2018 26th Mediterranean Conference on Control and Automation (MED)
Zadar, Croatia, June 19-22, 2018

Efficient Tracking of Statistical Properties of Data Streams With Rapid Changes

Hugo Lewi Hammer[1] and Anis Yazidi[1]

Abstract— **Many real-life dynamical systems change rapidly followed by almost stationary periods. In this paper, we consider streams of data with such rapidly changing behavior and investigate the problem of tracking their statistical properties in an online manner.**

The streaming estimator is accompanied with a second estimator, suitable to adjust to rapid changes in the data stream. When a statistically significant difference is observed between the two estimators, the current estimate jumps to a more suitable value. Such a tracking procedure have previously been suggested in the literature. However, our contribution lies in building the estimation procedure based on the difference between the stationary estimator and a Stochastic Learning Weak Estimator (SLWE). The SLWE estimator is known to be the state-of-the art approach to tracking properties of non-stationary environments and thus should be a better choice to detect changes in rapidly changing environments than the far more common sliding window based approaches.

Extensive simulation results demonstrate that our estimation procedure is easy to tune and performs very well. Further, the suggested estimator outperforms the popular and state-of-the-art estimator ADWIM2 with a clear margin.

I. Introduction

In many real-life applications, the assumption on the stationarity of the data does not hold and the true underlying parameter being estimated changes over time. The Stochastic Learning Weak Estimators (SLWE) are known to be the state-of-the-art approach for such an estimation problem [17], [27]. The SLWE enjoys a multiplicative update form that makes it superior to the state-of-the-art estimation approaches which are mainly of additive flavor. However, the right choice of the intrinsic parameter of the SLWE, λ, is still an open issue. The latter parameter controls the forgetting of old data and controls the ability of the scheme to adapt to changes in the environments. If the system changes rapidly the parameter should be chosen to rapidly forget the old stale data. On the other hand, if the environment is stabilizing, the rate of forgetting should decrease.

The SLWE has found numerous successful applications in the literature. Applications of the SLWE include adaptive classifiers for spam filtering [27], adaptive file encoding with nonstationary distributions [20], intrusion detection in computer networks [24], tracking shifts of languages in online discussions [23], learning user preferences under concept-shift [16], [26], fault-tolerant routing in Ad-hoc networks [15], digital content forensics for detecting illicit images [8], detection and tracking of malicious nodes in both Ad-hoc networks [18], vehicular mobile WiMAX networks [13], and optimizing firewall matching time via dynamic rule ordering [14]– to mention a few.

In many of such practical problems the dynamical system changes rapidly followed by periods where the system is almost stationary. Unfortunately, the SLWE is not well suited for such cases. By choosing a high value of λ, the estimator will rapidly adjust after an abrupt change, but on the other hand, it will result in a higher estimation uncertainty when the system stabilizes. By choosing a low value of λ, the estimation uncertainty will be low in the stationary parts, but on the other hand, the estimation procedure will suffer from adjusting too slowly after a rapid change.

In this paper, we suggest a computationally efficient estimation procedure for a dynamical system that contains both abrupt changes and stationary parts. The estimator combines an estimator that is suitable for the stationary parts together with an event detection procedure. When an abrupt change is detected, the estimator rapidly jumps to a more suitable estimate. The far most common event detection approach is to compare the properties of the data stream on a long term time window with a more short term time window [7]. In such window based approaches, each sample in the window is given an equal weight, but intuitively it is more reasonable to give more weight to the most recent data which is done by the SLWE. In this paper we therefore suggest to build the event detection procedure by comparing the estimate by the stationary estimator with an SLWE estimator. Through lightweight and subtle hypothesis testing mechanisms, we decide, in each iteration, if the stationary estimate should jump to new value (event detected) or not. Quite surprisingly, we have found only one other paper in the literature using the advantages of the SLWE for event detection, namely the paper by Ross et al. (2012) [19]. However, the focus of [19] is different from ours. While [19] focus on event detection, our focus is on *tracking of statistical properties* of the data streams with sudden rapid changes. Compared to [19], our suggested approach is simpler and better founded theoretically. We present the estimation procedure for the binomial distribution, but can be applied to other distributions as well.

II. Related Work and State-of-the-Art

In this Section we review related work. First, in Section II-A we will review legacy scheme for estimation under non-stationary environment. Then, in Section II-B we will review the different approaches for controlling the parameters of estimators operating in non-stationary environments.

[1]Hugo Lewi Hammer and Anis Yazidi are with Department of Computer Science at Oslo and Akershus University College of Applied Sciences, Pilestredet 35, N-0166 Oslo, Norway {hugo.hammer,anis.yazidi}@hioa.no

A. Estimation in Non-Stationary Environments

Traditionally available methods that cope with non-stationary distributions resort to the so-called *sliding window* approach, which is a limited-time variant of the well-known maximum likelihood estimator scheme. The latter model is useful for discounting stale data in data stream observations. Data samples arrive continuously and only the most recent observations are used to compute the current estimates. Any data occurring outside the current window is forgotten and replaced by the new data. The problem with using sliding windows is the following: If the time window is too small the corresponding estimates tend to be poor. As opposed to this, if time window is too large, the estimates prior to the change of the parameter have too much influence on the new estimates. Moreover, the observations during the entire window width must be maintained and updated during the process of estimation.

Apart from the sliding window approach, many other methods have been proposed, which deal with the problem of detecting change points during estimation. In general, there are two major competitive sequential change-point detection algorithms: Page's cumulative sum (CUSUM) [1] detection procedure, and the Shiryaev-Roberts-Pollak detection procedure. In [22], Shiryayev used a Bayesian approach to detect changes in the parameter's distribution, where the change points were assumed to obey a geometric distribution. CUSUM is motivated by a maximum likelihood ratio test for the hypothesis that a change occurred. Both approaches utilize the log-likelihood ratio for the hypotheses that the change occurred at the point, and that there is no change. Inherent limitations of CUSUM and the Shiryaev-Roberts-Pollak approaches for on-line implementation are the demanding computational and memory requirements. In contrast to the CUSUM and the Shiryaev−Roberts−Pollak approaches, our SLWE approach avoids the intensive computations of ratios.

B. Estimation using Adjustable parameters

Oommen and Rueda [17] presented a strategy by which the parameters of a binomial/multinomial distribution can be estimated when the underlying distribution is non-stationary. The method has been referred to as the Stochastic Learning Weak Estimator (SLWE), and is based on the principles of *continuous* stochastic Learning Automata (LA).

As opposed to the above-mentioned papers, in the last decades, a wide range of techniques for estimation in dynamically changing environments have appeared. We provide here a brief overview of representative *on-line* approaches particularly relevant to the family of techniques pioneered in the present paper. For a more detailed and comprehensive treatment of these, we refer the reader to recent surveys [7], [12], which include approaches based on *adaptive windowing*, *aging factors*, *instance selection* and *instance weighting*. Additionally, there is a distinction between *passive* approaches, which intrinsically adapt to changes, and *active* approaches that actively search for changes.

Gama *et al*. [7] presents a clear distinction between memory management and forgetting mechanisms. Adaptive windowing [25] works with the premise of growing the size the sliding window indefinitely until a change is detected via a change detection technique. In this situation, the size of the window is reset whenever a changed is detected.

Another strategy, which avoids windowing, involves using *aging* factors. In this approach, while training data spans all of the data points, each data point is assigned a weight that reflects its importance. Recent data points are prioritized, for instance using a fading factor that gradually reduces the influence of older data points [4]. The decay can be linear [11] or exponential [10]. By using weights combined with instance selection, referred to as instance weighting, one can take advantage of the ability of support vector machines to process weighted instances [10].

In the same perspective, the estimated error can be used for re-initializing the estimation as performed in [19]. In all brevity, changes are detected based on comparing sections of data, using statistical analysis to detect distributional changes, i.e., abrupt or gradual changes in the mean of the data points when compared with a baseline mean with a random noise component. One option is also to keep a reference window and compare recent windows with the reference window to detect changes [6]. This can, for example, be done based on comparing the probability distributions of the reference window and the recent window using Kullback-Leibler divergence [5], [21].

III. Stochastic Learning Weak Estimator

Let $X_1, X_2, X_3, \ldots$ represent a stream of independent and identically distributed Bernoulli stochastic variables with parameter p. That is

$$\begin{aligned} P(X_n = 0) &= 1 - p \\ P(X_n = 1) &= p \end{aligned} \tag{1}$$

for $n = 1, 2, 3, \ldots$.

We now want to estimate the parameter p from the stream of Bernoulli variables. Using the weak estimator, the estimate of p is updated by the following recursion

$$\begin{aligned} \hat{p}_1 &= X_1 \\ \hat{p}_n &= \lambda_n \hat{p}_{n-1} && \text{if } X_n = 0 \\ \hat{p}_n &= 1 - \lambda_n (1 - \hat{p}_{n-1}) && \text{if } X_n = 1 \end{aligned} \tag{2}$$

where $\hat{p}_n$ represents the estimate of p after the arrival of X_n and $\lambda_n,\ n = 1, 2, \ldots$ are constants between zero and one. The intuition is that if $X_n = 0$ we should reduce our current estimate of p (the probability of one) which is achieved by multiplying the current estimate of p by λ_n. On the other hand, if $X_n = 1$ we should reduce the estimate of $1 - p$ (the probability of zero) which gives

$$\begin{aligned} 1 - \hat{p}_n &= \lambda_n (1 - \hat{p}_{n-1}) \\ \hat{p}_n &= 1 - \lambda_n (1 - \hat{p}_{n-1}) \end{aligned}$$

which is equal to the last equation in (2).

The recursions in (2) can be written as follows

$$\hat{p}_n = X_n(1-\lambda_n(1-\hat{p}_{n-1})) + (1-X_n)\lambda_n\hat{p}_{n-1},\ n = 1, 2, \ldots$$

with $\lambda_1 = 0$. Using straight forward calculations this simplifies to

$$\hat{p}_n = \lambda_n\hat{p}_{n-1} + (1-\lambda_n)X_n \tag{3}$$

which can be recognized as the exponentially weighted moving average.

We can prove by induction that $\hat{p}_n$ is an unbiased estimator for p for every n as follows

$$\begin{aligned} E(\hat{p}_1) &= E(X_1) = p \quad \text{(recall } \lambda_1 \text{ is set to 0)} \\ E(\hat{p}_n) &= E(\lambda_n\hat{p}_{n-1} + (1-\lambda_n)X_n) \\ &= \lambda_n p + (1-\lambda_n)p \\ &= p \end{aligned}$$

The variance depends on the choice of the λ's. We look at two special cases.

λ *constant*: It can be proved that if we set all $\lambda_n = \lambda$, the limiting variance is given by [27]

$$\lim_{n\to\infty} \text{Var}\,(\hat{p}_n) = \frac{1-\lambda}{1+\lambda}p(1-p)$$

An advantage of the constant λ approach is that if the value of p is changing with time in the underlying Bernoulli data stream, the estimator will rapidly adjust to these changes [27]. A disadvantage is that if p is not changing, the variance of the estimator will have a lower limit and never reaches zero.

Sample mean: The sample mean is the maximum likelihood estimator of p and is the natural estimator to use if p is not changing with time. Let $\bar{p}_{n-1}$ denote the sample mean of the first $n-1$ Bernoulli variables from the stream

$$\bar{p}_{n-1} = \frac{1}{n-1}\sum_{i=1}^{n-1} X_i$$

When X_n arrives, the sample mean can be updated as follows

$$\bar{p}_n = \frac{n-1}{n}\bar{p}_{n-1} + \frac{1}{n}X_n \tag{4}$$

which is equivalent to (3) with $\lambda_n = (n-1)/n$. This means that the sample mean is a special case of the general recursion in (3). It is well known that $\lim_{n\to\infty} \text{Var}\,(\bar{p}_n) = 0$. A disadvantage of the sample mean is that if p is changing with time, the sample mean will become very slow at adjusting to these changes. On the other hand if p is not changing, the sample mean is the optimal estimator in the sense that no other unbiased estimators can achieve less variance.

IV. Estimation in a shifting environment

Suppose a situation where p is switching between different values with time. An example could be a news stream where the topic of the news stream suddenly changes due to different real life events. Another example could be a machine operated by different employees at different time periods each with its own error rate characterized by p. We assume that the instants in which p switches value are unknown.

For such systems a natural strategy would be to use the sample mean whenever the p is not changing, and a mechanism to "jump" fast towards a new estimate if the value of p has changed. In this paper we suggest a method that combines the sample mean and a weak estimator with constant λ. Ross et al. (2012) [19] is the only paper we have found in the literature that uses the same idea. Let $\hat{p}_n^\lambda$ and $\bar{p}_n$ denote the weak estimator with constant λ and the sample mean, respectively, after the arrival of X_n. If p switches value, $\hat{p}_n^\lambda$ will rapidly adjust to the new value of p, while minor changes will appear to $\bar{p}_n$. This can be used to build an efficient method to detect changes in p and "jump" to the new value of p. The key ingredient will be the distribution of the difference between the estimators $\hat{p}_n^\lambda - \bar{p}_n$. If p switches value we expect that $\hat{p}_n^\lambda - \bar{p}_n$ will be larger in absolute value than what we would expect if p remains constant. This can be used to build a statistical test if p has changed value or not. We start by presenting the expectation and variance of this distribution.

Theorem 1: Let $X_1, X_2, X_3, \ldots$ represent a stream of independent and identically distributed Bernoulli stochastic variables with parameter p. Further let $\hat{p}_n^\lambda$ and $\bar{p}_n$ denote the weak estimator with constant λ and the sample mean, respectively, after the arrival of X_n. Then

$$\text{E}\left(\hat{p}_n^\lambda - \bar{p}_n\right) = 0 \tag{5}$$

$$\begin{aligned} \text{Var}\left(\hat{p}_n^\lambda - \bar{p}_n\right) &= p(1-p)\times \\ &\left(\left(\frac{1}{n} - \lambda^{n-1}\right)^2 + \sum_{i=2}^{n}\left(\frac{1}{n} - (1-\lambda)\lambda^{n-i}\right)^2\right) \end{aligned} \tag{6}$$

For the sake of brevity, the proof is omitted.

Please note that $\text{Var}(\hat{p}_n^\lambda - \bar{p}_n)$ can be computed recursively such that all variances up to n can be computed in $O(n)$ time. The actual recursions are not shown, but are straightforward to compute from (6). Another appealing property is that the variance does not depend on the stream of observations and can be computed before the data streaming starts. This lays the foundations for building very efficient algorithms.

Theorem 1 stated the expectation and variance of the distribution of $\hat{p}_n^\lambda - \bar{p}_n$. Next we investigate other properties of the distribution. The difference $\hat{p}_n^\lambda - \bar{p}_n$ can be written as follows

$$\hat{p}_n^\lambda - \bar{p}_n = \left(\frac{1}{n} - \lambda^{n-1}\right)X_1 + \sum_{i=2}^{n}\left(\frac{1}{n} - (1-\lambda)\lambda^{n-i}\right)X_i$$

which is a weighted sum of the independent Bernoulli variables. If the sum satisfies the Lindeberg criterion (and thus the Lyapunov criterion), the sum will, according to the Central Limit Theorem, converge to a normal distribution [9]. The sum does not satisfy this criterion. A second option is to study the distribution of $\hat{p}_n^\lambda - \bar{p}_n$ by stochastic simulation which documents that $\hat{p}_n^\lambda - \bar{p}_n$ is very close to the normal distribution. For the sake of brevity, the details are omitted.

We then get the following test.

Theorem 2: Let $X_1, X_2, X_3, \ldots$ represent a stream of independent and identically distributed Bernoulli stochastic variables with parameter p. Further let z_α denote the α quantile of the standard normal distribution. Define the hypotheses

H_0: The underlying p has not changed value

H_1: The underlying p has changed value

Suppose that we decide to reject H_0 if

$$\frac{|\hat{p}_n^\lambda - \bar{p}_n|}{\sqrt{\text{Var}(\hat{p}_n^\lambda - \bar{p}_n)}} > z_{\alpha/2} \tag{7}$$

Then the probability of rejecting H_0 if H_0 is true is approximately α and the rejection rule (7) controls the type I error.

For the sake of brevity, the proof is omitted.

From Theorem 1 we see that $\text{Var}(\hat{p}_n^\lambda - \bar{p}_n)$ depends on p which of course is unknown. To perform the test above, a natural choice is to substitute p with the sample mean $\bar{p}_n$ since this is our best estimate of p under H_0.

The basic idea of our method is to estimate p using the sample mean, but perform occasional jumps if the test in Theorem 2 brings evidence that p has switched value. In the next section we describe how to perform the jump.

A. Performing a jump

Let $\tilde{p}_n$ denote the estimate using the sample mean with jumps method after the arrival of X_n. Further let $\tilde{\lambda}_n$ denote the value used for λ_n in the recursions in (3) to compute $\tilde{p}_n$. Assume now that the test in Theorem 2 brings evidence that p has switched value which means that the current estimate $\tilde{p}_n$ is not reliable (since it is based on the sample mean). Our currently best estimate of p thus is based on the weak estimator with constant λ. Thus we set $\tilde{p}_n$ equal to $\hat{p}_n^\lambda$.

To continue the update of the estimator $\tilde{p}_n$ after the jump, we also need to decide a new value for $\tilde{\lambda}_n$. Recall that by using $\lambda_n = (n-1)/n$ in (3), we get the sample mean. After the jump we assume that we only have $\tilde{n} = 1$ observation (we do not trust the previous observations) and thus use $\lambda_n = (1-1)/1 = 0$

Before the algorithm can be run, we also need to decide a value for α in the test proposed in Theorem 2. When we run the test, the probability of wrongly detecting a change in p, is approximately α. In practice we may run the test many times, for example every tenth iteration. If we run the test many times, the chance of wrongly detecting a change in p in some of these tests naturally will be larger then α. This refers to the multiple testing problem in the statistical literature, see e.g. [2]. A simple and much used approach is the Bonferroni correction where a significance level of α/M is used instead of α, where M is the number of tests. There are two challenges with applying this approach (and other standard corrections). First, we do not know the number of tests we need to run. Second, the Bonferroni correction assumes that all the tests are independent. In our case this is far from true, since two subsequent tests are based on almost the same data stream (only a few extra observation have been added since the last test) and the outcomes are highly correlated. Using the Bonferroni correction will result in a too low significance level, and the tests may never detect that p has changed. In practice, setting α to about 10^{-3} overall performs well and is, as expected, somewhere between standard significance levels (0.05) and Bonferroni corrected levels.

The resulting algorithm is shown in Algorithm 1.

Algorithm 1 The sample mean with jumps algorithm.

Input:

$X_1, X_2, X_3, \ldots$ //Stream of Bernoulli variables

λ

α

D //How often to perform the test in Theorem 2

N //Max number of iterations

Method:

```
1: ñ ← 0
2: p̂_1^λ ← X_1
3: p̃_1 ← X_1
4: for n ∈ 1, 2, ..., N do
5:    p̂_n^λ ← λ p̂_{n-1}^λ + (1 − λ) X_n
6:    ñ ← ñ + 1
7:    p̃_n ← ((ñ − 1)/ñ) p̃_{n-1} + (1/ñ) X_n
8:    if n mod D == 0 then
9:       if |p̂_n^λ − p̄_n| / sqrt(Var(p̂_n^λ − p̄_n)) > z_{α/2} then
10:         p̃_n ← p̂_n^λ
11:         ñ ← 1
12:      end if
13:   end if
14: end for
```

V. Experiments

In this Section we evaluate the methodology above. A myriad of methods exists to detect events in data streams, however very few apply a detection procedure as part of tracking statistical properties of a data stream in abruptly changing environments. A prominent exception is the popular ADWIN2 algorithm [3]. We thus compare the performance of our estimator with ADWIN2 for the binomial data stream.

We considered two different cases.

- Large changes: p switches between 0.2 and 0.8 every 600 iteration.
- Small changes: p switches between 0.4 and 0.6 every 600 iteration.

As argued in the previous section, a reasonable value for $\alpha = 10^{-3}$ and is what is used in the experiments below.

Figure 1 shows the performance of the jump algorithm ($\tilde{p}_n$) and the original SLWE with constant λ ($\hat{p}_n^\lambda$) for different values of the tuning parameter λ. Comparing the black and blue curves, we see that $\tilde{p}_n$ outperforms $\hat{p}_n^\lambda$ for all choices of the tuning parameters. For the small jumps case in the right panel, the difference in performance is fairly small for the optimal value of λ around 0.98. However in reality this value is unknown and it is far more important with an algorithm

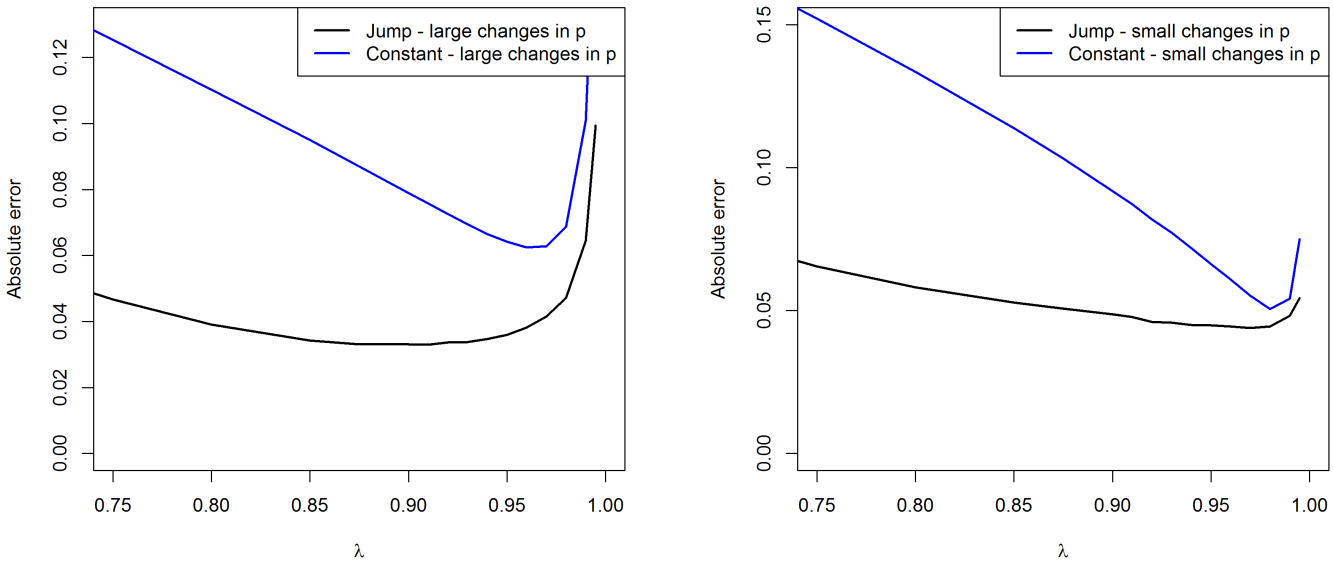

Fig. 1. Estimation error as a function of λ. The left and right panels refer to experiments where the changes in p are large and small, respectively. The black and blue curves refer to the estimators $\tilde{p}_n$ and $\hat{p}_n^\lambda$, respectively.

Large changes in p	Small changes in p
0.143	0.093

TABLE I
MEAN ESTIMATION ERROR IN ABSOLUTE VALUE FOR THE ADWIN2 ALGORITHM WHEN TRACKING p.

where the performance is robust with respect to choice of the tuning parameter. We see that the suggested jump algorithm ($\tilde{p}_n$) is far more robust than the original SLWE estimator ($\hat{p}_n^\lambda$).

For the large jumps case, we observe that for $\hat{p}_n^\lambda$ an optimal value of λ is about 0.96. We also observe that the optimal value for λ for the jump estimator is about 0.9. Recall that this is the λ we should choose for the weak estimator with constant λ that runs in parallel with the sample mean. This difference may come as a surprise, but remember that the purpose of the weak estimator with constant λ is different in these two cases. For $\hat{p}_n^\lambda$ (original SLWE) we chose λ to minimize the estimation error. For the jump estimator, we chose λ to detect changes in p as fast as possible to rapidly perform a jump.

Table I shows the performance of the popular event detection and tracking algorithm ADWIN2. By comparing the black lines of Figure 1 and Table I, we see that our jump estimator ($\tilde{p}_n$) outperforms ADWIN2 for *all* choices of the tuning parameter λ and with a clear margin for almost any choice of the tuning parameter.

VI. CLOSING REMARKS

In this paper we have constructed an estimation procedure that combines the strengths of a weak estimator with constant λ and and decreasing λ (sample mean). We have developed a hypothesis test procedure to rapidly detect a change in the underlying p. Further we have proposed an efficient procedure to jump to a new estimate when a change is detected. The experiments show that the procedure efficiently detects changes in the underlying distribution and outperforms the original SLWE with constant λ and the ADWIN2 algoritm with a large margin.

The experiments also showed that the performance of the jump estimator $\tilde{p}_n$ is less sensitive to the choice of λ compared to the SLWE with constant λ. Said in another way, the jump estimator performs well for a large range of different choices of λ while the SLWE with constant λ, $\hat{p}_n^\lambda$, performed well only for a small range of choices for λ. This is a very attractive property of the jump estimator since in practical situations we do not know what is an optimal value for λ.

A potential direction for future research is to develop a procedure to automatically adjust the value of the tuning parameter λ depending on the properties of the data stream. A starting point in this direction could be to follow the ideas on how the ADWIN2 algorithm adjusts the window size [3].

REFERENCES

[1] Michle Basseville and Igor V. Nikiforov. *Detection of abrupt changes: theory and application*. Prentice-Hall, Inc., 1993.

[2] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300, 1995.

[3] Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 443–448. SIAM, 2007.

[4] Edith Cohen and Martin Strauss. Maintaining time-decaying stream aggregates. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–233. ACM, 2003.

[5] Tamraparni Dasu, Shankar Krishnan, Suresh Venkatasubramanian, and Ke Yi. An information-theoretic approach to detecting changes in multi-dimensional data streams. In *In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications*. Citeseer, 2006.

[6] Anton Dries and Ulrich Rückert. Adaptive concept drift detection. *Stat. Anal. Data Min.*, 2(5):311–327, December 2009.

[7] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4):44:1–44:37, March 2014.

[8] Amin Ibrahim and Miguel Vargas Martin. Detecting and preventing the electronic transmission of illicit images and its network performance. In *International Conference on Digital Forensics and Cyber Crime*, pages 139–150. Springer, 2009.

[9] A. F. Karr. *Probability*. Springer, New York, 2012.

[10] Ralf Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intell. Data Anal.*, 8(3):281–300, August 2004.

[11] Ivan Koychev. Gradual forgetting for adaptation to concept drift. Proceedings of ECAI 2000 Workshop on Current Issues in Spatio-Temporal Reasoning,, 2000.

[12] Pallavi Kulkarni and Roshani Ade. Incremental learning from unbalanced data with concept class, concept drift and missing features: A review. *International Journal of Data Mining and Knowledge Management Process (IJDKP)*, 4(6):15–29, November 2014.

[13] Sudip Misra, Nayan Ranjan Kapri, and Bernd E Wolfinger. Selfishness-aware target tracking in vehicular mobile wimax networks. *Telecommunication Systems*, 58(4):313–328, 2015.

[14] Ratish Mohan, Anis Yazidi, Boning Feng, and B John Oommen. Dynamic ordering of firewall rules using a novel swapping window-based paradigm. In *Proceedings of the 6th International Conference on Communication and Network Security*, pages 11–20. ACM, 2016.

[15] B. J. Oommen and S. Misra. Fault-tolerant routing in adversarial mobile ad hoc networks: an efficient route estimation scheme for non-stationary environments. *Telecommunication Systems*, 44:159–169, 2010.

[16] B. J. Oommen, A. Yazidi, and O-C. Granmo. An adaptive approach to learning the preferences of users in a social network using weak estimators. *Journal of Information Processing Systems*, 8(2), 2012.

[17] B. John Oommen and Luis Rueda. Stochastic learning-based weak estimation of multinomial random variables and its applications to pattern recognition in non-stationary environments. *Pattern Recogn.*, 39(3):328–341, 2006.

[18] Nasser-Eddine Rikli and Aljawharah Alnasser. Lightweight trust model for the detection of concealed malicious nodes in sparse wireless ad hoc networks. *International Journal of Distributed Sensor Networks*, 12(7):1550147716657246, 2016.

[19] Gordon J. Ross, Niall M. Adams, Dimitris K. Tasoulis, and David J. Hand. Exponentially weighted moving average charts for detecting concept drift. *Pattern Recognition Letters*, 33(2):191 – 198, 2012.

[20] L. Rueda and B. John Oommen. Stochastic automata-based estimators for adaptively compressing files with nonstationary distributions. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 36(5):1196 –1200, October 2006.

[21] Raquel Sebastião and João Gama. Change detection in learning histograms from data streams. In *Proceedings of the Aritficial Intelligence 13th Portuguese Conference on Progress in Artificial Intelligence*, EPIA'07, pages 112–123, Berlin, Heidelberg, 2007. Springer-Verlag.

[22] Albert Nikolaevich Shiryayev. *Optimal Stopping Rules*. Springer, 1978.

[23] A. Stensby, B. J. Oommen, and O-C. Granmo. The use of weak estimators to achieve language detection and tracking in multilingual documents. *International Journal of Pattern Recognition and Artificial Intelligence*, 27(04):1350011, 2013.

[24] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blazek, and Hongjoong Kim. A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods. *IEEE Transactions on Signal Processing*, 54:3372–3382, September 2006.

[25] Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[26] A. Yazidi, O-C. Granmo, B. J. Oommen, M. Gerdes, and F. Reichert. A user-centric approach for personalized service provisioning in pervasive environments. *Wireless Personal Communications*, 61(3):543–566, 2011.

[27] Justin Zhan, B. John Oommen, and Johanna Crisostomo. Anomaly detection in dynamic systems using weak estimators. *ACM Trans. Internet Technol.*, 11:3:1–3:16, July 2011.