SALAD: A Split Active Learning based Unsupervised Network Data Stream Anomaly Detection method using Autoencoders

Christopher Nixon[a], Mohamed Sedky[a], Justin Champion[a], Mohamed Hassan[b]

[a] *School of Digital, Technologies and Arts, Staffordshire University, UK*
[b] *Department of Computing, Leeds Beckett University, UK*

Abstract

Machine learning based intrusion detection systems monitor network data streams for cyber attacks. Challenges in this space include detecting un- known attacks, adapting to changes in the data stream such as changes in underlying behaviour, the human cost of labeling data to retrain the machine learning model and the processing and memory constraints of a real-time data stream. Failure to manage the aforementioned factors could result in missed attacks, degraded detection performance, unnecessary expense or de- layed detection times. This research proposes a new semi-supervised network data stream anomaly detection method, Split Active Learning Anomaly De- tector (SALAD), which combines our novel Adaptive Anomaly Threshold and Stochastic Anomaly Threshold with Fading Factor methods. A novel Reconstruction Error based Distance from Threshold strategy is proposed and evaluated as part of an active stream framework to demonstrate reduc- tion in labeling costs. The proposed methods are evaluated with the KDD Cup 1999, and UNSW-NB15 data sets, using the scikit-multiflow framework. Results demonstrated that the proposed SALAD method offered equivalent performance to full labeled and alternative Naïve Bayes (NB) and Hoef- fiding Adaptive Tree (HAT) methods, with a labeling budget of just 20%,

significantly reducing the required human expertise to annotate the network data. Processing times of the SALAD method were demonstrated to be significantly lower than NB and HAT methods, allowing for greatly improved responsiveness to attacks occurring in real time.

Keywords: Active Learning, Online Learning, Autoencoders, Anomaly Detection, Intrusion Detection System

---

1. Introduction

Autoencoders (AE) can be used as an unsupervised computer network anomaly detector for cyber security use cases. Anomaly detection is typically achieved by comparing the resulting AE Reconstruction Error (RE) value for a given data item against a threshold value, with values below threshold belonging to the normal population, and those above considered an anomaly. Choosing a suitable threshold value is non-trivial for computer network data streams, where the normal and anomaly distributions can overlap and change overtime due to concept drift, meaning that achieving an optimal accuracy is practically impossible where the threshold value is fixed. A number of offline threshold methods such as average RE (Vaiyapuri and Binbusayyis, 2020), Naïve (Mirsky et al., 2018), Stochastic (Nicolau and McDermott, 2016; Autoencoder et al., 2022; Aktar and Yasin Nur, 2023), and Density based (Catillo et al., 2023), are proposed in the literature, but the area of online threshold adaptation is under explored.

In our previous paper (Nixon et al., 2020) we introduced two AE threshold methods for network data streams: *Naïve Threshold Method with Decay* (NATD), which decayed the maximum observed RE threshold value over time to force a new value to be adopted as the data stream evolves, avoiding fixing the threshold at an unrealistic maximum during early training; and *Stochastic Anomaly Threshold* (SAT) which selected a threshold that achieved maximum accuracy, from between the mean RE and three standard deviations from the mean, requiring data labels in order to calculate the accuracy. Overall the SAT method achieved the higher accuracy and F1-score over NATD, although was not able to outperform other supervised Naïve Bayes (NB) and Hoeffding Adaptive Tree (HAT) methods, that were used as a benchmark. A key finding of our previous work was that the performance of the AE method is influenced by the selection of a suitable anomaly threshold method, and that the running time is significantly lower than the

benchmark NB and HAT, making the AE method highly desirable for online data stream processing.

This work aims to test the hypothesis that the AE anomaly threshold method can be further improved to adapt to the data stream with the inclusion of forgetting mechanism to balance the effects of gradual and abrupt concept drift. A further aim is to explore a semi-supervised approach for stream-based anomaly detection, with the inclusion of an Active Learning (AL) framework to balance the labelling budget throughout the data stream and test if this can further enhance performance by highlighting more relevant samples for both training new AE models, in the event that the normal population drifts, and selecting suitable threshold values.

We propose a semi-supervised Split Active Learning Anomaly Detection (SALAD) method that combines the autoencoder anomaly detector with a novel Adaptive Anomaly Threshold (AAT) or Stochastic Anomaly Threshold with Fading Factor (SAT FF) threshold method which uses a memory based fading factor method to incorporate previous data instances into threshold decisions, allowing for short and long term change to be balanced. Secondly the AAT method introduces a novel method to identify chunks of normal data instances to improve overall detection accuracy.

The contributions of this paper are as follows:

1. Inclusion of a fading factor to extend our previous SAT method (Nixon et al., 2020), SAT FF, allowing the effects of gradual and abrupt concept drift to be accommodated when updating the anomaly threshold.
2. An Adaptive Anomaly Threshold (AAT) method that broadens the threshold search range to the maximum observed RE value instead of three standard deviations, allowing for chunks of normal instances to be appropriately classified where values greater than three standard deviations would have been missed by SAT.
3. AE anomaly detection is combined with an Active Stream Framework (Žliobaitė et al., 2013), using a split strategy that combines the random strategy with a novel RE based distance from threshold strategy. This reduces the necessary labels to find an appropriate anomaly threshold whilst improving overall performance of the AE method by allowing retraining to occur on the most relevant normal samples to adapt to drift.

The remainder of this article is organised as follows: Section 2, introduces related work; Section 3, describes the proposed Split Active Learning

Anomaly Detector (SALAD) method; Section 4, presents the evaluation results; Section 5, discusses how SALAD provides a low cost anomaly detector for network data streams; and Section 6, presents conclusions.

2. Related Work

2.1. Autoencoder Anomaly Detection

An autoencoder uses an encoding function to produce a latent code representation of the input data, and a decoding function to reconstruct the input from the code representation (Nixon et al., 2020). The mean square Reconstruction Error RE between the reconstructed output $\hat{X}$ and original input X can be calculated using equation 1, where f is the encoding function, g is the decoding function, and n is the number of samples (Nixon et al., 2020), which can then be compared to an anomaly threshold to label a sample as either normal or anomalous.

$$\begin{aligned} \hat{X} &= g(f(X)) \\ \text{RE} &= \frac{1}{n}\sum_{j=1}^{n}(X_j - \hat{X}_j)^2 \end{aligned} \tag{1}$$

Autoencoder based anomaly intrusion detection methods are well established, whereby single layer denoising models (Nicolau and McDermott, 2016), Long Short Term Memory (LSTM), Recurrent Neural Network (Mirza and Cosan, 2018; Kieu et al., 2019), ensembled stacked autoencoders (Mirsky et al., 2018; Li et al., 2020), and sparsely connected networks (Chen et al., 2017; Kieu et al., 2019) have been previously demonstrated across a range of IDS data sets.

Several methods were proposed in the literature to determine the anomaly threshold, an important parameter in deciding whether to label a sample as a positive detection. The threshold can be set to the average RE value observed during training (Vaiyapuri and Binbusayyis, 2020). Naïve Anomaly Threshold (NAT) sets the threshold at the maximum observed RE during training (Mirsky et al., 2018). Stochastic Anomaly Threshold (SAT) (Nicolau and McDermott, 2016; Autoencoder et al., 2022; Aktar and Yasin Nur, 2023) sets the threshold based on the best observed accuracy or F1-score when stepping through threshold values within a range of the RE value distribution.

Nicolau and McDermott (2016) proposed an anomaly threshold method using Kernel Density Estimation. Catillo et al. (2023) proposed CPS-GUARD, where the RE is calculated for sets of inliers and outliers and a threshold is chosen that balances both sets in order to trade off between legitimate vs anomalous outliers.

There are few examples of online threshold methods in the literature. Odiathevar et al. (2022) developed a hybrid framework whereby an AE was trained offline on high dimension network data, and the resulting latent representation and median threshold used to select normal samples for incremental training of an online 1-class SVM classifier. A drawback of this framework, is that it requires both an offline and online model, which could add additional processing time and memory requirements.

Aiming to find an optimal network configuration, we evaluated in Nixon et al. (2020), an undercomplete autoencoder, regulated with connection dropout, with a prequential online test using the KDD Cup 1999 and UNSW-NB15 data sets. Applying a single layer autoencoder with dropout probability of 0.1, using the Stochastic Anomaly Threshold method, provided an accuracy of 98% and F1-score of 0.812, using the KDD Cup 1999 data set, with a significantly improved running time compared to traditional Naïve Bayes (NB) and HAT online methods. Evaluation on the UNSW-NB15 data set using a 3-layer network and dropout probability of 0.2 returned an accuracy of 79.1% and F1-score of 0.703. The results showed that the SAT threshold performed better than the NAT, and that more complex data sets benefit from experimenting with the number of layers and regularisation of the network.

2.2. Concept Drift Detection with Active Learning

Non-stationary network data streams may experience real concept drift (Gama et al., 2014), whereby the posterior probability of classes will change over time due to changes in network behaviors, the cause of which could be either benign or adversarial in nature. The posterior probability is defined as $p(y|X)$ which represents the probability of class y given an observation X (Gama et al., 2014). Autoencoders determine outliers using the RE-score, based on the hypothesis that adversarial behaviour deviates from the learned 'normal' representation resulting in scores above the anomaly threshold. Real concept drift presents a challenge that the aforementioned hypothesis will weaken overtime, with changing benign data also scoring above threshold, raising the false positive rate. Increasing the anomaly threshold does not

present an optimal solution as although the false positive rate may lower, the false negative rate could increase and so is not recommended.

Active learning (AL) aims to select the most relevant samples for training based on the use of a learning strategy and restriction of the labelling cost to a specified budget (Žliobaitė et al., 2013), which is useful where it is infeasible to label the entire data set for training and choosing more relevant samples should speed up convergence time. Tharwat and Schenck (2023) provides an overview of state-of-the-art AL methods, covering membership query synthess, stream-based, and pool-based scenarios, and related query strategies for both information and representation-based approaches. DeepAL is presented as an advanced topic and further expanded on by Ren et al. (2022), requiring batch based strategies to provide the volume of labels and diversity required by deep learning training. The DeepAL methods are focused on pool-based approaches and using the probabilistic output of the softmax layer, which is not relevant to the AE approach.

Stream-based research has recently focused on partially-labelled data streams and propose both Semi-Supervised (SSL) and AL as methods to train classifiers in thses scenarios (Gomes et al., 2022; Fahy et al., 2023). Adaptation to concept drift is highlighted as challenge, with statistical based Confidence Distribution Batch Detection, and a performance based Active Stream Framework (ASF) (Žliobaitė et al., 2013) both outlined. ASF combines performance change detection with a labeling strategy and a fixed budget B.

Žliobaitė et al. (Žliobaitė et al., 2013) discussed three requirements for stream-based AL strategies:

1. balance the labeling budget B over infinite time $\sum_D p(label|X)p(X) \leq B$;
2. detect changes anywhere in the instance space $x \in D$ then $p(label|X) > 0$;
3. preserve the distribution of incoming data for detecting changes $p(X|label) = p(X)$.

A number of strategies were evaluated against the aforementioned, including *fixed uncertainty* as demonstrated by Sethi and Kantardzic (2017), and *uncertainty with randomisation*, whereby the sensitivity threshold is randomly selected from a standard distribution to occasionally include samples outside of the uncertainty margin. Fixed uncertainty is only able to satisfy requirement one, and randomised uncertainty satisfies requirement one and two, but

neither can preserve the probability density of labelled data compared to the original distribution, which can bias the model (Žliobaitė et al., 2013). A further *split* strategy is introduced which satisfies all three requirements by splitting the the data stream into two, using uncertainty and random strategy exclusively on either stream. Both streams are used for training, but only the randomised stream is used for change detection (Žliobaitė et al., 2013). Shan et al. (2018) presents a split strategy, although in this approach adaptation is *blind*, based on incrementally updating the ensemble members with both uncertainty and random labels, offering no pro-active change detection, this could reduce overall adaptation speeds (Gama et al., 2014).

Shan et al. (2018) also proposed a stream-based AL change detection strategy based on margin uncertainty, 'OALEnsemble', however in this approach the ensemble members are trained on different windows of the data set, with a stable classifier and a series of short window 'dynamic' classifiers that are continually replaced as new blocks of the data stream are processed, to balance the detection of both sudden and gradual concept drifts. Labeling is restricted to samples within the uncertainty margin, with the addition of a random labeling algorithm to randomly include samples outside of the margin where drift may also be occurring. The stable classifier is incrementally trained with all new data, whilst dynamic classifiers are only trained on the most recent block and given a weight, providing importance to more recent data (Shan et al., 2018). Use of windows will result in larger memory requirement compared to use of a fading factor as outlined by Gama et al. (2013).

3. Methods

In our previous work (Nixon et al., 2020) we evaluated dropout probability, NAT with decay and SAT anomaly thresholds, and single vs stacked network structure, to find optimal autoencoder parameters. Building on this work, in this article, we further introduce a new Split Active Learning Anomaly Detector (SALAD) method, using a novel Adaptive Anomaly Threshold (AAT) and Stochastic Anomaly Threshold with Fading Factor (SAT FF) threshold methods. A novel Reconstruction Error based Distance from Threshold AL strategy is evaluated with an AL based Active Stream Framework (ASF) (Žliobaitė et al., 2013) with which we compare blind, random, RE distance from threshold, variable distance from threshold and split AL strategies. All methods are evaluated using a prequential, interleaved

Table 1: List of Symbols used in this paper

Symbol	Remarks
ϕ	Anomaly threshold
α	Fading factor
S	Fading sum
N	Fading number of instances
RE	Reconstruction error
RE_μ	Fading mean RE
RE_{MAX}	Maximum RE
β	Threshold sensitivity
B	Labelling budget
$\hat{b}$	Estimated budget
d	Distance from anomaly threshold
θ	Uncertainty confidence
v	Step Size

test-then-train method (Gama et al., 2014), whereby the model is first tested on previously unseen samples before training in a chunk wise fashion (Nixon et al., 2020), after an initial period of pre-training. Results are compared against traditional NB and HAT online learning methods using the KDD Cup 1999[1] 10% (Tavallaee et al., 2009) and UNSW-NB15[2] (Moustafa and Slay, 2015) data sets.

The Keras[3] neural networking (Chollet et al., 2015), version 2.3.1, and Scikit-Multiflow[4] stream learning (Montiel et al., 2018), version 0.4.1, frameworks for Python were used for this evaluation. The experiments were ran on a Windows 10 64bit PC with Intel i7 1.8GHz processor and 8GB RAM.

Observed metrics during evaluation included: accuracy, F1-score, kappa and total running time. For prequential evaluation the scikit-multiflow default of updating evaluation metrics every 200 samples was used.

3.1. Stream-based Threshold Methods

Stochastic Anomaly Threshold with Fading Factor (SAT FF)

[1]http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[2]https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-data sets/

[3]https://keras.io/

[4]https://scikit-multiflow.github.io/

Table 2: Threshold Method Comparison

Threshold Method	**Proactive**	**Forgetting Mechanism**	**Comments**
Average RE (Vaiyapuri and Binbusayyis, 2020)	No	No	Threshold blindly updates.
NAT (Mirsky et al., 2018)	No	No	Threshold set too high during initial training.
SAT (Nicolau and McDermott, 2016)	Yes	No	Threshold selected based on accuracy but does not consider previous data.
AAT (Proposed)	Yes	Yes	Threshold based on accuracy, tries max value first to simplify normal sample processing, includes fading average so threshold is based on previous data.

The challenge of a constantly changing data stream cannot be addressed by the SAT threshold where only the most recent data chunk is considered for update of the anomaly threshold. In order to represent the evolving data stream then the SAT method should consider previously observed samples. It is necessary to balance the influence of abrupt vs gradual change by use of a forgetting mechanism, such as a sliding window or fading factor (Gama et al., 2013). Fading factors are preferred as they are more memory efficient as they do not require the storage of previous samples.

The Stochastic Anomaly Threshold method (Aygun and Yavuz, 2017) was developed for offline learning and has been adapted to a novel *SAT with Fading Factor* (SAT FF) method for online data streams, where data instances are processed in a chunk wise fashion, so that previously observed data instances can influence the threshold decision. The SAT FF method uses a fading average RE value so that the relevance of previous samples can be gradually forgotten based on a fading factor α (Gama et al., 2013). The fading average, RE_μ, is calculated in lines 4 to 6 of algorithm 1, where S is the fading sum RE, and N is the fading number of instances, and i is the current sample number.

Adaptive Anomaly Threshold (AAT)

When dealing with chunks that contain only normal samples, where D is the data stream and chunk $C \subset D$, and $X \subseteq C$ is a subset of normal

instances, then if $|X| = |C|$ the predicted accuracy will be 100% for chunk C when threshold $\phi \geq \max(\text{predictRE}(x \in C))$, when applying the classification function a provided by equation 2, where ϕ is the anomaly threshold and β is the threshold sensitivity. Therefore the threshold update function should assume the maximum RE value when the current accuracy is 100%.

$$a(\text{RE}) = \begin{cases} 1, & \text{if RE} \geq \phi\beta \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The proposed *Adaptive Anomaly Threshold* (AAT) method, extending the previously outlined SAT FF method, is given in its entirety in algorithm 1. Lines 7 to 9 maintain RE_{MAX} throughout the data stream, lines 10 to 12 calculate the accuracy using RE_{MAX} as ϕ, and line 13 checks for perfect 100% accuracy to signify $|X| = |C|$, saving the need to iterate using SAT. If accuracy is lower then SAT (Nixon et al., 2020) continues in lines 14 to 24 in order to find an optimal ϕ. The proposed method is compared to previous threshold methods in table 2.

Algorithm 1: Adaptive Anomaly Threshold with SAT FF

Input : autoencoder m, input X, labels y, anomaly threshold ϕ, step size $v \leftarrow [> 0]$, fading factor α

Output: ϕ

$S_0 \leftarrow 0; N_0 \leftarrow 0; RE_{\max} \leftarrow 0;$
$X_{y \leftarrow 0} \subseteq X;$
$\mathrm{RE}_i \leftarrow$ `predictRE(`m, $X_{y \leftarrow 0}$`)`;
$S_i \leftarrow \mathrm{RE}_i + \alpha * S_{i-1};$
$N_i \leftarrow 1 + \alpha * N_{i-1};$
$\mathrm{RE}_\mu \leftarrow \frac{S_i}{N_i};$
if $RE_i > RE_{MAX}$ **then**
    $\mathrm{RE}_{\mathrm{MAX}} \leftarrow \mathrm{RE}_i;$
end
$\phi \leftarrow \mathrm{RE}_{\mathrm{MAX}};$
$\hat{y} \leftarrow$ `predict(`m,ϕ,X`)`;
$\mathrm{acc}_w \leftarrow$ `calcAccuracy(`$\hat{y}$,y`)`;
if $acc_w < 1.0$ **then**
    $\phi_w \leftarrow \phi;$
    while $\phi > RE_\mu$ **do**
        $\phi \leftarrow \phi - v;$
        $\hat{y} \leftarrow$ `predict(`m,ϕ,X`)`;
        $\mathrm{acc} \leftarrow$ `calcAccuracy(`$\hat{y}$,y`)`;
        if $acc > acc_w$ **then**
            $\phi_w \leftarrow \phi;$
            $\mathrm{acc}_w \leftarrow \mathrm{acc};$
        end
    end
    $\phi \leftarrow \phi_w;$
end

3.2. Stream-based Active Learning Framework

A primary challenge with the proposed AAT method is the requirement for supervised labels in order to calculate the accuracy value. This is not practical for infinite data streams and so it is proposed to combine the AE and AAT methods with a semi-supervised Active Learning approach in order to reduce the labelling cost. In addition to this, the data stream can

evolve and so a change detection method is required to identify drift occurring within the normal sample space and retrain the AE anomaly detector in order to adapt.

Active Stream Framework

Žliobaitė et al. (2013), proposed an *Active Stream Framework*, which combines change detection with a labelling strategy and a fixed budget B, and this framework is adopted for our evaluation. Algorithm 2 gives the framework that was evaluated, where $\hat{b}$ is the estimated budget. The labelling budget and AL query strategy, line 4, determine if the label of a given chunk should be queried. Lines 6 to 14 handle change detection, if a warning signal is received then a new autoencoder (AE_L) is trained with the most recent examples, and when a change is signalled, the current model is replaced with AE_L, completing adaptation to the new concept. For this evaluation the Drift Detection Method (DDM) (Gama et al., 2004) change detector is used.

The framework maintains a running estimate of label usage $\hat{u}_i$ over a fading window, line 18, using equation 3, where w is the size of the fading window and $label_i$ is the labelling decision either 0 or 1 at time i. The spending estimate $\hat{b}$ is then calculated from $\hat{u}_i$ over w, given in equation 4 (Žliobaitė et al., 2013). During this evaluation, w was set to 1000.

$$\hat{u}_i = \hat{u}_{i-1} * \frac{(w-1)}{w} + label_i \tag{3}$$

$$\hat{b} = \frac{\hat{u}_i}{w} \tag{4}$$

Algorithm 2: Active Stream Framework

Input : Autoencoder AE, Labeling budget B, budget window w, strategy(parameters), Change Detector D

Output: AE

$\hat{b} \leftarrow 0; \hat{u}_0 \leftarrow 0;$

if $\hat{b} < B$ *AND* `strategy(`*parameters*`)` $= 1$ **then**
  Update label estimate $\hat{u}_i$ (equation 3) where $label_i = 1$;
  AE $\leftarrow$ `partialFit(`AE, X_i, y_i`)`;
  $\hat{y}_i \leftarrow$ `predict(`AE, X_i`)`;
  `updateChangeDetector(`$D, y_i \neq \hat{y}_i$`)`;
  if AE_L **then**
    $\text{AE}_L \leftarrow$`partialFit(`$AE_L X_i, y_i$`)`;
    if `changeSignalled(`D`)` **then**
      Replace AE with AE_L;
    end
  else if `warningSignalled(`D`)` **then**
    Create new AE_L;
    $\text{AE}_L \leftarrow$`partialFit(`AE_L, X_i, y_i`)`;
else
  Update label estimate $\hat{u}_i$ (equation 3) where $label_i = 0$;
end
Update spending estimate $\hat{b}$ (equation 4);

3.3. Query Strategies

The AL query strategy is an important part of the framework as it determines whether or not the current data sample X_i should be labelled. In this work we evaluate random, and information-based distance from threshold , and variable distance from threshold strategies, as well as combining both random and uncertainty strategies into a split strategy. We propose a novel distrance from threshold strategy that is adapted to the non-probabilistic RE value.

Random Strategy

A *random* active learning strategy randomly selects a sample to label based on Bernoulli probability with a given budget B. The method implemented in this research is given in algorithm 3. The random strategy satisfies all three objectives of (Žliobaitė et al., 2013).

Algorithm 3: Random Strategy

Input : Labeling budget B
Output: label

```
p ← random(0,1);
if p ≤ B then
    label ← 1;
else
    label ← 0;
end
```

Reconstruction Error based Distance from Threshold

As discussed by Tharwat and Schenck (2023) uncertainty strategies can be based on least-confidence, margin, entropy, or more recently, partitioned Gaussian Process (Lee et al., 2023), all of which utilise the predicted probability from the classifier $P(y_c|X)$ (Žliobaitė et al., 2013; Sethi and Kantardzic, 2017; Shan et al., 2018). AE methods instead provide an RE value which requires a new approach in order to utilise an uncertainty strategy.

We propose a new *Reconstruction Error based Distance from Threshold*, whereby the RE squared difference from the anomaly threshold ϕ is used as a measure of uncertainty, equation 5, where d_i is the squared distance, assuming the hypothesis that the lower the difference compared to the average of the population, then the greater the uncertainty for the sample.

The difference is squared to make all values positive, resulting in a right-tailed distriburion. The absolute value $|\phi - RE_i|$ could also be utilised at a higher computational cost, and may provide more stability by removing the exponential effect of larger distances.

$$d_i = (\phi - RE_i)^2 \tag{5}$$

In order to accommodate changes in the data stream and avoid a scenario where the strategy stops learning due to high variance, a fading factor α was used to produce a fading average of differences d_{avg}, calculated using equation 6. This allowed for the more recent samples to have a greater bearing on the strategy outcome.

$$
\begin{aligned}
S_i &= d_i + \alpha * S_{i-1} \\
N_i &= 1 + \alpha * N_{i-1} \\
d_{avg} &= \frac{S_i}{N_i}
\end{aligned}
\tag{6}
$$

Using d_{avg} the fading standard deviation d_{std} of the stream is calculated using equation 7, where V_i is the fading sample variance.

$$
\begin{aligned}
V_i &= (d_i - d_{avg})^2 + \alpha * V_{i-1} \\
d_{std} &= \sqrt{\frac{V_i}{N_i}}
\end{aligned}
\tag{7}
$$

Finally, the strategy returns a labeling decision of 1 where $d_i < d_{avg} - d_{std}\theta$, equation 8, requiring a sample to be below the average by so many θ standard deviations, where θ is the confidence threshold. $\theta = 2$ should capture samples where the difference is the lowest 5% of all samples.

$$
\text{labeling} = \begin{cases} 1, & d_i < d_{avg} - d_{std}\theta \\ 0, & \text{otherwise} \end{cases}
\tag{8}
$$

The distance from threshold strategy is given in algorithm 4, whereby the autoencoder AE model is used to predict the RE for sample X_i, line 2, and the fading average and standard deviation of the difference from the anomaly threshold ϕ over the stream used to provide a label output of 0 or 1 based on equation 8, lines 3 to 8. On its own, an uncertainty-like strategy cannot satisfy all three active learning objectives as: the number of labeled samples will depend on the amount of uncertainty within the data stream and could vary above the intended budget, this is instead limited by line 2 of algorithm 2; only samples within the uncertainty margin are labeled, changes occurring outside of the margin will be missed; and change detection will be based on the distribution of uncertain samples (Žliobaitė et al., 2013). The strategy should reflect regions where real concept drift is occurring as higher uncertainty could reflect a change, resulting in faster adaptation times (Sethi and Kantardzic, 2017; Shan et al., 2018).

Algorithm 4: Reconstruction Error based Distance from Threshold

Input : Confidence θ, Fading Factor α, input X, autoencoder AE, Threshold ϕ

Output: label

$S_0 \leftarrow 0; N_0 \leftarrow 0; V_0 \leftarrow 0;$ label $\leftarrow 0;$

$\text{RE}_i \leftarrow$ `predictRE(`AE, X_i`)`;

Calculate difference d_i of RE_i from ϕ, using equation 5;

Calculate the fading average difference d_{avg}, using equation 6;

Calculate the fading standard deviation of differences d_{std} using equation 7;

if $d_i < d_{avg} - d_{std}\theta$ **then**

  label $\leftarrow 1$;

end

Variable Distance from Threshold Strategy

Variable Distance from Threshold is based on the distance from threshold strategy, but instead of using a fixed confidence θ, this is instead varied depending on the amount of labeling that is being requested from the strategy, so that more labels will increase the confidence and fewer will decrease to attenuate the labeling and better manage budget (Žliobaitė et al., 2013). This approach also has the benefit that it is not limited to a fixed labeling ceiling and can better utilise higher budgets to accurately identify concept drift (Shan et al., 2018). Similar to the uncertainty strategy this also does not satisfy all three requirements (Žliobaitė et al., 2013).

The strategy evaluated in this research is given in algorithm 5. Here higher θ will result in fewer labels as a higher confidence reflects a smaller proportion of samples below the average difference from ϕ, where $\theta = 2$ would be approximately 5% and $\theta = 3$ 1% of samples. To avoid scaling to ∞, the algorithm is bounded from 0 to 3, where 0 would be equivalent to all samples that are below the average difference. θ is adjusted in the step size $s \in (0, 1]$, with the recommended value of 0.01 used in the experiment (Žliobaitė et al., 2013; Shan et al., 2018).

Algorithm 5: Variable Distance from Threshold

Input : Confidence θ, Fading Factor α, X, autoencoder AE, Threshold ϕ, Step s

Output: label

```
label ← distanceFromThresholdStrategy(θ,α,X_i, AE, φ);
if label = 1 then
    if θ < 3.0 then
        θ ← θ(1 + s);
    end
else
    if θ > 0.0 then
        θ ← θ(1 − s);
    end
end
```

Split Strategy

The *split strategy*, combines the random and variable distance from threshold strategies to benefit from their respective strengths of accessing the entire stream distribution for change detection, and adapting to potential change in higher regions of uncertainty. Due to the incorporation of the random strategy, this also meets all three requirements of Žliobaitė et al. (2013).

Algorithm 6, was evaluated as a simplistic form of the strategy. The probability of an uncertain sample being selected by the random strategy is $P(\text{label}) = B$, where B is the selected budget, which is the same for all members of X, therefore the random strategy is checked first as this is the lowest time cost, and the variable distance from threshold strategy second if random strategy does not label.

Algorithm 6: Split Strategy

Input : Label Budget B, Confidence θ, Fading Factor α, X, autoencoder AE, Threshold ϕ, Step s

Output: label

label $\leftarrow$ 0;

if `randomStrategy(`B`)` $=$ *True* **then**

  label $\leftarrow$ 1;

else if `varDistanceFromThresholdStrategy(`θ,α,X_i, *AE*, ϕ,s`)` $=$ *True* **then**

  label $\leftarrow$ 1;

3.4. Split Active Learning Anomaly Detector (SALAD)

Table 3: Active Learning Method Comparison to Žliobaitė et al. AL Requirements.

AL Method	**Req. 1**	**Req. 2**	**Req. 3**	**Comments**
MD3 (Sethi and Kantardzic, 2017)	No	No	No	MD3 only compares margin distributions.
OALEnsemble (Shan et al., 2018)	No	Yes	Yes	No budget constraint, performs blind change adaptation.
AL for IDS (Dang, 2020)	No	No	No	Labelling occurs on instances with biggest probability change. No pro-active change detection.
Open-CNN AL (Zhang et al.)	No	No	No	Uncertainty strategy is used, budget is not constrained, no pro-active change detection.
SALAD (Proposed)	Yes	Yes	Yes	Proposed method utilises a split strategy to ensure coverage of whole distribution. Budget is constrained by a budget parameter.

The proposed Split Active Learning Anomaly Detector (SALAD) method is depicted in figure 1. This method reduces the labeling cost of the data stream to a fixed budget by adopting a split active learning strategy to determine which labels should be updated, satisfying the requirements of Žliobaitė et al. (2013). The output of the anomaly detector is monitored for real concept drift by a change detector (Gama et al., 2014). Where real concept drift

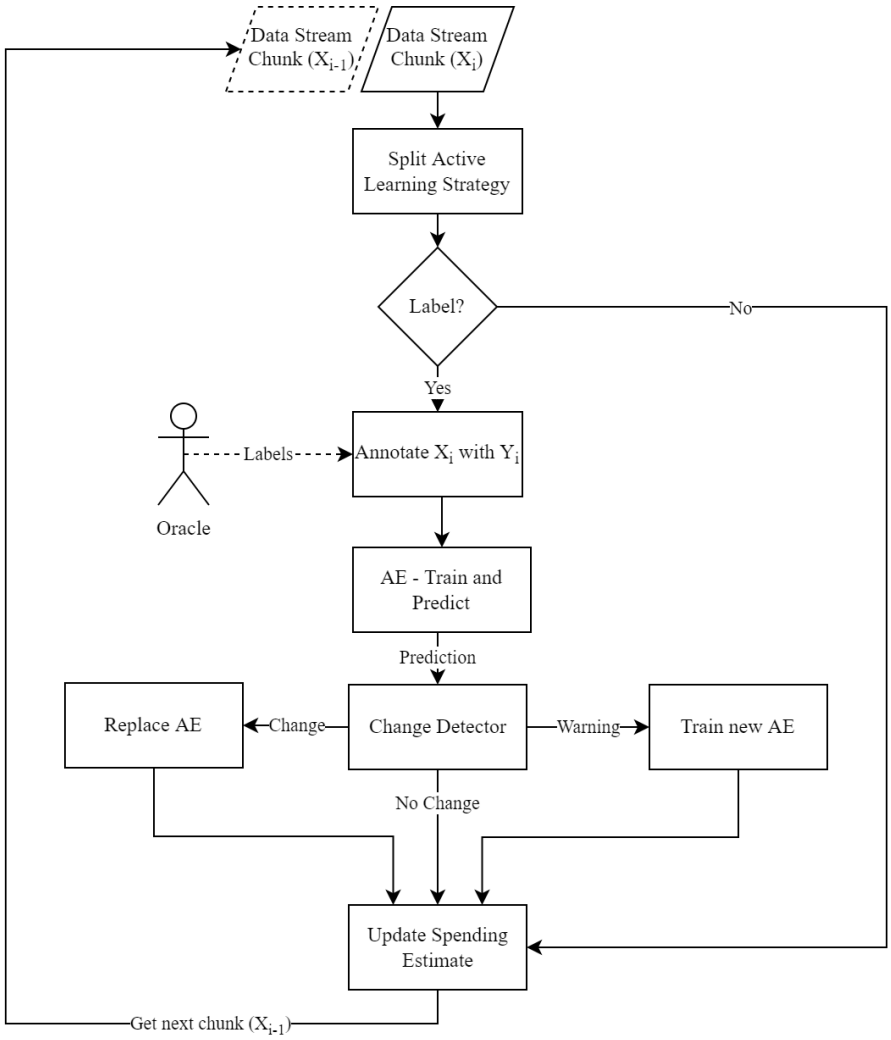


Figure 1: Split Active Learning Anomaly Detector

occurs, the current anomaly detector is replaced with a new one that has been trained on samples since a warning signal was produced. The result of this method is faster training of the anomaly detector and the ability to quickly adapt to changes occurring in the data stream. The proposed method is compared to other reviewed cyber AL methods in table 3.

4. Results

4.1. α and β Parameter Tuning

Fading Factor (α) was found for the SAT FF method by comparing the accuracy during prequential evaluation of different values of α (shown as FF in the figure), as given in Figure 2. $\alpha = 0.4$ demonstrated the marginally highest accuracy. Note that higher values of α, at 0.6 and 0.9, resulted in significantly lower accuracy, most likely due to slow reaction to changes in the data stream. $\alpha = 0.4$ is adopted as the comparable benchmark for all experiments in this paper.

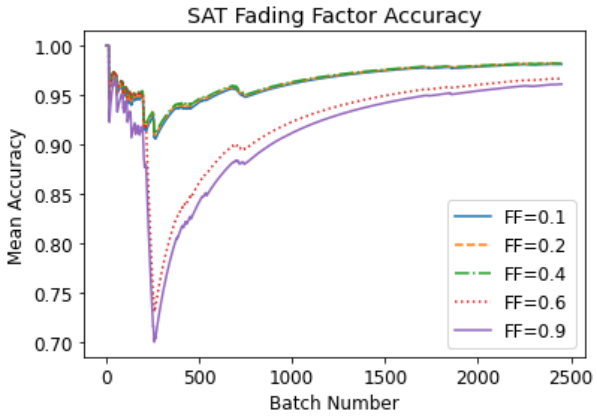


Figure 2: SAT FF Fading Factor (α) Accuracy

The mean accuracy produced by different values of β are given in figure 3. Here is can be seen that β values between 1.05 and 1.20 produced the highest accuracy, with 1.40 and 1.80 showing a significant degradation.

4.2. Adaptive Anomaly Threshold

The accuracy and F1-score of the Adaptive Anomaly Threshold method was compared to the SAT FF, HAT and NB algorithms. The parameter values for the autoencoder methods are given in table 4, where p represents the dropout probability; l is the number of hidden layers, h is the ratio of hidden units to visible units; *opt* is the optimiser used to train the network with n learning rate; β is the threshold sensitivity; α is the fading factor; and v is the step size. These parameters were found by exhaustive search. NB and HAT algorithms used the scikit-multiflow default parameters (Montiel et al., 2018).

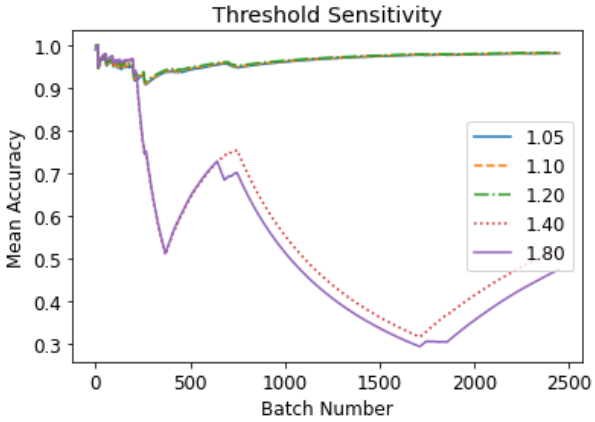


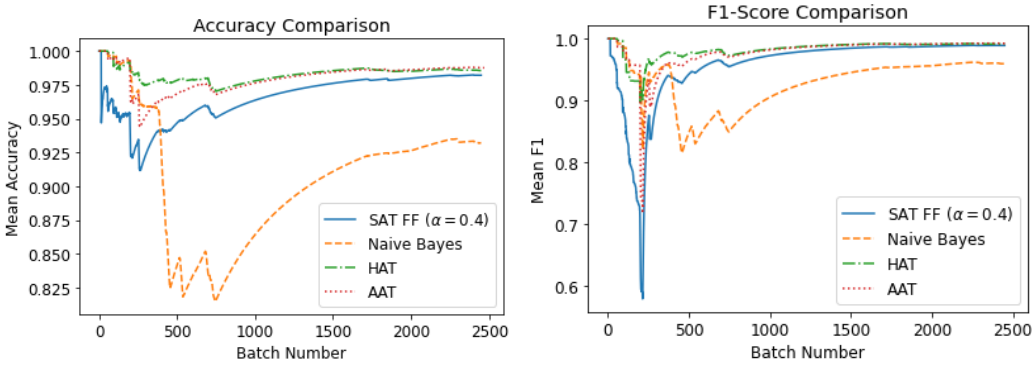
Figure 3: SAT FF Threshold Sensitivity (β) Accuracy

Table 4: Evaluation Parameters

Method	**Parameters**
Prequential Evaluation	batch size = 100, pretrain size = 10000
Autoencoder	$l = 1, p = 0.1, h = 0.6, opt = adagrad$ $(n = 0.01)$
SAT FF	$\beta = 1.1, v = 0.001, \alpha = 0.4$
Adaptive Anomaly Threshold	$\beta = 1.18, v = 0.001, \alpha = 0.4$

The accuracy and F1-scores with the KDD Cup 1999 data set are plotted in figure 4. SAT FF and AAT are close to HAT in terms of mean performance, with better kappa and F1 metrics when taken as an average across all batches, as shown in table 5. SAT FF and AAT were also significantly faster with a total running time (RT) of 14.04s and 19.18s, compared to 510.93s and 794.76s with NB and HAT, respectively. Note that running time will vary based on the underlying system performance and frameworks used, however the time of SAT FF is an order of magnitude better compared to both NB and HAT algorithms. Overall AAT returned the best mean accuracy and kappa results, an important metric for data stream learning. The accuracy results for both SAT FF and AAT (98.8%), outperform Naïve Threshold with Decay and SAT reported by Nixon et al. (2020), which achieved an accuracy of 95.4% and 98.0% respectively.

As demonstrated in our previous work (Nixon et al., 2019), the UNSW-



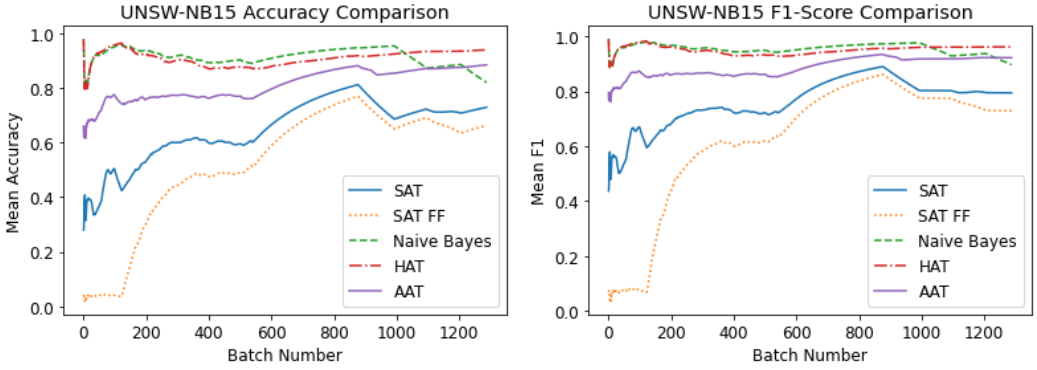
(a) Accuracy (b) F1-score

Figure 4: KDD Cup 1999 AAT, SAT FF, NB and HAT accuracy and F1-score

Table 5: KDD Cup 1999 AAT, SAT FF, NB and HAT Results

Algorithm	**Accuracy %** $\mu\pm$**SD**	**Kappa** $\mu\pm$**SD**	**F1-score** $\mu\pm$**SD**	**RT (s)**
AE AAT	**98.78±7.88**	**0.954±0.202**	0.802±0.395	19.18
AE SAT FF	98.16±8.65	**0.854±0.360**	0.812±0.387	**14.04**
NB	93.34±20.22	0.721±0.445	0.810±0.380	510.93
HAT	98.57±0.60	0.820±0.379	0.811±0.383	794.76

NB15 data set proved to be more challenging for online learning, requiring the number of network layers and dropout probability to be adjusted to better provide separation between normal and anomaly class distributions, with $l = 3$ and $p = 0.2$ being selected. The accuracy and F1-score results of the AAT method compared to SAT, SAT FF, NB and HAT are plotted in figure 5. Table 6 gives average accuracy of the SAT and SAT FF algorithms as 70.39% and 62.96%, respectively, which is considerably lower than that of NB and HAT. AAT wIth a 3 layer AE (AE AAT L3) returned the highest overall accuracy, compared to single layer (AE AAT L1) and SAT FF, although kappa was lower, demonstrating reduced confidence in the anomaly decision for all methods. The results show that AAT is able to provide near equivalent performance to NB and HAT methods with a significantly lower running time.



(a) Accuracy (b) F1-score

Figure 5: UNSW-NB15 AAT, SAT, SAT FF, NB and HAT accuracy and F1-score

Table 6: UNSW-NB15 AAT, SAT, SAT FF, NB and HAT Results

Algorithm	**Accuracy %** $\mu\pm$**SD**	**Kappa** $\mu\pm$**SD**	**F1-score** $\mu\pm$**SD**	**RT** (s)
AE AAT L3	**86.31±16.32**	0.298±0.411	**0.767±0.335**	18.55
AE AAT L1	81.20±24.65	**0.472±0.465**	0.714±0.366	17.3
AE SAT	70.39±32.71	0.364±0.443	0.613±0.390	12.14
AE SAT FF	62.96±38.95	0.420±0.458	0.528±0.418	**11.01**
NB	83.69±28.99	0.399±0.480	0.832±0.343	350.39
HAT	92.85±11.19	0.436±0.479	0.813±0.340	610.94

4.3. Active Stream Framework

4.3.1. Labeling Budget

The effects of the labeling budget was evaluated with the random strategy as this is the only strategy to maintain the sample distribution of the stream so as to not add any bias to the results. Budget B was evaluated at values of 0.2 (20%), 0.5 (50%) and 1.0 (100%). The results are given in table 7 and mean accuracy was plotted against the blind adaption AAT approach for comparison in figure 6. The greater the labeling budget, typically the higher the accuracy, kappa and F1-scores, the exception being UNSW-NB15 where $B = 0.5$ has a slightly higher accuracy and kappa. The difference in accuracy between 20% and 100% labels is 0.76% (KDD'99) and 2.69% (UNSW-NB15), demonstrating a small loss in performance for an 80% saving in labeling cost and approximate running time reduction of 54-62%; this reflects the results

Table 7: Random Strategy Budget Size: KDD'99 and UNSW-NB15 Comparison

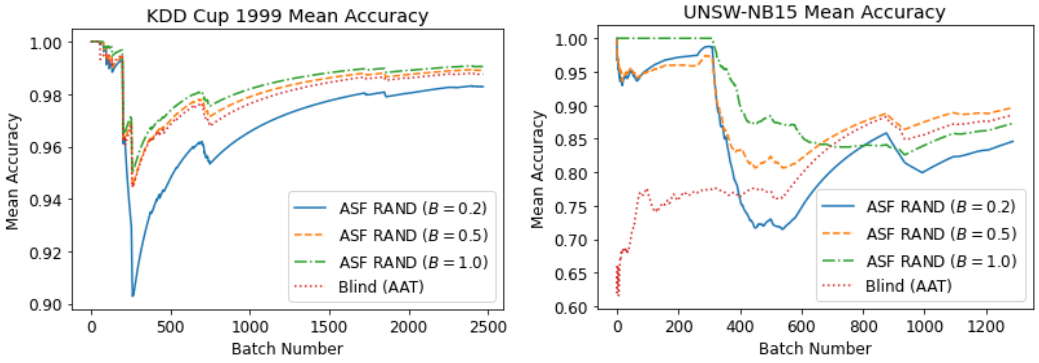
Strategy	B	**Accuracy %** $\mu\pm$**SD**	**Kappa** $\mu\pm$**SD**	**F1-score** $\mu\pm$**SD**	**RT (s)**
KDD Cup 1999					
Random	0.2	98.32±8.50	0.932±0.217	0.811±0.381	55.9
Random	0.5	98.94±7.27	0.956±0.182	0.821±0.376	85.8
Random	1.0	99.08±7.02	0.962±0.176	0.825±0.374	145.4
Blind	1.0	98.78±7.88	0.954±0.202	0.802±0.395	19.18
UNSW-NB15					
Random	0.2	87.07±19.48	0.598±0.376	0.752±0.350	55.5
Random	0.5	90.85±12.16	0.619±0.265	0.791±0.338	84.0
Random	1.0	89.76±12.74	0.549±0.431	0.793±0.334	121.2
Blind	1.0	86.31±16.32	0.298±0.411	0.767±0.335	18.55

of Žliobaitė et al. (Žliobaitė et al., 2013), where a small loss of accuracy was observed between a B of 100% and 10% when tested with a number of non-cyber data sets.

Comparing to the blind adaptation of previous experiments, whereby no active learning is used, a labeling budget of 0.5 achieved a higher accuracy and F1-score for half of the labeling cost on both data sets. ASF RAND 1.0 is equivalent to the blind approach with full labels, but with the addition of change detection, where average accuracy and F1-score were improved across both data sets, although they lower towards the end of the UNSW-NB15 stream as shown in figure 6b. Note the lower running time of the blind approach due to use of a chunk size of 100 vs 10 which influences the number of gradient updates and hence training time of the network.

4.3.2. Active Learning Query Strategies

The results of each active learning strategy with a budget of 0.2 (20%) are given in Table 8, with accuracy and F1-score for both data sets plotted in figure 7, where 'ASF R' is random, 'ASF U' is fixed RE based distance from threshold, 'ASF VARU' is variable distance from threshold, and 'ASF S' is the split strategy. Each strategy was executed 5 times with the average and standard deviation presented. The worst performing strategy was the fixed distance from threshold strategy, reflecting the results of Žliobaitė et al. (Žliobaitė et al., 2013), which was expected as the algorithm is biased only towards uncertain samples and cannot vary the amount of samples labeled, meaning that change occurring outside of the fixed margin will be missed. It



(a) KDD'99 Accuracy

(b) UNSW-NB15 Accuracy

Figure 6: Labeling Budget Accuracy Comparison for Random Strategy

is also possible that the RE-value of normal samples outside of the margin may increase as the AE is trained more on uncertain samples, leading to higher false positives and lower F1-score.

The split strategy, returned the best results across both data sets, combining random and variable distance from threshold strategies. Note that the total running time is between that of the random and variable distance from threshold strategies, indicating time complexity savings where uncertain samples were first selected by the random strategy. The Kappa of the split strategy was observed as 0.717 (table 8) for the UNSW-NB15 data set, this is much higher than the performance of the blind AAT, NB, HAT and other AL strategies, indicating a higher level of confidence in the anomaly decisions.

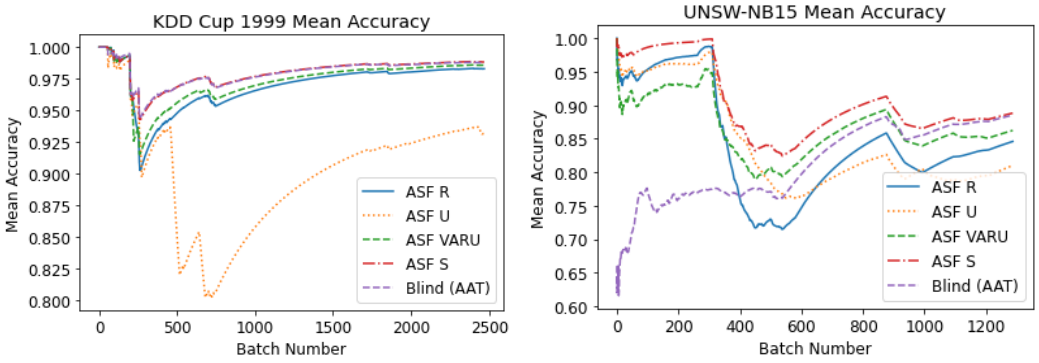
An absoulte value distance based RE based distance from threshold strategy was also tested as part of the split strategy, using $|\phi - RE_i|$, represented as 'Split Abs'. Accuracy and F1-score are slightly improved compared to the original squared difference approach but at a much larger running time penalty.

4.4. Change Detection Results

In order to determine the effectiveness of change detection, The SALAD method was evaluated against a blind version, SALAD NOCD, whereby no proactive change detection algorithm was used. The F1-score results for each data set are plotted in figure 8. The performance was similar for the KDD

Table 8: Active Learning Strategy Comparison (DfT = Distance from Threshold)

Strategy	**Accuracy %** $\mu\pm$**SD**	**Kappa** $\mu\pm$**SD**	**F1-score** $\mu\pm$**SD**	**RT (s)**
KDD Cup 1999				
Random	98.32±8.50	0.932±0.217	0.811±0.381	**55.9**
DfT	93.32±23.40	0.892±0.303	0.762±0.422	81.6
Var DfT	98.61±8.65	**0.951±0.194**	0.817±0.379	74.1
Split	**98.85±7.55**	0.947±0.199	**0.819±0.378**	69.6
UNSW-NB15				
Random	87.07±19.48	0.598±0.376	0.752±0.350	55.5
DfT	83.95±16.40	0.348±0.304	0.762±0.334	**53.8**
Var DfT	87.51±16.26	0.452±0.368	0.768±0.339	64.6
Split	90.88±14.96	**0.717±0.363**	0.791±0.343	63.3
Split Abs	**91.18±12.34**	0.617±0.345	**0.799±0.338**	89.5



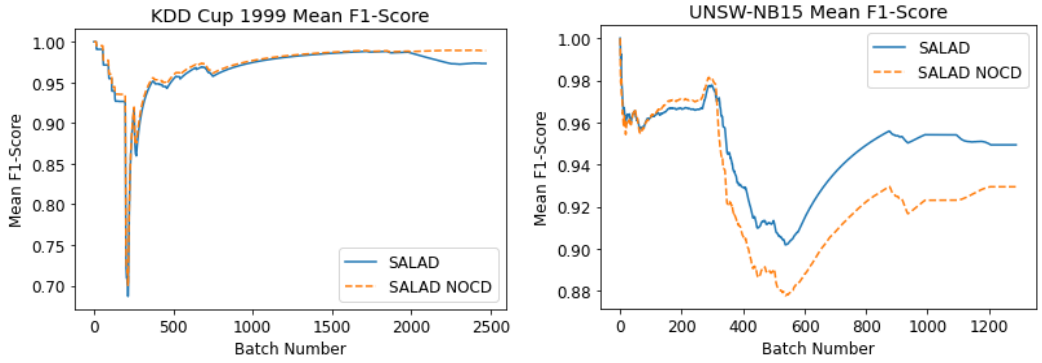
(a) KDD'99 Accuracy (b) UNSW-NB15 Accuracy

Figure 7: ASF Strategy Comparison, $B = 20\%$

Cup 1999 data set, but was lower for the blind method with UNSW-NB15, suggesting that the change detection method can add advantage depending on the nature of the data stream.

4.5. Attack Category Results

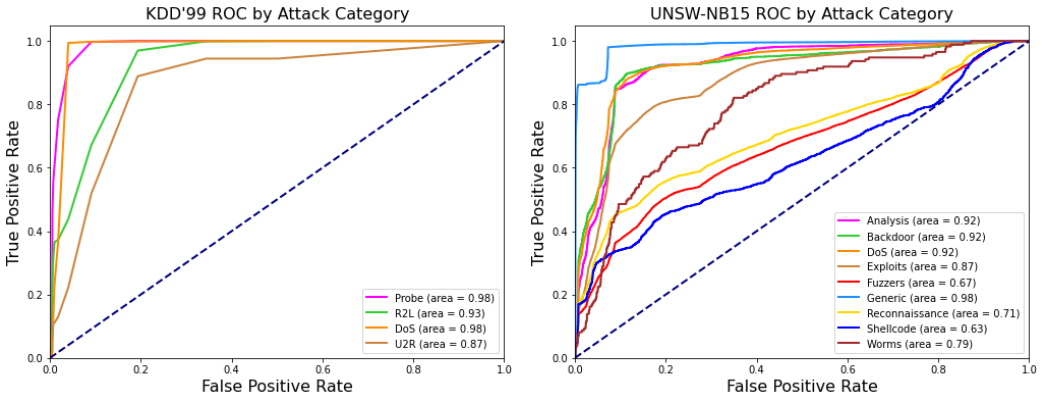
The ROC-AUC of known attack categories are plotted for both data sets in figure 9. The proposed method performed well in all categories for KDD Cup 1999, with U2R being the lowest performer. For the UNSW-NB15 data set, SALAD performed well for Exploits, DoS, Backdoor, Analysis and Generic attacks, but performed poorly for all other attack categories. This poor performance is most likely explained by the overlapping RE distribution



(a) KDD'99 F1-Score (b) UNSW-NB15 F1-Score

Figure 8: SALAD Change Detection vs No Change Detection F1-Score

for normal and anomaly data shown in figure 10 and further investigation would be required into features that would better separate individual classes.



(a) KDD'99 ROC-AUC (b) UNSW-NB15 ROC-AUC

Figure 9: Attack Category ROC-AUC Comparison

5. Discussion

This research evaluated online anomaly detection in the form of a prequential evaluation method whereby the model is first tested on the next sample or chunk in the stream before training. The adaptive anomaly threshold (AAT) was introduced as a stream-based novel hybrid of the naïve and

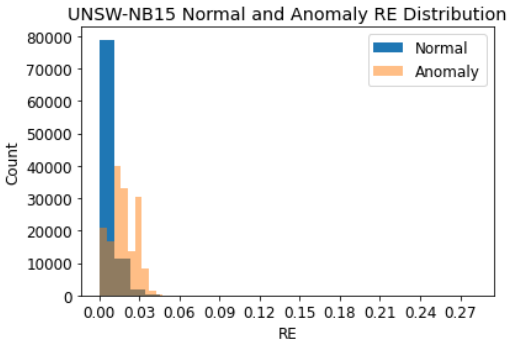


Figure 10: UNSW-NB15 RE Distribution

stochastic methods in order to better adapt to chunks of normal or anomaly samples based on initial observed accuracy. Overall AAT outperformed other methods and is a recommended contribution of this research to be explored further.

The results observed with the KDD'99 data set and AAT threshold method provide strong evidence that the hypothesis of effective anomaly detection for network data streams can be supported by the autoencoder method with both strong detection and run time performance compared to traditional methods. UNSW-NB15 results could be strengthened by further design choices.

The AAT method makes use of *blind* adaptation, whereby the model is trained on all labeled samples. This has the drawback of high cost due to full labels and slow adaptation times to change occurring in the data stream. The research further explored change detection and active learning strategies, as outlined by Žliobaitė et al. (2013), to further improve performance for a lower overall cost.

An ASF framework was implemented along with the random, distance from threshold, variable distance from threshold and split active learning strategies. A new Reconstruction Error based Distance from Threshold Strategy for AE was proposed, whereby the average RE difference from the threshold is used as a baseline to detect samples with high uncertainty, defined as being in the proportion of the population with the smallest difference, tuned by a confidence parameter.

The use of ASF demonstrated that better accuracy, kappa and F1-scores can be achieved for stream based anomaly threshold methods, compared to blind adaptation, with just 20% of the labeling cost, enabled by active

learning of the most important samples to accelerate the learning process (Žliobaitė et al., 2013). The results align to those presented by Žliobaitė et al. (2013), with a split strategy being recommended as this fulfils all three active learning requirements to maintain a fixed budget, access to all samples within the stream and preserve the distribution of incoming data for detecting changes. Unlike Žliobaitė et al. (2013), this research recommends inclusion of the uncertain samples with the change detection to improve per class performance.

Overall we have demonstrated that finding an optimal anomaly threshold for stream-based learning is possible using a fading factor based AAT method, and the labeling cost and performance of this method improved when combined with an AL approach, using a novel RE based uncertainty strategy. The use of ASF allows for change detection and re-training of the AE and anomaly threshold to adapt to real concept drift occurring in the normal sample space, demonstrated by enhanced performance when compared to the same approach with no pro-active change detection. We achieve near equivalent performance to the supervised online HAT method, at a greatly reduced running cost, making this a viable method for future stream-based semi-supervised applications.

6. Conclusion

The aim of this research was to explore semi-supervised online autoencoder methods for the task of anomaly intrusion detection on non-stationary network data streams, adapting to concept drift over time, with minimal labeling cost, by adopting an active learning change detection strategy. A unique contribution of this research was to compare a selection of anomaly threshold methods, proposing memory adaptations for data streams and a hybrid Adaptive Anomaly Threshold method which demonstrated superior performance. One of the more striking findings of the research is that the processing time of the autoencoder anomaly detector method is significantly lower when compared to traditional online learning techniques, making it well adjusted for high speed online network data streams, demonstrating an ability to detect an equivalent number of cyber attacks to traditional online learning methods, in a significantly reduced time frame. An area of future research would be to explore alternative threshold methods, such as clustering, which may allow for better identification of classes that overlap with normal samples and multi-label classification.

A further contribution of this research was to evaluate the autoencoder method with an Active Stream Framework, allowing the labeling cost of the data stream to be significantly reduced to a budget of 20%. A novel RE based variable distance from threshold strategy was proposed for autoencoders where the posterior probability is not available, instead tracking the distribution of sample RE distances from the anomaly threshold to determine uncertainty. An area of future research should be how to efficiently annotate samples, possibly by unsupervised clustering methods such as those demonstrated by Cataltepe et al. (2016).

Overall this research has demonstrated that the proposed Split Active Learning Anomaly Detector (SALAD) method can demonstrate high levels of performance with network data streams, which significantly reduced the labeling cost. The results are not perfect however, and it would be recommended to combine in a hybrid intrusion detection model whereby misuse detection is used before or after the anomaly detector to further identify classes, reduce false positives and better identify minority classes. Multi-label classification would be a further research area to expand on this work and provide additional context to detections.

References

Aktar, S., Yasin Nur, A., 2023. Towards DDoS attack detection using deep learning approach. Computers and Security 129, 103251. URL: https://doi.org/10.1016/j.cose.2023.103251, doi:10.1016/j.cose.2023.103251.

Autoencoder, T.b., Salahuddin, M.A., Pourahmadi, V., Alameddine, H.A., Bari, F., Boutaba, R., 2022. Chronos : DDoS Attack Detection Using 19, 627–641.

Aygun, R.C., Yavuz, A.G., 2017. Network anomaly detection with stochastically improved autoencoder based models, in: 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), IEEE. pp. 193–198.

Cataltepe, Z., Ekmekci, U., Cataltepe, T., Kelebek, I., 2016. Online feature selected semi-supervised decision trees for network intrusion detection, in: NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium, IEEE. pp. 1085–1088.

Catillo, M., Pecchia, A., Villano, U., 2023. CPS-GUARD: Intrusion detection for cyber-physical systems and IoT devices using outlier-aware deep autoencoders. Computers and Security 129, 103210. URL: https://doi.org/10.1016/j.cose.2023.103210, doi:10.1016/j.cose.2023.103210.

Chen, J., Sathe, S., Aggarwal, C., Turaga, D., 2017. Outlier detection with autoencoder ensembles, in: Proceedings of the 2017 SIAM International Conference on Data Mining, SIAM. pp. 90–98.

Chollet, F., et al., 2015. Keras. https://keras.io.

Dang, Q.V., 2020. Active learning for intrusion detection systems, in: IEEE Research, Innovation and Vision for the Future.

Fahy, C., Yang, S., Gongora, M., 2023. Scarcity of Labels in Non-Stationary Data Streams: A Survey. ACM Computing Surveys 55. doi:10.1145/3494832.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A., 2014. A survey on concept drift adaptation. ACM computing surveys (CSUR) 46, 44.

Gama, J., Medas, P., Castillo, G., Rodrigues, P., 2004. Learning with drift detection, in: Brazilian symposium on artificial intelligence, Springer. pp. 286–295.

Gama, J., Sebastião, R., Rodrigues, P.P., 2013. On evaluating stream learning algorithms. Machine Learning 90, 317–346.

Gomes, H.M., Grzenda, M., Mello, R., Read, J., Le Nguyen, M.H., Bifet, A., 2022. A Survey on Semi-supervised Learning for Delayed Partially Labelled Data Streams. ACM Computing Surveys 55. doi:10.1145/3523055, arXiv:2106.09170.

Kieu, T., Yang, B., Guo, C., Jensen, C.S., 2019. Outlier detection for time series with recurrent autoencoder ensembles, in: 28th international joint conference on artificial intelligence.

Lee, C., Wang, K., Wu, J., Cai, W., Yue, X., 2023. Partitioned Active Learning for Heterogeneous Systems. Journal of Computing and Information Science in Engineering 23, 1–27. doi:10.1115/1.4056567, arXiv:2105.08547.

Li, X., Chen, W., Zhang, Q., Wu, L., 2020. Building auto-encoder intrusion detection system based on random forest feature selection. Computers & Security , 101851.

Žliobaitė, I., Bifet, A., Pfahringer, B., Holmes, G., 2013. Active learning with drifting streaming data. IEEE transactions on neural networks and learning systems 25, 27–39.

Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A., 2018. Kitsune: an ensemble of autoencoders for online network intrusion detection. arXiv preprint arXiv:1802.09089 .

Mirza, A.H., Cosan, S., 2018. Computer network intrusion detection using sequential lstm neural networks autoencoders, in: 2018 26th Signal Processing and Communications Applications Conference (SIU), IEEE. pp. 1–4.

Montiel, J., Read, J., Bifet, A., Abdessalem, T., 2018. Scikit-multiflow: a multi-output streaming framework. The Journal of Machine Learning Research 19, 2915–2914.

Moustafa, N., Slay, J., 2015. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), in: 2015 military communications and information systems conference (MilCIS), IEEE. pp. 1–6.

Nicolau, M., McDermott, J., 2016. A hybrid autoencoder and density estimation model for anomaly detection, in: International Conference on Parallel Problem Solving from Nature, Springer. pp. 717–726.

Nixon, C., Sedky, M., Hassan, M., 2019. Practical application of machine learning based online intrusion detection to internet of things networks, in: 2019 IEEE Global Conference on Internet of Things (GCIoT), IEEE. pp. 1–5.

Nixon, C., Sedky, M., Hassan, M., 2020. Autoencoders: A low cost anomaly detection method for computer network data streams, in: Proceedings of the 2020 4th International Conference on Cloud and Big Data Computing, Association for Computing Machinery, New York, NY, USA. p. 58–62. URL: https://doi.org/10.1145/3416921.3416937, doi:10.1145/3416921.3416937.

Odiathevar, M., Seah, W.K., Frean, M., Valera, A., 2022. An Online Offline Framework for Anomaly Scoring and Detecting New Traffic in Network Streams. IEEE Transactions on Knowledge and Data Engineering 34, 5166–5181. doi:10.1109/TKDE.2021.3050400.

Ren, P., Xiao, Y., Chang, X., Huang, P.Y., Li, Z., Gupta, B.B., Chen, X., Wang, X., 2022. A Survey of Deep Active Learning. ACM Computing Surveys 54. doi:10.1145/3472291, arXiv:2009.00236.

Sethi, T.S., Kantardzic, M., 2017. On the reliable detection of concept drift from streaming unlabeled data. Expert Systems with Applications 82, 77–99. doi:10.1016/j.eswa.2017.04.008.

Shan, J., Zhang, H., Liu, W., Liu, Q., 2018. Online active learning ensemble framework for drifted data streams. IEEE transactions on neural networks and learning systems 30, 486–498.

Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A., 2009. A detailed analysis of the kdd cup 99 data set, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, IEEE. pp. 1–6.

Tharwat, A., Schenck, W., 2023. A Survey on Active Learning: State-of-the-Art, Practical Challenges and Research Directions. Mathematics 11. doi:10.3390/math11040820.

Vaiyapuri, T., Binbusayyis, A., 2020. Application of deep autoencoder as an one-class classifier for unsupervised network intrusion detection: a comparative evaluation. PeerJ Computer Science 6, 1–26. doi:10.7717/peerj-cs.327.

Zhang, Z., Zhang, Y., Niu, J., Guo, D., . Unknown network attack detection based on open-set recognition and active learning in drone network. Transactions on Emerging Telecommunications Technologies n/a, e4212. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4212, doi:https://doi.org/10.1002/ett.4212, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.4212.
i