Regime Shifts in Streams: Real-time Forecasting of Co-evolving Time Sequences

Yasuko Matsubara
Kumamoto University
yasuko@cs.kumamoto-u.ac.jp

Yasushi Sakurai
Kumamoto University
yasushi@cs.kumamoto-u.ac.jp

ABSTRACT

Given a large, online stream of multiple co-evolving event sequences, such as sensor data and Web-click logs, that contains various types of non-linear dynamic evolving patterns of different durations, how can we efficiently and effectively capture important patterns? How do we go about forecasting long-term future events?

In this paper, we present REGIMECAST, an efficient and effective method for forecasting co-evolving data streams. REGIMECAST is designed as an adaptive non-linear dynamical system, which is inspired by the concept of "regime shifts" in natural dynamical systems. Our method has the following properties: (a) *Effective*: it operates on large data streams, captures important patterns and performs long-term forecasting; (b) *Adaptive*: it automatically and incrementally recognizes the latent trends and dynamic evolution patterns (i.e., regimes) that are unknown in advance; (c) *Scalable*: it is fast and the computation cost does not depend on the length of data streams; (d) *Any-time*: it provides a response at any time and generates long-range future events.

Extensive experiments on real datasets demonstrate that REGIMECAST does indeed make long-range forecasts, and it outperforms state-of-the-art competitors as regards accuracy and speed.

Categories and Subject Descriptors: H.2.8 [**Database management**]: Database applications–*Data mining*
Keywords: Time series; Real-time forecasting; Regime shifts;

1. INTRODUCTION

Time-series event analysis is an important topic that has attracted huge interest in countless domains such as sensor network monitoring [10, 24], financial and economic analysis [37, 35], social activity mining [13, 18], online text [6, 7], and medical and health record data analysis [9, 4, 19]. In the real applications, a massive volume and variety of time-stamped data is generated and collected at a very high logging rate, and this situation provides various new demands and opportunities for data scientists and analysts.

In this paper, we focus on the most important and challenging time-series analysis task, namely, the real-time forecasting of co-evolving event streams, and specifically, we present REGIMECAST, an efficient forecasting method for co-evolving time sequences. REGIMECAST is designed as an adaptive non-linear dynamical system, inspired by the concept of "regime shifts" in natural ecological systems [31, 3].

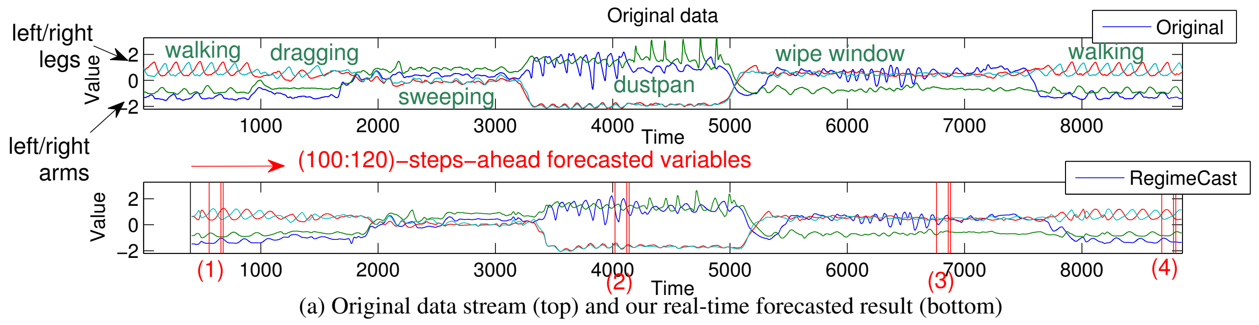
KDD '16, August 13-17, 2016, San Francisco, CA, USA
© 2016 ACM. ISBN 978-1-4503-4232-2/16/08...$15.00
DOI: http://dx.doi.org/10.1145/2939672.2939755

Intuitively, the problem we wish to solve is as follows:

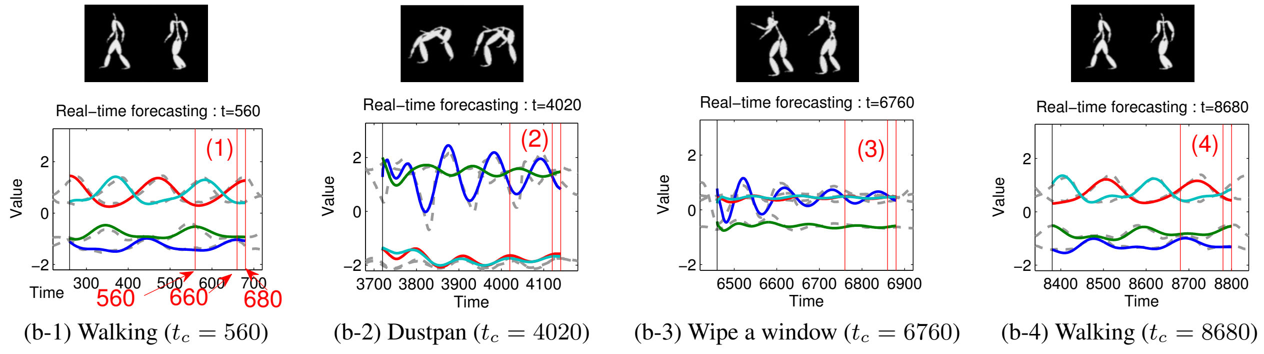
INFORMAL PROBLEM 1. **Given** *a data stream X, which consists of event entries of d-dimensional data, i.e., $X = \{\boldsymbol{x}(1), \ldots, \boldsymbol{x}(t_c)\}$, where t_c is the current time tick,* **forecast** *l_s-steps-ahead future event $\boldsymbol{x}(t_c + l_s)$, immediately, at any point in time.*

What is real-time forecasting over data streams? Assume that we have a data stream X, which is a semi-infinite sequence of event entries of d-dimensional data, $\{\boldsymbol{x}(1), \boldsymbol{x}(2), \ldots, \boldsymbol{x}(t_c), \ldots\}$, where $\boldsymbol{x}(t_c)$ is the most recent event, and t_c increases with every new time tick. So, what would be the real requirements for forecasting data streams? We need an efficient algorithm that reports upcoming unknown future events, immediately, at any point in time, while discarding redundant information. In short, the ideal solution would satisfy the following requirements.

- **l_s-steps-ahead forecasting**: Assume that we have multiple sensors (e.g., accelerometers in automobiles or motion sensors in public areas), which generate d-dimensional co-evolving event entries, every millisecond. Given such a huge collection of event sequences, we want to forecast future events, efficiently and effectively, (e.g., to perform traffic accident prevention or home security monitoring and intrusion detection). Basically, our forecasting algorithm should be (a) *long-term*: it should generate more than, say, $l_s = 100$ steps-ahead predictions, every time tick; and (b) *continuous*: it should estimate future events, smoothly and continuously, by monitoring current trends and dynamic evolving patterns. With respect to the first property (a), it is very important to predict long-range future events. For example, if we receive an alert message, such as *"there will be a traffic accident in a second"*, it would be too late to avoid the accident. That is, faced with real-time forecasting problems, short-term (say, $1, 2, 3, \ldots$-steps-ahead) prediction is *meaningless*. The second property, i.e., (b) continuous processing is a requirement for data streams. Unlike the traditional time-series forecasting approaches (e.g., ARIMA), which are *static* (as opposed to dynamic), the ideal method should report future events, at every point in time, by capturing and updating the current dynamic evolving patterns, flexibly and continuously.
- **Adaptive non-linear modeling**: As we will see later (e.g., in Figure 1 (a)), real-life time-series event streams contain various types of distinct, dynamic evolving patterns of different durations, e.g., the walking and house-cleaning motion patterns in Figure 1 (a), the normal and abnormal patterns seen in network traffic monitoring streams, and the patterns



(a) Original data stream (top) and our real-time forecasted result (bottom)



(b-1) Walking ($t_c = 560$) (b-2) Dustpan ($t_c = 4020$) (b-3) Wipe a window ($t_c = 6760$) (b-4) Walking ($t_c = 8680$)

Figure 1: Forecasting power of REGIMECAST for a "house cleaning" motion event stream: (a) The original data (top), our (100:120)-steps-ahead forecasted results (bottom), and (b) snapshots of video clips (top) and REGIMECAST outputs (bottom) at four different time ticks. Given an original data stream, REGIMECAST incrementally and automatically identifies typical motions (such as "walking" and "using a dustpan"), and also forecasts long-range future events (e.g., in figure (b-1), at the current time tick $t_c = 560$, it makes (100:120)-steps-ahead predictions, i.e., from $t_s = 660$ to $t_e = 680$). We note that our algorithm requires no prior training and no hint regarding the number of motions (i.e., regimes). For a more detailed explanation of our forecasting mechanism, please see Figure 3.

of growth, maturity and decline seen in product sales or Web-based online user activities (e.g., Figure 7). Hereafter, we refer to such a distinct dynamic evolving pattern as a *"regime"*. As shown in Figure 1 (a), a single event stream X consists of multiple regimes, where there are some abrupt changes of regimes, namely, *"regime shifts"*, such as, $t = 1,000$, from a walking motion to a dragging motion. Consequently, our algorithm should identify any sudden discontinuity in an event stream, and recognize the current regime, immediately, so that it can forecast l_s-steps-ahead future events, adaptively, at any time. We should also emphasize that these regimes are unknown in advance, and thus the algorithm should incrementally learn all arrived events, summarize them into a compact yet powerful representation. We thus propose using an *adaptive* non-linear modeling approach that satisfies all the above requirements. We will discuss this further in subsection 3.2.

Preview of our results. Figure 1 (a) shows the original event stream of a "house cleaning" motion (top) and the result we obtained with REGIMECAST (bottom). The original data stream consists of four co-evolving motion capture sensors: left/right arms and left/right legs. As shown in the figure, the original event stream is composed of several consecutive motions, including "walking", "dragging a mop", "using a dustpan" and "wiping a window".

The bottom of Figure 1 (a) shows our (100:120)-steps-ahead forecasting results, and specifically, REGIMECAST forecasts (100:120)-steps-ahead future events, for every (reporting) time [1]. Figure 1 (b-1)-(b-4) shows original video clips (top) and snapshots of REGIMECAST outputs (bottom) at four different time ticks (i.e., $t_c = 560$, 4020, 6760, 8680, respectively). In figures (b-1)-(b-4), the estimated events are shown by bold colored lines, and the original sequences are shown by dotted gray lines. The red vertical axes show the time ticks $\{t_c, t_s, t_e\}$, where, t_c is the current time tick. At each time tick t_c, the algorithm tries to forecast (100:120)-steps-ahead future events, that is, from t_s to t_e. For example, in the figure (b-1), at the current time tick $t_c = 560$, it predicts future events from $t_s = 660$ to $t_e = 680$. We will see this snapshot-plot so often that we have given it a name, "REGIMESNAP". Also, We will explain our detailed forecasting mechanism later in Figure 3 in subsection 4.1.

Consequently, given a large collection of co-evolving events, our method identifies important motion patterns (e.g., "walking" and "wiping"), and forecasts l_s-steps-ahead future events, incrementally and automatically. Most importantly, our algorithm requires no prior training and no hint regarding the number of motions (i.e., regimes) or trend-changing (i.e., regime shift) positions.

Contributions. In this paper, we present REGIMECAST, which is a real-time forecasting method for large co-evolving data streams. Our method has the following desirable properties:

1. **Effective**: it operates on a large collection of time series and performs long-term event forecasting.
2. **Adaptive**: it incrementally and automatically recognizes latent trends and non-linear dynamic patterns (i.e., regimes) while these patterns are unknown in advance.
3. **Scalable**: the computation cost does not depend on data stream length.
4. **Any-time**: it generates long-range predictions, immediately, at any point in time.

Outline. The rest of the paper is organized in the conventional way: Next we describe related work, followed by our proposed model and algorithms, experiments, discussion and conclusions.

[1]The reporting-time window l_p can be set by the user (e.g., $l_p = 20$). If it is set at $l_p = 1$, the algorithm reports a l_s-steps-ahead single event, every 1 time tick.

2. RELATED WORK

In recent years, there has been an explosion of interest in mining time-stamped data [29, 30, 24, 23, 33]. Similarity search and pattern discovery in time sequences have attracted huge interest [32, 21, 28, 24, 25, 20, 2]. Rakthanmanon et al. [27] proposed a similarity search algorithm for "trillions of time series" under the dynamic time warping (DTW) distance. The work in [17] proposed a scalable method for forecasting complex time-stamped events, while, [14] developed AutoPlait, which is a fully-automatic mining algorithm for co-evolving sequences. Traditional approaches applied to time series data mining typically use linear method, such as auto-regression (AR), autoregressive integrated moving average (ARIMA), linear dynamical systems (LDS), Kalman filters (KF) and their derivatives including AWSOM [22], TBATS [12], PLiF [11] and TriMine [17]. Note that these methods are fundamentally unsuitable for our setting: they are all based on *linear* equations, and are thus incapable of modeling data governed by non-linear equations. Existing non-linear methods for forecasting tend to be hard to interpret and cannot be used easily for long-term prediction, because they rely on a nearest-neighbor search [36, 1]. Non-linear mining and analyses of epidemics and social activities have attracted a lot of interest [4, 15, 16, 26, 18]. The work described in [18] studied the rise and fall patterns in the information diffusion process through online social media.

3. DESIGN PHILOSOPHY OF REGIMECAST

In this section, we present the fundamental concepts and design philosophy behind our method.

3.1 Regime shifts in natural systems

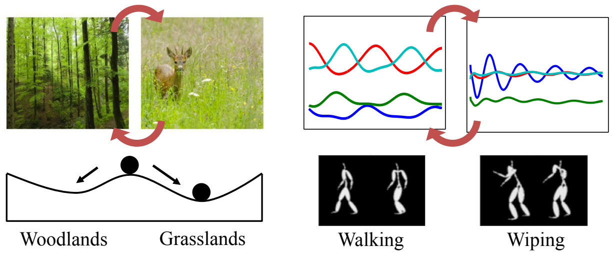
Large, abrupt, long-lasting changes in the structure of complex systems – regime shift– it is a key concept for understanding real-world dynamical phenomena. Because of its importance and substantial impact on human society, regime shift has been widely studied over recent decades in various research domains, especially, in the field of environmental ecology [31, 5, 3, 34].

In ecology, a *regime* is defined as a characteristic behavior of a natural phenomenon over time, and a *regime shift* implies an abrupt change, in relation to the duration of a regime, from one characteristic behavior to another [5]. Typically, regime shifts are triggered either by internal processes, e.g., weakening of stability, or by external shocks that exceed the stabilizing capacity of a system. Also, these transition patterns are sometimes gradual, rather than abrupt [34].

For example, let us consider biological ecosystems, such as lakes, coral reefs and woodlands, where biotic/abiotic components (e.g., fishes, corals, water, mineral soils) are interacting with each other, as a complex, adaptive non-linear system. In such a natural ecosystem, we can observe several sudden changes of states. For example, one of the most dramatic state shifts is the the sudden loss of transparency and vegetation in shallow lakes caused by human-induced eutrophication. Similarly, in a marine ecosystem, there is a transition from a coral dominated system to an fleshy brown macroalgae dominated system, which is possibly induced by hurricane or climate change. Also, as described in Figure 2 (a), a woodland ecosystem can be alternated with a grassy open landscape. Landscapes can be kept open by herbivores (or, fires and tree cutting), while woodlands, once established, are stable because adult trees cannot be destroyed by herbivores [31] [2].

Theoretical analysis and mathematical modeling. The most basic and minimum form of an ecological dynamical system can be described with the following ordinary differential equation (ODE) [31]:

[2] Image courtesy of dan at FreeDigitalPhotos.net.



(a) Ecological system (b) Motion sensors

Figure 2: Illustration of regime shifts in an ecological system vs. sensor stream: (a) Regime shifts between two alternative stable states: woodlands and grasslands, and (b) regime shifts in motion sensors, between walking and wiping activities.

$$\frac{ds(t)}{dt} = a_0 + a_1 s(t) + a_2 f(s(t)), \quad (1)$$

where, $s(t)$ shows the time-evolving ecosystem property, such as nutrients or soils at time tick t, and a_0 describes an environmental factor that promotes $s(t)$, (e.g., nutrient loading), a_1 is the rate at which $s(t)$ grows/decays in the system (e.g., nutrient removal rate, for $a_1 < 0$). The last term, a_2 represents the rate at which $s(t)$ recovers again as a function $f(s(t))$, (e.g., internal nutrient recycling), and the function f can cause alternative stable states.

Because of its simplicity and generality, the above dynamical equation and the concept of regime shifts have been applied outside ecology, to sociology, economics, political science and beyond. Next we present our proposed method, REGIMECAST, which is inspired by the regime shifts in natural dynamical systems.

3.2 Proposed model

Given a large event stream X, which contains various dynamic evolving patterns (i.e., regimes), our goal is to find important trends and summarize them as a compact but powerful and adaptive representative model, to perform, long-term event forecasting.

So, what exactly are the most important properties for forecasting the event stream X? Specifically, we want to describe the following three important properties:

- (**P1**) Latent non-linear dynamics
- (**P2**) Regime shifts in event streams
- (**P3**) Nested structure

Similar to the natural dynamical systems described in subsection 3.1, real event streams evolve naturally over time, depending on many latent factors, such as traffic, weather and driver's condition for automobile sensors, or user's preferences and customs for online Web-click activities. We need to capture the complicated and hidden dynamics of real time-series events that involve *non-linear* phenomena. To handle (**P1**), we propose using latent non-linear differential equations. We also want to automatically detect the switching positions of dynamic evolution patterns, that is, (**P2**) regime shifts in event streams. Figure 2 compares regime shifts in an ecological system vs. an event stream (i.e., a motion sensor stream). We conjecture that event streams behave as a complex, adaptive non-linear dynamical system in the same way that biotic/abiotic components in a natural ecosystem evolve and interact over time, where there are multiple alternative stable states (i.e., regimes), that can be switched to one another. We should also note that real event streams are composed of multi-level nested systems, each of which has a different time scale, and can be integrated as a part of a larger system, that is, we need (**P3**). For example, as shown in Figure 7, Web-click activities exhibit long-term evolution such as ten-year patterns of growth and decline, as well as short-term, say weekly, daily and hourly patterns. Similar behavior can be observed in natural systems, e.g., both long-term climate change and sudden hurricanes can affect a coral reef system [31]. We thus propose using a multi-level nested modeling structure that enables more effective modeling and forecasting.

Now, we describe our concept in steps, adding complexity.

3.2.1 Latent non-linear dynamical systems (P1)

We begin with the simplest case, where we have only a single dynamical pattern (i.e., regime), e.g., walking motion in a sensor event stream, where, there are no regime shifts in the sequence (**P1**). In our basic model, we assume that there are two classes of time-evolving activities:

- $\boldsymbol{s}(t)$: Potential activity, i.e., k-dimensional latent activity at time tick t, $(\boldsymbol{s}(t) = \{s_i(t)\}_{i=1}^{k})$.
- $\boldsymbol{v}(t)$: Estimated event, i.e., d-dimensional actual activity that can be observed at time tick t, $(\boldsymbol{v}(t) = \{v_i(t)\}_{i=1}^{d})$.

Here, we can only observe the actual event $\boldsymbol{v}(t)$ at time tick t, (such as, actual variables generated by d sensors), while $\boldsymbol{s}(t)$ is a hidden vector that evolves over time as a dynamical system. Consequently, a single regime can be described with the following equations:

MODEL 1. *Let $\boldsymbol{s}(t)$ be the k-dimensional potential activity at time tick t, and $\boldsymbol{v}(t)$ be the d-dimensional estimated event at time tick t. Our base model is governed by the following equations,*

$$\frac{d\boldsymbol{s}(t)}{dt} = \boldsymbol{p} + \mathbf{Q}\boldsymbol{s}(t) + \mathcal{A}\mathbf{S}(t) \qquad (2)$$

$$\boldsymbol{v}(t) = \boldsymbol{u} + \mathbf{V}\boldsymbol{s}(t) \qquad (3)$$

with initial conditions $\boldsymbol{s}(0) = \boldsymbol{s}_0$, where, $d\boldsymbol{s}(t)/dt$ is the derivative with respect to time tick t and $\mathbf{S}(t)$ shows the quadratic form matrix of $\boldsymbol{s}(t)$, i.e., $\mathbf{S}(t) = \boldsymbol{s}(t)^T\boldsymbol{s}(t)$.

Here, $\boldsymbol{p}$, $\mathbf{Q}$, $\mathcal{A}$ describe the potential activity $\boldsymbol{s}(t)$, each of which captures linear, exponential, and non-linear dynamical activities [3], while, $\boldsymbol{u}$, $\mathbf{V}$ show the observation projection, which generates the estimated event $\boldsymbol{v}(t)$ at each time tick t. Also note that the non-linear activity tensor $\mathcal{A}$ should be *sparse*, to eliminate the complexity of the system. We will explain this later in section 4.

Consequently, we have the following:

DEFINITION 1 (SINGLE REGIME PARAMETER SET: $\boldsymbol{\theta}$). *Let $\boldsymbol{\theta}$ be a parameter set of a single non-linear dynamical system, i.e., $\boldsymbol{\theta} = \{\boldsymbol{s}_0, \boldsymbol{p}, \mathbf{Q}, \mathcal{A}, \boldsymbol{u}, \mathbf{V}\}$. We refer to $\boldsymbol{\theta}$ as a single regime parameter set.*

3.2.2 Regime shifts in event streams (P2)

Our next step is to answer the most important question, namely, how can we describe regime shifts in an event stream (**P2**)? Assume that we have a sensor stream that consists of $c = 2$ regimes, e.g., walking and wiping motions (Figure 2 (b)). We want an adaptive model that can generate c distinct time-evolving patterns, such as walking and wiping motions, where these patterns appear interchangeably, that is, the regimes can be switched to others. So what would be the best way to describe the dynamic evolving regime shift patterns among c distinct regimes? We thus introduce an additional time-evolving activity, namely,

- $\boldsymbol{w}(t)$: Regime activity, i.e., c-dimensional latent regime shift activity among c regimes at time tick t.

[3] In this paper, for simplicity, we use a quadratic function for the non-linear activity tensor $\mathcal{A}$.

Here, $\boldsymbol{w}(t)$ describes the dynamical activity for each i-th regime $(1 \le i \le c)$ at time tick t.

Consequently, we extend Model 1, and propose the following model, which is designed as a complex and adaptive dynamical system:

MODEL 2. *Let $\boldsymbol{s}_i(t)$ be the k dimensional latent activity of i-th regime at time tick t (i.e., $\boldsymbol{s}_i(t) = \{s_{ij}(t)\}_{j=1}^{k}$), $\boldsymbol{w}(t)$ be the contribution strength of each i-th regime at time tick t (i.e., $\boldsymbol{w}(t) = \{w_i(t)\}_{i=1}^{c}$), and $\boldsymbol{v}(t)$ be the d-dimensional estimated event at time tick t. Our full model can be described as the following equations,*

$$\frac{d\boldsymbol{s}_i(t)}{dt} = \boldsymbol{p}_i + \mathbf{Q}_i\boldsymbol{s}_i(t) + \mathcal{A}_i\mathbf{S}_i(t) \quad (i = 1, \ldots, c) \qquad (4)$$

$$\frac{d\boldsymbol{w}(t)}{dt} = \boldsymbol{r}(t) \qquad (5)$$

$$\boldsymbol{v}(t) = \sum_{i=1}^{c} w_i(t)\,[\boldsymbol{u}_i + \mathbf{V}_i\boldsymbol{s}_i(t)] \qquad (6)$$

where, $d\boldsymbol{w}(t)/dt$ shows the derivative with respect to time tick t.

In Model 2, we need an additional parameter, $\boldsymbol{r}(t)$, which is a c-dimensional vector at time tick t. Let $\mathbf{R}$ be a parameter set of regime shift dynamics, i.e., $\mathbf{R} = \{\boldsymbol{r}(t)\}_{t=1}^{t_c}$, where t_c shows the length of the event sequence. We refer to $\mathbf{R}$ as a regime shift matrix. Please note that if there is a single regime (i.e., $c = 1$), our full model is identical to Model 1. Consequently, we have the following:

DEFINITION 2 (REGIME SET: $\boldsymbol{\Theta}$). *Let $\boldsymbol{\Theta}$ be a parameter set of regime set, namely, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_c, \mathbf{R}\}$, that describes non-linear co-evolving patterns of c regimes.*

3.2.3 With nested structure (P3)

Until now, we have assumed a single-level dynamical system, which consists of a regime set $\boldsymbol{\Theta}$. In reality, however, we should consider multi-level, nested structure to describe multi-scale dynamical activities, such as long-term evolution (e.g., ten-year pattern) and short-term (e.g., weekly) dynamical events in online activities. So, is there any simple solution to capture this phenomenon? To handle (**P3**), we propose using multi-level regime set $\mathcal{M} = \{\boldsymbol{\Theta}^{(1)}, \boldsymbol{\Theta}^{(2)}, \ldots\}$, where each regime set $\boldsymbol{\Theta}^{(i)}$ in $\mathcal{M}$ describes a dynamical system at a different time scale. The idea is that, given a set of multi-level regimes in $\mathcal{M}$, we generate the local estimated event $\boldsymbol{v}^{(i)}(t)$ at the i-th level. The estimated event $\boldsymbol{v}(t)$ can be organized simply by gathering all the local event set together.

Full model parameter set of RegimeCast. Our complete model consists of the following:

DEFINITION 3 (COMPLETE SET OF REGIMECAST: $\mathcal{M}$). *Let $\mathcal{M}$ be a full parameter set, namely, $\mathcal{M} = \{\boldsymbol{\Theta}^{(1)}, \ldots, \boldsymbol{\Theta}^{(h)}\}$, that describes multi-level time evolving patterns.*

4. STREAMING ALGORITHM

We now introduce our full algorithm, namely, REGIMECAST, which is an efficient and effective forecasting method for large co-evolving event streams.

4.1 Problem formulation

We first define some key concepts.

DEFINITION 4 (EVENT STREAM: X). *Let X be a data stream that consists of event entries of d-dimensional data, i.e., $X = \{\boldsymbol{x}(1), \ldots, \boldsymbol{x}(t_c)\}$, where t_c is the current time tick. We refer to X as an event stream.*

Table 1: Symbols and definitions.

Symbol	Definition
d	Number of dimensions
t_c	Current time tick
X	d co-evolving event stream, i.e., $X = \{\boldsymbol{x}(1), \ldots, \boldsymbol{x}(t_c)\}$
$\boldsymbol{x}(t)$	d-dimensional event at time tick t, i.e., $\boldsymbol{x}(t) = \{x_i(t)\}_{i=1}^{d}$
$\boldsymbol{s}(t)$	Potential activity at time tick t, i.e., $\boldsymbol{s}(t) = \{s_i(t)\}_{i=1}^{k}$
$\boldsymbol{w}(t)$	Regime activity at time tick t, i.e., $\boldsymbol{w}(t) = \{w_i(t)\}_{i=1}^{c}$
$\boldsymbol{v}(t)$	Estimated event at time tick t, i.e., $\boldsymbol{v}(t) = \{v_i(t)\}_{i=1}^{d}$
X_C	Current window, i.e., $X_C = X[t_m : t_c]$
V_F	Forecast window, i.e., $V_F = V[t_s : t_e]$
$c^{(i)}$	Number of regimes at i-th level
$\boldsymbol{\theta}_j{}^{(i)}$	Parameter set of j-th regime at i-th level
$\mathbf{R}^{(i)}$	Parameter set of regime shift matrix at i-th level
$\boldsymbol{\Theta}^{(i)}$	Full parameter set at i-th level
$\mathcal{M}$	Complete set of REGIMECAST, i.e., $\mathcal{M} = \{\boldsymbol{\Theta}^{(i)}\}_{i=1}^{h}$

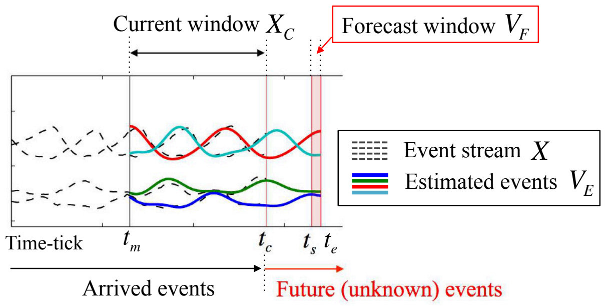


Figure 3: Illustration of REGIMESNAP: Given an event stream X (black dotted lines), our algorithm estimates the current time-series pattern V_E (colored bold lines), and reports l_s-steps-ahead future events V_F (in a red box), incrementally and continuously. Here, $X_C = X[t_m : t_c]$ is a current window (i.e., recently arrived events) and $V_F = V[t_s : t_e]$ is a l_s-steps-ahead future (i.e., unknown) event set.

Assume that we receive a new event $\boldsymbol{x}(t_c)$, continuously, and t_c increases with every new time tick. It would be convenient to treat the most recently arrived events, as a current window.

DEFINITION 5 (CURRENT WINDOW: X_C). *Let $X_C = X[t_m : t_c]$ be the subsequence of length l_c, starting from time tick t_m and ending at t_c ($1 \leq t_m \leq t_c$)* [4].

Given the current window X_C, our next step is to find the optimal regimes in $\mathcal{M}$, and predict l_s-steps-ahead future activities: $V_F = \{\boldsymbol{v}(t_s), \ldots, \boldsymbol{v}(t_e)\}$ using Model 2.

DEFINITION 6 (l_s-STEPS-AHEAD FORECAST WINDOW: V_F). *Let $V_F = V[t_s : t_e]$ denote the l_s-steps-ahead future events starting from time-tick t_s and ending at t_e ($t_c \leq t_s \leq t_e$), where, $t_s = t_c + l_s$, $t_e = t_s + l_p$, and l_p is the length of the reporting window.*

Figure 3 shows a snapshot of REGIMECAST at the current time tick t_c, namely, REGIMESNAP. Here, the black dotted lines show the original event stream X, which consists of a $d = 4$ dimensional event sequence. The bold colored lines show our estimated events V_E from time tick t_m to t_e. Note that the subsequence from t_c to t_e is future (unknown) events, and we need to estimate these hidden dynamical patterns, incrementally and continuously.

[4]We set $l_c = 3 \cdot l_s$ in our setting.

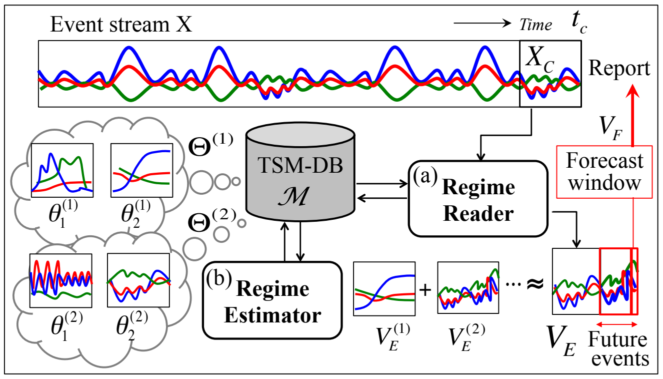


Figure 4: Overview of the REGIMECAST algorithm: Given an event stream X, it first extracts the current window X_C, and then (a) searches for the optimal regime patterns in the time-series model database, and generates the l_s-steps-ahead events V_F. If there is a new (i.e., unknown) regime pattern in X_C, (b) it also estimates the new regime model parameter set $\boldsymbol{\theta}$, and inserts it into the database.

Consequently, given an event stream X, our goal is to capture the current time-evolving patterns in X_C, as a complex, non-linear dynamical system, and predict the l_s-steps-ahead forecast window V_F, at any point in time. We formally define our problem as follows:

PROBLEM 1. **Given** *a data stream $X = \{\boldsymbol{x}(1), \ldots, \boldsymbol{x}(t_c), \ldots\}$ at regular time intervals,* **report** *l_s-steps-ahead future events V_F at any point in time, i.e.,*

- *identify the optimal regime patterns in X_C,*
- *update the model parameter set $\mathcal{M}$ that describe the dynamic patterns in X_C,*
- *report l_s-steps-ahead future events, i.e., V_F.*

4.2 Overview

We now introduce our forecasting algorithm, REGIMECAST, which consists of the following algorithms.

- REGIMEREADER: Estimates good regime dynamics and generates the estimated events $V_E = V[t_m : t_e]$, when we are given the current window X_C and the model parameters $\boldsymbol{\Theta}$.
- REGIMEESTIMATOR: Estimates a good regime parameter set $\boldsymbol{\theta}$ for X_C, when the current window X_C contains a new regime pattern (if required).
- REGIMECAST: Estimates the optimal event set $V_E{}^{(i)}$ at each level i ($i = 1, \ldots, h$), computes estimated event event set $V_E = V_E{}^{(1)} + V_E{}^{(2)} + \ldots$, and reports the l_s-steps-ahead future events i.e., forecast window V_F. It also maintains the full model parameter set $\mathcal{M}$.

Figure 4 illustrates how the REGIMECAST algorithm works. Given an event stream $X = \{\boldsymbol{x}(1), \ldots, \boldsymbol{x}(t_c)\}$, where t_c is the current time tick, our algorithm incrementally extracts the current window X_C, estimates the current regime pattern $V_E = V_E{}^{(1)} + V_E{}^{(2)} + \ldots$, and reports l_s-steps-ahead future events V_F, at any point in time. It also maintains the time-series model database (TSM-DB), and updates model parameters in $\mathcal{M}$, if required.

4.3 Proposed algorithms

Here we present our proposed algorithms in steps. To simplify the discussion, let us focus on a simple step first, where we have a single-level window X_C and a regime parameter set $\boldsymbol{\Theta}$ (that is, we have a single-level structure, $h = 1$).

Algorithm 1 REGIMEREADER $(X_C, \mathbf{\Theta})$

```
1: Input: current window $X_C$ and current regime parameters $\mathbf{\Theta}$
2: Output: Estimated events $V_E = V[t_m : t_e]$ and updated regimes $\mathbf{\Theta}$
3: /* (I) Individual regime estimation */
4: for $i = 1 : c$ do
5:     /* Estimate $\boldsymbol{s}'_0$ and activity $V_C{}'_i$ for $i$-th regime $\boldsymbol{\theta}_i$ */
6:     $\{\boldsymbol{\theta}_i[\boldsymbol{s}_0], V_{Ci}\} = \arg\min_{\boldsymbol{s}'_0, V_C{}'_i} ||X_C - V_C{}'_i||$; // $V_C{}'_i = f_C(\boldsymbol{s}'_0|\boldsymbol{\theta}_i)$;
7: end for
8: /* (II) Estimate regime activity at current time tick $t_c$ */
9: $\boldsymbol{w}(t_c) = \arg\min_{w_1,\dots,w_c} ||X_C - \sum_{i=1}^{c} w_i V_{Ci}||$;
10: $\boldsymbol{r}(t_c) = \boldsymbol{w}(t_c) - \boldsymbol{w}(t_c - 1)$; // Calculate regime shift variable
11: $\mathbf{R} = \mathbf{R} \cup \boldsymbol{r}(t_c)$;  $\mathbf{\Theta} = \{\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_c, \mathbf{R}\}$; // Update full parameters
12: $V_E = f_E(\mathbf{\Theta})$; // Calculate estimated event $V_E$
13: return $\{V_E, \mathbf{\Theta}\}$;
```

4.3.1 RegimeReader

Assume that we are given the current window X_C and the current regime parameter set $\mathbf{\Theta} = \{\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_c, \mathbf{R}\}$. Our first goal is to estimate the dynamical event sequence $V_E = V[t_m : t_e]$, as shown in Figure 4 (a). So, how can we estimate events V_E, given the current regime set $\mathbf{\Theta}$? The most straightforward solution would be simply to use the fixed parameters in $\mathbf{\Theta}$ and calculate $\boldsymbol{v}(t_m), \boldsymbol{v}(t_m + 1), \dots$, using Model 2. However, in real event streams, the latent trends of the current window X_C would dynamically and continuously change over time. That is, We should identify the optimized regimes in $\mathbf{\Theta}$ that describe the characteristics of the current window X_C. Most importantly, the algorithm needs to adaptively update parameters in $\mathbf{\Theta}$, so that it can describe the current activities in X_C. Algorithm 1 shows the overall procedure for the adaptive regime detection, namely, REGIMEREADER. In short, the algorithm consists of two parts, namely, (I) individual regime optimization, and (II) regime shift identification.

(I) Individual regime optimization. For each i-th regime parameters $\boldsymbol{\theta}_i \in \mathbf{\Theta}$ $(i = 1, \dots, c)$, the algorithm tries to optimize the initial condition of potential activity, i.e., $\boldsymbol{s}_0$ in $\boldsymbol{\theta}_i$, so that it minimizes the mean square errors between the original events and the estimated events, i.e., $\min ||X_C - V_{Ci}||$. Here, let $f_C(\boldsymbol{s}_0|\boldsymbol{\theta})$ be a function that generates estimated events $V_C = \{\boldsymbol{v}(t_m), \dots, \boldsymbol{v}(t_c)\}$ in Model 2, given regime parameters $\boldsymbol{s}_0$ and $\boldsymbol{\theta}$.

(II) Regime shift identification. Given a set of c optimized event sequences $\{V_{Ci}\}_{i=1}^{c}$, the algorithm estimates the latent dynamics of regime shifts at time tick t_c, and specifically, it computes regime activity $\boldsymbol{w}(t_c)$ to optimize the regime set in $\mathbf{\Theta}$, and updates regime shift matrix $\mathbf{R}$ in $\mathbf{\Theta}$, according to Equation 6 (i.e., $\min ||X_C - f_C(\mathbf{\Theta})||$). Finally, it computes the estimated event $V_E = f_E(\mathbf{\Theta})$ as the optimal events for the current window X_C.

Here, we use the Levenberg-Marquardt (LM) algorithm to minimize the mean square errors $||\cdot||$.

4.3.2 RegimeEstimator

Next, let us tackle the next question, namely, what if there is a new, unknown regime in X_C? We want to estimate the new regime parameter set $\boldsymbol{\theta}$ that describes the dynamical patterns in X_C, and insert it into the full parameter set $\mathbf{\Theta}$ (Figure 4 (b)). Since $\boldsymbol{\theta}$ consists of a large number of parameters, it is extremely expensive to optimize all the parameters simultaneously. Also, as we mentioned in subsubsection 3.2.1, the non-linear activity tensor $\mathcal{A}$ should be *sparse*, to avoid overfitting, and to simplify the dynamical patterns in a single regime. We thus propose an efficient and effective algorithm, namely, REGIMEESTIMATOR, which searches for the optimal solution in terms of both the linear and non-linear parameters. The idea is that we split parameter set $\boldsymbol{\theta}$ into two subsets, i.e.,

Algorithm 2 REGIMEESTIMATOR (X_C)

```
1: Input: current window $X_C$
2: Output: Estimated model parameter set $\boldsymbol{\theta} = \{\boldsymbol{s}_0, \boldsymbol{p}, \mathbf{Q}, \mathcal{A}, \boldsymbol{u}, \mathbf{V}\}$
3: /* Estimate linear dynamical parameters $\boldsymbol{\theta}_L = \{\boldsymbol{p}, \mathbf{Q}, \boldsymbol{u}, \mathbf{V}\}$*/
4: $\mathcal{A} = 0$; // Initialize tensor $\mathcal{A}$
5: $\{\boldsymbol{s}_0, \boldsymbol{\theta}_L\} = \arg\min_{\boldsymbol{s}'_0, \boldsymbol{\theta}'_L} ||X_C - V_C||$; //$V_C = f_C(\boldsymbol{s}'_0, \boldsymbol{\theta}'_L, \boldsymbol{\theta}_N)$
6: /* Estimate non-linear dynamical parameters $\boldsymbol{\theta}_N = \{\mathcal{A}\}$ */
7: $\{\boldsymbol{s}_0, \boldsymbol{\theta}_N\} = \arg\min_{\boldsymbol{s}'_0, \boldsymbol{\theta}'_N} ||X_C - V_C||$; //$V_C = f_C(\boldsymbol{s}'_0, \boldsymbol{\theta}_L, \boldsymbol{\theta}'_N)$
8: $\boldsymbol{\theta} = \{\boldsymbol{s}_0, \boldsymbol{\theta}_L, \boldsymbol{\theta}_N\}$; // Full parameter set
9: return $\boldsymbol{\theta} = \{\boldsymbol{s}_0, \boldsymbol{p}, \mathbf{Q}, \mathcal{A}, \boldsymbol{u}, \mathbf{V}\}$;
```

$\boldsymbol{\theta}_L = \{\boldsymbol{p}, \mathbf{Q}, \boldsymbol{u}, \mathbf{V}\}$ and $\boldsymbol{\theta}_N = \{\mathcal{A}\}$, each of which corresponds to a linear/non-linear parameter set, and try to fit the parameter sets separately. Algorithm 2 shows the steps performed by REGIMEESTIMATOR in detail. Given a current window X_C, it first set $\mathcal{A} = 0$, and estimates the initial condition $\boldsymbol{s}_0$ and linear parameters $\boldsymbol{\theta}_L$ that describe linear dynamical patterns in X_C. Here, we use the expectation-maximization (EM) algorithm to optimize the parameters. It then estimates non-linear elements in $\mathcal{A}$ to minimize the errors between X_C and potential activity V_C, using the LM algorithm. Also note that, for the non-linear tensor $\mathcal{A}$, we only estimate parameters for the diagonal elements $a_{ijk} \in \mathcal{A}$ $(i = j = k)$ to eliminate the complexity.

4.3.3 RegimeCast

Until now, we have discussed how to generate the estimated event sequence V_E at a single-level regimes $\mathbf{\Theta}$. Our final goal is to capture multi-level dynamical patterns $\mathcal{M} = \{\mathbf{\Theta}^{(1)}, \dots, \mathbf{\Theta}^{(h)}\}$, and predict l_s-steps-ahead forecast window V_F (see Figure 4). As we described in subsubsection 3.2.3, given a event stream X, we want to describe multi-level dynamical activities, e.g., yearly, weekly, and daily patterns. We thus propose multi-scale modeling approach that enables more effective forecasting. The idea is that we decompose the current window X_C into a set of multi-scale event sequences $X_C = {X_C}^{(1)} + \dots {X_C}^{(h)}$, where ${X_C}^{(i)}$ shows the current event sequence at the i-th level. Specifically, the i-th level sequence can be computed as follows: ${X_C}^{(i)} = g(X_C - \sum_{j=1}^{i} {X_C}^{(j)}|H(i))$, where $g(\cdot|t)$ denotes the moving average of length t. [5]

The detailed REGIMECAST algorithm is shown in Algorithm 3. Given a new event $\boldsymbol{x}(t_c)$, it extracts the current window ${X_C}^{(i)}$ at the i-th level, and (I) estimates event sequence ${V_E}^{(i)}$. If there is no appropriate regime in $\mathbf{\Theta}^{(i)}$ (i.e., the error between the current window and estimated events is more than ϵ, e.g., $\epsilon = 1/2||{X_C}^{(i)}||$), (II) it creates a new regime $\boldsymbol{\theta}$ and update the regime set $\mathbf{\Theta}^{(i)}$. Finally, (III) it reports l_s-steps-ahead forecast window V_F.

Efficient event generation and dynamic point set (DPS). Since our estimation algorithm, REGIMEREADER, is based on a complex dynamical system, described in Model 2, it requires O(l_e) time to calculate the potential activity $S_E = \{\boldsymbol{s}(t_m), \dots, \boldsymbol{s}(t_e)\}$ at every time tick t_c, where l_e shows the length of S_E. However, it is a potential bottleneck for the real-time processing. We thus propose an efficient approach for the dynamic event generation. Specifically, instead of computing all events $S_E = \{\boldsymbol{s}(t_m), \boldsymbol{s}(t_m + 1), \boldsymbol{s}(t_m + 2), \dots, \boldsymbol{s}(t_e)\}$, we generate a subset of S_E, namely, a dynamic point set (DPS): $\hat{S}_E = \{\boldsymbol{s}(t_m), \boldsymbol{s}(t_m+\delta), \boldsymbol{s}(t_m+2\delta), \dots, \boldsymbol{s}(t_e)\}$, where δ shows the time interval of the potential activity (say, $\delta = 0.1 \cdot l_s$). We use a simple yet powerful, numerical integration method, namely, the forth-order Runge-Kutta method [8], to generate the dynamic point set $\hat{S}_E$:

[5]In this paper, we set $H = \{2 \cdot l_s, 1\}$.

Algorithm 3 REGIMECAST ($\boldsymbol{x}(t_c)$)

```
1: Input: a new event $\boldsymbol{x}(t_c)$ at time tick $t_c$
2: Output: $l_s$-steps-ahead future events $V_F$
3: /* Initialize forecast window $V_F = 0$ */
4: for $i = 1 : h$ do
5:     Compute ${X_C}^{(i)}$; // current window at $i$-level
6:     /* (I) Parameter fitting for regime activities */
7:     $\{{V_E}^{(i)}, \boldsymbol{\Theta}^{(i)}\} = $ REGIMEREADER$({X_C}^{(i)}, \boldsymbol{\Theta}^{(i)})$;
8:     /* (II) Regime estimation (if required) */
9:     ${V_C}^{(i)} = V^{(i)}[t_m : t_c]$; // Estimated events from $t_m$ to $t_c$
10:    if $||{X_C}^{(i)} - {V_C}^{(i)}|| > \epsilon$ then
11:        $\boldsymbol{\theta} = $ REGIMEESTIMATOR$({X_C}^{(i)})$; $\boldsymbol{\Theta}^{(i)} = \{\boldsymbol{\Theta}^{(i)} \cup \boldsymbol{\theta}\}$;
12:    end if
13: end for
14: /* (III) $l_s$-steps-ahead future event generation */
15: $V_E = {V_E}^{(1)} + \ldots, {V_E}^{(h)}; V_F = V[t_s : t_e]$;
16: return $V_F$;
```

$$\boldsymbol{s}(t+\delta) = \boldsymbol{s}(t) + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) + O(\delta^5) \quad (7)$$

where, we define $d\boldsymbol{s}(t)/dt = F(\boldsymbol{s}(t))$, and $K_1 = \delta F(\boldsymbol{s}(t))$, $K_2 = \delta F(\boldsymbol{s}(t) + \frac{1}{2}K_1)$, $K_3 = \delta F(\boldsymbol{s}(t) + \frac{1}{2}K_2)$, $K_4 = \delta F(\boldsymbol{s}(t) + K_3)$. Consequently, the algorithm requires $O(l_e/\delta)$ time to compute $\hat{S}_E$, where l_e/δ shows the length of $\hat{S}_E$.

Theoretical analysis. Let l_e be the length of the estimate event set V_E, and c be the number of regimes in $\mathcal{M}$.

LEMMA 1. *The computation time of* REGIMECAST *is at least $O(c \cdot l_e/\delta)$ time per time tick, at most $O(c \cdot l_e/\delta + l_c)$ time per time tick.*

PROOF. For each time tick t_c, REGIMEREADER requires $O(c \cdot l_e/\delta)$ time to estimate c optimal regimes V_E. If there is a new regime in the current window X_C, REGIMEESTIMATOR requires $O(l_c)$ to estimate parameter set $\boldsymbol{\theta}$. Thus, we have at least $O(c \cdot l_e/\delta)$ time at most $O(c \cdot l_e/\delta + l_c)$ time per time tick. ☐

5. EXPERIMENTS

In this section we demonstrate the effectiveness of REGIMECAST with real event streams. The experiments were designed to answer the following questions:

Q1 *Effectiveness*: How successful is our method in forecasting long-term events in given input streams?

Q2 *Accuracy*: How well does our method forecast future event entries?

Q3 *Scalability*: How does our method scale in terms of computational time?

Our experiments were conducted on an Intel Core i7-3770K 3.50GHz with 32GB of memory, running Linux. We normalized the values of each dataset so that they had the same mean and variance (i.e., z-normalization), and set $k = 4$ for all datasets.

5.1 Q1: Effectiveness

We demonstrate the forecasting power of REGIMECAST in terms of capturing important patterns of event streams.

Sensor event streams. We first examine our forecasting results on real motion event streams. Figure 1, Figure 5 and Figure 6 show our real-time forecasting results for three motion capture steams: *"house-cleaning"*, *"exercise"* and *"chicken-dance"*, respectively. The datasets were obtained from the CMU motion capture database[6]. Each event stream consists of $d = 4$ dimensional vectors (left-right legs and arms), and it contains various motions (i.e., regimes), such as walking and dancing. One of the results has already been presented in section 1 (i.e., Figure 1). It automatically and effectively identifies multiple regime shift positions (such as from the wiping motion to the walking motion), and forecasts long-range future activities.

Figure 5 shows our real-time forecasting results and some snapshots obtained at several different time ticks. Specifically, Figure 5 (a) consists of several regimes, which correspond to some exercise activities. The bottom of Figure 5 (a) shows the output for (100:120)-steps-ahead real-time predictions, where we set the reporting window $l_p = 20$. More specifically, REGIMECAST generates (100:120)-steps-ahead future events for every $l_p = 20$ time tick, (e.g., at time tick $t_m = 100$, it reports $X[200 : 220]$, at time tick $t_m = 120$, it reports $X[220 : 240]$, and so on). Figure 5 (b) shows four examples of REGIMESNAP at different time points, each of which belongs to one of four different regimes. As shown in the figures, our modeling approach successfully captures dynamical regime shift patterns, as well as the non-linear dynamical evolving activities of four different regimes.

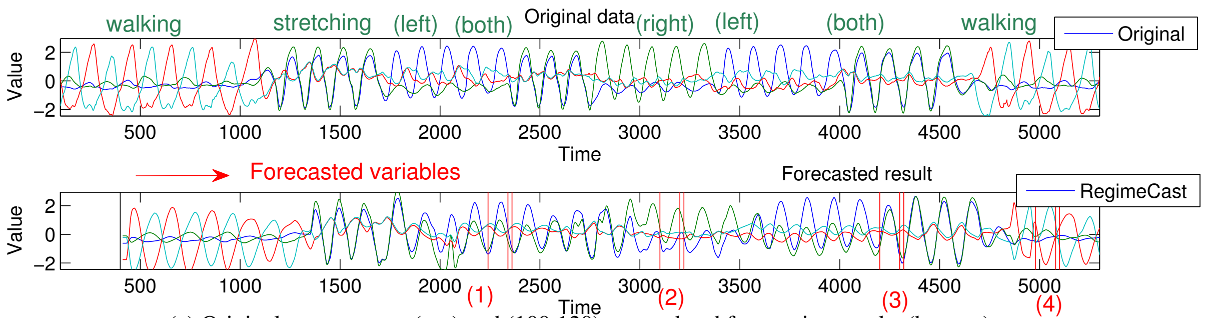
Figure 6 shows REGIMECAST for the chicken-dance stream. Specifically, the top of Figure 6 (a) shows the original stream, which is composed of four steps in the following order: "beaks", "wings", "tail feathers" and "claps", and these steps are repeated again. It is a much more challenging task to forecast upcoming dance steps because the event stream contains several nested multi-scale regimes. As shown at the top of Figure 6 (b), each step can be decomposed into several basic movements, where each movement has a different tempo/speed. For example, "tail feathers" consists of "moving arms, quickly" and "bending knees, once", The bottom of Figure 6 (a) and (c-1)-(c-3) show our entire output of (30:35)-steps-ahead forecasting results, and REGIMESNAP at several time ticks, respectively. As shown in the figures, our proposed method captures complicated dynamical activities, which consist of multiple hidden regimes, and successfully generates long-range upcoming motion patterns. We should also note that our algorithm does not use any prior training or hint regarding the steps. It incrementally finds important dynamical patterns (i.e., regimes), estimates the parameters of the new regime and inserts it into the model database.

Online activity streams. Next, we demonstrate our forecasting power as regards Web-click activities. Figure 7 shows our results for *GoogleTrend* event streams, which consist of the search volumes for various queries (i.e., words) on Google[7]. Each query represents the search volumes related to keywords over time (over twelve years, on a weekly basis). We performed our real-time forecasts on four event streams, which were taken from the following domains: (a) *OnlineTV*, (b) *Beer*, (c) *Social media* and (d) *Software*. For each event stream, it continuously generates three-months-ahead predictions, every time tick.

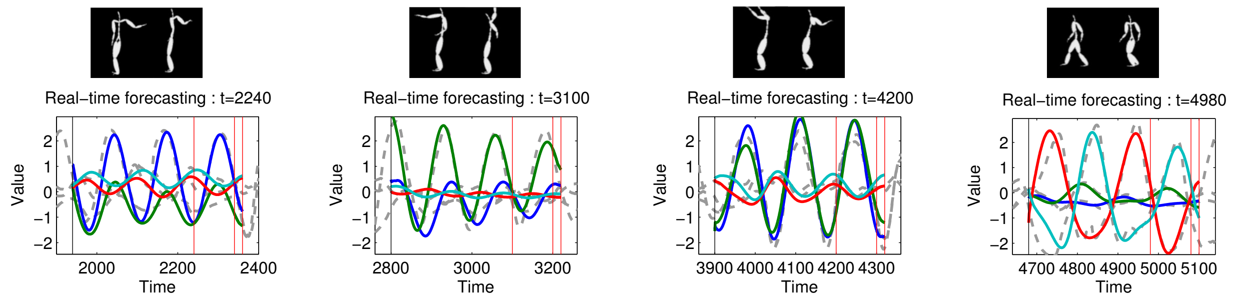
Figure 7 (a) shows our forecasting result for *OnlineTV*, which contains $d = 4$ dimensional events: Netflix ($\boldsymbol{x}_1$), Hulu ($\boldsymbol{x}_2$), YouTube ($\boldsymbol{x}_3$) and Amazon Prime ($\boldsymbol{x}_4$). Recently, there has been a rapid increase in new video streaming services, and REGIMECAST successfully captures the long-range evolution and exponential rising patterns in all co-evolving keywords. Note that there was a regime shift point from 2011 to 2012, i.e., Hulu ($\boldsymbol{x}_2$, shown as the green line) has had a declining pattern since 2011, which coincided with the ascent of Netflix ($\boldsymbol{x}_1$, shown as the blue line), possibly indicating that there was competition/interaction between the two services, and Netflix has been drawing customers' attention away from Hulu. Most importantly, our algorithm, REGIMECAST, can auto-

[6]*MoCap*: http://mocap.cs.cmu.edu/

[7]*GoogleTrend*: http://www.google.com/trends/

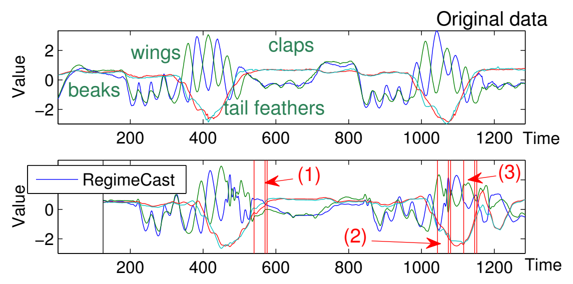


(a) Original event stream (top) and (100:120)-steps-ahead forecasting results (bottom)

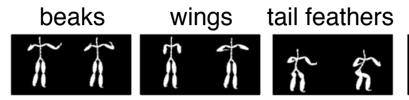


(b-1) Stretch/left ($t_c = 2240$) (b-2) Stretch/right ($t_c = 3100$) (b-3) Stretch/both ($t_c = 4200$) (b-4) Walking ($t_c = 4980$)

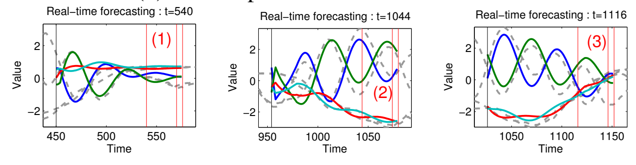
Figure 5: REGIMECAST is effective: Forecasting power of REGIMECAST for the motion event stream (*"exercise"*). (a) The original data (top), our (100:120)-steps-ahead predictions (bottom), and (b) snapshots of video clips (top) and REGIMESNAP (bottom) at four different time ticks. REGIMECAST automatically identifies important regime patterns (e.g., "stretching" and "walking"), and also forecasts future events, incrementally, and effectively.



(a) Original stream (top) and our forecasted events (bottom)

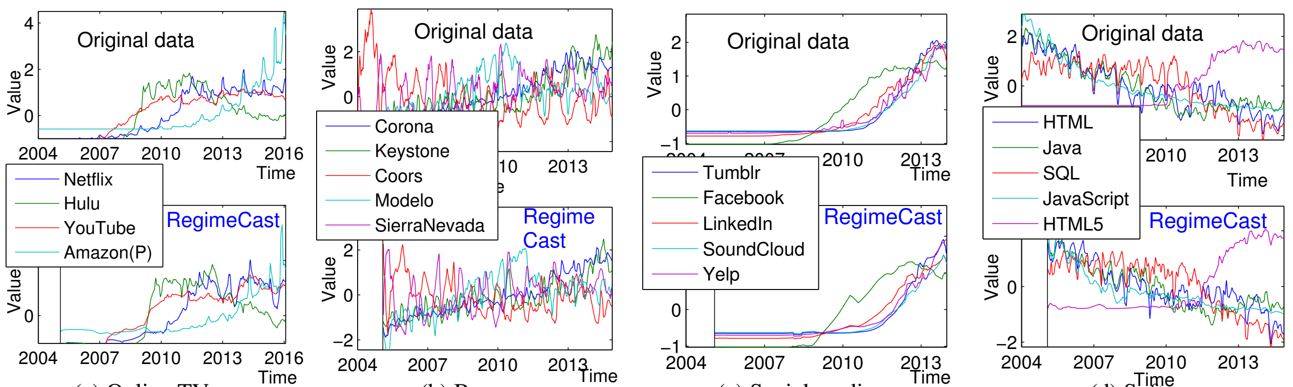


(b) Four steps of "chicken dance"



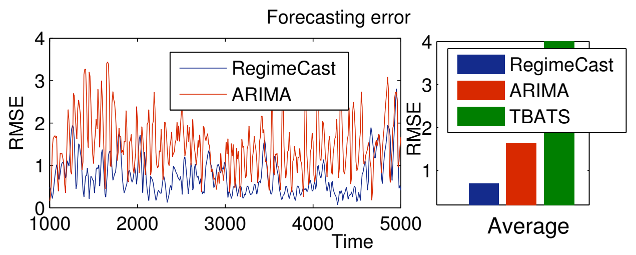
(c-1) $t_c = 540$ (c-2) $t_c = 1044$ (c-3) $t_c = 1116$

Figure 6: Real-time forecasting of REGIMECAST for "chicken dance": (a) Given the dance motion stream (bottom), our proposed method incrementally identifies four basic steps, i.e., (b) "beaks", "wings", "tail feathers" and "claps" in the sequence, and adaptively forecasts long-range future patterns. (c) REGIMESNAP for four different time points. Here, it generates (30:35)-steps-ahead future events. We emphasize that our algorithm does not need any prior training or knowledge regarding the motions (i.e., regimes).

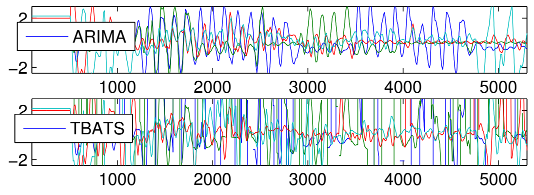


(a) Online TV (b) Beers (c) Social media (d) Software

Figure 7: REGIMECAST successfully forecasts 3-months-ahead future events of online user activities: Each event sequence consists of the Google search volume for several keywords (from 2004 to the present), i.e., (a) *OnlineTV* (1:Netflix, 2:Hulu, 3:YouTube, 4:Amazon Prime), (b) *Beer* (1:Corona, 2:Keystone, 3:Coors, 4:Modelo, 5:Sierra Nevada), (c) *Social media* (1:Tumblr, 2:Facebook, 3:LinkedIn, 4:SoundCloud, 5:Yelp), (d) *Software* (1:HTML, 2:Java, 3:SQL, 4:JavaScript, 5:HTML5). Given the online user activities (top), it incrementally forecasts events three months (i.e., 13 weeks) ahead, at every reporting window $l_p = 2$ weeks (bottom).

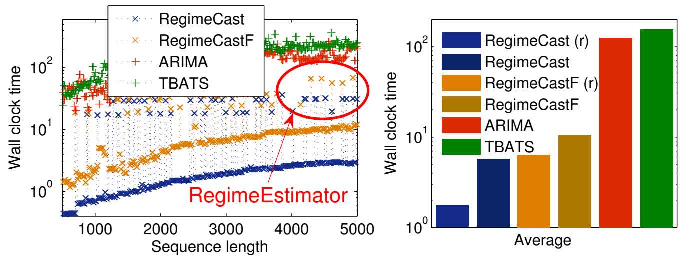


(a) Forecasting error for each time tick (left) and average (right)

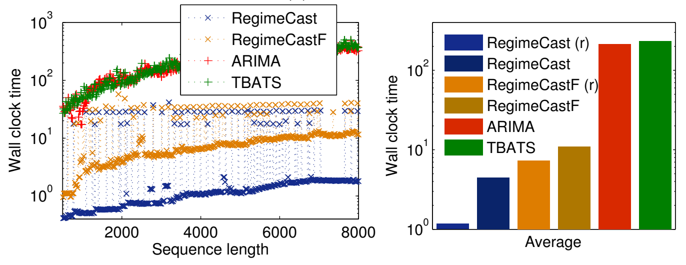


(b) Forecasting results of ARIMA (top) and TBATS (bottom)

Figure 8: Forecasting error (RMSE) for the motion event stream: REGIMECAST consistently outperforms the state-of-the-art methods with respect to accuracy between real values and the (100:120)-steps-ahead forecasted results. (a) Forecasting error for each time tick (left) and the average (right). Lower is better. (b) Forecasting results of our competitors. Please also see the original stream and our result shown in Figure 5 (a).

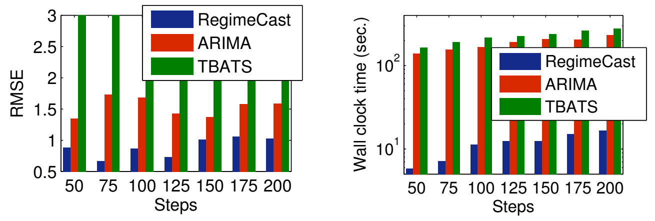


(a) "Exercise"



(b) "House-cleaning"

Figure 9: Wall clock time vs. sequence length t_c (left) and average (right): REGIMECAST consistently wins. It is up to 270 times faster than TBATS.



(a) forecasting error (b) computation time

Figure 10: l_s-steps-ahead forecasting over the motion event stream (*"exercise"*): REGIMECAST consistently wins. (a) Forecasting error (RMSE) vs. prediction length l_s; (b) Wall clock time vs. prediction length l_s. Lower is better.

matically identify sudden changes of regimes, and forecast upcoming events, adaptively and immediately.

Figure 7 (b) shows our result for *Beer*, which successfully captures the long-term non-linear dynamical evolution of the beer industry. There is significant growth of all the keywords with one exception, Coors (shown as the red line), which is a popular beer, brewed in Colorado, US. Similarly, as shown in Figure 7 (c) and (d), our proposed method also successfully identifies non-linear dynamic patterns of *Social media* and *Software*, including growing and competing activities, and estimates optimal regimes. Our forecasted results are very close to the real event streams.

5.2 Q2. Accuracy

Next, we discuss the quality of REGIMECAST in terms of forecasting accuracy. We compared our method with the following methods: (a) ARIMA, where we determined the optimal parameter set using AIC, and (b) TBATS [12], which is a state-of-the-art forecasting algorithm for complex seasonal time series.

Figure 8 shows the forecasting power of REGIMECAST for the motion event stream: "exercise", which is shown in Figure 5 (a). Specifically, Figure 8 (a) shows the root mean square error (RMSE) between the original and the (100:120)-steps-ahead forecasted events, where the left figure shows the RMSE for each time tick, and the right figure shows the average errors. A lower value indicates a better forecasting accuracy. Note that we omit the TBATS result for the left figure, due to high error values.

Figure 8 (b) shows the actual forecasting results of ARIMA and TBATS. Compared to our forecasted result, shown in Figure 5 (a), ARIMA and TBATS are *unsuitable* for capturing complex, non-linear dynamics and regime shifts; they are *linear* models, and failed to forecast abrupt changes of regimes.

5.3 Q3. Scalability

We also evaluate the efficiency of our forecasting algorithm. Figure 9 compares REGIMECAST with ARIMA and TBATS in terms of computation time for varying sequence lengths t_c. Note that the figures are shown in linear-log scales. To evaluate the efficiency of the dynamic point set (DPS), which is described in section 4, we also compared them with special versions of our method, REGIMECAST-F, which uses full point set (i.e., it sets the time interval $\delta = 1$). As we expected, REGIMECAST generates long-range future events, significantly faster than the competitors for the large streams (i.e., up to two orders of magnitude). In the left column of Figure 9, each spike corresponds to REGIMEESTIMATOR process, which creates a new regime (please see the red circle). The right column of Figure 9 shows the average computation time of entire event streams. Here, REGIMECAST/REGIMECAST-F (r) shows the average computation time of REGIMEREADER, and REGIMECAST/REGIMECAST-F is the computation time of REGIMECAST.

Discussion: l_s-steps-ahead prediction. As we mentioned in the introduction section, one of our motivations is the long-range forecasting over data streams. So, how long ahead can our method forecast future events? Is there any difference between, say, 50-steps-ahead and 200-steps-ahead forecasting results? We thus examine the forecasting power of REGIMECAST in terms of the future event length l_s. Figure 10 shows the forecasting errors and speeds of REGIMECAST for varying the steps: $l_s = 50, 75, \ldots, 200$. Specifically, Figure 10 compares our method with other methods in terms of (a) forecasting accuracy and (b) computation time. Thanks to our carefully designed models and algorithms, our method achieved a large reduction in both computation time and forecasting error for every step l_s.

6. CONCLUSIONS

In this paper, we focused on the problem of real-time forecasting over co-evolving event streams. Our proposed method, REGIMECAST exhibits all the desirable properties:

1. It is **Effective**: It captures complex non-linear dynamic patterns in event streams and forecasts long-term future events.
2. It is **Adaptive**: It incrementally and automatically recognizes the current regimes and finds regime shift points immediately, while it requires no prior training.
3. It is **Scalable**: Thanks to our efficient streaming algorithms, the computation time does not depend on data stream length.
4. It is **Any-time**: It provides a response at any time and generates long-range future events, immediately.

Acknowledgement. This work was supported by JSPS KAKENHI Grant-in-Aid for Scientific Research Number JP15H02705, JP16K12430, JP26730060, JP26280112, and the MIC/SCOPE #162110003.

7. REFERENCES

[1] D. Chakrabarti and C. Faloutsos. F4: Large-scale automated forecasting using fractals. *CIKM*, 2002.

[2] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.

[3] C. Folke, S. Carpenter, B. Walker, M. Scheffer, T. Elmqvist, L. Gunderson, and C. S. Holling. Regime shifts, resilience and biodiversity in ecosystem management. *Annual Review of Ecology, Evolution, and Systematics*, 35:557–581, 2004.

[4] J. Ginsberg, M. Mohebbi, R. Patel, L. Brammer, M. Smolinski, and L. Brilliant. Detecting influenza epidemics using search engine query data. *Nature*, 457:1012–1014, 2009.

[5] S. Hare and N. Mantua. Empirical evidence for North Pacific regime shifts in 1977 and 1989. *Progress in oceanography*, 47(2000):103–145, 2000.

[6] M. D. Hoffman, D. M. Blei, and F. R. Bach. Online learning for latent dirichlet allocation. In *NIPS*, pages 856–864, 2010.

[7] T. Iwata, T. Yamada, Y. Sakurai, and N. Ueda. Online multiscale dynamic topic models. In *KDD*, pages 663–672, 2010.

[8] E. Jackson. *Perspectives of Nonlinear Dynamics:*. Cambridge University Press, 1992.

[9] E. J. Keogh, S. Chu, D. Hart, and M. J. Pazzani. An online algorithm for segmenting time series. In *ICDM*, pages 289–296, 2001.

[10] J. Letchner, C. Ré, M. Balazinska, and M. Philipose. Access methods for markovian streams. In *ICDE*, pages 246–257, 2009.

[11] L. Li, B. A. Prakash, and C. Faloutsos. Parsimonious linear fingerprinting for time series. *PVLDB*, 3(1):385–396, 2010.

[12] A. M. D. Livera, R. J. Hyndman, and R. D. Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*, 106(496):1513–1527, 2011.

[13] M. Mathioudakis, N. Koudas, and P. Marbach. Early online identification of attention gathering items in social media. In *WSDM*, pages 301–310, 2010.

[14] Y. Matsubara, Y. Sakurai, and C. Faloutsos. Autoplait: Automatic mining of co-evolving time sequences. In *SIGMOD*, pages 193–204, 2014.

[15] Y. Matsubara, Y. Sakurai, and C. Faloutsos. The web as a jungle: Non-linear dynamical systems for co-evolving online activities. In *WWW*, pages 721–731, 2015.

[16] Y. Matsubara, Y. Sakurai, and C. Faloutsos. Non-linear mining of competing local activities. In *WWW*, 2016.

[17] Y. Matsubara, Y. Sakurai, C. Faloutsos, T. Iwata, and M. Yoshikawa. Fast mining and forecasting of complex time-stamped events. In *KDD*, pages 271–279, 2012.

[18] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: model and implications. In *KDD*, pages 6–14, 2012.

[19] Y. Matsubara, Y. Sakurai, W. G. van Panhuis, and C. Faloutsos. FUNNEL: automatic mining of spatially coevolving epidemics. In *KDD*, pages 105–114, 2014.

[20] A. Mueen and E. J. Keogh. Online discovery and maintenance of time series motifs. In *KDD*, pages 1089–1098, 2010.

[21] T. Palpanas, M. Vlachos, E. Keogh, and D. Gunopulos. Streaming time series summarization using user-defined amnesic functions. *IEEE Transactions on Knowledge and Data Engineering*, 20(7):992–1006, 2008.

[22] S. Papadimitriou, A. Brockwell, and C. Faloutsos. Adaptive, hands-off stream mining. In *VLDB*, pages 560–571, 2003.

[23] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *VLDB*, pages 697–708, 2005.

[24] S. Papadimitriou and P. S. Yu. Optimal multi-scale patterns in time series streams. In *SIGMOD*, pages 647–658, 2006.

[25] P. Patel, E. J. Keogh, J. Lin, and S. Lonardi. Mining motifs in massive time series databases. In *Proceedings of ICDM*, pages 370–377, 2002.

[26] B. A. Prakash, A. Beutel, R. Rosenfeld, and C. Faloutsos. Winner takes all: competing viruses or ideas on fair-play networks. In *WWW*, pages 1037–1046, 2012.

[27] T. Rakthanmanon, B. J. L. Campana, A. Mueen, G. E. A. P. A. Batista, M. B. Westover, Q. Zhu, J. Zakaria, and E. J. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *KDD*, pages 262–270, 2012.

[28] Y. Sakurai, C. Faloutsos, and M. Yamamuro. Stream monitoring under the time warping distance. In *ICDE*, pages 1046–1055, Istanbul, Turkey, April 2007.

[29] Y. Sakurai, Y. Matsubara, and C. Faloutsos. Mining and forecasting of big time-series data. In *SIGMOD, Tutorial*, pages 919–922, 2015.

[30] Y. Sakurai, S. Papadimitriou, and C. Faloutsos. Braid: Stream mining through group lag correlations. In *SIGMOD*, pages 599–610, 2005.

[31] M. Scheffer, J. A. Foley, S. R. Carpenter, C. Folke, and B. H. Walker. Catastrophic shifts in ecosystems. *Nature*, 413(6856):591–6, 2001.

[32] M. Vlachos, D. Gunopulos, and G. Kollios. Discovering similar multidimensional trajectories. In *ICDE*, pages 673–684, 2002.

[33] M. Vlachos, G. Kollios, and D. Gunopulos. Elastic translation invariant matching of trajectories. *Mach. Learn.*, 58(2-3):301–334, Feb. 2005.

[34] Y. R. Zelnika, E. Meron, and G. Bel. Gradual regime shifts in fairy circles. *PNAS*, 2015.

[35] Y. Zhao, N. Sundaresan, Z. Shen, and P. S. Yu. Anatomy of a web-scale resale market: a data mining approach. In *WWW*, pages 1533–1544, 2013.

[36] J. Zhou and A. K. H. Tung. Smiler: A semi-lazy time series prediction system for sensors. In *SIGMOD*, pages 1871–1886, 2015.

[37] Y. Zhu and D. Shasha. Statstream: Statistical monitoring of thousands of data streams in real time. In *VLDB*, pages 358–369, 2002.