Applying Radial Basis Function Networks and Markov Chains for on-line detection of concept drift in non-stationary environments

Ruivaldo Neto, Adrien Brilhault, Ricardo Rios

Salvador, Brazil

Abstract

The amount of data produced by computer systems has grown sharply in recent decades, and a significant part of it is generated as uninterrupted and potentially infinite sequences known as data streams. Generally, these streams are produced by non-stationary environments, in which the data distribution can change over time, possibly deteriorating the system performance. In the literature, this phenomenon is named concept drift. Nevertheless, most drift detection methods are unsuited for non-stationary environments with data streams. These algorithms usually require the correct labeling of data - infeasible in these settings - or do not match the strict response time and resource usage restrictions inherent to scenarios with data streams. In an attempt to mitigate the aforementioned problem, this paper proposes a novel proactive method for on-line drift detection, called RBFChain. The proposed method relies on Radial Basis Function Networks implicit clustering property and uses Markov Chains to model the drifts transitions. To assess the proposed method as a viable concept drift detector, an analysis of sensitivity, accuracy, and noise tolerance was performed using synthetic datasets, and results were compared to the most established algorithms in the literature. Furthermore, the algorithm was applied to the real-world problem of eye-tracking. A critical issue for different areas of knowledge, since many behavioral experiments use eye-tracking information as a relevant analysis factor. Experimental results suggest that RBFChain is statistically better or equivalent to other detectors as it offers greater or equal overall classification accuracy. Also, the technique demonstrated good performance in the eye-tracking problem, being able to identify fixations and saccades in real-time with precision comparable to state of the art.

Keywords: Concept Drift, Drift Detection, Eye tracking

1. Introduction

In recent years, the volume of data produced by computer systems has grown dramatically. Technological advances favored this growth, such as the pervasiveness of mobile devices, the popularization of social networks, and the expansion of the internet of things [1].

A significant portion of this data is produced in the form of uninterrupted and potentially infinite sequences [2]. In literature, sequences with these characteristics are called data streams. These streams are present in various fields of application, such as financial market monitoring [3], road traffic monitoring [4], telecom network management [5], real-time sentiment analysis [6] and intruder prevention and identification systems [7].

Most of the environments that produce data streams are non-stationary. That is, the joint probability distribution changes arbitrarily over time, such as a switch in the conditional probability distribution on a classification problem, or a change of some moment (such as mean and variance) on a time series forecasting problem [8]. Systems applied to these environments may be unable to adapt to the new information, hence dramatically deteriorating their performance. This phenomenon is known as concept drift [9].

Still, most drift detection methods are unsuitable for non-stationary environments with data streams. These methods usually require the correct labeling of data - impracticable in these contexts - or do not meet the severe response time and resource usage restrictions inherent to contexts with data streams.

This paper proposes a novel proactive method for on-line drift detection, called RBFChain. The proposed algorithm is based on Radial Basis Function Networks implicit clustering property and employs Markov Chains to model the drifts transitions. To validate the proposed method as a viable concept drift detector, an examination of sensitivity, accuracy, and noise tolerance was performed using synthetic datasets, and results were compared to the most established algorithms in the literature.

Moreover, the algorithm was also applied to the real-world problem of eye-tracking. A problem with impact in different areas of knowledge, since many behavioral experiments use eye-tracking information as a relevant analysis factor.

Experimental results suggest that RBFChain is statistically better or equivalent to other detectors as it offers greater or equal overall classification accuracy in most situations. Also, the technique is applicable to eye-tracking problems as it was able to identify fixations and saccades in real-time with precision comparable to state of the art.

The rest of the paper is organized as follows: Section 2 describes the concept drift phenomenon and the main detection techniques; Section 3 presents the eye-tracking problem; Section 4 describes the RBFChain algorithm and its pseudo-code; Section 5 shows the configuration and results for the experiment with synthetic datasets; Section 6 presents the configuration and results for the experiment with the eye-tracking problem; and, finally, Section 7 provides conclusions and discusses future work.

2. Concept Drift

Non-stationary environments generate most real-world problems data [9]. In these environments, the joint probability distribution can change over time, such as a switch in the conditional probability distribution on a classification problem, or a change of some moment (such as mean and variance) on a time series forecasting problem [8].

Systems applied to these environments may be unable to adapt to the new information, hence dramatically deteriorating its performance. In the literature, this phenomenon is called concept drift.

Bayesian Theory is commonly used as a background to define concept drift formally [10]: consider the posterior probability of a sample x belonging to a class y, a concept drift happens when this probability changes over time, that is, $Pt + 1(y|x) \neq Pt(y|x)$. In a supervised learning scenario, this can be interpreted as when the relationship between the input data and the target variable change over time.

According to [8, 9], concept drifts can occur in four main patterns:

- **Abrupt:** occurs when a concept A switches abruptly to another concept B.
- **Gradual:** occurs when a concept A is being exchanged for the B concept gradually. In this case, while there is no definitive change from concept A to concept B, occurrences of B become more frequent, while fewer events of A are observed.

- **Incremental:** occurs when a concept A is being exchanged for B through intermediate concepts. These concepts differ little from its predecessor and successor. So changes are noticeable only in the long run.

- **Recurrent:** occurs when a previously active concept reappears after a certain period. However, this can not be understood as a periodic seasonality.

Figure 1 demonstrates these patterns:

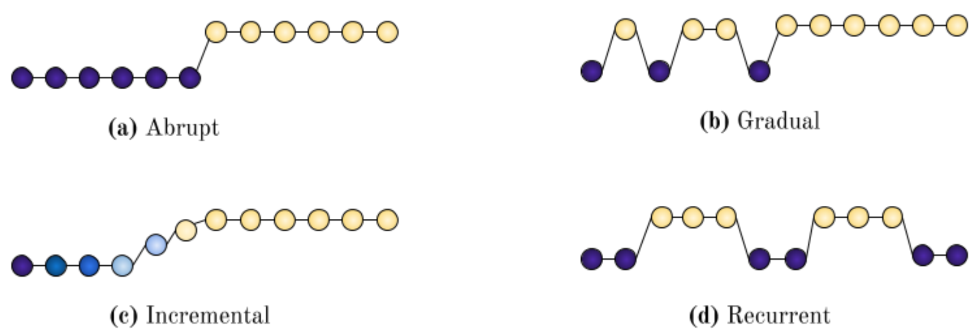

Figure 1: Concept Drift Patterns

Algorithms for detecting concept drift characterize and quantify concept drifts through the delimitation of the moments or time intervals in which changes occur [11]. These algorithms fall into two categories, according to the need for data labeling [12]:

- **Explicit Algorithms/Supervised**: These methods adopt a passive approach, as they depend on the correct labeling of the data to act. The model performance is monitored continuously, and drifts are detected when its performance starts to deteriorate, reaching a threshold.

- **Implicit Algorithms/Unsupervised:** These algorithms take a proactive approach and are independent of correct data labeling. Concept drifts are detected through the analysis of incoming data or indicators produced by the applied learning techniques. Although they are more prone to false alarms, they are an alternative to scenarios where obtaining labels is expensive, time-consuming or unviable. Also, this approach can lead to better results, since it is possible to refit the model or adjust the data, before the deterioration of the predictions.

The algorithm proposed in this paper classifies itself as an unsupervised algorithm and adopts a proactive approach. Briefly, its operation can be described: The Radial Basis Function Networks continuously cluster all incoming data. Changes in the generated cluster (a different center is activated) reflect in a Markov Chain, which keeps an online model of the possible system transitions and its probabilities. Drifts are triggered when the transition probability reaches a parametric threshold.

To assess the proposed method as a viable concept drift detector, an analysis of sensitivity, accuracy, and noise tolerance was performed using synthetic datasets. Moreover, results were compared to the most established algorithms in the literature, demonstrating the competitivity of the method.

3. Eye-tracking

Visual perception involves six types of eye movements [13], among which fixations and saccades are the most relevant. During fixation, the eye is kept relatively stable on an area of interest (AOI). In contrast, saccades are fast eye movements enabling the fovea to fixate different regions of the scene [14]. Thus, the process of looking at a scene can be represented by a sequence of fixations and saccades, the so-called visual scan path. Research on scan path analysis and visual perception has benefited from the recent development of eye trackers. Todays eye-tracking systems allow a precise recording of eye movements at high sampling rates, thus enabling a detailed analysis of the viewing behavior.

Despite recent advances, reliable automated clustering of eye movements is still challenging, even more so in dynamic scenarios. In many applications, e.g., human-computer gaze-based interaction, driving assistance systems, online adaptation of digital content based on gaze analysis, the identification of fixations and saccades has to occur in an online fashion. There is a wide variety of methods for the online analysis of eye-tracking data and the recognition of fixations and saccades. However, only a few of them are suited for online applicability to dynamic scenes. Such methods have to quickly adapt not only to the individual viewing behavior but also to the changes occurring in the viewing scene. This small group of highly promising methods is based on probabilistic formalizations, e.g., as Markov Models [15, 16], Bayesian Mixture Models [17], etc.

Prior techniques for the automated recognition of different types of eye movements from eye-tracking data fall into two main categories: (i) threshold-based methods, where the distinction of fixations from saccades is based on dispersion, velocity, or acceleration thresholds, and (ii) probabilistic methods. These groups of techniques will be briefly discussed in the following.

Threshold-based methods distinguish between fixations and saccades based on the assumption that the distances, velocities, or accelerations occurring between subsequent fixations differ from those occurring between saccades. The goal then is to identify a threshold based on which saccades can be reliably distinguished from fixations.

When distance thresholds are used, fixation clusters are usually identified by searching for data points that are close enough to each other (i.e., below the established threshold) within a predefined time window [18]. A representative of this group, is the Dispersion Threshold Identification (I-DT) algorithm [15]. Other similar approaches differ mainly in the way the threshold is calculated [19, 20].

Other algorithms in this realm are based on the computation of Minimum Spanning Trees (MST). In [15] an MST is built on the eye-tracking points within a temporal window of predefined length. An edge (i.e., representing the distance between two points) is classified as a saccade if its length is significantly larger than the lengths of neighboring edges, which have been previously classified as distances between fixations. Yet other methods employ smart clustering algorithms, e.g., [21, 22] but have serious limitations concerning their applicability to dynamic online scenarios, since, in such scenarios, the cluster properties for fixations and saccades show high variability.

Methods that are based on velocity or acceleration thresholds work similarly. A representative of this group is the Velocity-Threshold Identification (I-VT) algorithm, where a point is identified as a saccade point, if the implicit velocity along the distance from the previous data point to that point exceeds a predefined threshold. Otherwise the data point is assigned to a fixation cluster [15].

In summary, the major drawback of threshold-based methods is that they rely on thresholds that have to be empirically adjusted to the individual viewing behavior, the viewing area, and the specific task. Each of these parameters can have significant influence on the classification result [16, 15].

For this reason and because of the fact that the viewing behavior is strongly physically and physiologically-dependent, such methods are not reliable, especially when real-time analysis of eye-tracking data is needed.

Probabilistic methods are built on soft decision rules, which are formalized as probabilities, e.g., the probability of a data point being a saccade given the previous observations. The probabilities and thus, the decisions are adjusted to the observations.

One of the most prominent probabilistic methods applied to the identification of fixations and saccades is the Hidden Markov Model (HMM). An HMM is a simple dynamic Bayesian network with variables representing values from a discrete state and observation space. The state of a variable represents the class of the current observation. It is only dependent on the state (i.e., class of the previous observation). Because of this sequential nature, such models are a popular choice for the analysis of successively arising data points (i.e., observations). For the detection of fixations and saccades from eye data, HMMs have been used with velocity observations between successive data points, thus allowing the adaptation of the model to the physiological viewing behavior [15]. In the model of [15] (coined I-HMM), the two states used represent discretized velocity distributions over fixations and saccades. Transition probabilities between the states represent the probability of the current sample belonging to a fixation cluster or a saccade, given the previous state [18]. Due to the above probabilistic representation, no thresholds are needed. The I-HMM is reported to outperform fixed-threshold methods, such as I-VT [15]. In summary, the sequential, dynamic, and probabilistic nature of HMMs makes them an adequate choice for data arising in an online fashion and containing variability in its features.

Probabilistic mixture models, such as the Bayesian Mixture Model (BMM) presented in [17], build on the assumption that the observed data is generated from a mixture of unknown density distributions. The goal is to estimate the parameters of these distributions based on observed data points and to derive the most probable distribution that might have generated a given data point.

The algorithm presented in [17] could distinguish between fixations and saccades in an online fashion, only by considering the Euclidean distances between subsequent data points. The underlying model is based on the assumption that distances between subsequent fixation points will, in general, be shorter than distances between subsequent saccade points; that is, distances between subsequent fixation points would be generated from a specific

Gaussian distribution and those between subsequent saccade points from another. This intuition was modeled by a Bayesian Online Mixture Model. The benefit of the Bayesian formalization of the mixture model is that the parameters of the two distributions are updated and learned in an online fashion as more and more data is observed. For every new data point, the prior probabilities are replaced by the latest estimates. For practical purposes, this means that for every new user the algorithm needs a relatively small number of data points to adjust to that user and learn user- or scene-dependent parameters.

In summary, probabilistic methods come with three main advantages over threshold-based ones:

1. No fixed thresholds are needed. Instead, the parameters of the model (e.g., state transition probabilities, label emission probabilities, and other settings) are learned from labeled data.
2. Both HMMs and BMMs can adapt to the individual (i.e., physiological) viewing behavior of a subject and the specific task.
3. Given the dynamic nature of the underlying models, the methods are naturally suited for data arising in an online fashion, such as eye-tracking data.

4. RBFChain algorithm

This section details the RBFChain implementation. However, before describing the proposed method, it is significant to present the main applied concepts of Radial Base Function Networks and Markov Chains.

4.1. Radial Basis Function Networks (RBFN)

Radial Basis Function Networks (RBFN) are used in various disciplines with a reasonable degree of success. The broad applicability is a result of their excellent ability to make function approximation, especially when the relationships among the variables of interest are nonlinear [23].

A radial basis function network is a type of artificial neural network (ANN), and most neural networks are known to be useful in modeling complex and nonlinear relationships. An RBFN has advantages in specific applications in that for a given parameter set, RBFN networks do not require an iterative procedure to learn the model. Iterative learning for most ANN types is computationally expensive and vulnerable to the local minima problem.

The topology of an RBFN is given in Fig. 2 as a multiple input single output feedforward network. Assume that there are n input variables labeled from x_1 to x_n. The network receives input samples as vectors $x = (x_1, x_2, \ldots, x_n)$ of size $1 \times n$. The initial layer is only a buffer that feeds the input values to the intermediate layer, which is called the hidden layer. There are n_h processing elements in the hidden layer. Each processing element in the hidden layer processes the input vector and produces a single value output. This processing is performed through a basis function ϕ. Finally, the output layer weights the results of the intermediate layer by weights, aggregating them linearly to compose the final network response.

Among many candidates for basis functions, Gaussian radial basis function (RBF), presented in Eq. 1, is used in this study. The main reason for this choice is that it can be shown that an RBFN with Gaussian RBF can sufficiently approximate any given function for a large enough number of hidden layer elements [24].

Probabilistic methods are built on soft decision rules, which are formalized as probabilities, e.g., the probability of a data point being a saccade given the previous observations. The probabilities and thus, the decisions are adjusted to the observations.

$$\varphi(v_i) = e^{-(\sigma r)^2} \tag{1}$$

In the hidden layer, each processing element has a separate vector called the center, which has the same dimensions as the input vector. For n_h hidden layer elements we have n_h center vectors as $(c_1; c_2; \ldots; c_{nh})$. Then each processing element looks at the distance between the input vector and its center and uses this distance to create its output (activation phase).

This work uses only the initial and intermediary layers of the presented architecture. The initial layer channels the incoming data to the middle layer, which implicitly forms clusters during the activation phase. The formed grouping has an active center that changes according to the processed value. Changes in the active center are interpreted as possible concept drifts.

4.2. Markov Chains

A Markov chain model can be defined by the tuple $(S; A; \lambda)$. S corresponds to the state space, A is a matrix representing transition probabilities from one state to another, and λ is the initial probability distribution of the states in S. If there are n states in our Markov chain, then the matrix of transition probabilities A is of size $n \times n$.

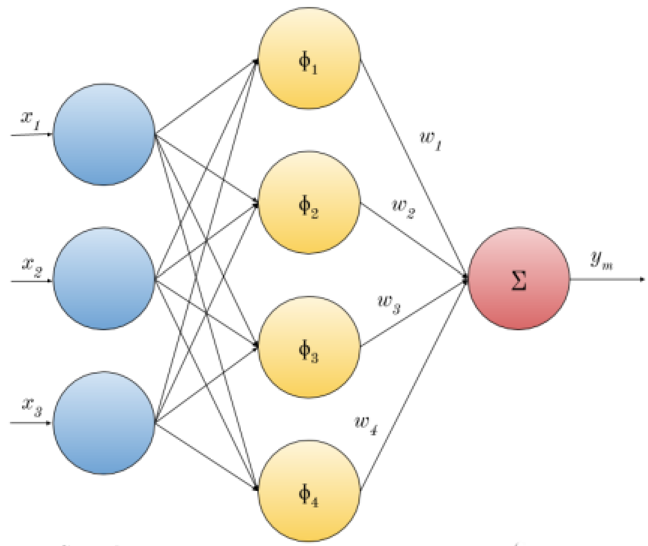

Figure 2: Topology of a RBFN

The fundamental property of the Markov model is the dependency on the previous state. If the vector $s(t)$ denotes the probability vector for all the states at time t, then:

$$\hat{s}(t) = \hat{s}(t-1)A \tag{2}$$

In this proposal, Markov chains are used to model the transitions (activations) between centers in the Radial Basis Function Network. For this formulation, a Markov state corresponds to one of the centers.

When the RBFN identifies a different center, a new state is registered in the Markov Chain. Initially, all possible transitions from this center have a zero value. If another center is activated, this change produces an increment in the probability of the correspondent transition. In paralell, all other transitions probabilites are decreased proportionally to the total number of possible transitions.

The use of a Markov Chain allows the proposed algorithm to keep an online model of the transitions. The probabilities sustained in this model are compared to parametric thresholds, to indicate when a warning zone is triggered, or a concept drift happens.

4.3. RBFChain

...

5. Analyses on Synthetic Datasets with Concept Drift

5.1. Experimental Setup

5.2. Results

6. Detection of Saccade and Fixation

6.1. Experimental Setup

6.2. Results

7. Concluding Remarks

References

[1] J. Cohen, B. Dolan, M. Dunlap, J. M. Hellerstein, C. Welton, Mad skills: New analysis practices for big data, Proc. VLDB Endow. 2 (2009) 1481–1492.

[2] C. C. Aggarwal, Data Streams: Models and Algorithms (Advances in Database Systems), Springer-Verlag, Berlin, Heidelberg, 2006.

[3] L. Zhou, J. Jiang, R. Liao, T. Yang, C. Wang, Fpga based low-latency market data feed handler, in: W. Xu, L. Xiao, J. Li, C. Zhang, Z. Zhu (Eds.), Computer Engineering and Technology, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015, pp. 69–77.

[4] F. Wang, L. Hu, D. Zhou, R. Sun, J. Hu, K. Zhao, Estimating online vacancies in real-time road traffic monitoring with traffic sensor data stream, Ad Hoc Netw. 35 (2015) 3–13.

[5] M. Delattre, B. Imbert, Method for management of data stream exchanges in an autonomic telecommunications network, 2015. US Patent 8,949,412.

[6] J. Kranjc, J. Smailovi, V. Podpean, M. Grar, M. nidari, N. Lavra, Active learning for sentiment analysis on data streams: Methodology and workflow implementation in the clowdflows platform, Information Processing & Management 51 (2015) 187 – 203.

[7] P. S. Kenkre, A. Pai, L. Colaco, Real time intrusion detection and prevention system, in: S. C. Satapathy, B. N. Biswal, S. K. Udgata, J. Mandal (Eds.), Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014, Springer International Publishing, Cham, 2015, pp. 405–411.

[8] A. Tsymbal, The problem of concept drift: definitions and related work, Computer Science Department, Trinity College Dublin 106 (2004) 58.

[9] K. Gama, D. Donsez, Deployment and activation of faulty components at runtime for testing self-recovery mechanisms, SIGAPP Appl. Comput. Rev. 14 (2014) 44–54.

[10] R. Elwell, R. Polikar, Incremental learning of concept drift in nonstationary environments, IEEE Transactions on Neural Networks 22 (2011) 1517–1531.

[11] M. Basseville, I. V. Nikiforov, Detection of Abrupt Changes: Theory and Application, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[12] I. Zliobaite, Learning under concept drift: an overview, CoRR abs/1010.4784 (2010).

[13] R. Leigh, D. Zee, The Neurology of Eye Movements, Contemporary neurology series, Oxford University Press, 2015.

[14] C. Privitera, L. Stark, Scanpath Theory, Attention, and Image Processing Algorithms for Predicting Human Eye Fixations, pp. 296–299.

[15] D. D. Salvucci, J. H. Goldberg, Identifying fixations and saccades in eye-tracking protocols, in: Proceedings of the 2000 Symposium on Eye Tracking Research & Applications, ETRA '00, ACM, New York, NY, USA, 2000, pp. 71–78.

[16] O. V. Komogortsev, A. Karpov, Automated classification and scoring of smooth pursuit eye movements in the presence of fixations and saccades, Behavior Research Methods 45 (2013) 203–215.

[17] E. Tafaj, G. Kasneci, W. Rosenstiel, M. Bogdan, Bayesian online clustering of eye movement data, in: Proceedings of the Symposium on Eye Tracking Research and Applications, ETRA '12, ACM, New York, NY, USA, 2012, pp. 285–288.

[18] K. Holmqvist, M. Nystrm, R. Andersson, R. Dewhurst, J. Halszka, J. van de Weijer, Eye Tracking : A Comprehensive Guide to Methods and Measures, Oxford University Press, 2011.

[19] P. Blignaut, Fixation identification: The optimum threshold for a dispersion algorithm, Attention, Perception, & Psychophysics 71 (2009) 881–895.

[20] F. Shic, B. Scassellati, K. Chawarska, The incomplete fixation measure, in: Proceedings of the 2008 Symposium on Eye Tracking Research & Applications, ETRA '08, ACM, New York, NY, USA, 2008, pp. 111–114.

[21] A. Santella, D. DeCarlo, Robust clustering of eye movement recordings for quantification of visual interest, in: Proceedings of the 2004 Symposium on Eye Tracking Research & Applications, ETRA '04, ACM, New York, NY, USA, 2004, pp. 27–34.

[22] T. Urruty, S. Lew, N. Ihadaddene, D. A. Simovici, Detecting eye fixations by projection clustering, ACM Trans. Multimedia Comput. Commun. Appl. 3 (2007) 5:1–5:20.

[23] C. M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag, Berlin, Heidelberg, 2006.

[24] S. Theodoridis, K. Koutroumbas, Pattern Recognition, Fourth Edition, Academic Press, Inc., Orlando, FL, USA, 4th edition, 2008.