Communications in Mathematics and Applications
Vol. 15, No. 4, pp. 1317–1330, 2024
ISSN 0975-8607 (online); 0976-5905 (print)
Published by RGN Publications
DOI: 10.26713/cma.v15i4.3260

http://www.rgnpublications.com

***Special Issue*:**
Frontiers in Applied and Computational Mathematics
Editors: M. Vishu Kumar, S. Lakshmi Narayana, B. N. Hanumagowda, U. Vijaya Chandra Kumar

Research Article

Multi-Label Classification for Drift Detection in IoT Data Streams

Mashail Althabiti* , Manal Abdullah and Omaima Almatrafi

Department of Computer Information System, King Abdulaziz University, Jeddah, Saudi Arabia
***Corresponding author:** malthabiti0001@stu.kau.edu.sa

Received: August 22, 2024 **Accepted:** December 6, 2024

Abstract. With the prevalence of *Internet of Things* (IoT) systems, data is exponentially growing, resulting in data streams. Data streams are massive, potentially non-stop, and arrive at high speed. These systems process IoT data streams in a non-stationary manner, making them susceptible to concept drift occurrence and class imbalance. Concept drift occurs as a result of the change in the underlying distribution over time, while class imbalance occurs when some class distribution is uneven. This paper proposes an *Implicit Drift Detection model with Multi-Label kNN* (IDD-MLkNN), aimed at addressing concept drift in multi-label classification for IoT data streams. While the model is applicable across various domains, its performance was specifically assessed using two IoT datasets — Bot_IoT and Edge_IIoTset — associated with intrusion detection systems. Applying the proposed model to oversee IoT network traffic offers practical advantages, potentially reducing the time and expenses of re-examining attack data. The evaluation was conducted for sudden and gradual concept drift scenarios. Experimental results show the superiority of the IDD-MLkNN over other well-known multilabel classification models in different performance measures such as the Subset Accuracy, Accuracy, Hamming Score, and F-measure. However, it was less efficient in terms of Evaluation Time compared to other multi-label classification methods.

Keywords. Multilabel classification, Data stream, IoT, Concept drift, Class imbalance, kNN

Mathematics Subject Classification (2020). 68W01

1. Introduction

The *Internet of Things* (IoT) enables devices and networks to interconnect and collaborate, facilitating daily tasks among interconnected entities in smart homes and cities (Ferrag *et al.* [5]). Data transmitted within IoT systems are not static. They exhibit heterogeneity and dynamism, leading to the emergence of data streams. Data streams are high-volume, sequenced data. Learning from such data is a complex task. Unlike static data, the data stream is potentially unbounded, massive, and processed in online mode (Gama *et al.* [6]). It arrives at high speed, making classification from such data challenging.

Classification algorithms must adapt to data stream properties to learn effectively. One of the main concerns is that data instances are accessed once. In addition, the data stream is also expected to evolve, leading to *Concept Drift* (CD) [6], which is the unpredictable change in the underlying data distribution. It could take different forms, such as the change in the input or the relation between the input and the output.

CD can occur in four different ways: suddenly, gradually, incrementally, and recurrently, as presented in Figure 1 ([6]). Sudden drift happens when the two concepts change suddenly from one to another. In gradual drift, the second concept takes longer to replace the first one. Incremental drift occurs when many intermediate concepts in between consist incrementally. Recurrent drift is a type of drift that has been seen before and reappears after a certain time.

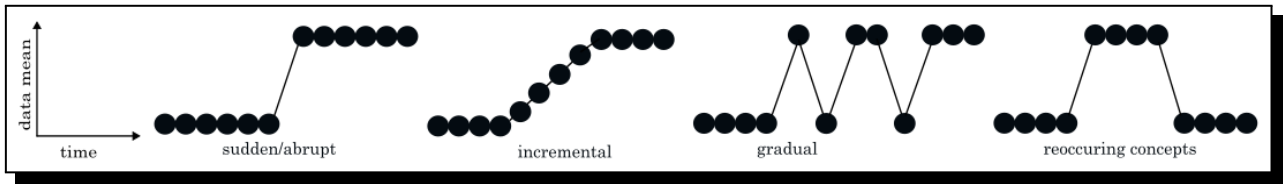


Figure 1. Types of concept drift in machine learning [6]

The dynamic nature of IoT data poses a significant challenge in different real-world applications. For example, in tasks such as image annotation, activity recognition, text classification, or *Intrusion Detection Systems* (IDS), an example may be linked with more than one label, a task known as *Multi-label Classification* (MLC). MLC involves classifying data into more than one category, as opposed to binary classification, where the output is a single class label.

This paper aims to introduce a multi-label classification model for IoT data streams. The model, called *Implicit Drift Detection with Multi-label kNN* (IDD-MLkNN), is based on a multilabel k nearest neighbor algorithm. The IDD-MLkNN model is primarily designed to adapt to concept drift in data with multiple labels. The proposed model has undergone validation using IoT datasets with class imbalance. Furthermore, its performance has been benchmarked against various multi-label classifiers, ensuring the reliability and robustness of the proposed model.

Among different potential applications, the proposed model is evaluated on IDS, which plays a vital role in network security (Ferrag *et al.* [5]). Prior works have utilized machine learning methods to detect the drift and attacks in network traffic (Andresini *et al.* [1], Kuppa and Le-Khac [11], Mulimani *et al.* [12]). Attack methods and behaviors constantly evolve as malicious actors develop them. This evolution results in concept drift, where normal and adversarial traffic characteristics change over time. In addition, a single data record might indicate multiple states or categories in numerous cyber scenarios (e.g., detecting an attack and its location in

smart grids (Hallaji *et al.* [7])). The proposed model is designed to classify instances for multiple labels and undergoes evaluation to assess its adaptability to concept drift.

This article is structured as follows: Section 2 presents related work on multi-label classification approaches and the methods used to address concept drift. Section 3 presents the proposed model in detail, along with a flowchart. Section 4 details the IoT datasets and experimental settings, while section five provides the discussions and results. Lastly, Section 6 presents the conclusion and future work.

2. Related Work

Problem Transformation (PT) and *Algorithm Adaptation* (AA) are two main broad approaches to deal with multi-label classification. In the PT approach, multi-label problems are transformed into other, more minor problems, which are then addressed by single-label methods (Zhang and Zhou [25]). Binary relevance is a well-known PT method, which transforms the multi-label issue into many smaller issues, each solved by binary classification. *Classifier Chains* (CC) are another PT that decomposes the problem into smaller problems, similar to the binary relevance method (Read *et al.* [18]). CC trains a chain of binary classifiers and utilizes the current classifier's prediction in building subsequent binary classifiers.

In the realm of streaming environments, Nguyen *et al.* [13] have developed an incremental MLC method based on the Bayesian theorem and the correlation between labels. It tackles drift by assigning weights to instances, with recent instances having the highest weight. Also, Sousa and Gama [20] have proposed MLC method utilizing a multi-target AMRules regressor called Ml-AMRules. It works well in data streams and uses *Page-Hinkley* (PH) for drift detection. Xioufis *et al.* [23] have developed a MLC method with multiple windows. It monitors two sliding windows for drift detection and uses sampling techniques to address data stream imbalance.

Problem transformation methods can also leverage ensemble for better performance. Wei *et al.* [22] have introduced a multi-label model for data streams with a weighted ensemble. It divides the stream into chunks, where each chunk is transformed into single-label data. These transformed chunks are then used to train an ensemble of binary classifiers, including support vector machine (SVM), Naive Bayes (NV), and C4.5. Classifiers are weighted according to the expected classification accuracy of the test data. Qu *et al.* [16] also developed a dynamic weighted-based ensemble method, which considers the dependence information between labels. The algorithm employs an ensemble of classifiers that keeps only the recently trained classifier and discards the old ones. This would make it better to represent the current concept in stream applications since data are evolving and changing. It works by partitioning the incoming stream into chunks, where the last one represents the recent chunk. Chunks are utilized for training the improved binary relevance classifier, which results in more features. The chunk produced is then used to train an Improved Binary Relevance classifier.

Furthermore, multilabel classification could be addressed by algorithm adaptation methods, in which a traditional classifier is modified to cope with the nature of multi-label data. The Multi-Label k-Nearest Neighbor (ML-kNN) is a well-known AA method based on traditional kNN (Zhang and Zhou [26]). It identifies the k nearest neighbor for each instance and then finds the relevant labels for each neighbor. *Maximum a Posteriori* (MAP) is then used to predict

the labels. In the stream setting, MLC could utilize an ensemble of classifiers under the AA approach. Kong and Yu [9] have introduced a method for MLC with drift detection. It uses a set of random trees and considers the label correlations. The tree node's statistics, such as label relevance and cardinality, are updated with the stream. Then, an ensemble of random trees is trained, and the final results are averaged over all the trees. For drift detection, a fading factor is attached to the tree nodes, which also reduces the impact of the old data. Similarly, Read *et al.* [17] proposed a hoeffding trees-based method for multi-label drifting streams called EaHTps. The only difference is its way of dealing with concept drift. It employs a drift detection method called ADWIN-bagging ensemble, which adapts the concept drift in the evolving data.

Osojnik *et al.* [14] have proposed a decision tree-based method, named iSOUP-Tree. It is a bagging method that implements adaptive perceptions in the leaves. Consequently, it effectively deals with MLC and multi-target regression problems in a non-stationary environment. For drift detection, the Page Hinckley detector is employed. Buyukcakir *et al.* [3] have introduced a MLC model based on Geometrically Optimum. It splits the stream into fixed-sized chunks and utilizes the spatial model to assign weights to the classifiers. The proposed model has also been evaluated with ADWIN, the well-known drift detection method.

In the Internet of Things context, MLC is utilized to recognize human activities in smart homes, such as described by Jethanandani *et al.* [8]. Various MLC methods, including Classifier Chain, *Naïve Bayes* (NV), Decision Tree, *k-Nearest Neighbors* (kNN), and Logistic Regression, were used to classify multi-resident activities in smart homes. Nevertheless, concept drift (for example, changes in the resident activities) and class imbalance were not considered. Hallaji *et al.* [7] have introduced a network-based IDS that utilizes MLC methods. The system implements an ensemble of multi-label neural network models, each performing a single-label classification. The final labelset is made by aggregating the results after each model's prediction. The proposed algorithm by Xu *et al.* [24] uses a *Recurrent Neural Network* (RNN) to discover anomalies in IoT datasets collected from smart homes. Anomalies detected can belong to more than one category. In studies of Hallaji *et al.* [7] and Xu *et al.* [24], concept drift detection is adopted, while class imbalance is not addressed. Furthermore, Pezze *et al.* [15] have proposed a multilabel classification method to monitor packaging equipment. The method has also been assessed on an industrial IoT dataset that has class imbalance and distribution shifts. However, the authors highlighted points for further improvement, such as testing the method in various real-world applications, especially those with significant label imbalances.

The literature shows that most studies employing MLC in IoT applications fail to consider both concept drift and imbalance. While considerable research has been conducted on addressing concept drift in single-label data streams, the challenges associated with multilabel data streams still need to be explored, especially in imbalanced IoT data streams. Class imbalance is common in data streams where the number of instances with a particular set of labels is greater than those with another set of labels. Addressing these challenges jointly is crucial for better predictive performance. Thus, this paper introduces an MLC model for handling *Concept Drift* (CD) and class imbalance in IoT streams. It evaluates the proposed model's in predicting attacks, their category, and subcategories in IoT network traffic datasets by comparing it with well-known models. The evaluation was conducted on different concept drift scenarios, including sudden and gradual concept changes.

3. Implicit Drift Detection with Multi-label kNN (IDD-MLkNN)

DD-MLkNN is a multi-label classification model that handles concept drift and class imbalance in IoT data streams. It entails three main steps: predict, diagnose, and update. The first step is the prediction, where a set of labels is produced for the given instance. During classification, the diagnosis step is carried out to diagnose concept drift in the IoT data stream. The last step is updating, which is activated according to the diagnosis estimations. Figure 2 presents the flow chart of the IDD-MLkNN model.

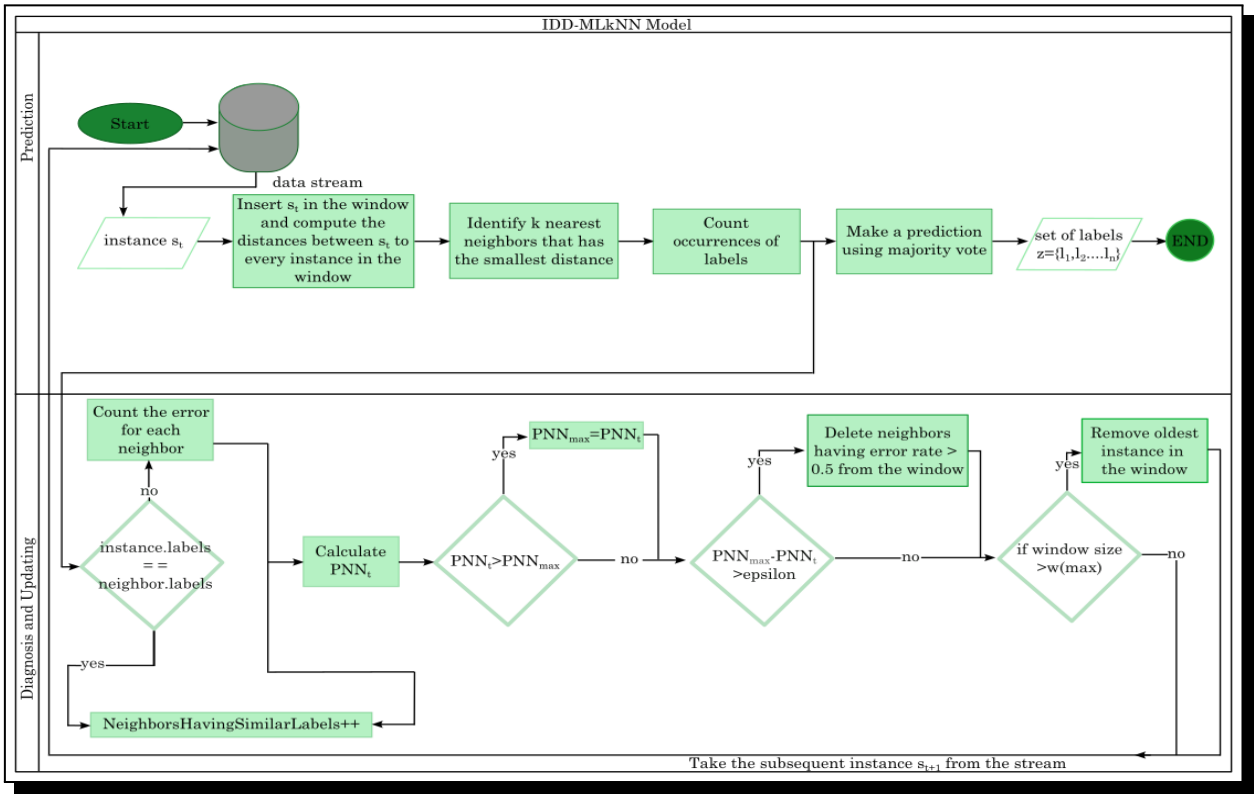


Figure 2. Flowchart of IDD-MLkNN

The input to the IDD-MLkNN model is labeled instances in the form of {X, Y}, where X represents a set of features and Y represents a set of labels. For example, an instance in the Flags dataset arrives with many features (X) such as the country landmass, language, religion, and many others, and seven labels (Y) representing the flag colors[1]. An instance s_t is passed to the FIFO sliding window. The sliding window is dynamic, so its size is adjusted with each incoming instance; at the same it has a maximum size that cannot be exceeded.

The underlying algorithm for the prediction phase in IDD-MLkNN is ML-kNN, which works like a traditional k nearest neighbor but with a majority vote mechanism. Initially, it finds the Euclidean distance between the new instance (s_t) and all instances in the sliding window to identify the k nearest neighbors. Subsequently, the most appearing labels among these k neighbors determine the final predicted label set (a voting mechanism). To better understand the concept, for example, suppose we have a Flags dataset as described earlier. If $k = 3$, three

[1]*Multi-Label Classification Dataset Repository*, accessed: May 13, 2024, URL: https://www.uco.es/kdis/mllresources/#GoncalvesEtAl2013

instances are selected as nearest neighbors to the new instance. The first neighbor has three labels {Green, Yellow, White}, the second neighbor has two labels {Green, White}, and the last neighbor has two labels {White, Red}. From labels among the selected neighbors, the new instance will be classified as Green and White because most of the instances are from class white and class green. Mathematically, labels with a relative frequency greater than 0.5 are selected in the predicted label set, (number of label occurrences)/k. Thus, Red and Yellow are not selected since their relative frequency is 0.3, less than 0.5. During the classification process, IDD-MLkNN implicitly evaluates the new instances's label set against the labelset of each nearest neighbor. It counts each neighbor's error, where each unmatched label in the label set is counted as 1. If there are two unmatched labels, then the error becomes 2, and so on. After that, the probability of getting a neighbor's label set identical to the new instance's label set is computed after each prediction made by the classifier which is calculated as

$$\mathrm{PNN}_t = \frac{\text{Number of neighbours with label set identical to } s_t \text{ label set}}{k},$$

IDD-MLkNN assumes that low similarity suggests a potential drift and change in the data distribution. Therefore, it keeps storing the highest value reached during the process in a variable called $\mathrm{PNN}_{\mathrm{Max}}$ to identify significant changes by comparing the similarity at a given time t with the highest similarity reached during the learning process. If the difference between $\mathrm{PNN}_{\mathrm{Max}}$ and PNN_t exceeds a certain epsilon value, it removes instances with high error rates above 0.5 to keep a balanced class distribution in the sliding window. The error rate is computed for each neighbor as the number of errors divided by the number of labels. IDD-MLkNN allows instances with low error rates to remain in the window since they might represent a new concept and need more instances to form a different data distribution.

This technique also implicitly deals with class imbalance instead of the preprocessing method. In a streaming environment, the majority label set may become a minority as time passes. It assumes that the majority label set skews the classifier, causing errors. The removal process here is deemed as undersampling as it is introduced by Roseberry *et al.* [19], enabling the minority instances to be present in the sliding window and ensuring a balanced class distribution.

After that, the window size is evaluated whether instances are removed or no significant change between the $\mathrm{PNN}_{\mathrm{Max}}$ and PNN_t is found. IDD-MLkNN maintains a sliding window with a maximum size. It works with different window sizes until it reaches its pre-defined maximum size. In this case, the old instance is removed, making space for a new instance to be inserted.

4. Experiments

IDD-MLkNN model is written in JAVA and implemented in *Massive Online Analysis* (MOA) software [2]. MOA is a widely used tool for executing online machine-learning algorithms and conducting stream experiments. This section overviews the IoT datasets used and presents their characteristics. It also outlines the designed scenarios that simulate the sudden and gradual drift and the evaluation measures used to assess the IDD-MLkNN against the reference algorithms.

An analysis was performed to showcase the effectiveness of IDD-MLkNN in MLC. The model performance was benchmarked against five established multilabel classification methods:

ML-kNN (Zhang and Zhou [26]), ML-NV (based on the Binary Relevance approach) (Zhang and Zhou [25]), EaHTps (Read *et al.* [17]), iSOUP-Tree (Osojnik *et al.* [14]), and Ml-AMRules (Sousa and Gama [20]) all of which have been described in Section 2, and their code available in MOA. The IDD-MLkNN's window size is 1000, k is three, and the threshold is one. The other algorithms' parameters were set to their defaults in MOA, presented in Table 1.

Table 1. The algorithms parameters values

Method	Parameters
ML-kNN [25]	$K = 10$, Limit = 1,000
EaHTps [17]	Baselearner: Hoeffding tree, EnsembleSize = 10, and changeDetector = ADWIN
iSOUP-Tree [14]	splitConfidence: $1.0E^{-7}$, tieThreshold: 0.05, PageHinckleyAlpha: 0.005, and PageHinckleyThreshold: 50
Ml-AMRules [20]	splitConfidence: $1.0E^{-7}$, tieThreshold: 0.05, Baselearner: MultiLabelNaive-Bayes, and changeDetector = DDM
IDD-MLkNN	$K = 3$, MaximumWindowSize = 1,000, Threshold = 1

4.1 Datasets

To assess the IDD-MLkNN's ability to adapt to concept drift on multi-label IoT data classification. Two network IoT/IIoT datasets were utilized and processed into multi-label datasets in Arff format, in which class appearance is represented by one. The datasets used in the experiments of this study are described below:

4.1.1 The Bot_IoT Dataset

Bot_IoT is a realistic dataset that simulates IoT network traffic, with various types of attacks and normal traffic (Koroniotis *et al.* [10]). It includes *Distributed Denial of Service* (DDoS) and *Denial of Service* (DoS) with different protocols. It also contains OS fingerprint, service scan, keylogging, and data exfiltration attacks. The dataset consists of around a million instances, each instance has 43 features and 3 class labels. The first label indicates normal or attack traffic. The second label and the third label identify the type of attack and its subcategory. For instance, the label set {Attack, DoS, HTTP} shows the type and sub-type of the attack, which specify the protocol used.

Bot_IoT dataset has been preprocessed to ensure the existence of drift and class imbalance. A random sample of 100,000 instances has been selected with ten features to evaluate the proposed model and 12 labels. These features are selected as the best in [10] including Seq, Stddev, N_IN_Conn_P_SrcIP, Min, State_number, Mean, N_IN_Conn_DstIP, Drate, Srate, and Max. Two concepts (C1 and C2) are designed; each has different label sets with varying numbers of instances forming the class imbalance, as shown in Table 2. The first concept contains only normal traffic and reconnaissance attacks with {Attack, Reconnaissance, Service _Scan} label set as majority and {Normal, Normal, Normal} as minority class label. The second concept involves a different type of DDoS and DoS attacks with different protocols.

4.1.2 The Edge_IIoTset Dataset

Edge-IIoTset is a real-world cyber security dataset generated using the Internet of Things/ Industrial Internet of Things testbeds with several devices and sensors (Ferrag *et al.* [5]).

The attacks are classified into categories including DoS/DDoS attacks, information gathering, injection attacks, man-in-the-middle attacks, and malware attacks. It comprises 157,800 instances, with 61 features and two class labels; the first label indicates normal or attack, and the second shows the type of attack.

Moreover, a sample of 100,000 instances has been randomly selected form Edge_IIoTset and preprocessed to simulate the drift by designing two concepts with 23 features as selected by the authors in [5] and 12 labels. Table 2 shows the set of labels available. The first concept contains normal distribution and DDoS attacks with different techniques, including TCP, UDP, HTTP, and ICMP. The second concept is based on normal distribution and Information-gathering attacks, which use three techniques: port scanning, OS fingerprinting, and vulnerability scanning. It also includes man-in-the-middle attacks using two protocols, DNS and ARP.

Table 2. The preprocessed Bot_IoT and Edge_IIoTset dataset

Bot_IoT dataset			Edge_IIoTset dataset		
	The label set for each instance {l1, l2, l3}	Number of instances		The label set for each instance {l1, l2}	Number of instances
C1	{Normal, Normal, Normal}	477	C1	{Normal, Normal}	1238
	{Attack, Reconnaissance, OS_Fingerprint}	9,942		{Attack, Attack, DDoS_HTTP}	10,361
	{Attack, Reconnaissance, Service_Scan}	39,5812		{Attack, DDoS_ICMP}	14,026
				{Attack, DDoS_TCP}	10,151
				{Attack, DDoS_UDP}	14,224
C2	{Attack, DDoS, UDP}	13,332	C2	{Normal, Normal}	17,534
	{Attack, DDoS, TCP}	13,668		{Attack, Fingerprinting}	778
	{Attack, DDoS, HTTP}	15		{Attack, Port_Scanning}	7,715
	{Attack, DoS, UDP}	14,392		{Attack, SQL_injection}	7,933
	{Attack, DoS, TCP}	8,574		{Attack, Vulnerability_scanner}	7,684
	{Attack, DoS, TCP}	8,574		{Attack, Vulnerability_scanner}	7,684
	{Attack, DoS, HTTP}	19		{Attack, XSS}	8,356
Total		100,000	Total		100,000

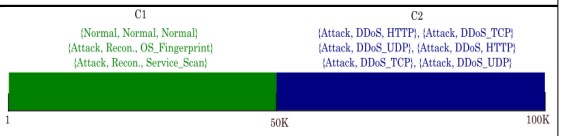
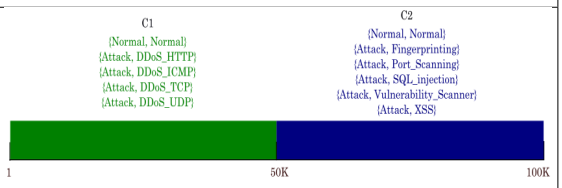
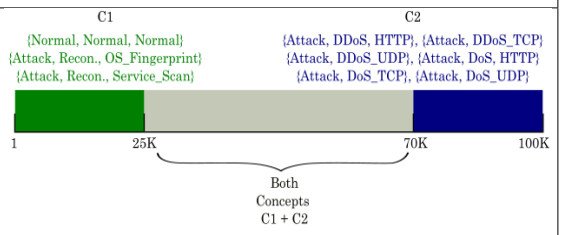
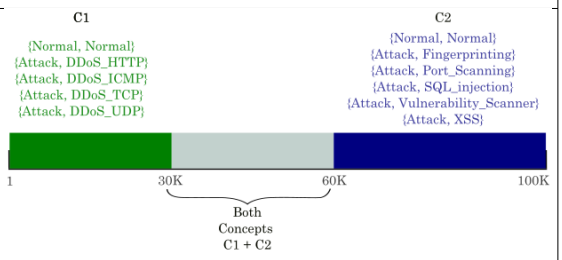
4.2 Drift Scenarios

Drift is simulated in the extracted and preprocessed two datasets by changing the distribution of network data streams and replacing the first concept with the second. In this study, sudden and gradual drift were are selected for simplicity forming different scenarios, as proposed in Table 3. For example, the first scenario involves two concepts that abruptly replace each other. The model will initially process 50,000 instances related to normal activity and two types of reconnaissance attacks. After that, this concept will no longer be available, as new one with DDoS and DoS attacks, emerge. Moreover, gradual drift is simulated in scenario three, where the second concept emerges after 25,000 instances and takes around 45,000 instances to completely replace the first concept.

4.3 Evaluation Measures

The evaluation method used in the experiment is prequential, where the classifier is evaluated on the stream by testing first and then training and updating the model with each instance in

Table 3. The proposed drift scenarios

Dataset	Drift type	# of instances	Description
Bot-IoT (12 labels)	Scenario 1 - Sudden	100,000	
Edge-IIoTset (12 labels)	Scenario 2 - Sudden	100,000	
Bot-IoT (12 labels)	Scenario 3 - Gradual	100,000	
Edge-IIoTset (12 labels)	Scenario 4 - Gradual	100,000	

sequence (Wang *et al.* [21]). The evaluation of MLC is different from that used in the single-label classification. Five measures are used to evaluate IDD-MLkNN performance: (1) Subset Accuracy, which checks all the predicted labels to match exactly the truth label set; Accuracy, which divides the number of correctly predicted labels by the sum of the predicted and true labels, (3) Hamming Score, which finds the fraction of labels correctly predicted, averaged over the instances and labels; (4) F-Measure, which is the mean of the precision and recall, and (5) Evaluation Time, the time spent to run the classifier.

5. Results and Discussion

The results of evaluating IDD-MLkNN are organized into two sub-sections according to the type of drift addressed below:

5.1 Sudden Drift Experiment Results

The first experiment tested the IDD-MLkNN's performance in detecting sudden drift in two IoT datasets; the results are shown in Table 4.

IDD-MLkNN achieved the highest accuracy, 0.93 and 0.90, in BoT-IoT and Edge-IIoTset, respectively. In both datasets, it outperformed the others with high Subset Accuracy, the strictest metric. In scenario 1, the Hamming Score for IDD-MLkNN and ML-kNN is 85, which means 85% of the labels are correctly predicted by the models across all instances. On the other hand,

in scenario 2, EaHTps, iSOUP-Tree, IDD-MLkNN have the best Hamming score of 0.67. Besides, IDD-MLkNN outperforms the others in terms of F-measure in the first scenario, which is a measure that considers the model's predictions precision and recall. For example, in the second scenario, IDD-MLkNN and EaHTps achieve the same score of 90. This indicates the high balance between producing accurate positive label predictions and identifying the most positive labels across all instances. Among the evaluated models, ML-NV was the fastest with the lowest classification time.

Table 4. Results of adapting sudden drift over two IoT datasets

Dataset / Scenario	# Evaluation Metric	ML-kNN	ML-NV	EaHTps	iSOUP-Tree	ML-AMRules	IDD-MLkNN
BoT-IoT (Sudden) - Scenario 1	Subset Accuracy	0.714	0.612	0.672	0.336	0.624	0.729
	Accuracy	0.83	0.807	0.908	0.633	0.815	0.938
	Hamming Score	0.85	0.808	0.836	0.705	0.812	0.856
	F-measure	0.875	0.86	0.936	0.743	0.866	0.956
	Evaluation Time	17301.9 s	5.4 s	478.9 s	7.17 s	67.5 s	219.5 s
Edge-IIoTset (Sudden) - Scenario 2	Subset Accuracy	0.172	0.337	0.406	0.4	0.337	0.405
	Accuracy	0.543	0.73	0.889	0.663	0.733	0.903
	Hamming Score	0.557	0.6	0.672	0.679	0.606	0.674
	F-measure	0.667	0.753	0.901	0.75	0.756	0.904
	Evaluation Time	34850.1 s	7.4 s	375.7 s	9.7 s	73.5 s	272.7 s

5.2 Gradual Drift Experiment Results

Table 5 exhibits the average results of running the gradual scenarios, where the second concepts take longer time to substitute the old ones.

Table 5. Results of adapting gradual drift over two IoT datasets

Dataset / Scenario	# Evaluation Metric	ML-kNN	ML-NV	EaHTps	iSOUP-Tree	ML-AMRules	IDD-MLkNN
BoT-IoT (Sudden) - Scenario 1	Subset Accuracy	0.512	0.571	0.606	0.271	0.583	0.681
	Accuracy	0.7	0.718	0.825	0.59	0.799	0.94
	Hamming Score	0.716	0.616	0.793	0.662	0.781	0.822
	F-measure	0.781	0.8	0.878	0.71	0.855	0.958
	Evaluation Time	17484.9 s	4.7 s	478.7 s	7.5 s	71.9 s	219.1 s
Edge-IIoTset (Sudden)-Scenario 2	Subset Accuracy	0.169	0.33	0.399	0.382	0.331	0.393
	Accuracy	0.539	0.7	0.86	0.651	0.739	0.891
	Hamming Score	0.552	0.587	0.664	0.67	0.609	0.665
	F-measure	0.663	0.724	0.88	0.741	0.762	0.893
	Evaluation Time	32021.6 s	6.9 s	390.8 s	10.9 s	65.9 s	279.7 s

As shown in Table 5, IDD-MLkNN has the highest values in Subset Accuracy, Accuracy, Hamming Score, and F-measure in scenario 3. For scenario 4, the model is superior in accuracy

and F-measure. In addition, IDD-MLkNN shares similar Subset Accuracy with EaHTps and a similar Hamming Score with iSOUP-Tree and EaHTps.

5.3 Statistical Analysis

To further examine the performance of IDD-MLkNN, the Friedman test with the Bonferroni-Dunn posthoc test introduced in [4] are used to find the significance of the results statistically. The Friedman test ranks the algorithms based on their performance by giving the best rank of one and the second best rank of two, and so on, for each dataset separately. The average performance over four datasets and their ranking results is presented in Table 6.

Table 6. The average performance results for all algorithms across all four datasets and their ranking

# Evaluation metric	ML-kNN	ML-NV	EaHTps	iSOUP-Tree	ML-AMRules	IDD-MLkNN
Subset Accuracy	0.39175	0.4625	0.52075	0.34725	0.46875	0.552
Accuracy	0.653	0.738	0.870	0.634	0.771	0.918
Hamming Score	0.668	0.652	0.741	0.679	0.702	0.754
F-measure	0.746	0.784	0.898	0.736	0.809	0.9278
Evaluation Time	25414.6	6.1	431.025	8.8175	69.7	247.75
# Evaluation Metric	ML-kNN	ML-NV	EaHTps	iSOUP-Tree	ML-AMRules	IDD-MLkNN
Subset Accuracy	5	4	2	6	3	1
Accuracy	5	4	2	6	3	1
Hamming Score	5	6	2	4	3	1
F-measure	5	4	2	6	3	1
Evaluation Time	6	1	5	2	3	4
Average Rank	5.2	3.8	2.6	4.8	3	1.6

As presented in Table 6, IDD-MLkNN achieved the best ranking (1) in terms of Subset Accuracy, Accuracy, Hamming Score, and F-measures, followed by EaHTps. Regarding Evaluation Time, ML-NV achieved the best ranking, followed by iSOUP-Tree. Figure 3 illustrates the ranking results for each method's average rankings of the evaluation measure across all the IoT datasets. The height of each bar indicates the method's average ranking for that particular evaluation measure for all the datasets, shorter bar indicates better performance.

Moreover, the Bonferroni-Dunn test with critical difference = 3.07 at $\alpha = 0.1$ shows that IDD-MLkNN significantly differs from ML-kNN and ISOUPTree. At the same time, the other methods do not show significant differences compared to IDD-MLkNN. Overall, the results show that the IDD-MLkNN can maintain high accuracy even in the presence of sudden and gradual drift in IoT datasets. The performance of learning models encountering concept drift and class imbalance deteriorates over time unless addressed. The IDD-MLkNN model proactively removes instances that introduce errors. This removal is done only when the model notices significant deviations between two main variables; these variables monitor the similarity between the new instance and its nearest neighbors, making it responsive to abrupt drift. In addition, the adopted long sliding windows allow for gradual drift adaptation. ML-NV achieves the shortest classification time due to the method's simplicity, as it does not diagnose for drift. On the contrary, in IDD-MLkNN, a diagnosis and update is made after each instance's arrival which increases the time complexity.

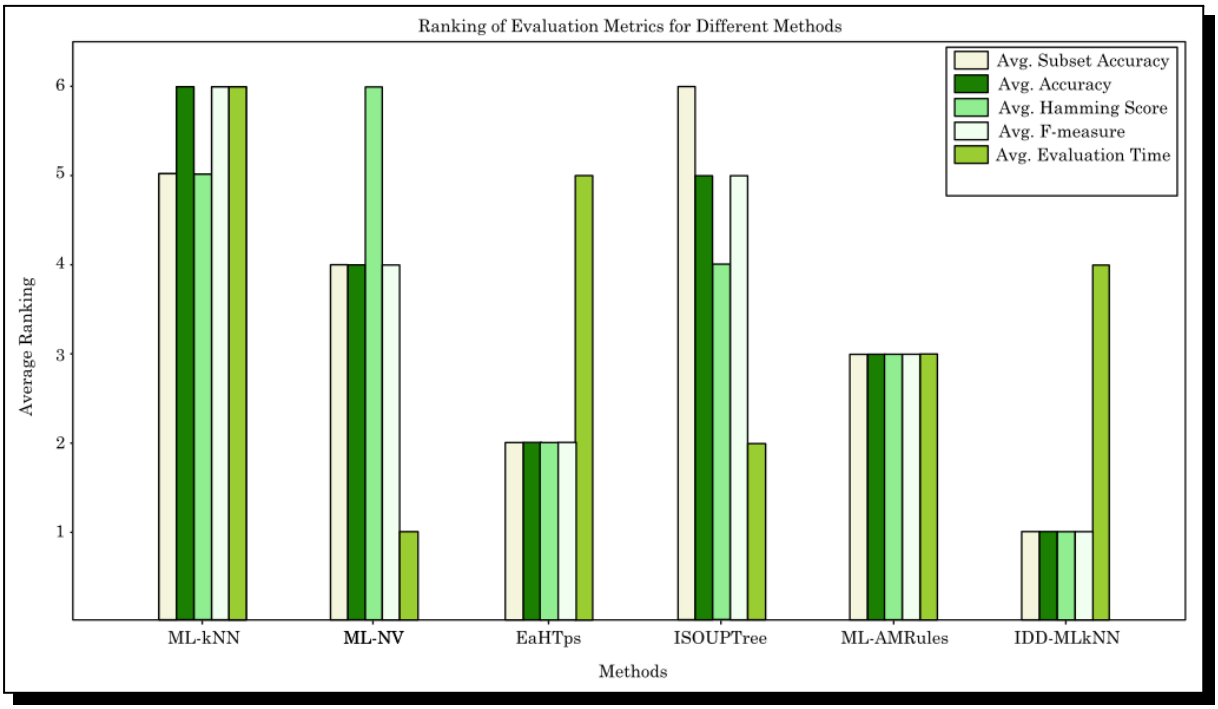


Figure 3. Average rankings of MLC methods across different evaluation measures

6. Conclusion and Future Work

In this article, multi-label classification is utilized for IoT data streams. The proposed IDD-MLkNN model implicitly handles concept drift and class imbalance. It involves three main steps, which are classification, diagnose the stream instances in the window, and update the window when needed. Two IoT datasets, Edge-IIoTset and Bot-IoT, were used to simulate two types of drift: sudden and gradual. The IDD-MLkNN model outperformed other methods in the comparison analysis in identifying the label set related to network traffic attacks in four measures, including Subset Accuracy, Accuracy, Hamming Score, and F-measure. Yet, it failed to get the best ranking in classification time. The practical implications of implementing the proposed model for monitoring IoT network traffic are to reduce the time and cost that may result from re-analyzing the attack information. As future work, IDD-MLkNN will be extended to work with other types of data streams form different domains. It also suggested using a dynamic threshold instead of static value, which is automatically adjusted according to the type of data stream or the window size.

Competing Interests

The authors declare that they have no competing interests.

Authors' Contributions

All the authors contributed significantly in writing this article. The authors read and approved the final manuscript.

References

[1] G. Andresini, F. Pendlebury, F. Pierazzi, C. Loglisci, A. Appice and L. Cavallaro, INSOMNIA: Towards concept-drift robustness in network intrusion detection, in: *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security* (AISec'21), Association for Computing Machinery, 2021, pp. 111 – 122, New York, USA, (2021), DOI: 10.1145/3474369.3486864.

[2] A. Bifet, G. Holmes, R. Kirkby and B. Pfahringer, MOA: Massive Online Analysis, *The Journal of Machine Learning Research*, **11** (2010), 1601 – 1604, URL: http://dl.acm.org/citation.cfm?id=1859890.1859903.

[3] A. Buyukcakir, H. Bonab and F. Can, A novel online stacked ensemble for multi-label stream classification, in: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (CIKM'18), 2018, 1063 – 1072, ACM, (2018), DOI: 10.1145/3269206.3271774.

[4] J. Demšar, Statistical comparisons of classifiers over multiple data sets, *The Journal of Machine Learning Research* **7** (2006), 1 – 30, URL: https://dl.acm.org/doi/10.5555/1248547.1248548.

[5] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras and H. Janicke, Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning, *IEEE Access* **10** (2022), 40281 – 40306, DOI: 10.1109/ACCESS.2022.3165809.

[6] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy and A. Bouchachia, A survey on concept drift adaptation, *ACM Computing Surveys* **46**(4) (2014), article number 44, DOI: 10.1145/2523813.

[7] E. Hallaji, R. Razavi-Far and M. Saif, Expanding analytical capabilities in intrusion detection through ensemble-based multi-label classification, *Computers & Security* **139** (2024), 103730, DOI: 10.1016/j.cose.2024.103730.

[8] M. Jethanandani, A. Sharma, T. Perumal and J.-R. Chang, Multi-label classification based ensemble learning for human activity recognition in smart home, *Internet of Things* **12** (2020), 100324, DOI: 10.1016/j.iot.2020.100324.

[9] X. Kong and P. Yu, An ensemble-based approach to fast classification of multi-label data streams, in: *7th International Conference on Collaborative Computing: Networking, Applications and Worksharing* (CollaborateCom), Orlando, 2012, IEEE, 10 pages (2012), DOI: 10.4108/icst.collaboratecom.2011.247086.

[10] N. Koroniotis, N. Moustafa, E. Sitnikova and B. Turnbull, Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset, *Future Generation Computer Systems* **100** (2019), 779 – 796, DOI: 10.1016/j.future.2019.05.041.

[11] A. Kuppa and N.-A. Le-Khac, Learn to adapt: Robust drift detection in security domain, *Computers and Electrical Engineering* **102** (2022), 108239, DOI: 10.1016/j.compeleceng.2022.108239.

[12] D. C. Mulimani, S. G. Totad and P. R. Patil, Concept drift adaptation in intrusion detection systems using ensemble learning, *International Journal of Natural Computing Research* **10**(4) (2021), 22 pages, DOI: 10.4018/IJNCR.2021100101.

[13] T. T. Nguyen, T. T. T. Nguyen, A. V. Luong, Q. V. H. Nguyen, A. W.-C. Liew and B. Stantic, Multi-label classification via label correlation and first order feature dependance in a data stream, *Pattern Recognition* **90** (2019), 35 – 51, DOI: 10.1016/j.patcog.2019.01.007.

[14] A. Osojnik, P. Panov and S. Džeroski, Multi-label classification via multi-target regression on data streams, *Machine Learning* **106** (2017), 745 – 770, DOI: 10.1007/s10994-016-5613-5.

[15] D. D. Pezze, D. Deronjic, C. Masiero, D. Tosato, A. Beghi and G. A. Susto, A multi-label continual learning framework to scale deep learning approaches for packaging equipment monitoring, *Engineering Applications of Artificial Intelligence* **124** (2023), 106610, DOI: 10.1016/j.engappai.2023.106610.

[16] W. Qu, Y. Zhang, J. Zhu and Q. Qiu, Mining multi-label concept-drifting data streams using dynamic classifier ensemble, in: *Advances in Machine Learning* (ACML 2009), Z. H. Zhou and T. Washio (editors), Lecture Notes in Computer Science series, Vol. 5828, Springer, Berlin — Heidelberg, 308 – 321, (2009), DOI: 10.1007/978-3-642-05224-8_24.

[17] J. Read, A. Bifet, G. Holmes and B. Pfahringer, Scalable and efficient multi-label classification for evolving data streams, *Machine Learning* **88** (2012), 243 – 272, DOI: 10.1007/s10994-012-5279-6.

[18] J. Read, B. Pfahringer, G. Holmes and E. Frank, Classifier chains for multi-label classification, in: *Machine Learning and Knowledge Discovery in Databases*, W. Buntine, M. Grobelnik, D. Mladenić and J. Shawe-Taylor (editors), ECML PKDD 2009, Lecture Notes in Computer Science series, Vol. 5782, Springer, Berlin — Heidelberg, pp. 254 – 269, (2009), DOI: 10.1007/978-3-642-04174-7_17.

[19] M. Roseberry, B. Krawczyk and A. Cano, Multi-label punitive kNN with self-adjusting memory for drifting data streams, *ACM Transactions on Knowledge Discovery from Data* **13**(6) (2019), article number 60, DOI: 10.1145/3363573.

[20] R. Sousa and J. Gama, Online multi-label classification with adaptive model rules, in: *Advances in Artificial Intelligence* (CAEPIA 2016), Lecture Notes in Computer Science series, Vol. 9868, Springer, Cham., pp. 58 – 67, (2016), DOI: 10.1007/978-3-319-44636-3_6.

[21] X. Wang, P. Kuntz, F. Meyer and V. Lemaire, Multi-label kNN classifier with online dual memory on data stream, in: *2021 International Conference on Data Mining Workshops* (ICDMW), Auckland, New Zealand, 2021, 405 – 413, IEEE, (2021), DOI: 10.1109/ICDMW53433.2021.00056.

[22] Q. Wei, Y. Zhang, J. Zhu and W. Yong, Mining multi-label concept-drifting data streams using ensemble classifiers, in: *Proceedings of the Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, Tianjin, China, 2009, pp. 275 – 279, (2009), DOI: 10.1109/FSKD.2009.315.

[23] S. Xioufis, M. Spiliopoulou, G. Tsoumakas and I. Vlahavas, Dealing with concept drift and class imbalance in multi-label stream classification, in: *Proceedings of the International Joint Conference on Artificial Intelligence* (IJCAI2011), 1583 – 1588, (2011), DOI: 10.5591/978-1-57735-516-8/IJCAI11-266.

[24] R. Xu, Y. Cheng, Z. Liu, Y. Xie and Y. Yang, Improved long short-term memory based anomaly detection with concept drift adaptive method for supporting IoT services, *Future Generation Computer Systems* **112** (2020), 228 – 242, DOI: 10.1016/j.future.2020.05.035.

[25] M.-L. Zhang and Z.-H. Zhou, A review on multi-label learning algorithms, *IEEE Transactions on Knowledge and Data Engineering* **26**(8) (2014), 1819 – 1837, DOI: 10.1109/TKDE.2013.39.

[26] M.-L. Zhang and Z.-H. Zhou, ML-KNN: A lazy learning approach to multi-label learning, *Pattern Recognition* **40**(7) (2007), 2038 – 2048, DOI: 10.1016/j.patcog.2006.12.019.