2017

# Hoeffding Tree Algorithms for Anomaly Detection in Streaming Datasets: A Survey

Asmah Muallem

Sachin Shetty
*Old Dominion University*, sshetty@odu.edu

Jan W. Pan

Juan Zhao

Biswajit Biswal



## Original Publication Citation

Muallem, A., Shetty, S., Pan, J. W., Zhao, J., & Biswal, B. (2017). Hoeffding tree algorithms for anomaly detection in streaming datasets: A survey. *Journal of Information Security, 8*(4), 339-361. doi:10.4236/jis.2017.84022

Journal of Information Security, 2017, 8, 339-361
http://www.scirp.org/journal/jis
ISSN Online: 2153-1242
ISSN Print: 2153-1234

# Hoeffding Tree Algorithms for Anomaly Detection in Streaming Datasets: A Survey

**Asmah Muallem[1], Sachin Shetty[2], Jan Wei Pan[3], Juan Zhao[4], Biswajit Biswal[5]**

[1]Department of Electrical and Computer Engineering, Tennessee State University, Nashville, TN, USA
[2]Virginia Modeling, Analysis and Simulation Center, Old Dominion University, Norfolk, VA, USA
[3]R&D Engineering Department, Roadstar.AI., Mountain View, CA, USA
[4]Department of Electrical and Computer Engineering, Tennessee State University, Nashville, TN, USA
[5]Department of Mathematics and Computer Science, South Carolina State University, Orangeburg, SC, USA
Email: asma.muallem@gmail.com, sshetty@odu.edu, janweipan1@gmail.com, juanzhaowendy@gmail.com, biswal.biswajit@gmail.com

**How to cite this paper:** Muallem, A., Shetty, S., Pan, J.W., Zhao, J. and Biswal, B. (2017) Hoeffding Tree Algorithms for Anomaly Detection in Streaming Datasets: A Survey. *Journal of Information Security*, **8**, 339-361.
https://doi.org/10.4236/jis.2017.84022

**Received:** September 6, 2017
**Accepted:** October 22, 2017
**Published:** October 25, 2017





## Abstract

This survey aims to deliver an extensive and well-constructed overview of using machine learning for the problem of detecting anomalies in streaming datasets. The objective is to provide the effectiveness of using Hoeffding Trees as a machine learning algorithm solution for the problem of detecting anomalies in streaming cyber datasets. In this survey we categorize the existing research works of Hoeffding Trees which can be feasible for this type of study into the following: surveying distributed Hoeffding Trees, surveying ensembles of Hoeffding Trees and surveying existing techniques using Hoeffding Trees for anomaly detection. These categories are referred to as compositions within this paper and were selected based on their relation to streaming data and the flexibility of their techniques for use within different domains of streaming data. We discuss the relevance of how combining the techniques of the proposed research works within these compositions can be used to address the anomaly detection problem in streaming cyber datasets. The goal is to show how a combination of techniques from different compositions can solve a prominent problem, anomaly detection.

## Keywords

Hoeffding Trees, Distributed, Ensembles, Anomaly Detection, Machine Learning, Spark

## 1. Introduction

A wide variety of application domains use streaming data. These domains in-

clude intrusion detection, network sensors, bioinformatics, weather prediction, web and mobile applications, ecommerce purchases, social networks, and many more.

In most scenarios where new, dynamic data is continuously generated, streaming data is beneficial. Data collection of information is the first step in streaming data and it can evolve into a more sophisticated real-time processing. Initially these applications may perform simple data analysis with simple actions as a response in which may ultimately evolve into more sophisticated types of data analysis, such as applying machine learning algorithms to extract precise information from the data. There are several challenges on data mining algorithm design posed by data streams [1]. One is the limited use of resources [1]. Most big data, in which data are produced continuously, can be recorded as data streams [2].

Many machine learning algorithms have been developed for sophisticated data analysis on streaming data. These include classification, regression, and clustering algorithms. The model in the data streams is that it must address the problem of three features of big data: large volume, large velocity, and large variety. A fundamental model in data streams is the necessary of dealing with data whose nature or distribution changes over time [1]. Strategies for detecting and quantifying change, forgetting stale experiments, and model revision is required in dealing with time-changing data [1].

Decision trees are a type of classifier algorithms [3] [4]. A decision tree is learned top-down by recursively replacing leaves by test nodes, starting at the root [1]. All available attributes are compared and choosing the best one according to some heuristic measure is how the attribute at a node is tested [1]. Classical decision tree learners are severely limited in the number of examples they can learn from, since they assume that all training examples can be stored simultaneously in memory [1]. Hoeffding Trees, an incremental, anytime decision tree induction algorithm capable of learning from massive data streams, was developed by Domingos and Hulten [1] [5]. The fact that a small sample can often be enough to choose an optimal splitting attribute is the theory of Hoeffding Trees [1]. The Hoeffding bound mathematically supports this idea by quantifying the number of observations or examples needed to estimate some statistics within a prescribed decision or goodness of an attribute [1]. The Hoeffding Trees have sound guarantees of performance, a theoretically interesting feature not shared by other incremental decision tree learners. **Figure 1** provides the Hoeffding Tree Induction Algorithm referenced from [6].

Anomaly detection for streaming data is an important topic of research due to its nature of detecting vital information. This vital information can include intrusion and other failure information. There is extensive work on anomaly detection techniques [7] [8] [9] for streaming data. These techniques look into fault detection and intrusion detection by exploring methods for identifying anomalies based on the scalability and generality of the data streams, they do not

**Algorithm 1** Hoeffding Tree Induction Algorithm.

Let $HT$ be a tree with a single leaf (the root)
**for all** training examples **do**
  Sort example into leaf $l$ using $HT$
  Update sufficient statistics in $l$
  Increment $n_l$, the number of examples seen at $l$
  **if** $n_l \ mod \ n_{min} = 0$ **and** examples seen at $l$ not all of same class **then**
    Compute $\overline{G}_l(X_i)$ for each attribute
    Let $X_a$ be attribute with highest $\overline{G}_l$
    Let $X_b$ be attribute with second-highest $\overline{G}_l$
    Compute Hoeffding bound $\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n_l}}$
    **if** $X_a \neq X_\emptyset$ **and** $(\overline{G}_l(X_a) - \overline{G}_l(X_b) > \epsilon$ **or** $\epsilon < \tau)$ **then**
      Replace $l$ with an internal node that splits on $X_a$
      **for all** branches of the split **do**
        Add a new leaf with initialized sufficient statistics
      **end for**
    **end if**
  **end if**
**end for**

**Figure 1.** Algorithm 1 [6]: hoeffding tree induction algorithm.

take the transformations in the underlying organization of the data into consideration. Concept drift is known as the underlying distribution of the data changing swiftly over time within streaming data [10] [11]. The occurrence of concept drift is when the concept about which data are being collected shifts from time to time after a minimum stability period. When exploring various types of anomaly detection such as intrusion or fault detection, one must explore foundations of cyber-attacks. Abrupt, incremental, gradual or recurring changes can be created by cyber-attacks [12]. According to data mining, the target information that a model is trying to predict are the concepts [12]. The change of the underlying concept over time is known as concept change. [12]. A relatively slow change of concept is the representation of concept drift, and an abrupt change in concept is the representation of concept shift [12]. The general focus of machine learning is the representation of one type of concept drift.

There are many proposed methods of Hoeffding Trees for data streams. Most of these are built on characteristics for dealing with distribution in data streams such as concept drift. These models can be a single Hoeffding Tree, decision trees based on the Hoeffding bound, window-based Hoeffding Trees, weight-based Hoeffding Trees, distribution Hoeffding Trees, and/or an ensemble of Hoeffding Trees. Each of these models addresses various attributes of concept drift in data streams or identifying the best approach to classify data in data streams without concept drift. In many instances these models are combined with other techniques to provide improved accuracy, performance, or drift detection.

An Anomaly Detection System can also be known as an Intrusion Detection System, in which intursions are identified by classifying activities as either normal or anomalous and leading to a training phase to be implemented to recognize "new" attacks [13]. In the case of using machine learning for anomaly detection, classification algorithms can be used to determine if uncommon pat-

terns exist within the data. As stated earlier, the Hoeffding Tree is the best classifier due to its extensive capabilities in data stream classification.

This survey aims to deliver an extensive and well-constructed overview of the existing proposed research related to Hoeffding Trees in various domains. The objective of this survey paper is based on providing an overall comprehension of the use of Hoeffding Trees within streaming datasets and how the techniques proposed for Hoeffding Trees within various domains can be applicable to solve the problem of detecting anomalies in streaming cyber datasets through machine learning. We categorize these techniques according to compositions and for each composition we survey an existing proposed research work using Hoeffding Trees while providing the techniques used for this research work. We also provide advantages and disadvantages of each research works. In Section 6, we provide a comparison chart listing the evaluation metrics produced from each proposed research work based on accuracy, Kappa statistic, time, and/or memory as well as identifying the corresponding dataset which produced the evaluation metric for the given proposed research work algorithm.

In Figure 2, we present the three compositions of Hoeffding Tree algorithms for streaming datasets within various application domains including anomaly detection. These application domains include machine learning classification accuracy and performance in streaming data, concept drift detection in streaming data, machine learning distributed processing to reduce execution time and

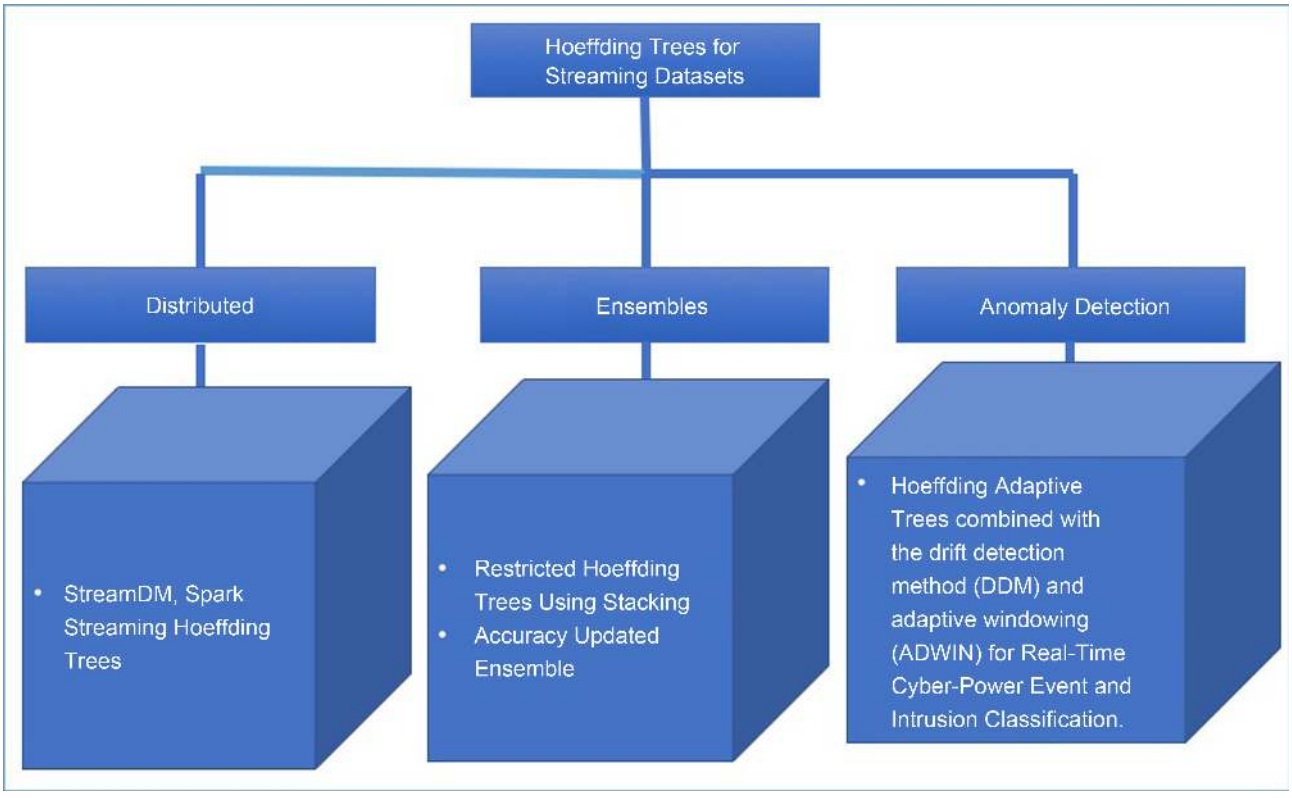


Figure 2. Overview of existing hoeffding tree algorithms for streaming datasets within various application domains including anomaly detection.

memory consumption as well as speed up processing complexity in streaming data, and anomaly detection in streaming data.

The organization of the remaining of this survey paper is as follows. Section 2 describes the first composition: surveying different application domains of distributed Hoeffding algorithms in streaming datasets. We describe the second problem composition: surveying different application domains of ensembles of Hoeffding Tree algorithms in streaming datasets, in Section 3. We describe the third problem composition in Section 4, surveying existing proposed research work of Hoeffding Tree algorithms for anomaly detection. In Section 5, we provide a literature review on existing surveys about ensembles of Hoeffding Tree algorithms. Section 6 discusses how the different Hoeffding Tree compositions presented in this survey can be combined to be applicable in the context of anomaly detection in streaming cyber datasets. Conclusions and suggestions of this research survey are presented in Section 7.

## 2. Surveying Distributed Hoeffding Trees

In this section, we survey a proposed research work of distributed Hoeffding Trees based on Spark Streaming and provide the advantages and disadvantages of this research work as well as metrics from our experimental evaluation performed on the proposed research work.

### 2.1. Spark Streaming Hoeffding Trees

Bifet *et al*. in [14] introduce an application called StreamDM, which they indicate is a new open-source data mining and machine learning library based on Spark Streaming. The authors define Spark Streaming as well as its advantages and disadvantages of and the reasoning behind their choice of this platform. The authors also note the purpose of their design in which their concentration was on implementing an extensible library containing advanced Spark Streaming mining algorithms which can be easily available for use by developers, researchers, and others. The authors note that StreamDM benefits from its design above Spark Streaming due to its existence within the Hadoop open-source environment. The authors also indicate they use Scala as the programming language for the implementation and due its benefits as well as its compatibility with other platforms. The authors inform their goal is to provide StreamDM as a machine learning library which can be accessible for researchers and developers to easily design additional algorithms above the implemented open-source machine learning library within StreamDM. StreamDM currently contains four classification algorithms: Multinomial Naïve Bayes, SGD Learner and Perceptron Classifier using the Stochastic Gradient Descent optimizer for learning various linear models, Hoeffding Decision Trees, and Bagging.

#### 2.1.1. Evaluating StreamDM

In this section, we discuss our evaluation of StreamDM by creating some experiments to compare it to MOA and WEKA [15]. We also provide evaluation me-

trics based on our results. Our comparison is initially based on how the StreamDM Spark Hoeffding Tree compares to its counterparts, the Hoeffding Tree in MOA and Hoeffding Tree in WEKA. Although, we have generated an artificial attack dataset, for this experiment we use the NSL-KDD dataset, since there is a need to further pre-process our generated artificial attack dataset for such as performing feature extraction and feature engineering on the data. For future use, we would like to extend our experiments using the generated artificial attack dataset for a diverse comparison evaluation. The NSL-KDD dataset is a data set which was used for the Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with the Fifth International Conference on Knowledge Discovery and Data Mining, in which the competition revolved around building an intrusion detector which was based on a predictive model capable of identifying intrusions or attacks versus normal connections simulated in a military network environment. The attacks within this dataset are listed in Table 1.

The first experiment we run is using MOA. In this experiment, we input the NSL-KDD dataset and during execution, we monitor the Accuracy (Correctly Classified Instances), Incorrectly Classified Instances, Number of Instances and

**Table 1.** List of attack types within NSL-KDD dataset.

| |
|---|
| back dos |
| buffer_overflow u2r |
| ftp_write r2l |
| guess_passwd r2l |
| imap r2l |
| ipsweep probe |
| land dos |
| loadmodule u2r |
| multihop r2l |
| neptune dos |
| nmap probe |
| perl u2r |
| phf r2l |
| pod dos |
| portsweep probe |
| rootkit u2r |
| satan probe |
| smurf dos |
| spy r2l |
| teardrop dos |
| warezclient r2l |
| warezmaster r2l |

Kappa statistic. The second experiment uses WEKA with the NSL-KDD dataset as input and during execution, we monitor the same items monitored during execution within MOA along with other items available for monitoring within WEKA such as the mean absolute error, root relative squared error, root mean error, True Positive Rate, False Positive Rate, relative absolute error and Precision. WEKA also provides the 10-fold cross validation used to build the model and the resulting confusion matrix. The Hoeffding Tree built during the model is also displayed to help with examining the weights of the features during classification. For the experiment using MOA, we select the default Evaluate Prequential task which trains and tests at the same time versus the cross-fold validation available in WEKA. At the end of the execution MOA does not provide the resulting confusion matrix and the construction of Hoeffding Tree flowchart visualization with the probabilities for each feature which is provided by WEKA.

The third experiment is implemented using StreamDM. The Hoeffding Trees within the StreamDM tool required adjustments to allow for the capability of working with various types of datasets, such as the NSL-KDD dataset. This was noticed after running the first experiment with StreamDM as the execution resulted in errors due to a minor failure to handle null values when coming across null values as input from null features. The necessary changes were made to the code to handle these errors due to the flexibility of StreamDM's machine learning library open-source availability. Following this, we ran another set of experiments using the Evaluate Prequential task and monitored the results. During this execution, the only items reported were the construction of the Hoeffding Tree flowchart visualization with the probabilities of each feature, as in WEKA, and the tree depth, nodes, and other parameters of the tree being built. Statistics such as Accuracy, Incorrectly Classified Instances, Number of Instances and Kappa statistic were not yet included within StreamDM. The necessary changes were also made to the code to add the calculations for these statistics to StreamDM based on the calculations to produce these statistics given within the open-source code machine learning library of MOA. As a result, these statistics began reporting within StreamDM and we could further evaluate StreamDM in comparison to MOA and WEKA.

#### 2.1.2. Preliminary Results

Figure 3 plots accuracy, Kappa statistic, and Incorrectly Classified Instances using the Hoeffding Tree within each MOA, WEKA, and StreamDM. The Accuracy is based on the number of Correctly Classified Instances.

We can see from these results, the Accuracy is very close if not the same for the Hoeffding Tree in WEKA and MOA, WEKA is 99% and MOA is 100%. The Hoeffding Tree in StreamDM has an Accuracy close to its former counterparts of between 92% and 95%, 93%, but still needs some improvement. The Incorrectly Classified Instances is also higher within the StreamDM Spark Hoeffding Tree in comparison to WEKA which is about 8% as opposed to MOA and WEKA which is 0%. We found the Kappa statistic in MOA and WEKA to be

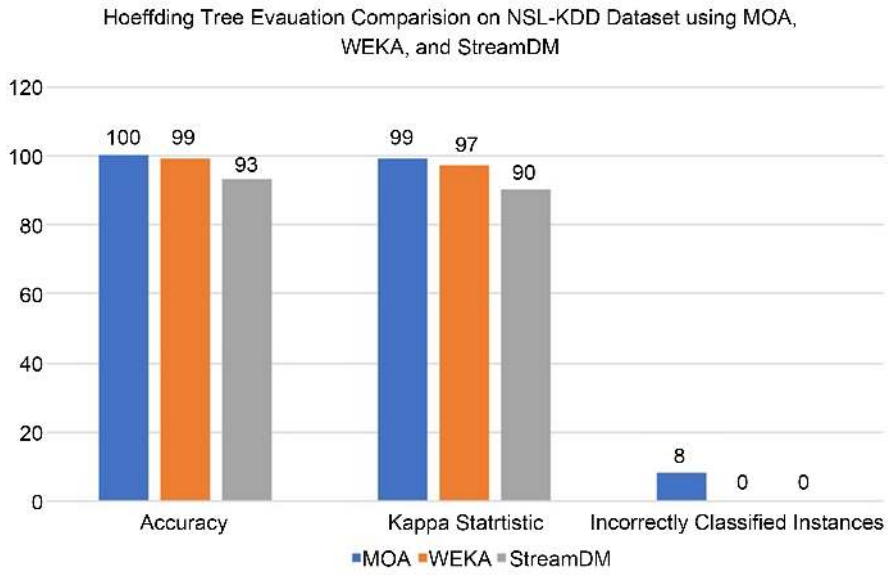


**Figure 3.** Comparison evaluation of a hoeffding tree learner on NSL-KDD dataset using MOA, WEKA, and StreamDM.

fairly close as 99% for MOA and 97% for WEKA, while the Kappa statistic in StreamDM was slightly behind as 90%. Based on these experiments we can see a need for enhancements to the Hoeffding Trees in StreamDM.

**Advantages of this research:** To our knowledge, StreamDM is the first implementation of the Hoeffding Tree algorithm built on top of Spark using Spark Streaming. StreamDM offers flexibility for extending the application to add new algorithms through the availability of its open-source machine learning library as well extending its learner. Our thoughts of StreamDM are similar to the authors' thoughts in [14], it is a great tool for streaming machine learning models as it not only provides Hoeffding Trees but also SGD Learner, Bagging, and Multinomial Naïve Bayes. The developers plan to extend the tool to include additional machine learning algorithms which include: the classification Random Forests, Hoeffding Regression Trees, and much more. Random Forest Trees are available within the Spark Machine Learning Library, MLib, but do not utilize Spark Streaming. In our opinion, it would be beneficial to have a tool which provides Spark Streaming capabilities for the machine learning algorithms within MLib. In reference to our survey, we believe StreamDM is a great tool which can be used as a distributed IDE or platform for developing a model to address anomaly detection within streaming cyber datasets. The detail of this hypothesis is described in Section 6. Although, the evaluation metrics revealed the StreamDM statistics were not as close to the statistics reported from MOA and WEKA, we feel a minor adjustment can be made to fix these calculations.

**Disadvantages of this research:** To our knowledge, no experiments and/or evaluations, apart from our experiment and evaluation, have been implemented on this platform thus far. Therefore, no prior credible existing proposed research work and results are available to thoroughly compare our eval-

uations of the system.

## 3. Surveying Ensembles of Hoeffding Trees

For streaming processes, ensemble classifiers possess an adequate number of advantages in comparison to single classifier designs [16]. This is because ensembles can be easily scaled and parallelized [16]. A key factor of ensembles is their ability to adaptively change quickly through the process of pruning parts of the ensemble which are performing poorly, as a result generating more accurate concept descriptions [16]. In this section, we evaluate several types of proposed research works on the ensembles of Hoeffding Trees and provide the advantages and disadvantages of each proposed work.

### 3.1. Ensembles of Restricted Hoeffding Trees Using Stacking

Bifet *et al*. in [17], present an algorithm or classification model based on the ensemble of restricted decision trees, *i.e*. Hoeffding Trees, using Stacking. The authors indicate they use a stacking approach which is detailed on how each tree is built within this algorithm using attributes, log-odds of probabilities from predicted classes, applying sigmoid perceptrons, using perceptron classifiers and more. The authors also describe how their stacking approach differs from boosting and elaborate on the method they use for forming their ensemble classifier [18] The authors also note the significance of the use of Hoeffding Trees within their work since they are working with a data stream scenario.

The authors thoroughly elaborate on the method they present in their proposed research work and why they use ADWIN, which they note is due to its theoretical guarantees on false positives within the context of change. The authors also elaborate on the benefits of ADWIN as well as how they use ADWIN for dealing with evolving data streams, replacing poorly performing ensemble members when an accuracy for one of the Hoeffding Trees has dropped significantly, and resetting the learning rate. The authors describe a detailed process of the ADWIN change detector to replace poorly performing ensemble members due to their decline in accuracy as well as the learning rate reset process.

The authors in [17] describe their need for using Naïve Bayes Hoeffding Trees and propose the use of Naïve Bayes models at the leaves of the Hoeffding Trees, called Naïve Bayes Hoeffding Trees (hnbt), due to their presumption that it can increase the predictive accuracy of normal Hoeffding Trees since Normal Hoeffding Trees only perform prediction through the selection of the majority of the class of each leaf. The authors also cite work by Holmes *et al*. [19] about findings regarding the use of Naïve Bayes models at the leaves of the Hoeffding Trees and its advantages as well as disadvantages. The authors reiterate their thoughts on the excellent predictive performance for evolving data streams through the use of bagging using ADWIN [16] based on the online bagging method of [20] with the ADWIN algorithm for detecting accuracy changes for each ensemble members. The authors state they use the Hoeffding Naïve Trees as the

base learner for their new ensemble learner comparing it to their proposed method of ADWIN bagging.

The authors perform various experiments to investigate the resources and performance of the proposed research work. These experiments include comparing bagging to their new method of using stacking with restricted Hoeffding Trees consisting of 1, 2, 3, and 4 attributes consequently using an Interleaved Test-Then-Train or Prequential evaluation. The authors also note they performed a separate experiment for the disabling of the ADWIN-based change detection for resetting the learning rate and found a decrease in average accuracy in which they note these results reinforce the benefits of using ADWIN-based change detection for resetting the learning rate. The authors note from the experimental results it can be seen that the use of more trees with bagging using ADWIN is not any more accurate than the use of less trees with bagging using ADWIN, in which they believe the positive performance of their proposed algorithm is not related to the use of many trees. The authors believe the vital factor in the improvement of accuracy obtained is their methodology of using attribute subsets and perceptron weighting. The authors note during the experimental evaluation, they found an increase in ensemble diversity produced from their new stacking method in which they indicate is due the tight clusteration of the ADWIN bagging ensembles and increased discrepancy amdist classifiers within the ensemble produced by the stacking method. To expose this divergence, the authors use the Kappa statistic method $k$ [21] and describe its use within their newly stacking based classifier to increase ensemble diversity. The authors also introduce a new strategy to replace ensemble classifiers.

We have organized some of the evaluation metrics produced from the authors' experimental evaluation based on the results illustrated within [17], these evaluation metrics are included in a comparison chart provided within **Table 1** in Section 6.

**Advantages of this research:** The authors in [17] present proposed research work about combining restricted Hoeffding Trees using stacking. Their methodology is based on using log-odds probability for the classes along with ADWIN change detectors to detect changes and reset the learning rate. The ADWIN change detector also detects when the accuracy has dropped significantly for an ensemble member as removing the poorest performing ensemble member. The authors give discrete details on the development and implementation of the proposed algorithm as well as their findings during experimental evaluation such as the ensemble diversity, and a strategy for replacing classifiers. The experiments evaluated were based on accuracy, time, and memory. The authors inform they performed the experiments on real-world data sets and observed excellent classification accuracy. The authors conclude on the importance of stacking to learn how to combine predictions of tree classifiers in a suitable manner. In reference to this survey, it can be beneficial to construct an ensemble of restricted Hoeffding Trees using stacking to detect anomalies within stream-

ing cyber data sets. Since an ensemble of classifiers is stronger than a single classifier, the method and techniques presented in this proposed research work can be applied to effectively address problems in the domain of anomaly detection and cyber data streams. This idea is also explained more in detail in Section 6.

**Disadvantages of this research:** The authors note some deficiencies with initial experiments of pruning the ensembles and discarding the least important ensemble member, in which their goal is to address this with smarter pruning methods and techniques to increase accuracy classification.

### 3.2. The Accuracy Updated Ensemble (AUE2)

Brezinski *et al.* [22] propose a new data stream classifier focusing on reacting uniformly well to various types of drift, called the Accuracy Updated Ensemble, or AUE2. The authors inform the theory of their proposed research is based on the basis that one of the most significant challenges in data streams is concept drift and that several classification algorithms have been proposed for dealing with concept drift but most of these focus on a single type of drift. In the authors definition, concept drift is the unforeseen transformations of the underlying distribution of the data stream. The authors indicate they combine Hoeffding Trees based on accuracy-based and block-based weighting mechanisms within their AUE2 algorithm due to the incremental nature of Hoeffding Trees.

The authors give details about the advantages and disadvantages of previous block-based ensembles and incremental ensembles such as the reactions of block-based ensembles to sudden types of drifts. The authors describe in some cases these block-based ensembles may react slowly to sudden types of drifts but in contrast to incremental ensembles may react quicker to sudden drifts without the benefits of some mechanisms as well as the computational costs of incremental ensembles. As a result of these observations, the authors inform they put forward a proposal based on providing a proposed method which is cost-effective and beneficial in combining characteristic features from both approaches, block-based methods and online incremental ensembles. The authors refer to their algorithm as a hybrid algorithm which should react much better to different types of drifts as opposed to related adaptive ensembles. The authors indicate their goal is to design a block-based algorithm with learning component classifiers that use weighting predictions as well as elements known from online methods which can incrementally update to provide the improvement of the ensemble to react to various types of concept drift. The authors elaborate on how this new AUE2 algorithm differs and enhances the preliminary version of the algorithm, Accuracy Updated Ensmeble (AUE1). The authors also describe in detail the construction of their algorithm such as the methods and techniques used within the algorithm including its weighting mechanisms. The Accuracy Updated Ensemble Algorithm (AUE2), referenced from [22] is shown **Figure 4**.

The authors report they perform experiments to compare their introduced algorithm to what they refer to as 11 state-of-the-art streaming classifiers or

**Algorithm 2** Accuracy Updated Ensemble (AUE2) Algorithm.

**Input:** $\mathcal{S}$: data stream of examples partitioned into chunks, $k$: number of ensemble members, $m$: memory limit

**Output:** $\mathcal{E}$: ensemble of $k$ weighted incremental classifiers

1: $\mathcal{E} \leftarrow \emptyset$;
2: **for all** data chunks $B_i \in \mathcal{S}$ **do**
3: apply $C' \leftarrow$ new component classifier built on $B_i$;
4: $w'_C \leftarrow \frac{1}{MSE_{r+c}}$ ([?]);
5: **for all** classifiers $C_j \in \mathcal{E}$: **do**
6: apply $C_j$ on $B_i$ to derive $MSE_{ij}$;
7: compute weight $w_{ij}$ based on ([?]);
8: **end for**
9: **if** $|\mathcal{E}| < k$ **then**
10: $\mathcal{E} \leftarrow \mathcal{E} \cup C'$;
11: **else**
12: substitute least accurate classifier in $\mathcal{E}$ with $C'$;
13: **end if**
14: **for all** classifiers $C_j \in \mathcal{E} \setminus C'$ **do**
15: incrementally train classifier $C_j$ with $B_i$;
16: **end for**
17: **if** $memory\_usage(\mathcal{E}) > m$ **then**
18: prune (decrease size of) component classifiers;
19: **end if**
20: **end for**

**Figure 4.** Algorithm 2 [22]: Accuracy Updated Ensemble Algorithm (AUE2).

methods including the Naïve Bayes algorithm in reference to memory costs, processing time, and classification accuracy using various drift scenarios. The experiments were implemented on the MOA framework using real-world and synthetic datasets representing concept drifts occurring at various rates such as gradual, recurrent, and sudden. The authors indicate during the experiments they also observe the differences in the algorithms' performance.

The authors note the Naïve Bayes algorithm was added to their comparison as a reference in comparing an algorithm which has no drift mechanism.

The authors indicate their experimental evaluation was done using an evaluation method of data chunk evaluation in which they explain is an evaluation method which works similarly to the test-then-train design, but instead of using single examples it uses data chunks [23]. The authors also describe the benefits of this type of evaluation method as well as why they used it within their experimental evaluation.

The authors report their experiments revealed that the NB algorithm was severely malfunctioning in the presence of drifts, followed by a few other evaluated algorithms in which they believe NB failed due to its lack of containing a drift reaction mechanism and in return does not successfully learn from data streams with recurrent gradual drifts. The authors denote the experiments also revealed that in the existence of sudden recurring drifts, AUE1 and AUE2 performed the best and the accuracy produced had a considerable impact by only the first type of drift. The authors indicate the experimental evaluation showed the remaining

algorithms were lagging behind the AUE1 and AUE2 algorithm in terms of accuracy within gradual drifts which are recurring. The authors also note during the experimental evaluation, AUE1 and AUE2 abruptly reduced their memory. The authors conclude AUE2 produced the best average classification accuracy while consuming the least amount of memory.

We have organized some of the evaluation metrics produced from the authors' experimental evaluation based on the results illustrated within [23], these evaluation metrics are included in a comparison chart provided within **Table 1** in Section 6.

**Advantages of this research:** The authors introduce an algorithm which they call the Accuracy Updated Ensemble (AUE2) derived from the preliminary version, AUE1, with enhancements and modifications. The authors reiterate their hybrid algorithm's ability to react equally well to various types of drift scenarios such as gradual, sudden, recurring, mixed and short-term. The authors note their algorithm is inspired by the ensemble weighting mechanism of the Accuracy Weighted Ensemble with the combination of the incremental training of component classifiers. The authors report their experimental evaluation demonstrated AUE2 can offer substantial classification accuracy within static environments as well as environments containing various types of drifts [22]. The authors conclude their experimental evaluation revealed in comparison to the other ensemble approaches, AUE2 produced the most excellent average classification accuracy and least memory consumption. In reference to our survey, we believe the Accuracy Updated Ensemble (AUE2) can be of great benefit and effective in our approach of constructing an ensemble of Hoeffding Trees for anomaly detection within cyber data streams. This proposed approach is explained in more detail in Section 6.

**Disadvantages of this research:** The authors indicated that AUE2 as the rest of the algorithms was not sufficient and failed in reactions to the change. The authors note this could be an interesting topic for further research, the complex combination of drifts.

## 4. Surveying Hoeffding Trees for Anomaly Detection

### 4.1. Hoeffding Trees for Anomaly Detection

In this section, we evaluate a type of Hoeffding Tree used for Anomaly Detection and provide the advantages and disadvantages of the proposed research work.

#### Hoeffding Trees as Adaptive Trees for Real-Time Cyber-Power Event and Intrusion Classification

Adhikari *et al*. in [12] introduce the use of Hoeffding Adaptive Trees (HAT), with the augmentation of the drift detection method (DDM), along with adaptive windowing (ADWIN) for effectively classifying real-time cyber contingencies such as the interruption of power delivery by cyber-attacks implemented against electronic transmission systems. The authors introduce what they call research proposed towards an Intrusion Detection System for cyber-power

Event, called EIDS in which they use HAT with DDM & ADWIN to classify binary-class and multi-class cyber-attacks of contingencies of traditional power systems.

The authors define the Hoeffding Adaptive Trees (HAT) algorithm as a derivation from Concept-adapting Very Fast Decision Trees, CVFDT, as well as a modification to the original version of the Hoeffding Tree algorithm with enhanced features such as incremental design for its construction and the constant updating of its model at any moment of the detection of change and classifying the events based on this updated model. The authors mention they also use ADWIN with HAT for monitoring the classification error rate.

The authors evaluate their proposed research work through several experiments performed in which they use the prequential evaluation technique during experimentation and define this prequential evaluation technique. The authors indicate they use the Kappa statistic [24] along with the classification accuracy as a means of providing an indication of the algorithms ability to correctly classify samples individually. The authors note the Kappa statistic was introduced by Cohen *et al*. in [25] and is a great way to evaluate performance for stream classification The authors use a dataset containing 45 classes of artifacts of cyber-power contingencies classes for the experimental evaluation and perform binary-class as well as multiclass classification on the datasets using their newly proposed method of HAT with DDM & ADWIN. The authors note the MOA application was used for the experimental evaluation in which the datasets were initially preprocessed then submitted to the application.

The authors note the experimental evaluation revealed classification remained high although there were significant changes within the data which they believe demonstrates HAT's ability to dynamically adapt to the changing behavior of cyber-attacks. The authors also note other interesting features revealed during experimentation such as the increased changes detected for multiclass by the algorithm as opposed to binary-class and HAT's ability to not only dynamically adapt to the changing behavior of power systems, but update the model as well as detect changes within the data.

The authors suggest in accordance to the Kappa statistic and comparison with other various classifiers, HAT with DDM & ADWIN had promising performance and had the highest Kappa statistic and accuracy in which there was a classification accuracy produced greater than 94% for multi-class and greater than 98% for multi-class.

We have organized some of the evaluation metrics produced from the authors' experimental evaluation based on the results illustrated within [12], these evaluation metrics are included in a comparison chart provided within **Table 1** in Section 6.

**Advantages of this Research:** The authors within this research report the support of their presumption that HAT with DDM & ADWIN is an ideal approach for the classification of cyber attacks within power systems and this is

due to the high Kappa statistic, high accuracy, small evaluation time, and low model cost produced from this algorithm combination. The authors also indicate this is due to the capabilities of the algorithm's combination of HAT with DDM & ADWIN to continuously update the model as streaming data arrives as opposed to traditional batch processing which requires to train periodically. In reference to our survey, we believe this algorithm combination of HAT with DDM & ADWIN can be a significant in the combination of a Hoeffding Tree ensemble for detecting anomalies in streaming cyber data streams. We explain our premise of the use of this algorithm combination in Section 6.

**Disadvantages of this research:** This research work is only centered on Cyber Events within Power Systems, but can be extended for future work for the use of the Hoeffding Adaptive Trees for anomaly or intrusion detection in other types of systems.

## 5. Literature Review on Ensembles

Due to their superior performance and simple use in various application domains in comparison to strong single classifier learners, ensemble-based methods have become the most common type of conventional method used for data stream classification. It is important to note that ensemble learners are not only feasible for simple data stream learning but can be combined with drift detection algorithms and incorporate dynamic updating or a weighting system to remove selected or additional classifiers. Several research works evolve around ensemble structures including the works mentioned in Section 3. To have a better understanding of all the types of proposed ensemble-based algorithms, Gomes *et al*. in [26], provide a survey based on proposing a taxonomy for data stream ensemble learning derived from reviewing over 60 ensembles learning algorithms. Their survey thoroughly discusses important aspects of these algorithms such as combination, diversity, and dynamic updates. The authors also mention a list of open-source tools and discuss the current data stream challenges and how they relate to ensemble learning in reference to big data streams, concept evolution, feature drifts, temporal dependencies, and etc.

The authors in [26] highlight other surveys and books focusing on data stream learning for machine learning in general and also existing works addressing ensembles for batch learning environments. The authors in [26] differentiate their survey from these surveys, books, and research work based on the fact that other machine learning surveys and research works only refer to ensemble methods as an option for data stream learning especially in the context of concept drift or they focus on a general problem or specific machine learning task. In reference to previous ensembles for batch-learning environments, the authors in [26] suggest previous works conclude based on assuming learning is performed on static datasets. Based on the exploration of how the characteristics of stream setting change ensemble methods, the authors in [26] mention works on the example for handling concept drift that include combination of weighting functions, vot-

ing, and training methods. The authors in [26] indicate their goal is to clarify the characteristics involving the application of ensemble learners on a data stream context by proposing a taxonomy that organizes general techniques, presenting a classification of over 60 ensemble algorithms according to this taxonomy and discussing current and future trends for ensemble learning on a stream setting, which also includes big data stream processing. The authors in [26] convey their proposed taxonomy outlines the intersections between ensemble learning on static datasets with that of dynamic data streams while highlighting characteristics from ensemble learning that are unique and beneficial to the data stream learning setting.

For the proposed taxonomy in [26], the authors not only arrange ensemble-related techniques based on diversity, base learner, and combination, but they also discuss characteristics that influence the ensemble formulation that are unique to data stream learning in which they refer to as "update dynamics". An example of this is representing important methods for stream learning such as strategies to cope with drifts, how learning is performed and when to remove or add classifiers. The taxonomy presented in [26] is organized based on general aspects related to algorithms in a data stream learning setting which include aspects such as when they are directly mapped as characteristics of an actual algorithm, they are better represented as values rather than dimensions. An example is cardinality corresponds to a dimension, while fixed and dynamic are values.

The authors in [26] discuss combination techniques of ensemble methods such as the overall performance of combining ensemble members' predictions and its benefits and hindrance. In this particular discussion in [26], the authors differentiate between the voting method and the ensemble members' architecture. The authors discuss the different type of combination architectures such as Flat, Meta-Learner, Hierarchical, and Network. The meta-level combiner is used within the research work we discussed in Section 3, the Ensembles Restricted Hoeffding Trees. The authors in [26], also discuss the different types of voting methods such as majority voting, weighted majority voting, rank voting, classifier selection voting, and relational voting. In addition, the authors in [26] elaborate on diversity and the importance of diversity in ensemble-based classifiers while including background information on different ways of inducing diversity, such as input manipulation, output manipulation, base learner manipulation, and Heterogeneous base learners, as well as techniques for measuring diversity. The authors in [26] discuss in detail the base learner and different types of Update dynamics: cardinality; learning, focusing on learning mode, adaptation methods; and model management. The authors describe the different learning modes such as Incremental, landmark windows, sliding windows, damped windows, and adaptive windows.

To review the ensemble algorithms for data stream learning from a broader perspective, the authors in [26], discuss the heuristics in the development of new ensemble algorithms, the computational resources management and big data

streaming mining, ensemble classifiers in stream setting concerns and the open-source software. Some of the topics discussed within ensemble classifiers in Stream Setting Concerns in [26], include concept evolution, feature evolution, drift, selection, and partially labeled instances. The open-source software discussion in [26], entails open-source frameworks and libraries and workflow engine that comprise data stream learning, including ensemble-based algorithms. Developed among research groups, the authors in [26] remark these frameworks allow researchers to directly test their ideas without being concerned with common problems like algorithm evaluation or parallel implementations. These open-source frameworks include: Massive Online Analysis (MOA), Advanced Data Mining and Machine Learning System (ADAMS) [27], Scalable Advanced Massive Online Analysis, JUBATUS [28], Vowpal Wabbit [29], StreamDM, and others. A reference to MOA and StreamDM can be found in Section 3, where we perform experiments using these frameworks.

The authors in [26] conclude their work as a presentation to the main characteristics of ensemble learners, identifying open-source software, and discussing current and expected future trends in ensemble learning for data streams, such as concept and feature evolution, diversity measurement, big data stream learning, temporal dependencies, and instance labeling.

Our survey differentiates from [26], as we focus on ensembles of Hoeffding Trees in the context of anomaly detection in streaming cyber datasets. The information presented from the authors in [26] can be used as a basis for understanding our work and the need for this type of research on ensemble learning in reference to real-world problems and/or data stream classification in general. The authors in [26], also describe various existing research work which can help in the context of gaining more in depth knowledge on the use of ensembles for various problems including the scope of concept drift. Although concept drift can be deeply related to anomaly detection, as concept drift is the study of the changes in the underlying distribution of data and anomaly detection is the study of addressing normal versus abnormal patters in data, there is still a need to put forward research work exploring ensembles for addressing anomaly detection as a separate commodity as anomaly detection and/or intrusion detection is an immense field which can benefit from these techniques. This introduces our goal in this research survey. The work in [26], presents material which serves as a foundation in the process of constructing the necessary ensemble learning algorithm for this type of research work: ensembles addressing the problem of detecting anomalies in cyber datasets.

## 6. Discussions

Hoeffding Tree models with integrated features enhance existing machine learning models for anomaly detection in streaming datasets. This is due to their significant performance in streaming datasets; as well as their flexibility to add an adaptive, incremental learning, distributive, and drift detection approach.

Combining these methods along with the approach of using Hoeffding Trees for an intrusion detection system, we can suggest which Hoeffding Trees are a best fit for detecting threats in streaming cyber datasets. The Accuracy Updated Ensemble (AUE2) uses ensembles of Hoeffding Trees to react to various types of drifts while the Restricted Ensembles of Hoeffding Trees using Stacking uses an ensemble of Hoeffding Trees along with ADWIN change detection to detect changes as well as reset the learning rate. The Hoeffding Adaptive Trees use DDM along with ADWIN to detect cyber-attacks within power systems. The distributed Hoeffding Trees, StreamDM, are an extension of the original Hoeffding Trees, but use parallelism to manage memory consumption and execution time as well as Spark Streaming for real-time processing.

In reference to the surveyed algorithms mentioned above, a comparison chart, shown in Table 2, has been created indicating measurements produced from each algorithm based on accuracy, Kappa statistic, time, and/or memory. All of the surveyed algorithms may not include measurements for each of these statistics but the statistic for the corresponding algorithm is included within the chart along with the corresponding evaluated dataset which produced those statistics.

As stated in the introduction, we introduce the use of techniques from one composition to solve the problem of a different composition. Here we discuss how the combination of some of the different compositions presented in the previous sections of this survey can solve the problem of anomaly detection in streaming cyber datasets. An overview of our proposed combination of compositions is depicted in Figure 5.

In Figure 5, the composition techniques we identify as key components are the ensembles of Hoeffding Trees built on a distributed platform. This ensemble of Hoeffding Trees contains two types of Hoeffding Trees, general Hoeffding Trees and HAT with ADWIN & DDM (Hoeffding Adaptive Trees with DDM and ADWIN). This ensemble of general Hoeffding Trees and HAT with ADWIN & DDM could be based on the methodology presented in Section 3.1, discussing the Hoeffding Trees using Stacking or Section 3.2, discussing the

**Table 2.** Comparison chart of surveyed hoeffding tree algorithms based on the dataset producing the highest accuracy, Kappa statistic, time, and/or memory produced.

| Hoeffding Tree Algorithm | Dataset | Accuracy | Kappa Statistic | Time | Memory |
|---|---|---|---|---|---|
| StreamDM | NSL-KDD | 93% | 90% | | |
| Restricted Hoeffding Trees | RT | 92.15% | | 1732.25 s | 201.34 MB |
| AUE2 with Buffer | RBF | 94.07% | | | 2.73 MB |
| AUE2 without Buffer | RBF | 94.06% | | | 2.15 MB |
| HAT + DDM + ADWIN | cyber-attacks (2 classes) | 98% | 94% | | |
| HAT + DDM + ADWIN | cyber-attacks (41 classes) | 92% | 91% | | |

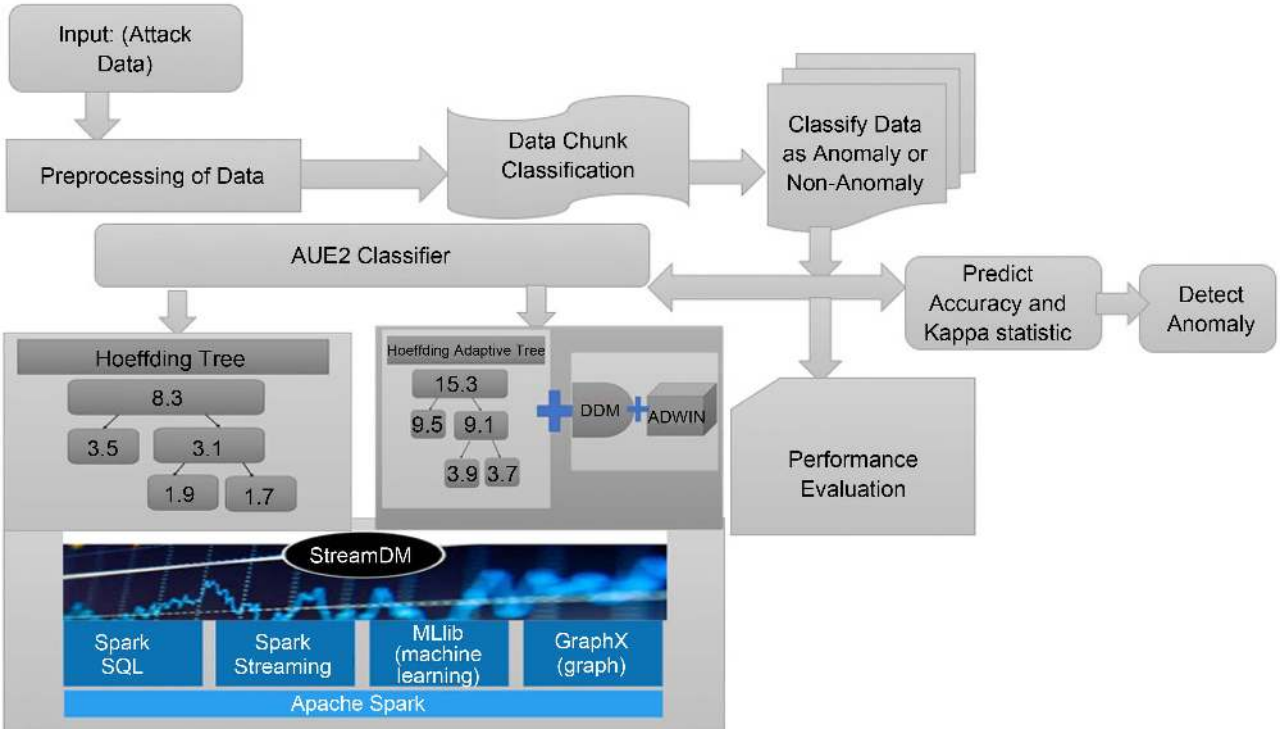


**Figure 5.** Combination of compositions to address anomaly detection in streaming cyber datasets.

AUE2 ensemble. In Figure 5, we propose using the AUE2 ensemble architecture for our proposed methodology; however, in the case of implementation, either ensemble techniques are effective and can be used as well as examined further to determine the best fit for the type of work.

In Figure 5, we also show that the ensembles of Hoeffding Trees for the proposed method we suggest are to be developed on StreamDM, the distributed platform for Hoeffding Trees built on top of Spark using Spark streaming. This StreamDM platform not only allows for distribution during execution but also the ability for real-time streaming data. Due to Spark Streaming as a category for real-time streaming.

A further explanation of Figure 5 can be as follows:

- Input (Attack Dataset): This can be the type of attack dataset that will be inputted into the system. An example is the NSL-KDD dataset, artificially generated attack dataset, or any other type of dataset containing a single attack or multiple attacks.
- Preprocessing of Data: This is the stage where the data is formatted into the necessary format required by the system. An example is using the NSL-KDD dataset in a format that is handled by StreamDM, such as. arff format or any other stream reader format including Kafka or Socket Streaming for real-time Spark Streaming.
- Data Chunk Classification: The AUE2 ensemble algorithm works by passing data chunks to each classifier within the ensemble. In our case, these classifiers are the general Hoeffding Trees and HAT with ADWIN & DDM.

- Classify Data as Normal or Attack: This is the stage where the ensembles within the AUE2 classifier perform classification. This can also be known as a stage where these models (general Hoeffding Trees and HAT with ADWIN & DDM) classify the data within the dataset as normal or anomalous.
- Predict Accuracy and Kappa statistic: As we know in order to know the true classification of an algorithm we use the accuracy (number of correctly classified instances) it produces as well as other common statistics such as the Kappa statistic. We choose the Kappa statistic because of its as a good indicator of classification performance in streaming data. In our case we will use these metrics to determine the effectiveness of anomaly detection by the classifiers within the ensemble.
- Anomaly Detection: As stated in the list item above, this is used as a blueprint for the models (classifiers) within the AUE2 ensemble of predicting anomaly detection.
- Performance Evaluation: This stage other metrics are taken into consideration such as the performance of the classifiers.
- AUE2 Classifier: As we stated we will use the AUE2 ensemble approach, this item in the diagram signifies that this complete procedure resides on top of the AUE2 classifier ensemble approach.
- Hoeffding Trees/Hoeffding Adaptive Tree + DDM + ADWIN: This represents the classifiers within the AUE2 ensemble, in which this case, is the Hoeffding Trees and Hoeffding Adaptive Trees with ADWIN & DDM.
- StreamDM/Apache Spark: This item in the diagram signifies that this complete AUE2 ensemble approach as well as complete procedure is built on top of the StreamDM/Apache Spark Streaming platform.

From our analysis of the proposed research works surveyed in the previous sections, we believe in terms of machine learning, the combination of these compositions can be effective in addressing the problem of anomaly detection in streaming cyber datasets. A pivotal factor in determining if the characteristics of this type of proposed research study is feasible, is the experimentation and implementation of this proposed combination as well as the use of a diverse cyber dataset which allows for the flexibility to include public datasets such as the NSL-KDD dataset and more, an artificially generated attack dataset, and/or both.

## 7. Conclusions and Suggestions

Anomaly detection in streaming datasets is the ability to handle high volumes of abnormal data patterns in the distribution of data. In this survey, we have discussed different problem compositions that are relevant in varied streaming data applications domains within and outside of anomaly detection. These domains are explained in the Introduction section. We note that two of the three compositions are not particularly exhaustive and not focused on anomaly detection, but can be used for addressing the problem. We also note that the anomaly detection problem might be composed in other ways and with other machine learning al-

gorithms but for the use of Hoeffding Trees most of the existing work can be covered under these three compositions. We also bring fourth background literature from existing surveys addressing the need for ensembles, in which we relate our work to this background literature to reinforce the need for this type of research, the combination of compositions within ensembles in the area of anomaly detection and/or intrusion detection.

For each problem composition, we present advantages and disadvantages of the techniques within the problem composition. We discuss our proposed combination of compositions within ensembles to address anomaly detection within streaming cyber datasets based on the surveyed proposed research work of each composition. Within our discussion we also provide a comparison chart based on the evaluation metrics produced from the experimental evaluations performed for the proposed research work within each composition in terms of accuracy, Kappa statistic, time, and/or memory.

As stated in our discussion, the information surveyed in this paper has helped bring fourth an understanding of state-of-the-art classification algorithms or compositions which can be combined to effectively address the anomaly detection problem in streaming cyber datasets. Since this paper only focused on the surveying of these compositions for introducing a research study on an ensemble approach using diversified proposed research works to solve a problem within the anomaly detection domain, a further understanding of determining if this type of proposed research is feasible and effective, is the experimentation and implementation of our proposed combination of compositions shown in Figure 5 and thoroughly described in the "Discussions" section along with the use of a diverse cyber datasets which allows for the flexibility to include public datasets, artificially generated attack datasets, and/or both.

Although, the focus of this survey paper has been on Hoeffding Trees for streaming datasets and how these can be used for anomaly detection in streaming cyber datasets, many of the techniques discussed here can also be applicable to concept drift in streaming datasets.

While the literature on anomaly detection for streaming datasets is rich, there are several research directions that need to be explored in the future. In this survey, we have provided a high level comparison of various techniques which can be used. An experimental study of these techniques is essential for a more in-depth understanding of their characteristics.

## Acknowledgements

The research presented in this paper was supported by the Office of the Assistant Secretary of Defense for Research and Engineering (OASD (R&E)) agreement FA8750-15-2-0120 and Boeing Data Analytics agreement.

## References

[1] Bifet, A. and Gavaldà, R. (2009) Adaptive Parameter-Free Learning from Evolving

Data Streams. *Proceedings of the* 8*th International Symposium on Intelligent Data Analysis*: *Advances in Intelligent Data Analysis*, 249-260. https://doi.org/10.1007/978-3-642-03915-7_22

[2] Sun, Y., Wang Z., Liu, H., Du, C. and Yuan, J. (2016) Online Ensemble using Adaptive Windowing for Data Streams with Concept Drift. *International Journal of Distributed Sensor Networks*. https://doi.org/10.1155/2016/4218973

[3] Breiman, L., Friedman, J., Olshen, R. and Stone, C. (1994) Classification and Regression Trees. Wadsworth and Brooks, Monterey.

[4] Quinlan, R.J. (1993) C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning). Morgan Kaufmann.

[5] Domingos, P. and Hulten, G. (2000) Mining High-Speed Data Streams. *Proceedings of the* 6*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, 71-80. https://doi.org/10.1145/347090.347107

[6] Yang, H. and Fong, S. (2011) Moderated VFDT in Stream Mining using Adaptive Tie Threshold and Incremental Pruning. 13*th International Conference on Data Warehousing and Knowledge Discovery*, Toulouse, 471-483. https://doi.org/10.1007/978-3-642-23544-3_36

[7] Zhang, L. and Lin, J. (2017) Sliding Window-Based Fault Detection from High-Dimensional Data Streams. *IEEE Transactions on Systems*, *Man*, *and Cybernatics Systems*, **47**, 289-303.

[8] Rettig, L., Khayati, M., Cud Mauoux, P. and Piòrkowski, M. (2015) Online Anomaly Detection over Big Data Streams. *IEEE International Conference on Big Data* (*Big Data*), Santa Clara.

[9] Du, Y., Liu, J., Fang, L. and Chen, L. (2014) A Real-Time Anomalies Detection System Based on Streaming Technology. 6*th International Conference on Intelligent Human-Machine Systems and Cybernatics*, Hangzhou.

[10] Tsymbal, A. (2204) The Problem of Concept Drift: Definitions and Related Work. Tech. Rep. Department of Computer Science, Trinity College, Dublin.

[11] Widmer, G. (1996) Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning*, **23**, 69-101. https://doi.org/10.1007/BF00116900

[12] Adhikari, U. and Pan, S. (2017) Applying Hoeffding Adaptive Trees for Real-Time Cyber-Power Event Intrusion Classification. *IEEE Transactions on Smart Grid*, **PP**, 1. https://doi.org/10.1109/TSG.2017.2647778

[13] Choudhary, P. (2017) Introduction to Anomaly Detection. Data Science Company. https://www.datascience.com/blog/intro-to-anomaly-detection-learn-data-science-tutorials

[14] Bifet, A., Mantu, S., Qian, J., Tian, G., He, C. and Fan, W. (2015) Stream DM: Advanced Data Mining in Spark Streaming. *IEEE International Conference on Data Mining Workshop*, Atlantic City.

[15] Smith, T.C. and Eibe, F. (2016) Statistical Genomics: Method and Protocols. Chapter Introducing Machine Learning Concepts with WEKA, Springer, New York, 353-378.

[16] Bifet, A., Holmes, G., Pfahringer, B. and Gavaldà, R. (2009) New Ensemble Methods for Evolving Data Streams. *Proceedings of the* 15*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Paris, 139-148. https://doi.org/10.1145/1557019.1557041

[17] Bifet, A., Frank, E., Holmes, G. and Pfahringer, B. (2012) Ensembles of Restricted Hoeffding Trees. *Journal ACM Transactions on Intelligent Systems and Technolo-*

*gy*, **3**, Article No. 30. https://doi.org/10.1145/2089094.2089106

[18] Wolpert, D.H. (1992) Stacked Generalization. *Journal Neural Networks*, **5**, 241-259.

[19] Holmes, G., Kirkby, R. and Pfahringer, B. (2005) Stress-Testing Hoeffding Trees. *Proceedings of the* 9*th European Conference on Principles and Practice of Knowledge Discovery in Databases*, Porto, 495-501. https://doi.org/10.1007/11564126_50

[20] Oza, N.C. and Russel, S.J. (2001) Online Bagging and Boosting. *Proceedings of the Conference on Artificial Intelligence and Statistics*, Key West, 105-112.

[21] Margineantu, D.D. and Dietterich, T.G. (1997) Pruning Adaptive Boosting. *Proceedings of the* 14*th International Conference on Machine Learning*, San Francisco, 211-218.

[22] Brzeninski, D. and Stefanowski, J. (2014) Reacting to Different Types of Concept Drift: The Accuracy Updated Ensemble Algorithm. *Proceedings of IEEE Transactions on Neural Networks and Learning Systems*, 81-94. https://doi.org/10.1109/TNNLS.2013.2251352

[23] Brzezinski, D. (2010) Mining Data Streams with Concept Drift. MS Thesis, Inst. Comput. Sci., Poznan Univ. Technology, Poznan.

[24] Julius, S. and Wright, C. (2005) The Kappa Statistic in Reliability Studies: Use, Interpretation, and Sample Size Requirements. *Journal Physical Therapy*, **85**, 257-268.

[25] Cohen, L., Avrahami-Bakish, G., Last, M., Kandel, A. and Kipersztok, O. (2008) Real-Time Data Mining of Non-Stationary Data Streams from Sensor Networks. *Journal Information Fusion*, **3**, 344-353.

[26] Gomes, H.M., Barddal, J.P., Enembreck, F. and Bifet, W. (2017) A Survey on Ensemble Learning Data Stream Classification. *ACM Computing Surveys*, **50**, Article No. 23.

[27] Reutemann, P. and Vanschoren, P. (2012) Scientific Workflow Management with ADMS. Machine Learning with Knowledge Discovery in Databases, Bristol, 833-837.

[28] Hido, S., Tokui, S. and Oda, S. (2013) Jubatus: An Open Source Platform for Distributed Online Machine Learning. NIPS 2013 Workshop on Big Learning Lake Tahoe, Nevada.

[29] Langford, J. (2015) vowpal wabbit. https://github.com/JohnLangford/vowpal_wabbit