Motif Discovery in the Irregulary Sampled Time Series Data

by

Anton Alyakin

A thesis submitted to The Johns Hopkins University

in conformity with the requirements for the

Senior Honors Thesis program in Computer Science

Baltimore, Maryland

May, 2019

© 2019 by Anton Alyakin

All rights reserved

Abstract

Motifs are patterns that repeat within and across different time series. They can be used for various applications, such as clustering or discovering association rules. For example, in patinet monitoring they can be used to identify features that are predictive of a diagnosis. Most of the motif definitions in literature are not applicable to the case when the data is irregularly sampled, which is often the case in the areas such as medical data.

In this work, we present a generative model for unsupervised identification of motifs for the case when the observation times are highly irregular. In particular, we model each motif as a combination of a Poisson Point Process for the distribution of the timestamps and a Gaussian Process for the distribution of the observations. This allows us to use both the sampling frequency and the observation values in order to identify a motif. The whole time series is modeled as a Hidden Markov Model, in which each time step corresponds to a new motif. We present a version of the Viterbi Training procedure for the learning of the parameters of this model. We demonstrate experimentally that this procedure is able to re-learn the motifs in the data set generated from this model. Lastly, we present the results of using this model on laboratory tests data of the MIMIC-III, a well-known critical care dataset.

Acknowledgments

I would like to thank my advisor Dr. Suchi Saria for providing me with an opportunity to work in her lab and supporting the work presented here.

I would like to thank both the whole JHU Computer Science, and the JHU Applied Mathematics and Statistics, which are my primary and secondary departments, respectively. I am very thankful for the challenging, yet very rewarding undergraduate experience I had while being a part of them.

I would like to express my unmeasurable gratitude to Noam Finkelstein for his direct and indirect contributions to this work. Over two years of working alongside him, I have learned an uncountable number of things from him, ranging on topics from machine learning and programming to approach to life and the fact that the Caps Lock just *must* be mapped to Escape. This work simply could not have been completed without his help and guidance.

I would like to sincerely thank all my friends who have served both as my source of mathematical knowledge and moral support in working on this thesis. Specfically, I want to thank Ali Geisa, who has kindly provided a rigorous editing of both my math and my English use in this paper.

Lastly, I want to thank my mother for always believing in me, and my father for teaching me resilience, as it took a lot of it to complete this work.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Continuous time series data is collected across many domains, including patient monitoring, finance and pose tracking (Saria, Duchi, and Koller, 2011). In time series literature, frequently repeating patterns within and across time series are commonly called motifs. They are the continuous counterparts to the sequence motifs that are common in fields such as computational biology. (Lin et al., 2002) Some examples of time series motifs include an EEG pattern that commonly precedes a seizure, or a burst in traffic of an antenna when a major social event is located nearby. (Castro and Azevedo, 2010) Discovered motifs may be used for various subsequent tasks, ranging from unsupervised knowledge discovery and domain understanding to being used as higher-level features for segmentive or discriminative tasks. (Saria, Duchi, and Koller, 2011; Mcmillan et al., 2012)

Although they all describe a similar phenomenon, the definitions of what is considered a motif differ significantly across the literature: from subsequences that are similar enough in some distance metric (Lin et al., 2002; Gao and Lin, 2018; Shokoohi-Yekta et al., 2015; Mueen and Keogh, 2010) to continuous

functions from which motif subsequences are sampled (Saria, Duchi, and Koller, 2011).

None of the motif definitions, at least to our knowledge, can be used to discover motifs in irregularly sampled time series data without its prior discretization. The naive ways of extending the regularly sampled approaches work best when the gaps in data are similar to each other. However, there are many cases when the time series which are sampled very irregularly. For example, in the ICU monitoring setting, the laboratory measurements, such as creatinine or blood cell counts can significantly vary in the frequency of measurements. (Soleimani, Hensman, and Saria, 2017)

Furthermore, naive approaches do not consider the sampling patterns of the data as something significant to the motifs themselves, yet it is clear that they can contain information that is not captured in the data otherwise. In the medical time series data, a common example of such is the physicians' intuition on the patient's condition, which promotes more or less frequent sampling. (Soleimani, Hensman, and Saria, 2017)

In this paper, we present a probabilistic framework that models the generation of the data as a sequence of the motifs, each of which is generated from a certain template. We propose an unsupervised algorithm for learning such templates which comes from an extension of the Viterbi training approach for the Hidden Markov Models.

We then show that our learning procedure is able to relearn the motifs in the data generated from the model. Lastly, we present results from a run of our model on MIMIC-III creatinine lab data.

Chapter 2

Related Work

We consider there to be three major branches of motif discovery in the time series literature: "set motifs", "pair motifs" and generative models.

The concept of a motif as a previously unknown, frequently occurring pattern in a time series data has been first introduced in Lin et al., 2002. In this classic form, the problem of motif discovery focuses on finding the most frequent patterns. (Gao and Lin, 2018) This approach identifies motifs as patters that consist of two or more similar subsequences based on some distance threshold (Lin et al., 2002; Li, Lin, and Oates, 2012). Formally speaking, if $\{x_i : i = 1, \ldots, N\}$ are possible patterns, $d(\cdot, \cdot)$ is some distance metric and M is a threshold, then these algorithms search for

$$\hat{x} := \arg\max_{x} |\{x_i : d(x, x_i) < M\}| \tag{2.1}$$

This approach is sometimes referred to as "set motifs". There also has been work on applying symbolic representations of time series to the problem of identifying sets of similar subsignals. (Lin et al., 2007)

In much of the recent research, motifs have instead been defined as the

most similar pair of subsignals, judged, again, by some distance metric. (Mueen and Keogh, 2010; Mueen et al., 2009; Shokoohi-Yekta et al., 2015) This alternative approach is sometimes called "pair motifs" (Gao and Lin, 2018). Within this notion, motif discovery does not provide information about structure in the data that is repeated more than twice.

Lastly, there is a generative model approach in which signals are assumed to be subsampled from a low-variance model. Motifs are then considered to be parameters of this model, which has been previously assumed to be either a multivariate normal distribution (Minnen et al., 2007) or a more flexible class of functions such as Bézier splines (Saria, Duchi, and Koller, 2011).

Both the "set-motifs" and "pair-motifs" approaches cannot be applied to the irregularly-sampled time series in a straight-forward manner, since they require defining a distance metric between two motifs. No obvious metric exists for the case when the number of observations and their times are flexible. There has been work in discretizing irregularly time series (Mcmillan et al., 2012). This approach work best when the gaps in the data are similar to each other, which is often not the case in fields such as medical time series.

Data missing not at random is an important topic of research in biomedical data science. Soleimani, Hensman, and Saria, 2017 use Gaussian Processes to jointly model various time series of vital signs and lab values. Fauber and Shelton, 2018 choose to use a Marked Point Process, similarly to us, but chose a piecewise-constant conditional model for the marks. As far as we are aware, no previous work has directly applied missing data modeling to the topic of motif discovery.

Chapter 3

Preliminaries

3.1 Markov and Hidden Markov Models

3.1.1 Markov Chain

We only consider the case of a discrete-time discrete-state Markov Chain, since it serves as the backbone for Hidden Markov Models. In such setting, a Markov Chain, or a Markov Model, is probabilistic model for a sequence of states $z_1, \ldots, z_N$ of an arbitrary length N. The basic assumption of a Markov Chain is that an observation z_n contains all relevant information for predicting the future, making it a sufficient statistic. For discrete time steps, this allows us to decompose the joint distribution as following:

$$p\left(z_1, \ldots, z_N\right) = p\left(z_1\right) \prod_{n=2}^{N} p\left(z_n | z_{n-1}\right) \tag{3.1}$$

It is typical to assume that the distribuiton $p\left(z_n | z_{n-1}\right)$ is independent of time, also known as homogeneous, or time-invariant.

For all our purposes, the states come from a pre-specified categorical set $z_n \in \{1, \ldots, M\}$. In this case, the conditional distribution $p\left(z_n | z_{n-1}\right)$ can be

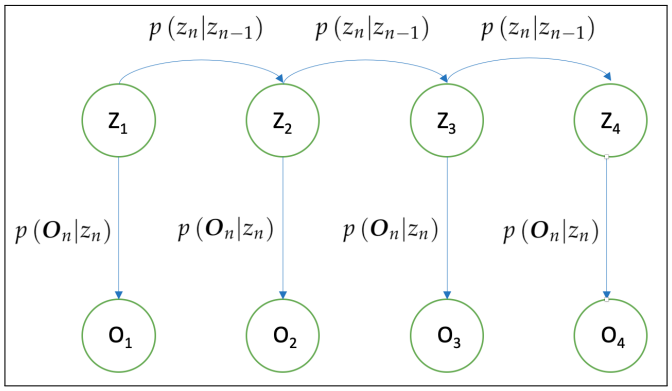

Figure 3.1: A schematic of a Hidden Markov Model

written as a matrix with dimensions $N \times N$, with the property that $\sum_{j=1}^{M} A_{ij} = 1$. Each entry in the matrix represents the probability of transitioning to state j conditioned on being in state i: $A_{ij} = p\left(z_n = j | z_{n-1} = i\right)$. From hereafter, we refer to this as a transition matrix. (Bishop, 2006; Murphy, 2012)

3.1.2 Hidden Markov Model

A Hidden Markov Model (HMM) consists of a discrete-time, discrete-state Markov chain and an observation model $p(\boldsymbol{O}_n|z_n)$. A schematic of an HMM is provided in the Figure 3.1 Note that the distribution of an observation at a time-step is fully specified conditioned on knowing the state that the chain is in.

The full joint distribution of the HMM has the form

$$p\left(z_1, \ldots, z_N, \boldsymbol{O}_1, \ldots, \boldsymbol{O}_N\right) = p\left(z_1, \ldots, z_N\right) \prod_{n=1}^{N} p\left(\boldsymbol{O}_n | z_n\right) \tag{3.2}$$

$$= p\left(z_1\right) \prod_{n=2}^{N} p\left(z_n | z_{n-1}\right) \prod_{n=1}^{N} p\left(\boldsymbol{O}_n | z_n\right) \tag{3.3}$$

The learning of the parameters in a Hidden Markov Model is typically achieved through a specialized Expectation-Maximization algorithm, called Baum-Welch, or through one of its approximations, such as Viterbi training. (Bishop, 2006; Murphy, 2012) We go into further detail on parameter estimation in the Chapter 5 of this work.

3.2 Point Processes

Point processes are distributions of points randomly located on some underlying space. Within the scope of this work, this space is constrained to a segment of real line, $[0, a)$, and the points can be thought of as timestamps, associated with some observation.

3.2.1 Stationary Poisson Process

We are specifically interested in the Stationary Poisson Process, which is a Point Process that has the following three properties:

1. The number of points in each finite interval has a Poisson distribution.

$$p\left(\dim\left(\boldsymbol{t}\right)|\lambda\right) = \frac{\left(\lambda a\right)^{\dim(\boldsymbol{t})}}{\left(\dim\left(\boldsymbol{t}\right)\right)!}\exp\left(-\lambda a\right) \tag{3.4}$$

2. The number of points in disjoint intervals are independent random variables.

3. The distributions are stationary, meaning that they depend only on the lengths of the intervals.

(Daley and Vere-Jones, 2003)

The Stationary Poisson process has some other characterizations. For example, the distribution until the next observation, also known as the interarrival time, conditioned on previous history is exponential. Because of this, it has a rather nice form for the conditional likelihood of the locations of the points within a closed interval of length a, conditioned on their number

$$p\left(\boldsymbol{t}|\dim\left(\boldsymbol{t}\right),\lambda\right)=\frac{\dim\left(\boldsymbol{t}\right)!}{a^{\dim(\boldsymbol{t})}} \tag{3.5}$$

(Ross, 1996)

With these properties the likelihood of observing a sample $\boldsymbol{t}$ on the interval $[0,a)$, without knowing its size a priori, is simply:

$$p\left(\boldsymbol{t},\dim\left(\boldsymbol{t}\right)|\lambda\right)=p\left(\boldsymbol{t}|\dim\left(\boldsymbol{t}\right),\lambda\right)p\left(\dim\left(\boldsymbol{t}\right),\lambda\right) \tag{3.6}$$

$$=\frac{\dim\left(\boldsymbol{t}\right)!}{a^{\dim(\boldsymbol{t})}}\frac{\left(\lambda a\right)^{\dim(\boldsymbol{t})}}{\left(\dim\left(\boldsymbol{t}\right)\right)!}\exp\left(-\lambda a\right) \tag{3.7}$$

$$=\left(\lambda\right)^{\dim(\boldsymbol{t})}\exp\left(-\lambda a\right) \tag{3.8}$$

One may notice that while using the Stationary Poisson Process, we do not retrieve any information from a sample other than the number of observations. There certainly may be more complex patterns in the timestamp generation process. For the discussion on possible use of other process in this model, see Chapter 7 of this work.

3.3 Gaussian Processes

3.3.1 Gaussian Processes Introduction

A Gaussian Process (GP) is a stochastic process for which every finite collection of its random variables has a multivariate normal distribution, i.e. that $f(t_1), \ldots, f(t_N)$ are jointly Gaussian. In supervised learning, GPs are used to infer a distribution over functions of data and then use this to make predictions given new inputs.

That is, if we assume that $x_i = f(t_i) + \epsilon_i$ for some unknown function f, and distribution of noise ϵ_i, we aim to compute:

$$p\left(\boldsymbol{f}_*|\boldsymbol{t}_*, \boldsymbol{t}_\#, \boldsymbol{x}_\#\right) = \int p\left(\boldsymbol{f}_*|f, \boldsymbol{t}_*\right) p\left(f, \boldsymbol{t}_\#, \boldsymbol{x}_\#\right) df \tag{3.9}$$

(Murphy, 2012; Bishop, 2006)

Throughout this section, we will use the notation in which $\boldsymbol{t}_\#$ and $\boldsymbol{x}_\#$ represent training inputs and outputs, correspondingly. $\boldsymbol{f}_\#$ are hypothetical noise-less training outputs, which are never observed. $\boldsymbol{t}_*$ and $\boldsymbol{f}_*$ are test inputs and outputs. This way, f is a function and $\boldsymbol{f}_\#$ and $\boldsymbol{f}_*$ are outputs of this function evaluated at $\boldsymbol{t}_\#$ and $\boldsymbol{t}_*$. The primary reason for this notation is the consistency and intuitive correspondence with the notation used for our model later. In order to not overcomplicate the preliminaries, we assume that each individual input is one-dimensional, since this is the case for our model in which inputs are timestamps.

3.3.2 Gaussian Processes for Regression

We restrict our scope investigation of Gaussian processes to the topic of regression, since it is the only one relevant to the further discussion.

We let the prior on the regression function be a Gaussian Process, which we denote as $f(t) \sim GP\left(m(t), \kappa(t, t')\right)$. $m(t)$ is the mean function:

$$m(t) = \mathbb{E}\left(f(t)\right). \tag{3.10}$$

We use the common assumption that $m(t) = 0$, which is reasonable due to the fact that GPs are flexible enough to model the mean arbitrarily well. (Murphy, 2012)

$\kappa(t, t')$ is the kernel or covariance function, which is required to be positive.

Commonly used kernels in Gaussian processes work include the RBF kernel, defined as

$$\kappa(t, t') = \sigma_f^2 \exp\left(-\frac{(t - t')^2}{2l^2}\right), \tag{3.11}$$

where the parameter l controls the horizontal length scale and σ_f^2 controls the vertical variation. Some other non-trivial kernels include Matern Kernel, which is a generalization of the RBF kernel, rational quadratic kernel and periodic kernels. We use the RBF kernel in our work.

Using the previously defined notation for the test and training set, we can

define the kernel matrices as following:

$$\boldsymbol{K}_{\#\#} = \left(\kappa(t_{\#,i}, t_{\#,j})\right) \tag{3.12}$$

$$\boldsymbol{K}_{\#*} = \left(\kappa(t_{\#,i}, t_{*,j})\right) \tag{3.13}$$

$$\boldsymbol{K}_{**} = \left(\kappa(t_{*,i}, t_{*,j})\right), \tag{3.14}$$

where $t_{\#,i}$ refers simply to the ith element of the vector $\boldsymbol{t}_{\#}$ and similarly for $t_{*,i}$. (Murphy, 2012; Bishop, 2006)

3.3.2.1 Regression with Noiseless Observations

From the definition of the GPs, if we were able to observe the noise-less observations, with the joint distribution of the form,

$$\begin{pmatrix} \boldsymbol{f}_{\#} \\ \boldsymbol{f}_{*} \end{pmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{pmatrix} \boldsymbol{K}_{\#\#} & \boldsymbol{K}_{\#*} \\ \boldsymbol{K}_{\#*}^{T} & \boldsymbol{K}_{**} \end{pmatrix}\right), \tag{3.15}$$

we could very simply determine the posterior distribution using the rule of conditioning for a Gaussian:

$$p\left(\boldsymbol{f}_{*} | \boldsymbol{t}_{*}, \boldsymbol{t}_{\#}, \boldsymbol{f}_{\#}\right) = \mathcal{N}\left(\boldsymbol{f}_{*} | \boldsymbol{\mu}_{*}, \boldsymbol{\Sigma}_{*}\right), \tag{3.16}$$

where the parameters are

$$\boldsymbol{\mu}_{*} = \boldsymbol{K}_{\#*}^{T} \boldsymbol{K}_{\#\#}^{-1} \boldsymbol{f}_{\#} \tag{3.17}$$

$$\boldsymbol{\Sigma}_{*} = \boldsymbol{K}_{**} - \boldsymbol{K}_{\#*}^{T} \boldsymbol{K}_{\#\#}^{-1} \boldsymbol{K}_{\#*}. \tag{3.18}$$

In this case, the Gaussian Process becomes an interpolator, collapsing the density at every already observed point to a mass with no uncertainty.

3.3.2.2 Regression with Noisy Observations

In order to avoid this effect, instead of assuming that out observations are noise-free, we assume that we observe noisy observations $x_{\#,i} = f\left(t_{\#,i}\right) + \epsilon_i$, and the model for the noise is Gaussian $\epsilon_i \sim \mathcal{N}\left(0, \sigma_y^2\right)$. The covariance of the observed noisty responses is then:

$$\boldsymbol{K}'_{\#\#} = \boldsymbol{K}_{\#\#} + \boldsymbol{I}\sigma_y^2 \tag{3.19}$$

In this case, the joint density of observed data and the noise-free responses becomes:

$$\begin{pmatrix} \boldsymbol{x}_{\#} \\ \boldsymbol{f}_* \end{pmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{pmatrix} \boldsymbol{K}'_{\#\#} & \boldsymbol{K}_{\#*} \\ \boldsymbol{K}_{\#*}^T & \boldsymbol{K}_{**} \end{pmatrix}\right) \tag{3.20}$$

and the posterior predictive density is obtained very similarly to the noise-less case:

$$p\left(\boldsymbol{f}_* | \boldsymbol{t}_*, \boldsymbol{t}_{\#}, \boldsymbol{x}_{\#}\right) = \mathcal{N}\left(\boldsymbol{f}_* | \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*\right) \tag{3.21}$$

with the following parameters:

$$\boldsymbol{\mu}_* = \boldsymbol{K}_{\#*}^T \left(\boldsymbol{K}'_{\#\#}\right)^{-1} \boldsymbol{f}_{\#} \tag{3.22}$$

$$\boldsymbol{\Sigma}_* = \boldsymbol{K}_{**} - \boldsymbol{K}_{\#*}^T \left(\boldsymbol{K}'_{\#\#}\right)^{-1} \boldsymbol{K}_{\#*} \tag{3.23}$$

(Bishop, 2006; Murphy, 2012)

Chapter 4

Model

4.1 Model Assumptions

In this work we aim to identify the motifs, or the patterns in the data set of some small length that repeat throughout this time series. For the brevity of notation, we will define the model for only one time series of length L that is defined by one vector of timestamps $\boldsymbol{\tau} \in \mathbb{R}^L$ and one vector of corresponding marks, or observations[1], $\boldsymbol{x} \in \mathbb{R}^L$. All of the results generalize to the presence of multiple time series, each with its own vectors of timestamps and observations.

We impose two additional assumptions: that the time length of each motif is constant, a, and that each motif can start only at a timepoint that is a multiple of this constant: na, where n is some integer. These two assumptions allow us to use the discrete-time HMM in order to model the distribution of states. Although there has been work done on the continuous-time HMMs (Shelton and Ciardo, 2014), there doesn't exist an analogous Viterbi algorithm

[1]The terms observations and marks are used interchangeably throughout the paper.

for them that is applicable to our setting. We emphasize that despite the use of discrete-time HMMs, the space of timestamps is still continuous.

From now on, we will refer to the observation windows within the data time series as *motifs* and the models that each of those observation windows could be generated from as *templates*. We denote the total number of these motifs in the sequence as N and the total number of possible templates as M, where $M << N$. Clearly, the last time point in the vector $\boldsymbol{\tau}$ is upper bounded by Na. Each motif has a corresponding variable $z_n \in 1, \ldots, M$ which denotes the index of the template that it was generated from.

Every template is a combination of parameters for two processes: one to describe the distribution of the timepoints and one to describe the distribution of the associated marks. For the former, we use a Poisson Process for all the templates, with a unique, learned intensity λ_m for each template. For the associated marks, we use a separate Gaussian Process for each template.

Lastly, we make the Hidden Markov Model assumptions. The sequence of observation windows is treated as the observed data. The sequence of corresponding templates that generated these windows is treated as the latent states. Specifically, the assumptions imply two things: that the data-generating process is fully specified conditioned on the knowledge of which template this motif is generated from, and that the distribution of the next template state is fully specified conditioned on the knoweldge of the current template state.

4.2 Notation

Before we jump into estimating the parameters of the model, we would like to present some notations that are essential to the rest of the paper. First of all, we, without any loss of generality, assume that the vector of timestamps $\boldsymbol{\tau}$ has been ordered from the smallest to largest, and the associated vector of $\boldsymbol{x}$ has been ordered to preserve the positional correspondance with $\boldsymbol{\tau}$.

Furthermore, we are generally concerned with the temporal offset of the observation from the beginning of the motif as opposed to the beginning of the whole time series. Thus, we define a new vector $\boldsymbol{t}$, which denotes the timestamps in the modular space, i.e. $\boldsymbol{t} := \boldsymbol{\tau} \mod a$, sometimes also represented as $\boldsymbol{t} = \boldsymbol{\tau} \,\%\, a$. The positional organization of the original vector is preserved. Thus, our vector $\boldsymbol{t}$ is in $[0, a)^L$, compared to $\boldsymbol{\tau} \in \mathbb{R}^L$.

As a simple example, consider an original vector of timestamps $\boldsymbol{\tau} = [0.5, 0.75, 1.5, 3.25]$. If we choose the length of motif to be $a = 1$, then the modified vector would be $\boldsymbol{t} = [0.5, 0.75, 0.5, 0.25]$.

Now, we define new sub-vectors $\boldsymbol{t}_{*_n}$ and $\boldsymbol{x}_{*_n}$, which correspond to the timestamps or observations, respectively, that belong only to one specific motif.

$$\boldsymbol{t}_{*_n} = [t_i | (n-1)a \leq \tau_i < na]^T \tag{4.1}$$

$$\boldsymbol{x}_{*_n} = [x_i | (n-1)a \leq \tau_i < na]^T \tag{4.2}$$

Since our vectors are properly ordered, it may be intuitive to think about this

as a computer science array slicing operation. [2]

Lastly, we present a way to refer to all the timestamps and observations that belong to the same template. Denote the set of all motifs that belong to some template m as

$$\boldsymbol{S}_m = \{n | z_n = m\} \tag{4.3}$$

Then, we call $\boldsymbol{t}_{\#_m}$ a vector of all timestamps that belong to motifs that were generated from the template m and, similarly, $\boldsymbol{x}_{\#_m}$ a vector of all observations that were generated from the template m:

$$\boldsymbol{t}_{\#_m} = [t_i | t_i \in \boldsymbol{t}_{*_n}, n \in \boldsymbol{S}_m]^T \tag{4.4}$$

$$\boldsymbol{x}_{\#_m} = [x_i | x_i \in \boldsymbol{x}_{*_n}, n \in \boldsymbol{S}_m]^T \tag{4.5}$$

In computer programming this can be thought of as simply the concatenation of array slices $\boldsymbol{t}_{*_n}$ for all n such that $z_n = m$.

[2] Abusing Python notation slightly, if our motif n starts at index i and the motif $n+1$ starts at index j, then $\boldsymbol{t}_{*_n}$ is nothing more than $\boldsymbol{t}[i:j]$

Chapter 5

Learning

We use Viterbi training in order to learn the parameters, which is similar to the Expectation-Maximization (EM) algorithm.

5.1 Expectation Step

Unlike the EM algorithm, Viterbi training uses Viterbi decoding instead of the forward-backward algorithm in the E step. Thus, instead of the posterior marginals of all hidden state variables given a sequence of observations/emissions, we obtain the most likely sequence of hidden states. (Murphy, 2012) This is prefered for our application, since it is not intuitive how to train Gaussian Processes when an observation belongs to the process with some probability.

Viterbi decoding requires the knowledge of the transition probabilities, which are summarized in the transition matrix $\boldsymbol{A}$, and an ability to compute the likelihood of the observation, conditioned on the state $p(\boldsymbol{O}_n|z_n = m)$. We update the transition matrix every M step. In this model, the observations are

the full motifs, i.e. it is a tuple of timestamps and marks that belong to that motif

$$\boldsymbol{O}_n = (\boldsymbol{t}_{*_n}, \boldsymbol{x}_{*_n}). \tag{5.1}$$

We show how to compute probabilities of such emissions in the following section.

5.1.1 Emission Probability

Each of the emission likelihoods requires us to compute the joint distribution of the timestamps and the observations in the template. However, we can use a simple conditional probability to decompose it as following:

$$p(\boldsymbol{O}_n | z_n = m) = p\left(\boldsymbol{t}_{*_n}, \boldsymbol{x}_{*_n} | z_n\right) \tag{5.2}$$

$$= p\left(\boldsymbol{x}_{*_n} | \boldsymbol{t}_{*_n}, z_n\right) p\left(\boldsymbol{t}_{*_n} | z_n\right) \tag{5.3}$$

5.1.1.1 Emission Probability - Timestamps

Recall that we have assumed that for each motif timestamps are generated according to a Poisson Process with an intensity λ_m that is unique to the template it is generated from. Thus, the likelihood of the number of observation times is the expression presented in the preliminaries, which is proportional to the Poisson distribution for any given a and a specific sample:

$$p\left(\boldsymbol{t}_{*_n} | z_n = m\right) = (\lambda_m)^{\dim(\boldsymbol{t}_{*_n})} \exp\left(-\lambda_m a\right) \tag{5.4}$$

$$\propto \frac{(\lambda_m a)^{\dim(\boldsymbol{t}_{*_n})}}{(\dim\left(\boldsymbol{t}_{*_n}\right))!} \exp\left(-\lambda_m a\right) \tag{5.5}$$

5.1.1.2 Emission Probability - Observations

Recall that in equation 3.21 we have specified exactly the posterior probability distribution of a trained Gaussian Process. The notation used there transitions directly to the one we are using to denote motifs and templates. Computing the likelihood of the marks given the timestamps and the template it was generated from is simply:

$$p\left(\boldsymbol{x}_{*_n}|\boldsymbol{t}_{*_n}, z_n\right) = p\left(\boldsymbol{f}_* = \boldsymbol{x}_{*_n}|\boldsymbol{t}_{*_n}, \boldsymbol{t}_{\#_m}, \boldsymbol{x}_{\#_m}\right) \tag{5.6}$$

$$= \mathcal{N}\left(\boldsymbol{f}_* = \boldsymbol{x}_{*_n}|\boldsymbol{\mu}_{*_n}, \boldsymbol{\Sigma}_{*_n}\right) \tag{5.7}$$

where the parameters are

$$\boldsymbol{\mu}_{*_n} = \boldsymbol{K}_{\#_m*_n}^T \left(\boldsymbol{K}'_{\#\#_m}\right)^{-1} \boldsymbol{f}_{\#_m} \tag{5.8}$$

$$\boldsymbol{\Sigma}_{*_n} = \boldsymbol{K}_{**_n} - \boldsymbol{K}_{\#_m*_n}^T \left(\boldsymbol{K}'_{\#\#_m}\right)^{-1} \boldsymbol{K}_{\#_m*_n} \tag{5.9}$$

The Kernel matrix here intuitively extends the notations we have used to designate train and test datasets of the GPs and the motif and template subselection. Specifically,

$$\boldsymbol{K}_{\#\#_m} = \left(\kappa(t_{\#_m,i}, t_{\#_m,j})\right) \tag{5.10}$$

$$\boldsymbol{K}_{\#_m*_n} = \left(\kappa(t_{\#_m,i}, t_{*_n,j})\right) \tag{5.11}$$

$$\boldsymbol{K}_{**_n} = \left(\kappa(t_{*_n,i}, t_{*_n,j})\right) \tag{5.12}$$

5.1.2 Viterbi Decoding

We use the likelihood of the data given any template computed above, as well as the matrix of the transition probabilities $\boldsymbol{A}$, which is updated in the Maximization step, to determine the most probable path.

Define $\delta_n(m)$ to be the probability of ending up in state m at step n, given that we take the most probable path to it. More formally:

$$\delta_n(m) := \max_{z_1,\ldots,z_{n-1}} \{p\left(z_1, ..., z_{n-1}, z_n = m|\mathcal{H}_{1,n}\right)\} \tag{5.13}$$

An important observation is that the path the most probable path to the this state corresponds to the most probable path to the previous step and then transition to the current state, i.e.:

$$\delta_n(m) := \max_k \delta_{n-1}(k) A_{k,m} p(\boldsymbol{O}_n|z_n = m) \tag{5.14}$$

In order to prevent numerical underflow, we work in the log domain. Since $\log\max = \max\log$, we can just use the following:

$$\log\left(\delta_n(m)\right) := \max_k \left\{\log\left(\delta_{n-1}(k)\right) + \log\left(A_{k,m}\right) + \log\left(p(\boldsymbol{O}_n|z_n = m)\right)\right\}. \tag{5.15}$$

We want to store the most likely previous state for the most probable state to $z_n = m$ as $a_n(m)$:

$$a_t(m) := \arg\max_k \left\{\log\left(\delta_{n-1}(k)\right) + \log\left(A_{k,m}\right) + \log\left(p(\boldsymbol{O}_n|z_n = m)\right)\right\}. \tag{5.16}$$

For only one time series, the first most probable state cam be inferred from

data

$$\log\left(\delta_1(m)\right) := \log\left(p(\boldsymbol{O}_1|z_1 = m)\right). \tag{5.17}$$

When we are using multiple time series, we can instead initialize this using the initial probabilities:

$$\log\left(\delta_1(m)\right) := \log\left(\pi(m)\right) + \log\left(p(\boldsymbol{O}_1|z_1 = m)\right). \tag{5.18}$$

When we terminate in the last step, which we denote T, we use the traceback procedure in order to compute the sequence of the most probable steps. The most probable final step is:

$$z_T^* = \arg\max_m \left\{\delta_T(m)\right\} \tag{5.19}$$

and every step before it is recovered using a recursive procedure:

$$z_t^* = a_{t+1}\left(z_{t+1}^*\right) \tag{5.20}$$

which leaves us with the jointly most probable sequence of states.

5.2 Maximization Step

5.2.1 Transition Matrix

The optimal i, jth entry to the transition matrix is simply the counts of the transitions from the state i to state j that occur in the most likely path recovered by the Viterbi, normalized by the total number of times the state i has been visited, excluding the very last state:

$$A_{i,j} = \frac{\sum_{k=1}^{N-1} \mathbb{1}\left(z_k = i, z_{k+1} = j\right)}{\sum_{k=1}^{N-1} \mathbb{1}\left(z_k = i\right)} \tag{5.21}$$

5.2.2 Intensities

The likelihood of the timestamps is different from the likelihood of a sample of Poissons by some a multiplier that is a function of the data and motif length, thus the MLE for the intensity of the process is simply the MLE of Poissons:

$$\lambda_m = \frac{\dim(\boldsymbol{t}_{\#_m})}{a \sum_{k=1}^{N-1} \mathbb{1}\left(z_k = i\right)}. \tag{5.22}$$

The numerator is the total number of emissions across the motifs that are assumed to be generated from template m and the denominator is the count of motifs that have been generated from this template scaled by length.

5.2.3 Gaussian Processes - Kernel Matrices

Since Gaussian Processes are nonparametric, we store all the data corresponding to the specific motif template after the most recent iteration, In other words, we just define store updated $\boldsymbol{t}_{\#_m}$ and $\boldsymbol{x}_{\#_m}$, as there is no less dimensional sufficient statistic.

5.2.4 Gaussian Processes - Kernel Parameters

For Kernel Parameters, we use a continuous optimization method that maximize the marginal likelihood, defied as for the marks of a template m as

$$p\left(\boldsymbol{x}_{\#_m}|\boldsymbol{t}_{\#_m}\right) = \int p\left(\boldsymbol{x}_{\#_m}|\boldsymbol{f}_{\#_m}, \boldsymbol{t}_{\#_m}\right) p\left(\boldsymbol{f}_{\#_m}|\boldsymbol{t}_{\#_m}\right) d\boldsymbol{f}_{\#_m} \tag{5.23}$$

Since by assumptions, we know that $p\left(\boldsymbol{f}_{\#_m}|\boldsymbol{t}_{\#_m}\right) = \mathcal{N}\left(\boldsymbol{f}_{\#_m}|\boldsymbol{0}, \boldsymbol{K}_{\#\#_m}\right)$ and the noise is homoscedastic and Gaussian, the marginal log likelihood is given by:

$$\log\left(p\left(\boldsymbol{x}_{\#_m}|\boldsymbol{t}_{\#_m}\right)\right) = \log\left(\mathcal{N}\left(\boldsymbol{f}_{\#_m}|\boldsymbol{0}, \boldsymbol{K}'_{\#\#_m}\right)\right) \tag{5.24}$$

$$= -\frac{1}{2}\boldsymbol{x}_{\#_m}^T\left(\boldsymbol{K}'_{\#\#_m}\right)^{-1}\boldsymbol{x}_{\#_m} - \frac{1}{2}\log\left(|\boldsymbol{K}_{\#\#_m}|\right) - \frac{N\log(2\pi)}{2} \tag{5.25}$$

We can note that the first term represents the quality of the data fit, the second term is the term that penalizes the model complexity and the last term is constant. (Murphy, 2012)

Let's denote the kernel parameters by $\boldsymbol{\theta}$. For the RBF kernel that we are using, this is a tuple (σ_f, l). We can determine the partial with respect to each of these parameters

$$\frac{\partial}{\partial\theta_j}\log\left(p\left(\boldsymbol{x}_{\#_m}|\boldsymbol{t}_{\#_m}\right)\right) = \frac{1}{2}\boldsymbol{x}_{\#_m}^T\left(\boldsymbol{K}'_{\#\#_m}\right)^{-1}\frac{\partial\boldsymbol{K}'_{\#\#_m}}{\partial\theta_j}\left(\boldsymbol{K}'_{\#\#_m}\right)^{-1}\boldsymbol{x}_{\#_m}$$

$$-\frac{1}{2}\mathrm{tr}\left(\left(\boldsymbol{K}'_{\#\#_m}\right)^{-1}\right)\frac{\partial\boldsymbol{K}'_{\#\#_m}}{\partial\theta_j} \tag{5.26}$$

and then use a gradient-based optimization method in order to maximize over them.

Chapter 6

Experimental Results

6.1 Artificial Dataset Template Relearning

Since our model is generative, we can generate the data from our model and test whether our learning procedure indeed recovers the templates to a good extent. We generate a dataset of two time series, 500 motifs each. The motifs are sampled from the templates with the parameters presented in the table 6.1. The functions were sampled with a homoscedastic white Gaussian noise with $\sigma^2 = 1$. Note that templates 4 and 5 have identical mark generating process, but a very different timestamp intensity. The templates are visualized in the Figure 6.1

m	Intensity: λ_m	Function Sampled	A_{m1}	A_{m2}	A_{m3}	A_{m4}	A_{m5}
1	5	$f(t) = 0$	0.5	0.125	0.125	0.125	0.125
2	5	$f(t) = \cos(2\pi t)$	0.125	0.5	0.125	0.125	0.125
3	5	$f(t) = 2t^2 - 2/3$	0.125	0.125	0.5	0.125	0.125
4	5	$f(t) = \sin(2\pi t)$	0.125	0.125	0.125	0.5	0.125
5	50	$f(t) = \sin(2\pi t)$	0.125	0.125	0.125	0.125	0.5

Table 6.1: Parameters used for the artificial dataset templates.

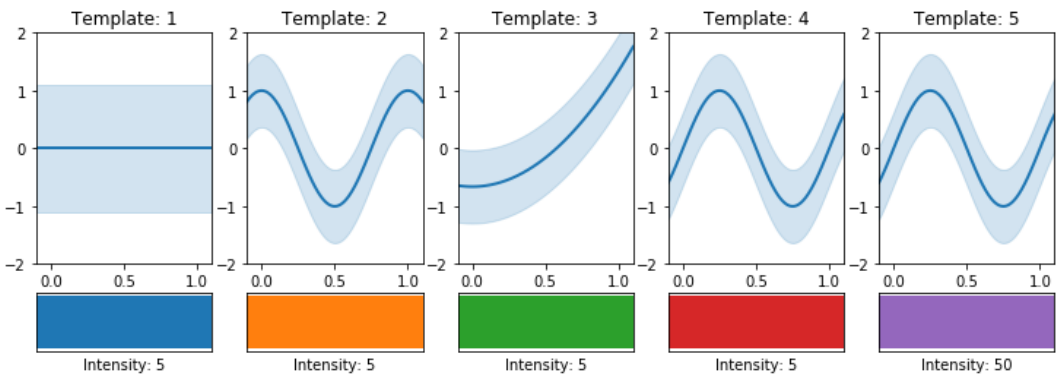

Figure 6.1: Artificial dataset true motif templates.

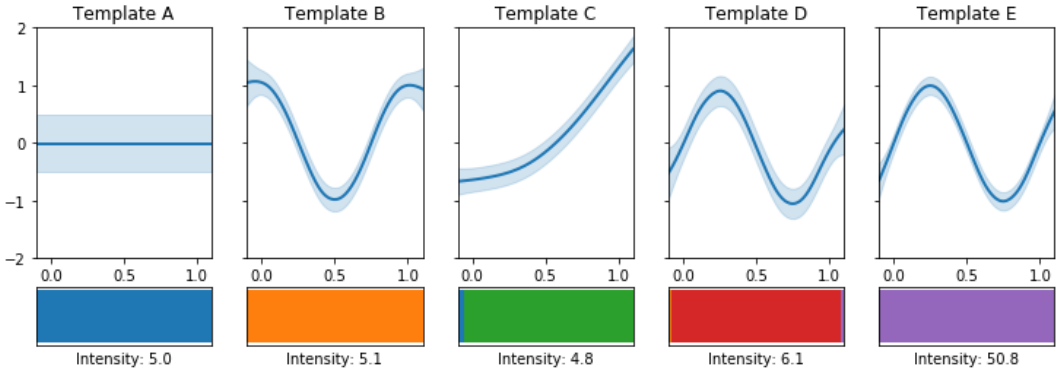

Figure 6.2: Artificial dataset learned motif templates.

We've specified the number of learnable templates to 5 (we refer to them as A, B, C, D and E in order to avoid confusion with the true templates) and ran our training algorithm on the dataset until convergence. The final templates learned are presented in the Figure 6.2. The color bar below is a visualization that allows to understand how pure are the clusters. Specifically, it represents the breakdown of the Viterbi decoding from the final step of the E by the proportion of the true templates. To give an example, all motifs identified as template A by the Viterbi decoding actually came from the same template, template 1. Whereas of motifs identified as template C in the final, 0.04 came from template 1 and the rest from template 3. Note that the model

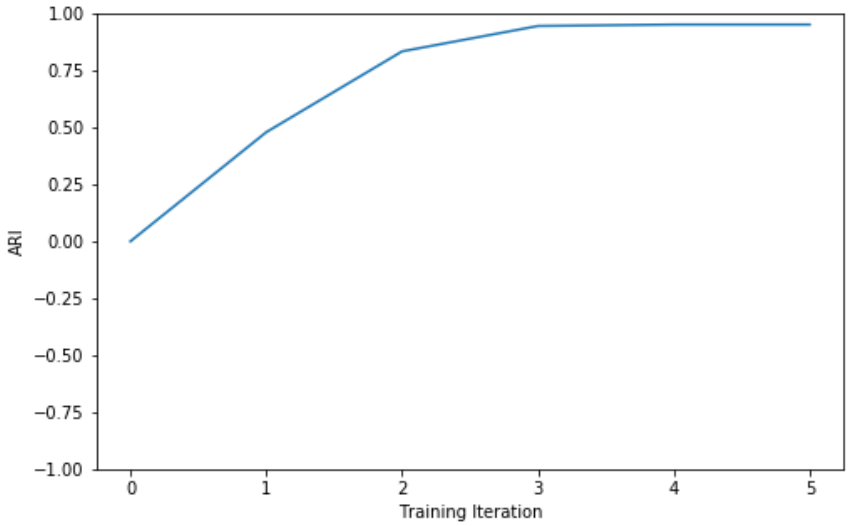

Figure 6.3: Artificial dataset Adjusted Rand Index evolution over EM steps. Step 0 identifies intialization.

is unidentifiable since any permutation of the templates produces exactly the same likelihoods. We have the permuted the learned templates for the purpose of the figure to allow an easier visual comparison.

Since we know the true templates that the motifs were generated from, we can use the Adjusted Rand index (ARI) (Rand, 1971) in order to evaluate the performance. The plot of the evolution of the ARI is presented in the Figure 6.3. The final ARI achieved by the model is equal to 0.95. Note that the templates 4 and 5 were classified in almost pure clusters, despite the equivalent mark generating process.

6.2 MIMIC-III Dataset

We have also used our motif discovery algorithm on the MIMIC-III Dataset. MIMIC-III ('Medical Information Mart for Intensive Care') is a large, single-center database comprising patient monitoring information collected at a large tertiary care hospital. (Johnson et al., 2016) We choose creatinine laboratory measurements as our time series. As was discussed in Chapter 2, labaroty measurements are often very irregularly sampled, from once every several days to several times over the span of a single day.

We chose creatinine as it is the lab with one of the most frequent observations, and it is an important indicator of renal disfunction. Since for most of the patients, most of the days contain 0 to 1 observations, we use intervals of 9 days as our motif length. With a shorter motif length, motif is selected with posterior mean that happens to be close, whereas we want to learn trajectories.

We restrict our data to patients who have more than 3 full motifs, that is 27 days of observations. For computational reasons, we further downsample to a sample size of 5000 patients.

We initialize 25 templates, however some templates end up not having any motifs associated with them by the end of training. We present all of the final templates in Figure 6.4. Intenisty represents the parameter of the Poisson process and can be interpreted as the average number of observations per 9 days, rounded to the nearest tenth. P represents the proportion of the motif generated from the specific template in the whole training dataset. One may notice that motifs vary significantly both by intensity and by the shape of the

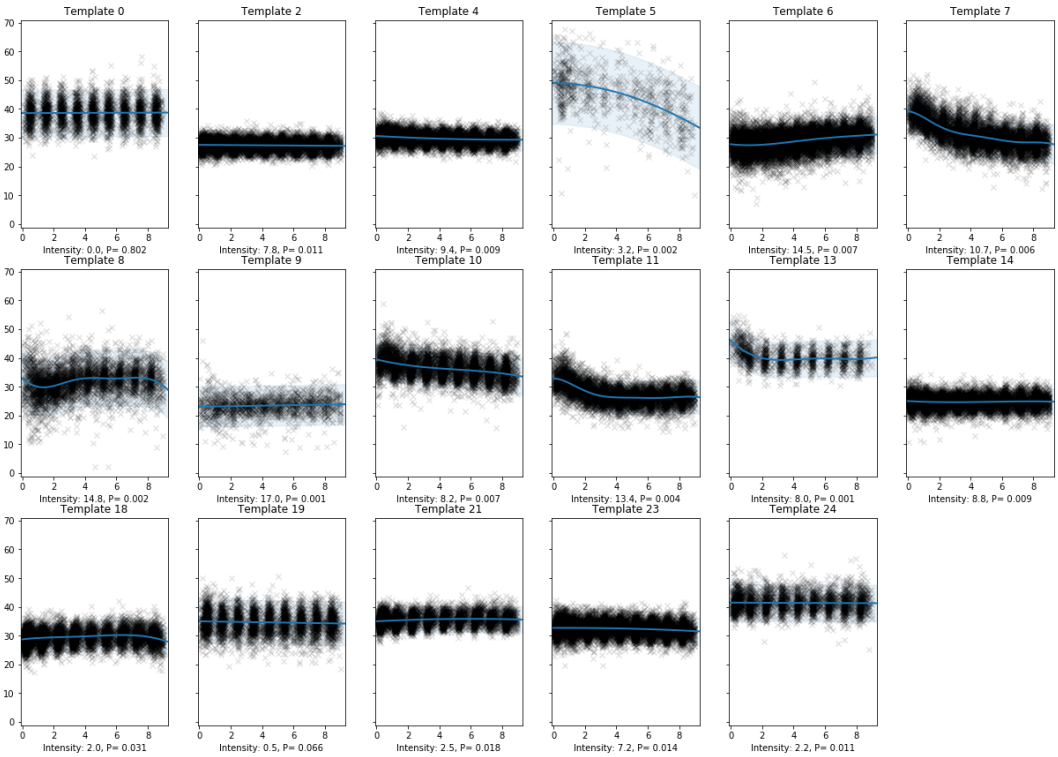

Figure 6.4: Motifs identified in the MIMIC-III creatinine lab data

observation generating process.

There are some interesting motifs identified in the data from the knowledge discovery perspective. For example, template 7 corresponds to the rapid decline in the patient's creatinine. We present two time series from the dataset with motifs superimposed in Figure 6.5. Template 7 is the one shown in red.

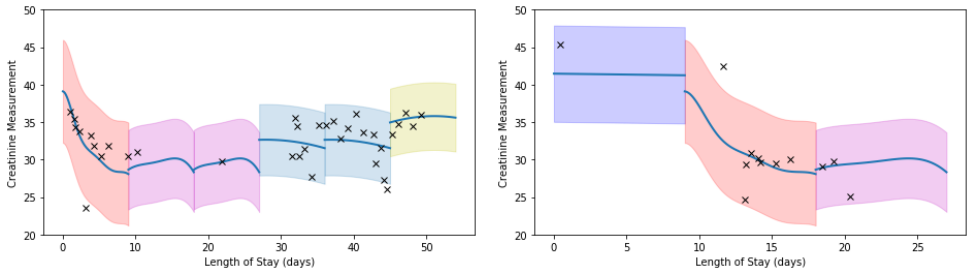

Figure 6.5: Exampled time series creatinine lab data with motifs superimposed

Chapter 7

Discussion and Conclusion

7.1 Limitations and Future Work

7.1.1 Constant Motif Length

The most important limitation of our work spawns from the use of the discrete HMM in order to model the transition between the states. Doing so limits us to the constant length of each template and to the fact that motifs can start exclusively at a multiple of that length. Both of these assumptions narrow the possible applications of our model quite significantly.

It makes most sense to apply model to the case when there is an a priori knowledge both of the constant length of the motifs and of the time of their start. Good examples of such is data that is known to have regular cycles, such as physiological processes that follow a roughly 24-hour cycle due to the circadian rhythms.

On the other hand, in the EKG data, a typical signal for the motif discovery, the length of the action potential can vary quite significantly. Furthermore, even if we approximate the average length well, without any preprocessing,

the sequence of the action potentials is more likely than not to be 'off-beat' with the beginning of the first AP being aligned to the middle of our first interval, rather than to the beginning.[1]

A natural place to look for the relaxation of this assumption is the continuous-time HMMs. (Shelton and Ciardo, 2014) Unfortunately, significantly less theory is developed for those, compared to their discrete-time counterparts. Specifically, the formulations of the Viterbi algorithm the continuous-time HMMs make assumptions such one-Gaussian-per-state emissions, which makes them inapplicable to our case. (Saerens, 1993) Furthermore, even for the continuous-time forward-backward algorithm, which has been developed (Shelton and Ciardo, 2014), the use of such complex emissions possesses a non-trivial challenge.

We shall note that one of the primary reasons we chose to use the Poisson processes in order to model the distribution of timestamps and Gaussian processes to model the distribution of marks is the fact that both of these processes generalize well to the arbitrary length intervals, which should make the relaxation of this assumption simpler. Further work is required in order to understand how the work presented in this paper can be extended to the case of the arbitrary length motifs.

[1] We should note that although we provide the EKG as an example of the case in which our assumptions break down, EKG data is generally regularly sampled. Thus, even if these assumptions were to be relaxed, it would make more sense to apply more traditional motif discovery tools, to the EKG, rather than a model that models sampling frequency, as one would be wasting power

7.1.2 Nonflexible Point Process

We use the Stationary Point Process in order to model the distribution of the timestamps. As was mentioned in Chapter 3, learning this process discards a lot of information from the data. Specifically, the sufficient statistic is the number of observations for a given time interval, which means their relative position does not impact the likelihood. Under this model, producing two time points arbitrarily close to each other is as likely as two points some space apart, as long as they are both within an interval, which is not the case in many applications, such as medical time series. Certainly, receiving two creatinine lab results with a 12-hour difference is significantly more likely than with a difference of 1 minute.

In order to model the sampling patterns in a more flexible manner, it would make sense to extend the model to use a point process different from a Poisson process. A good candidate process for such is Hawkes process, which has been previously used to model the missing data.(Shelton, Qin, and Shetty, 2018) One of the benefits of using an EM or a Hard EM algorithm for learning is that it only requires specifying a way to compute the likelihood and to maximize it over the parameters. Thus, an extension to a process such as Hawkes should be easily accomplishable in the current framework.

7.2 Conclusion

In this work, we have developed a model for unsupervised discovery of motifs in irregularly sampled continuous time series data. Our model treats the entire series as a sequence of templates from which both the observation times and the observation marks are drawn from. We have presented a training procedure that can learn the parameters of our model and have demonstrated that our procedure can indeed learn the parameters of the model for the data generated according to the model, up to unidentifiability. Lastly, we have shown the results of our motif discovery model for the MIMIC-III creatinine lab data.

References

Saria, Suchi, Andrew Duchi, and Daphne Koller (2011). "Discovering Deformable Motifs in Continuous Time Series Data". In: *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*. IJCAI'11. Barcelona, Catalonia, Spain: AAAI Press, pp. 1465–1471. ISBN: 978-1-57735-514-4. DOI: 10.5591/978-1-57735-516-8/IJCAI11-247. URL: http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-247.

Lin, Jessica, Eamonn Keogh, Stefano Lonardi, and Pranav Patel (2002). "Finding Motifs in Time Series". In: *Proceedings of the Second Workshop on Temporal Data Mining*.

Castro, Nuno and Paulo J. Azevedo (2010). "Multiresolution Motif Discovery in Time Series". In: *SDM*.

Mcmillan, Sean, Chih-Chun Chia, A Van Esbroeck, Ilan Rubinfeld, and Syed Zeeshan (2012). "ICU mortality prediction using time series motifs". In: pp. 265–268. ISBN: 978-1-4673-2076-4.

Gao, Yifeng and Jessica Lin (2018). "Efficient Discovery of Variable-length Time Series Motifs with Large Length Range in Million Scale Time Series". In:

Shokoohi-Yekta, Mohammad, Yanping Chen, Bilson Campana, Bing Hu, Jesin Zakaria, and Eamonn Keogh (2015). "Discovery of Meaningful Rules in Time Series". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '15. Sydney, NSW, Australia: ACM, pp. 1085–1094. ISBN: 978-1-4503-3664-2. DOI: 10.1145/2783258.2783306. URL: http://doi.acm.org/10.1145/2783258.2783306.

Mueen, Abdullah and Eamonn Keogh (2010). "Online Discovery and Maintenance of Time Series Motifs". In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '10. Washington, DC, USA: ACM, pp. 1089–1098. ISBN: 978-1-4503-0055-1. DOI:

10.1145/1835804.1835941. URL: http://doi.acm.org/10.1145/1835804.1835941.

Soleimani, Hossein, James Hensman, and Suchi Saria (2017). "Scalable Joint Models for Reliable Uncertainty-Aware Event Prediction". English (US). In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2017.2742504.

Li, Yuan, Jessica Lin, and Tim Oates (2012). "Visualizing Variable-Length Time Series Motifs". In: pp. 895–906. DOI: 10.1137/1.9781611972825.77. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611972825.77. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611972825.77.

Lin, Jessica, Eamonn Keogh, Li Wei, and Stefano Lonardi (2007). "Experiencing SAX: a novel symbolic representation of time series". In: *Data Mining and Knowledge Discovery* 15.2, pp. 107–144. ISSN: 1573-756X. DOI: 10.1007/s10618-007-0064-z. URL: https://doi.org/10.1007/s10618-007-0064-z.

Mueen, Abdullah, Eamonn J. Keogh, Qiang Zhu, Sydney Cash, and M. Brandon Westover (2009). "Exact Discovery of Time Series Motifs". In: *SDM*.

Minnen, David, Charles L. Isbell, Irfan Essa, and Thad Starner (2007). "Discovering Multivariate Motifs Using Subsequence Density Estimation and Greedy Mixture Learning". In: *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 1*. AAAI'07. Vancouver, British Columbia, Canada: AAAI Press, pp. 615–620. ISBN: 978-1-57735-323-2. URL: http://dl.acm.org/citation.cfm?id=1619645.1619744.

Fauber, Jacob and Christian R. Shelton (2018). "Modeling "Presentness" of Electronic Health Record Data to Improve Patient State Estimation". In: *Proceedings of Machine Learning for Healthcare*.

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag. ISBN: 0387310738.

Murphy, Kevin P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press. ISBN: 0262018020, 9780262018029.

Daley, D. J. and D. Vere-Jones (2003). *An introduction to the theory of point processes. Vol. I*. Second. Probability and its Applications (New York). New York: Springer-Verlag, pp. xxii+469. ISBN: 0-387-95541-0.

Ross, S.M. (1996). *Stochastic processes*. Wiley series in probability and statistics: Probability and statistics. Wiley. ISBN: 9780471120629. URL: https://books.google.com/books?id=ImUPAQAAMAAJ.

Shelton, Christian R. and Gianfranco Ciardo (2014). "Tutorial on Structured Continuous-Time Markov Processes". In: *J. Artif. Intell. Res.* 51, pp. 725–778.

Rand, William M. (1971). "Objective Criteria for the Evaluation of Clustering Methods". In: *Journal of the American Statistical Association* 66.336, pp. 846–850. ISSN: 01621459. URL: http://www.jstor.org/stable/2284239.

Johnson, Alistair E. W., Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark (2016). "MIMIC-III, a freely accessible critical care database". In: *Scientific Data* 3, 160035 EP –. URL: https://doi.org/10.1038/sdata.2016.35.

Saerens, Marco (1993). "A continuous-time dynamic formulation of Viterbi algorithm for one-Gaussian-per-state hidden Markov models". In: *Speech Communication* 12.4, pp. 321 –333. ISSN: 0167-6393. DOI: https://doi.org/10.1016/0167-6393(93)90081-U. URL: http://www.sciencedirect.com/science/article/pii/016763939390081U.

Shelton, Christian, Zhen Qin, and Chandini Shetty (2018). *Hawkes Process Inference With Missing Data*. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16985.