CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Edmundo Albuquerque de Souza e Silva

Rio de Janeiro
Janeiro de 2017

CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

RIO DE JANEIRO, RJ – BRASIL
JANEIRO DE 2017

Ximenes Mendes, Diego

Change Point Detection in End-to-End Measurements Time Series/Diego Ximenes Mendes. – Rio de Janeiro: UFRJ/COPPE, 2017.

IX, 13 p.: il.; 29, 7cm.

Orientador: Edmundo Albuquerque de Souza e Silva

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2017.

Bibliography: p. 13 – 13.

1. Change Point Detection. 2. Time Series. 3. Machine Learning. I. Albuquerque de Souza e Silva, Edmundo. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

Janeiro/2017

Orientador: Edmundo Albuquerque de Souza e Silva

Programa: Engenharia de Sistemas e Computação

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

CHANGE POINT DETECTION IN END-TO-END MEASUREMENTS TIME SERIES

Diego Ximenes Mendes

January/2017

Advisor: Edmundo Albuquerque de Souza e Silva

Department: Systems Engineering and Computer Science

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Contributions

1.2 Dissertation Outline

Chapter 2

Change Point Detection

A change point detection algorithm is concerned to identify points in time where the statistical properties of a time series have changed. This problem have a broad application in different knowledge fields, and in general, the algorithms performance are closely related with the input characteristics. Also, when the latent information of the procedures which generated a time series are abscent the target statistical properties can be considered subjective, bringing difficulties not only in the detection phase but also in the problem formalization.

In this context this chapter specifies the problem and briefly discusses several change point detection algorithms. The literature of this area is extensive, and it is common to find methods which presents a poor performance due to a variety of reasons, such as the algorithms are too specific to a data characteristic or because the mechanisms were only analyzed through theoretical aspects. Therefore it was chosen a set of methods that can provide a good practical and theoretical perspective and also flexibility to insert adaptations that can better handle some input peculiarities. Also, through this chapter is exposed several obstacles when dealing with real data and some adopted solutions which are not described in literature.

2.1 Problem Definition

The problem can be categorized in offline or online. In the offline version, to decide if a specific point at time t is a change point the solver has available the whole time series, including past and future information on t. In the other hand, in the online version the information is available up to time t. The choice between these options are defined by the application domain, in some cases data are processed in real time and the change points should be detected as soon as possible, but in others the change points are identifyied by historical purpouses and offline algorithms can be used.

It is intuitive that the offline case is more robust since there are more information

to make a decision. In practice, to increase the statistical confidence of the decision, the online definition is relaxed and to decide if a point at time t is a change point it is possible to use data up to a small window in future of t, which in real time processing means that the application should wait until more data are available. This trick plays a trade-off between minimizing the time to detect a change and correctly classify a point. Therefore, in some cases, the online version can be transformed in the offline version by only modifying the input availability.

In this work it is considered the following input and change points caracteristics, which were defined considering the final application scenario:

- Univariate time series. However, it is possible to easily extend several methods presented here to deal with multivariate data.
- Unevenly time series, that is, data is not regularly sampled in time.
- Unknow number of change points.
- Time series with different lengths.
- Different number of points between change points.
- Focus on changes in the underlying mean and distribution, disconsidering other kinds of changes such as in periodicity.
- Outliers are not considered statistical changes.
- There is no latent information of the time series.
- It is considered the online and offline options.

2.2 Notation

An univariate time series composed of n points is defined by two vectors: $\mathbf{x} = (x_1, ..., x_n)$ and $\mathbf{y} = (y_1, ..., y_n)$. The value y_i indicates the $i-$th sampled value and x_i indicates the associated sample time. It is assumed that the points are sorted by time, that is, $x_{i-1} < x_i$ for $i = 2, ..., n$. Since unevenly time series is considered, $x_i - x_{i-1}$ can be different for different i values. For $s \geq t$ the following convention is adopted $\mathbf{y}_{s:t} = (y_s, ..., y_t)$.

The presence of k change points indicates that the data is splitted into $k + 1$ segments, also called windows. Let τ_i indicate the $i-$th change point for $i = 1, ..., k$. Also let $\tau_0 = 0$, $\tau_{k+1} = n$ and $\boldsymbol{\tau} = (\tau_0, ..., \tau_{k+1})$. Then, the $i-$th segment is defined by $\mathbf{y}_{\tau_{i-1}+1:\tau_i}$, assuming that $\tau_{i-1} < \tau_i$ for $i = 1, ..., k + 1$.

Through the previous definitions, change point detection algorithms mainly aim to find both k and $\boldsymbol{\tau}$.

2.3 Sliding Window Techniques

Sliding window techniques use two sliding windows over the time series and reduce the problem of detecting change points to the problem of testing whether data from the windows were generated by different distributions. One approach is to consider a distance metric between the two empirical distributions as the base to infer the change points. Let $d(\mathbf{a}, \mathbf{b})$ be the distance between two empirical distributions defined by the windows $\mathbf{a}$ and $\mathbf{b}$, and considering windows of length m, the 2.3 presents a simple sliding window algorithm.

Algorithm 1 Sliding Window

```
1: i ← 1
2: while i + 2m − 1 ≤ n do
3:     if d(y_{i:i+m−1}, y_{i+m:i+2m−1}) > α then
4:         Report i + m − 1 as a change point
5:         i ← i + m
6:     else
7:         i ← i + 1
8:     end if
9: end while
```

In this method, when the distance between the distributions of the two windows is above some threshold α a change point is reported. This is a common aproach for an online application, however it is possible to increase the classification accuracy in offline cases. As an example, the top plot of figure **??** presents a simulated time series, the segment $\mathbf{y}_{1:k}$ was generated sampling a $N(..., ...)$ distribution and $\mathbf{y}_{x+1:y}$ was sampled through $N(..., ...)$. The bottom plot of the same figure presents the associated distance between two sliding windows, where the point (i, d_i) represents the distance between the windows $\mathbf{y}_{i-m:i-1}$ and $\mathbf{y}_{i:i+m-1}$.

It can be observed that the peak of the distance is in the exact location where the distribution changed. However, using the threshold method is possible to prematuraly infer the location of the change point. Therefore, an alternative is to use a peak detection algorithm in the distance of two sliding windows time series.

The distance function choice can signifanctly impact the performance. There are several empirical distribution distances such as mahalanobis distance, hellinger distance, jensen shannon and EMD (Earth's Mover's Distance).

A performance improvement can be achieved concurrently executing the same sliding window algorithm with different window lengths, which enable the segmentation of segments with different sizes more easily.

2.4 Optimization Model

Given a fixed value of k, one approach is to define a cost function that measure the homogeneity of a segment and therefore choose the change points that globally optimize this homogeneity. Let the cost of the i-th segment be defined as $C(y_{\tau_{i-1}+1:\tau_i})$. The cost of a segmentation is the sum of all segments costs.

A commom choice for function C is the MSE (Mean Squared Error) which can capture changes in the mean. Another usual approach is to consider distribution changes through negative maximum log-likelihood functions, considering that data within a segment is iid.

Therefore, given a fixed k, the optimal segmentation is obtained through the following optimization problem, which is called the constrained case:

$$\min_{\boldsymbol{\tau}_{1:k}} \sum_{i=1}^{k+1} C(\mathbf{y}_{\tau_{i-1}+1:\tau_i}) \tag{2.1}$$

This problem can be solved using dynamic programming with $O(kn^2 f(n))$ time complexity, where $f(n)$ is related with C evaluation. Several segment cost functions can be evaluated in $O(1)$ after a $O(n)$ preprocessing phase, implying in an overall $O(kn^2)$ complexity. It is possible to proof that MSE, negative maximum log-likelihood function of normal, exponential, poisson and binomial distributions have this characterestic. Also the formulation can consider a minimum value of a segment length.

Modelling segments with continuous distributions can lead to practical dificulties. One of them is the fact that segments can form degenerate distributions, that is, the points of a segment can have zero variance, which is always the case of unitary length segments. In these cases the negative maximum likelihood are undefined. Two approaches can be used to avoid this situation. The first one tries to avoid degenerate segments adding a white noise with small variance to the time series. The second one considers that the cost of any degenerate distribution segment is equal to a small constant.

When the number of change points is unknown a common approach is to introduce a non decreasing penalty function $g(k)$. Then the new optimization problem, called penalized case, is:

$$\min_{k,\boldsymbol{\tau}_{1:k}} \sum_{i=1}^{k+1} C(\mathbf{y}_{\tau_{i-1}+1:\tau_i}) + g(k) \tag{2.2}$$

An approach to solve this problem is to solve the constrained case for $k = 0, ..., K$. This lead to an $O(K^2 n^2 f(n))$. However if the penalty function is linear in k the problem can be formulated in a more efficiently and solved in $O(n^2 f(n))$ time

complexity with dynamic programming.

Also there are several pruning algorithms to speedup the computation, in general trying to reduce the $\boldsymbol{\tau}$ search space but maintaning optimality.

2.5 Bayesian Inference

I will erase this section if I don't use bayesian inference. Describe Fearnheard (offline) and MacKay (online) solutions. Say that there are other versions.

2.6 HMM (Hidden Markov Model)

The idea that each segment of the time series is associated with a specific latent parameter of the procedure that generated the time series has a direct interpretation to a HMM model. In this context, each segment of a time series is related to a hidden state of a HMM model, and the observation distribution of this state represents the distribution of that segment. Therefore, the mechanism is resumed to model the target time series using a HMM and assessing when occurs a transition between different hidden states in the hidden state path.

There are several approaches in the detection and training phase. For example, given a trained HMM is possible to analyze the most probable hidden state path that a time series can follow through the viterbi algorithm. Also is possible to evaluate the probability of a transition between different hidden states at time t and apply a threshold method or peak detection as in the sliding window method. For the training phase is possible to use several time series to train a single HMM and use this model to detect change points in all time series. Another way is to train a model using only the time series to be processed.

It is important to note that structure of the hidden state graph has a big impact in the performance. Using a full connected structure the number of states defines the number of different distribution parameters of the different segments. Using a left to right structure the number of hidden states will induce the number of segments.

In **??** is stated that when using a full connected structure the time that a time series stays in the same hidden state is low, which can not reflect that fact that in real data the number of change points is also low. To overcome this problem **??** suggest to increase the time that a time series stay in the same hidden state through a dirichlet prior penalization.

2.7 Performance Evaluation

Describe how datasets are constructed in literature. Describe how an algorithm output is evaluated.

This problem can viewed as a binary classification of all points in a time series, the points are classified in change points or not change points.

Chapter 3

Dataset

3.1 Description of End-to-End Packet Loss Measurements Time Series

Here will be presented the TGR dataset. Small description on how data are collected, including client informations (geographic position, routes, etc) Plots: distribution between two consecutive measures, autocorrelation after time binarization, loss distribution, hour of day x loss, day of week x loss. Maybe: clusterize clients by distribution or time series.

3.2 Change Points Classification Survey

Describe web system used to get "true" change points. Describe majority voting. Describe how data were divided in train/test dataset.

Chapter 4

Applying Change Point Detection

In each algorithm section I will: Describe adaptations and aproaches. Describe dificulties of this algorithms in real data and in the current dataset that lead to adaptation.

4.1 Preprocessing

Filters applied to time series before presenting to algorithms.

4.2 Tuning Hyperparameters

4.2.1 Grid Search and Randomized Grid Search

4.2.2 Bayesian Optimization

I don't know if I am going to use this method.

4.2.3 Particle Swarm Optimization

I don't know if I am going to use this method.

4.3 Sliding Window

4.4 Dynamic Programming

4.5 HMM

4.6 Bayesian Inference

I don't know I will use this: poor performance.

4.7 LSTM

I don't know I will use this: maybe I will not have enough data.

4.8 Ensembles

If there are enough models to be tested describe how to use ensembles.

Chapter 5

Results

5.1 Classification Accuracy

Present false positive/false negative/...

5.2 Unsupervised Analysis

Clusterize clients according with change points detected and check if latent information of clusters are also clusterized.

5.3 Algorithms Comparison

Compare algorithms results and computational performance.

Chapter 6

Conclusions

Bibliography