: Q3, CiteScore 2.0

https://doi.org/10.22266/ijies2021.1031.31

3- **Paper Title:** Classifying the Human Activities of Sensor Data Using Deep Neural Network

Conference name: Intelligent Systems and Pattern Recognition, Second International Conference, ISPR 2022, Hammamet, Tunisia.

Authors: Hussein A. A. Al-Khamees, Nabeel Al-A'araji, Eman S. Al-Shamery

Paper details: p.p:107-118, Year 2022

https://doi.org/10.1007/978-3-031-08277-1 9

4- **Paper Title:** An Evolving Fuzzy Model to Determine an Optimal Number of Data Stream Clusters

Journal name: International Journal of Fuzzy Logic and Intelligent Systems

Authors: Hussein A. A. Al-Khamees, Nabeel Al-A'araji, Eman S. Al-Shamery

Journal details: Scopus: Q3, CiteScore 2.7

http://doi.org/10.5391/IJFIS.2022.22.3.267

II. Final Submission – Under Printing Articles

1- **Paper Title:** Enhancing the Stability of the Deep Neural Network Using a Non-Constant Learning Rate for Data Stream

Journal name: International Journal of Electrical and Computer Engineering

Authors: Hussein A. A. Al-Khamees, Nabeel Al-A'araji, Eman S. Al-Shamery

Journal details: Scopus: Q2, CiteScore 3.2

2- **Paper Title:** Data Stream: Statistics, Challenges, Concept Drift Detector Methods, Applications and Datasets

Journal name: International Journal of Computing and Digital Systems

Authors: Hussein A. A. Al-Khamees, Nabeel Al-A'araji, Eman S. Al-Shamery

Journal details: Scopus: Q4, CiteScore 1.1

III. Under Reviewing Articles

1- **Paper Title:** Human Activities Classification from Sensor Data Using Deep Learning

Journal name: SN Computer Science

Authors: Hussein A. A. Al-Khamees, Nabeel Al-A'araji, Eman S. Al-Shamery

Journal details: Clarivate journal

Chapter One

General Introduction

General Introduction

1.1 Introduction

Different real world applications are able to generate data continuously at a high speed, which is called a data stream. These applications cover many domains with various environment, conditions, size, challenges and other characteristics. The data stream has unique characteristics that differ from the traditional data (Bahri et al., 2021).

The real world applications include most sectors of society such as, health care, economy, energy, transportation, social networks, weather forecasting, and so on (Halstead et al., 2021).

The data stream needs to be processed in real time because offline processing of it causes mainly delayed analysis. The stream mining is the step of extracting and finding meaningful information from this huge data (Zubaroglu & Atalay, 2022).

The major challenge in stream mining is the evolving (dynamic) of the data since, the stream environment is non-stationary. Therefore, the traditional algorithms are not efficient for data stream (Din et al., 2020). Various tasks of data stream mining can be applied, the most widely used and important are, classification and clustering (Din et al., 2021).

Actually, the intelligent systems (ISs) are major and essential class of information computing systems that are applied in a variety of fields. These systems can solve problems of different types of complexity (Frolov et al., 2021). Moreover, ISs are able to simulate some aspects of intelligence behavior such as, learning, adaptability, improving efficiency, and others. These systems provide a solution by taking an intelligent decision (or action) at a certain time (Schmucker et al., 2021).

Within the context of systems, the evolving systems (ESs) represent one of the most important solutions for handling data streams (Mollá et al., 2022).

One of the most important characteristics of ES is its ability to change the general structure of the model designed to describe the data stream. The general idea behind the ESs is the training step and how this step can update continuously (Bahri & Bifet, 2021).

The machine learning (ML) is a major branch of artificial intelligence (AI) that consists of different techniques. Usually, ISs and ESs are built through one or more ML techniques (Injadat et al., 2021).

The evolving intelligent systems (EISs) are robust systems which can update and adapt their structure and also parameters depending on the data nature. Accordingly, EIS can present a good solution to overcome the stream mining issues. Indeed, this is due to the ability to change its structure according to the changing in the data behavior, specifically in the non-stationary environment (Khamassi et al., 2018).

Most of real-world applications in different domains can generate massive amount of data; It is known as a data stream (Mehmood et al., 2021). The data stream applications cover many life action daily, for example, health care, economy, different monitoring systems, energy, transportation, social networks, weather forecasting, financial transactions, and many others that generated by different devices such as, sensors, satellite, smartphones, monitoring systems and others (Kolajo et al., 2019).

The data stream has various unique properties that traditional data do not have (Bahri et al., 2021). Some of these characteristics are: boundless data size, continuous and rapid access, on-line arriving, limited memory and processing time, single processing, and the most important feature, the nature of this data is not static but evolve over time i.e. data stream environments is a non-stationary (L. Lu & Zhou, 2021).

Batch learning is a classic method for ML models. However, these models are built in an offline mode and all dataset samples are available. Therefore, these models are not updated absolutely. In contrast, the models that process the data stream are built in an online mode and the dataset samples are arriving continuously. Thus, these models are characterized by the possibility of updating its structure (Souza et al., 2020).

As a result, most traditional data algorithms fail when dealing with the data stream, this is due to newly characteristics of this data type as well as the updating step (Weiss et al., 2021).

Therefore, there is a real need to design systems that can change their structure, adopt these changes as well as evolve depending on the nature of the receiving data (that matching the evolving nature of the data stream).

ESs have to change its structure for describing the behavior of the data stream as well as able to update and adapt parameters (Ojha et al., 2019). ESs can be defined as intelligent systems can mainly learn their structure and parameters simultaneously using the data (Tavakoli et al., 2021).

From the point of view of system types, ESs can classify as data-driven systems, that are automatically adapted and dynamically evolved based on the new data samples (Leite et al., 2020).

EIS is an important type of the ISs. However, EISs classified as robustly systems which can adapt (and also update their parameters and structure) over time. Furthermore, EISs can be applied in a non-stationary environment (Gu & Angelov, 2019). The EISs are widely used for the data stream and the fuzzy rule-based architectures (Angelov & Kasabov, 2006).

In other words, during the implementation of these systems, their structure as well as parameters do not remain constant throughout the implementation period, but rather varying over time (Luo & Schuur, 2020).

The most challenging characteristic of ESs is their ability to implement the evolving mechanisms (such as, adding, splitting, merging..., etc.) constantly of the units they represent. These mechanisms are applied in order to keep the change in system structure during the progress of those systems implementation (continuous access of data stream samples) (Škrjanc et al., 2019).

Model adaptation is an important and even key criterion when the model processes the data stream. The adaptation strategy expresses the flexibility of the model due to its ability to change the model structure (Abdallah et al., 2018).

The applying of the evolving mechanisms represents the core of ESs to implement the adaptation strategy. These mechanisms are always associated with system units (Ojha et al., 2019). As long as the proposed system is based on the clustering technique, the above mechanisms will synchronize their work with the clusters. Consequently, adding, splitting, and merging clusters are applied as evolving mechanisms in this system.

Moreover, the fuzzy rules can be included in diverse systems such as, EIS. These rules strike a good balance between model accuracy and interpretability (human-readable rules). Thus, will provide a better understanding of the incoming stream data samples (Cano & Krawczyk, 2019).

1.2 Problem Statement

The significant challenge of the data stream is the nature of this data is not static (it evolves over time) this is due to the non-stationary environment (Din et al., 2021). Therefore, the stream mining and analysis is a daunting task (Bahri et al., 2021).

Different systems have been previously proposed to address the data stream, but without handling the system's detection of a new class

The number of streaming datasets to test the proposed system is eleven, there is a difference in the number of features and classes for each one. Furthermore, the behavior of the data stream is unknown, and therefore new data may appear for which a similar behavior is not trained before.

1.3 Challenges of the Dissertation

There are a number of different challenges including:

- The evolving nature of the data stream which also known as the behavioral fluctuations in the stream mining community. This dynamic nature considers the most prominent feature of the data stream. More deeply, while the system is trained the data samples, suddenly an unknown sample appears that is the system has not trained it.
- The proposed system handles several types of data streams, each of them has a specific behavior that differs from the behavior of others thus, the results will reflect the constant performance of the model. Accordingly, a threshold value is specified to every stream dataset.
- Most stream clustering algorithms suffer from generating a large number of clusters (NoC) during execution.
- The other challenge for most data stream clustering algorithms, the generated clusters contain a number of misclassifications data samples.

1.4 Aims of the Dissertation

The main aim of the proposed evolving intelligent system is to make a decision based on fuzzy rules that change over time to assign a data stream sample accurately whether this sample trained through the system or not. Moreover, some other objectives are as follows:

- Building an evolving intelligent system capable of assigning whatever data sample accurately. Thus, the system creates a new cluster when it detects new data samples. Therefore, the system improves the results accuracy.
- Building an evolving intelligent system that is able to change its structure and behavior during the implementation to handle the change of data behavior.
- Building an evolving intelligent system that is generalizable. More deeply, for each stream dataset, a specific threshold and rules are defined thus, the system able to apply in online mode.

1.5 Contributions of the Dissertation

The major contributions of this dissertation are as follows:

- Developing a DL method to find a dynamic learning rate (DLR) value, and training the network with that value. More specifically, this dynamic value is not constant, that impacts positively on the network stability and updates the weights.
- Developing the e-Cauchy data stream clustering algorithm to be able to solve its main problem which is, the large NoC.
- The proposed evolving intelligent system detects any untrained stream data sample and then, create a new cluster to it. In other words, this system able to assign a stream data sample to the suitable cluster accurately whether this sample is trained or not.

1.6 Related Works

This section discusses the latest methods of stream mining, which usually applied to the stream datasets.

evoStream model was suggested by (Carnein & Trautmann, 2018) to handle the fast arriving of the data stream samples. The core of this model is the applying of the evolutionary optimization algorithm for idle times by isolation of dense areas in the data space. This model took an advantage of time to build clusters. One of the objectives of evoStream is to refine the clusters created during implementation gradually. evoStream model suffers when process the stream dataset that has high dimensions.

A study by (Fahy et al., 2018) designed a framework for clustering the data stream samples in online fashion is called as, ant colony stream clustering (ACSC). Firstly, it detects the micro-clusters by a density-based clustering method and implementing the stochastic approach to find approximate clusters. Then, refine these clusters by applying the ant colony optimization. This method suffers from a computational problem due to the implementation of the optimization method within clusters.

Just as the data stream evolves over time, so do the features of that data. Accordingly, xStream is an algorithm proposed by the authors in (Manzoor et al., 2018) to process the features-evolving of the data stream over time. The core of this work is to detect the outliers by implementing the random projection scheme on the streaming features. However, this algorithm classified as a density-based method which has a constant space and time, in addition to being able to process high dimensional stream datasets. Despite the accurate performance of the xStream algorithm, it lacks in several aspects, the most notable of which is the lack of discussion of how the clusters are evolved, especially since it assumes stability in space. Various stream datasets were used to evaluate xStream algorithm but some of them have less than 550 data sample.

According to (Skrjanc et al., 2018), the authors suggested the evolving Cauchy (e-Cauchy) clustering algorithm to monitor the cyber-attacks. Its main feature is the ability to adjust in a real time environment due to applying the evolving mechanisms (adding, splitting, merging, and deleting clusters). This algorithm has proven to be effective in detecting cyber-attacks at a cost and less time than traditional methods. Moreover, this algorithm used the transformation method stochastic neighbor embedding (SNE) for visualizing the high dimensional stream data in two dimensional maps. This study suffers from; the algorithm was tested using one stream dataset that is KDDCUP1999 intrusion detection and, the number of generating clusters is large.

The authors in (Fahy & Yang, 2019) proposed a method to track the change in the data stream, this method is known as Multi-Density Stream Clustering (MDSC). The clusters in this method (for buffer) are detected by using the ant-inspired swarm intelligence. While in the online step, a neighborhood threshold value is set to every constructed cluster in a step for detecting the multi-density clusters. The disadvantage of this method is its inability to track rapid changes of the clusters.

A density-based approach was designed by (Islam et al., 2019). This approach is known as a buffer-based online clustering for evolving data stream (BOCEDS). It consists of two phases, micro-clusters and macro-clusters. Firstly, it constructed the macro-clusters in the high dense areas. However, the information in the micro-clusters is used repeatedly for updating its radius parameter (adaptive radius). In addition, this approach utilized an energy function depended on the spatial information of the data stream. The BOCEDS approach favors reduced computation over memory consumption.

The study of (Sarfraz et al., 2019) presented a First Integer Neighbor for Clustering Hierarchy (FINCH) method. This method is a hierarchical agglomerative approach that proved the first neighbor of every data sample, is the most influential factor for detecting clusters from the given dataset. Therefore, it is a method based on the relationships among the data samples. Despite its importance and successful in different domains, but it required many search methods as well as its not trainable.

An online-offline grid-based clustering method presented by (Ahmed et al., 2020) that is known as (DGStream) for processed the data stream. DBSCAN algorithm was used in this method for determining a macro-cluster (therefore alike DenStream clustering method). Furthermore, the using of DBSCAN improving the implementation time as well as to increase the speed for the proposed method. Essentially, this method applied the delete mechanism when the grids become sparse, especially, if the dense grids are constructed. However, DGStream method required the NoC before the execution.

An AutoCloud algorithm for clustering the data stream to detect the anomaly tasks was suggested by (Gomes Bezerra et al., 2020) that depended on the data analytics of the typicality and also the eccentricity. Its most prominent feature, it did not need the processing in offline manner, thus, it can be worked in the online manner only. The evolving mechanisms for this algorithm are, the creation, and merging clusters which depend on the evolving behavior of the data stream and the data clouds are the granular units used in this method. Most of the processed stream datasets for this algorithm were synthetic datasets.

The authors in (Al-Khamees et al., 2021), presented an improvement e-Cauchy stream clustering algorithm was proposed to cluster the data samples where this algorithm contains two thresholds. Moreover, the evolving mechanisms were applied to cope with the non-stationary environment of the data stream. Two different stream datasets were used that are; NSL-KDD and keystroke datasets. However, in this method, the effective threshold differs in each stream dataset. After normalizing all datasets, these data split into training and testing data to test the proposed method. The results proved the effectiveness of the difference in the threshold assigned to each dataset with the applying of the evolving mechanisms. This improvement method tested only two stream datasets.

In (Challa et al., 2022), the authors proposed a method for clustering the arriving data stream at any time the user request it, this method is known as ANYCLUS. This method was able to produce micro-clusters that have a high quality. Moreover, one of the issues addressed in this method is the difference in the arrival rate of data stream samples. ANYCLUS method consists of online and offline phases. In the first phase, the method summarized the arriving samples as micro-clusters through the hierarchical structure. While in the second phase, the leaf level of the micro-clusters was processed by the logarithmic method (Tilted-Time Window Framework (TTWF)). Different stream datasets were used such as forest cover, KDDCUP 1999, and also synthetic dataset.

Improved version of the FINCH method is called Streaming FINCH (S-FINCH) which was suggested by (Cunningham et al., 2022). S-FINCH classified as a cluster-graph method and contained three steps, firstly, updating all cluster graphs that constructed by FINCH method and then, determining the nearest neighbor to the new instance. Secondly, applying a local search of the cluster graph for updating the cluster labels. Lastly, hierarchically updating the clusters. S-FINCH has reduced time complexity when compared to the FINCH method. The cluster quality metric for this method is the normalized mutual information (NMI) which scored degraded results for the MNIST dataset due to the use of the actual representatives of the clusters.

Table 1.1 explains briefly the summary of the related works as well as, the technique used in each work, dataset, evaluation measure and processing manner.

Table 1.1: Summary of the related works

Seq.	**Ref.**	**Dataset No. / Names**	**Technique**	**Evaluation Measure**	**Processing Manner**
1.	(Carnein & Trautmann, 2018)	4 / Powersupply, Sensor, KDDCup'99, Covertype	Evolutionary algorithm	Sum of Squares (SSQ), Rand Index (RI), Silhouette Coefficient (SC), purity, precision, recall, F1-measure	Online
2.	(Fahy et al., 2018)	6 / Network intrusion, Forest cover, 1CDT, 2CHT, 4CR, 4CE1CF	Density-based method	Purity, F1-measure, RI	online

3.	(Manzoor et al., 2018)	8/ Gisette, isolet, letter, madelon, cancer, ionosphere, telescope, Indians	Density-based method	Precision	Offline
4.	(Skrjanc et al., 2018)	1/ KDDCUP1999	Density-based method	NoC, accuracy	Online
5.	(Fahy & Yang, 2019)	6 / Network, Forest cover, Keystroke, COIL, Air Quality, 2CSurr	Density based-method and swarm intelligence	Purity, F1-score, RI	Online
6.	(Islam et al., 2019)	4 / Mackey-Glass, Helical, KDDCUP'99, atmospheric	Density-based method and graph-based method	Accuracy, purity	Online
7.	(Sarfraz et al., 2019)	6/ Mice Protein, REUTERS, STL10, BBTs01, BFs05, MNIST	Hierarchical agglomerative methods	NMI, accuracy	Offline
8.	(Ahmed et al., 2020)	5/ KDDCup'99, Cover Type, Adult, NSE Stocks, Chameleon	Grid-based method	F1-score, purity, precision, recall	Offline - online
9.	(Gomes Bezerra et al., 2020)	8/ S1, S2, A1, A2, dim512, dim1024, Aggregation, Compound	Data analytics of typicality and, Cloud concept	NoC, accuracy	Online

10.	(Al-Khamees et al., 2021)	NSL-KDD, keystroke	Density-based method	SC, NoC	Offline
11.	(Challa et al., 2022)	7/ FOREST cover, KDDCUP1999, MPAGD3.2 M, SFONT1M, FOF57M, MPAGD1B, SYTHETIC	R-tree, AnyRTree	Granularity, purity	Offline - online
12.	(Cunningham et al., 2022)	4/ Aggregation, Gestalt, MNIST, CIFAR-10,	Graph-based method	NMI	Offline

On the other hand, different previous ML models have been proposed that used as classifiers to process the streaming datasets. Table 1.2 describes with details some of these models.

Table 1.2: The structure of the previous ML models as classifiers to process the stream datasets

Item	**Ref.**	**Dataset**	**MLP**	**Structure**	**Accuracy**
1.	(Andrean et al., 2020)	Keystroke	√	2 hidden layers, 23 neuron	91.67%
2.	(Javed et al., 2020)	Keystroke	√	------	89.0%
3.	(ABBASI et al., 2021)	Electricity	√	100 hidden layers	81.06%
4.	(Seedat et al., 2022)	Electricity	√	2 hidden layers	82.7%
5.	(Lobo et al., 2020)	Electricity	√	------	57.97%
6.	(Abrar et al., 2020)	NSL-KDD	√	1 hidden layers, 100 neuron	98.19%
7.	(Kyatham et al., 2020)	NSL-KDD	√	Hybrid model of MLP and kNN	97.05%

8.	(Ahanger et al., 2021)	NSL-KDD	√	2 hidden layers, 10 neuron	97.97%
9.	(Al-khamees et al., 2022)	HuGaDB	√	3 hidden layers, 100,75,50 neuron	91.7%
10.	(Javeed et al., 2021)	HuGaDB	√	A hybrid model	92.5%
11.	(Kumari et al., 2020)	HuGaDB	X	Long short-term memory, 3 hidden layers	91.1%
12.	(Bozkurt, 2021)	UCI-HAR	√	3 hidden layers, 48, 24, 12 neuron	96.81%
13.	(Ghate & C, 2021)	UCI-HAR	√	Hybrid models 3 hidden layers, 100 neuron	90.0%
14.	(Wan et al., 2020)	UCI-HAR	√	2 hidden layers, 256 neuron	86.83%
15.	(Myo et al., 2019)	UCI-HAR	√	3 hidden layers, 371 neuron	98.32%
16.	(S. Wang & Zhu, 2020)	UCI-HAR	√	------	93.72%
17.	(Pinagé et al., 2020)	Luxembourg	X	Hybrid models	97.53%

1.7 Dissertation Outline

The remainder of the dissertation is organized as follows:

- **Chapter Two:** discusses the basic principles of what are used in the proposed system that are; the evolving intelligent systems, the evolving mechanisms, the data stream (with different aspects), ML, DL, clustering techniques, the fuzzy logic, evaluation measures of each technique, and the stream datasets.

- **Chapter Three:** illustrates the major steps of the proposed system and how each model is built, as well as its algorithms.
- **Chapter Four:** summarizes the experimental results of the proposed system.
- **Chapter Five:** reviews the conclusions of this dissertation, besides, the suggestions for the future work.

Chapter Two

Theoretical Background

Theoretical Background

2.1 Introduction

This chapter gives a background about the theoretical concepts, which used in the proposed system. However, the proposed evolving intelligent system consists of two models, where each model is based on a specific ML technique. Thus, two ML techniques are used that are, DL and clustering. In addition, this chapter explains the evolving concept, the intelligent system, the data stream, ML techniques, the evolving mechanisms, the data stream and the streaming datasets, the fuzzy logic and finally the evaluation measurements.

2.2 Evolving System

The traditional models usually fail when they deal with the streaming data and this is due to the challenges of this type of data. The ESs represent one of the important and attractive solutions for dealing with data streams (Michelon, 2020).

According to the main property of ES, the major task and key feature of any ESs is associated with several mechanisms, e.g. adding, splitting, merging and removing of the entities (Leite et al., 2020).

Thus, this will ensure a greater flexibility when changing the situations of the system and non-stationary environments if the data evolve over time. In traditional models, the data distribution is assumed to be stationary besides, the structure of the models and their parameters do not change over time. In the ESs, the data environment is non-stationary; therefore, both structure and parameters of the models change over time accordingly, it is classified as a dynamic state (Tong et al., 2021).

In these cases, the model is more related to non-stationary environments that can generate a continuous stream of data where the data distribution will not remain the same (Baek et al., 2020).

The online analytics may implement through the methods of evolving identification. The evolving identification approaches allow us to adapt structure and parameters simultaneously (Škrjanc et al., 2018).

Furthermore, the system can describe the various behaviors through the evolving of the model structure and also identifying the parameters online (Souza et al., 2020).

Indeed, the ESs can classify as data-driven systems, that are (Leite et al., 2020) :

1- Automatically adapted and extended.
2- Dynamically evolved based on the new data samples.

The IS can be defined as a system has the ability for taking intelligent decisions (actions) depending on input or perception (Angelov et al., 2010).

EISs are robustly adaptive systems which can update and adapt their structure and parameters depended on the data stream. Furthermore, these systems overcoming many problems for different techniques, such as, classification, prediction and data processing in a nonstationary, dynamic environment. Therefore, EISs models are self-developed and self-learning from the data stream. As long as these systems adapted its structure and learning, the term “evolving” is used for distinguishing it from adaptive systems. In other word, term “evolving” to specify a higher level adaptation (Faris et al., 2019).

Undoubtedly, EISs are now simulating reality more than ever, benefit from the data stream learning and gradually evolving to be more accurate and better in performance. In all, the EISs can deal with many problems as well as provide timely solutions (Ojha et al., 2019).

2.3 Data Stream

Many sources able to generate the data stream. Some of these sources are, sensors, satellite data, surveillance and tracking, network traffic, stock market, retail and so on. Meanwhile, many of these sources include different sectors of daily life, especially the daily ones, such as, the health care, economy, computer, environment, and others (Al-Khamees et al., 2022).

2.3.1 Data Stream Concept

The data stream can be defined as a sequence of a boundless amount of data arriving online at a high speed from the source causing limited storage and memory. The main characteristic of this data, it can be evolved over time (Baek et al., 2020).

Data stream has many several challenges that can be summarized as follows (Bahri et al., 2021), (Baek et al., 2020), (Aguiar et al., 2022):

1. Data stream samples boundless (endless arriving of data) and their samples arrive continuously and sequentially, this vital and important aspect leads to generate a number of challenges to the data stream that are:
   - Bounded memory (limited storage size).
   - The whole data cannot be stored never, hence, techniques of the data reduction are implemented such as windowing, sampling, synopsis, ..., etc.
   - In the processing step, a single pass is allowed to every sample, then, it can be removed or stored if needed (because of the high costs of storage devices).
2. Fast online access: the data stream sources able to generate the data stream consistently and send its data speedy. Moreover, these data samples are appearing sequentially (over time) and the method of arriving data cannot control. Accordingly, the challenges of data stream are:
   - Data stream samples are arriving at high speed from sources.
   - There are not a large variety of data pre-processing methods available.
3. Data stream techniques: many techniques for data stream can be implemented depending on the type of problem to be solved and the algorithms applied in a specific model. Most of these techniques are linear and sublinear that

usually implemented in real time. Accordingly, the challenges for data streams are:

- The response of the model should be immediately (providing the user with results at any time it requests). Simply, it is promptly real-time data analytics.
- Running time is very crucial since data samples must be processed as quickly as possible, otherwise the algorithm will be ineffective.

4. The data stream has unexpected properties besides, in general, it classifies as heterogeneous data. This state of heterogeneity resulted from the heterogeneous sources. Furthermore, as long as there is diversity in the data stream applications implemented in a variety of disciplines, the system encounters an uncertain data and also the presence of a specific class label more than the other. Therefore, the challenges are:
   - High dimensional data
   - Imbalanced labels of the data samples.
5. The nature of the data stream: unlike traditional data, where the data is static, the data stream is a dynamic that is evolving over time. This may be due to the non-stationary of the data stream environment. The interesting issue is that this dynamism and non-stationary is causing one of the most important data stream challenges and the most studied, analyzed and designed models to detect, that is the concept drift. Concept drift occurs because of the changing of the data distribution over time. This appearance causes the performance of the model to deteriorate, and the model results will be less accurate (or inaccurate). Wherefore, the algorithm must be modified (updated) to deal with any changes that occur in the data. Hence, the model is automatically updated for adapting the most recent changes. Thus, the challenge is:
   - Concept drift.

According to these challenges, the most important ones that the proposed system deals with are the unexpected properties as well as the nature of the data stream.

2.3.2 Data Stream Comparison with Traditional Data

There are many differences between the data stream on the one hand and traditional data on the other hand. Table 2.1 details these differences (Krawczyk et al., 2017), (Wares et al., 2019):

Table 2.1: The comparison between traditional data and data stream

Item	Traditional data	Stream data
1.	Data samples are exists.	Sequentially appear over time.
2.	Data samples can control.	The way of samples arrive cannot be controlled.
3.	Data size is bounded.	Unbounded in size.
4.	No problem for storing all raw data.	It is not possible to store the entire data, data reduction techniques can use like synopsis, sampling,...etc.
5.	Many pass to every sample during their processing since its size is limited.	A single pass is allowed to every sample, then, it can be removed or stored (if needed).
6.	Data nature is static that is not change.	Data nature is not static, rather it dynamic and evolving over time.
7.	Most of its tasks carry out in an offline mode.	Most tasks are being implemented in online mode.
8.	The system response doesn't have to be momentary.	The response of the system must be an immediately.
9.	The system output results are accurate.	The results of the system approximate.
10.	It has known properties.	It has unexpected properties.
11.	It used complex techniques if necessary (according to need).	The most common techniques are linear and also sublinear
12.	The data is homogeneous.	Heterogeneous data.

2.3.3 Data Stream Mining

Recently, the demand for mining and analysis of streaming data are increased. The secret is to enable the organizations to respond to non-stationary environments (conditions) in real-time (Kolajo et al., 2019).

The mining as well as the analysis of the data stream is an arduous and attractive task to extract knowledge from. This is due to the various challenges of this data (Bahri et al., 2021).

The mining models of the data stream must cope with different aspects of the data stream. Sometimes these aspects known as dimensions. Therefore, there is a need to deal with volume, velocity, variety, veracity and value which are mostly known as V5 (Kaur & Sood, 2017).

The most prominent and most widely used tasks in the stream mining are classification and clustering (Ramírez et al., 2021).

Evidently, the DL frameworks are gaining wide popularity for their use to deal with changes in data behavior during the dynamic classification tasks for the data stream. In fact, it is more suitable for such a dynamic environment since it achieves impressive results (Jameel et al., 2020). Practically, these DL frameworks (classifiers) continuously receive the data stream samples and classify them automatically, this makes it a significant approach to data stream mining and analysis (Jan et al., 2019).

On the other hand, the idea behind the data stream clustering is to group the similar samples in a cluster through implementing clustering algorithms (Din et al., 2021).

2.3.4 Stream Datasets

A large number of these stream datasets are publicly available and free to download can be used in many systems. These datasets are collected from different sources and applications for various domains.

The number of stream datasets used to test the proposed system is eleven datasets that are generated from different applications such as, sensor, monitoring systems, devices and diverse domains such as questionnaire, behavior recognition and human activity recognition. These stream datasets as follows:

1. Electricity: it has been gathered from the Market of Electricity in New South Wales, Australia, therefore, it is affected by demand and supply. The prices are constantly changing, and it should be determined every five minutes. The dataset collected from 1996-1998 and each sample was created during 30 minutes, so it produced 48 samples in a day.
2. Keystroke: it was collected in 2015 from 4 users who typed the same password (. tie5Roanl) and the "Enter" key during a specific period of time through eight sessions. The main feature of this dataset is its ability to evolve according to the behavior of its users. Most studies based on the flight time to distinguish the user.
3. NSL-KDD: the main shortcoming in the KDD dataset, it has a massive number of the redundant records that will effect on the final results of the model. Accordingly, and to overcome imperfections described above, an enhanced version of the original KDD’99 dataset was presented, known as NSL-KDD dataset. In the new dataset, each redundant record is omitted besides, all the records are re-balanced. As a result, the NSL-KDD dataset becomes more realistic and practical for evaluating algorithms. It was collected in 2009.
4. HuGaDB01_01: The general dataset is the Human Gait Database (HuGaDB) is a sensor streaming dataset that was presented in 2017. The main HuGaDB dataset consists of 364 sub-datasets. It was gathered from 18 healthy participants at average age of 23.6 by recording for 10 hours. In total, six wearable inertial sensors were used that placed on the right and left thighs, shins and feet, and 12 diverse activities were included in the all HuGaDB dataset. In the proposed system, four sub-datasets from the main HuGaDB are selected and used to test the system. HuGaDB01_01 is the first sub-dataset.
5. HuGaDB05_12: indicates the second sub-dataset of the main HuGaDB sensor dataset for the proposed system evaluation

6. HuGaDB13_11: it is the third sub-dataset of the HuGaDB to test the evolving intelligent system.
7. HuGaDB14_05: it refers to the last sub-dataset from the main HuGaDB sensor dataset that tests the proposed system.
8. UCI-HAR: it is one of the most widely used datasets in human activity recognition (HAR) research since its publication. UCI-HAR dataset was proposed in 2012 and was collected from 30 participants of different ages (18-48) years with a Samsung Galaxy S2 smartphone placed around the waist. Every participant performed six different activities.
9. Luxembourg: it is built in 2011 by using the European Social Survey data. Every sample has 30 features that reflect the answers to this survey, which have two cases, either high or low internet usage.
10. Keystroke_1: it is the tenth stream dataset to test the proposed system. In fact, this dataset is derived from the first dataset, which is the keystroke dataset, after hiding (masking) all data samples belong to class "3". The new dataset is known as, keystroke_1 to distinguish it from the original keystroke dataset.
11. HuGaDB01_01_1: the last stream dataset to evaluate the proposed evolving intelligent system. As in the previous dataset, this dataset is derived from the fourth stream sensor dataset (HuGaDB01_01) after all data samples belonging to class "2" have been hidden. The new dataset is known as, HuGaDB01_01_1 to differentiate it from the original dataset HuGaDB01_01.

However, Table 2.2 shows the websites for free downloading these datasets.

Table 2.2: The websites to download the stream datasets

Item	Dataset	Website
1.	Electricity	https://sites.google.com/view/uspdsrepository
2.	Keystroke	https://www.sites.google.com/site/nonstationaryarchive/
3.	NSL-KDD	https://www.unb.ca/cic/datasets/nsl.html
4.	HuGaDB	https://www.kaggle.com/romanchereshnev/hugadb-human-gait-database

5.	UCI-HAR	https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones
6.	Luxembourg	https://sites.google.com/site/zliobaite/resources-1

2.4 Machine Learning

Computer science includes many fields, one of the most important and widely used field is the AI. It aims to solve real problems in addition to building systems that have the ability to learn and think like humans (Al-Khamees et al., 2022).

AI includes any technique that enables the computers to simulate human behavior in solving complex functions. Therefore, AI is consisted of many branches, one of them is the ML. It has proven overwhelming success due to the ability to gain knowledge and gradually improve the behavior of learning. Due to its ability to achieve practical successes, ML is considered a backbone of AI (Joshi, 2020).

Firstly, the learning can be defined as any step by which the performance (and thus results) from previous experiences is improved. Therefore, the essential goal of ML models is to build computer programs that able to learn from data (Shyam & Singh, 2021).

During the past years, many ML techniques has been used and developed in different fields. In general, ML models are based on one or more of these techniques which involves the algorithm/s can solve different problems in different domains (Afshan & Rout, 2021).

The ML techniques have proven to be successful and efficient when applied in the mining models even if the data type is a stream (Ramírez et al., 2021).

Classification for the data stream aims to classify the data stream samples depend on the previous classes. Thus, the classification algorithm (classifier) is utilized for classifying the incoming (unseen) data stream samples (M. K. Gupta & Chandra, 2020).

On the other hand, the clustering is applied for the data stream to partition (segment) the given stream datasets into clusters (Chamikara et al., 2018).

Like the traditional data, the main difference between the classification and clustering for the data stream is the classification is a supervised learning while, the clustering is an unsupervised learning (Kokate et al., 2018).

Clustering data streams is considered one of the difficult techniques as it requires the ability to cluster data samples continuously within certain restrictions (M. K. Gupta & Chandra, 2020).

2.4.1 Deep Learning Technique

I. Deep Learning Concept

The DL depends mainly on Artificial Neural Networks (ANNs), which are originally inspired by neurons in the human brain. ANNs has two types, shallow neural networks or deep neural networks (DNNs). The main difference between them is the later uses one hidden layer while the DNNs uses more than one hidden layer in its structure (Vinayakumar et al., 2017).

Any system that falls under the branch of AI and based on a neural network that has multiple hidden layers can be described as DNN or DL system (Joshi, 2020).

One of the attractive features of DL technique is its high representation ability without relying on a great deal of specialized knowledge as well as the capability of learning the unlabeled (unstructured) data (Gao et al., 2019).

Each layer in ANN consists of many units that called neuron. In DNNs, every layer utilizes former layer output as an input (Denuit et al., 2019). Due to this property, the neurons in the layers form the hierarchy. So, when DL first started it is known as hierarchical learning (VARGAS & RUİZ, 2017).

The DL models have been applied in many areas such as, computer, education, medicine, agriculture, manufacturing, chemistry, physics, space sciences, detection processing, identification, and others (Ahmad et al., 2019). In computer science, the models of DL are proposed and implemented in many fields for example, computer vision, information retrieval, natural language processing, image classification, speech recognition, games, entertainment, object detection…, etc (Zhong et al., 2018), (Ahmad et al., 2019).

II. DNN Structure

There are three types of layer in DNNs. Input, hidden, and output layers. The data is received from the external source through the input layer, therefore there is not any processing (computations). Most of the processing steps that are implemented in the hidden layers are nonlinear computations, whereas the processing in the output layer either linear or nonlinear (Karhunen et al., 2015).

However, the structure of DNNs is arranged in multi-layers. Due to the similarity of DNNs with ANNs, every layer in DNNs contains several neurons, take into consideration that the neuron number differs from a layer to another. In a specific layer, every neuron is connected to their counterparts in adjacent layers. This connection can be indicated by weights which reflecting both strength and direction. Every neuron can transform data through computation of weighted sum (of the output neurons in past layer) and then passes it by a nonlinear function (activation functions) for deriving the neuron outputs (Vieira et al., 2020).

Based on the inputs ($x_1, x_2, \dots, x_n$) and its weights ($w_1, w_2, \dots, w_n$), the total net input is computed according to Equation 2.1 (Al-khamees et al., 2022):

$$\boldsymbol{Net} = \sum_{i=1}^{n} \boldsymbol{x_i} * \boldsymbol{w_i} \tag{2.1}$$

Figure 2.1 shows both DNNs and shallow neural networks (Bevilacqua et al., 2019).

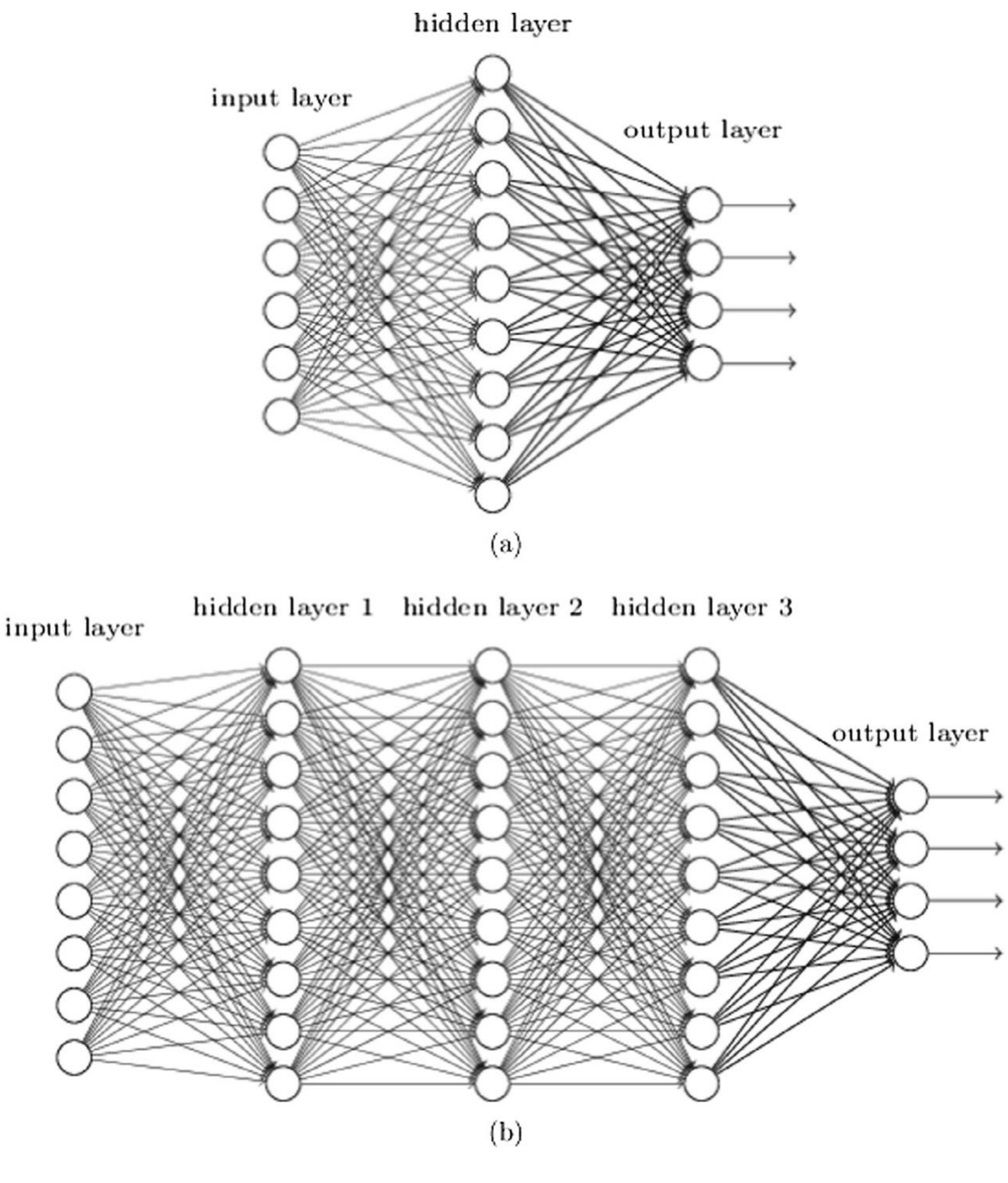


Figure 2.1: The structure of neural networks (a) Shallow neural network, (b) DNN

To simplify, proposed the input in the network is [2, 3] and the initial weights are (Lee et al., 2016):

$w_1 = 0.11$, $w_2 = 0.21$, $w_3 = 0.12$, $w_4 = 0.08$, $w_5 = 0.14$ and finally $w_6 = 0.15$. The results as follows:

$$[2 \quad 3]\begin{bmatrix}0.11 & 0.12\\0.21 & 0.08\end{bmatrix} = [0.85 \quad 0.48] * \begin{bmatrix}0.14\\0.15\end{bmatrix} = [0.191]$$

Accordingly, the matrix multiplication result is 0.191. which comes from:

2*0.11 + 3*0.21=0.85

2*0.12 + 3*0.08=0.48

0.85*0.14 + 0.48*0.15=0.191

III. Multi-layer Perceptrons

Multi-layer perceptrons (MLPs) is one of the most important structures of DL. Its most important feature is the ability to deal with massive data such as, data streams (Moayedi & Mosavi, 2021).

MLP is a feed-forward neural network with multi hidden layers that does not require any prior assumptions about the distribution of data. In MLP, the neurons are connected by weights and also the signals of output that represented as a function of the sum of the inputs to the neuron modified by an activation function (Lyu et al., 2022).

Usually, the training of the DNN is more difficult and complex than the training of the shallow neural network (Vieira et al., 2020).

The training of DNN contains many sequential steps for adjusting the weights between the neurons in the network, in a similar way to the learning of the human brain. But, before the adjustment step, the model must initialize these weights. This initialization is done randomly, where the resulting weights have the ability to (Vieira et al., 2020):

- Maximizing the relationship strength between network input and its output.
- Minimizing the difference of the neural networks (such as an error) between a specific task and its real target (i.e. between the network prediction and its associated target). Usually, the neural network technique aims to minimize this error value.

More specifically, the back propagation (BP) is the most successful and widely used algorithm for MLP training. BP repeatedly can analyze the errors and optimize every value of weight depending on the errors generated by the next layer. Accordingly, this algorithm is used in the current model (Lyu et al., 2022).

To simplify the weight computation, suppose a neural network contains (m) neuron, this neuron is driven by input vector X_n, where n indicates to the time step of the iterative process contains the adjusting step of the input weights $w_{(mi)}$. Therefore, each sample of data passes through the training step of a DNN containing $X_{(n)}$ and its output denoting by $d_{(n)}$.

Then the processing step to $X_{(n)}$, of a neuron (m) is generating an output which is referred by $y_{m(n)}$, and can be computed by Equation 2.2:

$$\boldsymbol{y_{m(n)}} = \text{ʃ} \sum_{i=1}^{j} \boldsymbol{x} * \boldsymbol{y_{m(i)}} \qquad \textbf{(2.2)}$$

Where ʃ indicates to the activation function.

This output is compared with the target output $d_{m(n)}$ which normally is given in a sample. The error $e_{m(n)}$ can compute according to Equation 2.3:

$$\boldsymbol{e_{m(n)}} = (\boldsymbol{d_{m(n)}} - \boldsymbol{y_{m(n)}}) \qquad \textbf{(2.3)}$$

Because the BP capacity, it is a very appropriate method to problems that do not have any relation between the input and output (Khaireddin & Chen, 2021).

IV. Deep Learning

Like many networks, the MLPs include many parameters whose values need to be determined (or adjusted) before the training step, including the learning rate which its value is often constant throughout the model implementation (Westby et al., 2021).

It is clear that the learning rate has a significant impact on the network performance and therefore, it must be determined precisely. In reality, it can control the change amount based on the estimated error every once the parameters of the model can be updated (Khaireddin & Chen, 2021).

Indeed, there is no general method (or a rule) it can be adopted to determine the value of the learning rate, so this value is different from one model to another. In general, the learning rate value starts with a large value such as, 0.1, then try exponentially lower values: 0.01, 0.001, etc. In the proposed system, the value 0.001 is chosen (Park et al., 2020).

The comparison between the large and small learning rate is explained in Table 2.3 (Wu et al., 2019) (Sung et al., 2020).

Table 2.3: The comparison between large and small learning rate value

Item	Large learning rate	Small learning rate
1.	The network is diverging and there may be oscillations.	The network convergence achieves a satisfactory level.
2.	Need fewer training epochs, this is due to quick changes in the updating of the weights. Therefore, it has a fast training.	Need many training epochs, this is due to small and slow changes in the updating of weights. So, the training speed is slow.
3.	Get stuck in the local minimum easily.	It avoids the local minimum.
4.	Requires less time to train.	It takes a lot of time to train.
5.	doesn't improve the generalization.	Can improve the generalization.

In the training step, the updating of the weights can be calculated as in Equation 2.4:

$$\boldsymbol{Nw_i} = \boldsymbol{w_i} - \eta \left(\frac{\partial\, \boldsymbol{e_{m(n)}}}{\partial\, \boldsymbol{w_i}}\right) \qquad \textbf{(2.4)}$$

Where Nw_i is the new weight, w_i is the old weight and η is the learning rate. In this context, most methods adopted a constant value to the learning rate throughout the implementation time.

2.4.2 Clustering Technique

The clustering technique represents one of the most important and most widely used ML techniques in recent times. It aims by different methods to group (cluster) the data samples of a given dataset into clusters where, each one contains many samples that have a high degree of similarity with each other meanwhile, less similar to the data samples in the other clusters (T. Chen et al., 2021).

The environment and behavior of the data stream differ from the traditional data. Therefore, the data stream clustering algorithm should be carefully chosen to be an appropriate method, otherwise it will be a worthless method (Din et al., 2021).

However, the meaning does not differ much when applying clustering to the data stream, taking into account the challenges that characterize the data stream. In fact, due to the characteristics of the data stream, traditional clustering methods cannot be implemented on the data stream (Mehmood et al., 2021).

Generally, there are five types of the clustering methods that can be applied to the data stream (Kokate et al., 2018):

1. Partitioning method.
2. Hierarchical method.
3. Grid-based method.
4. Density-based method.
5. Model-based method.

Actually, all algorithms of density-based method are based on the computation step of the data density to build the clusters. In general, these algorithms need to determine many parameters (Mehmood et al., 2021).

The e-Cauchy is one of the most modern stream clustering algorithm. This algorithm is categorized as a density-based method because, it continuously calculates the density of the data sample as it arrives to construct the clusters (which are typically large number). This algorithm has two thresholds that are,

Θ_{con} and the density threshold Θ_{den}. Indeed, the second threshold Θ_{den} is more influential than the first one (Skrjanc et al., 2018).

However, the density value for a data stream sample (x) is denoted by Den$_{(x)}$ can calculate according to Equation 2.5 (Skrjanc et al., 2018):

$$\boldsymbol{Den}_{(x)} = \left[\frac{\mathbf{1}}{\mathbf{1}+\left(\frac{\mathbf{1}}{\Theta_{cont}^{\mathbf{2}}}\right) * (\boldsymbol{x_i}- \boldsymbol{CC^j})^{\boldsymbol{T}} * (\sum \boldsymbol{j})^{-\mathbf{1}} * (\boldsymbol{x_i}- \boldsymbol{CC^j}) + \left(\frac{\mathbf{1}}{\Theta_{cont}^{\mathbf{2}}} * \frac{\boldsymbol{SC^j}-\mathbf{1}}{\boldsymbol{CC^j}}\right)}\right] \quad \textbf{(2.5)}$$

Where, $\Theta_{con}$is a predefined first threshold, x_i is a data sample, CC^j is the center of cluster j, $\sum j$ is the covariance matrix, and SC^j is the samples in cluster j. The covariance matrix can compute by Equation 2.6 (Leite et al., 2020):

$$\sum \boldsymbol{j} = \frac{\mathbf{1}}{\boldsymbol{SC^j}} \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{SC^j}} \left(\boldsymbol{x_i^j} - \boldsymbol{CC^j}\right) \left(\boldsymbol{x_i^j} - \boldsymbol{CC^j}\right)^{\boldsymbol{T}} \quad \textbf{(2.6)}$$

Where, x_i^j refers to data sample (i) in cluster j.

After computing the density value, it compares with the second threshold Θ_{den} to either construct a new cluster or to append the current data sample to an existing cluster. In both cases, all cluster parameters are updated.

In the same context of the distance measurement, the Euclidean distance indicates to the most widely used method to measure the distance between two points based on Equation 2.7 (Zhang et al., 2022):

$$\boldsymbol{d}(\boldsymbol{x}.\boldsymbol{y}) = \sqrt{\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{n}} (\boldsymbol{x_i} - \boldsymbol{y_i})^{\mathbf{2}}} \quad \textbf{(2.7)}$$

Where d (x, y) is the distance between x (x_1, x_2,…, x_n) and y (y_1, y_2,…, y_n), and n is the space dimension.

2.5 Evolving Mechanisms

The main characteristic of ESs it can change their structure by applying the evolving mechanisms (Škrjanc et al., 2019).

As long as the proposed evolving intelligent system is based on the clustering, the above mechanisms will synchronize their work with the clusters. Therefore, the adding clusters, splitting clusters, and merging clusters are applied in this system.

2.5.1 Adding Mechanism

Different methods are used to add clusters mechanism. Most of these methods are based on distance calculation. In general, this mechanism is highly depended on the algorithm that is being implemented and on the class that belongs (J. Chen et al., 2020). As long as the algorithm adopts in this work is a density-based algorithm, the value of the density will be as a condition for adding clusters.

However, the cluster is added when the next condition is satisfied:

$$\boldsymbol{Den}_{(\boldsymbol{x})} < \Theta_{\boldsymbol{den}}$$

2.5.2 Splitting Mechanism

Various methods have been used to carry out the splitting clusters mechanism. The most common methods are based on calculating the distance between the constructed clusters after it added by adding clusters mechanism and then, sets a threshold for the splitting process (Škrjanc et al., 2019).

2.5.3 Merging Mechanism

Also the merging clusters mechanism is highly depending on the distance computations between the formed clusters as well as determine a threshold (Leite et al., 2020).

More deeply, the splitting and merging clusters mechanisms generally aim to reduce the clusters number.

2.6 Data Preprocessing

The data preprocessing considers an essential step during the knowledge discovery in the data stream. More generally, the main goal of the data preprocessing techniques is to reduce the complexity in the datasets. Moreover, the implementation of the data preprocessing techniques aims to prepare the data to implement a specific task (Ramírez-Gallego et al., 2017).

The data preprocessing includes different techniques such as, the normalization, the integration, the transformation, the cleaning and others (Alasadi & Bhaya, 2017).

2.6.1 Normalization Technique

Normalization techniques transform the dataset in such a way as to ensure that they will be in the same scale (range) thus is reflected in improved performance and stability of the model. The normalization techniques include, Min-Max, decimal scaling, Z-score and others (Obaid et al., 2019).

The Min-Max normalization represents one of the most common technique (David & Rögnvaldsson, 2021). Mathematically, if there is a set of matching scores (Ms) where, s=1, 2, . . ., n, the normalized scores (Ms') calculate according to Equation 2.8 (Alasadi & Bhaya, 2017):

$$\boldsymbol{Ms'} = \left(\frac{\boldsymbol{Ms-min}}{\boldsymbol{max-min}}\right) \qquad \textbf{(2.8)}$$

Where, max is the maximum value and min is the minimum value of all data samples.

2.6.2 Windowing Technique

The windowing represents an important and major strategy to process the data stream samples by segmenting it into limited sizes. Its importance comes from different reasons such as, the boundless size and single processing to every data sample (Bahri et al., 2021).

From an analytical point of view, windowing is the process of observing a certain number of stream data samples during a period (Poepsel-lemaitre et al., 2021).

Thus, in the windowing technique, there is a need to determine the first and the last data sample for the streaming to build the window. In other words, the number of data samples in a window should be determined. Three main types of the windowing technique that are, sliding window, landmark window, and damped window (S. Priya & R. A. Ythra, 2021).

The most widely used and effective type is the sliding window, therefore in the proposed system, this type is used.

2.6.3 Splitting Dataset

The dataset used by the ML models is usually divided into two parts, training part (training data) and testing part (testing data).

Over the past years, many methods and techniques have been proposed to split the dataset. The cross-validation technique is the most common one which used to split the dataset into training data and testing data (Rácz et al., 2021).

The ratio of splitting data differs from one model to another such as, 80%: 20%, 70%: 30%, and so on (Afshan & Rout, 2021).

Figure 2.2 illustrates the general framework of ML models.

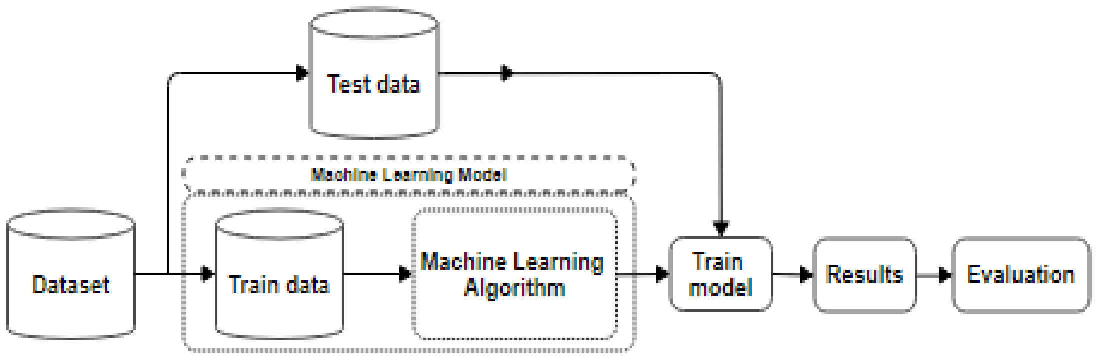


Figure 2.2: The general framework ML models

The first part of the partitioned dataset which is usually is used in the training phase. Practically, through the training data the model can be trained and form its main structure (rules), and generalized to other data, i.e. the model learns from this data (Joseph & Vakayil, 2022).

However, the method of training depends on the task of the model, and also the algorithms that is applied.

The second part is the testing dataset. It tests the ability of the model, which trained by the training data previously to perform a specified task. This means the testing data tests the model, whether it has been trained accurately (Joseph & Vakayil, 2022). Usually, the testing data is used during the testing phase.

The ML models (classifier) assign an invisible data sample to a previously predefined classes which is usually results from the training phase. It is very likely this step will be done automatically, otherwise is done manually by experts (Mousavi et al., 2019).

Several methods have been proposed to assign the testing data, with the most common being distance-based.

2.7 The Step Function

In Mathematics, the step function (also known as, floor function, greatest integer function, and also staircase function) is a constant function that has different indicators of given intervals (X. Wang, 2020).

The step-function differs from the other functions in its ability to relieve some restrictions and thus acts as a filter function. Moreover, it can be classified as a powerful analysis tool (Teixeira & Huszák, 2022).

The step function ƒ: $\mathbb{R} \rightarrow \mathbb{R}$ can be written as Equation 2.9 (Shi et al., 2022):

$$\boldsymbol{f(x) = \sum_{i=1}^{n} a_i \, X_{Ai} \; (X)} \tag{2.9}$$

Where n ⩾0, a_i all real numbers, A_i are the intervals, X_{Ai} is an indicator function.

Accordingly, the indicator of the step function can be computed as Equation 2.10 (Iliev et al., 2017):

$$X_A(X) = \begin{Bmatrix} 1; if\ x \in A \\ 0; if\ x \notin A \end{Bmatrix} \qquad \textbf{(2.10)}$$

In this function, every value of x, f(x) takes the value of the greatest integer. Whilst, the step function domain is a set of real numbers that are divided into different intervals (Teixeira & Huszák, 2022).

For example, when $f(x)$ represented by:

$$f(x) = \begin{Bmatrix} -2; & if & x < -1 \\ 0; & if & -1 \leqslant x \leqslant 2 \\ 2; & if & x > 1 \end{Bmatrix}$$

Then, the graph of this step function is displayed in Figure 2.3.

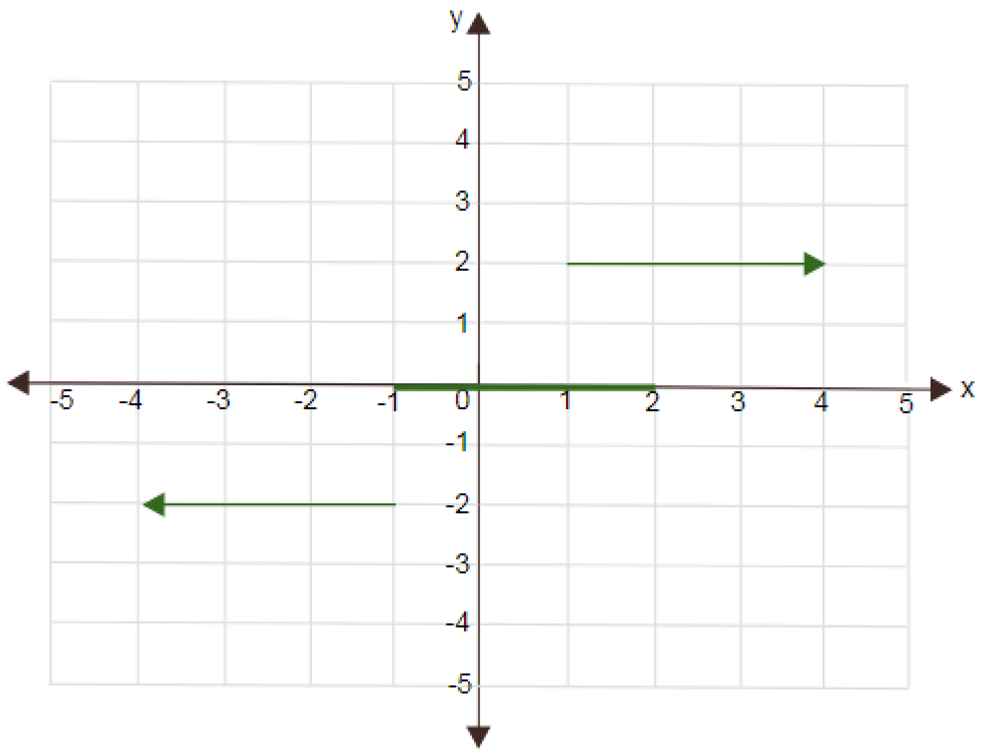


Figure 2.3: The step function

2.8 Fuzzy Logic

In the system based the classical (crisp or Boolean) set concept, the truth value consists of two statements, true or false (1 or 0). Typically, the Boolean expressions (such as, AND, OR, and XOR) help these systems to be more clear (Rahman et al., 2019).

In contrast, in some cases, these statements may be not clear (vague). In this case, the system cannot determine if the statement is either true or false. The fuzzy systems have proven to be successful and flexible when dealing with these values. Historically, the fuzzy concept is introduced in 1965 (Garcia et al., 2019).

As a result, fuzzy logic is used as a powerful tool for testing the membership degree of the vague statements. In other words, every data sample has a different degrees of membership in fuzzy sets. The membership function organizes these degrees in a structure therefore, the membership function considers as a major part in the fuzzy logic system (Khairuddin et al., 2021).

However, Figure 2.4 illustrates the membership function meanwhile explains the difference between the classical and the fuzzy sets.

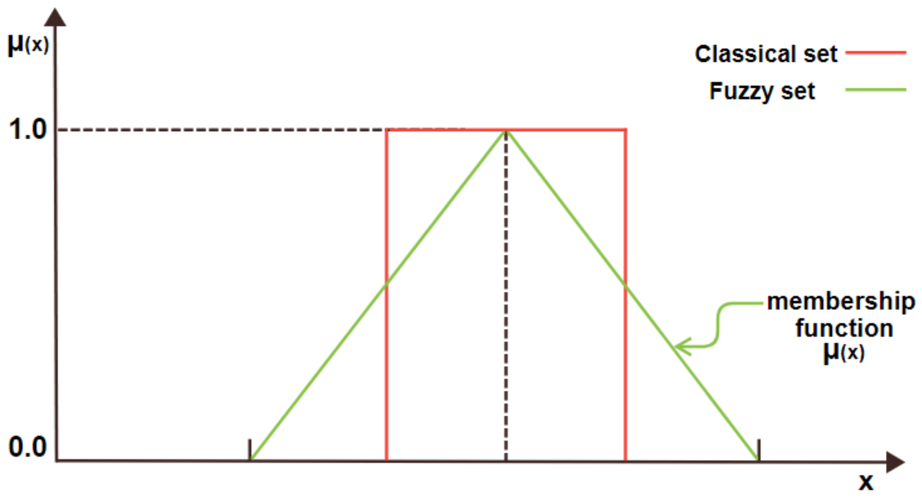


Figure 2.4: The different between the classical and fuzzy sets

The value (0) refers to the sample cannot consider a member of the fuzzy set whilst, the value (1) refers the sample considers a fully a member of the fuzzy set. Moreover, any value in the range [0-1] means the sample belongs partially to the fuzzy set (Khan et al., 2020). Simply, the membership function is a curve that defines how each point (data sample) in the input space is mapped to a membership value (or degree of membership). The input space is sometimes referred to as the universe of discourse (Babanezhad et al., 2021).

Different membership functions can be applied such as, linear, exponential, logistic, hyperbolic, parabolic, S-curve, and tangent membership functions, that is mainly based on the choice of the decision maker (S. M. S. K. Gupta, 2021). In other words, a decision maker may choose the membership function which fits best to their problem (Khan et al., 2020).

In the fuzzy systems, the exponential membership function is considered the most commonly used and successful type (Liu et al., 2020). Obviously, the usage of the exponential membership function would give a more realistic results than the linear ones in many problems (KARA & KÖÇKEN, 2022). Moreover, in real problems, the exponential membership function type may be preferred (Khan et al., 2020).

For example, when the system aims to determine a measurement of the quality. The results of the classical set are either excellent (true or 1) or poor (false or 0), while the results of the fuzzy set will be three classes that are, excellent, medium, and poor. Every quality value has a degree of membership in every of these sets.

Figure 2.5 shows the membership function of these fuzzy sets.

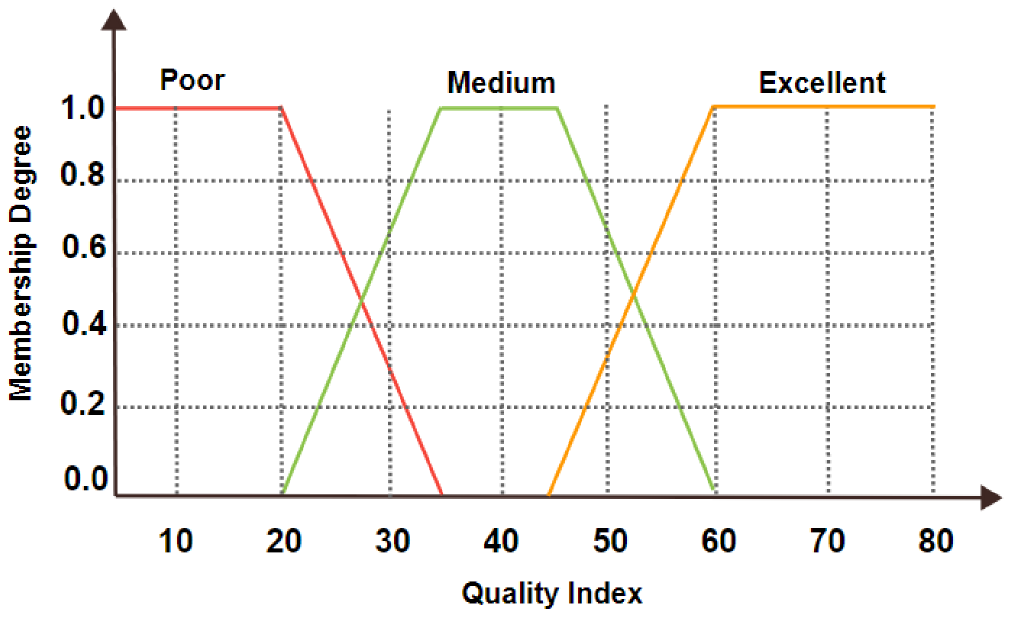


Figure 2.5: The membership function for three classes of fuzzy sets

Where x in the range [0,80] that refers to quality index. The degree of classes is shown in the next formulas:

$$\boldsymbol{Poor - degree.\mu_{P(x)}} = \begin{cases} \mathbf{1.}\ \boldsymbol{x} \leqslant \mathbf{20} \\ \frac{\mathbf{35}-\boldsymbol{x}}{\mathbf{15}}\ .\ \mathbf{20} < \boldsymbol{x} < \mathbf{35} \\ \mathbf{0.}\ \boldsymbol{x} \geqslant \mathbf{35} \end{cases}$$

$$\boldsymbol{Medium - degree.\mu_{M(x)}} =$$

$$\begin{cases} \mathbf{0.}\ \mathbf{20} \geqslant \boldsymbol{x} \geqslant \mathbf{60}\ ;\ \mathbf{1.35} \leqslant \boldsymbol{x} \leqslant \mathbf{45} \\ \frac{\boldsymbol{x}-\mathbf{20}}{\mathbf{15}}\ .\ \mathbf{20} < \boldsymbol{x} < \mathbf{35} \\ \frac{\mathbf{60}-\boldsymbol{x}}{\mathbf{15}}\ .\ \mathbf{45} < \boldsymbol{x} < \mathbf{60} \end{cases}$$

$$\mathbf{Excellent} - \boldsymbol{degree.\mu_{E(x)}} = \begin{cases} \mathbf{0.}\ \boldsymbol{x} \leqslant \mathbf{45} \\ \frac{\boldsymbol{x}-\mathbf{45}}{\mathbf{15}}\ .\ \mathbf{45} < \boldsymbol{x} < \mathbf{60} \\ \mathbf{1.}\ \boldsymbol{x} \geqslant \mathbf{60} \end{cases}$$

2.8.1 Fuzzy Systems Applications

The fuzzy system provides a very effective solution to complex problems in various areas of life as it is similar to human thinking and decision making (Garcia et al., 2019).

Therefore, the fuzzy systems are extensively used in different fields that are depended on the idea of the concept of fuzzy logic. In addition, the evolving fuzzy algorithms are classified as important kind of the ESs due to their ability to interact with the given dataset (Kour et al., 2020).

2.8.2 Fuzzy Rules

The fuzzy rules are derived from fuzzy knowledge that usually depended in either expert knowledge or historical data samples and aims to find (or define) relationships between inputs and outputs (Iuliis et al., 2019).

Simply, the fuzzy rules are the spine of any fuzzy system and their application helps greatly in describing the output (Ghani et al., 2018).

These fuzzy rules are used mainly for modeling the dynamic systems in various areas including, the science and engineering. However, the fuzzy rules should cover the whole discourse domain (J. Lu et al., 2019).

Each rule has two parts that are, antecedent and consequent parts. The first specifies when and what entails the rule to be used, while the second specifies the actions to be taken if its first part (antecedent) is met. A rule should be fired when, the antecedent part of it matches (satisfactorily) the case currently being processed (Kabir et al., 2020).

The two most popular fuzzy inference systems are, Mamdani and Takagi-Sugeno. The general structure of the Mamdani as in Equation 2.11 (Ghani et al., 2018):

$$\boldsymbol{R : IF\ X_1\ is\ x_1 and\ X_2\ is\ X_2\ and\ \ldots THEN\ Y\ is\ y_1} \qquad \textbf{(2.11)}$$

More deeply, based on Equation (2.11), the fuzzy rule that contains (IF: THEN) statement including membership function (**μ**) is constructed according to Equation 2.12:

$$\boldsymbol{IF\ \mu\ (operator)\ value\ THEN\ CS \in Cluster_j} \qquad \textbf{(2.12)}$$

Where CS is the current sample.

2.9 Evaluation Measures of the Classification

Many different measurements of the Classification technique are based on the Confusion Matrix (CM) which provides an accurate information on the classifier performance. CM consists of four cases that are (Markoulidakis et al., 2021):

1. True Positive (TP): refers to the number of correctly classified positive data samples.
2. False Positive (FP): indicates to the number of negative data samples which are incorrectly classified as positive data samples.
3. True Negative (TN): points out to the number of correctly classified negative data samples.
4. False Negative (FN): refers to the number of positive data samples that are incorrectly classified as negative data samples.

   However, Table 2.4 explains the cases of the CM.

Table 2.4: The confusion matrix

		Predicted class	
		+	–
Actual Class	+	***TP***	**FN**
	–	***FP***	**TN**

The most common measurements of the DL technique as a classifier are (Selim et al., 2021), (Nayak et al., 2022), (Al-khamees et al., 2022):

1. Accuracy: is defined as the ratio between the correct classified data samples to all data samples of the dataset. Its Equation as 2.13:

$$\boldsymbol{Accuracy} = \frac{(\boldsymbol{TP}+\boldsymbol{TN})}{(\boldsymbol{TP}+\boldsymbol{TN}+\boldsymbol{FP}+\boldsymbol{FN})} * (\mathbf{100})\% \qquad \textbf{(2.13)}$$

2. Precision: sometimes called True Positive Rate (TPR) or Positive Predictive Value (PPV). It is the proportion between the data samples that are correctly positive TP to all positive data samples (TP and FP). It can calculate by Equation 2.14:

$$\boldsymbol{Precision} = \frac{(\boldsymbol{TP})}{(\boldsymbol{TP}+\boldsymbol{FP})} * (\mathbf{100})\% \qquad \textbf{(2.14)}$$

3. Recall: is the ration between the correctly positive data samples to the correct positives and false negatives data samples. Recall is computed by Equation 2.15:

$$\boldsymbol{Recall} = \frac{(\boldsymbol{TP})}{(\boldsymbol{TP}+\boldsymbol{FN})} * (\mathbf{100})\% \qquad \textbf{(2.15)}$$

4. F1_measure: it can be computed based on mixing the precision and recall simultaneously. Equation 2.16 describes F1_measure:

$$\boldsymbol{F1\_score} = \frac{\mathbf{2}*(\boldsymbol{Precision}*\boldsymbol{Reacall})}{(\boldsymbol{Precision}+\boldsymbol{Recall})} * (\mathbf{100})\% \qquad \textbf{(2.16)}$$

Despite the importance of the above measurements, the error rate is calculated by Equation 2.17:

$$\boldsymbol{Error\ rate} = \mathbf{1} - \frac{(\textbf{Accuracy})}{(\mathbf{100})} \qquad \textbf{(2.17)}$$

In the same context, the error rate for the proposed model (ER_1), and for the standard model (ER_2) are computed separately. The error enhancement indicates to the amount of decreasing error through the proposed DL model that is computed by Equation 2.18:

$$\boldsymbol{Enhancement\ rate} = \boldsymbol{ER_2} - \boldsymbol{ER_1} \qquad \textbf{(2.18)}$$

2.10 Evaluation Measures of the Clustering

One of the main measurements of the clustering technique is the NoC which, represents the final output of a clustering model.

Consequently, the final results of a clustering model include many clusters which need to measure their quality due to contain different properties of data samples in every cluster. Mainly the evaluation measurements can be classified into two types (Al-Khamees et al., 2021):

1. The internal measures: also known as unsupervised measures. This measures divide into two kinds, cluster cohesion and cluster separation.
2. The external measures: in contrast to the first type that also known as supervised measures.

SC is one of the most widely used measures of the clustering evaluation which belongs to the internal measure. The general idea of the SC is to compute the mean distance (Lengyel & Botta-Dukát, 2019).

The SC merges the cohesion and separation as follows (Kaoungku et al., 2018):

- K is a cluster which contains many data samples $x_{(i)}$.
- $ax_{(i)}$ indicates the average distance from $x_{(i)}$ to each data sample in the same cluster K.
- $bx_{(i)}$ represents the lowest average distance between $x_{(i)}$ and each data sample in other clusters that is not K.

Afterwards, the SC of the data sample $x_{(i)}$ can compute by Equation 2.19:

$$\boldsymbol{SCx_{(i)}} = \frac{(\boldsymbol{bx_{(i)}} - \boldsymbol{ax_{(i)}})}{\boldsymbol{max}\,(\boldsymbol{ax_{(i)}}\ .\ \boldsymbol{bx_{(i)}})} \qquad \textbf{(2.19)}$$

The value of the SC is in the range [−1, 1]; the closer it is to 1, the better is the clusters result; the closer it is to −1, the worse is the clusters result (Yang et al., 2019).

Chapter Three

The Proposed System

The Proposed System

3.1 Introduction

This chapter focuses on the design of the proposed evolving intelligent system. It basically explains the algorithms to implement in this system in both models, the DL and the evolving clustering models, the advantage of applying each model is detailed, and the evaluation stage.

3.2 The Proposed System Architecture

The proposed evolving intelligent system consists of two models based on two different ML techniques, namely, DL and clustering as shown in the block diagram of Figure 3.1.

The proposed system consists of four stages, which are:

3.2.1 Preprocessing Stage

Before implementing the proposed system, the preprocessing stage is applied, which aims to prepare the data in an appropriate form. Practically, the preprocessing stage consists of two steps which are, the normalization and the windowing.

The first step is the normalization technique which, is implemented to transform the dataset in the range [0,1]. The Min–max method is applied depends on Equation (2.8).

The second step for the preprocessing is the applying of the windowing technique to segment the given stream dataset. Indeed, the applying of this processing technique depends on the type of stream data being processed for example, when processing UCI-HAR stream dataset, the windowing technique should be applied.

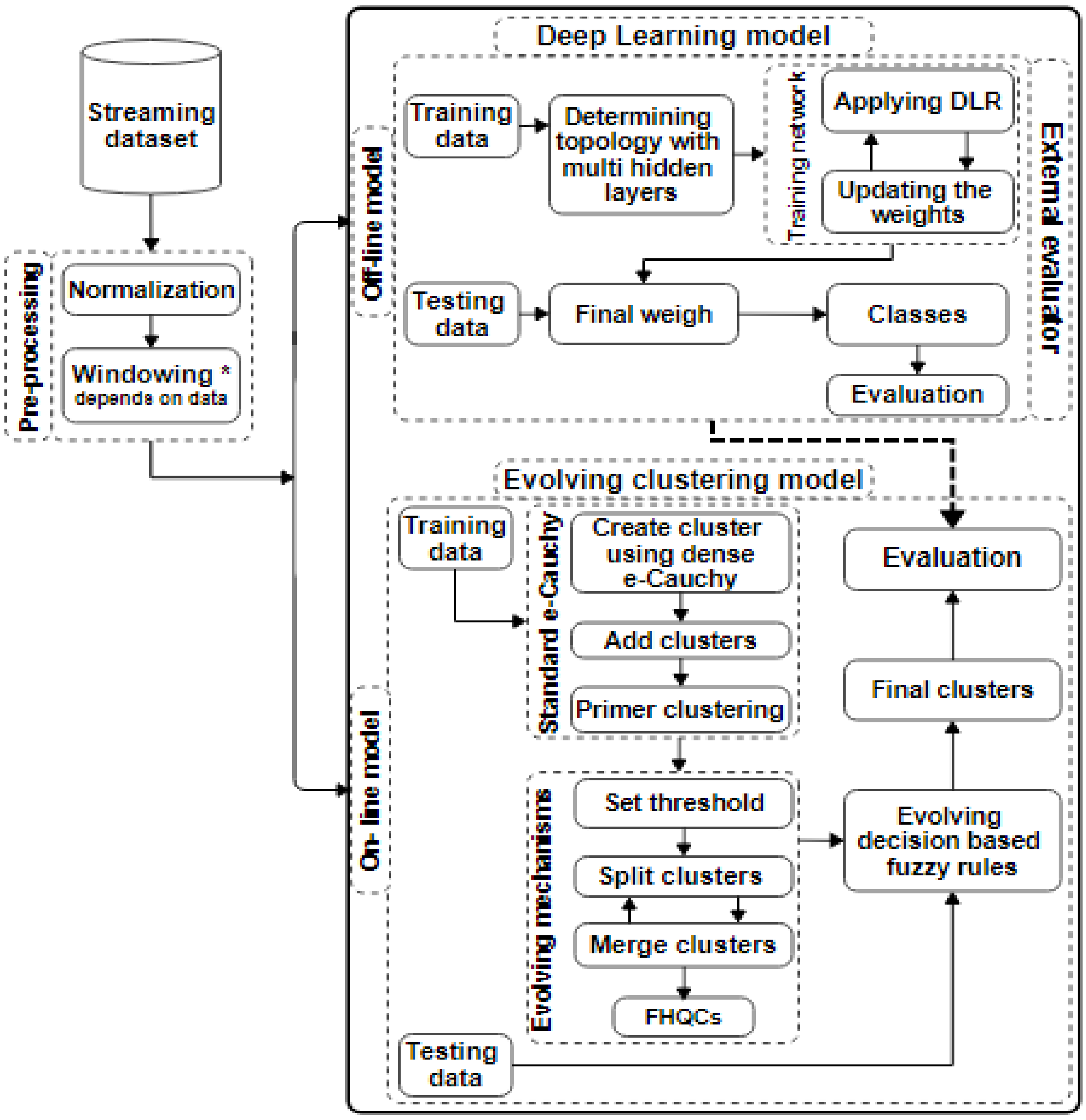


Figure 3.1: The block diagram of the proposed system

In this evolving intelligent system, the sliding window is used and the number of data samples in the window is 100. This window moves over the stream while keeping the same size.

Then, splitting a specific stream dataset is done as the following ratio, 70% of the whole dataset as training data, and the rest of the dataset is 30% as a testing data.

However, Algorithm (3.1) displays the preprocessing techniques.

Preprocessing data algorithms

- ***Input***: *Dataset (D)*
- ***Output:*** *Normalized data*

```
1.  Begin:
2.    min, max ← min, max in (D)  /* Find min, max values of Dataset */
3.    normalized data= []
4.      For d in D do
5.        Normalized data [D] ← $\frac{D-min}{max-min}$
6.      End for
7.        For d in D do                    /* Windowing algorithm (if needed) */
8.          i = 100
9.          d = d +100
10.     End for
11. Return normalized data
12.End
```

Algorithm 3.1: Preprocessing data algorithm

3.2.2 Deep Learning Model

It is the first model in this system that implements in off-line mode, and depends on the DL technique as described in Figure 3.1. The objective of DL model is to classify all data samples of the given stream dataset into classes. Moreover, this model is used as an external evaluator. Practically, the MLP structure uses in this model. Figure 3.2 shows the general MLP structure.

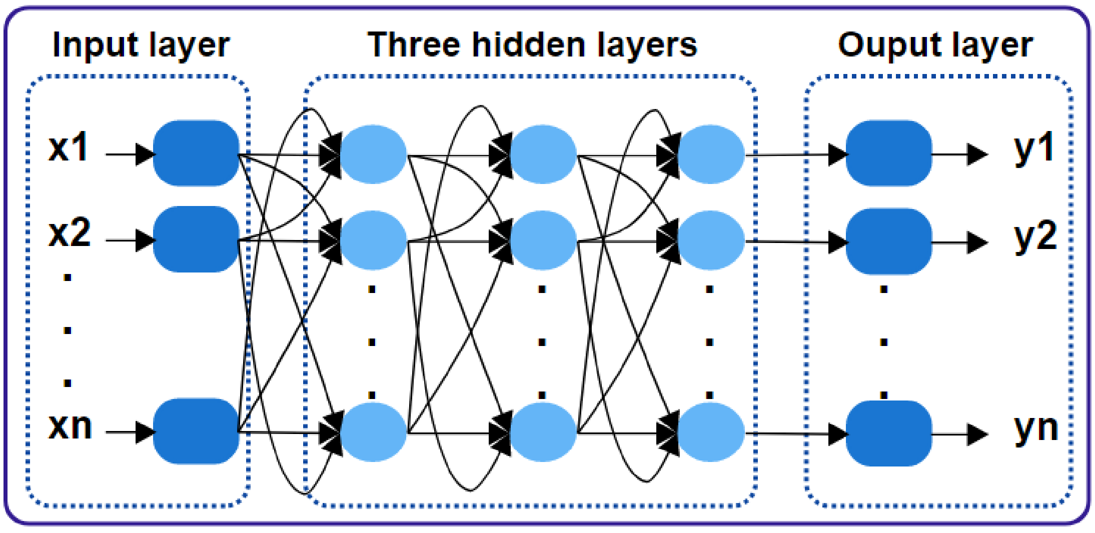


Figure 3.2: The MLP structure

Due to the nature and properties of each stream dataset in terms of the different number of features and classes, each dataset has its own MLP architecture. More specifically, the different is in the number of nodes of the input and output layers, while all MLP structures contain three hidden layers as follows, the first hidden layer has 100 neurons, while the second hidden layer contains 50 neurons, and the last hidden layer has 25 neurons. Table 3.1 describes the difference of the nodes number for the input and output layers for all these MLP structure.

Table 3.1: The number of nodes for MLP architecture for all stream datasets

Item	Dataset name	Input layers	Output layers
1.	Keystroke	10 nodes	4 nodes
2.	Electricity	8 nodes	2 nodes
3.	NSL-KDD	106 nodes	5 nodes
4.	HuGaDB01_01	39 nodes	4 nodes
5.	HuGaDB05_12	39 nodes	3 nodes
6.	HuGaDB13_11	39 nodes	3 nodes
7.	HuGaDB14_05	39 nodes	2 nodes
8.	UCI-HAR	561 nodes	6 nodes
9.	Luxembourg	31 nodes	2 nodes
10.	Keystroke_1	10 nodes	3 nodes
11.	HuGaDB01_01_	39 nodes	3 nodes

Practically, this model uses MLP architecture for classifying the data samples depends on the DLR value. This is done by determining this value, and training the network depending on it. This model is presented to tackle the issue of learning rate stability as well as to improve results.

The new method summarized as follows:

- Initialize the learning rate value at 0.001.
- A new parameter is derived (γ) that is added and subtracted at each time and evaluated to determine DLR value.

- Determine the search area to find DLR value between λ_{min} and λ_{max} i.e. these two values define as the boundaries of the DLR value, as long as the LR values close to zero affect in both of generalization and enhance the stability of the network. Therefore, λ_{min} and λ_{max} set to 0.0005 and 0.0009 respectively.
- After determining the DLR value, the network is trained with that value.

Accordingly, these steps will ensure that the DL model gets the DLR value through it, the network is trained. Practically, the DLR value is located in the range of Equation 3.1:

$$\boldsymbol{\lambda_{min}} < \mathbf{DLR} < \boldsymbol{\lambda_{max}} \tag{3.1}$$

Algorithm (3.2) shows the DLR algorithm.

The DLR algorithm

```
- Input: Stream dataset D
- Output: DLR
  1.  Begin:
  2.  Set parameters as: $\lambda_{min}$= 0.0005, $\lambda_{max}$= 0.0009, $Max_{LR}$=0.1, $Min_{LR}$=
      0.0, LR= 0.001, i=1, $Max_{iter}$=100
  3.  While i < $Max_{iter}$ do
  4.     $\Upsilon = (\lambda_{\max} - \lambda_{\min})$
  5.     $\lambda = \lambda_{\max} - \Upsilon * \left(\frac{current_{iter}}{max_{iter}}\right)^2$
  6.     $Temp_{1LR} \leftarrow 0.001$
  7.     $Temp_{2LR} \leftarrow 0.001$
  8.     $Temp_{1LR} \leftarrow Temp_{1LR} + \lambda$
  9.     $Temp_{2LR} \leftarrow Temp_{1LR} - \lambda$
  10.     IF $Temp_{1LR} > Max_{LR}$ Then
  11.        $Temp_{1LR} \leftarrow Max_{LR}$
  12.        $EV_1 \leftarrow (Temp_{1LR}, data)$
  13.    End
  14.    IF $Temp_{2LR} < Min_{LR}$ Then
  15.        $Temp_{2LR} \leftarrow Min_{LR}$
  16.        $EV_2 \leftarrow (Temp_{2LR}, data)$
  17.    End
  18.    IF $EV_1 > EV_2$ Then DLR $\leftarrow Temp_{1LR}$
  19.         Else: DLR $\leftarrow Temp_{2LR}$
  20.    End
  21.    i= i+1
```

```
    22. End While
    23. Return DLR
    24. End
```

Algorithm 3.2: The DLR algorithm

Instead of training the network according to Equation (2.4) that adopts a constant learning rate, and after the DLR vale is obtained by Algorithm (3.2), the network is trained based on this value, and then, updating of the weights as Equation 3.2:

$$\boldsymbol{Nw_i} = \boldsymbol{w_i} - (\boldsymbol{DLR})\left(\frac{\partial\, \boldsymbol{e_{m(n)}}}{\partial\, \boldsymbol{w_i}}\right) \qquad \textbf{(3.2)}$$

Finally, many valuable aspects of applying DL model are achieved as described in Figure 3.3.

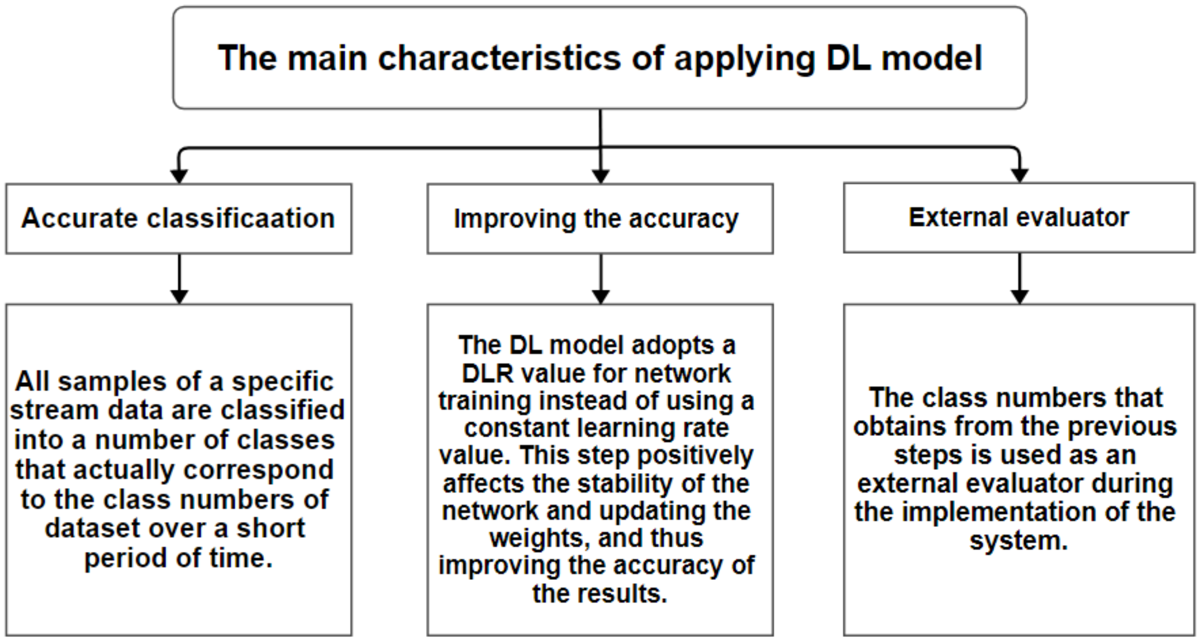

The main characteristics of applying DL model
Accurate classificaation
Improving the accuracy
External evaluator
All samples of a specific stream data are classified into a number of classes that actually correspond to the class numbers of dataset over a short period of time.
The DL model adopts a DLR value for network training instead of using a constant learning rate value. This step positively affects the stability of the network and updating the weights, and thus improving the accuracy of the results.
The class numbers that obtains from the previous steps is used as an external evaluator during the implementation of the system.

Figure 3.3: The main characteristics of applying DL model

3.2.3 Evolving Clustering Model

It is the second model in the evolving intelligent system that is applied in on-line mode and depends on the clustering technique as shown in Figure 3.1. The evolving clustering model aims to cluster all data samples of the given stream dataset, as well as to decrease the number of the generated clusters (primer clustering), and then, construct accurate clusters by implementing evolving mechanisms (adding clusters, splitting clusters and merging clusters). More specifically, Figure 3.4 explains this model.

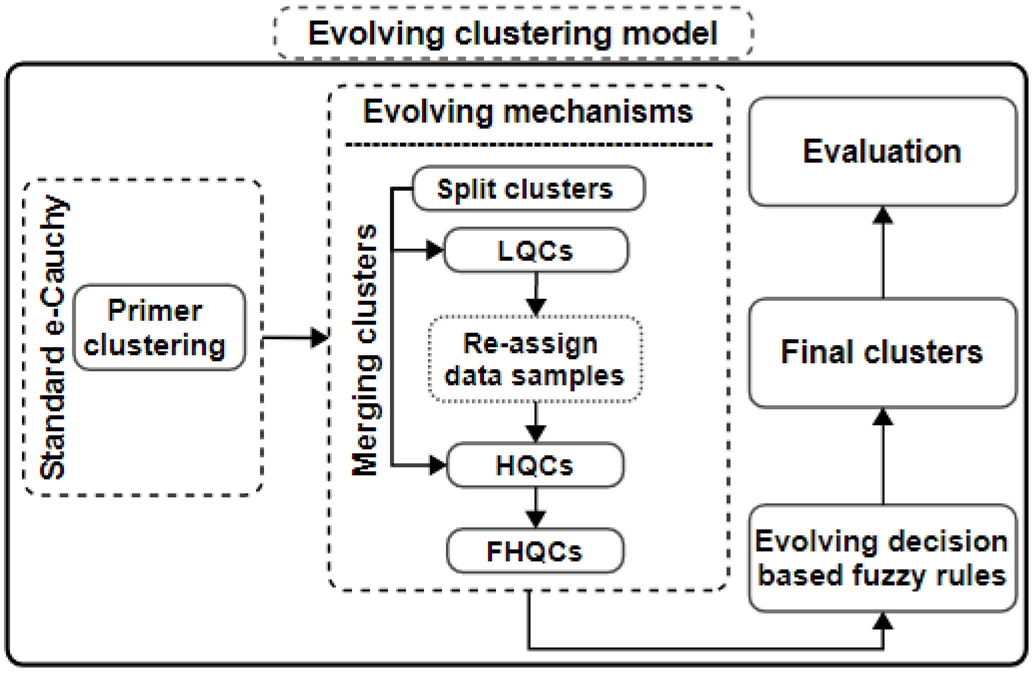


Figure 3.4: The block diagram of the evolving clustering model

Firstly, this model applies the standard e-Cauchy clustering algorithm (which is classified as a density-based clustering methods) to create a cluster. Then, it repeatedly applies the first mechanism that is the adding clusters. In the evolving clustering model, many values are tested as a density threshold Θ_{den} until get the optimal value. Thus, this threshold value differs from one stream dataset to another.

When the first data sample arrives, the first cluster is constructed and set its parameters. Otherwise, the density value is computed according to Equation (2.5). This value will compare with Θ_{den}, if the density value is less than Θ_{den}, then construct a new cluster otherwise, the current data sample is appended to an existing cluster, and modify the cluster parameters. These steps are repeated to all samples of the selected stream dataset to build the primer clustering.

Algorithm (3.3) explains the standard e-Cauchy algorithm. In this regard, a data stream D consists of {x_1, x_2,..., x_i}, SC_j refers to the samples in cluster (j), CC_j is the center of the cluster (j), and NoC is the number of clusters

```
e-Cauchy algorithm
- Input: Stream dataset D
- Output: Primer clusters
  1.  Begin:
  2.   Set thresholds as: $\Theta_{con}$= 0.1, $\Theta_{den}$
  3.   Set parameters as: $SC^j$, $CC^j$, NoC
  4.   Receiving data sample $x_{(i)}$
  5.     IF $x_{(i)}$ is the first data sample Then
  6.        Construct the first cluster $C_1$
  7.        Set cluster parameters, $SC^1$=1, $CC^1$= $x_{(1)}$
  8.        Increase NoC by 1
  9.    Else compute the density value $Den_{(x)}$ by applying equation 2.5
  10.     IF Den(x) < $\Theta_{den}$ Then
  11.       Generate a new cluster
  12.       Set parameters of a new cluster (j) as: $SC^j$=1, $CC^1$= $x_{(i)}$
  13.       Increase NoC by 1
  14.     Else
  15.       Append $x_{(i)}$ to an existing cluster and modify the cluster parameters
          as:
  16.       $SC^j$= $SC^j$ +1
  17.       $D_{j(x)}$= x(i) - $CC^j$
  18.       $CC_{j+1}$= $CC_j$ ( $\frac{1}{SC^j+1}$ ) $D_{j(x)}$
  19.     End if
  20.  End if
  21.  Return Primer clusters
  22.  End

- Input: Primer clusters                    /* Splitting clusters algorithm*/
- Output: LQCs and HQCs
  23.  Begin:
  24.   Set splitting thresholds: $\Theta_{spl}$= 0.2
  25.   Set parameters $\mu_{SC_{c(i)}}$, $a_{x(i)}$ is the average distance from x(i) to every
       data in the same cluster; $b_{x(i)}$ is the smallest average distance between
       x(i) and each data sample in other clusters.
```

```
26.    For each primer cluster do
27.        Compute $\mu_{SC_{c(i)}}$ by applying equation 2.19
28.    IF $\mu_{SC_{c(i)}} \geqslant \Theta_{spl}$ Then
29.        $c_{(i)} = HQC$                                /* Determine HQCs*/
30.    Else $c_{(i)} = LQC$                               /* Determine LQCs*/
31.    End if
32. End For
33. Return LQCs and HQCs
34. End
- Input: LQCs and HQCs                  /* Merging clusters algorithm*/
- Output: FHQCs
35. Begin:
36.    For each LQC do
37.        Merge each data sample to HQC by applying equation 2.7
38.    End For
39.    Return FHQCs
40. End
```

Algorithm 3.3: The e-Cauchy algorithm

The core step of e-Cauchy clustering algorithm is the density computation for every training data sample continuously (line 9 in Algorithm (3.3)). The implementation of this clustering algorithm generates the primer clusters. This number is usually large and needs further processing to generate the optimum number of clusters that must match the number of classes of the dataset being processed. Therefore, the evolving mechanisms are included in this model to overcome this issue.

The goal of these mechanisms is to change the structure of the system according to the behavior of the data. Since the data stream environment is a non-stationary, the evolving mechanisms are very useful and efficient. The evolving mechanisms that applied in this model are, adding clusters mechanism, splitting clusters mechanism, and merging clusters mechanism.

1- Adding clusters mechanism: in the evolving clustering model, the e-Cauchy clustering algorithm is used and as this algorithm is a density-based method, the data sample density value is computed continuously to add the clusters.
   After the first cluster is constructed, the density value is computed for each data stream sample by Equation (2.5). Then, this value is compared with the second threshold Θ_{den} to add a cluster (line 10. in Algorithm 3.3). Otherwise, the current data sample is appended to an existing cluster (line 15.).
   It should be noted, that after creating each cluster, its parameters are set (as in line 12). On the other hand, if the current data sample is appended to an existing cluster, the parameters of the cluster are modified (lines 16,17, and 18).
2- Splitting clusters mechanism: after implementing the adding clusters mechanism, the splitting clusters mechanism is applied to split the generated clusters (the primer clusters) into high quality clusters (HQCs) and low quality clusters (LQCs).
   The step function is used as a condition to split the cluster. To do that, the SC measure is included to build the first membership function in this system. The SC membership function is written as $\mu_{SC_{c(i)}}$, and the splitting threshold denotes as Θ_{spl} is set to 0.2 (line 24). The splitting clusters method is described in Algorithm (3.3) by lines 23 - 34. Consequently, the first rule to define HQCs as Equation 3.3:

   R1: If $\boldsymbol{\mu_{SC_{ci}} \geqslant 0.2}$ **(3.3)**

   While the LQCs are defined by the rule of Equation 3.4:

   R2: If $\boldsymbol{\mu_{SC_{ci}} < 0.2}$ **(3.4)**

   As a result of applying the splitting mechanism in the proposed model, the primer clusters are evolved, and thus, the HQCs are constantly obtained.

3- Merging clusters mechanism: to date, there are two types of clusters which are HQCs and LQCs. The latter type degrades the overall performance of the model since it contains mis-classified data samples. Accordingly, the data samples of each LQC can be merged into the HQCs. Thus, only HQCs are produced as result to apply the merging mechanism.
In other words, during this mechanism, the proposed model able to evolve the clusters by re- assigning all the data samples in each LQC to HQCs to produce only final HQCs (FHQCs) as described in Algorithm (3.3) lines 35 - 40.

In this model, a new membership function is used, that depends on the idea of the exponential membership function. Practically, for each HQC, three membership functions μf_i are defined according to three unit value (UV_i) which are, μf_1, μf_2, and μf_3. Thereafter, the center of gravity is computed as $\mu f_{c(i)}$.

Figure 3.5 illustrates the structure of these membership functions.

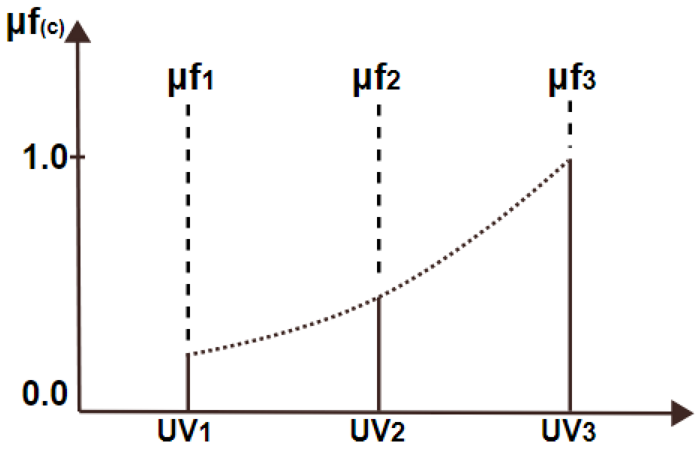


Figure 3.5: The structure of the exponential membership function

The re-assign method of the data samples of LQCs to HQCs is based on these membership functions.

Indeed, every cluster contains three parameters (as in Figure 3.5) which are, (1) UV_1, refers to the closest data sample in the cluster, (2) UV_2, corresponds the center of the cluster, and (3) UV_3, indicates to the farthest data sample in the cluster. Therefore, instead of comparing the current data sample (which belongs to the LQC) with all the data samples, the comparison will be only with the number of clusters multiplied by 3. As a result, the maximum comparison is the number of clusters multiplied by three.

Despite of the importance of reducing the time by reducing the comparison numbers, this method is an effective in solving the new sample problem.

Practically, the distance from the current sample (CS) is calculated to:

1- The closest data sample in a cluster which is denoted by $\mu f_{1(cs.clo)}$.
2- The center of the cluster as $\mu f_{2(cs.cen)}$
3- The farthest data sample in a cluster that is referred by $\mu_{f3(cs.far)}$.

Then, for every cluster and from these three membership functions, the $\mu f_{c(i)}$ is constructed. Eventually, many $\mu f_{c(i)}$ (i= 1,2,…,n: where n, is the number of HQCs) are formed. Actually, the number of $\mu f_{c(i)}$ is equal to the number of HQCs.

On the other hand, different fuzzy set operations like, union, intersection, complement, and many others can be used to obtain clusters that are representative of reality and close to the results of human experts. In this system, the intersection operation is used.

For that reason, the belongingness of current sample (Bel_{cs}) to all clusters can be calculated by applying the intersection operation among all $\mu f_{c(i)}$ as Equation 3.5:

$$\boldsymbol{Bel_{cs} = \mu f_{c(1)} \cap \mu f_{c(2)} \cap \ldots \mu f_{c(n)}} \quad \textbf{(3.5)}$$

In reality, the value of Bel_{cs} determines the accurate cluster to the current sample (CS) thus, this sample assigns to an appropriate cluster. More deeply, the Bel_{cs} value is very important because the fuzzy rules that this system constructs, are based on this value.

Practically, the intersection of the fuzzy sets determines how much the data sample belongs to the fuzzy sets. Mostly, it has different degrees of membership in each set. Algorithm (3.4) explains the method of assigning the testing data samples.

```
The assigning data algorithm
- Input: FHQCs, data samples
- Output: Final clusters
  1.  Begin:
  2.    For each FHQC $\epsilon$ FHQCs do
  3.      Determine ($far.cen.and\ clo$) points
  4.    End for
  5.    For each data sample $\epsilon$ FHQCs do
  6.      Compute the Euclidean distance to the points in step (2)
  7.      Construct membership functions as: $\mu f_{1(cs.clo)}$, $\mu f_{2(cs.cen)}$, and
          $\mu f_{3(cs.far)}$
  8.      Construct $\mu f_{c(i)}$, where i= 1,2,…,n
  9.      Find the intersection among all the constructed membership
          functions $\mu f_{c(i)}$ according to equation 3.5
  10.     Assign current sample ($CS$) according to $Bel_{cs}$
  11.   End for
  12. Return Final clusters
  13. End
```

Algorithm 3.4: The assigning data algorithm

As results of applying the evolving mechanisms, Figure 3.6 details the main characteristics and benefits that are achieved by these mechanisms.

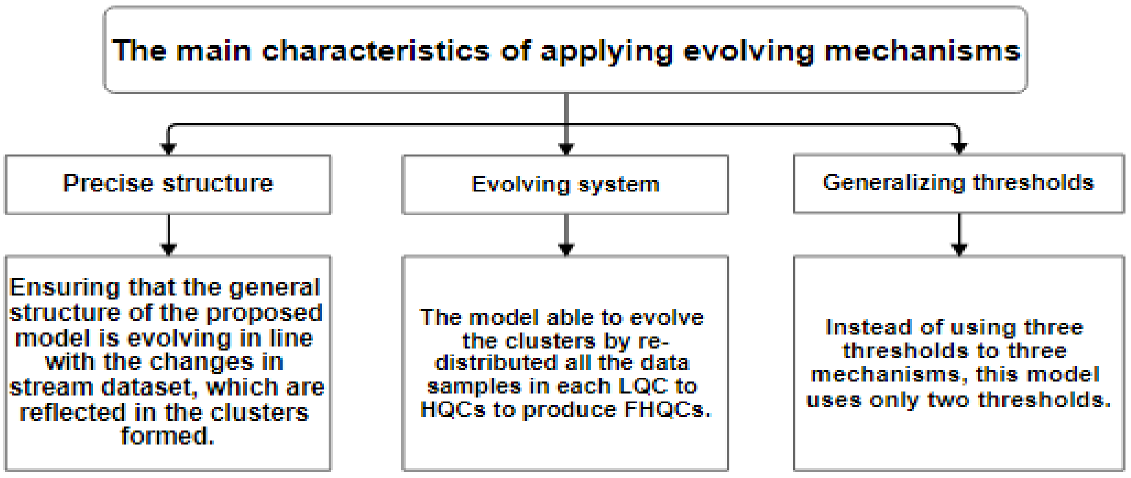

The main characteristics of applying evolving mechanisms
Precise structure
Evolving system
Generalizing thresholds
Ensuring that the general structure of the proposed model is evolving in line with the changes in stream dataset, which are reflected in the clusters formed.
The model able to evolve the clusters by re-distributed all the data samples in each LQC to HQCs to produce FHQCs.
Instead of using three thresholds to three mechanisms, this model uses only two thresholds.

Figure 3.6: The main characteristics of applying the evolving mechanisms

So far, all training dataset which constitute 70% of all data has been processed. The primer clustering is produced as a result of applying the standard e-Cauchy clustering algorithm, and also the evolving mechanisms. The splitting mechanism responsible to split the generated clusters into the HQCs and LQCs. Finally, the merging mechanism is applied to merge all data samples in the LQCs into the HQCs to produce FHQCs.

Typically, these mechanisms are implemented over many epochs that differ in numbers from one stream dataset to another. As for the testing data, when forming the HQCs after performing the evolving mechanisms, the testing data must be assigned to these clusters.

The concept of fuzzy logic can be used as a powerful and effective tool in the distribution (assigning data samples). Actually, the distribution of the testing data depends entirely on the fuzzy rules that will be created according to the clusters that have been formed.

In the same context, the membership function plays a pivotal role in terms of determining the affiliation of the current testing data sample to any cluster by relying on a specific condition that is usually calculated during the implementation of the work and included in the fuzzy rules. Thus, it will facilitate the accurate decision-making in terms of affiliation of the current sample.

The fuzzy rules based membership function can determine the belongingness of the current testing data sample to a specific cluster perfectly by adopting the value of Bel_{cs} from Equation (3.5). The structure of these rules are based on Equations (2.11) and (2.12).

Let's illustrate this with an example, suppose there are two fuzzy sets (clusters). After determining the three membership functions for each cluster, which are, $\{\mu f_{(1)}. \mu f_{(2)}.\ and\ \mu f_{(3)}\}$.

And propose the value of these $\mu f_{(i)}$ to each cluster as in Figure 3.7, where the first cluster with red color while, the second cluster with yellow color.

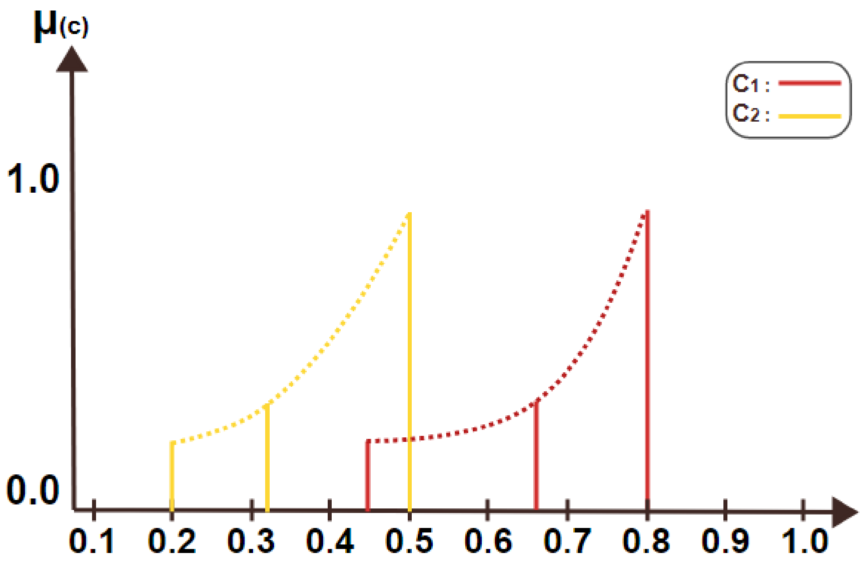


Figure 3.7: $\mu f_{(i)}$ values for tow clusters

Afterwards, $\mu f_{c(i)}$ are constructed as, $\mu f_{c(1)}$ and $\mu f_{c(2)}$, and then, the Bel_{cs} is computed according to Equation 3.5 where n=2.

Finally, different intervals (I_x) are specified as shown in Table 3.2, and the fuzzy rules are built as detailed in Table 3.3.

Table 3.2: The intervals to two clusters

Interval No	Interval details	Interval No	Interval details
I_1	[0, 0.42]	**I_6**	[0.46, 0.91]
I_2	[0, 0.5]	**I_7**	[0.38, 0.46]
I_3	[0.34, 0.53]	**I_8**	[0.27, 0.49]
I_4	[0.14, 0.23]	**I_9**	[0.43, 0.79]
I_5	[0.44, 0.87]	**I_{10}**	[0.51, 0.8]

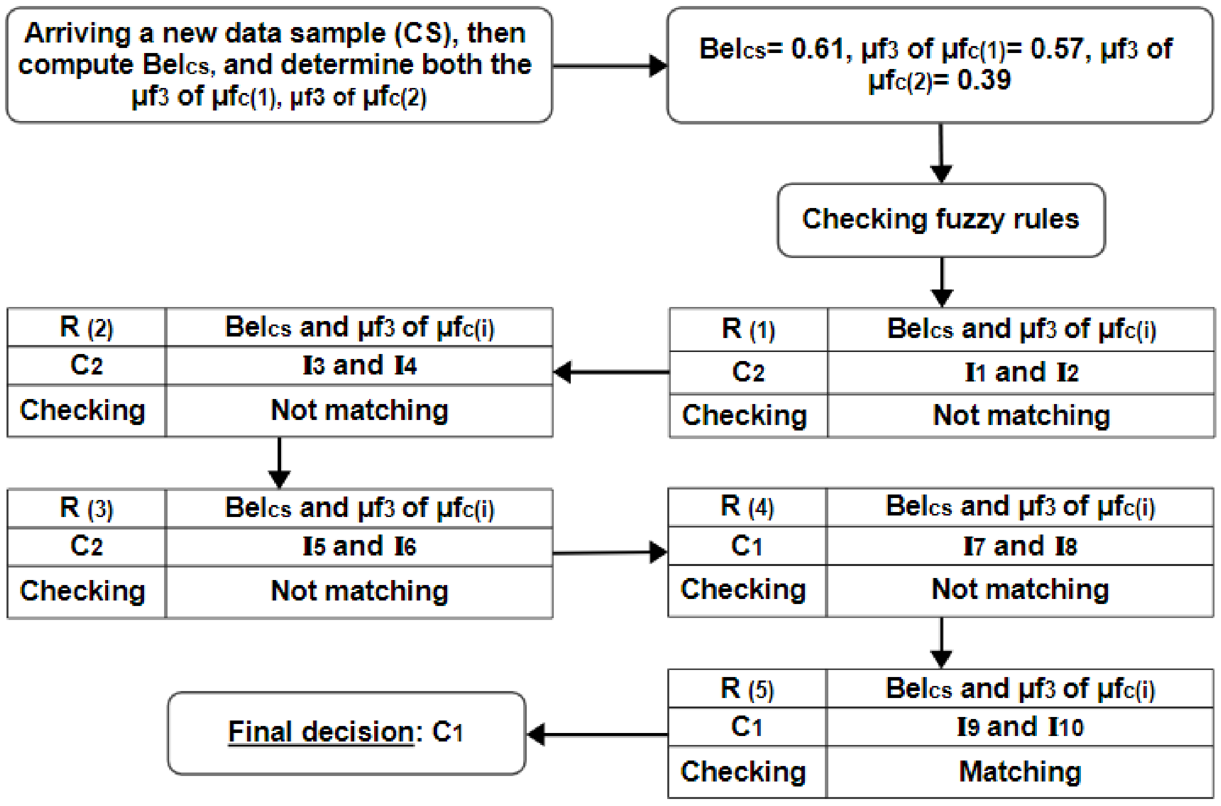
Table 3.3: The construction of fuzzy rules to two clusters

Rule No.	Rule description
R_1	IF Bel_{cs} ∈ **I_1** and μf_3 of $\mu f_{c(2)}$ ∈ **I_2** THEN $CS \in C_2$
R_2	IF Bel_{cs} ∈ **I_3** and μf_3 of $\mu f_{c(2)}$ ∈ **I_4** THEN $CS \in C_2$
R_3	IF Bel_{cs} ∈ **I_5** and μf_3 of $\mu f_{c(2)}$ ∈ **I_6** THEN $CS \in C_2$
R_4	IF Bel_{cs} ∈ **I_7** and μf_3 of $\mu f_{c(1)}$ ∈ **I_8** THEN $CS \in C_1$
R_5	IF Bel_{cs} ∈ **I_9** and μf_3 of $\mu f_{c(1)}$ ∈ **I_{10}** THEN $CS \in C_1$

To date, training data processing and clusters formation have been completed. If there is a testing data sample has been arrived, the distances from this data sample to the three membership functions μf_i in cluster C_i are computed.

Then, according to the value of Bel_{cs} and by applying the fuzzy rules (such as, that described in Table 3.3), the decision is making to assign the current sample (CS) to a suitable cluster.

Suppose arrives a new data sample, whose Bel_{cs} is 0.61, μf_3 of $\mu f_{c(1)}$ is 0.57, and μf_3 of $\mu f_{c(2)}$ is 0.39. Then, the final decision of this current sample (CS), it belongs to C_1. Figure 3.8 shows these details.



Arriving a new data sample (CS), then compute Bel$_{cs}$, and determine both the μf3 of μfc(1), μf3 of μfc(2)
Bel$_{cs}$= 0.61, μf3 of μfc(1)= 0.57, μf3 of μfc(2)= 0.39
Checking fuzzy rules
R (1) | Bel$_{cs}$ and μf3 of μfc(i) | C2 | I1 and I2 | Checking | Not matching
R (2) | Bel$_{cs}$ and μf3 of μfc(i) | C2 | I3 and I4 | Checking | Not matching
R (3) | Bel$_{cs}$ and μf3 of μfc(i) | C2 | I5 and I6 | Checking | Not matching
R (4) | Bel$_{cs}$ and μf3 of μfc(i) | C1 | I7 and I8 | Checking | Not matching
R (5) | Bel$_{cs}$ and μf3 of μfc(i) | C1 | I9 and I10 | Checking | Matching
Final decision: C1

Figure 3.8: The appending of a new data sample to an existing cluster (when it matches the constructed fuzzy rule)

The models that perform in offline mode assume all their dataset samples are existing. Therefore, it is easy to process these samples, analysis, and then, know the data behavior. In contrast, the models that implement in an online manner, process the dataset (such as the data stream) that is generated in real life or often known as real world applications. In this case, the behavior of the data is unknown.

Accordingly, the probability of appearing new data samples is more likely in the online mode than in the offline mode, and these data samples are definitely not pre-trained therefore, the system must detect it immediately, know its behavior, and adopt it because it is possible that the upcoming data samples are similar to it in terms of behavior.

Practically, the system creates a new cluster and assigns all data samples that have a different behavior than the previously trained data samples to this new cluster. The system thus ensures that all future samples belonging to this new cluster are assigned correctly.

For the above example, if a new data sample is arrived, whose Bel_{cs} is 0.77, μf_3 of $\mu f_{c(1)}$ is 0.83, and μf_3 of $\mu f_{c(2)}$ is 0.27 which is actually does not match any fuzzy rule in Table 3.3. In this case, the evolving clustering model creates a new cluster (C_3), and assigns this sample meanwhile, a new fuzzy rule is created. Figure 3.9 shows all these details.

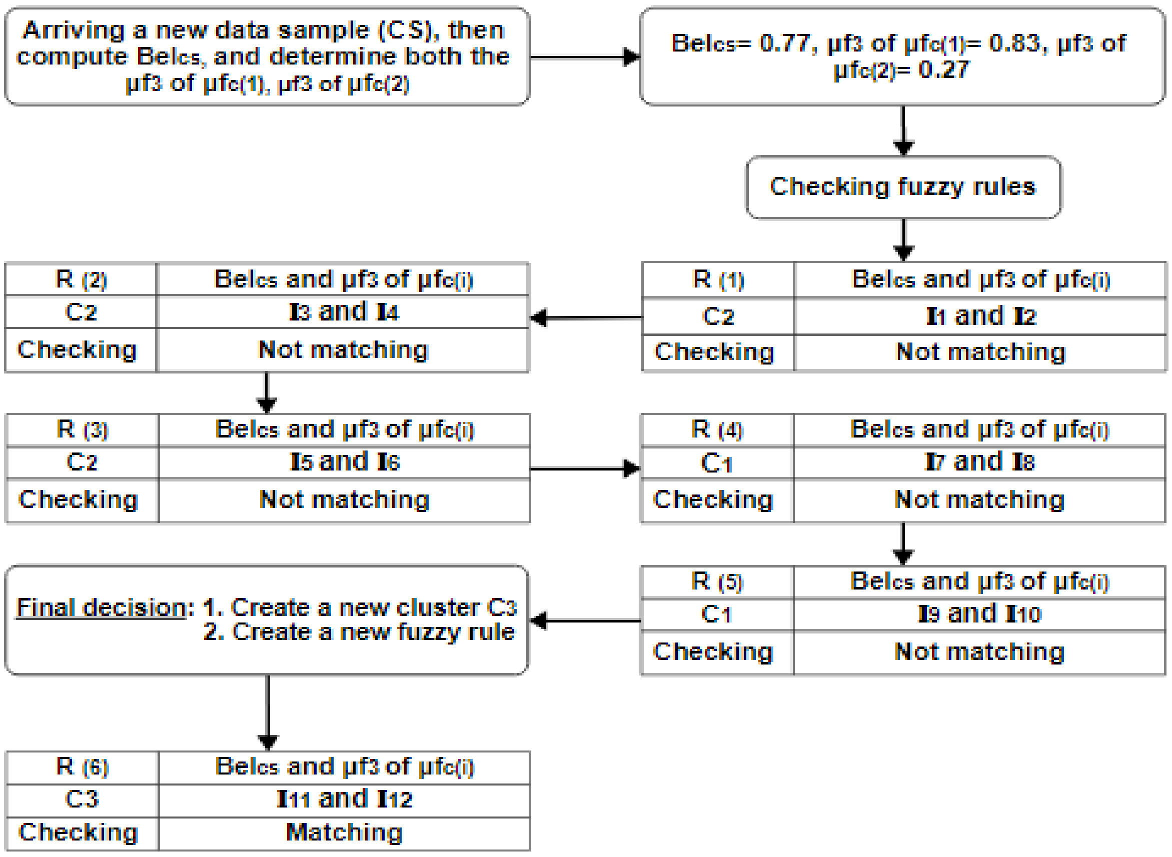

Arriving a new data sample (CS), then compute Belcs, and determine both the µf3 of µfc(1), µf3 of µfc(2)
Belcs= 0.77, µf3 of µfc(1)= 0.83, µf3 of µfc(2)= 0.27
Checking fuzzy rules
R (1) | Belcs and µf3 of µfc(i) | C2 | I1 and I2 | Checking | Not matching
R (2) | Belcs and µf3 of µfc(i) | C2 | I3 and I4 | Checking | Not matching
R (3) | Belcs and µf3 of µfc(i) | C2 | I5 and I6 | Checking | Not matching
R (4) | Belcs and µf3 of µfc(i) | C1 | I7 and I8 | Checking | Not matching
R (5) | Belcs and µf3 of µfc(i) | C1 | I9 and I10 | Checking | Not matching
Final decision: 1. Create a new cluster C3 2. Create a new fuzzy rule
R (6) | Belcs and µf3 of µfc(i) | C3 | I11 and I12 | Checking | Matching

Figure 3.9: Create a new cluster and fuzzy rule to a new data sample (when the previous fuzzy rules are not matching)

Due to the creation of a new fuzzy rule, new intervals are created that are: **I**$_{11}$ and **I**$_{12}$, where **I**$_{11}$ $\leqslant$ 1.0, and **I**$_{12}$ in the range [0.81, 1.0]. Accordingly, the sixth fuzzy rule is built as:

$$\textbf{IF } \boldsymbol{Bel_{cs}} \in \textbf{I}_{11} \textbf{ and } \boldsymbol{\mu f_3} \textbf{ of } \boldsymbol{\mu f_{c(3)}} \in \textbf{I}_{12} \textbf{ THEN } \boldsymbol{CS} \in \boldsymbol{C_3}$$

Regarding the NoC, in the case of assigning all data samples to the clusters that construct earlier exactly, it can compute as, NoC= $\{C_1, C_2, ..., C_i\}$.

On the other hand, in the case of detecting new data samples, and construct a new cluster, the NoC is increased by 1 as, NoC= $\{C_1, C_2, ..., C_{i+1}\}$.

Figure 3.10 explains the main characteristics and benefits from applying the fuzzy rules in the evolving clustering model.

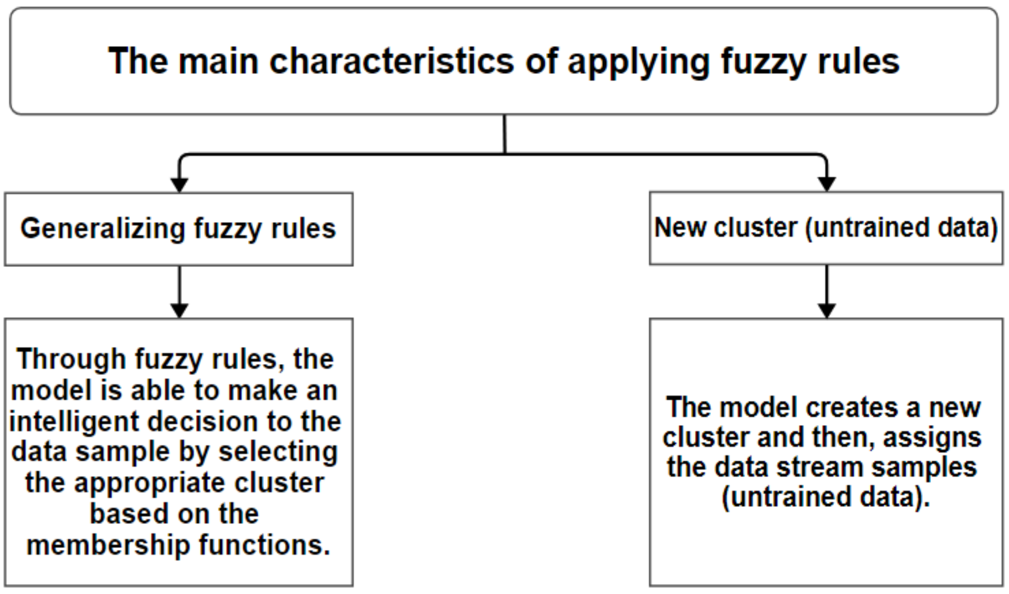

The main characteristics of applying fuzzy rules
Generalizing fuzzy rules
New cluster (untrained data)
Through fuzzy rules, the model is able to make an intelligent decision to the data sample by selecting the appropriate cluster based on the membership functions.
The model creates a new cluster and then, assigns the data stream samples (untrained data).

Figure 3.10: The main characteristics of applying the fuzzy rules

Finally, Figure 3.11 describes the main characteristics and the advantages of applying the evolving clustering model.

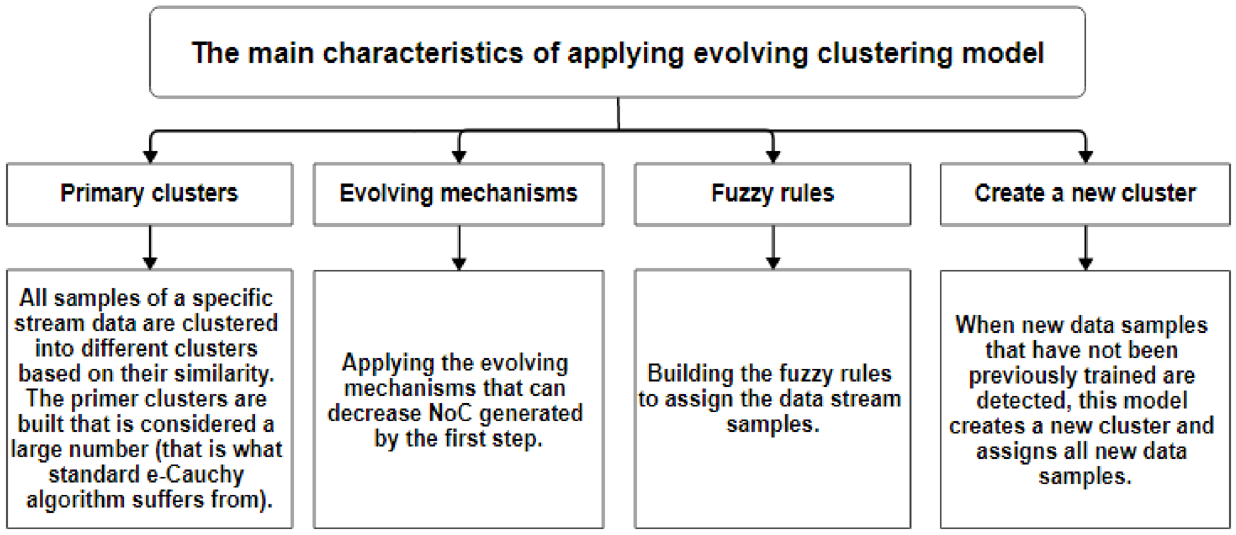

The main characteristics of applying evolving clustering model
Primary clusters
Evolving mechanisms
Fuzzy rules
Create a new cluster
All samples of a specific stream data are clustered into different clusters based on their similarity. The primer clusters are built that is considered a large number (that is what standard e-Cauchy algorithm suffers from).
Applying the evolving mechanisms that can decrease NoC generated by the first step.
Building the fuzzy rules to assign the data stream samples.
When new data samples that have not been previously trained are detected, this model creates a new cluster and assigns all new data samples.

Figure 3.11: The main characteristics of applying the evolving clustering model

3.2.4 Evaluation Stage

It is the final stage in the evolving intelligent system. However, every model (DL and evolving clustering models) are measured by different evaluation measurements.

For the DL model, the results are evaluated by four different measurements that are, accuracy, precision, recall, and F1_score which are computed according to Equations ((2.13), (2.14), (2.15), and (2.16)). Firstly, the standard model that depends on a constant learning rate is performed, and its results are evaluated. Thereafter, the proposed DL model that bases a DLR value is applied, and also its results are evaluated by the same measurements.

Then, the comparison between the results of proposed DL model and the results of the standard model is done.

Although the importance of these measures, the error rate for both models (proposed DL and standard) is also calculated based on Equation (2.17). Then, the enhancement rate through the proposed DL model is calculated according to Equation (2.18).

While for the evolving clustering model, the NoC is generally seen as a strong measure, especially, if the system implements evolving mechanisms as this number will reflect the change in the structure of the system according to the data over many epochs. The NoC after implementing the evolving mechanisms (represented by FHQCs) should be equal to the number of classes for the same dataset. Therefore, the NoC is used as the first measure.

Another important measure is used that is, SC. The SC is computed by Equation (2.19) for both the training data and the testing data before and after the implementation of the evolving mechanisms.

In the same context of the evaluation stage of the proposed system, and during the implementation of this system, the DL model is used as an external evaluator for the evolving clustering model.

Chapter Four

The Experiment Results and Discussion

The Experimental Results and Discussion

4.1 Introduction

This chapter discusses the results that obtains from the implementation of the evolving intelligent system. In this system, two models are applied which are, DL and evolving clustering models. Totally, eleven stream datasets are used to test this system. Therefore, the results of each model are explained in detail and also the results for the specific datasets.

4.2 The Results of DL Model

The first model in the evolving intelligent system is the DL model. This sub-section discusses the results for each dataset through this model.

Firstly, the standard DL model is implemented with a constant learning rate of 0.001, then, the proposed DL model that depends on a DLR is implemented. Accordingly, the proposed model is called as a DLR model, and the results of both models are compared.

Moreover, the error rate for both models is computed. Then, the error enhancement through the DLR model is calculated.

4.2.1 Keystroke Dataset

The keystroke dataset is the first stream dataset tests by this model and the dataset consists of four classes. The error rate from the standard DL model is 0.0604 while the error rate by the DLR model is 0.0521. Consequently, the amount of enhancement rate is 0.0083. As for the accuracy, the standard DL model achieves an accuracy of 93.95%, while, the DLR model attains a higher accuracy of 94.79%.

Moreover, Table 4.1 displays the measurements metrics to the keystroke dataset.

Table 4.1: The measurement metrics for both standard DL and DLR models to keystroke dataset

Item	Model type	Accuracy %	Precision %	Recall %	F1_score %
1.	Standard DL model	93.95	94.05	93.95	94.00
2.	DLR model	94.79	94.81	94.79	94.80

In the context of learning rate, the standard DL model bases on a constant learning rate value (0.001) and this value remains constant i.e. does not change throughout the model implementation. In contrast to the concept of the standard DL model, the DLR model adopts a DLR value, and then trains the network based on this value.

However, for the keystroke dataset, Figure 4.1 shows the change in the learning rate values over epochs through the DLR model and compare it with learning rate of the standard DL model.

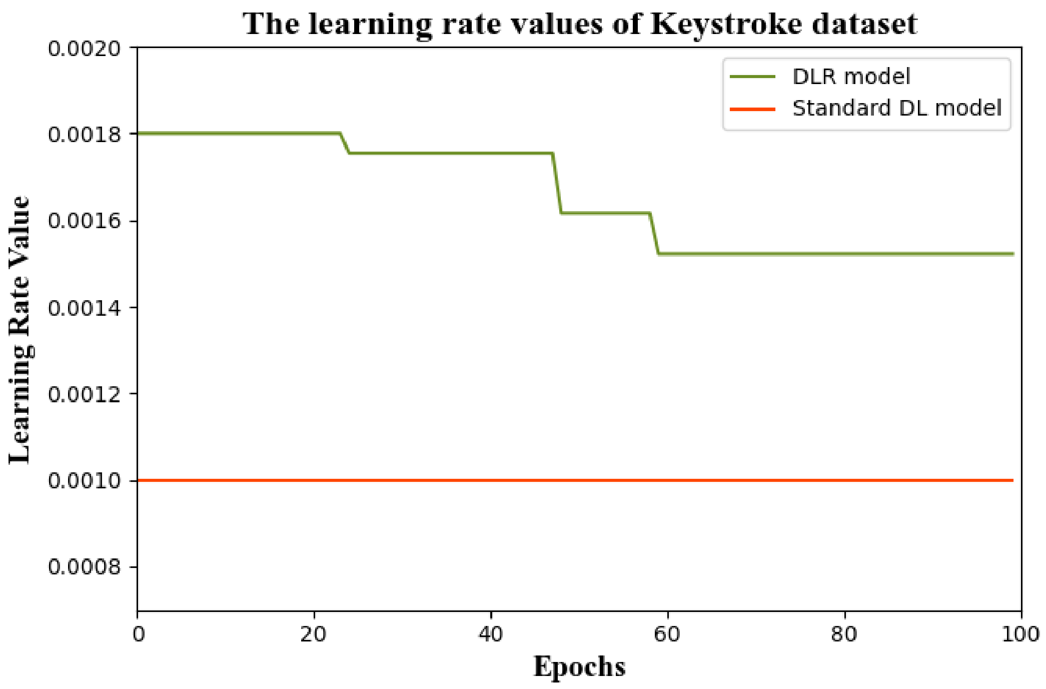


Figure 4.1: The change of the learning rate value over epochs of keystroke dataset for both standard DL and DLR models

4.2.2 Electricity Dataset

The second dataset that is tested by the DL model is the Electricity that contains two classes. The error rate resulting from the standard DL model is 0.1425 whereas, the error rate through the DLR model is 0.1125. Thus, the enhancement rate through DLR model is 0.30. In terms of accuracy, the standard DL model achieves 85.75%, which increases to 88.75% through DLR model.

Furthermore, Table 4.2 lists the measurements metrics to Electricity dataset.

Table 4.2: The measurement metrics for both standard DL and DLR models to Electricity dataset

Item	Model type	Accuracy %	Precision%	Recall %	F1_score %
1.	Standard DL model	85.75	85.67	85.75	85.71
2.	DLR model	88.75	88.74	88.75	88.74

Figure 4.2 explains the change in the learning rate values for both standard DL and DLR models to the Electricity dataset.

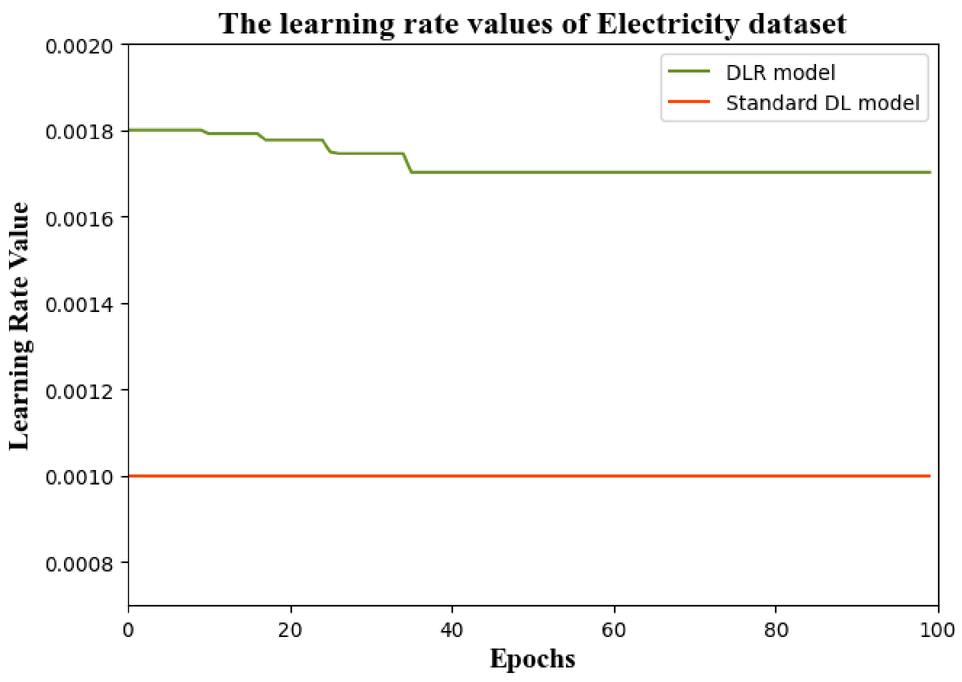


Figure 4.2: The change of the learning rate value over epochs of Electricity dataset for both standard DL and DLR models

4.2.3 NSL-KDD Dataset

The third dataset to evaluate the DL model is the NSL-KDD dataset that has five classes. The error rate resulting from the standard DL model is 0.0165 whilst, it through the DLR model is 0.0132. Hence, the amount of enhancement rate by the DLR model is 0.0033. With regard to the accuracy metric, although the standard DL model achieves an accuracy of 98.34%, DLR model has a higher accuracy than that, reaching 98.67%.

Moreover, Table 4.3 explains the measurements metrics to NSL-KDD dataset.

Table 4.3: The measurement metrics for both standard DL and DLR models for NSL-KDD dataset

Item	Model type	Accuracy %	Precision %	Recall %	F1_score %
1.	Standard DL model	98.34	98.44	98.34	98.39
2.	DLR model	98.67	98.67	98.67	98.67

In connection with the learning rate, Figure 4.3 describes the change in the learning rate values for NSL-KDD dataset.

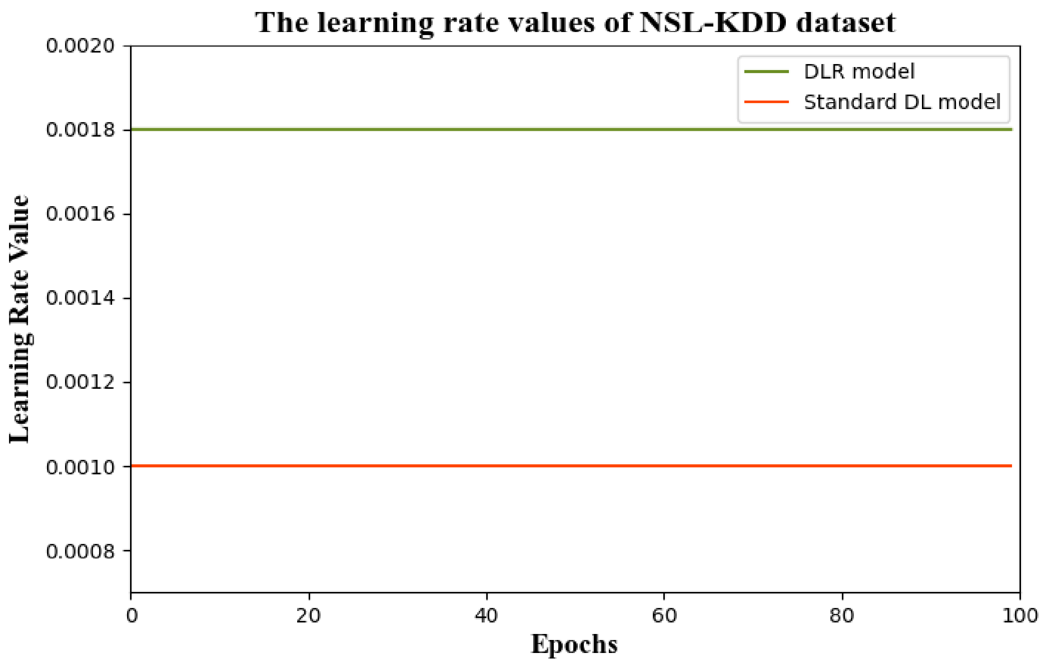


Figure 4.3: The change of the learning rate value over epochs of NSL-KDD dataset for both standard DL and DLR models

4.2.4 HuGaDB Dataset

The fourth dataset is the HuGaDB dataset. Four sub datasets are selected to test the DL model. The results for these four sub datasets as follow:

4.2.4.1 HuGaDB_01_01 Dataset

The first sub dataset is the HuGaDB_01_01 that contains four classes, the error rate is reduced from 0.0164 (by the standard DL model) to 0.0123 by the DLR model thus, the enhancement rate is 0.0041. From the accuracy aspect, the standard DL model realizes an accuracy about 98.35%, and the DLR model attains an accuracy of 98.76%.

4.2.4.2 HuGaDB_05_12 Dataset

The HuGaDB_05_12 is the second sub dataset that is tested by this model which has three classes. Practically, the error rate from the standard DL model is 0.0296, which decreases to 0.025 after applying the DLR model. Consequently, 0.0046 represents the enhancement rate. In terms of accuracy, the standard DL model has an accuracy about 97.04%, on the other hand, 97.49% is the accuracy of the DLR model.

4.2.4.3 HuGaDB_13_11 Dataset

Then, HuGaDB_13_11 sub dataset is evaluated by the DL model that consists of three classes. Firstly, the error rate from the standard DL model is 0.165 whereas, it decreases through the DLR model to 0.0474. Accordingly, 0.1176 is the amount of enhancement rate. Concerning the accuracy, the standard DL model reaches an accuracy about 83.5%, but, the DLR model attains an accuracy as 95.25%.

4.2.4.4 HuGaDB_14_05 Dataset

The last sub dataset of HuGaDB is the HuGaDB_14_05 which contains two classes. In this sub dataset, the error rate of the standard DL model is 0.0056 that decreases to 0.0028 by the DLR model. Therefore, the error rate enhancing is 0.0028. For this sub dataset, the accuracy of the standard DL model is 99.44%, which increases to 99.72% by the DLR model.

Furthermore, Table 4.4 lists the four measurements for all sub datasets of HuGaDB.

Table 4.4: The measurement metrics for both standard DL and DLR models of all HuGaDB sub datasets

Item	dataset	Model type	Accuracy %	Precision %	Recall %	F1_score %
1.	01_01	Standard DL	98.35	98.38	98.35	98.37
		DLR	98.76	98.77	98.76	98.77
2.	05_12	Standard DL	97.04	97.06	97.04	97.05
		DLR	97.49	97.49	97.49	97.49
3.	13_11	Standard DL	83.50	83.56	83.50	83.53
		DLR	95.25	95.32	95.25	95.29
4.	14_05	Standard DL	99.44	99.44	99.44	99.44
		DLR	99.72	99.72	99.72	99.72

Figure 4.4 explains the change in their learning rate values over epochs for all sub datasets of HuGaDB.

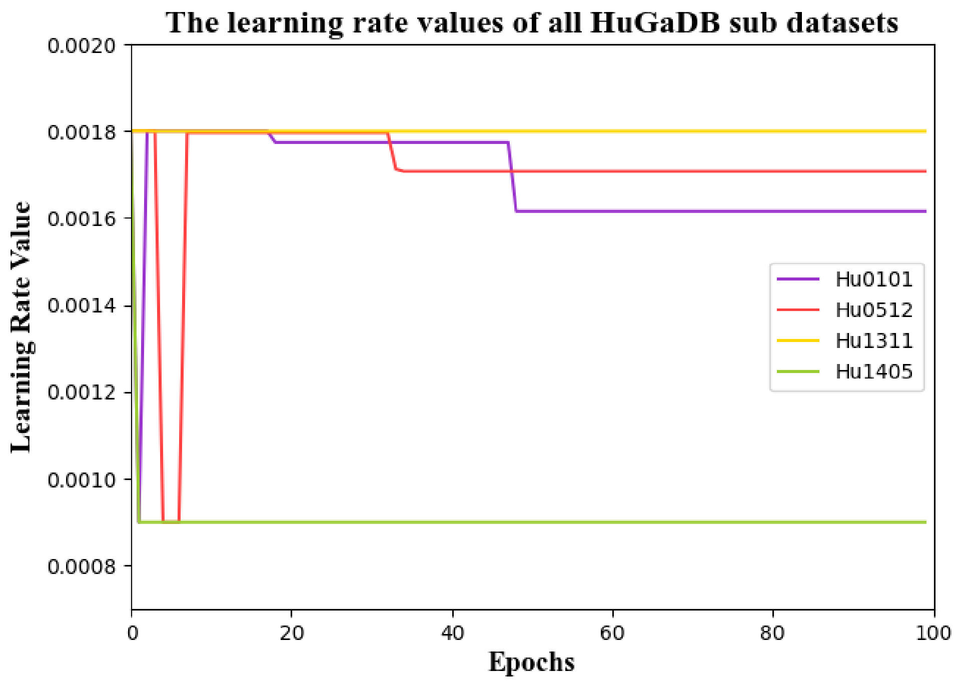


Figure 4.4: The change of the learning rate value over epochs of all HuGaDB sub datasets for both standard DL and DLR models

4.2.5 UCI-HAR Dataset

The UCI-HAR dataset is the next dataset tests by the DL model. This dataset consists of six classes. In this dataset, the error rate from the standard DL model is 0.0159 whereas, the error rate through the DLR model drops to 0.011. Hence, the enhancement rate through this model is 0.0049. With regard to the accuracy, it is in the standard DL model 98.41% that increases to 98.89% through the DLR model. Additionally, Table 4.5 elucidates the measurement metrics to UCI - HAR dataset.

Table 4.5: The measurement metrics for both standard DL and DLR models for UCI-HAR dataset

Item	Model type	Accuracy %	Precision %	Recall %	F1_score %
1.	Standard DL model	98.41	98.42	98.41	98.41
2.	DLR model	98.89	98.89	98.89	98.89

Nevertheless, the change in the learning rate values over epochs to UCI-HAR dataset is depicted in Figure 4.5.

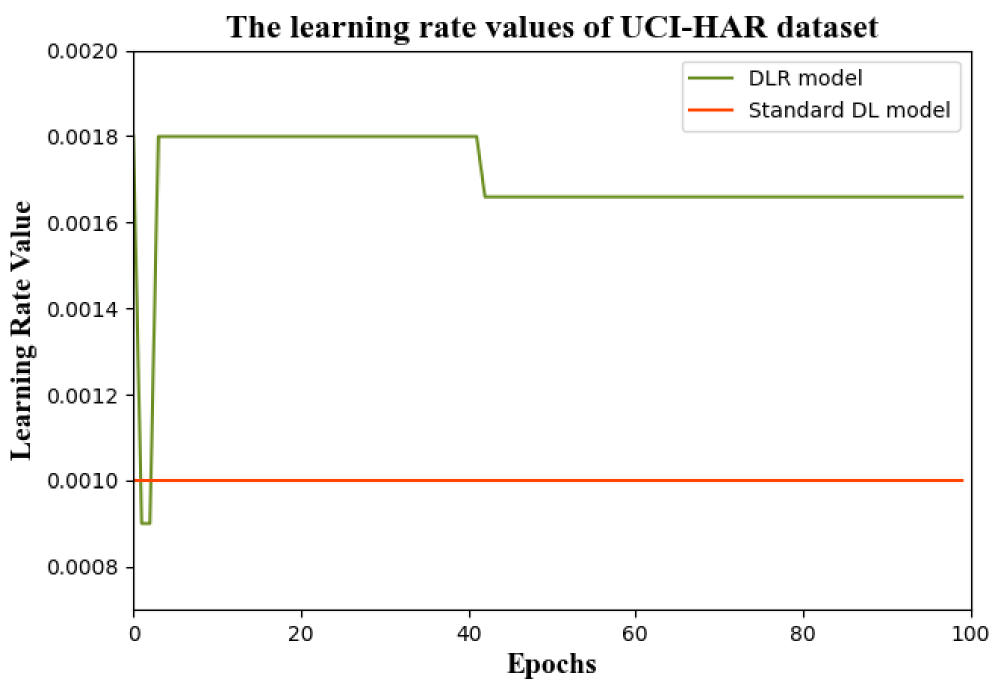


Figure 4.5: The change of the learning rate value over epochs of UCI-HAR dataset for both standard DL and DLR models

4.2.6 Luxembourg Dataset

Afterwards, the Luxembourg dataset is tested by DL model and this dataset has two classes. The enhancement rate for Luxembourg dataset is zero because the error rate for both standard DL and DLR models is zero. More description of these results and the accuracy of the two models.

The four measurements of Luxembourg dataset are described in Table 4.6.

Table 4.6: The measurement metrics for both constant and DL models to Luxembourg dataset

Item	Model type	Accuracy %	Precision %	Recall %	F1_score %
1.	Standard DL model	100.00	100.00	100.00	100.00
2.	DLR model	100.00	100.00	100.00	100.00

Figure 4.6 draws how the learning rate values change over epochs of the Luxembourg dataset and compare these values with the learning rate value through the standard DL model.

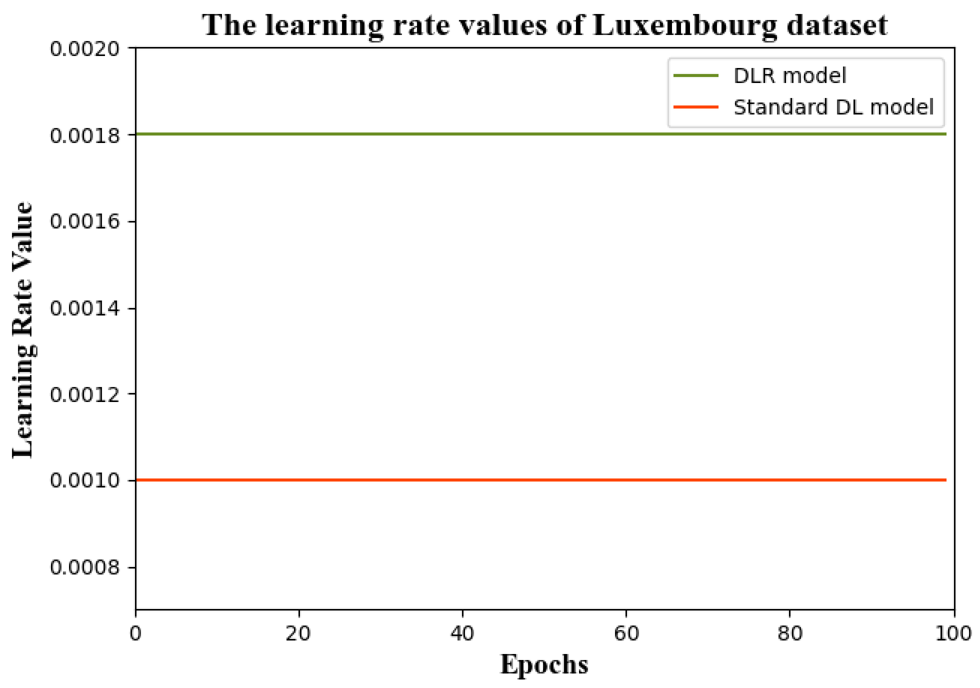


Figure 4.6: The change of the learning rate value over epochs of Luxembourg dataset for both standard DL and DLR models

4.2.7 Keystroke_1 Dataset

Subsequently, the DL model tests the keystroke_1 dataset that consists of three classes. The error rate by the standard DL model is 0.0444 whilst, the enhancement rate through the DLR model is 0.0278. Therefore, the amount of error enhancement is 0.0166. In terms of accuracy, the standard DL model achieves an accuracy of 95.55% that increases to 97.22% through the DLR model.

Additionally, Table 4.7 explains the measurements metrics to keystroke_1 dataset.

Table 4.7: The measurement metrics for both standard DL and DLR models to keystroke_1 dataset

Item	Model type	Accuracy %	Precision %	Recall %	F1_score %
1.	Standard DL model	95.55	95.67	95.55	95.61
2.	DLR model	97.22	97.32	97.22	97.27

Figure 4.7. explains the change of the learning rate values for keystroke_1 dataset.

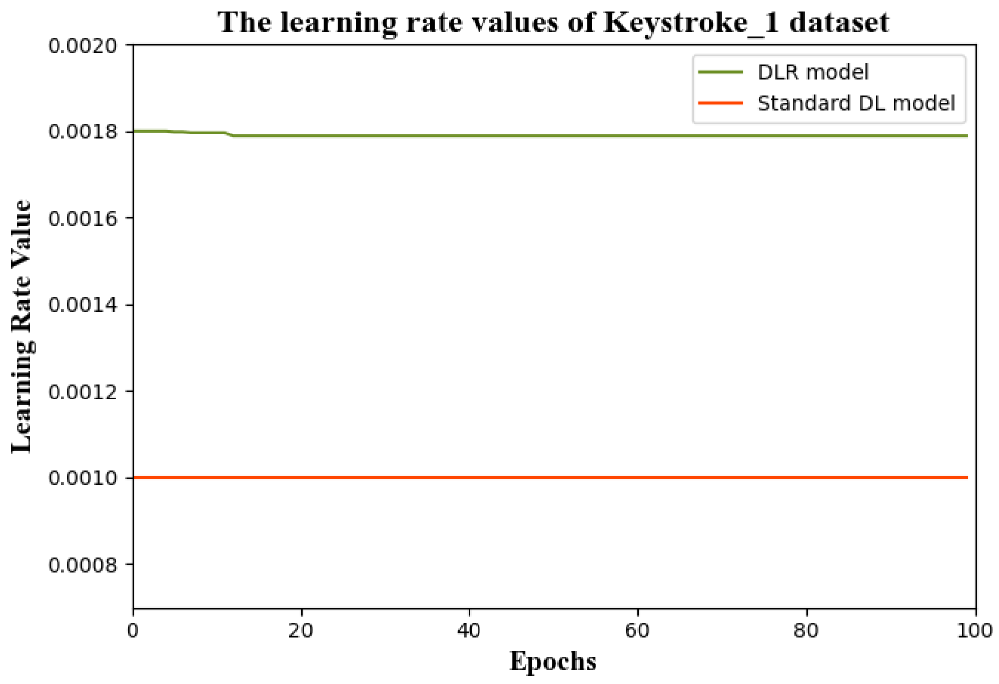


Figure 4.7: The change of the learning rate value over epochs of keystroke_1 dataset for both standard DL and DLR models

4.2.8 HuGaDB_01_01_1 Dataset

It is the last dataset tests by the DL model which contains three classes. The error rate resulting from the standard DL model is 0.0194 while, the error rate after applies the DLR model is 0.0145. Accordingly, the amount of enhancement rate by the DLR model is 0.0049. In terms of accuracy, the standard DL model achieves 98.06%, which becomes 98.54% by the DLR model. Table 4.8 explains the measurements metrics to the HuGaDB01_01_1 dataset.

Table 4.8: The measurement metrics for both standard DL and DLR models to HuGaDB01_01_1 dataset

Item	Model type	Accuracy%	Precision %	Recall %	F1_score %
1.	Standard DL model	98.06	98.10	98.06	98.08
2.	DLR model	98.54	98.55	98.54	98.55

Figure 4.8. shows the change of the learning rate values to this dataset.

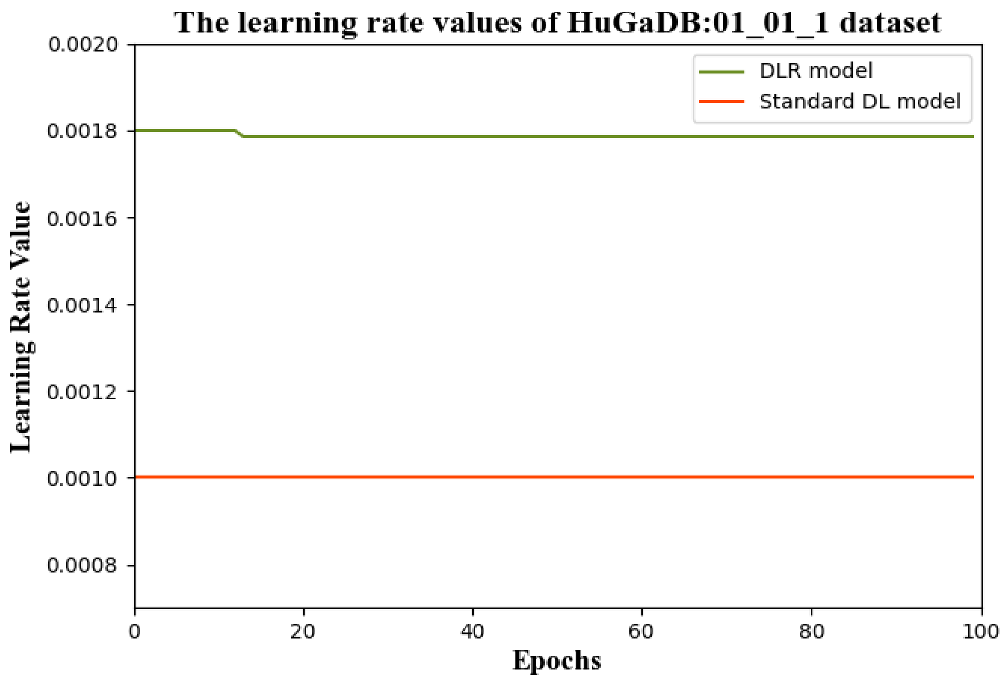


Figure 4.8: The change of the learning rate value over epochs of HuGaDB01_01_1 dataset for both standard DL and DLR models

The comparison of the error that resulted by the standard DL model and DLR model for all stream datasets is depicted in Figure 4.9. However, in this Figure, the results of the DLR model are highlighted in bold

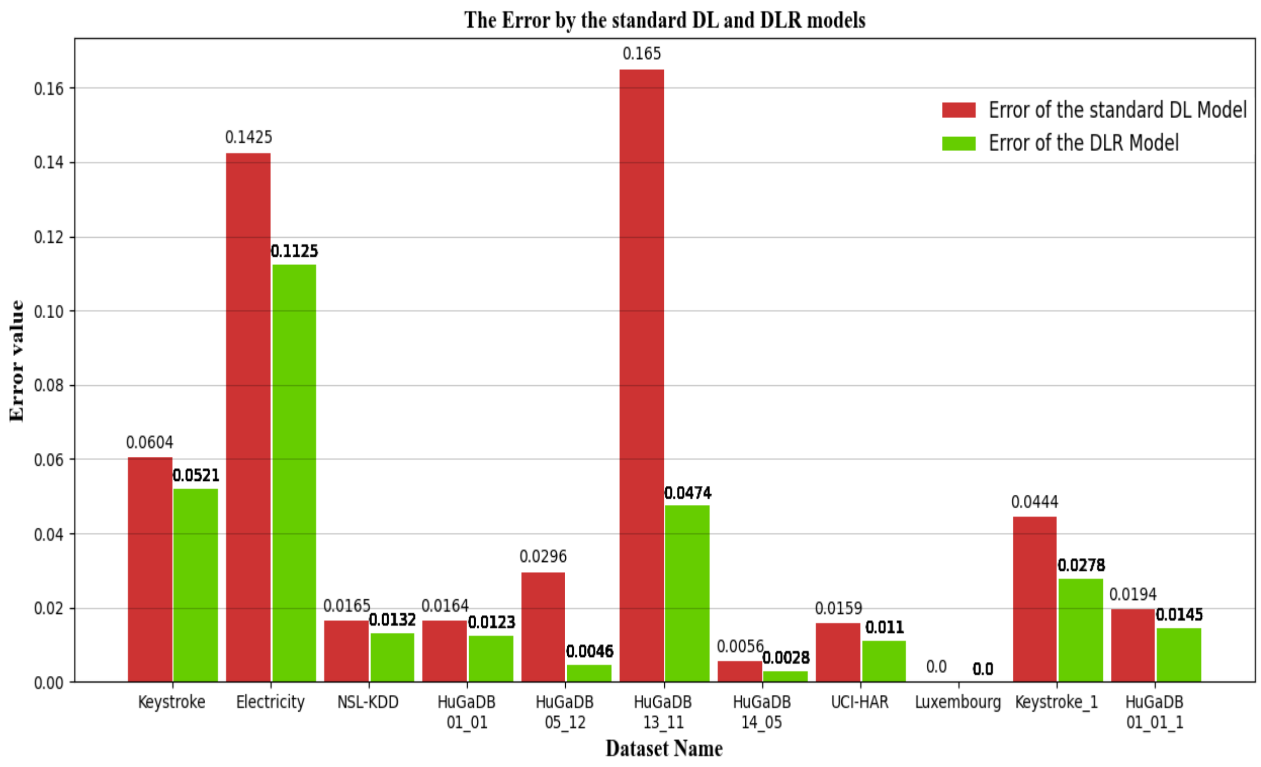


Figure 4.9: The error of the standard DL and DLR models to all stream datasets

While Figure 4.10 shows the accuracy comparison to both models.

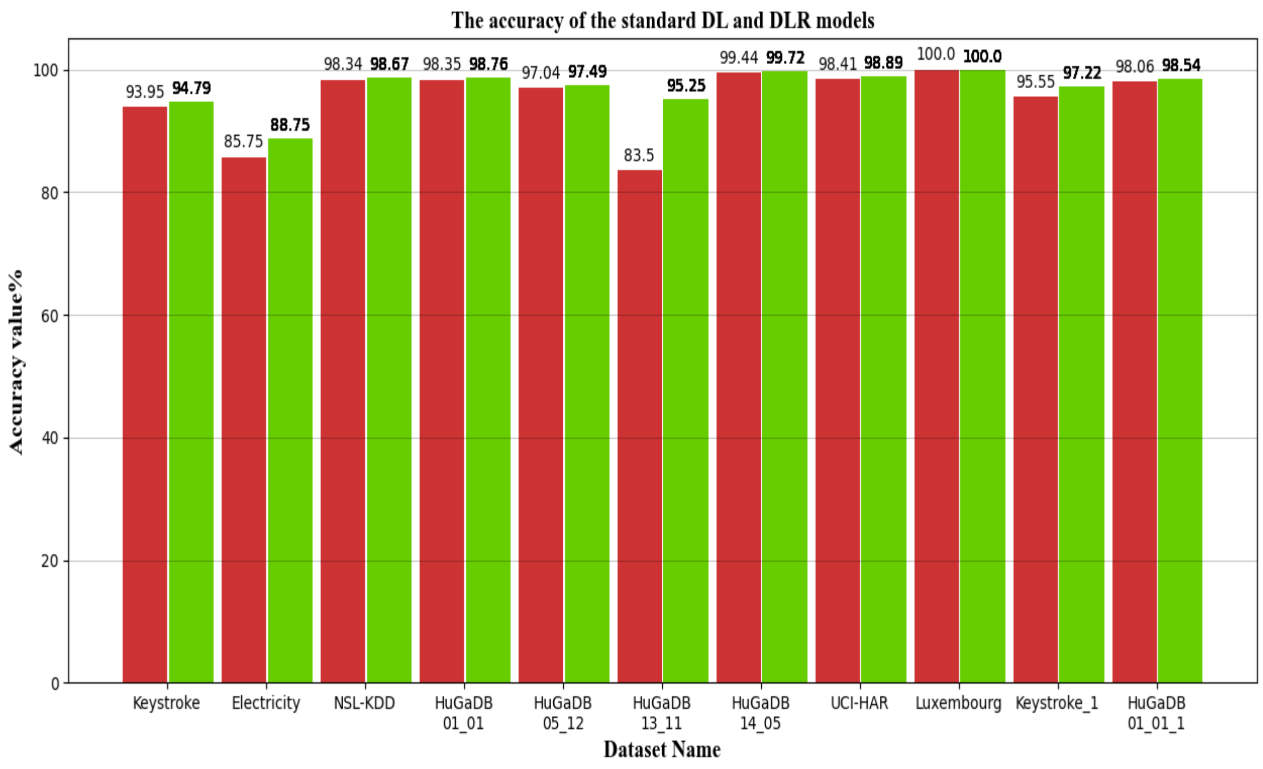


Figure 4.10: The accuracy of the standard DL and DLR models to all stream datasets

The accuracy of DLR model (which adopts a DLR value) outperforms the results of the standard DL model. Table 4.9 displays the accuracy comparison between these two models for all stream datasets.

Table 4.9: The accuracy comparison between the standard DL model and DLR model to all datasets

Item	**Dataset name**	**Accuracy of the standard DL model %**	**Accuracy of the DLR model %**
1.	Keystroke	93.95	94.79
2.	Electricity	85.75	88.75
3.	NSL-KDD	98.34	98.67
4.	HuGaDB _01_01	98.35	98.76
5.	HuGaDB _05_12	97.04	97.49
6.	HuGaDB _13_11	83.50	95.25
7.	HuGaDB _14_05	99.44	99.72
8.	UCI-HAR	98.41	98.89
9.	Luxembourg	100.00	100.00
10.	Keystroke_1	95.55	97.22
11.	HuGaDB _01_01_1	98.06	98.54

The accuracy of the DLR model to keystroke dataset is 94.79% that outperforms to the model proposed by (Andrean et al., 2020) which achieved an accuracy of 91.67%, as well as the model presented by (Javed et al., 2020) that attained an accuracy of 89%.

Based on the accuracy achieves by the DLR model for Electricity dataset that is 88.75%, it superiors the model proposed by (ABBASI et al., 2021) which achieved an accuracy of 81.06%.

According to the accuracy of NSL-KDD dataset which is 98.67%, the DLR model outperforms the model suggested by (Abrar et al., 2020) which achieved an accuracy about 98.19%, the model presented by (Kyatham et al., 2020) that attained an accuracy of 97.05%, and also the model proposed by (Ahanger et al., 2021) which achieved an accuracy of 97.97%.

With regard to the accuracy achieved by applying the DLR model to HuGaDB dataset, the overall accuracy to the four sub datasets of the HuGaDB dataset is 97.80%. Therefore, this DLR model superiors many previous models such as, our previous model that achieved an accuracy of 91.7% (Al-khamees et al., 2022), the model presented by (Javeed et al., 2021) which achieved an accuracy of 92.5 %, the model suggested by (Sun et al., 2018) that attained an accuracy of 88.0%, and the model proposed by (Kumari et al., 2020) that achieved an accuracy 91.1%.

According to the accuracy achieves by the DLR model to UCI-HAR dataset, which is 98.89%, this model outperforms previous models (on the UCI-HAR dataset) such as, the model proposed by (Bozkurt, 2021) which achieved an accuracy of 96.81%, the model suggested by (Wan et al., 2020) that attained an accuracy up of 86.83%, in addition to the model presented in (Myo et al., 2019) which achieved an accuracy as 98.32%.

Based on the accuracy of the DLR model to Luxembourg dataset that is 100%, this model superiors to the model suggested by (Pinagé et al., 2020) after it achieved 97.53%.

Table 4.10 explains the accuracy comparison between the DLR model and the previous models.

Table 4.10: The accuracy comparison of the DLR model with previous models

Item	Dataset	Ref. / Year	Accuracy %	DLR accuracy %
1.	Keystroke	(Andrean et al., 2020)	91.67	94.79
2.	Keystroke	(Javed et al., 2020)	89	94.79
3.	Electricity	(ABBASI et al., 2021)	81.06	88.75
4.	NSL-KDD	(Abrar et al., 2020)	98.19	98.67
5.	NSL-KDD	(Kyatham et al., 2020)	97.05	98.67
6.	NSL-KDD	(Ahanger et al., 2021)	97.97	98.67
7.	HuGaDB	(Sun et al., 2018)	88.0	97.80
8.	HuGaDB	(Kumari et al., 2020)	91.1	97.80

9.	HuGaDB	(Javeed et al., 2021)	92.5	97.80
10.	HuGaDB	(Al-khamees et al., 2022)	91.7	97.80
11.	UCI-HAR	(Myo et al., 2019)	98.32	98.89
12.	UCI-HAR	(Wan et al., 2020)	86.83	98.89
13.	UCI-HAR	(Bozkurt, 2021)	96.81	98.89
14.	Luxembourg	(Pinagé et al., 2020)	97.53	100

4.3 The Results of Evolving Clustering Model

The second model of the evolving intelligent system is the evolving clustering model that depends on the clustering technique. This sub-section explains the results for every stream dataset for this model.

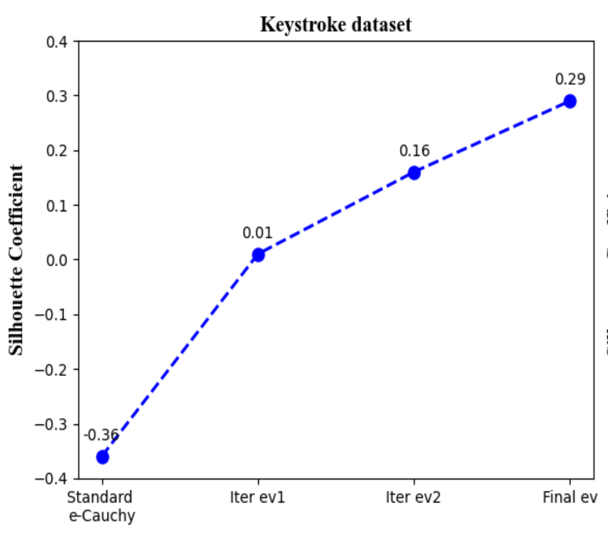
First of all, for every stream dataset, the Θ_{den} threshold is specified. Then, in terms of the evaluation step, firstly, SC measure is applied which used before and after the evolving process for both the training and testing dataset separately. Secondly, NoC is the second measure which used in the evolving clustering model. Accordingly, the step of the primer clusters construction is explained.

Thereafter, since the evolving clustering model applies the evolving mechanisms, this sub-section of results displays how the numbers of clusters is evolved over epochs which actually differ from one dataset to another according to the behavior of the data samples, the density value...etc.

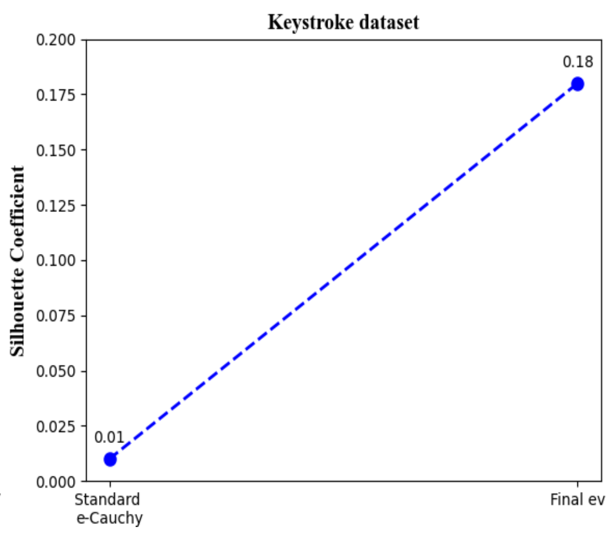
Finally, this sub-section explains the intervals as well as the fuzzy rules for each cluster.

4.3.1 Keystroke Dataset

Firstly, the evolving clustering model tests the keystroke dataset that contains four classes. Θ_{den} threshold of e-Cauchy algorithm sets to 0.055. The SC results for both training and testing phases are described in Figure 4.11.



(a)



(b)

Figure 4.11: (a) SC of training to keystroke dataset (b) SC of testing to keystroke dataset

As for NoC to this dataset, the standard e-Cauchy algorithm generates initially 592 clusters, then, after applying the evolving mechanisms, NoC reduces to 39 clusters, and then to 8 clusters, and finally, it becomes 4 clusters. Table 4.11 displays the details of the primer clusters construction for keystroke dataset.

Table 4.11: The description of primer clusters construction for keystroke dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.077601	Yes	C_1
2.	0.001146	Yes	C_2
3.	0.001290	No	C_2
4.	0.022175	Yes	C_3
5.	0.001676	No	C_3
6.	0.038295	No	C_3
7.	0.034471	Yes	C_4
8.	0.001437	No	C_4
9.	0.047535	No	C_4
10.	0.030900	No	C_4
11.	0.080139	No	C_4
12.	0.001570	No	C_4

13.	0.036016	No	C_4
14.	0.033753	No	C_4
15.	0.001627	Yes	C_5
…	…	…	…
…	…	…	…

While the decreasing of NoC for keystroke dataset during the evolving iterations of this model is shown in Figure 4.12.

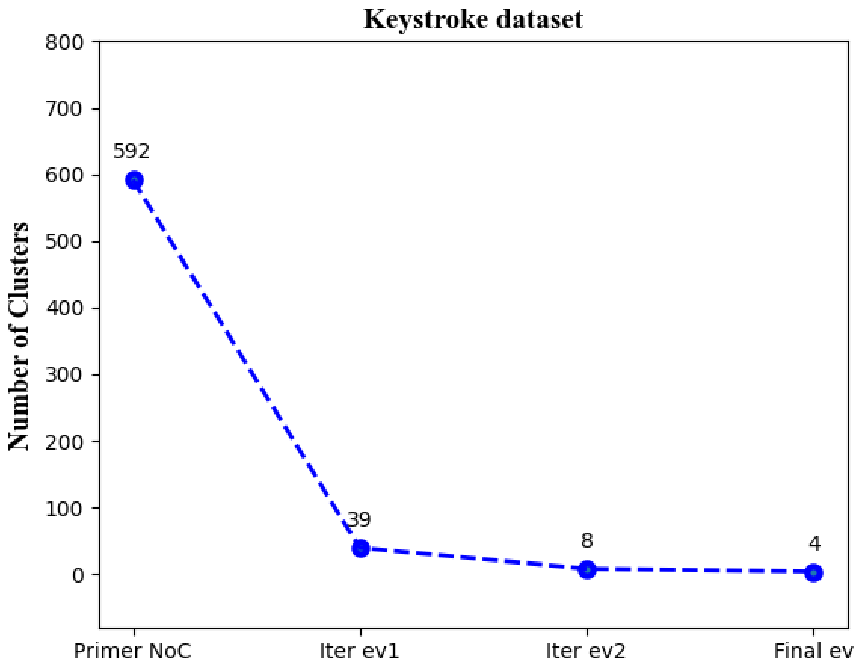


Figure 4.12: The decreasing of NoC for keystroke dataset

Additionally, the SC values and NoC are depicted in Table 4.12.

Table 4.12: SC and NoC for keystroke dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.36	0.01	592
2.	Iter ev1	0.01	---	39
3.	Iter ev2	0.16	---	8
4.	Final ev	0.29	0.18	4

It evident from Table 4.12, as a result of applying the standard e-Cauchy, SC of the training is -0.36, while SC of testing is 0.01, and NoC is 592. During the first iteration of evolving (Iter ev_1), SC of training is 0.01 and NoC is 39. In the second iteration of evolving (Iter ev_2), SC of training is 0.16 and NoC is 8. The final evolving (Final ev), SC of training is 0.29 while, SC of testing is 0.18, and NoC is 4.

Regarding the fuzzy rules, the construction of these rules done after applying the evolving mechanisms. In the same context, the intervals are displayed in Table 4.13, while the fuzzy rules that constructed by the evolving clustering model are described in Table 4.14.

Table 4.13: The intervals of keystroke dataset

Interval No.	**Interval details**	**Interval No.**	**Interval details**
I_1	[0, 0.0373]	**I_8**	[0.47]
I_2	[0, 0.12]	**I_9**	[0.0579, 0.150]
I_3	[0.0371, 0.0372]	**I_{10}**	[0.465, 0.7]
I_4	[0.13, 0.14]	**I_{11}**	[0.153, 0.154]
I_5	[0.0374, 0.0578]	**I_{12}**	[0.697, 0.699]
I_6	[0.13, 0.464]	**I_{13}**	[0.151, 1.0]
I_7	[0.0574, 0.0575]	**I_{14}**	[0.71, 0.89]

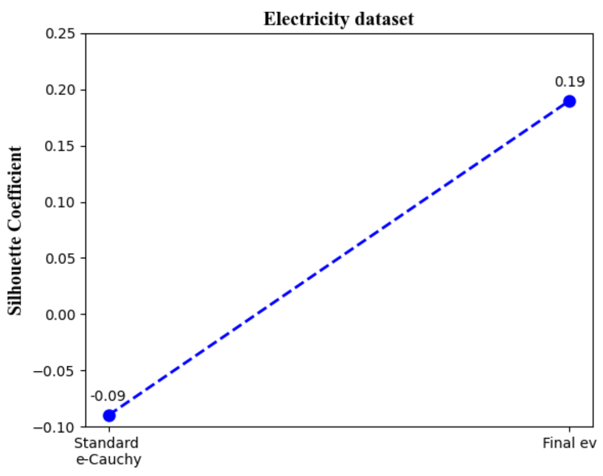
Table 4.14: The fuzzy rules of keystroke dataset

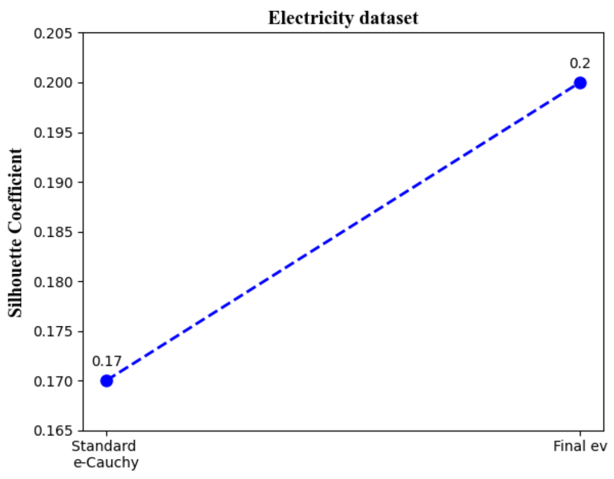
Rule No.	**Rule description**
R_1	IF $Bel_{cs} \in$ **I_1** and μf_3 of $\mu f_{c(1)} \in$ **I_2** THEN $CS \in C_1$
R_2	IF $Bel_{cs} \in$ **I_3** and μf_3 of $\mu f_{c(1)} \in$ **I_4** THEN $CS \in C_1$
R_3	IF $Bel_{cs} \in$ **I_5** and μf_3 of $\mu f_{c(2)} \in$ **I_6** THEN $CS \in C_2$
R_4	IF $Bel_{cs} \in$ **I_7** and μf_3 of $\mu f_{c(2)} =$ **I_8** THEN $CS \in C_2$
R_5	IF $Bel_{cs} \in$ **I_9** and μf_3 of $\mu f_{c(3)} \in$ **I_{10}** THEN $CS \in C_3$
R_6	IF $Bel_{cs} \in$ **I_{11}** and μf_3 of $\mu f_{c(3)} \in$ **I_{12}** THEN $CS \in C_3$
R_7	IF $Bel_{cs} \in$ **I_{13}** and μf_3 of $\mu f_{c(4)} \in$ **I_{14}** THEN $CS \in C_4$

4.3.2 Electricity Dataset

After that, the evolving clustering model tests the second dataset that is, Electricity which consists of two classes. The effective threshold Θ_{den} sets to

0.001101 for this dataset. The results of SC values for training and testing phases are depicted in Figure 4.13.





(a) *(b)*

Figure 4.13: (a) SC of training to Electricity dataset (b) SC of testing to Electricity dataset

Within NoC regard, the evolving clustering model generated firstly 29 clusters, then it reduces to two clusters that are the actual number of classes in Electricity dataset. More details to construct the primer clusters are listed in Table 4.15.

Table 4.15: The description of primer clusters construction for Electricity dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.010947	Yes	C_1
2.	0.001875	No	C_1
3.	0.001580	No	C_1
4.	0.001492	No	C_1
5.	0.001342	No	C_1
6.	0.001304	No	C_1
7.	0.001304	No	C_1
8.	0.001174	No	C_1
9.	0.001257	No	C_1
10.	0.001157	No	C_1
11.	0.001147	No	C_1

12.	0.001209	No	C_1
13.	0.001190	No	C_1
14.	0.001129	No	C_1
15.	0.001125	No	C_1
16.	0.001151	No	C_1
17.	0.001129	No	C_1
18.	0.001200	No	C_1
19.	0.001176	No	C_1
20.	0.001185	No	C_1
21.	0.001216	No	C_1
22.	0.001177	No	C_1
23.	0.001129	No	C_1
24.	0.001199	No	C_1
25.	0.001144	No	C_1
26.	0.001143	No	C_1
27.	0.001154	No	C_1
28.	0.001220	No	C_1
29.	0.001199	No	C_1
30.	0.001152	No	C_1
31.	0.001203	No	C_1
32.	0.001187	No	C_1
33.	0.001091	Yes	C_2
34.	0.001186	No	C_2
…	…	…	…
…	…	…	…

Whereas, Figure 4.14 displays the decreasing of NoC for Electricity dataset.

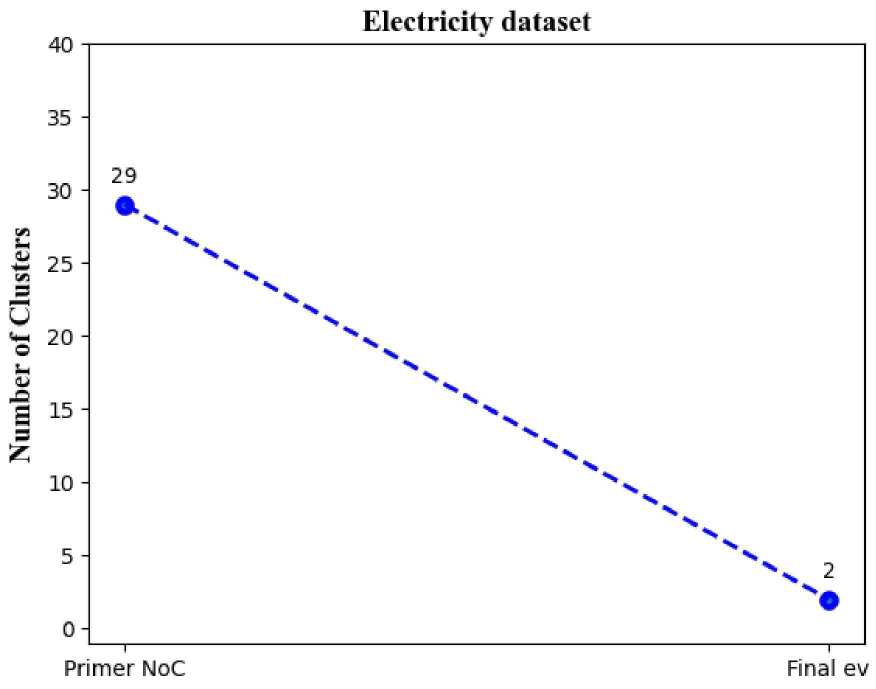


Figure 4.14: The decreasing of NoC for Electricity dataset

Moreover, SC and NoC values of Electricity dataset are illustrated in Table 4.16.

Table 4.16: SC and NoC for Electricity dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.09	0.17	29
2.	Final ev	0.19	0.20	2

Concerning the fuzzy rules for Electricity dataset, the intervals are detailed in Table 4.17, and the fuzzy rules are listed in Table 4.18.

Table 4.17: The intervals of Electricity dataset

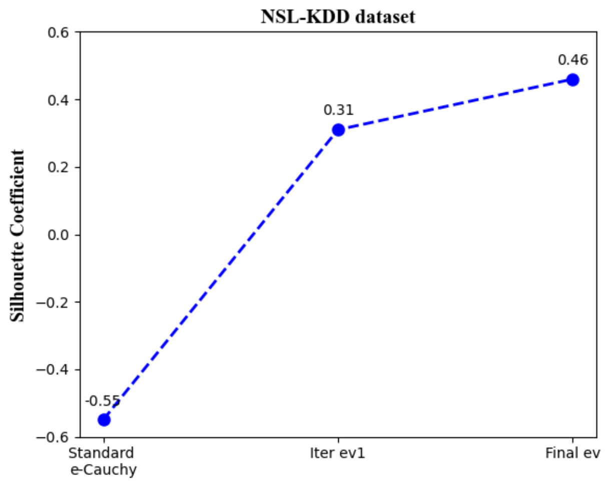
Interval No.	Interval details	Interval No.	Interval details
I_1	[0, 0.303]	I_5	[0.301, 0.302]
I_2	[0, 0.417]	I_6	[0.423]
I_3	[0.297, 0.301]	I_7	[0.304, 1.0]
I_4	[0.409, 0.42]	I_8	[0.418, 0.925]

Table 4.18: The fuzzy rules of Electricity dataset

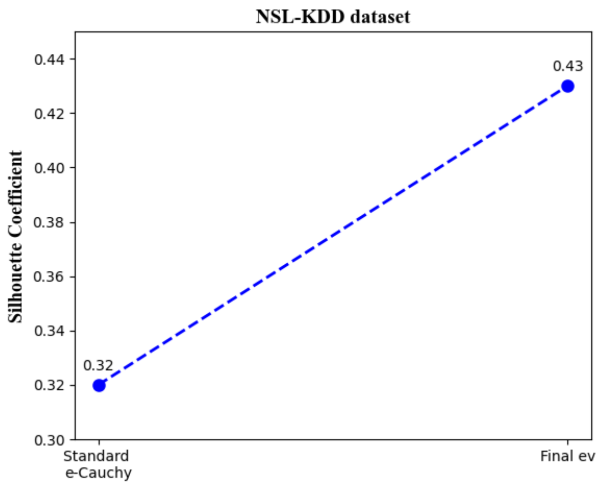
Rule No.	Rule description
R_1	IF Bel_{cs} ∈ $\mathbf{I}_1$ and μf_3 of $\mu f_{c(1)}$ ∈ $\mathbf{I}_2$ THEN $CS \in C_1$
R_2	IF Bel_{cs} ∈ $\mathbf{I}_3$ and μf_3 of $\mu f_{c(1)}$ ∈ $\mathbf{I}_4$ THEN $CS \in C_1$
R_3	IF Bel_{cs} ∈ $\mathbf{I}_5$ and μf_3 of $\mu f_{c(1)}$ = $\mathbf{I}_6$ THEN $CS \in C_1$
R_4	IF Bel_{cs} ∈ $\mathbf{I}_7$ and μf_3 of $\mu f_{c(2)}$ ∈ $\mathbf{I}_8$ THEN $CS \in C_2$

4.3.3 NSL-KDD Dataset

Thereafter, the evolving clustering model evaluates the NSL-KDD dataset. In this dataset, Θ_{den} is determined to 0.0038. However, this dataset consists of five classes. The SC measurements are detailed in Figure 4.15.



(a)



(b)

Figure 4.15: (a) SC of training to NSL-KDD dataset (b) SC of testing to NSL-KDD dataset

NoC for NSL-KDD dataset is 355 clusters, which reduces to 7 clusters and lastly to 5 clusters. In this context, the primer clusters are constructed depending on the density values as explains in Table 4.19.

Table 4.19: The description of primer clusters construction for NSL-KDD dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.00440	Yes	C_1
2.	0.000181	Yes	C_2

3.	0.000185	No	C_2
4.	0.002774	Yes	C_3
5.	0.000181	No	C_3
6.	0.007012	No	C_3
7.	0.002790	No	C_3
8.	0.000182	No	C_3
9.	0.000182	No	C_3
10.	0.003174	Yes	C_4
11.	0.000182	No	C_4
12.	0.000175	No	C_4
13.	0.003602	No	C_4
14.	0.002804	Yes	C_5
15.	0.000181	No	C_5
…	…	…	…
…	…	…	…

Whilst, the decreasing of NoC over epochs are depicted in Figure 4.16.

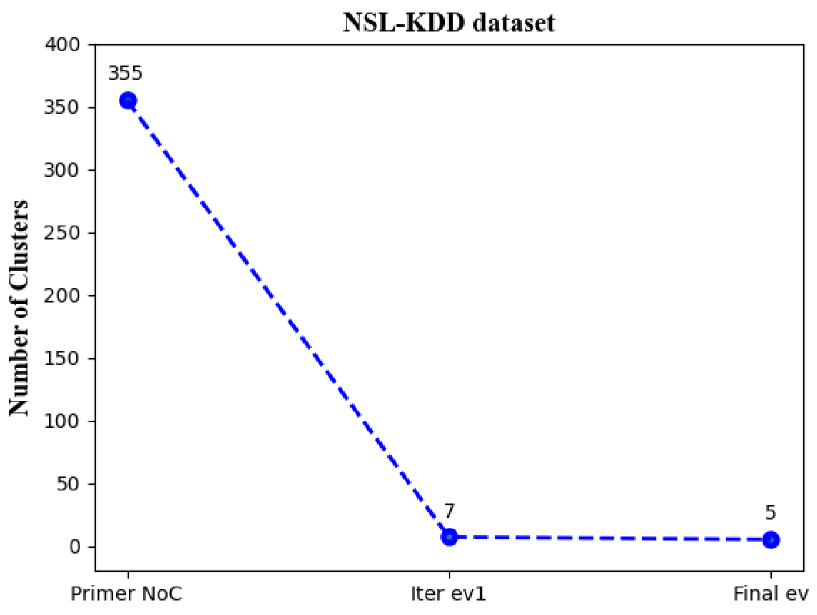


Figure 4.16: The decreasing of NoC for NSL-KDD dataset

Furthermore, the values of both SC and NoC are listed in Table 4.20.

Table 4.20: SC and NoC for NSL-KDD dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.55	0.32	355
2.	Iter ev1	0.13	---	7
3.	Final ev	0.46	0.43	5

As for the fuzzy rules of NSL-KDD dataset, Table 4.21 describes the intervals, and Table 4.22 explains the fuzzy rules to this dataset.

Table 4.21: The intervals of NSL-KDD dataset

Interval No.	Interval details	Interval No.	Interval details
$\mathbf{I}_1$	[0, 0.0987]	$\mathbf{I}_8$	[0.239, 0.240]
$\mathbf{I}_2$	[0, 0.117]	$\mathbf{I}_9$	[0.621, 1.0]
$\mathbf{I}_3$	[0.0978, 0.0987]	$\mathbf{I}_{10}$	[0.239, 0.892]
$\mathbf{I}_4$	[0.117, 0.118]	$\mathbf{I}_{11}$	[0.99452, 0.99458]
$\mathbf{I}_5$	[0.0988, 0.620]	$\mathbf{I}_{12}$	[0.802, 0.803]
$\mathbf{I}_6$	[0.118, 0.238]	$\mathbf{I}_{13}$	[0.707122, 0.707123]
$\mathbf{I}_7$	[0.586, 0.620]	$\mathbf{I}_{14}$	[0.841]

Table 4.22: The fuzzy rules of NSL-KDD dataset

Rule No.	Rule description
R_1	IF $Bel_{cs} \in \mathbf{I}_1$ and μf_3 of $\mu f_{c(1)} \in \mathbf{I}_2$ THEN $CS \in C_1$
R_2	IF $Bel_{cs} \in \mathbf{I}_3$ and μf_3 of $\mu f_{c(1)} \in \mathbf{I}_4$ THEN $CS \in C_1$
R_3	IF $Bel_{cs} \in \mathbf{I}_5$ and μf_3 of $\mu f_{c(2)} \in \mathbf{I}_6$ THEN $CS \in C_2$
R_4	IF $Bel_{cs} \in \mathbf{I}_7$ and μf_3 of $\mu f_{c(2)} \in \mathbf{I}_8$ THEN $CS \in C_2$
R_5	IF $Bel_{cs} \in \mathbf{I}_9$ and μf_3 of $\mu f_{c(3)} \in \mathbf{I}_{10}$ THEN $CS \in C_3$
R_6	IF $Bel_{cs} \in \mathbf{I}_{11}$ and μf_3 of $\mu f_{c(4)} \in \mathbf{I}_{12}$ THEN $CS \in C_4$
R_7	IF $Bel_{cs} \in \mathbf{I}_{13}$ and μf_3 of $\mu f_{c(5)} = \mathbf{I}_{14}$ THEN $CS \in C_5$

4.3.4 HuGaDB Dataset

The fourth dataset is HuGaDB. From this main dataset, four sub datasets are tested by the evolving clustering model. Practically, the Θ_{den} threshold sets to 0.0038 for all these sub datasets. However, the results as follows:

4.3.4.1 HuGaDB_01_01 Dataset

The first sub dataset is HuGaDB_01_01. The SC values for this sub dataset after applying the evolving clustering model are demonstrated in Figure 4.17.

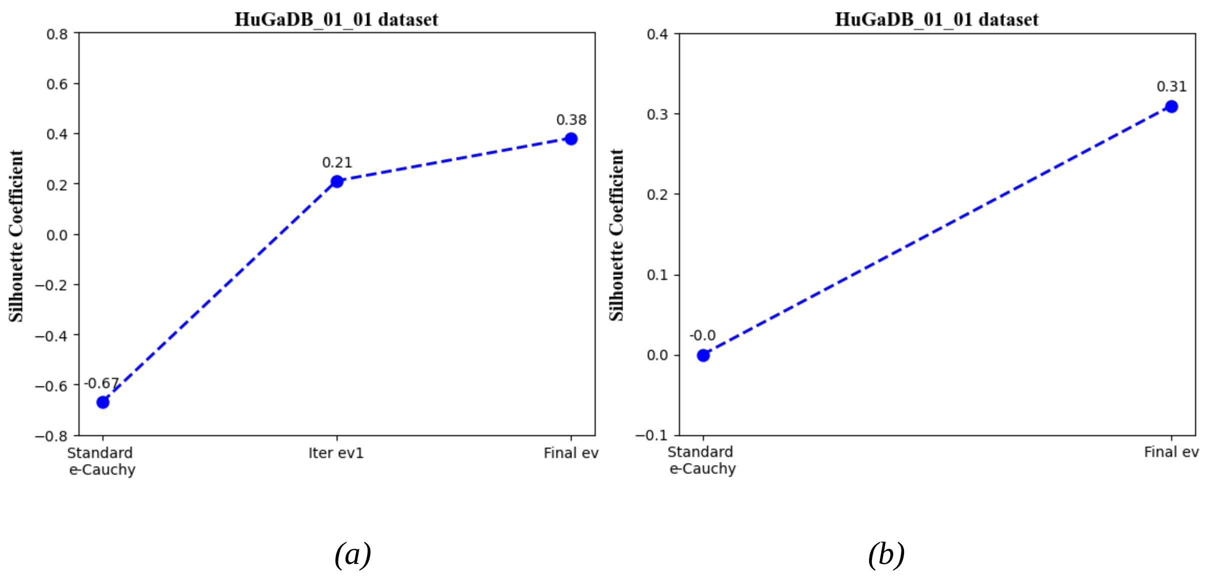


Figure 4.17: (a) SC of training to HuGaDB_01_01 dataset (b) SC of testing to HuGaDB_01_01

With regard to NoC, the e_Cauchy algorithm initially generates 952 clusters, which decreases to 9 clusters and finally, four clusters. According to the primer clusters that generates by above algorithm, Table 4.23 shows the construction step of these clusters.

Table 4.23: The description of primer clusters construction for HuGaDB_01_01 dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.005094	Yes	C_1
2.	0.000491	Yes	C_2
3.	0.000494	No	C_2
4.	0.004878	No	C_2
5.	0.000493	No	C_2
6.	0.000493	Yes	C_3
7.	0.000494	No	C_3
8.	0.000495	No	C_3

9.	0.006243	No	C_3
10.	0.000491	No	C_3
11.	0.000494	No	C_3
12.	0.000480	Yes	C_4
13.	0.000488	No	C_4
14.	0.000489	No	C_4
15.	0.000478	No	C_4
…	…	…	…
…	…	…	…

The decreasing of NoC over epochs are illustrated in Figure 4.18.

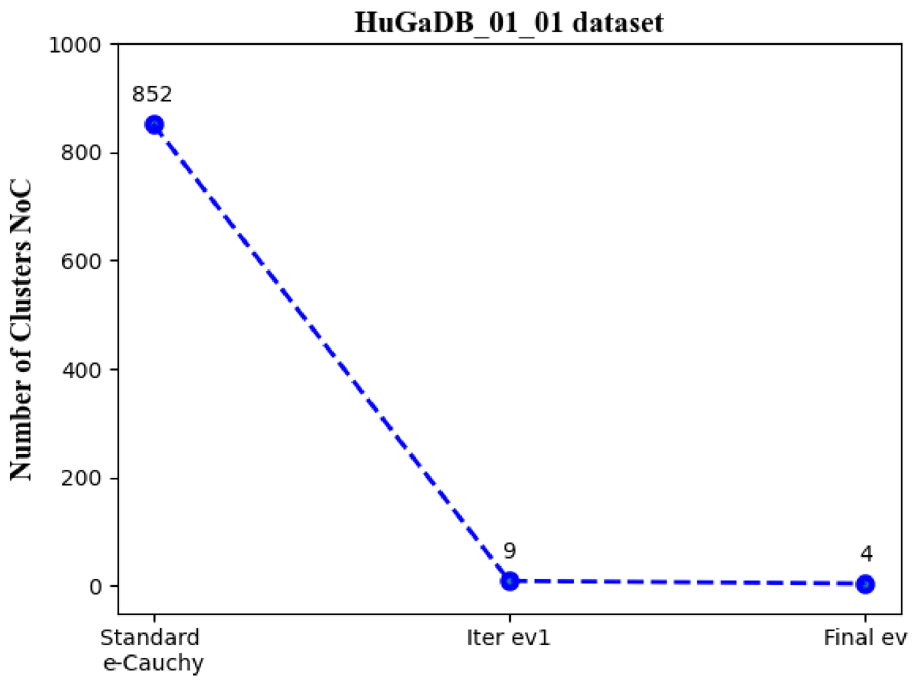


Figure 4.18: The decreasing of NoC for HuGaDB _01_01dataset

The SC details and also NoC are described in Table 4.24.

Table 4.24: SC and NoC for HuGaDB_01_01dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.67	-0.00	852
2.	Iter ev1	0.21	---	9
3.	Final ev	0.38	0.31	4

The intervals and the fuzzy rules are illustrated in Tables 4.25, and 4.26 respectively.

Table 4.25: The intervals of HuGaDB _01_01 dataset

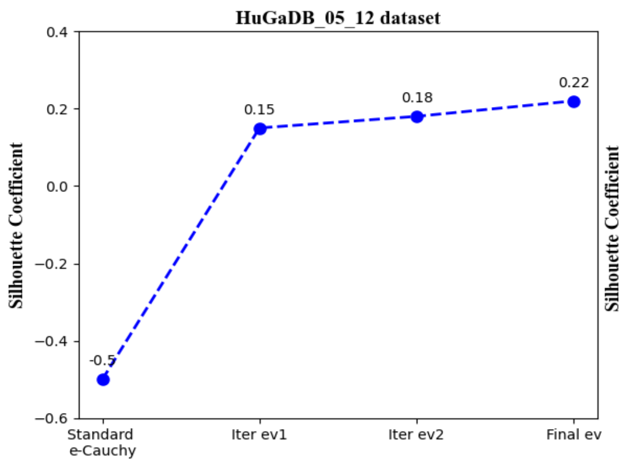
Interval No.	Interval details	Interval No.	Interval details
I_1	[0, 0.059]	I_7	[0.06]
I_2	[0, 0.152]	I_8	[0.152]
I_3	[0.058, 0.059]	I_9	[0.133, 0.407]
I_4	[0.153, 0.154]	I_{10}	[0.367, 0.625]
I_5	[0.06, 0.132]	I_{11}	[0.408, 1.0]
I_6	[0.153, 0.366]	I_{12}	[0.626, 0.923]

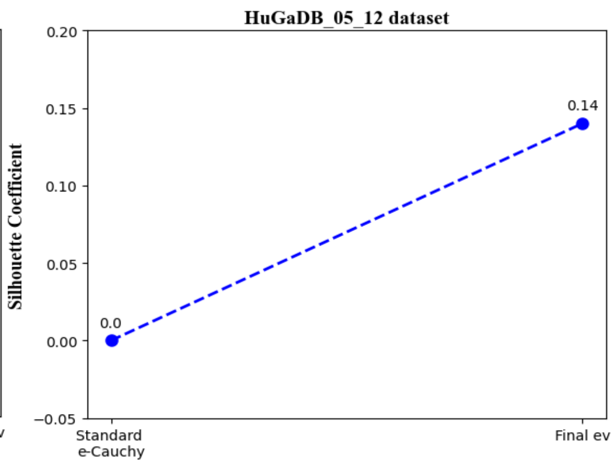
Table 4.26: The fuzzy rules of HuGaDB _01_01 dataset

Rule No.	Rule description
R_1	IF $Bel_{cs} \in \mathbf{I}_1$ and μf_3 of $\mu f_{c(1)} \in \mathbf{I}_2$ THEN $CS \in C_1$
R_2	IF $Bel_{cs} \in \mathbf{I}_3$ and μf_3 of $\mu f_{c(1)} \in \mathbf{I}_4$ THEN $CS \in C_1$
R_3	IF $Bel_{cs} \in \mathbf{I}_5$ and μf_3 of $\mu f_{c(2)} \in \mathbf{I}_6$ THEN $CS \in C_2$
R_4	IF $Bel_{cs} = \mathbf{I}_7$ and μf_3 of $\mu f_{c(2)} = \mathbf{I}_8$ THEN $CS \in C_2$
R_5	IF $Bel_{cs} \in \mathbf{I}_9$ and μf_3 of $\mu f_{c(3)} \in \mathbf{I}_{10}$ THEN $CS \in C_3$
R_6	IF $Bel_{cs} \in \mathbf{I}_{11}$ and μf_3 of $\mu f_{c(4)} \in \mathbf{I}_{12}$ THEN $CS \in C_4$

4.3.4.2 HuGaDB_05_12 Dataset

The second sub dataset is the HuGaDB_05_12. SC values for the training and testing phases are detailed in Figure 4.19.





(a) *(b)*

Figure 4.19: (a) SC of training to HuGaDB_05_12 dataset (b) SC of testing to HuGaDB_05_12

NoC firstly is 1538 clusters that are reduced to 7, and then to 4 and finally, to three clusters. Table 4.27 describes the building of primer cluster.

Table 4.27: The description of primer clusters construction for HuGaDB-05-12 dataset

# of sample	**Density value**	**Create cluster?**	**# of clusters**
1.	0.004761	Yes	C_1
2.	0.000486	Yes	C_2
3.	0.000486	No	C_2
4.	0.009034	No	C_2
5.	0.000488	No	C_2
6.	0.000469	Yes	C_3
7.	0.000493	No	C_3
8.	0.000475	No	C_3
9.	0.006868	No	C_3
10.	0.000483	No	C_3
11.	0.000482	No	C_3
12.	0.000480	Yes	C_4
13.	0.000487	No	C_4
14.	0.000495	No	C_4
15.	0.000486	No	C_4
…	…	…	…
…	…	…	…

On the other hand, Figure 4.20 explains how NoC is reduced over epochs to this sub dataset.

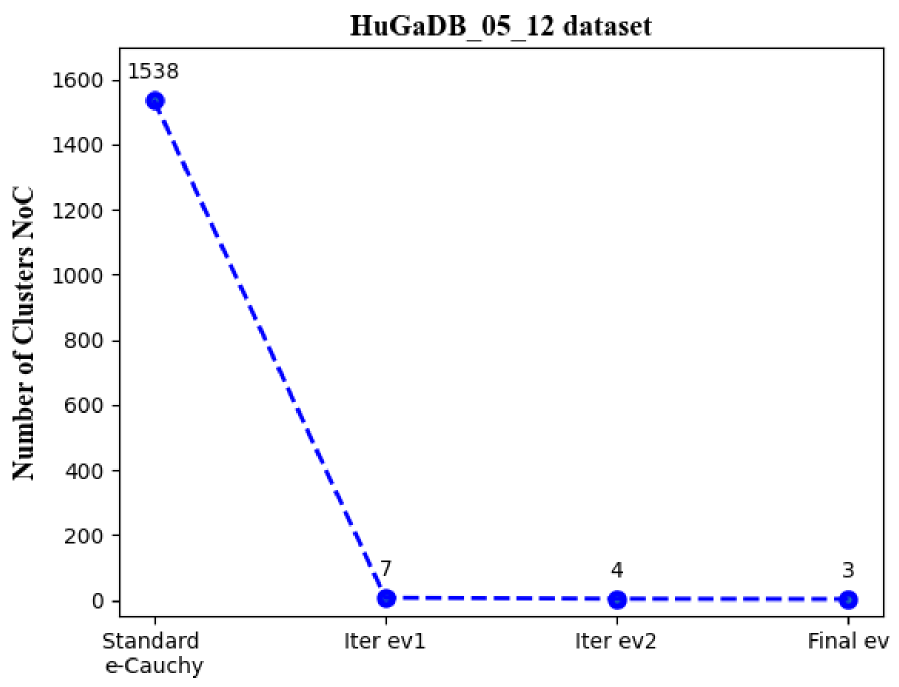


Figure 4.20: The decreasing of NoC for HuGaDB_05_12dataset

Table 4.28 lists the details of SC and NoC to HuGaDB_05_12 dataset.

Table 4.28: SC and NoC for HuGaDB_05_12 dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.50	0.00	1538
2.	Iter ev1	0.15	---	7
3.	Iter ev2	0.18	---	4
4.	Final ev	0.22	0.14	3

Tables 4.29 and 4.30 show the intervals as well as the fuzzy rules for this sub dataset.

Table 4.29: The intervals of HuGaDB_05_12dataset

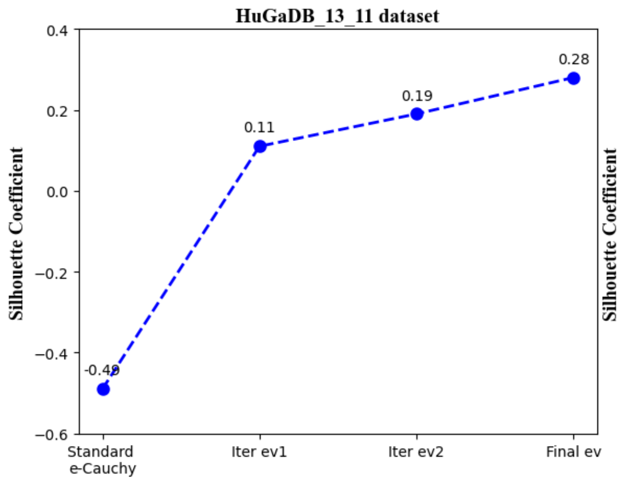
Interval No.	Interval details	Interval No.	Interval details
I_1	[0, 0.486]	**I_6**	[0.296, 0.622]
I_2	[0, 0.295]	**I_7**	[0.741, 0.742]
I_3	[0.482, 0.484]	**I_8**	[0.623]
I_4	[0.296, 0.297]	**I_9**	[0.744, 1.0]
I_5	[0.487, 0.743]	**I_{10}**	[0.623, 0.93]

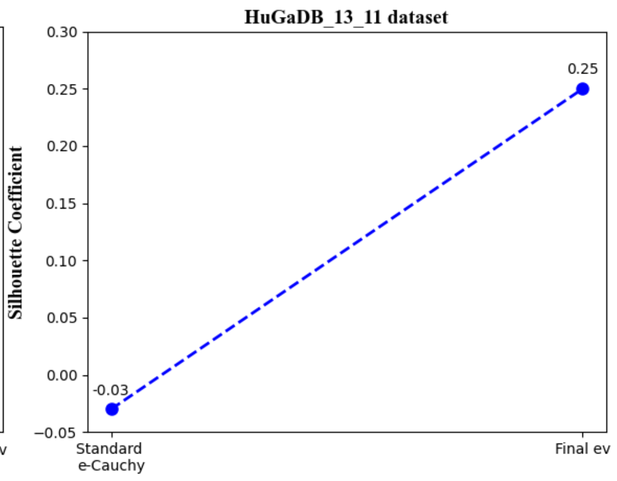
Table 4.30: The fuzzy rules of HuGaDB_05_12dataset

Rule No.	Rule description
R_1	IF Bel_{cs} ∈ $\mathbf{I}_1$ and μf_3 of $\mu f_{c(1)}$ ∈ $\mathbf{I}_2$ THEN $CS \in C_1$
R_2	IF Bel_{cs} ∈ $\mathbf{I}_3$ and μf_3 of $\mu f_{c(1)}$ = $\mathbf{I}_4$ THEN $CS \in C_1$
R_3	IF Bel_{cs} ∈ $\mathbf{I}_5$ and μf_3 of $\mu f_{c(2)}$ ∈ $\mathbf{I}_6$ THEN $CS \in C_2$
R_4	IF Bel_{cs} ∈ $\mathbf{I}_7$ and μf_3 of $\mu f_{c(2)}$ ∈ $\mathbf{I}_8$ THEN $CS \in C_2$
R_5	IF Bel_{cs} ∈ $\mathbf{I}_9$ and μf_3 of $\mu f_{c(3)}$ ∈ $\mathbf{I}_{10}$ THEN $CS \in C_3$

4.3.4.3 HuGaDB_13_11 Dataset

HuGaDB_ 13_11 is the next sub dataset that is tested by this model. Figure 4.21 displays the values of SC to this sub dataset.





(a) *(b)*

Figure 4.21: (a) SC of training to HuGaDB_13_11 dataset (b) SC of testing to HuGaDB_13_11

On the other hand, NoC starts with 1845 clusters, which decreases to 9 clusters and to 4 clusters. Lastly, NoC is three clusters. Moreover, the construction of the primer clusters is listed in Table 4.31.

Table 4.31: The description of primer clusters construction for HuGaDB_13_11 dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.011242	Yes	C_1
2.	0.000472	Yes	C_2

3.	0.000484	No	C_2
4.	0.000892	No	C_2
5.	0.000493	No	C_2
6.	0.000489	Yes	C_3
7.	0.000477	No	C_3
8.	0.000482	No	C_3
9.	0.010756	No	C_3
10.	0.000489	No	C_3
11.	0.000484	No	C_3
12.	0.000483	Yes	C_4
13.	0.000471	No	C_4
14.	0.000489	No	C_4
15.	0.000485	No	C_4
…	…	…	…
…	…	…	…

In the same context, Figure 4.22 displays the reducing of NoC for HuGaDB_13_11 sub dataset during the implementation of the evolving clustering model.

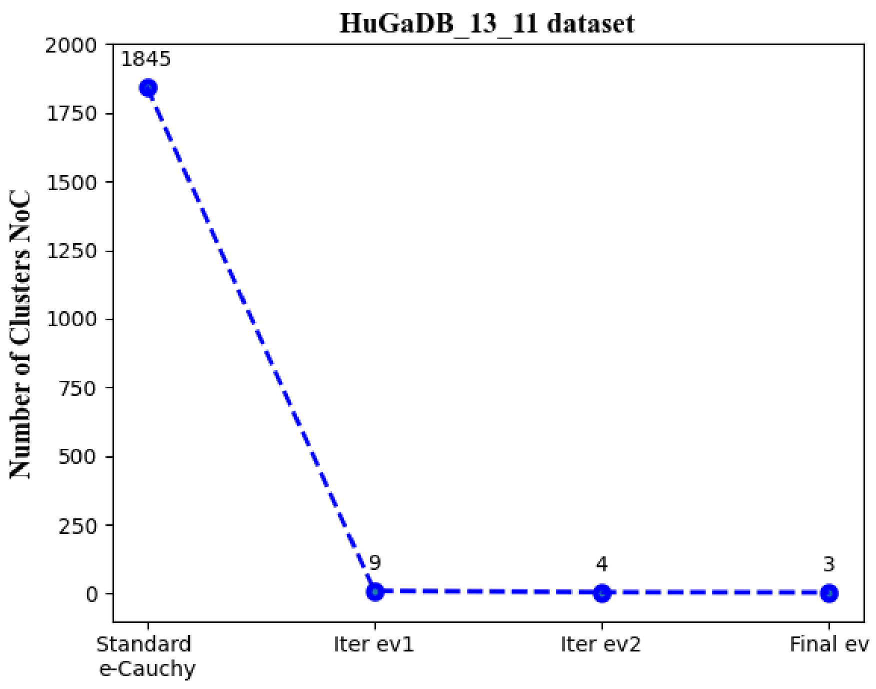


Figure 4.22: The decreasing of NoC for HuGaDB _13_11dataset

In addition, Table 4.32 lists the details of both the SC values and NoC to this sub dataset.

Table 4.32: SC and NoC for HuGaDB_13_11 dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.49	-0.03	1845
2.	Iter ev1	0.11	---	9
3.	Iter ev2	0.19	---	4
4.	Final ev	0.28	0.25	3

Table 4.33 explains the intervals of HuGaDB_13_11 dataset, while Table 4.34 shows the fuzzy rules.

Table 4.33: The intervals of HuGaDB_13_11dataset

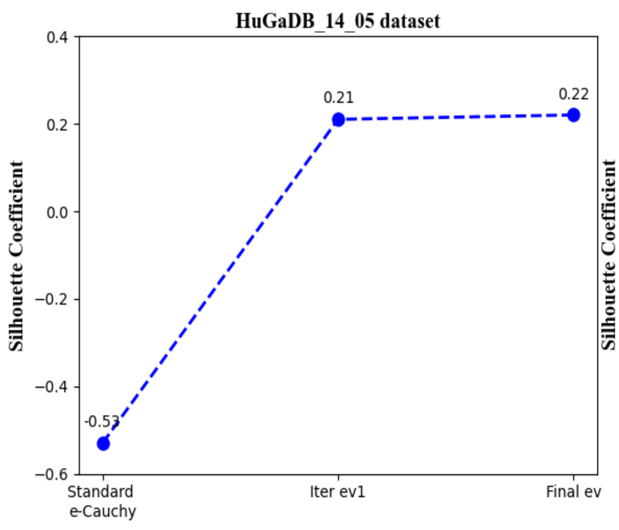
Interval No.	Interval details	Interval No.	Interval details
I$_1$	[0, 0.0930]	**I**$_6$	[0.64]
I$_2$	[0, 0.218]	**I**$_7$	[0.15, 0.1508]
I$_3$	[0.0931, 0.155]	**I**$_8$	[0.64, 0.65]
I$_4$	[0.219, 0.63]	**I**$_9$	[0.156, 1.0]
I$_5$	[0.152, 0.153]	**I**$_{10}$	[0.64, 0.895]

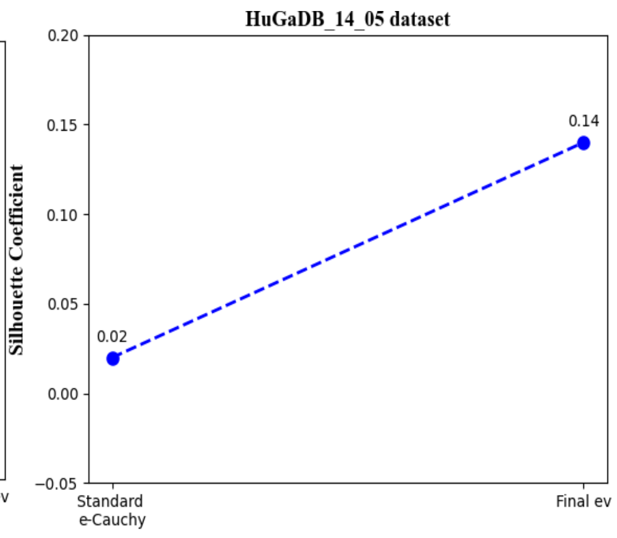
Table 4.34: The fuzzy rules of HuGaDB_13_11dataset

Rule No.	Rule description
R$_1$	IF Bel_{cs} ∈ **I**$_1$ and μf_3 of $\mu f_{c(1)}$ ∈ **I**$_2$ THEN $CS \in C_1$
R$_2$	IF Bel_{cs} ∈ **I**$_3$ and μf_3 of $\mu f_{c(1)}$ ∈ **I**$_4$ THEN $CS \in C_1$
R$_3$	IF Bel_{cs} ∈ **I**$_5$ and μf_3 of $\mu f_{c(2)} =$ **I**$_6$ THEN $CS \in C_2$
R$_4$	IF Bel_{cs} ∈ **I**$_7$ and μf_3 of $\mu f_{c(2)}$ ∈ **I**$_8$ THEN $CS \in C_2$
R$_5$	IF Bel_{cs} ∈ **I**$_9$ and μf_3 of $\mu f_{c(3)}$ ∈ **I**$_{10}$ THEN $CS \in C_3$

4.3.4.4 HuGaDB_14_05 Dataset

The final sub dataset to evaluate the evolving clustering model is the HuGaDB_14_05. Figure 4.23 indicates all SC values for the training and testing phases.





(a) *(b)*

Figure 4.23: (a) SC of training to HuGaDB_14_05 dataset (b) SC of testing to HuGaDB_14_05

On the other hand, Table 4.35 clarifies the steps of the primer clusters construction.

Table 4.35: The description of primer clusters construction for HuGaDB_14_05 dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.008574	Yes	C_1
2.	0.000456	Yes	C_2
3.	0.000458	No	C_2
4.	0.004104	No	C_2
5.	0.000456	No	C_2
6.	0.000489	Yes	C_3
7.	0.000485	No	C_3
8.	0.000488	No	C_3
9.	0.004887	No	C_3
10.	0.000462	No	C_3
11.	0.000482	No	C_3
12.	0.000483	Yes	C_4
13.	0.000494	No	C_4
14.	0.000480	No	C_4

15.	0.000477	No	C_4
…	…	…	…
…	…	…	…

Figure 4.24 lists the reducing of NoC for HuGaDB_14_05 sub dataset.

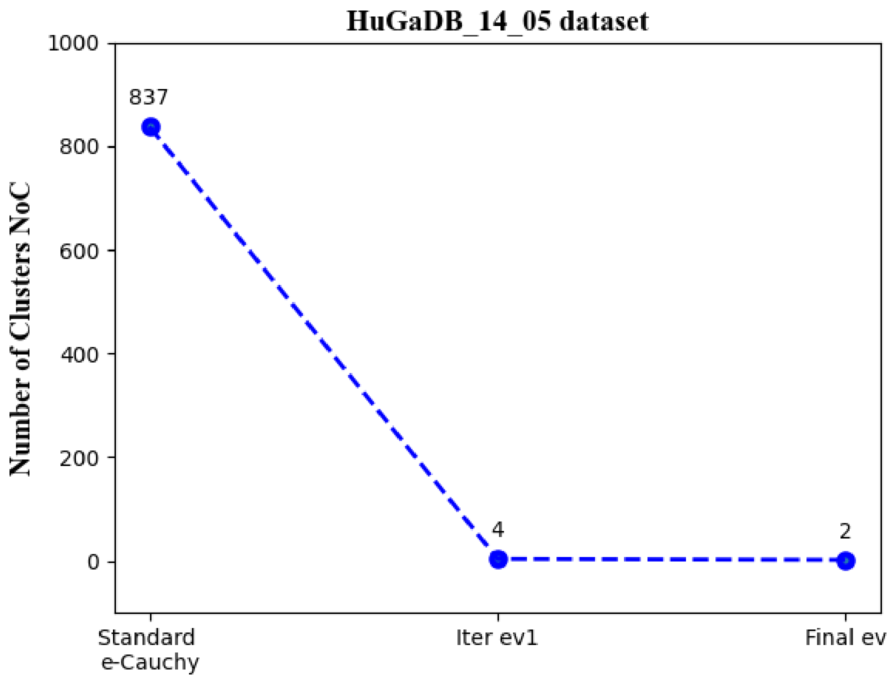


Figure 4.24: The NoC decreasing of HuGaDB_14_05 dataset

Moreover, Table 4.36 lists all the SC and NoC results of HuGaDB_14_05 dataset.

Table 4.36: SC and NoC for HuGaDB_14_05 dataset

Item	**Progress phase**	**Training phase**	**Testing phase**	**NoC**
1.	Standard e-Cauchy	-0.53	0.02	837
2.	Iter ev1	0.21	---	4
3.	Final ev	0.22	0.14	2

In terms of the intervals and fuzzy rules, Tables 4.37, and 4.38 shows all their values.

Table 4.37: The intervals of HuGaDB_14_05dataset

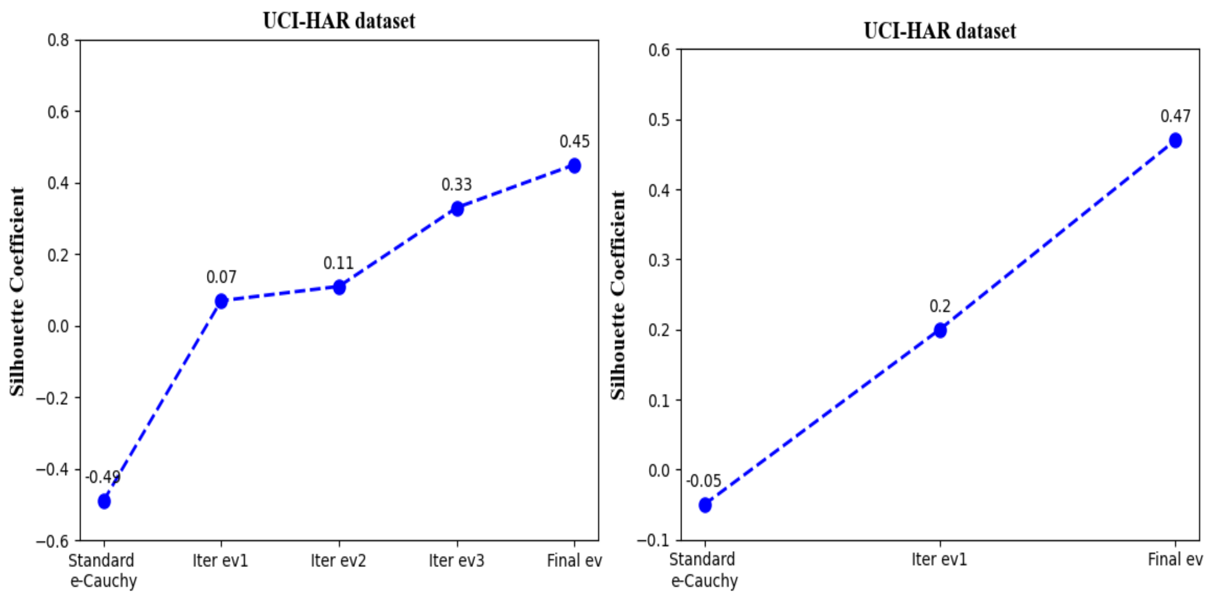
Interval No.	**Interval details**	**Interval No.**	**Interval details**
I$_1$	[0, 0.401]	**I**$_5$	[0.404, 0.405]
I$_2$	[0, 0.363]	**I**$_6$	[0.362, 0.363]
I$_3$	[0.399, 0.4]	**I**$_7$	[0.402, 1.0]
I$_4$	[0.364, 0.365]	**I**$_8$	[0.364, 0.855]

Table 4.38: The fuzzy rules of HuGaDB_14_05 dataset

Rule No.	**Rule description**
R$_1$	IF Bel_{cs} ∈ **I**$_1$ and μf_3 of $\mu f_{c(1)}$ ∈ **I**$_2$ THEN $CS \in C_1$
R$_2$	IF Bel_{cs} ∈ **I**$_3$ and μf_3 of $\mu f_{c(1)}$ ∈ **I**$_4$ THEN $CS \in C_1$
R$_3$	IF Bel_{cs} ∈ **I**$_5$ and μf_3 of $\mu f_{c(2)}$ ∈ **I**$_6$ THEN $CS \in C_1$
R$_4$	IF Bel_{cs} ∈ **I**$_7$ and μf_3 of $\mu f_{c(2)}$ ∈ **I**$_8$ THEN $CS \in C_2$

4.3.5 UCI-HAR Dataset

Thereafter, the evolving clustering model tests the UCI-HAR dataset. The effective threshold Θ_{den} of e-Cauchy algorithm sets to 0.00028. The measurement of SC to this dataset is displayed in Figure 4.25.



(a) (b)

Figure 4.25: (a) SC of training to UCI-HAR dataset (b) SC of testing to UCI-HAR dataset

Initially, NoC of UCI-HAR dataset is 3605 clusters, then it reduces to 933 clusters, and 412 clusters and at last, it becomes six clusters. The primer clusters are built to this dataset as detailed in Table 4.39.

Table 4.39: The description of primer clusters construction for UCI-HAR dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.012254	Yes	C_1
2.	0.000635	No	C_1
3.	0.000458	Yes	C_2
4.	0.000254	No	C_2
5.	0.000302	No	C_2
6.	0.000309	No	C_2
7.	0.000444	Yes	C_3
8.	0.000488	No	C_3
9.	0.000721	No	C_3
10.	0.000510	No	C_3
11.	0.000488	No	C_3
12.	0.000380	No	C_3
13.	0.000228	No	C_3
14.	0.000279	No	C_3
15.	0.000377	No	C_3
16.	0.000274	No	C_3
17.	0.000641	No	C_3
18.	0.000622	No	C_3
19.	0.000253	No	C_3
20.	0.000489	No	C_3
21.	0.000509	No	C_3
22.	0.000711	Yes	C_4
23.	0.000622	No	C_4
24.	0.000594	No	C_4
…	…	…	…
…	…	…	…

Furthermore, the decreasing of NoC is shown in Figure 4.26.

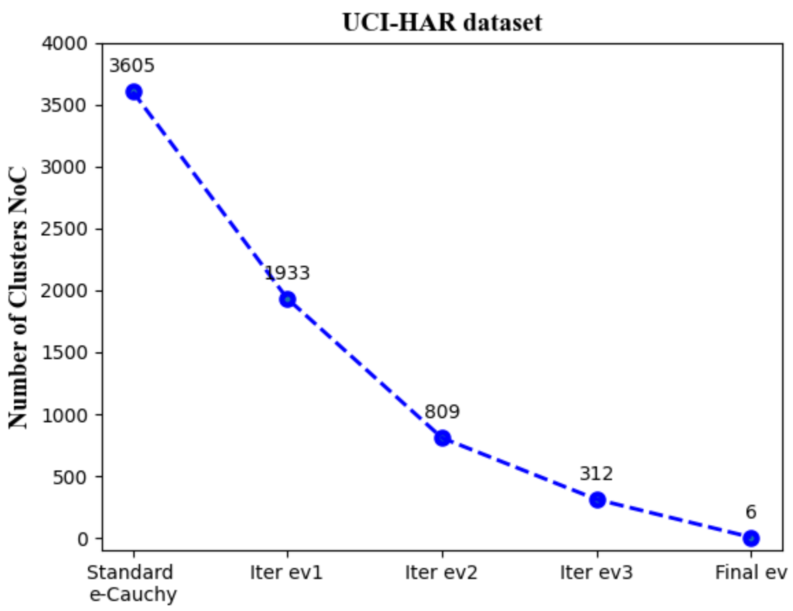


Figure 4.26: The decreasing of NoC for UCI-HAR dataset

All the information about SC and NoC are listed in Table 4.40.

Table 4.40: SC and NoC for UCI-HAR dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.49	-0.05	3605
2.	Iter ev1	0.07	0.2	1933
3.	Iter ev2	0.11	---	809
4.	Iter ev3	0.33	---	312
5.	Final ev	0.45	0.47	6

Concerning the fuzzy rules as well as the intervals which are built for this dataset, detailed in Tables 4.41 and 4.42.

Table 4.41: The intervals of UCI-HAR dataset

Interval No.	Interval details	Interval No.	Interval details
I_1	[0, 0.130]	**I_{10}**	[0.429, 0.431]
I_2	[0, 0.144]	**I_{11}**	[0.234, 0.297]
I_3	[0.128, 0.129]	**I_{12}**	[0.428, 0.601]
I_4	[0.146, 0.147]	**I_{13}**	[0.298, 0.362]
I_5	[0.131, 0.199]	**I_{14}**	[0.602, 0.788]

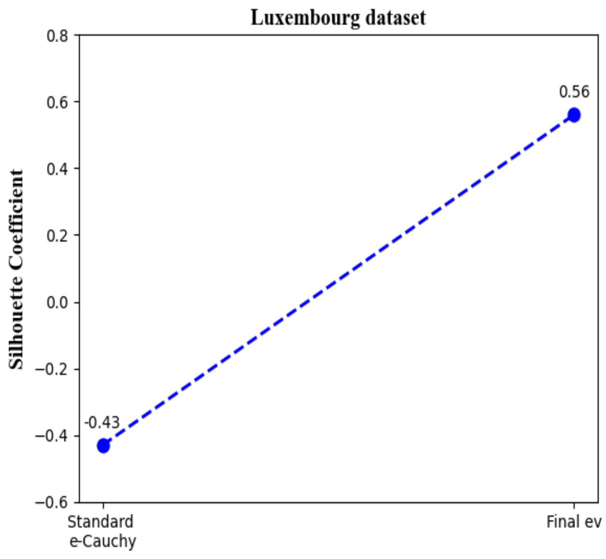
$\mathbf{I}_6$	[0.145, 0.278]	$\mathbf{I}_{15}$	[0.357, 0.360]
$\mathbf{I}_7$	[0.2, 0.233]	$\mathbf{I}_{16}$	[0.789, 0.791]
$\mathbf{I}_8$	[0.279, 0.427]	$\mathbf{I}_{17}$	[0.363, 1.0]
$\mathbf{I}_9$	[0.230, 0.232]	$\mathbf{I}_{18}$	[0.789, 0.916]

Table 4.42: The fuzzy rules of UCI-HAR dataset

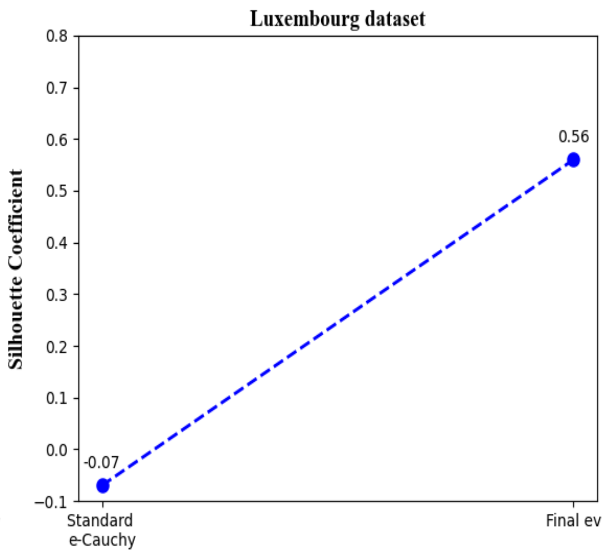
Rule No.	**Rule description**
R_1	IF $Bel_{cs} \in \mathbf{I}_1$ and μf_3 of $\mu f_{c(1)} \in \mathbf{I}_2$ THEN $CS \in C_1$
R_2	IF $Bel_{cs} \in \mathbf{I}_3$ and μf_3 of $\mu f_{c(1)} \in \mathbf{I}_4$ THEN $CS \in C_1$
R_3	IF $Bel_{cs} \in \mathbf{I}_5$ and μf_3 of $\mu f_{c(2)} \in \mathbf{I}_6$ THEN $CS \in C_2$
R_4	IF $Bel_{cs} \in \mathbf{I}_7$ and μf_3 of $\mu f_{c(3)} \in \mathbf{I}_8$ THEN $CS \in C_3$
R_5	IF $Bel_{cs} \in \mathbf{I}_9$ and μf_3 of $\mu f_{c(3)} \in \mathbf{I}_{10}$ THEN $CS \in C_3$
R_6	IF $Bel_{cs} \in \mathbf{I}_{11}$ and μf_3 of $\mu f_{c(4)} \in \mathbf{I}_{12}$ THEN $CS \in C_4$
R_7	IF $Bel_{cs} \in \mathbf{I}_{13}$ and μf_3 of $\mu f_{c(5)} \in \mathbf{I}_{14}$ THEN $CS \in C_5$
R_8	IF $Bel_{cs} \in \mathbf{I}_{15}$ and μf_3 of $\mu f_{c(5)} \in \mathbf{I}_{16}$ THEN $CS \in C_5$
R_9	IF $Bel_{cs} \in \mathbf{I}_{17}$ and μf_3 of $\mu f_{c(6)} \in \mathbf{I}_{18}$ THEN $CS \in C_6$

4.3.6 Luxembourg Dataset

Afterwards, the evolving clustering model tests the Luxembourg dataset. Θ_{den} threshold is 0.0026 where, its SC values are described in Figure 4.27.



(a)



(b)

Figure 4.27: (a) SC of training to Luxembourg dataset (b) SC of testing to Luxembourg dataset

The standard e-Cauchy algorithm firstly generates 666 clusters, which are reduced to two clusters. During this dataset, the primer clusters are built as shown in the Table 4.43.

Table 4.43: The description of primer clusters construction for Luxembourg dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.002427	Yes	C_1
2.	0.002670	Yes	C_2
3.	0.004970	No	C_2
4.	0.002652	No	C_2
5.	0.000608	No	C_2
6.	0.000610	No	C_2
7.	0.000609	No	C_2
8.	0.000609	Yes	C_3
9.	0.000614	No	C_3
10.	0.005838	No	C_3
11.	0.000607	No	C_3
12.	0.000601	No	C_3
13.	0.000595	No	C_3
14.	0.000609	Yes	C_4
15.	0.000604	No	C_4
16.	0.000599	No	C_4
17.	0.000524	No	C_4
18.	0.000608	No	C_4
19.	0.000547	No	C_4
20.	0.000531	No	C_4
21.	0.000545	No	C_4
22.	0.000606	Yes	C_5
23.	0.000591	No	C_5
24.	0.000584	No	C_5
…	…	…	…
…	…	…	…

Moreover, the decreasing of NoC for Luxembourg dataset is explained in Figure 4.28.

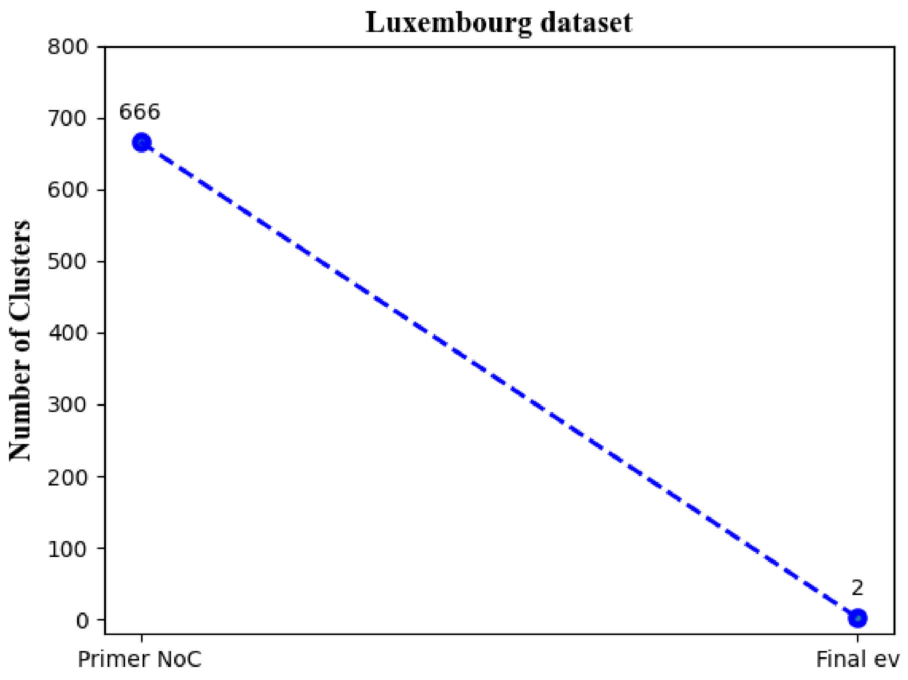


Figure 4.28: The decreasing of NoC for Luxembourg dataset

The information about SC values and NoC are detailed in Table 4.44.

Table 4.44: SC and NoC for Luxembourg dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.43	-0.07	666
2.	Final ev	0.56	0.56	2

The intervals for this dataset are constructed as explained in Table 4.45, while the fuzzy rules are built as Table 4.46.

Table 4.45: The intervals of Luxembourg dataset

Interval No.	Interval details	Interval No.	Interval details
I_1	[0, 0.349]	I_4	[0.425, 0.426]
I_2	[0, 0.423]	I_5	[0.35, 1.0]
I_3	[0.344, 0.345]	I_6	[0.424, 0.848]

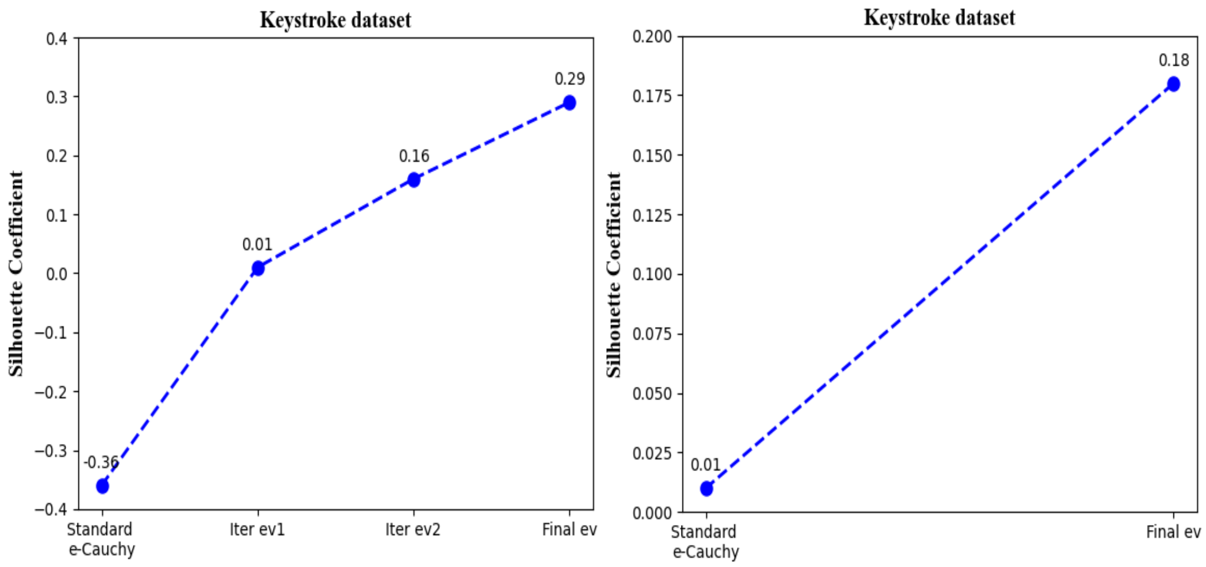
Table 4.46: The fuzzy rules of Luxembourg dataset

Rule No.	Rule description
R_1	IF Bel_{cs} ∈ $\mathbf{I}_1$ and μf_3 of $\mu f_{c(1)}$ ∈ $\mathbf{I}_2$ THEN $CS \in C_1$
R_2	IF Bel_{cs} ∈ $\mathbf{I}_3$ and μf_3 of $\mu f_{c(1)}$ ∈ $\mathbf{I}_4$ THEN $CS \in C_1$
R_3	IF Bel_{cs} ∈ $\mathbf{I}_5$ and μf_3 of $\mu f_{c(2)}$ ∈ $\mathbf{I}_6$ THEN $CS \in C_2$

4.3.7 Keystroke Dataset

The main keystroke dataset contains four classes while, the keystroke_1 dataset has only three classes (all data samples of class '3' have been hidden). Firstly, the DL model able to classify the data samples of keystroke_1 into three classes while, the evolving clustering model detects new data samples during processing of keystroke dataset (which, in fact, has never been trained before). Therefore, the model creates a new cluster and assigns the first data sample and continues to assign any sample to this new cluster as long as it has a different behavior from what this model is trained on.

However, for keystroke dataset, Θ_{den} sets to 0.055. The SC results to the training and testing phases are explained in Figure 4.29.



(a) (b)

Figure 4.29: (a) SC of training to Keystroke dataset (b) SC of testing to Keystroke dataset

In terms of NoC, the standard e-Cauchy clustering algorithm generates firstly 592 clusters, then it decreases to 39, 8 and lastly to four clusters. Table 4.47 shows the construction of the primer clusters for this dataset.

Table 4.47: The description of primer clusters construction for keystroke dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.077601	Yes	C_1
2.	0.001146	Yes	C_2
3.	0.001290	No	C_2
4.	0.022175	Yes	C_3
5.	0.001676	No	C_3
6.	0.038295	No	C_3
7.	0.034471	Yes	C_4
8.	0.001437	No	C_4
9.	0.047535	No	C_4
10.	0.030900	No	C_4
11.	0.080139	No	C_4
12.	0.001570	No	C_4
…	…	…	…
…	…	…	…

Figure 4.30 illustrates the diminution of NoC for keystroke dataset.

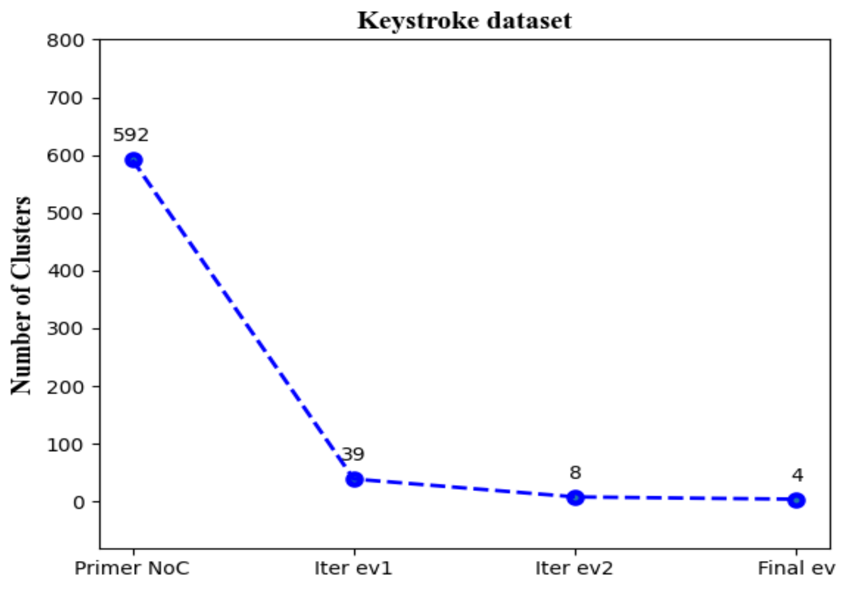


Figure 4.30: The decreasing of NoC for keystroke dataset

Furthermore, the values of SC as well as NoC are depicted in Table 4.48.

Table 4.48: SC and NoC for keystroke dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.36	0.01	592
2.	Iter ev1	0.01	---	39
3.	Iter ev2	0.16	---	8
4.	Final ev	0.29	0.18	4

As for the intervals and fuzzy rules, Tables 4.49, and 4.50 shows their values for the keystroke dataset.

Table 4.49: The intervals of keystroke dataset

Interval No.	Interval details	Interval No.	Interval details
I_1	[0, 0.0373]	**I_8**	[0.47]
I_2	[0, 0.12]	**I_9**	[0.0579, 0.150]
I_3	[0.0371, 0.0372]	**I_{10}**	[0.465, 0.7]
I_4	[0.13, 0.14]	**I_{11}**	[0.153, 0.154]
I_5	[0.0374, 0.0578]	**I_{12}**	[0.697, 0.699]
I_6	[0.13, 0.464]	**I_{13}**	[0.151, 1.0]
I_7	[0.0574, 0.0575]	**I_{14}**	[0.71, 0.89]

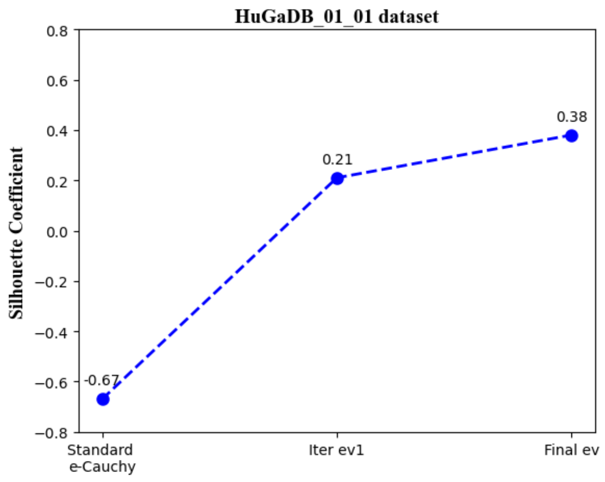
Table 4.50: The fuzzy rules of keystroke dataset

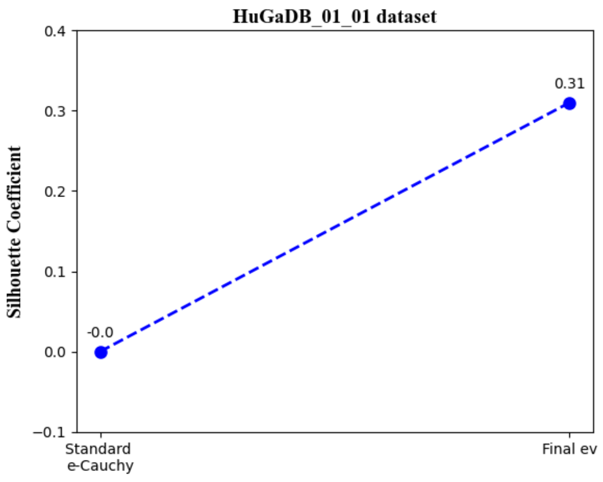
Rule No.	Rule description
R_1	IF Bel_{cs} ∈ $\mathbf{I}_1$ and μf_3 of $\mu f_{c(1)}$ ∈ $\mathbf{I}_2$ THEN $CS \in C_1$
R_2	IF Bel_{cs} ∈ $\mathbf{I}_3$ and μf_3 of $\mu f_{c(1)}$ ∈ $\mathbf{I}_4$ THEN $CS \in C_1$
R_3	IF Bel_{cs} ∈ $\mathbf{I}_5$ and μf_3 of $\mu f_{c(2)}$ ∈ $\mathbf{I}_6$ THEN $CS \in C_2$
R_4	IF Bel_{cs} ∈ $\mathbf{I}_7$ and μf_3 of $\mu f_{c(2)}$ ∈ $\mathbf{I}_8$ THEN $CS \in C_2$
R_5	IF Bel_{cs} ∈ $\mathbf{I}_9$ and μf_3 of $\mu f_{c(3)}$ ∈ $\mathbf{I}_{10}$ THEN $CS \in C_3$
R_6	IF Bel_{cs} ∈ $\mathbf{I}_{11}$ and μf_3 of $\mu f_{c(3)}$ ∈ $\mathbf{I}_{12}$ THEN $CS \in C_3$
R_7	IF Bel_{cs} ∈ $\mathbf{I}_{13}$ and μf_3 of $\mu f_{c(4)}$ ∈ $\mathbf{I}_{14}$ THEN $CS \in C_4$

4.3.8 HuGaDB_01_01 Dataset

The HuGaDB_01_01 dataset has four classes and HuGaDB_01_01_1 dataset consists of only three classes (where all data samples of class '2' are hidden). The DL able to classify HuGaDB_01_01_1 dataset into three classes. While the evolving clustering model detects new data samples and then, creates a new cluster when processes HuGaDB_01_01 dataset.

The effective threshold Θ_{den} sets to 0.0038. All SC values are illustrated in Figure 4.31 for both training and testing phases.





(a) *(b)*

Figure 4.31: (a) SC of training to HuGaDB_01_01 dataset (b) SC of testing to HuGaDB_01_01

In the regard of NoC, the model generates 952 clusters, that decreases to 9 clusters and finally to just four clusters. The primer clusters are built for this dataset as described in Table 4.41.

Table 4.51: The description of primer clusters construction for HuGaDB_01_01 dataset

# of sample	Density value	Create cluster?	# of clusters
1.	0.005094	Yes	C_1
2.	0.000491	Yes	C_2
3.	0.000494	No	C_2
4.	0.004878	No	C_2
5.	0.000493	No	C_2
6.	0.000493	Yes	C_3
7.	0.000494	No	C_3
8.	0.000495	No	C_3
9.	0.006243	No	C_3
10.	0.000491	No	C_3
11.	0.000494	No	C_3
12.	0.000480	Yes	C_4
13.	0.000488	No	C_4
14.	0.000489	No	C_4
15.	0.000478	No	C_4
…	…	…	…
…	…	…	…

The decreasing of NoC over epochs to this dataset is illustrated in Figure 4.32.

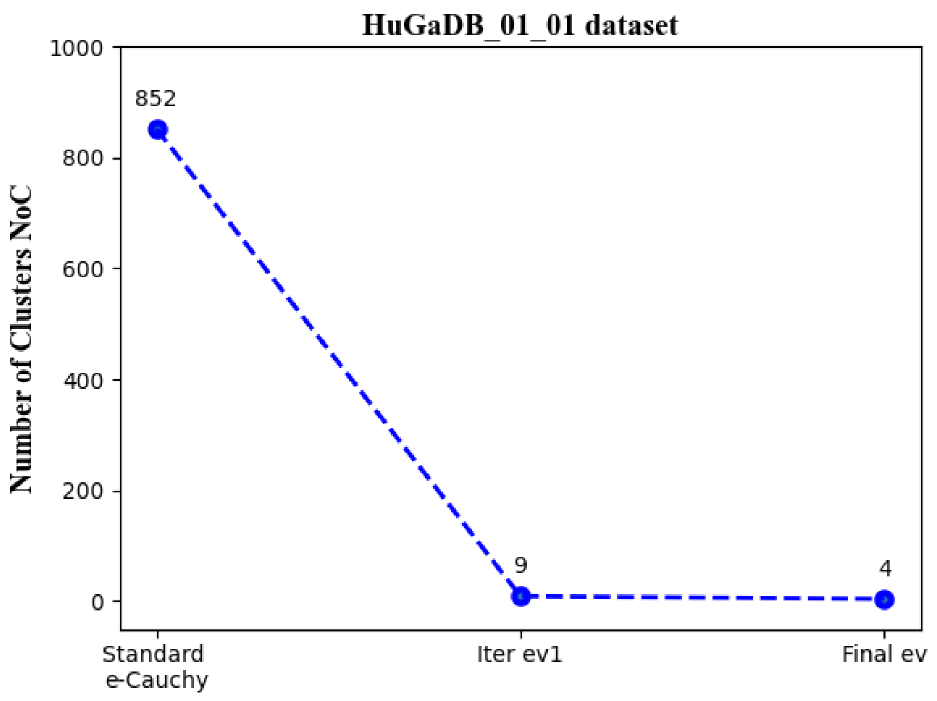


Figure 4.32: The decreasing of NoC for HuGaDB_01_01 dataset

The details of SC and also NoC are described in Table 4.52.

Table 4.52: SC and NoC for HuGaDB _01_01 dataset

Item	Progress phase	Training phase	Testing phase	NoC
1.	Standard e-Cauchy	-0.67	-0.00	852
2.	Iter ev1	0.21	---	9
3.	Final ev	0.38	0.31	4

The intervals for HuGaDB _01_01 dataset are built as detailed in Table 4.53.

Table 4.53: The intervals of HuGaDB _01_01 dataset

Interval No.	Interval details	Interval No.	Interval details
I_1	[0, 0.059]	**I_7**	[0.06]
I_2	[0, 0.152]	**I_8**	[0.152]
I_3	[0.058, 0.059]	**I_9**	[0.133, 0.407]
I_4	[0.153, 0.154]	**I_{10}**	[0.367, 0.625]

I$_5$	[0.06, 0.132]	**I**$_{11}$	[0.408, 1.0]
I$_6$	[0.153, 0.366]	**I**$_{12}$	[0.626, 0.923]

While the fuzzy rules for this dataset are constructed as described in Table 4.54.

Table 4.54: The fuzzy rules of HuGaDB _01_01 dataset

Rule No.	**Rule description**
R$_1$	IF Bel_{cs} ∈ **I**$_1$ and μf_3 of $\mu f_{c(1)}$ ∈ **I**$_2$ THEN $CS \in C_1$
R$_2$	IF Bel_{cs} ∈ **I**$_3$ and μf_3 of $\mu f_{c(1)}$ ∈ **I**$_4$ THEN $CS \in C_1$
R$_3$	IF Bel_{cs} ∈ **I**$_5$ and μf_3 of $\mu f_{c(2)}$ ∈ **I**$_6$ THEN $CS \in C_2$
R$_4$	IF $Bel_{cs} =$ **I**$_7$ and μf_3 of $\mu f_{c(2)} =$ **I**$_8$ THEN $CS \in C_2$
R$_5$	IF Bel_{cs} ∈ **I**$_9$ and μf_3 of $\mu f_{c(3)}$ ∈ **I**$_{10}$ THEN $CS \in C_3$
R$_6$	IF Bel_{cs} ∈ **I**$_{11}$ and μf_3 of $\mu f_{c(4)}$ ∈ **I**$_{12}$ THEN $CS \in C_4$

Regarding the improvement rate for SC during the evolving clustering model, Table 4.55 describes their values for all stream datasets.

Table 4.55: The improvement rate for SC to all stream datasets

Item	**Dataset**	**SC improvement rate**
1.	Keystroke	0.65
2.	Electricity	0.28
3.	NSL-KDD	0.93
4.	HuGaDB _01_01	0.99
5.	HuGaDB _05_12	0.72
6.	HuGaDB _13_11	0.77
7.	HuGaDB _14_05	0.75
8.	UCI-HAR	0.73
9.	Luxembourg	0.99
10.	Keystroke_1	0.65
11.	HuGaDB _01_01_1	0.99

In terms of required time for applying the evolving clustering model, Table 4.56 details the required time to all stream datasets.

Table 4.56: The required time (in seconds) for the evolving clustering model to process the stream datasets

Item	Dataset	Time (seconds)
1.	Keystroke	8.03
2.	Electricity	53.18
3.	NSL-KDD	4.47
4.	HuGaDB01_01	23.16
5.	HuGaDB05_12	63.23
6.	HuGaDB13_11	103.57
7.	HuGaDB14_05	17.41
8.	UCI-HAR	197.04
9.	Luxembourg	12.55
10.	Keystroke_1	8.03
11.	HuGaDB01_01_1	23.16

While Table 4.57 proves the effectiveness of the evolving clustering model by outperforming a number of previous methods for the stream datasets.

Table 4.57: The required time (in seconds) for previous online clustering methods to process the stream datasets

Item	Ref.	Dataset	Time (seconds)
1.	(Fahy & Yang, 2019)	Keystroke	15
2.	(Qaiyum et al., 2019)	Electricity	58.03
3.	(Singh & Mathai, 2019)	NSL-KDD	9.8
4.	(Lima et al., 2021)	UCI-HAR	209.79

Chapter Five

The Conclusions and Future Work

5.1 Conclusions

The principle objective of the evolving intelligent system is to make a decision that changes over time to assign the data stream sample to the cluster accurately. However, this system consists of two models.

The main conclusions of this dissertation by applying the evolving intelligent system as follows:

1- In the DL model, an adaptive method is developed that adopts a DLR value for network training instead of using a static value. Indeed, this DLR value positively affects the stability of the network, updating the weights, and thus improving the results accuracy. Accordingly, the DLR model superiors many other models that process data stream.

2- Through applying the evolving intelligent system, the DL model is used as an external evaluator. More deeply, the DL model is used to evaluate the results of the evolving clustering model by comparing the number of detected classes by the DL model with the final number of clusters that resulted from the evolving clustering model.

3- In the evolving clustering model, which depends on standard e-Cauchy algorithm, a certain threshold is applied to each stream dataset and this threshold value remains constant even if the data behavior changes (in the case of creating new clusters). Accordingly, these aspects help the proposed system to generalize.

4- During the evolving clustering model, the evolving mechanisms are applied since they are able to deal with non-stationary environments for data. These mechanisms are, (a) adding clusters (to add new clusters), (b) splitting clusters (to split the primer clusters into LQCs and HQCs, and (c) merging clusters (to re-assign all the data samples in each LQC to HQCs), and finally produce FHQCs.

5- Based on FHQCs, the model builds the fuzzy rules to assign the data stream samples since the fuzzy rules are essential and important pillar in the work of evolving mechanisms, this feature is exploited in building the current system.

6- The evolving clustering model creates a new cluster when detecting new data stream samples that are not trained during the DL model. Therefore, by using the fuzzy rules, the evolving intelligent system able to make a decision to assign the data stream sample online to the cluster accurately.

5.2 Future Work

Some future works can be described as follows:

1- Building a fully evolving deep neural network, which means it applies the evolving stages in this network directly.

2- Using an optimization method to find evaluation parameters of the clustering instead of using the external evaluator.

3- Applying the current system to other problems (that do not include streaming datasets) to elicit general thresholds and rules through which, the decision can make.

References

References

ABBASI, A., JAVED, A. R., CHAKRABORTY, C., NEBHEN, J., ZEHRA, W., & JALIL, Z. (2021). ElStream : An Ensemble Learning Approach for Concept Drift Detection in Dynamic Social Big Data Stream Learning. IEEE Access, 9, 66408--66419. https://doi.org/10.1109/ACCESS.2021.3076264

Abdallah, Z. S., Gaber, M. M., Srinivasan, B., & Krishnaswamy, S. (2018). Activity recognition with evolving data streams: A review. ACM Computing Surveys, 51(4). https://doi.org/10.1145/3158645

Abrar, I., Ayub, Z., Masoodi, F., & Bamhdi, A. M. (2020). A Machine Learning Approach for Intrusion Detection System on NSL- KDD Dataset. International Conference on Smart Electronics and Communication (ICOSEC), 919--924.

Afshan, N., & Rout, R. K. (2021). Machine Learning Techniques for IoT Data Analytics. Big Data Analytics for Internet of Things, 89--113.

Aguiar, G., Krawczyk, B., & Cano, A. (2022). A survey on learning from imbalanced data streams: taxonomy, challenges, empirical study, and reproducible experimental framework. ArXiv Preprint ArXiv:2204.03719.

Ahanger, A. S., Khan, S. M., & Masoodi, F. (2021). An Effective Intrusion Detection System using Supervised Machine Learning Techniques. 5th International Conference on Computing Methodologies and Communication (ICCMC), 1639--1644.

Ahmad, J., Farman, H., & Jan, Z. (2019). Deep Learning Methods and Applications. Springer Singapore. https://doi.org/10.1007/978-981-13-3459-7

Ahmed, R., Dalkılıç, G., & Erten, Y. (2020). DGStream : High quality and efficiency stream clustering algorithm. Expert Systems with Applications, 141, 112947. https://doi.org/10.1016/j.eswa.2019.112947

Al-khamees, H. A. A., Al-A’araji, N., & Al-Shamery, E. S. (2022). Classifying the Human Activities of Sensor Data Using Deep Neural. International Conference on Intelligent Systems and Pattern Recognition, 107--118.

Al-Khamees, H. A. A., Al-A’araji, N., & Al-Shamery, E. S. (2021). Data Stream Clustering Using Fuzzy-based Evolving Cauchy Algorithm.

Intelligent Networks and Systems Society, 14(5), 348–358. https://doi.org/10.22266/ijies2021.1031.31

Al-Khamees, H. A. A., Al-jwaid, W. R. H., & Al-shamery, E. S. (2022). The Impact of Using Convolutional Neural Networks in COVID-19 Tasks : A Survey. International Journal of Computing and Digital Systems, 11(1), 189--197.

Alasadi, S. A., & Bhaya, W. S. (2017). Review of Data Preprocessing Techniques in Data Mining. Journal of Engineering and Applied Sciences, 12(16), 4102--4107.

Andrean, A., Jayabalan, M., & Thiruchelvam, V. (2020). Keystroke Dynamics Based User Authentication using Deep Multilayer Perceptron. International Journal of Machine Learning and Computing, 10(1), 134--139. https://doi.org/10.18178/ijmlc.2020.10.1.910

Angelov, P., Filev, D., & Kasabov, N. (2010). Evolving intelligent systems 445. Proceedings of the International Symposium on Evolving Intelligent Systems - A Symposium at the AISB 2010 Convention, 1–17. https://doi.org/10.1002/047134608x.w8405

Angelov, P., & Kasabov, N. (2006). Evolving Intelligent Systems - eIS. IEEE SMC ENewsLetter, 15, 1--13.

Babanezhad, M., Behroyan, I., Marjani, A., & Shirazian, S. (2021). Artificial intelligence simulation of suspended sediment load with different membership functions of ANFIS. Neural Computing and Applications, 33(12), 6819--6833. https://doi.org/10.1007/s00521-020-05458-6

Baek, Y., Yun, U., Chun, J., Lin, W., Yoon, E., & Fujita, H. (2020). Efficiently mining erasable stream patterns for intelligent systems over uncertain data. International Journal of Intelligent Systems, 35(11), 1699--1734. https://doi.org/10.1002/int.22269

Bahri, M., & Bifet, A. (2021). Incremental k-Nearest Neighbors Using Reservoir Sampling for Data Streams. 24th International Conference, DS 2021, Halifax, NS, Canada. https://doi.org/10.1007/978-3-030-88942-5

Bahri, M., Bifet, A., Gama, J., Gomes, H. M., & Maniu, S. (2021). Data stream analysis : Foundations , major tasks and tools. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 11(3), e1405. https://doi.org/10.1002/widm.1405

Bevilacqua, V., Brunetti, A., Guerriero, A., Francesco, G., Telegrafo, M., & Moschetta, M. (2019). A performance comparison between shallow and deeper neural networks supervised classification of tomosynthesis breast lesions images. Cognitive Systems Research, 53, 3--19. https://doi.org/10.1016/j.cogsys.2018.04.011

Bozkurt, F. (2021). A Comparative Study on Classifying Human Activities Using Classical Machine and Deep Learning Methods A Comparative Study on Classifying Human Activities Using Classical. Arabian Journal for Science and Engineering, 47(2), 1507--1521. https://doi.org/10.1007/s13369-021-06008-5

Cano, A., & Krawczyk, B. (2019). Evolving Rule-Based Classifiers with Genetic Programming on GPUs for Drifting Data Streams. Pattern Recognition, 87, 248--268. https://doi.org/10.1016/j.patcog.2018.10.024

Carnein, M., & Trautmann, H. (2018). evoStream – Evolutionary Stream Clustering Utilizing Idle Times. Big Data Research, 14, 101–111. https://doi.org/10.1016/j.bdr.2018.05.005

Challa, J. S., Goyal, P., Kokandakar, A., Mantri, D., Verma, P., Balasubramaniam, S., & Goyal, N. (2022). Anytime clustering of data streams while handling noise and concept drift. Journal of Experimental & Theoretical Artificial Intelligence, 34(3), 399--429. https://doi.org/10.1080/0952813X.2021.1882001

Chamikara, M. A. P., Bertok, P., Liu, D., Camtepe, S., & Khalil, I. (2018). Efficient Data Perturbation for Privacy Preserving and Accurate Data Stream Mining. Pervasive and Mobile Computing, 48, 1---19.

Chen, J., Wang, Z., Zhu, T., & Rosas, F. E. (2020). Recommendation Algorithm in Double-Layer Network Based on Vector Dynamic Evolution Clustering and Attention Mechanism. Complexity, 2020.

Chen, T., Peng, L., Yang, J., & Cong, G. (2021). Analysis of User Needs on Downloading Behavior of English Vocabulary APPs Based on Data Mining for Online Comments. Mathematics, 9(12), 1341.

Cunningham, J., Davis, J., Tarplee, K., & Vasquez, J. (2022). S-FINCH : An Optimized Streaming Adaptation to FINCH Clustering. International Conference on Pattern Recognition (ICPR).

David, J., & Rögnvaldsson, T. (2021). Multi-Robot Routing Problem with Min – Max Objective. Robotics, 10(4), 122.

Denuit, M., Hainaut, D., & Trufin, J. (2019). Effective Statistical Learning

Methods for Actuaries III. Springer.

Din, S. U., Shao, J., Kumar, J., Ali, W., Liu, J., & Ye, Y. (2020). Online reliable semi-supervised learning on evolving data streams. Information Sciences, 525, 153–171. https://doi.org/10.1016/j.ins.2020.03.052

Din, S. U., Shao, J., Kumar, J., Mawuli, C. B., Mahmud, S. M. H., Zhang, W., & Yang, Q. (2021). Data stream classification with novel class detection : a review , comparison and challenges review , comparison and challenges. Knowledge and Information Systems, 63(9), 2231--2276. https://doi.org/10.1007/s10115-021-01582-4

Fahy, C., & Yang, S. (2019). Finding and Tracking Multi-Density Clusters in Online Dynamic Data Streams. IEEE Transactions on Big Data, 8(1), 178--192.

Fahy, C., Yang, S., & Gongora, M. (2018). Ant Colony Stream Clustering : A Fast Density Clustering Algorithm for Dynamic Data Streams. IEEE Transactions on Cybernetics, 49(6), 2215--2228.

Faris, H., Al-zoubi, A., Heidari, A. A., & Aljarah, I. (2019). An Intelligent System for Spam Detection and Identification of the most Relevant Features based on Evolutionary Random Weight Networks. Information Fusion, 48, 67--83.

Frolov, D., Radziewicz, W., Saienko, V., Kuchuk, N., Mozhaiev, M., Gnusov, Y., & Onishchenko, Y. (2021). Theoretical And Technological Aspects Of Intelligent Systems : Problems Of Artificial Intelligence. International Journal of Computer Science and Network Security, 21(5), 35--38.

Gao, Z., Liu, D., Huang, K., & Huang, Y. (2019). Context-Aware Human Activity and Smartphone Position-Mining with Motion Sensors. Remote Sensing, 11(21), 2531. https://doi.org/10.3390/rs11212531

Garcia, C., Leite, D., & Skrjanc, I. (2019). Incremental Missing-Data Imputation for Evolving Fuzzy Granular Prediction. IEEE Transactions on Fuzzy Systems, 28(10), 2348--2362. https://doi.org/10.1109/TFUZZ.2019.2935688

Ghani, U., Bajwa, I. S., & Ashfaq, A. (2018). A Fuzzy Logic Based Intelligent System for Measuring Customer Loyalty and Decision Making. Symmetry, 10(12), 761. https://doi.org/10.3390/sym10120761

Ghate, V., & C, S. H. (2021). Hybrid deep learning approaches for smartphone sensor-based human activity recognition. Multimedia Tools and Applications, 80(28), 35585--35604.

Gomes Bezerra, C., Sielly Jales Costa, B., Affonso Guedes, L., & Parvanov Angelov, P. (2020). An Evolving Approach to Data Streams Clustering Based on Typicality and Eccentricity Data Analytics. Information Sciences, 518, 13--28.

Gu, X., & Angelov, P. P. (2019). Self - Boosting First - Order Autonomous Learning Neuro - Fuzzy Systems. Applied Soft Computing, 77, 118--134. https://doi.org/10.1016/j.asoc.2019.01.005

Gupta, M. K., & Chandra, P. (2020). A comprehensive survey of data mining. International Journal of Information Technology, 12(4), 1243--1257. https://doi.org/10.1007/s41870-020-00427-7

Gupta, S. M. S. K. (2021). On fully intuitionistic fuzzy multiobjective transportation problems using different membership functions. Annals of Operations Research, 296(1), 211--241. https://doi.org/10.1007/s10479-019-03318-8

Halstead, B., Koh, Y. S., Riddle, P., Pears, R., Pechenizkiy, M., Bifet, A., Olivares, G., & Coulson, G. (2021). Analyzing and repairing concept drift adaptation in data stream classification. Machine Learning, 1--35. https://doi.org/10.1007/s10994-021-05993-w

Iliev, A., Kyurkchiev, N., & Markov, S. (2017). On the approximation of the step function by some sigmoid functions. Mathematics and Computers in Simulation, 133, 223--234. https://doi.org/10.1016/j.matcom.2015.11.005

Injadat, M., Moubayed, A., Nassif, A. B., & Shami, A. (2021). Machine Learning Towards Intelligent Systems: Applications, Challenges, and Opportunities. Artificial Intelligence Review, 54(5), 3299--3348.

Islam, M. K., Ahmed, M. M., & Zamli, K. Z. (2019). A buffer-based online clustering for evolving data stream. Information Sciences, 489, 113–135. https://doi.org/10.1016/j.ins.2019.03.022

Iuliis, M. De, Kammouh, O., Cimellaro, P. C., & Tesfamariam, S. (2019). DOWNTIME ESTIMATION OF BUILDING STRUCTURES USING FUZZY. International Journal of Disaster Risk Reduction, 34, 196--208. https://doi.org/10.1016/j.ijdrr.2018.11.017

Jameel, S. M., Hashmani, M. A., Rehman, M., & Budiman, A. (2020). An Adaptive Deep Learning Framework for Dynamic Image Classification in the Internet of Things Environment. Sensors, 20(20), 5811.

Jan, B., Farman, H., Khan, M., Imran, M., Islam, I. U., Ahmad, A., Ali, S., & Jeon, G. (2019). Deep learning in big data Analytics : A comparative study R. Computers and Electrical Engineering, 75, 275--287. https://doi.org/10.1016/j.compeleceng.2017.12.009

Javed, A. R., Beg, M. O., Asim, M., Baker, T., & Al-Bayatti, A. H. (2020). AlphaLogger: Detecting Motion-based Side-Channel Attack Using Smartphone Keystrokes. Journal of Ambient Intelligence and Humanized Computing, 1--14.

Javeed, M., Gochoo, M., Jalal, A., & Kim, K. (2021). HF-SPHR : Hybrid Features for Sustainable Physical Healthcare Pattern Recognition Using Deep Networks. Sustainability, 13(4), 1699.

Joseph, V. R., & Vakayil, A. (2022). SPlit : An Optimal Method for Data Splitting SPlit : An Optimal Method for Data Splitting ABSTRACT. Technometrics, 64(2), 166--176. https://doi.org/10.1080/00401706.2021.1921037

Joshi, A. V. (2020). Machine Learning and Artificial Intelligence. Springer.

Kabir, S., Islam, R. U., Hossain, M. S., & Andersson, K. (2020). An Integrated Approach of Belief Rule Base and Deep Learning to Predict Air Pollution. Sensors, 20(7), 1956.

Kaoungku, N., Suksut, K., Kerdprasop, R. C. K., & Kerdprasop, N. (2018). The Silhouette Width Criterion for Clustering and Association Mining to Select Image Features. International Journal of Machine Learning and Computing, 8(1), 69--73. https://doi.org/10.18178/ijmlc.2018.8.1.665

KARA, N., & KÖÇKEN, H. G. (2022). A golden section method for the multi-objective fractional solid transportation problem using the exponential membership function. Journal of Naval Sciences and Engineering, 18(1), 121–141.

Karhunen, J., Raiko, T., & Cho, K. (2015). Unsupervised deep learning: A short review. In Advances in Independent Component Analysis and Learning Machines. Elsevier Inc. https://doi.org/10.1016/B978-0-12-802806-3.00007-5

Kaur, N., & Sood, S. K. (2017). Efficient Resource Management System Based on 4Vs of Big Data Streams. Big Data Research, 9, 98--106.

References

Khaireddin, Y., & Chen, Z. (2021). Facial Emotion Recognition: State of the Art Performance on FER2013. ArXiv Preprint ArXiv:2105.03588.

Khairuddin, S. H., Hasan, M. H., Hashmani, M. A., & Azam, M. H. (2021). Generating Clustering-Based Interval Fuzzy Type-2 Triangular and Trapezoidal Membership Functions : A Structured Literature Review. Symmetry, 13(2), 239.

Khamassi, I., Sayed-Mouchaweh, M., Hammami, M., & Ghédira, K. (2018). Discussion and review on evolving data streams and concept drift adapting. Evolving Systems, 9(1), 1--23. https://doi.org/10.1007/s12530-016-9168-2

Khan, M. F., Hasan, M. G., Quddoos, A., Fügenschuh, A., & Hasan, S. S. (2020). Goal Programming Models with Linear and Exponential Fuzzy Preference Relations. SymmetrySymmetry, 12(6), 934. https://doi.org/10.3390/sym12060934

Kokate, U., Deshpande, A., Mahalle, P., & Patil, P. (2018). Data stream clustering techniques, applications, and models: Comparative analysis and discussion. Big Data and Cognitive Computing, 2(4), 1–30. https://doi.org/10.3390/bdcc2040032

Kolajo, T., Daramola, O., & Adebiyi, A. (2019). Big data stream analysis: a systematic literature review. Journal of Big Data, 6(1), 1--30. https://doi.org/10.1186/s40537-019-0210-7

Kour, H., Manhas, J., & Sharma, V. (2020). Usage and implementation of neuro-fuzzy systems for classification and prediction in the diagnosis of different types of medical disorders : a decade review. In Artificial Intelligence Review (Issue 53). Springer Netherlands. https://doi.org/10.1007/s10462-020-09804-x

Krawczyk, B., Minku, L. L., Gama, J., Stefanowski, J., & Woźniak, M. (2017). Ensemble learning for data stream analysis : A survey. Information Fusion, 37, 132–156. https://doi.org/10.1016/j.inffus.2017.02.004

Kumari, G., Chakraborty, J., & Nandy, A. (2020). Effect of Reduced Dimensionality on Deep learning for Human Activity Recognition. 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 1--7. https://doi.org/10.1109/ICCCNT49239.2020.9225419

Kyatham, A. S., Nichal, M. A., & Deore, B. S. (2020). A Novel Approach for Network Intrusion Detection using Probability Parameter to

Ensemble Machine Learning Models. Fourth International Conference on Computing Methodologies and Communication (ICCMC), 608–613.

Lee, A., Geem, Z. W., & Suh, K. (2016). Determination of Optimal Initial Weights of an Artificial Neural Network by Using the Harmony Search Algorithm : Application to Breakwater Armor Stones. Applied Sciences, 6(6), 164. https://doi.org/10.3390/app6060164

Leite, D., Škrjanc, I., & Gomide, F. (2020). An overview on evolving systems and learning from stream data. Evolving Systems, 11(2), 181–198. https://doi.org/10.1007/s12530-020-09334-5

Lengyel, A., & Botta-Dukát, Z. (2019). Silhouette width using generalized mean — A flexible method for assessing clustering efficiency. Ecology and Evolution, 9(23), 13231--13243. https://doi.org/10.1002/ece3.5774

Lima, W. S., Bragança, H. L. S., & Souto, E. J. P. (2021). NOHAR - NOvelty discrete data stream for Human Activity Recognition based on Smartphones with Inertial Sensors. Expert Systems with Applications, 166, 114093. https://doi.org/10.1016/j.eswa.2020.114093

Liu, W., Ci, L., & Liu, L. (2020). A New Method of Fuzzy Support Vector Machine Algorithm for Intrusion Detection. Applied Sciences, 10(3), 1065.

Lobo, J. L., Oregi, I., Bifet, A., & Del Ser, J. (2020). Exploiting a Stimuli Encoding Scheme of Spiking Neural Networks for Stream Learning. Neural Networks, 123, 118--133.

Lu, J., Zuo, H., & Zhang, G. (2019). Fuzzy Multiple-source Transfer Learning. IEEE Transactions on Fuzzy Systems, 28(12), 3418--3431.

Lu, L., & Zhou, J. (2021). Research on Mining of Applied Mathematics Educational Resources Based on Edge Computing and Data. Mobile Information Systems, 2021.

Luo, Y., & Schuur, E. A. G. (2020). Model Parameterization to Represent Processes at Unresolved Scales and Changing Properties of evolving Systems. Global Change Biology, 26(3), 1109--1117. https://doi.org/10.1111/gcb.14939

Lyu, Z., Yu, Y., Samali, B., Rashidi, M., Mohammadi, M., Nguyen, T. N., & Nguyen, A. (2022). Back-Propagation Neural Network Optimized by K-Fold Cross-Validation for Prediction of Torsional Strength of

Reinforced Concrete Beam. Materials, 15(4), 1477.

Manzoor, E., Lamba, H., & Akoglu, L. (2018). xStream : Outlier Dete ‘ x ’ ion in Feature-Evolving Data Stream s. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery \& Data Mining, 1963–1972.

Markoulidakis, I., Rallis, I., Georgoulas, I., Kopsiaftis, G., Doulamis, A., & Doulamis, N. (2021). Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem. Technologies, 9(4), 81.

Mehmood, H., Kostakos, P., Cortes, M., Anagnostopoulos, T., Pirttikangas, S., & Gilman, E. (2021). Concept Drift Adaptation Techniques in Distributed Environment for Real-World Data Streams. Smart Cities, 4(1), 349--371.

Michelon, G. K. (2020). Evolving System Families in Space and Time. Proceedings of the 24th ACM International Systems and Software Product Line Conference-Volume B, 104--111.

Moayedi, H., & Mosavi, A. (2021). Synthesizing Multi-Layer Perceptron Network with Ant Lion Biogeography-Based Dragonfly Algorithm Evolutionary Strategy Invasive Weed and League Champion Optimization Hybrid Algorithms in Predicting Heating Load in Residential Buildings. Sustainability, 13(6), 3198.

Mollá, N., Heavin, C., & Rabasa, A. (2022). Data-driven decision making: new opportunities for DSS in data stream contexts. Journal of Decision Systems, 1–15. https://doi.org/10.1080/12460125.2022.2071404

Mousavi, S. M., Tavana, M., Alikar, N., & Zandieh, M. (2019). A tuned hybrid intelligent fruit fly optimization algorithm for fuzzy rule generation and classification. Neural Computing and Applications, 31(3), 873–885. https://doi.org/10.1007/s00521-017-3115-4

Myo, W. W., Wettayaprasit, W., & Aiyarak, P. (2019). Designing Classifier For Human Activity Recognition Using Artificial Neural Network. 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), 81--85. https://doi.org/10.1109/CCOMS.2019.8821638

Nayak, D. R., Padhy, N., Mallick, P. K., Zymbler, M., & Kumar, S. (2022). Brain Tumor Classification Using Dense Efficient-Net. Axioms, 11(1), 34. https://doi.org/10.3390/axioms11010034

Obaid, H. S., Dheyab, S. A., & Sabry, S. S. (2019). The impact of data preprocessing techniques and dimensionality reduction on the accuracy of machine learning. 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), 279--283.

Ojha, V., Abraham, A., & Snášel, V. (2019). Heuristic design of fuzzy inference systems: A review of three decades of research. Engineering Applications of Artificial Intelligence, 85, 845–864. https://doi.org/10.1016/j.engappai.2019.08.010

Park, J., Yi, D., & Ji, S. (2020). A Novel Learning Rate Schedule in Optimization for Neural Networks and It ' s Convergence. Symmetry, 12(4), 660. https://doi.org/10.3390/sym12040660

Pinagé, F., dos Santos, E. M., & Gama, J. (2020). A drift detection method based on dynamic classifier selection. Data Mining and Knowledge Discovery, 34(1), 50–74. https://doi.org/10.1007/s10618-019-00656-w

Poepsel-lemaitre, R., Kiefer, M., Hein, J. Von, Quiané-Ruiz, J.-A., & Markl, V. (2021). In the Land of Data Streams where Synopses are Missing , One Framework to Bring Them All. Proceedings of the VLDB Endowment, 1818--1831. https://doi.org/10.14778/3467861.3467871

Qaiyum, S., Aziz, I., Jaafar, J., & Wong, A. K. L. (2019). Ant Colony Optimization of Interval Type - 2 Fuzzy C - Means with Subtractive Clustering and Multi - Round Sampling for Large Data. International Journal of Advanced Computer Science and Applications, 10(1). https://doi.org/10.14569/IJACSA.2019.0100106

Rácz, A., Bajusz, D., & Héberger, K. (2021). Effect of Dataset Size and Train/Test Split Ratios in QSAR/QSPR Multiclass Classification. Molecules, 26(4), 1111.

Rahman, S., Hasib, R., Sultana, B., Hussain, M. G., Rahman, M., & Rahaman, M. A. (2019). An Extensive Karnaugh Mapping Tool for Boolean Expression Simplification. International Conference on Sustainable Technologies for Industry 4.0 (STI), 1-- 5. https://doi.org/10.1109/STI47673.2019.9068037

Ramírez-Gallego, S., Krawczyk, B., García, S., Woźniak, M., & Herrera, F. (2017). A survey on data preprocessing for data stream mining: Current status and future directions. Neurocomputing, 239, 39–57. https://doi.org/10.1016/j.neucom.2017.01.078

Ramírez, A., Moreno, N., & Vallecillo, A. (2021). Rule-based preprocessing for data stream mining using complex event processing. Expert Systems, 38(8), e12762. https://doi.org/10.1111/exsy.12762

S. Priya & R. A. Ythra. (2021). Deep learning framework for handling concept drift and class imbalanced complex decision - making on streaming data. Complex and Intelligent Systems, 1--17. https://doi.org/10.1007/s40747-021-00456-0

Sarfraz, M. S., Sharma, V., & Stiefelhagen, R. (2019). Efficient Parameter-free Clustering Using First Neighbor Relations. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8934–8943.

Schmucker, B., Trautwein, F., Semm, T., Lechler, A., Zaeh, M. F., & Verl, A. (2021). Implementation of an Intelligent System Architecture for Process Monitoring of Machine Tools. CIRPe 2020 – 8th CIRP Global Web Conference – Flexible Mass Customisation, 96, 342–346. https://doi.org/10.1016/j.procir.2021.01.097

Seedat, N., Crabbe, J., & Schaar, M. Van Der. (2022). Data-SUITE: Data-centric identification of in-distribution incongruous examples. ArXiv:2202.08836.

Selim, G. E. I., Hemdan, E. E.-D., Shehata, A. M., & El-fishawy, N. A. (2021). An efficient machine learning model for malicious activities recognition in water-based industrial internet of things. Security and Privacy, 4(3), e154. https://doi.org/10.1002/spy2.154

Shi, Y., Wu, H., & Li, C. (2022). Constrained hybrid control for parametric uncertainty systems via step-function method. Mathematical Biosciences and Engineering, 19(11), 10741–10761.

Shyam, R., & Singh, R. (2021). A Taxonomy of Machine Learning Techniques. Journal of Advancements in Robotics, 9(3), 18--25p.

Singh, K., & Mathai, K. J. (2019). Performance Comparison of Intrusion Detection System Between Deep Belief Network (DBN) Algorithm and State Preserving Extreme Learning Machine (SPELM) Algorithm. IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), 1--7.

Škrjanc, I., Andonovski, G., Ledezma, A., Sipele, O., Iglesias, J. A., & Sanchis, A. (2018). Evolving cloud-based system for the recognition of drivers ' actions. Expert Systems with Applications, 99, 231--238. https://doi.org/10.1016/j.eswa.2017.11.008

Škrjanc, I., Iglesias, J. A., Sanchis, A., Leite, D., Lughofer, E., & Gomide, F. (2019). Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A Survey. Information Sciences, 490, 344–368. https://doi.org/10.1016/j.ins.2019.03.060

Skrjanc, I., Ozawa, S., Ban, T., & Dovˇzan, D. (2018). Large-scale cyber attacks monitoring using Evolving Cauchy Possibilistic Clustering. Applied Soft Computing, 62, 592--601. https://doi.org/10.1016/j.asoc.2017.11.008

Souza, V. M. A., dos Reis, D. M., Maletzke, A. G., & Batista, G. E. A. P. A. (2020). Challenges in benchmarking stream learning algorithms with real-world data. In Data Mining and Knowledge Discovery (Vol. 34, Issue 6). Springer US. https://doi.org/10.1007/s10618-020-00698-5

Sun, Y., Yang, G., & Lo, B. (2018). An Artificial Neural Network Framework for Lower Limb Motion Signal Estimation with Foot-Mounted Inertial Sensors. 2018 IEEE 15th International Conference on Wearable and Implantable Body Sensor Networks (BSN), 132--135.

Sung, M.-C., Orimoloye, L. O., Ma, T., & Johnson, J. E. V. (2020). Comparing the effectiveness of deep feedforward neural networks and shallow architectures for predicting stock price indices. Expert Systems with Applications, 139, 112828.

Tavakoli, P., Vaezi, N., Fahim, P., & Karimpour, A. (2021). A New Approach Based on Fuzzy-Adaptive Structure & Parameter Learning Applied in Meta-Cognitive Algorithm for ANFIS. 7th International Conference on Control, Instrumentation and Automation (ICCIA), 1--5.

Teixeira, L. H., & Huszák, Á. (2022). Reinforcement Learning Environment for Advanced Vehicular Ad Hoc Networks Communication Systems. Sensors, 22(13), 4732.

Tong, W., Liu, S., & Gao, X. (2021). A density-peak-based clustering algorithm of automatically determining the number of clusters. Neurocomputing, 458, 655--666. https://doi.org/10.1016/j.neucom.2020.03.125

VARGAS, R., & RUİZ, L. (2017). DEEP LEARNING : PREVIOUS AND PRESENT APPLICATIONS. Journal of Awareness, 2(Special 3), 11-- 20.

Vieira, S., Hugo, W., Pinaya, L., Garcia-dias, R., & Mechelli, A. (2020). Deep neural networks. In Machine Learning. Elsevier Inc. https://doi.org/10.1016/B978-0-12-815739-8.00009-2

Vinayakumar, R., Kp, S., & Poornachandran, P. (2017). Evaluating Effectiveness of Shallow and Deep Networks to Intrusion Detection System. International Conference on Advances in Computing, Communications and Informatics (ICACCI), 1282--1289.

Wan, S., Qi, L., Xu, X., Tong, C., & Gu, Z. (2020). Deep Learning Models for Real-time Human Activity Recognition with Smartphones. Mobile Networks and Applications, 25(2), 743--755.

Wang, S., & Zhu, X. (2020). A Hybrid Deep Neural Networks for Sensor-based Human Activity Recognition. 12th International Conference on Advanced Computational Intelligence (ICACI), 486--491.

Wang, X. (2020). Frequently-Used Properties of the Floor Function. International Journal of Applied Physics and Mathematics, 10(4), 135--142. https://doi.org/10.17706/ijamp.2020.10.4.135-142

Wares, S., Isaacs, J., & Elyan, E. (2019). Data stream mining: methods and challenges for handling concept drift. SN Applied Sciences, 1(11). https://doi.org/10.1007/s42452-019-1433-0

Weiss, M. N., Franks, D. W., Brent, L. J. N., Ellis, S., Silk, M. J., & Croft, D. P. (2021). Common datastream permutations of animal social network data are not appropriate for hypothesis testing using regression models. Methods in Ecology and Evolution, 12(2), 255–265. https://doi.org/10.1111/2041-210X.13508

Westby, I., Yang, X., Liu, T., & Xu, H. (2021). FPGA Acceleration on a Multi-Layer Perceptron Neural Network for Digit Recognition. The Journal of Supercomputing, 77(12), 14356--14373.

Wu, Y., Liu, L., Bae, J., & Chow, K. (2019). Demystifying Learning Rate Policies for High Accuracy Training of Deep Neural Networks. 2019 IEEE International Conference on Big Data (Big Data), 1971--1980.

Yang, D., Jiang, C., Cai, G., & Huang, N. (2019). Optimal sizing of a wind / solar / battery / diesel hybrid microgrid based on typical scenarios considering meteorological variability. IET Renewable Power Generation, 13(9), 1446--1455. https://doi.org/10.1049/iet-rpg.2018.5944

Zhang, J., Song, X., Jing, X., Yang, G., Yang, C., Feng, H., Wang, J., & Ming, S. (2022). Remote Sensing Monitoring of Rice Grain Protein

Content Based on a Multidimensional Euclidean Distance Method. Remote Sensing, 14(16), 3989.

Zhong, G., Ling, X., & Wang, L.-N. (2018). From shallow feature learning to deep learning : Benefits from the width and depth of deep architectures. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 9(1), e1255. https://doi.org/10.1002/widm.1255

Zubaroglu, A., & Atalay, V. (2022). Online embedding and clustering of evolving data streams. Statistical Analysis and Data Mining: The ASA Data Science Journal, 1–16. https://doi.org/10.1002/sam.11590

الملخص

معظم تطبيقات العالم الحقيقي قادرة على توليد دفق البيانات. دفق البيانات لها خصائص فريدة و مشاكل مختلفة كما أنها تتغير بمرور الوقت. يتم تقديم النظام المتطور كحل مناسب للتغلب على مشكلات التنقيب في دفق البيانات بسبب القدرة على تغيير هيكله وفقًا لتغيير سلوك البيانات. لذلك، تقدم هذه الرسالة نظامًا ذكيًا متطورًا يعتمد على تقنية التجميع لحل بعض مشاكل التنقيب في دفق البيانات.

في النظام المقترح، يتم تطبيق خوارزمية كوشي المتطورة (e-Cauchy). على الرغم من أن هذه الخوارزمية هي خوارزمية ناجحة لتجميع دفق البيانات، إلا أنها تعاني من العدد الكبير من المجموعات المتولدة، لذلك تم تطوير هذه الخوارزمية للتغلب على هذا القيد. علاوة على ذلك، فإن مشكلة التجميع العامة هي التقييم، والذي يتم حله باستخدام طريقة التعلم العميق وهذا يعني أنه يتم استخدامه كمقيم خارجي. ومع ذلك، فإن هذه الطريقة تزيد من دقة الشبكة وتضمن استقرارها من خلال تدريب الشبكة بقيمة معدل تعلم غير ثابت (ديناميكي). وفقًا لذلك ، يتكون النظام المقترح من نموذجين، نموذج التعلم العميق (وضع الاتصال) ونموذج التجميع المتطور (وضع غير متصل).

يهدف النظام المقترح إلى اتخاذ قرار مبني على قواعد غامضة يتغير بمرور الوقت لتعيين عينة دفق البيانات إلى الفئة بدقة. بالإضافة إلى ذلك، فإن النظام المقترح قادر على اكتشاف عينات البيانات الجديدة (التي لم يتم تدريبها مسبقًا) ومن ثم، إنشاء مجموعة جديدة.

عدد مجموعات البيانات المستخدمة لاختبار النظام المقترح هو أحد عشر مجموعة بيانات تدفق تم إنشاؤها من تطبيقات مختلفة مثل، أجهزة الاستشعار، أنظمة المراقبة والأجهزة وبمجالات متنوعة مثل الاستبيان، التعرف على السلوك والتعرف على النشاط البشري. علاوة على ذلك، تحتوي مجموعات البيانات هذه على بيئات مختلفة، تعقيدات مختلفة، وعدد مختلف من الميزات والفئات.

قياسات التقييم لهذا النظام هي: الدقة، الإِتقان، الاسترجاع، درجة F1 ومعامل الصورة الظلية. تراوحت الدقة، الإِتقان، الاسترجاع ودرجة F1 من ٨٨,٧٥٪ ، ٨٨,٧٤٪ ،٨٨,٧٥٪ و ٨٨,٧٤٪ إلى (١٠٠%، ١٠٠%، ١٠٠% و ١٠٠%). بينما تراوح معدل التحسن لمعامل الصورة الظلية من ٠,٢٨ الى ٠,٩٩ بين جميع مجموعات البيانات.

جمهورية العراق

وزارة التعليم العالي والبحث العلمي

جامعـــة بـابــل

كلية تكنولوجيا المعلومات

قسم البرمجيات

تعدين تدفق البيانات بإستخدام نظام ذكي متطور

اطروحة مقدمة إلى

مجلس كلية تكنولوجيا المعلومات -جامعة بابل كجزء من متطلبات

نيل درجة الدكتوراه في تكنولوجيا المعلومات

من قبل

حسين عبد الأمير عباس فـــهد

بإشــراف

أ.د. نبيل هاشم كاغد جارالله

أ.د. إيمان صالح صكبان ناصر

١٤٤٤ هـ ٢٠٢٢ م