Electronic Preprint for Journal of Information Processing Vol.28

Regular Paper

Real-time Forecasting of Non-linear Competing Online Activities

Thinh Minh Do[1,a)] Yasuko Matsubara[2,b)] Yasushi Sakurai[2,c)]

Received: September 9, 2019, Accepted: January 3, 2020

Abstract: Given a large, online stream of multiple co-evolving online activities, such as Google search queries, which consist of d keywords/activities for l locations of duration n, how can we analyze temporal patterns and relationships among all these activities? How do we go about capturing non-linear evolutions and forecasting long-term future patterns? For example, assume that we have the online search volume for multiple keywords, e.g., "HTML/Java/SQL/HTML5" or "Iphone/Samsung Galaxy/Nexus/HTC" for 236 countries/territories, from 2004 to 2015. Our goal is to capture important patterns and rules, to find the answer for the following issues: (a) Are there any periodical/seasonal activities? (b) How can we automatically and incrementally detect the sign of competition between two different keywords from the data streams? (c) Can we achieve a real-time snapshot of the stream and forecast long-range future dynamics in both global and local level? In this paper, we present RFCast, a unifying adaptive non-linear method for forecasting future patterns of co-evolving data streams. Extensive experiments on real datasets show that RFCast does indeed perform long-range forecasts and it surpasses other state-of-the-art forecasting tools in terms of accuracy and execution speed.

Keywords: time-series, non-linear, real-time forecasting

1. Introduction

The development of new technologies and online marketing systems has significantly led to stiff competition between many companies and brands. In terms of data mining, a massive volume and variety of time-stamped data of online user activity is generated and collected at a very high logging rate. In the real applications, there is a vital new opportunity for data scientists and analysts to measure the collective behavior of online users, in order to provide solutions to social, economic, and other important problems.

Our goal is to find patterns, relationships and deltas in a large collection of co-evolving online activities, consisting of tuples of the form: (activity, location, time). In other words, assume that we have a data stream $\mathbf{X}$ of d-dimensional data, e.g., "HTML/Java/SQL/HTML5", for 236 countries/territories, how can we find meaningful patterns and seasonal/annual activities? Specifically, we would like to answer the following questions: Is there any sign of competition, e.g., between "HTML" and "HTML5" during the stream monitoring? Can we execute the real-time forecasting of the future dynamics of online user activities in both global and local level?

In this paper, we present RFCast, a unifying adaptive non-linear method for forecasting future patterns of co-evolving data streams. Intuitively, we wish to solve the following problem:

Informal problem. Given data stream $\mathbf{X}$, which consists of multi-dimensional online activities of d keywords in l locations of duration n, i.e., $\mathbf{X} = \{\mathbf{x}(1), \ldots, \mathbf{x}(t_c)\}$, where t_c is the current time tick, we want to

- find global and local-level interaction and competition (e.g., HTML vs. HTML5),
- find seasonal/annual patterns, and
- spot external events (deltas).

Most importantly, we want forecast l_s-steps-ahead future event $\mathbf{x}(t_c + l_s)$, instantly, at any point in time.

Contributions. In this paper, we present RFCast, an efficient and effective non-linear method for forecasting future dynamics of co-evolving data streams. Our method has the following desirable properties:

(1) **Effective**: Our method can capture the meaningful properties in the co-evolving streams, such as the complex non-linear patterns and the sign of competition between activities, and it can also forecast long-term future dynamics.
(2) **Adaptive**: Thanks to our modeling framework, our method is fully automatic, requiring no manual tuning.
(3) **Scalable**: The computation cost of RFCast does not depend on data stream length.
(4) **Practical**: RFCast provides a response instantly and generates long-range future events.

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, and conclusions.

[1] Kumamoto University, Graduate School of Science and Technology, Kumamoto 860–8555, Japan
[2] Osaka University, The Institute of Scientific and Industrial Research, Ibaraki, Osaka 567–0047, Japan
[a)] madsheep2410@gmail.com
[b)] yasuko@sanken.osaka-u.ac.jp
[c)] yasushi@sanken.osaka-u.ac.jp

2. Related Work

Pattern Discovery and Forecasting in Time Series. In recent years, there has been an explosion of interest in mining time-stamped data, including pattern discovery and forecasting [1], [2], [19], [21], [22], [25], [27], [28]. Several traditional modeling and forecasting approaches typically use linear methods, such as autoregressive integrated moving average (ARIMA), linear dynamical systems (LDS), TBATS [11], Kalman filters (KF) and their variants [8], [9], [10], [26]. 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 [3].

AutoPlait [13] is a fully-automatic mining algorithm for co-evolving sequences based on HMMs; however, it cannot capture long-range non-linear dynamics of co-evolving data streams. Recently, [12] developed an adaptive non-linear dynamical system with the aim of understanding the dynamics of IoT data streams and social media and executing real-time forecasting.

Social Activity Analysis. Analysis of social media and online user behavior has attracted considerable interest. Through online social media, Matsubara et al. [17] explored the rise and fall patterns of data sequences that describes the information diffusion process. Prakash et al. [23] described the setting of two competing products/ideas spreading over a network, and provided a theoretical analysis of the propagation model for arbitrary graph topology. FUNNEL [18] is a novel non-linear method of mining spatially coevolving epidemic sequences using tensor analysis technique, while EcoWeb [14] has been one of the first methods that effectively discover patterns and do forecasting future dynamics of user online activities based on the idea of biological ecosystem. For online activity analysis, Gruhl et al. [7] provides a novel method of measuring the sales ranks on Amazon.com through online blogs. The work reported in Refs. [4], [6], [24] studied keyword volume, to predict consumer behavior. Besides, Ref. [15] also developed a compact and powerful representation of competition between time-evolving activities. Recently, Ref. [5] provided an automatic non-linear method to summarize and forecast co-evolving online activities.

Contrast to the Competitors. **Table 1** illustrates the relative advantages of our method. Only our RFCast matches all requirements, while

- The traditional AR, ARIMA and related forecasting methods including AWSOM [20], PLiF [10] and TriMine [16] are *fundamentally* unsuitable for our setting, because they are based on linear equations, while we employ *non*-linear equations. Moreover, most of them require parameter tuning.
- AutoPlait [13], EcoWeb [14] and CompCube [15] are comfortable non-linear models for time-series mining. However, they are not designed to capture long-range non-linear evolutions of co-evolving data streams. Especially, AutoPlait is incapable of forecasting.
- RegimeCast [12] provides a good summary of time-series evolution in streams, but cannot capture seasonal spikes, deltas and location specific activities.

Table 1 Capabilities of approaches. Our approach meets all specifications.

	ARIMA/++	FUNNEL	Δ-Spot	AutoPlait	EcoWeb	CompCube	RegimeCast	RFCast
Stream mining			√				√	√
Outliers			√	√	√	√		√
Competition					√	√		√
Seasonality	√	√		√	√	√		√
Local analysis		√	√			√		√
Parameter-free		√	√	√	√	√	√	√
Forecasting	√	√	√		√	√	√	√

3. Proposed Model

In this section, we describe the idea and structure of the proposed model.

Real-time Forecasting over Data Streams. Given a data stream $\mathbf{X}$, which consists of entries of d-dimensional data, i.e., $\mathbf{x}(1)$, $\mathbf{x}(2)$, ..., $\mathbf{x}(t_c)$, where $\mathbf{x}(t_c)$ is the most recent event, and t_c increases with every new time-tick. In order to execute real-time forecasting over the given data stream, it is necessary to build an algorithm that reports the upcoming future events, instantly, at any point in time, while ignoring the redundant information. Therefore, we came to a solution of l_s-steps-ahead forecasting.

l_s-steps-ahead Forecasting. Why should we employ "l_s-steps-ahead" for real-time forecasting? Given a large collection of event sequences, we want to instantly and efficiently forecast future events (e.g., to predict the search volume for a specified product in the next few weeks, or detect the sign of competition between two brands). Therefore, our forecasting algorithm should be able to generate months-ahead prediction, at every point in time. It should also estimate future events, smoothly and continuously, by monitoring current trends and dynamic evolving patterns. We understand that short-term prediction is meaningless while facing with real-time problems. By doing the l_s-steps-ahead forecasting, we can provide an effective long-term solution for various proposals (economics, marketing). Furthermore, continuous processing is a requirement for data stream monitoring. Since the traditional time-series forecasting approaches (e.g., ARIMA) are static and linear, we want to build an ideal method

Table 2 Symbols and definitions.

Symbol	Definition
d	Number of keywords/queries
l	Number of locations/countries
n	Duration of sequences
$\mathbf{X}$	d co-evolving event stream ($\mathbf{X} \in \mathbb{N}^{d \times l \times n}$)
$\mathbf{x}$(t)	d-dimensional event at time tick t, i.e., $\mathbf{x}(t) = \{\mathbf{x}_i(t)\}_{i=1}^{d}$
$\mathbf{B_G}$	Base global matrix ($d \times 3$)
$\mathbf{B_L}$	Base local matrix ($d \times l$)
$\mathbf{C_G}$	Competition global matrix ($d \times d$)
$\mathbf{C_L}$	Competition local matrix ($d \times l$)
$\mathbf{S_G}$	Seasonality global matrix ($d \times n_p$)
$\mathbf{S_L}$	Seasonality local matrix ($d \times l$)
$\mathbf{D}$	Delta tensor
$\mathbf{X}_C$	Current window, i.e., $\mathbf{X}_C = \mathbf{X}[t_m : t_c]$
$\mathbf{V}_E$	Estimated window, i.e., $\mathbf{V}_E = \mathbf{V}\,[t_m : t_e]$
$\mathbf{V}_F$	Forecast window, i.e., $\mathbf{V}_F = \mathbf{V}\,[t_s : t_e]$
$\mathbf{v}$(t)	d-dimensional estimated event at time tick t, i.e., $\mathbf{v}(t) = \{\mathbf{v}_i(t)\}_{i=1}^{d}$
$\mathbf{F}$	Complete parameter set of RFCast i.e., $\mathbf{F} = \{\mathbf{B_G}, \mathbf{B_L}, \mathbf{C_G}, \mathbf{C_L}, \mathbf{S_G}, \mathbf{S_L}, \mathbf{D}\}$

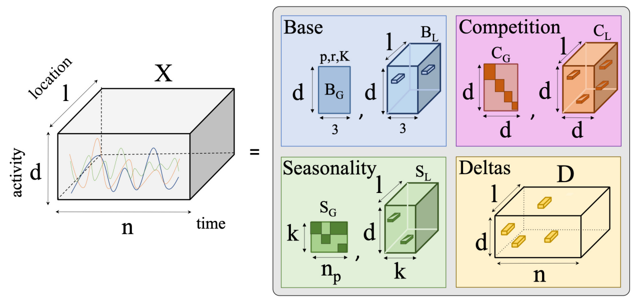


Fig. 1 Illustration of RFCast. Given a tensor $\mathbf{X} \in \mathbb{N}^{d \times l \times n}$, it extracts important patterns with respect to the following aspects: base properties of global and local trends ($\mathbf{B_G}$, $\mathbf{B_L}$), competition ($\mathbf{C_G}$, $\mathbf{C_L}$), seasonality ($\mathbf{S_G}$, $\mathbf{S_L}$), and deltas $\mathbf{D}$.

that instantly reports future events by continuously capturing and updating the current dynamic evolving patterns.

3.1 Intuition behind our Method

Assume that we receive a large event stream $\mathbf{X}$, which contains complex non-linear patterns, we then have a collection of sequences with d unique queries/keywords, l locations/countries with duration n. We can treat this set as a 3rd-order tensor, i.e., $\mathbf{X} \in \mathbb{N}^{d \times l \times n}$, where the element $\mathbf{x}_{im}(t)$ of $\mathbf{X}$ corresponds to the volume of the i-th keyword in the m-th country at time-tick t. **Figure 1** shows the illustration of our modeling framework. Given a tensor $\mathbf{X}$, it extracts important patterns of four aspects, base properties of global and local trends ($\mathbf{B_G}$, $\mathbf{B_L}$), competition ($\mathbf{C_G}$, $\mathbf{C_L}$), seasonality ($\mathbf{S_G}$, $\mathbf{S_L}$), and deltas $\mathbf{D}$.

- **Basic Trends:** The basic trends describe the non-linear evolution of individual activity, including potential popularity and growth rate. We assume that the popularity size of each keyword/activity dynamically evolves over time, and it corresponds to the aggregated volume of each user who pays attention and searches for the keyword in each country. For example, if a new technology (say, HTML5) is attractive, many users would spend more time studying it, or recommend it to their friends, and eventually this would lead to an exponential growth in popularity size.
- **Competition:** We assume that there is latent competition between different activities. For example, most users choose one of the products (smartphones), (e.g., iPhone, Samsung Galaxy, Blackberry, Nexus), based on the features and price of the products, or their own favors. Furthermore, it is highly important for us to capture location-specific trends and patterns. For example, each country has its own trends and user activities. These activities evolve naturally over time and depend on many factors including society, routine and economy.

Model 1 (RFCast-base) Let $P_{im}(t)$ be the potential popularity size of activity i in the m-th country at time tick t. Our base model is can be described as the following equations:

$$\boxed{\begin{gathered} P_{im}(t) = P_{im}(t-1)\left[1 + r_{im}\left(1 - \frac{\sum_{j=1}^{d} c_{ijm} \times P_{jm}(t-1)}{K_{im}}\right)\right], \\ (i = 1, \dots, d; m = 1, \dots, l; t = 1, \dots, n) \end{gathered}}$$

Model 1 consists of the following parameters:

- p_{im}: popularity size of activity i in m-th location at time tick $t = 0$ ($P_{im}(0) = p_{im}$).
- r_{im}: growth rate of activity i in m-th location ($r_{im} > 0$).
- K_{im}: available user resources of activity i in m-th location ($K_{im} > 0$).
- c_{ijm}: competition coefficient of j-th activity on i-th activity in m-th location ($c_{iim} = 1$, $c_{ijm} \geq 0$).

We assume that competing activities share some of the same finite user resources, and users cannot use their time/money for multiple purposes simultaneously. At time tick t, the percentage of users that might be interested in activity i in the m-th country can be described as $\left(1 - \frac{\sum_{j=1}^{d} c_{ijm} \times P_{jm}(t-1)}{K_{im}}\right)$, where c_{ijm} is the competition coefficient that describes the effect rate of activity j on activity i in the m-th country. If $c_{ijm} = 0$ ($i \neq j$), there is no interaction between activities i and j in the m-th country, (i.e., "neutralism"). In contrast, if $c_{ijm} = c_{jim} = 1$, these two activities compete with each other in the m-th location, by sharing exactly the same user resource.

- **Seasonality:** Seasonality describes yearly, cyclic, temporal user activities (e.g., Black Friday). We assume that each keyword (e.g., iPhone, Samsung Galaxy) always has a certain volume of popularity, however, user behavior changes dynamically according to the season, various annual events and customs (e.g., the number of sold smartphones significantly rises during Black Friday or sales seasons). We use the new parameter namely, $s_{im}(t)$, to describe the seasonal pattern.
- **Deltas:** We define deltas as extreme spikes, which represent major events, and are completely independent of long-range evolution and seasonality. They represent unrepeated, external events and anomalies, such as the official release of the first generation iPhone in 2007. To reflect these phenomena, we introduce an additional parameter, namely, $\delta_{im}(t)$: deltas.

Model 2 (RFCast-full) Let $\mathbf{v}_{im}(t)$ be the estimated volume of activity i in the m-th country at time tick t. Our full model is described as the following equations:

$$\boxed{\begin{gathered} \mathbf{v}_{im}(t) = P_{im}(t)\left[1 + s_{im}\left(t \mod n_p\right)\right], \\ (i = 1, \dots, d; m = 1, \dots, l; t = 1, \dots, n) \end{gathered}}$$

where, n_p stands for the period of the cycle (i.e., $n_p = 52$ weeks).

The estimated volume $\mathbf{v}_{im}(t)$ depends on the latent popularity size $P_{im}(t)$ and the seasonal/annual trends $s_{im}(t \bmod n_p)$, i.e., relative value of popularity size $P_{im}(t)$ vs. the actual volume $\mathbf{v}_{im}(t)$. Note that if there is no seasonality for activity i in the m-th country at time t, (i.e., $s_{im}(t \bmod n_p) = 0$), the estimated volume is equal to the popularity size (i.e., $\mathbf{v}_{im}(t) = P_{im}(t)$). Here, $\delta_{im}(t)$ describes non-cyclic, completely independent activity of the long-range evolution. We want to capture the deltas from the event stream $\mathbf{X}$ to ignore it from our forecasting result, since they do not link to the dynamical evolution of the data streams.

For the next step, we are also interested in capturing the above patterns from the following perspectives,

- **Global:** i.e., world-wide level: general trends and patterns.
- **Local:** i.e., country level: area-specific trends and local sensitivities.

Our complete model consists of the following:

Definition 1 (Complete set of RFCast) Let $\mathbf{F}$ be a complete parameter set ($\mathbf{F} = \{\mathbf{B_G}, \mathbf{B_L}, \mathbf{C_G}, \mathbf{C_L}, \mathbf{S_G}, \mathbf{S_L}, \mathbf{D}\}$) that describe the global/local patterns of the sequences in $\mathbf{X}$, i.e.,

- $\mathbf{B_G}$ ($d \times 3$): global-level basic patterns of each activity, including initial popularity size, growth rate, carrying capacity).
- $\mathbf{B_L}$ ($d \times 3 \times l$): local-level basic patterns for l countries.
- $\mathbf{C_G}$ ($d \times d$): global-level competition between d keywords.
- $\mathbf{C_L}$ ($d \times d \times l$): local-level competition for l countries.
- $\mathbf{S_G}$ ($k \times n_p$), $1 \leq k \leq d$: $\mathbf{S_G}$ consists of k components of period n_p, each component refers to an individual seasonal pattern, such as Black Friday or year-end vacation.
- $\mathbf{S_L}$ ($d \times k \times l$): local-level seasonal patterns for l countries.
- $\mathbf{D}$ ($d \times n \times l$): a tensor of deltas.

4. Algorithm

4.1 Overview

Firstly, we define several key concepts in this paper.

Definition 2 (Event stream X) Let $\mathbf{X}$ be a data stream that consists of event entries of d keywords, for l countries, i.e., $\mathbf{X} = \{\mathbf{x}(1), \ldots, \mathbf{x}(t_c)\}$, where t_c is the current time tick. We refer to $\mathbf{X}$ as an event stream.

Assume that we receive a new event $\mathbf{x}(t_c)$, 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 3 (Current window $\mathbf{X}_C$) Let $\mathbf{X}_C = \mathbf{X}[t_m : t_c]$ be the subsequence of length l_c, starting from time tick t_m and ending at t_c ($1 \leq t_m \leq t_c$).

Given the current window $\mathbf{X}_C$, our next step is to find the optimal parameter in $\mathbf{F}$, and predict l_s-steps-ahead future activities: $\mathbf{V}_F = \{\mathbf{v}(t_s), \ldots, \mathbf{v}(t_e)\}$ using Model 2

Definition 4 (l_s-steps-ahead foreacast window: $\mathbf{V}_F$) Let $\mathbf{V}_F = \mathbf{V}\ [t_s : t_e]$ 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 2 shows a snapshot of RFCast at the current time tick t_c. Here, the black dotted lines show the original event stream $\mathbf{X}$. The blue colored line show our estimated event $\mathbf{V}_E$ 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.

Consequently, given an event stream $\mathbf{X}$, our goal is to capture the current time-evolving patterns in $\mathbf{X}_C$, as a complex, non-linear dynamical system, and predict the l_s-steps-ahead forecast window $\mathbf{V}_F$, at any point in time.

Here, we introduce our forecasting algorithm, which consists of the following algorithms:

- RFCast-Reader: Generates the estimated events $\mathbf{V}_E = \mathbf{V}\ [t_m : t_e]$, when we are given the current window $\mathbf{X}_C$ and the model parameters $\mathbf{F}$.
- RFCast-Estimator: Complete the requirement of updating the parameter set $\mathbf{F}$ for $\mathbf{X}_C$, when the current time-tick t_c marks a new period (a new year).
- RFCast: Reports the optimal l_s-steps-ahead future events i.e., forecast window $\mathbf{V}_F$. It also maintains the full model parameter set $\mathbf{F}$.

Figure 3 illustrates how the our algorithm works. Given an event stream $\mathbf{X} = \{\mathbf{x}(1), \ldots, \mathbf{x}(t_c)\}$, where t_c is the current time tick, our algorithm incrementally extracts the current window $\mathbf{X}_C$, estimates the current dynamical pattern $\mathbf{V}_E$, and instantly reports l_s-steps-ahead future events $\mathbf{V}_F$. It also maintains the time-series model database, and updates model parameters in $\mathbf{F}$, if required. Next, we describe each of our algorithms in full details.

4.2 RFCast - Reader

Assume that we are given the current window $\mathbf{X}_C$ and the current parameter set $\mathbf{F} = \{\mathbf{B_G}, \mathbf{B_L}, \mathbf{C_G}, \mathbf{C_L}, \mathbf{S_L}, \mathbf{S_L}, \mathbf{D}\}$. We first estimate the dynamical event sequence $\mathbf{V}_E = \mathbf{V}[t_m : t_e]$, as shown in Fig. 3. Next, the question is how can we estimate events $\mathbf{V}_E$, given the current set $\mathbf{F}$? The most straightforward solution would be simply to use the fixed parameters in $\mathbf{F}$ and calculate $\mathbf{v}(t_m)$, $\mathbf{v}(t_m + 1)$, ..., using Model 2. We try to avoid that approach because the latent trends of the current window $\mathbf{X}_C$ in the event streams would dynamically and continuously evolve over time. Therefore, it is necessary for us to describe the characteristics of the current window $\mathbf{X}_C$ by identifing the optimized parameters in $\mathbf{F}$. More importantly, in order to describe the current activities in $\mathbf{X}_C$, the algorithm needs to adaptively update the parameters continuously. Algorithm 1 shows the overall process of detecting the complex non-linear patternin the current window $\mathbf{X}_C$, namely, RFCast-Reader. Firstly, for each i-th activity, the algo-

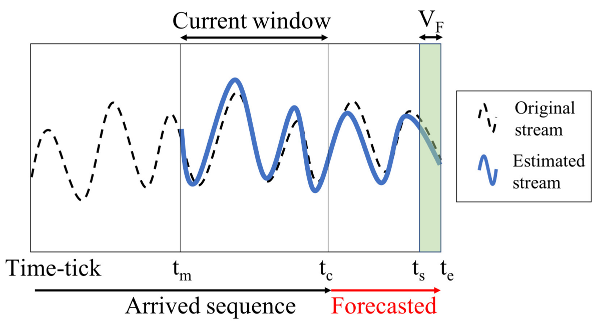


Fig. 2 Snapshot of our streaming algorithm: Given a data stream $\mathbf{X}$ (black dotted lines), our algorithm estimates the current time-series pattern $\mathbf{V}_E = \mathbf{V}\ [t_m : t_e]$ (blue bold lines), and reports l_s- steps-ahead future events $\mathbf{V}_F$ (in a red box),incrementally and continuously. Here, $\mathbf{X}_C = \mathbf{X}[t_m : t_c]$ is a current window (i.e., recently arrived events) and $\mathbf{V}_F = \mathbf{V}\ [t_s : t_e]$ is a l_s-steps-ahead future (i.e., unknown) event set.

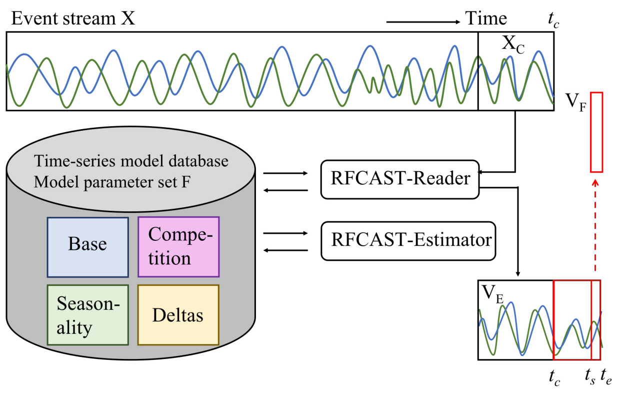


Fig. 3 Overview of our algorithm: Given an event stream $\mathbf{X}$, it extracts the current window $\mathbf{X}_C$, and then searches for the optimal parameters in the time-series model database, and generates the ls-steps-ahead events $\mathbf{V}_F$. If there is a new detected competition in $\mathbf{X}_C$, it also update the new parameter set, and inserts it into the database.

Algorithm 1 RFCast-Reader($\mathbf{X}_C, \mathbf{F}$)

```
1: Input: current window $\mathbf{X}_C$ ($d \times l \times l_c$) and current parameter set $\mathbf{F}$, i.e.,
   $\mathbf{F} = \{\mathbf{B_G}, \mathbf{B_L}, \mathbf{C_G}, \mathbf{C_L}, \mathbf{S_G}, \mathbf{S_L}, \mathbf{D}\}$
2: Output: Estimated event $\mathbf{V}_E = \mathbf{V}[t_m : t_e]$, update $\mathbf{F}$
3: for $i = 1 : d$ do
4:     $\mathbf{V}_{Ci} = \arg\min_{\mathbf{V}'_{Ci}} ||\mathbf{X}_{Ci} - \mathbf{V}'_{Ci}||$; // $\mathbf{V}'_{Ci} = f_C(\mathbf{F}_i)$;
5:     $\mathbf{F}_i = \arg\min_{\mathbf{F}'_i} ||\mathbf{X}_{Ci} - \mathbf{V}_{Ci}||$;
6: end for
7: $\mathbf{V}_E = f_E(\mathbf{F})$; // Calculate estimated event $\mathbf{V}_E$
8: return $\{\mathbf{V}_E, \mathbf{F}\}$;
```

rithm tries to optimize the parameters in $\mathbf{F}$, so that it minimizes the mean square errors between the original events and the estimated events, i.e., min $||\mathbf{X}_C - \mathbf{V}_C||$. Here, let $f_C(\mathbf{F})$ be a function that generates estimated events $\mathbf{V}_C = \{\mathbf{v}(t_m), \ldots, \mathbf{v}(t_c)\}$ in Model 2, given the parameter set $\mathbf{F}$. After being given a set of optimized event sequences $\mathbf{V}_C$, the algorithm computes the potential popularity size, and then the estimated volume of activities $\mathbf{v}_C$. Finally, it computes the estimated event $\mathbf{V}_E = f_E(\mathbf{F})$ as the optimal events for the current window $\mathbf{X}_C$.

4.3 RFCast - Estimator

Next, let us tackle the next question, namely, what if there is a change in dynamical pattern of the sequences in $\mathbf{X}_C$, for example, the sign of competition between activities, the change of seasonal pattern, the existence of deltas? In that case, we want to estimate the new parameter set $\mathbf{f}$ that describes the dynamical patterns in $\mathbf{X}_C$, and update it to the full parameter set $\mathbf{F}$ (see Fig. 3). Since $\mathbf{F}$ consists of a large number of parameters, it is extremely expensive to optimize all the parameters simultaneously, at every time-tick. We thus propose an efficient and effective algorithm, namely, RFCast-Estimator, which estimates for the optimal parameters in terms of both global and local levels, only if the current time-tick t_c marks a new period. In other words, we run RFCast-Estimator after every 52 time-ticks.

4.3.1 Model Description and Data Compression

Our goal is to efficiently and automatically estimate the full parameter set given $\mathbf{X}$. We introduce a new coding scheme, which is based on the Minimum Description Length (MDL) principle. Here, it follows the assumption that the more we can compress the data, the more we can detect its underlying patterns.

Model Description Cost. Firstly, we estimate the description complexity of a model parameter set, $Cost_M(\mathbf{F})$. It consists of the following terms:

- The number of activities d, countries l, and time-ticks n require $\log^*(d) + \log^*(l) + \log^*(n)$ bits [*1].
- Basic trends: $Cost_M(\mathbf{B_G}) = d \cdot c_F$, $Cost_M(\mathbf{B_L}) = |\mathbf{B_L}| \cdot (\log(d) + \log(3) + \log(m) + c_F) + \log^*(|\mathbf{B_L}|)$ [*2]
- Competition: $Cost_M(\mathbf{C_G}) = |\mathbf{C_G}| \cdot (\log(d) + \log(d) + c_F) + \log^*(|\mathbf{C_G}|)$, $Cost_M(\mathbf{C_L}) = |\mathbf{C_L}| \cdot (\log(d) + \log(d) + \log(l) + c_F) + \log^*(|\mathbf{C_L}|)$
- Seasonality: $Cost_M(\mathbf{S_G}) = |\mathbf{S_G}| \cdot (\log(k) + \log(n_p) + c_F) + \log^*(|\mathbf{S_G}| + \log^*(k))$, $Cost_M(\mathbf{S_L}) = |\mathbf{S_L}| \cdot (\log(d) + \log(k) + \log(l) + c_F) + \log^*(|\mathbf{S_L}|)$
- Deltas: $Cost_M(\mathbf{D}) = |\mathbf{D}| \cdot (\log(d) + \log(n) + \log(l) + c_F) + \log^*(|\mathbf{D}|)$

Data Coding Cost. The coding cost of $\mathbf{X}$ given the full parameter set $\mathbf{F}$ is computed by: $Cost_C(\mathbf{X}|\mathbf{F}) = \sum_{i,m,t=1}^{d,l,n} \log_2 p^{-1}_{Gauss(\mu,\sigma^2)}(\mathbf{x}_{im}(t) - \mathbf{v}_{im}(t))$, where $\mathbf{x}_{im}(t)$ and $\mathbf{v}_{im}(t)$ are the original and estimated volume of the i-th activity in the m-th country at time-tick t, μ and σ^2 are the mean and variance of the distance between the original and estimated values.

Total Code Length. The total code length for $\mathbf{X}$ can be described as: $Cost_T(\mathbf{X}; \mathbf{F}) = Cost_M(\mathbf{F}) + Cost_C(\mathbf{X}|\mathbf{F})$. Our goal here is to find an optimal parameter set $\mathbf{F}$ to minimize this function.

4.3.2 Global Parameters Estimation

Given a tensor $\mathbf{X}$, our sub-goal is to find the optimal global-level parameter set. Let $\mathbf{X}$ be the average volume of d activities for all l locations of length n, that is, $\mathbf{X} = \{\bar{\mathbf{x}}_i\}_{i=1}^d$, where $\bar{\mathbf{x}}_i = \{\frac{1}{m}\sum_{l=1}^m \mathbf{x}_{il}(t)\}_{t=1}^n$. For the next step, we iteratively optimize each parameter set for each i-th activity.

Specifically, let $\mathbf{F}_i$ be a set of global parameters for activity i, (i.e., $\mathbf{F}_i = \{\mathbf{B_G}_i, \mathbf{C_G}_i, \mathbf{S_G}_i, \mathbf{D}_i\}$) [*3]. Here, we initially set $c_{ij} = 0$ ($i \neq j$), which means there are no competitions between the d activities). The algorithm then separately and independently estimate parameters $\mathbf{F}_i$ for each individual sequence $\bar{\mathbf{x}}_i$ ($i = 1, \ldots, d$). In the case that competition between activities exists; for example, there is competition between two activities: $\bar{\mathbf{x}}_i$ and $\bar{\mathbf{x}}_j$. For each iteration, the algorithm searches for the best competitor $\bar{\mathbf{x}}_j$ of $\bar{\mathbf{x}}_i$, while minimizing the total cost function. Note that, this procedure continues until convergence.

One of the most important points here is that, we discover the deltas in $\mathbf{D}$ not for fitting the estimated events, but to distinguish it from the periodical spikes detected as seasonal patterns. However, during the stream monitoring, if the new detected deltas forms a seasonal pattern with the old ones, they will be considered the seasonal pattern and fit to the estimated result.

There are two main ideas behind the global parameter estimation algorithm: (1) TetraFit and (2) SubsetCollection.

TetraFit. The objective here is to purify parameters for $\mathbf{B_G}, \mathbf{C_G}$, as well as filter out seasonality $\mathbf{S_G}$ and deltas $\mathbf{D}$. However, we cannot optimize all the parameters simultaneously, as each component consists of several parameters, which makes the optimization too expensive. To deal with this problem, we propose an alternating method to optimize each parameter set of each i-th activity (i.e., $\mathbf{B_G}_i, \mathbf{C_G}_i, \mathbf{S_G}_i, \mathbf{D}_i$). Algorithm 3 shows the steps performed by TetraFit in detail. TetraFit iteratively estimates each parameter set that corresponds to the i-th activity, given a set of sequences $\mathbf{X}$, a current parameter set $\mathbf{F}$, and, index i. We also use the Levenberg-Marquardt (LM) algorithm and minimize the cost function. Note that, we ignore unrelated combinations/pairs (i, j) to reduce the computation cost.

SubsetCollection. We use a subset $\mathbf{X}_{[i]} \subset \mathbf{X}$ that competes with activity i to estimate the model parameters with respect to activity i for each iteration of TetraFit. It consists of sequences that compete directly (or indirectly) with the i-th sequence $\bar{\mathbf{x}}_i$, that is,

[*1] Here, $\log^*$ is the universal code length for integers.

[*2] $|\cdot|$ describes the number of non-zero elements, c_F is the floating point cost which is up to 4×8 bits.

[*3] $\mathbf{B_G}_i = \{p_i, r_i, K_i\}, \mathbf{C_G}_i = \{c_{ij}\}_{j=1}^d, \mathbf{S_G}_i = \{s_i(t)\}_{t=1}^{n_p}, \mathbf{D}_i = \{\delta_i(t)\}_{t=1}^n$. Initially, $p_i = r_i = K_i = 1, k = 0$.

Algorithm 2 RFCast-Estimator($\mathbf{X}_C$)

```
Input: current window $\mathbf{X}_C$ $(d \times l \times l_c)$
Output: full parameters, i.e., $\mathbf{F} = \{\mathbf{B_G}, \mathbf{B_L}, \mathbf{C_G}, \mathbf{C_L}, \mathbf{S_G}, \mathbf{S_L}, \mathbf{D}\}$;
/* (I) Global estimation */
Compute average volumes $\mathbf{X}_C$ $(d \times l_c)$, i.e., $\mathbf{X}_C = \{\bar{\mathbf{x}}_{Ci}\}_{i=1}^{d}$
Initialize parameter set $\mathbf{F}$
for $i = 1 : d$ do
    $\mathbf{F}_i$ = TetraFit $(i, \bar{\mathbf{x}}_{Ci}, \mathbf{F})$
end for
/* Estimate competition among all $d$ activities */
while improving the parameters do
    /* Select the most unfitted sequence $\bar{\mathbf{x}}_{Ci}$ */
    $i = \arg\max_{1 \le i' \le d} Cost_T(\bar{\mathbf{x}}_{Ci'}; \mathbf{F})$;
    /* Estimate parameter set $\mathbf{F}'_{ij}$ for each activity $\mathbf{x}_{Cj}$ */
    for $j = 1 : d$ do
        /* Find subset of sequences that have competition with $i$, $j$*/
        $\mathbf{X}_{C[i,j]}$ = SubsetCollection($\{\bar{\mathbf{x}}_{Ci}, \bar{\mathbf{x}}_{Cj}\}, \mathbf{C_G}$);
        $\mathbf{F}'_{ij}$ =TetraFit $(i, \mathbf{X}_{[i,j]}, \mathbf{F})$;
    end for
    /* Find the best competitor $\mathbf{x}_{Cj}$ of $\mathbf{x}_{Ci}$, and update $\mathbf{F}_i$ */
    $j = \arg\min_{1 \le j' \le d} Cost_T(\mathbf{X}_C; \mathbf{F}'_{ij'})$;
    $\mathbf{F}_i = \mathbf{F}'_{ij}$
end while
/* (II) Local estimation */
/* For each $i$-th activity in $l$-th location, $\mathbf{x}_{Cil}$ */
for $l = 1 : m$ do
    for $i = 1 : d$ do
        $\mathbf{X}_{C[il]}$ = SubsetCollection($\mathbf{x}_{Cil}, \mathbf{F}$);
        $\mathbf{B_L}, \mathbf{C_L}, \mathbf{S_L}, \mathbf{D}_{il}$ =TetraFit $(i, \mathbf{X}_{C[il]}, \{\mathbf{B_G}, \mathbf{C_G}\})$;
    end for
end for
return $\mathbf{F} = \{\mathbf{B_G}, \mathbf{B_L}, \mathbf{C_G}, \mathbf{C_L}, \mathbf{S_G}, \mathbf{S_L}, \mathbf{D}\}$;
```

Algorithm 3 TetraFit($i, \mathbf{X}_C, \mathbf{F}$)

```
Input: index $i$, sequence $\mathbf{X}_C$, current parameter set $\mathbf{F}$
Output: optimal parameters for $i$, i.e., $\mathbf{F}_i = \{\mathbf{B_G}_i, \mathbf{C_G}_i, \mathbf{S_G}_i, \mathbf{D}_i\}$
while improving the parameters do
    /* (I) Base and competition parameter fitting, i.e., $\mathbf{B_G}, \mathbf{C_G}$ */
    $\{\mathbf{B_G}_i, \mathbf{C_G}_i\} = \arg\min_{\mathbf{B_G}'_i, \mathbf{C_G}'_i} Cost_T(\mathbf{X}_C; \mathbf{B_G}, \mathbf{C_G}, \mathbf{S_G}, \mathbf{D})$;
    /* (II) Seasonal parameter fitting, i.e., $\mathbf{S_G}$ */
    $\{\mathbf{S_G}_i\} = \arg\min_{\mathbf{S_G}'_i} Cost_T(\mathbf{X}_C; \mathbf{B_G}, \mathbf{C_G}, \mathbf{S_G}, \mathbf{D})$;
    /* (III) Find new deltas, i.e., $\mathbf{D}$ */
    $\{\mathbf{D}_i\} = \arg\min_{\mathbf{D}'_i} Cost_T(\mathbf{X}_C; \mathbf{B_G}, \mathbf{C_G}, \mathbf{S_G}, \mathbf{D})$;
end while
return $\mathbf{F}_i = \{\mathbf{B_G}_i, \mathbf{C_G}_i, \mathbf{S_G}_i, \mathbf{D}_i\}$ ;
```

$\mathbf{X} = f(\bar{\mathbf{x}}_i)$, where, $f(\bar{\mathbf{x}}_i) = \{\bar{\mathbf{x}}_i \cup \bar{\mathbf{x}}_j \cup f(\bar{\mathbf{x}}_j)|^{\forall j} c_{ij} > 0\}$. If there is no competition (i.e., $^{\forall j} c_{ij} = 0 (i \neq j)$), then, $\mathbf{X}_{[i]} = \bar{\mathbf{x}}_i$. With this approach, we can superbly reduce the computation time needed for each iteration in TetraFit, because the subset $\mathbf{X}_{[i]}$ consists of only a small number of sequences.

4.3.3 Local Parameters Estimation

In this research, we also consider the issue of execute real-time forecasting for local-level events. The problem here is that, some of the countries have very sparse sequences, or contain different dynamical evolutions from the global pattern. To deal with this issue, we share the global competition for all l countries. Our algorithm ignores unrelated pairs of activities, and updates the coefficients only if $c_{ij} > 0$. The reason is that, if there is no local competition between two activities i and j for all l countries, there is no global competition between them. For each activity i in each country m, it finds an optimal parameter set $\mathbf{F}_{im}$, using both use SubsetCollection and TetraFit. Note that it uses global parameters $\mathbf{B_G}$ and $\mathbf{C_G}$ as the initial parameter set of TetraFit (see Algorithm 2).

Algorithm 4 RFCast ($\mathbf{x}(t_c)$)

```
Input: a new event $\mathbf{x}(t_c)$ at time-tick $t_c$
Output: $l_s$-steps-ahead future events $\mathbf{V}_F$
/* (I) Parameter fitting for activities */
$\{\mathbf{V}_E, \mathbf{F}\}$ = RFCast-Reader($\mathbf{X}_C, \mathbf{F}$);
/* (II) Update parameter (if required) */
if $t_c \mod n_p = 0$ then
    $\mathbf{f}$ = RFCast-Estimator($\mathbf{X}_C$);
    $\mathbf{F} = \{\mathbf{F} \cup \mathbf{f}\}$;
end if
/* (III) $ls$-steps-ahead future event generation */
$\mathbf{V}_F = \mathbf{V}[t_s : t_e]$;
return $\mathbf{V}_F$
```

4.4 RFCast

Our final goal is to capture the complex dynamical patterns $\mathbf{F}$, and predict l_s-steps-ahead forecast window $\mathbf{V}_F$. The detailed RFCast algorithm is shown in Algorithm 4. Given a new event $\mathbf{x}(t_c)$, it extracts the current window $\mathbf{X}_C$, estimates event sequence $\mathbf{V}_E$. When the current time tick t_c marks a new period (every 52 time-ticks), it launches RFCast-Estimator to estimate the seasonal patterns, detects new deltas and update the parameter set $\mathbf{F}$. Finally, it reports l_s-steps-ahead forecast window $\mathbf{V}_F$.

5. Experiments

In this section, we demonstrate the effectiveness of RFCast with real datasets. The experiments were designed to answer the following questions:

Q1 *Effectiveness*: Can our method succeed in modeling and forecasting long-term dynamics in given input streams?

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

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

We conducted our experiments on an Intel Core i7-3770K 3.50 GHz with 32 GB of memory, running Linux.

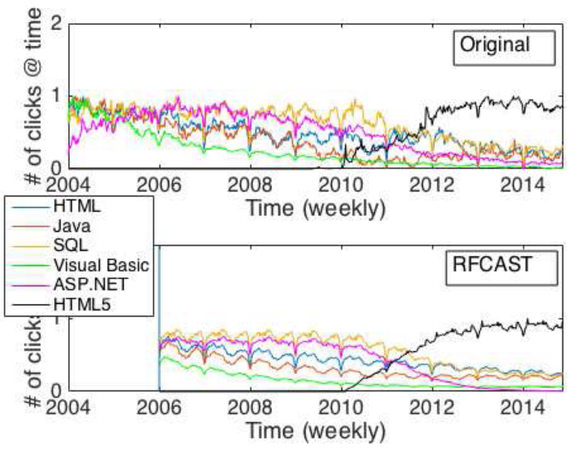
Dataset Description. We performed experiments on the *GoogleTrends* dataset. This dataset consists of the volume of searches for queries (i.e., keywords) in various topics (i.e., products, events, services, etc..) on Google [*4] from January 2004 to January 2015, collected in 236 countries. Each query represents the search volumes that are related to keywords over time (in weekly basis).

5.1 Effectiveness

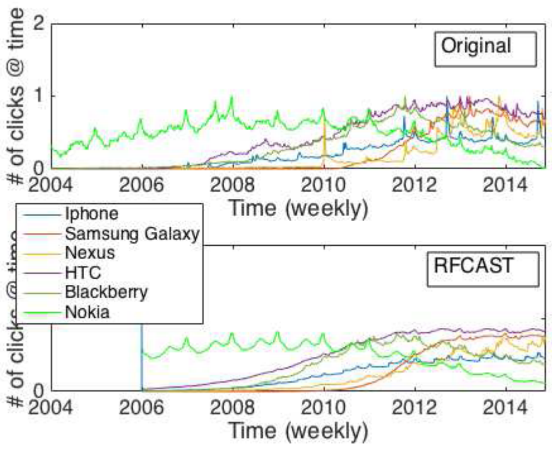
We demonstrate the forecasting power of RFCast in terms of capturing important patterns of event streams. We performed experiments on sequence sets of keywords/activities from four different domains on *GoogleTrends*. We picked up and performed our real-time forecasts on the top six major keywords for each domain, for $l = 236$ countries/territories, from January 2004 to January 2015. For each event stream, our proposed model continuously generates twelve-weeks-ahead (roughly three months) predictions, every time tick. Note that the dataset is scaled so that each sequence has a peak volume of 1.0.

#1. Development tools. Figure 4 (a) shows our discoveries on six keywords (i.e., 1: HTML, 2: Java, 3: SQL, 4: Visual Basic, 5: ASP.NET, 6: HTML5). HTML, Java, SQL, Visual Basic and ASP.NET has been popular tools for developers so far.

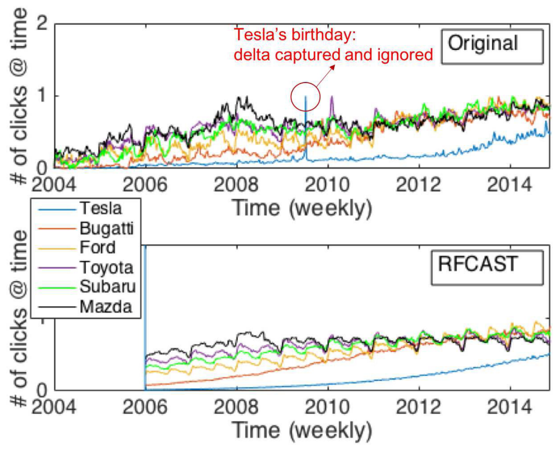
[*4] http://www.google.com/insights/search/



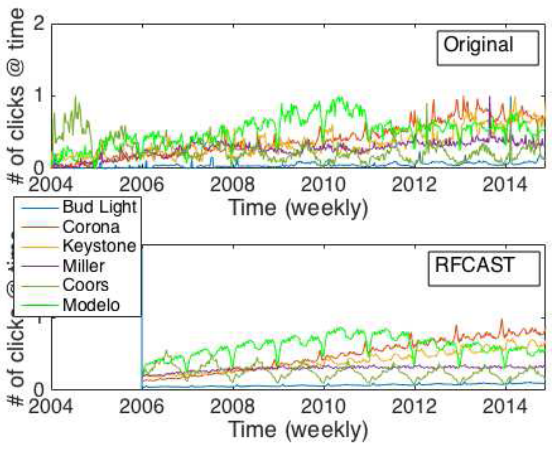
(a) Development tools



(b) Smartphones



(c) Car brands



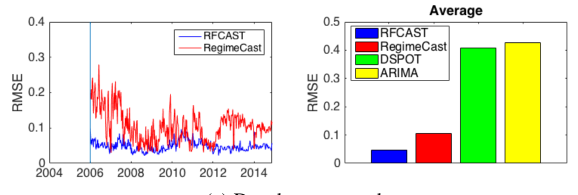
(d) Beer brands

Fig. 4 RFCast successfully forecasts three-months-ahead future events of online user activities: Each event sequence consists of the Google search volume for six keywords (from 2004 to 2015), i.e., (a) Development tools (1: HTML, 2: Java, 3: SQL, 4: Visual Basic, 5: ASP.NET, 6: HTML5), (b) Smartphones (1: Iphone, 2: Samsung Galaxy, 3: Nexus, 4: HTC, 5: Blackberry, 6: Nokia), (c) Car brands (1: Tesla, 2: Bugatti, 3: Ford, 4: Toyota, 5: Subaru, 6: Mazda), (d) Beer brands (1: Bud Light, 2: Corona, 3: Keystone, 4: Miller, 5: Coors, 6: Modelo). Given the online user activities (top), it incrementally forecasts events, at every reporting window = 1 week (bottom).

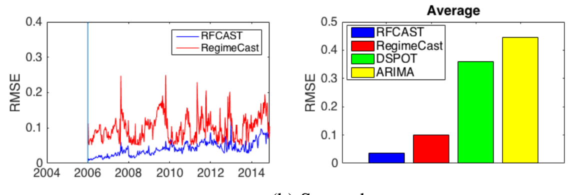
However, as soon as HTML5 (shown as the black line) became a topic of mainstream media attention around April 2010, the search volume for HTML5 has been rising rapidly, and RFCast successfully captures the long-range evolution and exponential rising patterns of this keyword. Furthermore, RFCast also captures the competitive evolution of all co-evolving keywords, i.e., the search volume of other five tools falls down because of the rising interest in the new comfortable tool, HTML5. We can conclude that HTML5 has been drawing developers' attention away from the other development tools. Most importantly, RFCast can automatically identify the sign of competition, as well as capture the non-linear dynamics of the sequences, also forecast the upcoming events, adaptively and instantly.

#2. Smartphones. Figure 4 (b) shows our forecasting result for smartphones, which contains six keywords (i.e., 1: Iphone, 2: Samsung Galaxy, 3: Nexus, 4: HTC, 5: Blackberry, 6: Nokia). Since the 90s, Nokia had become the best-selling mobile phone brand. However, with the existence of mobile phones with touch screen, the balance had changed. Many manufacturers started developing their smartphones with many features that dominated Nokia phones; especially, the release of iPhone in 2007 marked the rapid development of smartphones. Although Nokia also developed their own smartphones, their shares of the market started to drop dramatically. Our proposed method, RFCast successfully captures the long-range dynamics of the co-evolving keywords relating to the big smartphones brands. And it completes this task automatically and effectively.

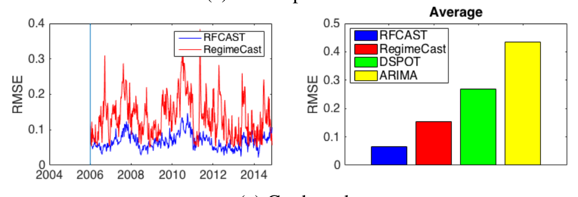
#3. Car brands. Figure 4 (c) shows our result for the group of car brands (i.e., 1: Tesla, 2: Bugatti, 3: Ford, 4: Toyota, 5: Subaru, 6: Mazda). Ford, Toyota, Subaru, Mazda has been four of the huge car brands in the world so far. Their products spread widely all over the world with various types of car. Recently, Bugatti and Tesla has caught the consumers' sight as the new luxury car brand. Especially, Tesla is one of the leading company in producing electric automobiles (AI driver). Since the car market has been speedily growing, the whole six brands' search volume keeps rising over years. RFCast, in this case, does not detect any sign of competition between the car brands, i.e., there



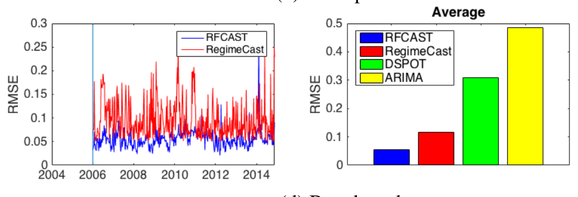
(a) Development tools



(b) Smartphones

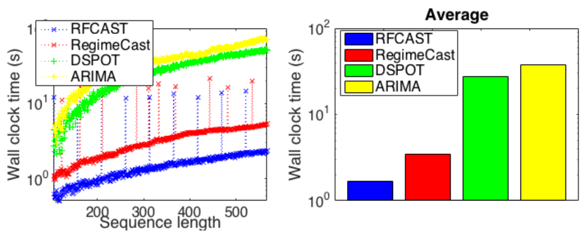


(c) Car brands

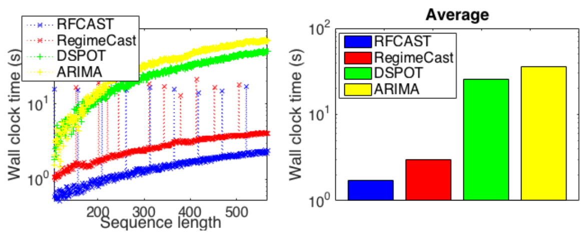


(d) Beer brands

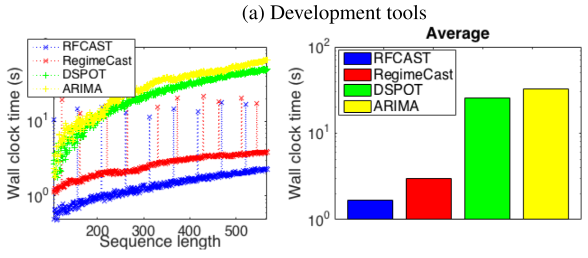
Fig. 5 Forecasting error (RMSE) for each time tick (left) and the average (right). A lower value indicates a better forecasting accuracy. RFCast consistently outperforms the state-of-the-art methods such as RegimeCast, Δ-Spot and ARIMA with respect to accuracy between real values and the three-months-ahead forecasted results.



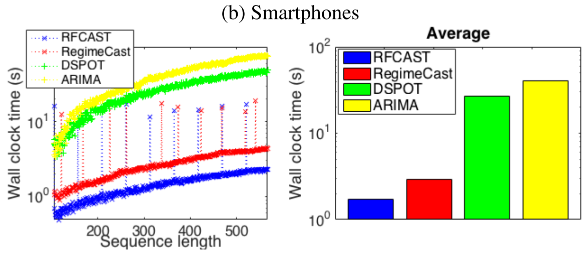
(a) Development tools



(b) Smartphones



(c) Car brands



(d) Beer brands

Fig. 6 Wall clock time vs. Sequence length t_c (left) and average (right): RFCast consistently wins. It is up to 2 times faster than RegimeCast, 16 times faster than Δ-Spot, and 20 times faster than ARIMA.

is neutral interaction between the activities. On the other side, RFCast still does a good job capturing non-linear dynamics of all the co-evolving sequences, including the exponential growth and the seasonal patterns. Moreover, our forecasted results are very close to the real event streams.

#4. Beer brands. Figure 4 (d) shows the result for the top six beer brands: (i.e., 1: Bud Light, 2: Corona, 3: Keystone, 4: Miller, 5: Coors, 6: Modelo), 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, Modelo (shown as the light green line). Our proposed model successfully identifies the sign of competition between Modelo and other beer brands, which makes Modelo's search volume drops since 2011, where the strongest competitor of Modelo is recognized as Corona. We also execute local streaming of the competition between these two beer brands for the area specification discovery. **Figure 7** shows our results that describe the different dynamical patterns of the competition between two beer brands, Corona and Modelo, in several countries in the world (i.e., AR-Argentina, BR-Brazil, CA-Canada, CL- Chile, CO-Colombia, FR-France,

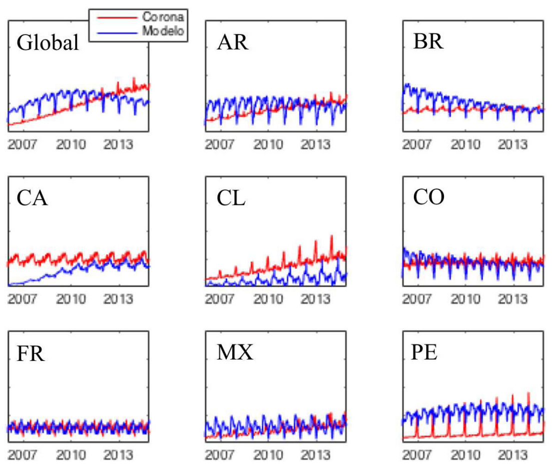


Fig. 7 RFCast captures local-level competition between Modelo and Corona, as well as long-range evolution in each country.

MX-Mexico, PE-Peru). Since we do not use the straightforward approach of applying the global parameter set for all l countries, we can discover various types of dynamical evolution of these two keywords. The variety of competitive situation in those countries show that Corona and Modelo play different marketing strategies in different countries. Also, in terms of the flavor, the Modelo class has a little more body and heavier finish than the Corona class, thus it depends on the different tasting flavors between countries.

5.2 Accuracy

Next, we discuss the quality of RFCast in terms of forecasting accuracy. We compared our method with the following methods: (a) ARIMA, where we determined the optimal parameter set using AIC, (b) Δ-Spot, a unifying non-linear method for online activity analysis and (c) RegimeCast, which is a state-of-the-art forecasting algorithm for complex time series.

Figure 5 shows the forecasting power of RFCast for the four domains of *GoogleTrends* dataset. Specifically, it shows the root mean square error (RMSE) between the original and the three-months-ahead forecasted events (lower is better), 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. Compared to our forecasted result, all the competitors demonstrate higher RMSE; especially, ARIMA is a linear model, thus cannot capture complex, non-linear dynamics and competition signs.

5.3 Scalability

We also evaluate the efficiency of our forecasting algorithm. In **Fig. 6** compares RFCast with RegimeCast, Δ-Spot and ARIMA in terms of computation time for varying sequence lengths t_c. Here the figures are shown in linear-log scales. Since RegimeCast and ARIMA are unable to analyze local-level streams, we only examine the executing speed of all methods for global-level streams. According to the result of four groups of activities, RFCast generates long-range future events, significantly faster than the competitors for the large streams, as we expected. Especially, in the left column of Fig. 6, each spike corresponds to RFCast-Estimator process, which updates the parameter set **F**, estimates the seasonal pattern and detects new sign of competition between activities. The right column of Fig. 6 shows the average computation time of entire event streams. Here, our proposed model shows lower average computation time compared to a state-of-the-art model for event stream mining, RegimeCast, also outperforms two other competitive methods, Δ-Spot and ARIMA in terms of executing time.

6. Conclusions

In this paper, we presented RFCast, an efficient and effective method that focused on the problem of real-time forecasting over co-evolving online activity streams. Our proposed method, RFCast demonstrates all the following desirable properties:

(1) It is **Effective**: RFCast detects non-linear patterns in the co-evolving streams and forecasts long-term future dynamics.
(2) It is **Adaptive**: it requires no training set and no domain expertise, thanks to our coding scheme.
(3) It is **Scalable**: the computation time of RFCast does not depend on data stream length.
(4) It is **Practical**: RFCast provides a response at any time and generates long-range future events.

References

[1] Beutel, A., Prakash, B.A., Rosenfeld, R. and Faloutsos, C.: Interacting viruses in networks: Can both survive? *KDD*, pp.426–434 (2012).
[2] Box, G.E., Jenkins, G.M. and Reinsel, G.C.: *Time Series Analysis: Forecasting and Control*, Prentice Hall, Englewood Cliffs, NJ, 3rd edition (1994).
[3] Chakrabarti, D. and Faloutsos, C.: F4: Large-scale automated forecasting using fractals, *CIKM* (2002).
[4] Choi, H. and Varian, H.R.: Predicting the present with google trends, *The Economic Record*, Vol.88, No.s1, pp.2–9 (2012).
[5] Do, T.M., Matsubara, Y. and Sakurai, Y.: Automatic and effective mining of coevolving online activities, *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (*PAKDD*) (2017).
[6] Goel, S., Hofman, J., Lahaie, S., Pennock, D. and Watts, D.: Predicting consumer behavior with web search, *PNAS* (2010).
[7] Gruhl, D., Guha, R., Kumar, R., Novak, J. and Tomkins, A.: The predictive power of online chatter, *KDD*, pp.78–87 (2005).
[8] Jain, A., Chang, E.Y. and Wang, Y.-F.: Adaptive stream resource management using kalman filters, *SIGMOD*, pp.11–22 (2004).
[9] Li, L., McCann, J., Pollard, N. and Faloutsos, C.: Dynammo: Mining and summarization of coevolving sequences with missing values, *KDD* (2009).
[10] Li, L., Prakash, B.A. and Faloutsos, C.: Parsimonious linear fingerprinting for time series, *PVLDB*, Vol.3, No.1, pp.385–396 (2010).
[11] Livera, A.M.D., Hyndman, R.J. and Snyder, R.D.: Forecasting time series with complex seasonal patterns using exponential smoothing, *Journal of the American Statistical Association*, Vol.106, No.496, pp.1513–1527 (2011).
[12] Matsubara, Y. and Sakurai, Y.: Regime shifts in streams: Real-time forecasting of co-evolving time sequences, *KDD*, pp.1045–1054 (2016).
[13] Matsubara, Y., Sakurai, Y. and Faloutsos, C.: Autoplait: Automatic mining of co-evolving time sequences, *SIGMOD* (2014).
[14] Matsubara, Y., Sakurai, Y. and Faloutsos, C.: The web as a jungle: Non-linear dynamical systems for co-evolving online activities, *WWW*, pp.721–731 (2015).
[15] Matsubara, Y., Sakurai, Y. and Faloutsos, C.: Non-linear mining of competing local activities, In *25th International World Wide Web Conference* (*WWW*), pp.737–747 (2016).
[16] Matsubara, Y., Sakurai, Y., Faloutsos, C., Iwata, T. and Yoshikawa, M.: Fast mining and forecasting of complex time-stamped events, *KDD*, pp.271–279 (2012).
[17] Matsubara, Y., Sakurai, Y., Prakash, B.A., Li, L. and Faloutsos, C.: Rise and fall patterns of information diffusion: Model and implications, *KDD*, pp.6–14 (2012).
[18] Matsubara, Y., Sakurai, Y., van Panhuis, W.G. and Faloutsos, C.: FUNNEL: Automatic mining of spatially coevolving epidemics, *KDD*, pp.105–114 (2014).
[19] Palpanas, T., Vlachos, M., Keogh, E. and Gunopulos, D.: Streaming time series summarization using user-defined amnesic functions, *IEEE Transactions on Knowledge and Data Engineering*, Vol.20, No.7, pp.992–1006 (2008).
[20] Papadimitriou, S., Brockwell, A. and Faloutsos, C.: Adaptive, hands-off stream mining, *VLDB*, pp.560–571 (2003).
[21] Papadimitriou, S., Sun, J. and Faloutsos, C.: Streaming pattern discovery in multiple time-series, *VLDB*, pp.697–708 (2005).
[22] Papadimitriou, S. and Yu, P.S.: Optimal multi-scale patterns in time series streams, *SIGMOD*, pp.647–658 (2006).
[23] Prakash, B.A., Beutel, A., Rosenfeld, R. and Faloutsos, C.: Winner takes all: Competing viruses or ideas on fair-play networks, *WWW*, pp.1037–1046 (2012).
[24] Preis, T., Moat, H.S. and Stanley, H.E.: Quantifying trading behavior in financial markets using google trends, *Sci. Rep.*, Vol.3, p.1684 (2013).
[25] Sakurai, Y., Matsubara, Y. and Faloutsos, C.: Mining and forecasting of big time-series data, *SIGMOD*, pp.919–922 (2015).
[26] Tao, Y., Faloutsos, C., Papadias, D. and Liu, B.: Prediction and indexing of moving objects with unknown motion patterns, *SIGMOD*, pp.611–622 (2004).
[27] Zhu, Y. and Shasha, D.: Statstream: Statistical monitoring of thousands of data streams in real time, *VLDB*, pp.358–369 (2002).
[28] Zoumpatianos, K., Idreos, S. and Palpanas, T.: Indexing for interactive exploration of big data series, *SIGMOD*, pp.1555–1566 (2014).

Thinh Minh Do is a Ph.D. student at Graduate School of Science and Technology, Kumamoto University, Japan. He obtained his B.E. and M.E. degrees from Kumamoto University in 2015 and 2017 respectively. His research interests include web mining and stream processing. He achieved Rakuten Award at WebDB Forum 2015 and Student Travel Award at ACM SIGMOD/PODS 2016.

Yasuko Matsubara is an Associate professor at Osaka University, Japan. She obtained her B.S. and M.S. degrees from Ochanomizu University in 2007 and 2009 respectively, and her Ph.D. from Kyoto University in 2012. She was a visiting researcher at Carnegie Mellon University during 2011–2012 and 2013–2014. She was an Assistant Professor at Kumamoto University, during 2014–2019. Her research interests include time-series data mining and non-linear dynamic systems.

Yasushi Sakurai received his B.E. degree from Doshisha University in 1991, and M.E. and Ph.D. degrees from Nara Institute of Science and Technology in 1996 and 1999, respectively. He joined NTT Laboratories in 1998. He was a visiting researcher at Carnegie Mellon University during 2004–2005. During 2013–2019, he was a Professor at Kumamoto University. Since 2019, he has been a Professor and Vice Director at Artificial Intelligence Research Center, The Institute of Scientific and Industrial Research, Osaka University. He received two KDD best paper awards in 2008 and 2010. His research interests include time-series analysis, web mining, and sensor data processing.

(Editor in Charge: *Keishi Tajima*)