Non-Linear Mining of Competing Local Activities

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

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

Christos Faloutsos
Carnegie Mellon University
christos@cs.cmu.edu

ABSTRACT

Given a large collection of time-evolving activities, such as Google search queries, which consist of d keywords/activities for m locations of duration n, how can we analyze temporal patterns and relationships among all these activities and find location-specific trends? How do we go about capturing non-linear evolutions of local activities and forecasting future patterns? For example, assume that we have the online search volume for multiple keywords, e.g., "Nokia/Nexus/Kindle" or "CNN/BBC" for 236 countries/territories, from 2004 to 2015. Our goal is to analyze a large collection of multi-evolving activities, and specifically, to answer the following questions: *( ) Is there ny sign of inter ction/competition between two different keywords? If so, who competes with whom? (b) In which country is the competition strong? (c) Are there ny se son l/ nnu l ctivities? (d) How c n we utom tic lly detect import nt world-wide (or loc l) events?*

We present COMPCUBE, a unifying non-linear model, which provides a compact and powerful representation of co-evolving activities; and also a novel fitting algorithm, COMPCUBE-FIT, which is *p r meter-free* and *sc l ble*. Our method captures the following important patterns: (**B**)asic trends, i.e., non-linear dynamics of co-evolving activities, signs of (**C**)ompetition and latent interaction, e.g., Nokia vs. Nexus, (**S**)easonality, e.g., a Christmas spike for iPod in the U.S. and Europe, and (**D**)eltas, e.g., unrepeated local events such as the U.S. election in 2008. Thanks to its concise but effective summarization, COMPCUBE can also forecast long-range future activities. Extensive experiments on real datasets demonstrate that COMPCUBE consistently outperforms the best state-of-the-art methods in terms of both accuracy and execution speed.

Categories and Subject Descriptors: H.2.8 [**Database management**]: Database applications–*D t mining*
Keywords: Time-series; Non-linear; Parameter-free; Forecasting;

1. INTRODUCTION

Online news, blogs, SNS and many other b-based services have been attracting considerable interest for business and marketing purposes. For example, the consumer electronics industry (e.g., "Kindle", "Nexus"), news and social media (e.g., "CNN", "BBC", "Yahoo! News") and many other world-wide companies have introduced new online marketing systems and competition has increased significantly. Our goal is to find patterns, relationships and outliers in a large collection of co-evolving online activities, consisting of tuples of the form: *(ctivity, loc tion, time)*. For example, assume that we have the online search volume for multiple keywords, e.g., "Nokia/Nexus/Kindle" for 236 countries/territories. So, how can we find meaningful trends with respect to three aspects: activity, location and time? Specifically, we would like to answer the following questions: *Is there ny sign of competition, e.g., between Noki nd Nexus? Who would be Kindle's most likely competitor? Are there ny se son l ctivities, e.g., Christm s nd New Ye r s les in the US nd Europe? Which countries re interested in US politics? C n we forec st the future evolution of e ch ctivity/keyword in e ch country?*

In this paper, we present COMPCUBE, [1] which answers all of the above questions, and provides a good summary of large collections of co-evolving activities. Intuitively, the problem we wish to solve is as follows:

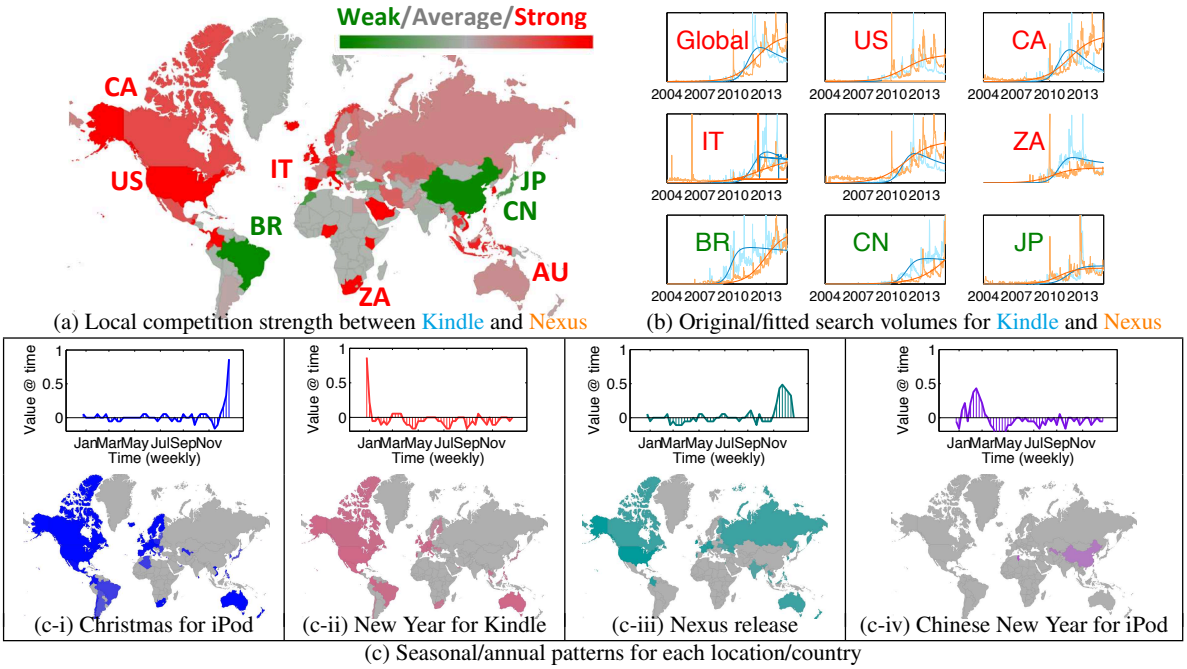
INFORMAL PROBLEM 1. **Given** *l rge collection of triplets (ctivity, loc tion, time), th t is, $\mathcal{X} \in \mathbb{N}^{d \times m \times n}$, which consists of d ctivities/keywords in m loc tions/countries of dur tion n,* **Find** *comp ct description of $\mathcal{X}$, i.e.,*

- *find glob l nd loc l-level inter ction nd competition (e.g., Kindle vs. Nexus in the US)*
- *find loc l se son l/ nnu l p tterns (e.g., Christm s)*
- *spot extern l events (e.g., the US election in 2008)*
- *forec st future ctivities*
- *utom tic lly nd quickly*

Preview of our results. Figure 1 shows some of our discoveries related to the consumer electronics market, specifically, online activities on *Google Se rch*[2] for $d = 10$ keywords (e.g., iPhone, Kindle and Nexus), for $m = 236$ countries and territories, from January 1, 2004 to the present. Note that the original sequences for all keywords can be seen later in Figure 3 *(#1) Products*.
Competition between ctivities: COMPCUBE automatically identifies the most probable competitor for each keyword among all d possible keywords. For example, it discovers there are potential interactions between Kindle and Nexus. Figure 1 (a) describes the strength of the competition between Kindle and Nexus for each country. Red areas (e.g. United States (US) and Canada (CA)) indicate there is a strong competition between two activities, while green areas (e.g., Brazil (BR), China (CN) and Japan (JP)) have a weaker interaction. Figure 1 (b) shows the search volumes (on a weekly basis) for two keywords: Kindle (skyblue) and Nexus (orange) for each country. Here, the original sequences are shown as faint lines, and our fitting results are shown as solid lines. Our algorithm captures location-specific patterns of growth and decline.

[1] http://www.cs.kumamoto-u.ac.jp/~yasuko/software.html
[2] http://www.google.com/trends/

WWW 2016, April 11–15, 2016, Montréal, Québec, Canada.
ACM 978-1-4503-4143-1/16/04.
http://dx.doi.org/10.1145/2872427.2883010.



(a) Local competition strength between Kindle and Nexus

(b) Original/fitted search volumes for Kindle and Nexus

(c-i) Christmas for iPod

(c-ii) New Year for Kindle

(c-iii) Nexus release

(c-iv) Chinese New Year for iPod

(c) Seasonal/annual patterns for each location/country

Figure 1: Modeling power of CompCube for the consumer electronics market, (i.e., iPhone, Samsung Galaxy, Nexus, HTC, iPad, BlackBerry, Nokia, iMac, iPod, Kindle, please also see Figure 3 (#1)). It automatically identifies (a) the latent competition between different products (e.g., Kindle vs. Nexus), (b) local-level patterns of growth/decline as well as (c) seasonal patterns, e.g., Christmas.

For example, there is strong competition in the United States (US), Canada (CA), Italy (IT), Australia (AU) and South Africa (ZA), summarizing the fact that the decline of Kindle coincides with the rising popularity of Nexus. On the other hand, the behavior is different in Brazil (BR), China (CN) and Japan (JP), where Kindle is still growing significantly. This information would be valuable when making marketing decisions.

Se son l/ nnu l p tterns: Figure 1 (c) shows major annual behavior (i.e., seasonality) in the consumer electrics market. For example, Figure 1 (c-i) corresponds to Christmas. The bottom map shows the strength of the annual Christmas spike for iPod in each country. A darker color indicates stronger seasonality, while gray means there is no Christmas spike. As shown on the map, there is a clear Christmas spike in places with Christian majorities, e.g., the US, Europe, South Africa and Australia, while there is no spike in e.g., China and India. Similarly, Figure 1 (c-ii) corresponds to New Year sales for Kindle. Figure 1 (c-iii) shows a typical pattern for Nexus, where many users (e.g., in the US, Europe and Russia) seem to be interested in Nexus during the winter. This is probably because a new Nexus model is usually released every winter (e.g., November). Also, CompCube can identify location-specific seasonal patterns. For example, Figure 1 (c-iv) shows the Chinese New Year spike for iPod in China.

Contributions. The main contribution of this work is the concept and design of CompCube, which has the following properties:

1. **Effective**: It operates on large collections of co-evolving activities and summarizes them succinctly with respect to three aspects (i.e., activity, location, time).
2. **Practical**: It achieves good fits for numerous diverse real data, and provides intuitive explanations for co-evolving activities, such as competition, seasonality and anomalies (i.e., deltas). It also enables long-range forecasting.
3. **Parameter-free**: It is fully automatic, and requires no "magic numbers" and no user-defined parameters.
4. **Scalable**: provide a scalable algorithm, CompCube-Fit, which scales linearly in terms of the input data size, and is thus applicable to long-range sequences.

2. RELATED WORK

provide a survey of the related literature.

Social activity analysis. Analyses of epidemics and social activities have attracted a lot of interest [13, 17, 40, 8, 1, 34, 14]. The work described in [21] studied the rise and fall patterns in the information diffusion process through online social media. The work in [7] investigated the effect of revisits on content popularity, while [33] focused on the daily number of active users. Prakash et al. [30] 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 [22] is a non-linear model for spatially coevolving epidemic tensors, while Eco b [19] is the first attempt to bridge the theoretical modeling of a biological ecosystem and user activities on the b. For online activity analysis, Gruhl et al. [10] explored online "chatter" (e.g., blogging) activity, and measured the actual sales ranks on Amazon.com, while Ginsberg et al. [8] examined a large number of search engine queries tracking influenza epidemics, while [6, 31, 9] studied keyword volume to predict consumer behavior.

Pattern discovery in time series. In recent years, there has been an explosion of interest in mining time-stamped data [4, 36, 29, 28]. Similarity search and pattern discovery in time sequences have attracted huge interest [38, 26, 2, 37, 35, 29, 24, 5]. Here, TriMine [20] is a scalable method for forecasting complex time-stamped events, while, [18] developed AutoPlait, which is a fully-automatic mining algorithm for co-evolving sequences. Rakthanmanon et al. [32] proposed a similarity search algorithm for "trillions of time series" under the dynamic time warping (DT distance.

Contrast with competitors. Table 1 illustrates the relative advantages of our method. Only CompCube matches all requirements.

velets (D T) and Fourier transforms (DFT) can detect bursts and typical patterns, but they cannot detect interactions between

Table 1: Only COMPCUBE meets all specifications.

	Basics&tools			Linear		Non-linear model				
	D F/DFT	PARAFAC	AUTOPLAIT	ARIMA/++	PLIF	SPIKEM	LV/ TA	ECO B	FUNNEL	**COMPCUBE**
Competition	-	-	-	-	-	-	√	√	-	√
Periodicity	√	-	√	√	√	√	-	√	√	√
Local pattern	-	-	-	-	-	-	-	-	√	√
Automatic	-	-	√	-	-	√	√	√	√	√
Forecasting	-	-	-	√	√	√	-	√	√	√
Outliers/deltas	√	√	√	-	√	√	-	-	√	√

multiple co-evolving sequences. PARAFAC [12] is capable of compression and 3-way analysis, but cannot capture non-linear temporal patterns. AutoPlait [18], pHMM [39] have the ability to capture the dynamics of sequences and perform segmentation, however, they are not intended to capture long-range non-linear evolutions.

All the traditional, linear methods are *fund ment lly unsuit ble*: AR, ARIMA and derivatives including A SOM [27], TBATS [16], PLiF [15] and TriMine [20] are all based on *line r* equations, and are thus incapable of modeling data governed by non-linear equations. Also, most of them require parameter tuning.

The susceptible-infected (SI) model, SpikeM [21], the Lotka-Volterra model, TA [30], Eco b [19] and FUNNEL [22] are non-linear, however, they cannot capture co-evolving activities, seasonal spikes, deltas and location specific activities.

In short, none of the existing methods focuses specifically on the automatic mining and forecasting of non-linear dynamics in multi-evolving and competing activities. Please also see section 5 and section 6 for more discussion.

3. PROPOSED MODEL

In this section we present our proposed model. Assume that we receive time-stamped activities of the form (activity, location, time). then have a collection of entries with d unique activities/keywords, m locations and n time ticks. That is, we have a 3rd-order tensor $\mathcal{X} \in \mathbb{N}^{d \times m \times n}$, where the element $x_{il}(t)$ corresponds to the volume of the i-th activity in the l-th location at time tick t. For example, (CNN, US, 01-01-2015; 100) means there were 100 activities/queries/clicks, for the keyword "CNN", in the US, on January 1, 2015.

Intuition behind our method. Our goal is to find a compact description that best summarizes a given co-evolving activity tensor $\mathcal{X}$. So, what exactly are the most important properties that we need to discover? Specifically, we need to answer all of the questions that we mentioned in the introduction. In short, what we would ideally like to discover is,

- (**B**)asic trends: Non-linear evolution of individual activity (e.g., potential popularity and growth rate).
- (**C**)ompetition: Latent interactions between different keywords (e.g., Nokia vs. Nexus).
- (**S**)easonality: Yearly, cyclic, temporal user activities (e.g., Christmas and summer vacation).
- (**D** (Dtionxus)ld-w-25,¥¥

specific trends and patterns. As shown in Figure 1 (b), each country has its own trends and user activities. These activities evolve naturally over time and depend on many factors including custom, education and economy.

MODEL 1. *Let $P_{il}(t)$ be the potenti l popul rity size of ctivity i in the l-th loc tion t time tick t. Our b se model is governed by the following equ tions,*

$$P_{il}(t) = P_{il}(t-1)\left[1 + r_{il}\left(1 - \frac{\sum_{j=1}^{d} c_{ijl} \cdot P_{jl}(t-1)}{K_{il}}\right)\right], \quad (i = 1, \cdots, d;\ l = 1, \cdots, m;\ t = 1, \cdots, n) \qquad (1)$$

where, $r_{il} > 0, K_{il} > 0, c_{iil} = 1, c_{ijl} \geq 0$, nd $P_{il}(0) = p_{il}$.

Model 1 consists of the following parameters,

- p_{il}: initial condition, i.e., popularity size of activity i in the l-th location at time tick $t = 0$ (i.e., $P_{il}(0) = p_{il}$).
- r_{il}: growth rate, i.e., strength of attractiveness of activity i in the l-th location.
- K_{il}: carrying capacity, i.e., available user resources of activity i in the l-th location.
- c_{ijl}: competition coefficient, i.e., effect rate of j-th activity on i-th activity in the l-th location.

Modeling ssumption: assume that competing activities share some of the same user resources. Similar to a biological ecosystem, the numbers of users and user resources are finite. The user resources could be anything, such as user interest/attention, or the amount of time and money they spend. Users cannot use their time/money for multiple purposes simultaneously. At time tick t, the percentage of potential (i.e., available) user resources in the l-th location for activity i (i.e., those who might be interested in activity i) can be described as $\left(1 - \frac{\sum_{j=1}^{d} c_{ijl} P_{jl}(t)}{K_{il}}\right)$, where, c_{ijl} is the competition coefficient, which describes the effect rate of activity j on activity i in the l-th location. If $c_{ijl} = 0$ $(i \neq j)$, there is no interaction between activities i and j in the l-th location, (i.e., "neutralism"). In contrast, if $c_{ijl} = c_{jil} = 1$, this means that these two activities compete with each other in the l-th location, by sharing exactly the same user resource group. If $c_{ijl} = 1, c_{jil} = 0$, there is an asymmetric competitive interaction, (i.e., "amensalism"). In this case, activity i is strongly affected by activity j, while activity j is almost unaffected by activity i.

(S)easonality and (D)eltas/extra-spikes. Let us move on to the next step, namely, how to capture seasonal/annual patterns as well as external events. Each activity (e.g., Kindle, CNN) always has a certain volume of popularity, however, user behavior changes dynamically according to the season, various annual events and customs (e.g., Christmas, summer vacation). should also take account of unrepeated, external events and anomalies, such as the election of President Barack Obama in 2008. To reflect these phenomena, we introduce two additional parameter sets, namely, $s_{il}(t)$: seasonality, and $\delta_{il}(t)$: deltas.

MODEL 2. *Let $V_{il}(t)$ be the estim ted volume of ctivity i in the l-th loc tion t time tick t. Our full model is described s the following equ tions:*

$$V_{il}(t) = P_{il}(t)\left[1 + s_{il}(t \mod n_p)\right] + \delta_{il}(t) \quad (i = 1, \cdots, d;\ l = 1, \cdots, m;\ t = 1, \cdots, n) \qquad (2)$$

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

The estimated volume $V_{il}(t)$ describes how many times activity i appears in the l-th location at time tick t. It depends on the latent popularity size $P_{il}(t)$ and two additional parameter sets,

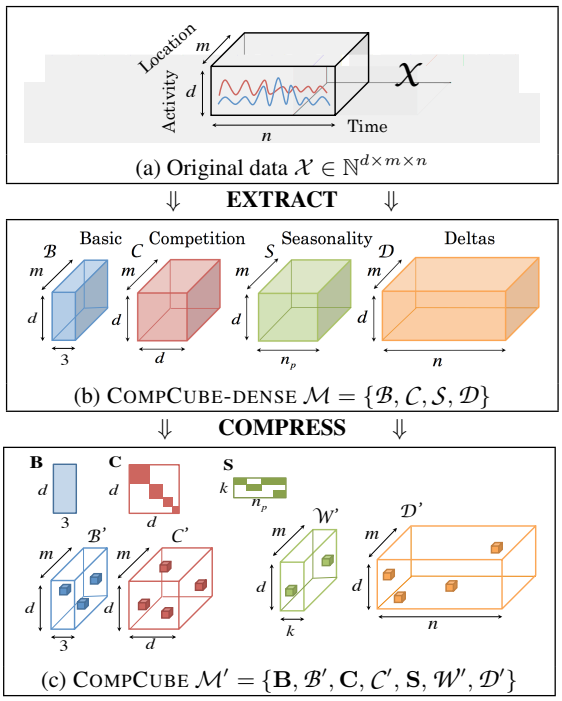


Figure 2: Illustration of CompCube. (a) Given a tensor $\mathcal{X} \in \mathbb{N}^{d \times m \times n}$, (b) it extracts four components/tensors, i.e., basic $\mathcal{B}$, competition $\mathcal{C}$, seasonality $\mathcal{S}$ and deltas $\mathcal{D}$ (we refer to this as CompCube-dense). (c) It then squeezes four dense tensors into global/local-level sparse parameters, i.e., basic $\{\mathbf{B}, \mathcal{B}'\}$ competition $\{\mathbf{C}, \mathcal{C}'\}$, seasonality $\{\mathbf{S}, \mathcal{W}'\}$ and deltas $\mathcal{D}'$.

- $s_{il}(t \mod n_p)$: seasonal/annual trends, i.e., relative value of popularity size $P_{il}(t)$ vs. the actual volume $V_{il}(t)$.
- $\delta_{il}(t)$: deltas, i.e., non-cyclic, completely independent activity of the long-range evolution.

Note that if there is no seasonality or delta for activity i in the l-th location at time t, (i.e., $s_{il}(t \mod n_p) = \delta_{il}(t) = 0$), the estimated volume is equal to the popularity size (i.e., $V_{il}(t) = P_{il}(t)$).

Full parameter set of CompCube-dense. Figure 2 shows our modeling framework. Given a tensor $\mathcal{X}$ (shown in Figure 2 (a)), we first extract four dense tensors (Figure 2 (b)). refer to it as CompCube-dense.

DEFINITION 1 (CompCube-dense). *Let $\mathcal{M} = \{\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}\}$ be full p r meter set of* CompCube-dense, *where,*

- *$\mathcal{B}$ $(d \times 3 \times m)$: b sic/individu l trends, consisting of initi l v lue, growth r te, c rrying c p city, i.e., $\mathcal{B} = \{p_{il}, r_{il}, K_{il}\}_{i,l=1}^{d,m}$.*
- *$\mathcal{C}$ $(d \times d \times m)$: competition coefficients between the i-th nd j-th ctivities in the l-th loc tion, i.e., $\mathcal{C} = \{c_{ijl}\}_{i,j,l=1}^{d,d,m}$.*
- *$\mathcal{S}$ $(d \times n_p \times m)$*

PCUBE-DENSE employs a large number of parameters (i.e., non-zero elements) to describe all the sequences of $\mathcal{X}$. Most importantly, we need to avoid redundancy, that is, the ideal model should provide a compact and powerful representation.

Compression and summarization. We introduce our final model, COMPCUBE. Figure 2 (c) illustrates how this is done. We decompose each tensor (i.e., $\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}$) into a set of sparse components so that we can reconstruct the original dense tensors, i.e.,

$$\mathcal{B} \simeq \mathbf{B} \cdot 2^{\mathcal{B}'}, \quad \mathcal{C} \simeq \mathbf{C} \cdot 2^{\mathcal{C}'}, \quad \mathcal{S} \simeq \mathbf{S} \cdot \mathcal{W}', \quad \mathcal{D} \simeq \mathcal{D}'. \tag{3}$$

Specifically, given dense tensors $\mathcal{B}$ and $\mathcal{C}$, we compress them into matrices $\mathbf{B}, \mathbf{C}$ and sparse tensors $\mathcal{B}', \mathcal{C}'$, where $\mathbf{B}, \mathbf{C}$ correspond to global-level (i.e., aggregated) trends of d activities, and $\mathcal{B}', \mathcal{C}'$ show the logarithms of the relative values of global vs. local-specific trends, i.e., $\mathcal{B}' = \log(\mathcal{B}/\mathbf{B})$, $\mathcal{C}' = \log(\mathcal{C}/\mathbf{C})$. In short, each element in $\mathcal{B}'$ and $\mathcal{C}'$ describes the way in which the behavior at each location differs from global behavior. For example, if $\mathcal{B}' = \mathbf{0}$ for all d activities in all m locations, it is identical to the global trend, i.e., $\mathcal{B} = \mathbf{B} \cdot 2^0 = \mathbf{B}$. Similarly, given a set of seasonal patterns (i.e., $\mathcal{S}$), we decompose it into a seasonal-component matrix $\mathbf{S}$ and a season-weight tensor $\mathcal{W}'$. Here, $\mathbf{S}$ consists of k-components of length n_p, and each component corresponds to an individual annual pattern, such as Christmas or summer vacation, while $\mathcal{W}'$ describes the participation weight of each activity in each location for each seasonal component. Similarly, we want to make tensor $\mathcal{D}'$ as sparse as possible. We should filter out extreme, unusual, non-cyclic events *automatically*. We will explain how this is done in section 4.

Full parameter set of COMPCUBE. Figure 2(c) shows our modeling framework. Our complete model consists of the following:

DEFINITION 2 (COMPCUBE). *Let $\mathcal{M}'$ be a complete set of* COMPCUBE*, namely, $\mathcal{M}' = \{\mathbf{B}, \mathcal{B}', \mathbf{C}, \mathcal{C}', \mathbf{S}, \mathcal{W}', \mathcal{D}'\}$, i.e.,*

- $\mathbf{B}$ $(d \times 3)$*: global-level basic trends of each activity (i.e., initial popularity size, growth rate, carrying capacity).*
- $\mathcal{B}'$ $(d \times 3 \times m)$*: local-level basic trends.*
- $\mathbf{C}$ $(d \times d)$*: global-level competition among all activities.*
- $\mathcal{C}'$ $(d \times d \times m)$*: local-level competition.*
- $\mathbf{S}$ $(k \times n_p)$*: a set of k seasonal components of period n_p.*
- $\mathcal{W}'$ $(d \times k \times m)$*: local participation weight of seasonalities.*
- $\mathcal{D}'$ $(d \times n \times m)$*: sparse tensor of deltas/outliers.*

4. OPTIMIZATION ALGORITHM

So far, we have shown how we summarize all the important patterns of co-evolving activities. Now, the question is *how to find* an optimal solution, and this is exactly the focus of this section. The problem that we want to solve is as follows:

PROBLEM 1. **Given** *a tensor $\mathcal{X} \in \mathbb{N}^{d \times m \times n}$, consisting of d activities in m locations of duration n,* **Find** *a compact description that describes the global/local patterns of co-evolving activities in $\mathcal{X}$, namely, $\mathcal{M}' = \{\mathbf{B}, \mathcal{B}', \mathbf{C}, \mathcal{C}', \mathbf{S}, \mathcal{W}', \mathcal{D}'\}$.*

Model description and data compression. Our goal is to find an optimal solution $\mathcal{M}'$ that solves Problem 1. So, how can we efficiently estimate full model parameters given $\mathcal{X}$? How should we determine the number of seasonal components, k, as well as filter out deltas $\mathcal{D}'$? How can we compress $\mathcal{M}'$, and make it as compact as possible? We want to answer these questions without any parameter fine tuning, that is, *fully automatically*. We thus introduce a new coding scheme, which is based on the minimum description length (MDL) principle. In short, it follows the assumption that the more we can compress the data, the more we can learn about its underlying patterns.

Model description cost: The description complexity of a model parameter set, $Cost_M(\mathcal{M}')$, consists of the following terms,

- The number of activities d, locations m, and time ticks n require $\log^*(d) + \log^*(m) + \log^*(n)$ bits. [4]
- (**B**)asic trends: $Cost_M(\mathbf{B}) = d \cdot 3 \cdot c_F$, $Cost_M(\mathcal{B}') = |\mathcal{B}'| \cdot (\log(d) + \log(3) + \log(m) + c_F) + \log^*(|\mathcal{B}'|)$
- (**C**)ompetition: $Cost_M(\mathbf{C}) = |\mathbf{C}| \cdot (\log(d) + \log(d) + c_F) + \log^*(|\mathbf{C}|)$, $Cost_M(\mathcal{C}') = |\mathcal{C}'| \cdot (\log(d) + \log(d) + \log(m) + c_F) + \log^*(|\mathcal{C}'|)$
- (**S**)easonality: $Cost_M(\mathbf{S}) = |\mathbf{S}| \cdot (\log(k) + \log(n_p) + c_F) + \log^*(|\mathbf{S}|) + \log^*(k)$, $Cost_M(\mathcal{W}') = |\mathcal{W}'| \cdot (\log(d) + \log(k) + \log(m) + c_F) + \log^*(|\mathcal{W}'|)$
- (**D**)eltas: $Cost_M(\mathcal{D}') = |\mathcal{D}'| \cdot (\log(d) + \log(n) + \log(m) + c_F) + \log^*(|\mathcal{D}'|)$

where, $|\cdot|$ describes the number of non-zero elements and c_F is the floating point cost. [5]

Data coding cost: Given a full parameter set $\mathcal{M}'$, we can encode the data $\mathcal{X}$ using Huffman coding [3], i.e., a number of bits is assigned to each value in $\mathcal{X}$, which is the logarithm of the inverse of the probability. The coding cost of $\mathcal{X}$ given $\mathcal{M}'$ is computed by: $Cost_C(\mathcal{X}|\mathcal{M}') = \sum_{i,l,t=1}^{d,m,n} \log_2 p^{-1}_{Gauss(\mu,\sigma^2)}(x_{il}(t) - V_{il}(t))$, where, $x_{il}(t)$ and $V_{il}(t)$ are the original and estimated volume of the i-th activity in the l-th location at time tick t (i.e., Model 2). [6]

Putting it all together: The total code length for $\mathcal{X}$ with respect to a given parameter set $\mathcal{M}'$ can be described as follows:

$$Cost_T(\mathcal{X}; \mathcal{M}') = Cost_M(\mathcal{M}') + Cost_C(\mathcal{X}|\mathcal{M}') \tag{4}$$

Thus, our next goal is to find an optimal solution $\mathcal{M}'$ to minimize the above function.

4.1 Proposed algorithms

What is the best way to find an optimal parameter set $\mathcal{M}'$, *efficiently* and *effectively*? We want to capture important properties of $\mathcal{X}$ from the both global and local perspectives, as well as eliminate meaningless properties and avoid redundancy. We propose an efficient algorithm, COMPCUBE-FIT, which consists of the following sub-algorithms.

1. **GLOBALFIT:** Find global trends and competition (i.e., $\mathbf{B}, \mathbf{C}$), as well as filter out seasonalities/deltas (i.e., $\mathbf{S}, \mathbf{D}$).
2. **LOCALFIT:** Find local-level trends and seasonalities as well as spot location-specific deltas/extras (i.e., $\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}$).
3. **AUTOCOMPRESS:** Find a compact summary $\mathcal{M}'$ of $\mathcal{X}$.

Algorithm 1 shows an overview of COMPCUBE-FIT. Given a tensor $\mathcal{X}$, it first finds global-level parameter sets: $\{\mathbf{B}, \mathbf{C}, \mathbf{S}, \mathbf{D}\}$. Given a tensor $\mathcal{X}$ and global parameters $\{\mathbf{B}, \mathbf{C}\}$, it then finds local-level trends, i.e., four dense tensors, $\{\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}\}$. Finally, it compresses them into a set of compact parameters according to our cost function (i.e., Equation 4) and returns the full parameter set $\mathcal{M}'$.

4.1.1 Global parameter optimization

We now describe our algorithms in steps. Given a tensor $\mathcal{X}$, our first step is to find a set of global-level parameters. Algorithm 2

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

[5] We digitize the floating number so that it has the optimal code length, i.e., $c_F = \log(b_F)$, where b_F is the number of buckets/digits and $b_F = argmin_{b'_F} Cost_T(\mathcal{X}; \mathcal{M}')$. Here, c_F is up to 4×8 bits.

[6] μ and σ^2 are the mean and variance of the distance between the original and estimated values.

Algorithm 1 COMPCUBE-FIT ($\mathcal{X}$)

Input: Tensor $\mathcal{X}$ ($d \times m \times n$)
Output: Full parameter set $\mathcal{M}' = \{\mathbf{B}, \mathcal{B}', \mathbf{C}, \mathcal{C}', \mathbf{S}, \mathcal{W}', \mathcal{D}'\}$.
/* Parameter fitting for global-level activities */
$\{\mathbf{B}, \mathbf{C}, \mathbf{S}, \mathbf{D}\}$ =GLOBALFIT ($\mathcal{X}$);
/* Parameter fitting for local-level activities */
$\{\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}\}$ =LOCALFIT ($\mathcal{X}, \mathbf{B}, \mathbf{C}$);
/* Automatic model compression */
$\{\mathcal{B}', \mathcal{C}', \mathbf{S}, \mathcal{W}', \mathcal{D}'\}$ =AUTOCOMPRESS ($\mathcal{X}, \mathbf{B}, \mathbf{C}, \mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}$);
return $\mathcal{M}' = \{\mathbf{B}, \mathcal{B}', \mathbf{C}, \mathcal{C}', \mathbf{S}, \mathcal{W}', \mathcal{D}'\}$;

Algorithm 2 GLOBALFIT ($\mathcal{X}$)

Input: Tensor $\mathcal{X}$ ($d \times m \times n$)
Output: Global-level parameters $\mathbf{M} = \{\mathbf{B}, \mathbf{C}, \mathbf{S}, \mathbf{D}\}$;
Compute average volumes $\boldsymbol{X}$ ($d \times n$), i.e., $\boldsymbol{X} = \{\bar{\boldsymbol{x}}_i\}_{i=1}^{d}$
Initialize parameter set $\mathbf{M}$
/* (I) Estimate individual parameters */
for $i = 1 : d$ **do**
  $\mathbf{M}_i$ = TETRAFIT($i, \bar{\boldsymbol{x}}_i, \mathbf{M}$); // Parameter fitting for i using $\bar{\boldsymbol{x}}_i$;
end for
/* (II) Estimate competition among all d activities */
while improving the parameters **do**
  /* Select the most unfitted sequence $\bar{\boldsymbol{x}}_i$ */
  $i = \arg\max_{1 \le i' \le d} Cost_T(\bar{\boldsymbol{x}}_{i'}; \mathbf{M})$;
  /* Estimate parameter set $\mathbf{M}'_{ij}$ for each activity $\boldsymbol{x}_j$ */
  for $j = 1 : d$ **do**
    /* Find subset of sequences that have competition with i, j */
    $\boldsymbol{X}_{[i,j]}$ = SUBSETCOLLECTION($\{\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{x}}_j\}, \mathbf{C}$);
    $\mathbf{M}'_{ij}$ = TETRAFIT($i, \boldsymbol{X}_{[i,j]}, \mathbf{M}$); // Fitting for i with $\boldsymbol{X}_{[i,j]}$;
  end for
  /* Find the best competitor $\boldsymbol{x}_j$ of $\boldsymbol{x}_i$, and update $\mathbf{M}_i$ */
  $j = \arg\min_{1 \le j' \le d} Cost_T(\boldsymbol{X}; \mathbf{M}'_{ij'})$; $\mathbf{M}_i = \mathbf{M}'_{ij}$;
end while
return $\mathbf{M} = \{\mathbf{B}, \mathbf{C}, \mathbf{S}, \mathbf{D}\}$;

shows the overall procedure. Let $\boldsymbol{X}$ be the average volume of d activities for all m locations of length n, that is, $\boldsymbol{X} = \{\bar{\boldsymbol{x}}_i\}_{i=1}^{d}$, where $\bar{\boldsymbol{x}}_i = \{\frac{1}{m}\sum_{l=1}^{m} x_{il}(t)\}_{t=1}^{n}$. Given a set of global activities $\boldsymbol{X}$, GLOBALFIT iteratively optimizes each parameter set for each i-th activity.

Specifically, let $\mathbf{M}_i$ be a set of global parameters for activity i, (i.e., $\mathbf{M}_i = \{\mathbf{B}_i, \mathbf{C}_i, \mathbf{S}_i, \mathbf{D}_i\}$) [7]. Here, GLOBALFIT (I) first assumes that there is no competition between the d activities (i.e., it sets $c_{ij} = 0$ ($i \neq j$)), and estimates parameters $\mathbf{M}_i$ for each individual sequence $\bar{\boldsymbol{x}}_i$ ($i = 1, \ldots, d$), separately and independently. Next, (II) it assumes that there is competition between two activities: $\bar{\boldsymbol{x}}_i$ and $\bar{\boldsymbol{x}}_j$. For each iteration, the algorithm searches for the best competitor $\bar{\boldsymbol{x}}_j$ of $\bar{\boldsymbol{x}}_i$, according to the cost function (i.e., Equation 4). It continues the fitting until convergence.

There are two main ideas behind GLOBALFIT: (1) TETRAFIT and (2) SUBSETCOLLECTION.

(1) TETRAFIT. Given a set of sequences $\boldsymbol{X}$, we want to find optimal, purified parameters for $\mathbf{B}$, $\mathbf{C}$, as well as filter out seasonality $\mathbf{S}$ and deltas $\mathbf{D}$. Since each component consists of a large number of parameters, it is extremely expensive to optimize all the parameters simultaneously. thus propose an efficient and effective algorithm, namely TETRAFIT, which optimizes each parameter set of each i-th activity (i.e., $\mathbf{B}_i, \mathbf{C}_i, \mathbf{S}_i, \mathbf{D}_i$) in an alternating way. Algorithm 3 shows the steps performed by TETRAFIT in detail. Given a set of sequences $\boldsymbol{X}$, a current parameter set $\mathbf{M}$, *nd*, index i, it iteratively estimates each parameter set that corresponds to the i-th activity. Here, we use the *Levenberg-M rqu rdt (LM)* algorithm

[7] $\mathbf{B}_i = \{p_i, r_i, K_i\}$, $\mathbf{C}_i = \{c_{ij}\}_{j=1}^{d}$, $\mathbf{S}_i = \{s_i(t)\}_{t=1}^{n_p}$, $\mathbf{D}_i = \{\delta_i(t)\}_{t=1}^{n}$,

Algorithm 3 TETRAFIT (i, $\boldsymbol{X}$, $\mathbf{M}$)

Input: (a) Index: i, (b) Sequences $\boldsymbol{X}$, (c) Current parameter set $\mathbf{M}$
Output: Optimal parameters for i, i.e., $\mathbf{M}_i = \{\mathbf{B}_i, \mathbf{C}_i, \mathbf{S}_i, \mathbf{D}_i\}$
while improving the parameters **do**
  /* (I) Base and competition parameter fitting, i.e., $\mathbf{B}_i, \mathbf{C}_i$ */
  $\{\mathbf{B}_i, \mathbf{C}_i\} = \arg\min_{\mathbf{B}'_i, \mathbf{C}'_i} Cost_T(\boldsymbol{X}; \mathbf{B}, \mathbf{C}, \mathbf{S}, \mathbf{D})$;
  /* (II) Seasonal parameter fitting, i.e., $\mathbf{S}_i$ */
  $\{\mathbf{S}_i\} = \arg\min_{\mathbf{S}'_i} Cost_T(\boldsymbol{X}; \mathbf{B}, \mathbf{C}, \mathbf{S}, \mathbf{D})$;
  /* (III) Find deltas, i.e., $\mathbf{D}_i$ */
  $\{\mathbf{D}_i\} = \arg\min_{\mathbf{D}'_i} Cost_T(\boldsymbol{X}; \mathbf{B}, \mathbf{C}, \mathbf{S}, \mathbf{D})$;
end while
return $\mathbf{M}_i = \{\mathbf{B}_i, \mathbf{C}_i, \mathbf{S}_i, \mathbf{D}_i\}$;

and minimize the cost function (i.e., Equation 4).

For each iteration, however, TETRAFIT still requires $O(d^2 n)$ time to compute the cost function. As described in Equation 1, it needs to calculate all possible combinations of competition, that is, $\{c_{ij}\}_{i,j=1}^{d,d}$. Here, one subtle but important point is that the competition matrix $\mathbf{C}$ is usually very *sp rse*, that is, most elements are zero, (i.e., $c_{ij} = 0$), and so we should ignore unrelated combinations/pairs (i, j). thus introduce our second idea.

(2) SUBSETCOLLECTION. To provide a more efficient solution, we additionally propose a new algorithm, namely SUBSETCOLLECTION, which extracts a small subset of interacting sequences. Specifically, for each step of TETRAFIT, to estimate the model parameters with respect to activity i, we use a subset $\boldsymbol{X}_{[i]} \subset \boldsymbol{X}$ that competes with activity i. Here, the subset $\boldsymbol{X}_{[i]}$ consists of sequences that compete directly (or indirectly) with the i-th sequence $\bar{\boldsymbol{x}}_i$, that is, $\boldsymbol{X}_{[i]} = f(\bar{\boldsymbol{x}}_i)$, where, $f(\bar{\boldsymbol{x}}_i) = \{\bar{\boldsymbol{x}}_i \cup \bar{\boldsymbol{x}}_j \cup f(\bar{\boldsymbol{x}}_j) | {}^{\forall j} c_{ij} > 0\}$. If there is no competition (i.e., ${}^{\forall j} c_{ij} = 0$ ($i \neq j$)), then, $\boldsymbol{X}_{[i]} = \bar{\boldsymbol{x}}_i$. Since the subset $\boldsymbol{X}_{[i]}$ consists of a small number of sequences, this approach greatly reduces the computation time needed for each iteration in TETRAFIT.

4.1.2 *Loc l p r meter optimiz tion*

Next, let us focus on the local-level trends, i.e., $\mathcal{M} = \{\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}\}$. How do we go about capturing local-level trends and find the optimal parameter set $\mathcal{M}$? The most straightforward solution would be simply to apply GLOBALFIT for all m locations, independently and separately, and obtain a local-level parameter set for each l-th location, i.e., $\mathcal{M} = \{\mathbf{M}_l\}_{l=1}^{m}$, ($\mathbf{M}_l = \{\mathbf{B}_l, \mathbf{C}_l, \mathbf{S}_l, \mathbf{D}_l\}$). However, this approach requires us to fit all combinations of competition (i.e., $d \times d$) for all local activities. Also, some of the locations have very sparse sequences, which derails the fitting result. More importantly, we are interested in *both* capturing global-level patterns and competition, *nd*, spotting location-specific trends and anomalies. For example, we want to answer such questions as in which countries are the example of competition (e.g., Nokia vs. Nexus) stronger/weaker than global activities? want to determine *relative* trends and patterns, rather than *independent* and *individu l* trends.

So, how can we deal with this issue? To achieve much better optimization, we propose "sharing" the global competition for all m locations. The idea is that, if there is no local competition between two activities i and j for all m locations, there is no global competition between them. Thus, given a global-level matrix $\mathbf{C}$, our algorithm ignores unrelated pairs (i.e., $\forall i, j, c_{ij} = 0$), and updates the coefficients only if $c_{ij} > 0$. Algorithm 4 shows the details of LOCALFIT. For each activity i in each location l, it finds an optimal parameter set $\mathcal{M}_{il}$. Here, it uses SUBSETCOLLECTION and TETRAFIT for the efficient optimization. Note that it uses global parameters $\mathbf{B}$ and $\mathbf{C}$ as the initial parameter set of TETRAFIT.

Algorithm 4 LOCALFIT ($\mathcal{X}, \mathbf{B}, \mathbf{C}$)

Input: Tensor $\mathcal{X}$, Global parameters $\mathbf{B}, \mathbf{C}$
Output: Local-level parameters i.e., $\mathcal{M} = \{\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}\}$
/* For each i-th activity in l-th location, $\boldsymbol{x}_{il}$ */
for $l = 1 : m$ **do**
  for $i = 1 : d$ **do**
    /* (I) Find subset of sequences that have competition with i */
    $\boldsymbol{X}_{[il]} = \text{SUBSETCOLLECTION}(\boldsymbol{x}_{il}, \mathbf{C})$;
    /* (II) Estimate parameters for $\boldsymbol{x}_{il}$ */
    $\mathcal{M}_{il} = \text{TETRAFIT}(i, \boldsymbol{X}_{[il]}, \{\mathbf{B}, \mathbf{C}\})$;
  end for
end for
return $\mathcal{M} = \{\mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}\}$;

Algorithm 5 AUTOCOMPRESS ($\mathcal{X}, \mathbf{B}, \mathbf{C}, \mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}$)

Input: Tensor $\mathcal{X}$, Global/local-dense parameters $\mathbf{B}, \mathbf{C}, \mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}$
Output: Compressed parameters i.e., $\{\mathcal{B}', \mathcal{C}', \mathbf{S}, \mathcal{W}', \mathcal{D}'\}$
$\mathcal{B}' = \log(\mathcal{B}/\mathbf{B})$; $\mathcal{C}' = \log(\mathcal{C}/\mathbf{C})$; // Compute $\mathcal{B}', \mathcal{C}'$;
$k = 1$; // Find k seasonal components ($k = 1, 2, \cdots$);
while improving the cost **do**
  $\{\mathbf{S}, \mathcal{W}'\} = \text{DECOMPOSE}(\mathcal{S}, k)$; $\{\mathbf{S}, \mathcal{W}'\} = \text{SPARSE}(\mathbf{S}, \mathcal{W}')$;
  $Cost_k = Cost_T(\mathcal{X}; \mathbf{S}, \mathcal{W}', k)$;
  if $Cost_k < Cost_{best}$ **then**
    /* Update best candidate set */
    $Cost_{best} = Cost_k, k_{best} = k; \{\mathbf{S}_{best}, \mathcal{W}'_{best}\} = \{\mathbf{S}, \mathcal{W}'\}$
  end if
  $k++$;
end while
$\{\mathbf{S}, \mathcal{W}'\} = \{\mathbf{S}_{best}, \mathcal{W}'_{best}\}$
/* Compress parameters */
$\{\mathcal{B}', \mathcal{C}', \mathcal{D}'\} = \text{SPARSE}(\mathcal{B}', \mathcal{C}', \mathcal{D}')$;
return $\mathcal{M}' = \{\mathcal{B}', \mathcal{C}', \mathbf{S}, \mathcal{W}', \mathcal{D}'\}$;

4.1.3 Automatic model compression

Given a tensor $\mathcal{X}$ and a set of dense components, $\{\mathbf{B}, \mathbf{C}, \mathcal{B}, \mathcal{C}, \mathcal{S}, \mathcal{D}\}$, our final step is to compress these parameters to make them as compact as possible, and provide an optimal representation $\mathcal{M}'$. We need to avoid redundancy and ignore meaningless patterns, but we do not want to miss any important trends or surprising anomalies. Now, we have two questions: (a) How can we find an optimal set of k seasonal components (i.e., $\mathbf{S}$)? (b) How can we determine the optimal number of seasonal components (i.e., k), and the number of non-zero elements in each tensor (i.e., $|\mathcal{B}'|, |\mathcal{C}'|, |\mathcal{W}'|, |\mathcal{D}'|$)?

As described in Figure 2 (b), the seasonal tensor $\mathcal{S}$ consists of $d \times m$ sequences of length n_p, where each sequence describes the temporal pattern of the i-th activity in the l-th location, e.g., the Christmas spike in December for Kindle in the US. However, it does not provide a good representation: it cannot capture global seasonal trends among multiple activities and locations. We thus propose employing independent component analysis (ICA) [11]. Given a seasonality tensor $\mathcal{S}$, it finds a set of k independent/non-Gaussian components thus minimizing the reconstruction errors (i.e., $\mathcal{S} \simeq \mathbf{S} \cdot \mathcal{W}'$).

With respect to the second question, we want to find the appropriate number of seasonal components (i.e., k), automatically. We also need to determine the number of non-zero variables in each tensor. Our coding scheme enables us to provide the answer. Our cost function (i.e., Equation 4) determines the optimal number of seasonal components, k, and also makes each tensor as sparse as possible. For example, if the element c_{ijl} in $\mathcal{C}'$ is very small and negligible, the cost function suggests that we ignore this variable, and set it at zero, i.e., $c_{ijl} = 0$. Algorithm 5 describes the overall procedure. It first computes $\mathcal{B}', \mathcal{C}'$ according to Equation 3. It then finds a set of k seasonal components using ICA. Finally, it compresses each component to minimize the cost function (Equation 4).

5. EXPERIMENTS

In this section we demonstrate the effectiveness of COMPCUBE on real datasets. To ensure the repeatability of our results, we used publicly available datasets. The experiments were designed to answer the following questions:

- Q1 *Effectiveness*: How well does it explain important patterns in given input data?
- Q2 *Accuracy*: How well does it fit real datasets?
- Q3 *Scalability*: How does it scale in terms of computational time?

5.1 Q1. Effectiveness - discoveries

We demonstrate how effectively COMPCUBE discovers important patterns in given data $\mathcal{X}$. We performed experiments on eight datasets, which were taken from different domains on *GoogleTrend*, namely, *(#1) Products*, *(#2) News sources*, *(#3) Beer*, *(#4) Cocktails*, *(#5) Car companies*, *(#6) Social media sites*, *(#7) Financial companies* and *(#8) Software*. We picked up the top-10 major keywords for each domain, and specifically, each dataset consists of the $d = 10$ keywords for $m = 236$ countries and territories, from January 1, 2004 to the present. Here, we describe our major discoveries, which correspond to the four properties of online activities.

(B)asic trends. Figure 3 (a) shows our fitting results for eight datasets. The original activities are shown as faint lines, and our estimated volumes are shown as solid lines. COMPCUBE successfully captured the long-range evolution and exponential growing patterns in all co-evolving activities for every dataset. For example, with *(#1) Products* in Figure 3 (a), from 2004 to 2012, most of the keywords (such as iPhone and Nexus) grew steadily, with Nokia and iPod being the exceptions (arguably due to the appearance of Android-based products).

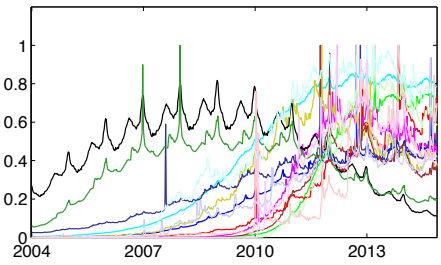
(C)ompetition. Figure 3 (b) shows major examples of competition for each dataset. COMPCUBE automatically detected latent competition among multiple activities/keywords. For example, it found that there was interaction between Kindle, Nokia and Nexus (*(#1) Products*). As we have already shown in the introduction section (Figure 1 (a)), our method can also capture local-level competition.

Figure 4 (a-i) and (a-ii) show the local competition between Modelo and Corona (*(#3) Beer*). Modelo and Corona are popular lagers that are produced by *Grupo Modelo* in Mexico. Compared with Corona, which is growing steadily, Modelo is declining significantly, especially in e.g., Mexico and Brazil. However, it behaves in different ways in several other countries, such as Guatemala (GT) and Chile (CL).

Figure 4 (b-i) and (b-ii) show the local competition between HTML and HTML5 (*(#8) Software*). There are no location-specific trends: most countries behave similarly to the global activities. For each country, the user attention/interest for HTML5 increases exponentially, while HTML is declining significantly.

(S)easonality. As shown in Figure 3 (a), COMPCUBE successfully captured yearly-cyclic patterns for all datasets. For example, *(#1) Products*, *(#3) Beer* and *(#8) Software* have strong seasonal patterns (e.g., Christmas and summer vacation), while there is no clear seasonality for *(#2) News sources*. With respect to location-specific seasonality, Figure 4 (a-iii) shows the major trend of Coors, which is a popular beer, brewed in Colorado, US. Our method discovered that many users in the US and Canada are interested in Coors during the summer, which is a season where drinking beer is popular.

Figure 4 (b-iii) shows the major seasonality for *(#8) Software*, which corresponds to New Year's breaks. The bottom map shows the strength of the seasonality for XML for each country. Note that we observed similar trends (i.e., a sudden decline around New


1
0.8
0.6
0.4
0.2
0
2004
2007
2010
2013

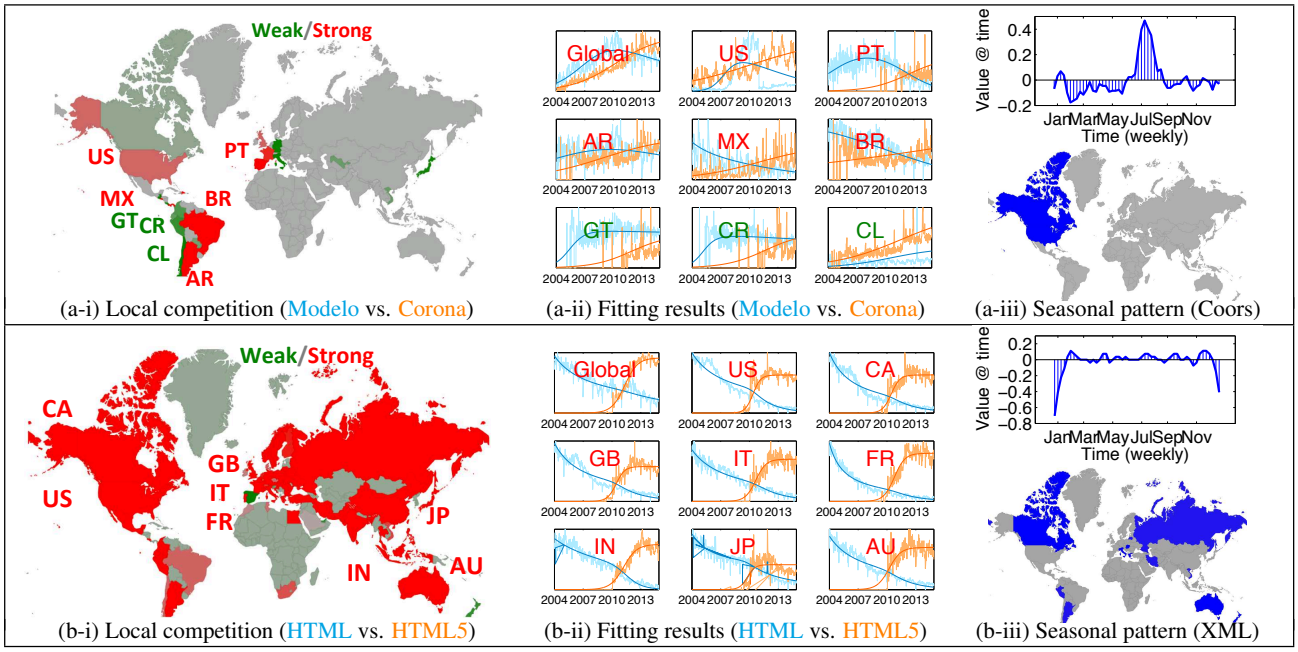


Figure 4: Discovery of local patterns for *(#3) Beer* and *(#8) Softw re*. COMPCUBE captures local-level competition coefficients between (a-i) Modelo and Corona and between (b-i) HTML and HTML5, as well as long-range evolutions in each location (as shown in (a-ii) and (b-ii)). It also detects seasonal patterns, e.g., (a-iii) the summer spike for Coors beer in the US and Canada, and (b-iii) the New Year holiday for XML in Europe, AU, etc.

(a) CNN: Nov 2-8, 2008 (US election)

(b) Tesla: Jul 5-11, 2009 (Google Doodle)

Figure 5: COMPCUBE automatically detects world-wide events, i.e., (D)eltas. (a) Local trend for CNN news: "Barack Obama presidential election" in 2008. (b) Google honored the great inventor: "Nikola Tesla" on his birthday on July 10, 2009. Please also see Figure 3 (*(#2) News sources* and *(#5) C r comp nies*).

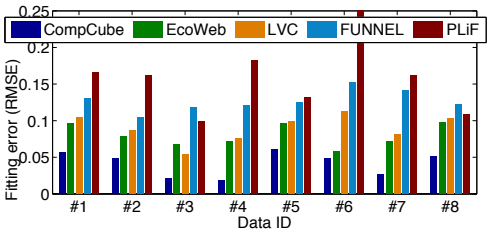
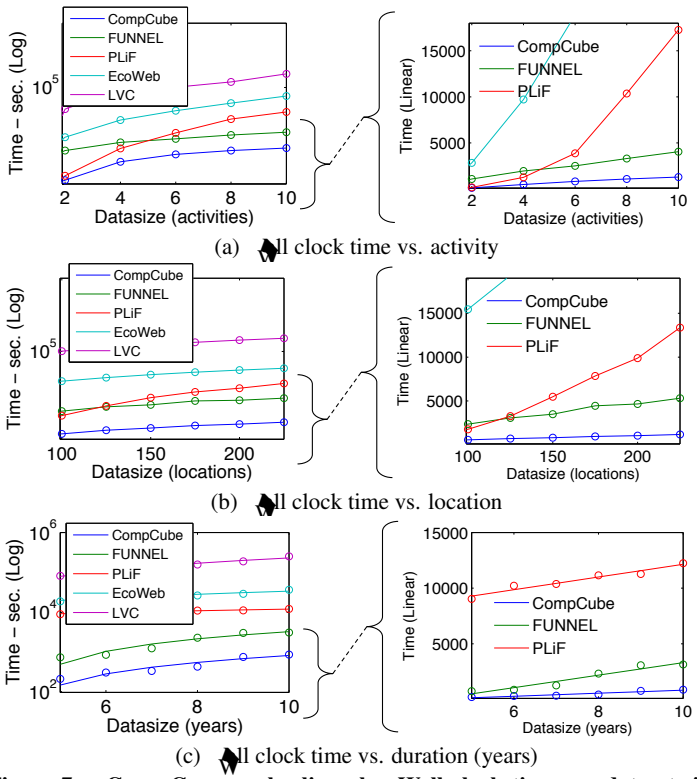


Figure 6: COMPCUBE consistently wins: Average fitting error (RMSE) between original ($x_{il}(t)$) and fitted ($V_{il}(t)$) volumes for each dataset (lower is better).



(a) Wall clock time vs. activity

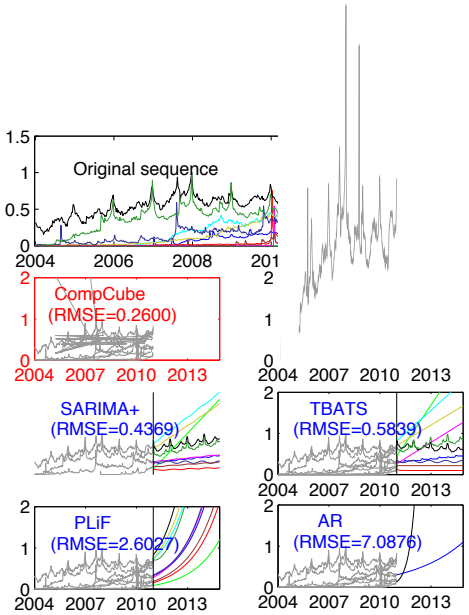
(b) Wall clock time vs. location

(c) Wall clock time vs. duration (years)

Figure 7: COMPCUBE scales linearly: Wall clock time vs. dataset size, i.e., (a) activity d, (b) location m, (c) duration n, shown in (left) linear-log scale to accommodate slow competitors, and (right) magnification, in linear-linear scale. Our method is linear on the data size.

Year) for other keywords (e.g., Java and SQL). Clearly, most developers and hackers stop coding during their vacations.

(D)eltas. Figure 5 shows two major global events corresponding to (a) CNN news: the United States presidential election in 2008 and (b) the Google Doodle for Nikola Tesla's birthday on July 10, 2009 (please see the red circles in Figure 3 *(#2) News sources* and *(#5) C r comp nies*). Figure 5 (a) indicates that CNN US political news attracts attention in the US, Canada, Europe, South Africa, Japan, Korea and elsewhere (i.e., apparently, English-speaking countries, and/or trading partners). Similarly, Figure 5 (b) reveals that technology-oriented countries (e.g., US, Canada, India and Brazil) seem to be interested in one of science's most important inventors, Nikola Tesla.



5.2 Q2. Accuracy

We now discuss the quality of CompCube in terms of fitting accuracy. We compared CompCube with the following state-of-the-art methods: (a) PLiF [15], i.e., linear dynamical systems for co-evolving time sequences, (b) FUNNEL [22], i.e., a non-linear model for spatially-coevolving epidemics, (c) LVC [23], i.e., traditional non-linear equations for biological competition, and (d) Eco B [19], i.e., non-linear dynamical systems for online user activities for a single space. Figure 6 shows the root mean square error (RMSE) between the original and estimated volumes for eight datasets (#1-#8). A lower value indicates a better fitting accuracy. As shown in the figure, our approach achieved high fitting accuracy, while (a) PLiF is *line r*, i.e., cannot capture non-linear evolutions, (b) FUNNEL has the ability to capture non-linear growing patterns as well as external spikes, but cannot capture competition between different sequences, (c) LVC can detect competition, but cannot capture seasonal dynamics, and (d) Eco B cannot handle extreme spikes or local trends.

5.3 Q3. Scalability

We performed experiments to evaluate the efficiency of our optimization algorithm, which we discussed in section 4. Figure 7 compares CompCube with other methods in terms of computation time for varying the dataset size with respect to (a) activity d, (b) location m, and (c) duration n. Note that each result is shown in both linear-log (left) and linear-linear (right) scales. For PLiF, we set $k = 5$ hidden variables and $iter = 20$, and treated the input tensor as a set of $d \times m$ sequences. As shown in the figure, CompCube is linear with respect to the data size.

Consequently, thanks to our carefully designed algorithms as described in section 4, our method achieved a large reduction in both computation time and fitting error.

6. COMPCUBE AT WORK - FORECASTING

As we discussed in the previous section, CompCube is capable of handling various types of co-evolving activities. Here, we tackle the most important and challenging task, namely, forecasting the non-linear dynamics of co-evolving activities. We compared our method with the following forecasting methods: (a) FUNNEL [22], (b) SARIMA+, i.e., seasonal ARIMA (we set $n_p = 52$ weeks), where we determined the optimal parameter set using AIC, (c) TBATS [16], i.e., a state-of-the-art forecasting algorithm for complex seasonal time series (we set $n_p = 52$ weeks), (d) PLiF [15] (we set $k = 5$), and (e) AR, i.e., a traditional forecasting algorithm (we set the order $p = 5$ weeks).

Figure 8 shows the forecasting power of CompCube for *(#1) Products*. As discussed in section 2, SARIMA+, TBATS, PLiF and AR are *unsuit ble* for capturing non-linear dynamics; they are *line r* models, and they can go to infinity over time. FUNNEL, on

8. REFERENCES

[1] N. Armenatzoglou, H. Pham, V. Ntranos, D. Papadias, and C. Shahabi. Real-time multi-criteria social graph partitioning: A game theoretic approach. In *SIGMOD*, pages 1617–1628, 2015.

[2] A. Banerjee and J. Ghosh. Clickstream clustering using weighted longest common subsequences. In *In Proceedings of the Web Mining Workshop t the 1st SIAM Conference on D t Mining*, pages 33–40, 2001.

[3] C. Böhm, C. Faloutsos, J.-Y. Pan, and C. Plant. Ric: Parameter-free noise-robust clustering. *TKDD*, 1(3), 2007.

[4] G. E. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series An lysis: Forec sting nd Control*. Prentice Hall, Englewood Cliffs, NJ, 3rd edition, 1994.

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

[6] H. Choi and H. R. Varian. Predicting the present with google trends. *The Economic Record*, 88(s1):2–9, 2012.

[7] F. Figueiredo, J. M. Almeida, Y. Matsubara, B. Ribeiro, and C. Faloutsos. Revisit behavior in social media: The phoenix-r model and discoveries. In *PKDD*, pages 386–401, 2014.

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

[9] S. Goel, J. Hofman, S. Lahaie, D. Pennock, and D. tts. Predicting consumer behavior with web search. *PNAS*, 2010.

[10] D. Gruhl, R. Guha, R. Kumar, J. Novak, and A. Tomkins. The predictive power of online chatter. In *KDD*, pages 78–87, 2005.

[11] A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. *Neur l Netw.*, 13(4-5):411–430, 2000.

[12] T. G. Kolda and B. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[13] R. Kumar, M. Mahdian, and M. McGlohon. Dynamics of conversations. In *KDD*, pages 553–562, 2010.

[14] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *KDD*, pages 462–470, 2008.

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

[16] A. M. D. Livera, R. J. Hyndman, and R. D. Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journ l of the Americ n St tistic l Associ tion*, 106(496):1513–1527, 2011.

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

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

[19] 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.

[20] 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.

[21] 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.

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

[23] R. M. May. Qualitative stability in model ecosystems. *Ecology*, 54(3):638–641, 1973.

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

[25] J. Murray. *M them tic l Biology II: Sp ti l Models nd Biomedic l Applic tions*. Intercisciplinary Applied Mathematics: Mathematical Biology. Springer, 2003.

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

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

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

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

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

[31] T. Preis, H. S. Moat, and H. E. Stanley. Quantifying trading behavior in financial markets using google trends. *Sci. Rep.*, 3, 04 2013.

[32] T. Rakthanmanon, B. J. L. Campana, A. Mueen, G. E. A. P. A. Batista, M. B. stover, 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.

[33] B. Ribeiro. Modeling and predicting the growth and death of membership-based websites. In *WWW*, pages 653–664, 2014.

[34] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th Intern tion l Conference on World Wide Web*, '10, pages 851–860, 2010.

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

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

[37] Y. Sakurai, M. Yoshikawa, and C. Faloutsos. Ftw: Fast similarity search under the time warping distance. In *PODS*, pages 326–337, Baltimore, Maryland, June 2005.

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

[39] P. ng, H. ng, and ng. Finding semantics in time series. In *SIGMOD Conference*, pages 385–396, 2011.

[40] L. Zhu, A. Galstyan, J. Cheng, and K. Lerman. Tripartite graph clustering for dynamic sentiment analysis on social media. In *SIGMOD*, pages 1531–1542, 2014.