Understanding and Engineering Social Dynamics

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in
Department of Electrical and Computer Engineering

Huan-Kai Peng

M.Sc., Computer Science, National Tsing Hua University
B.Sc., Computer Science, National Tsing Hua University

Carnegie Mellon University
Pittsburgh, Pennsylvania

February 3, 2016

Abstract

All activities on social media evolve with time. Consequently, being able to understand and engineer *social dynamics*, the way how various properties of social media evolve, is a central question for social networks research. While recent work has studied social dynamics from various angles, two important properties of social dynamics are yet to be addressed, i.e., heterogeneous features and signatures at multiple time scales. However, considering heterogeneous features is necessary to build a general tool with wide applicability, whereas considering multiple time scales is indispensable to study how social dynamics in different time scales interact with one another.

In this thesis, we aim at addressing these two properties using computational algorithms with statistical groundings. In particular, we propose scalable and effective methods for three basic tasks: pattern mining, structure decomposition, and data-driven dynamics engineering. For each task, the proposed methods are analyzed formally and verified empirically. The results reveal several interesting insights and demonstrate various practical applications, such as dynamics prediction, anomaly detection, and targeted intervention. Finally, the methods we propose in this thesis are general enough to handle multi-dimensional time series; we have explored this direction by considering other applications, such as human behavior recognition and macroeconomics.

Acknowledgments

First and foremost, I would like to thank my advisor Prof. Radu Marculescu. His motivation, inspiration, knowledge, and enthusiasm have guided me over the course of my whole PhD study while making this journey enjoyable. I would also like to thank my thesis committee: Prof. Christos Faloutsos, Prof. Rick Carley, and Dr. Tim Jia-Yu Pan. Their insights and comments have improved the breadth and the depth of this thesis.

My sincere gratitude also goes to Prof. Larry Wasserman, Prof. Ryan Tibshirani, Prof. Aarti Singh, Prof. Geoff Gordon, and Prof. Eric Xing. Their knowledge and patience helped build my mathematical foundation that is indispensable for this thesis. I also thank my co-authors: Prof. Diana Marculescu, Prof. Felix Wu, Dr. Ying Zhang, Dr. Rong Yan, Dr. Peter Pirolli, Dr. Tad Hogg, Dr. Da-Cheng Juan, Dr. Hao-Chih Lee, Dr. Dongzhen Piao, Dr. Keith Wang, Dr. Jiang Zhu, and Pang Wu; further, I thank Harshavardhan Pandit and Ben Siegel for implementation help. It was my privilege to work with all of them. In addition, the funding of the United States National Science Foundation, via Grant CCF-1314876, is gratefully acknowledged.

Moreover, I would like to thank my close friends: Kuan-Chieh Chen, Tsung-Hsien Lee, Wei-Chun Lin, and Yuster Yu. Your friendship and support has accompanied me from a very long time ago, which I plan to keep forever. Finally, I would like to thank my family: my father Jing-Shian Peng, my mother Tsui-Li Fang, my sister Hai-Jung Peng, and my wife Julliana Hsu. Your love, caring, and support have forged all aspects of my life. I dedicate this thesis to you.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Social media are driven by their massive amounts of users and the dynamical interactions among them. As a result, social media constantly emit rich temporal dynamics. Such *social dynamics*, i.e., the evolution of various properties of social media, are therefore a subject of great interests and importance. The reason is twofold. On one hand, *understanding* social dynamics can bring insights to the basic mechanisms of social interaction. On the other hand, being able to *engineer* social dynamics can offer a systematic way to utilize social media, with many important applications including political campaigning [61], viral marketing [23], or disaster response [72].

While recent work has investigated methods for pattern mining [2, 61, 44, 78, 29] and forecasting [57, 68, 67] of social dynamics, these approaches are still limited in their ability of addressing the following properties of social dynamics:

P1. *Heterogeneity*: social dynamics cannot be described by a single, homogeneous feature; instead, they need to be described by multiple (heterogeneous) ones [51, 52];

P2. *Multi-scale compositionality*: social dynamics are characterized by signatures at multiple time scales; these signatures can further interact with one another to generate higher-level meanings [53, 54].

Both properties are critically important for social dynamics. Indeed, we have demonstrated in [51] that not considering heterogeneity (P1) can mix up distinct types of social interactions that have radically different consequences and courses of action. Also, we have demonstrated in [53] that different signatures that characterize social dynamics can manifest at multiple time scales and generate different meanings (P2). Clearly, this calls for systematic methods to address these issues.

In this thesis, we address the above properties using computational algorithms that have statistical groundings. In particular, we focus on the following three basic tasks:

T1. *Pattern mining* [51, 52], i.e., identifying the representative patterns of social dynamics while considering P1.

T2. *Structure decomposition* [53], i.e., identifying the compositional structures of social dynamics while considering P2.

T3. *Data-driven dynamics engineering* [54], i.e., identifying the ideal intervention dynamics to influence social dynamics toward predefined goals while considering both P1 and P2.

Evidently, T1 and T2 fall in the "understanding" category, whereas T3 falls in the "engineering" category.

The contribution of this thesis are three-fold:

- *Novelty of Problem.* This is the first time that properties P1 and P2 of social dynamics are explicitly considered. This is also the first time that the "dynamic engineering problem" is introduced and addressed.

- *Design, Analysis, and Verification of Proposed Methods.* Each of the proposed methods (i.e., for T1, T2, and T3) is analyzed formally and verified empirically; their advantages are demonstrated in terms of both runtime scalability and solution quality.

- *Applications.* The proposed methods are applied to social media datasets, such as Twitter and Yelp. The results reveal several interesting insights and demonstrate many practical applications, such as dynamics forecasting, anomaly detection, and targeted intervention.

Taken together, our proposed methods to address T1 ~ T3 make it possible to overcome the fundamental limitations of the current state-of-the-art and hence improve our capability to understand and engineer the social dynamics.

From a broader context, the methods we propose are general enough to work with other multi-dimensional time series where P1 and P2 are present. We further demonstrate this point by exploring two other applications, namely, human behavior recognition and macroeconomics pattern mining.

The ultimate goal of this thesis is to build a comprehensive system for automatic understanding and engineering of social dynamics. That is, from the raw streaming data, such a system should be capable of mining and updating the patterns and hidden structures of data, indicating anomalies, recommending interventions, evaluating their effectiveness, and then using the feedback to update the patterns and structures again. Although such a system can be useful to many applications, several challenges are still yet to be addressed. Such challenges include the dynamic updates of patterns and structures, the efficient evaluation and implementation of interventions, and the feedback from interventions to further updates of patterns and structures. We hope the work in this thesis serves as the starting point toward answering all these challenges in the future.

Chapter 2

Background

2.1 Social Interaction

In order to define social dynamics, we first need to define *social interaction.* Suppose we are given a *social graph* (e.g., the upper-left of Figure 2.1), in which the edge $A \rightarrow C$ denotes that user C receives information from user A. Also, suppose we are given the *interaction trace* of what happens in this social graph (e.g., the upper-right of Figure 2.1). Based on the two, we define *social interaction* for each *information token* (or, simply, a *token*), which can be a keyword, an Youtube video, a photo, or a hashtag.

For example, we can use the keyword *Steelers* to construct the social interaction at the bottom of Figure 2.1. At $t = 1$, the users A and B initiated the discussion about the keyword (as *initiators*). Then at $t = 2$, the users C, D, and E act as *propagators* who forward this keyword *as is*, e.g., by retweeting from the users A and B. Later at $t = 3$, user B provide some follow-up comments about this message as a *commentator.* Naturally, each social interaction can be represented as an *evolving graph*, which is studied previously by various authors [34, 14, 68, 10, 42, 30, 33].

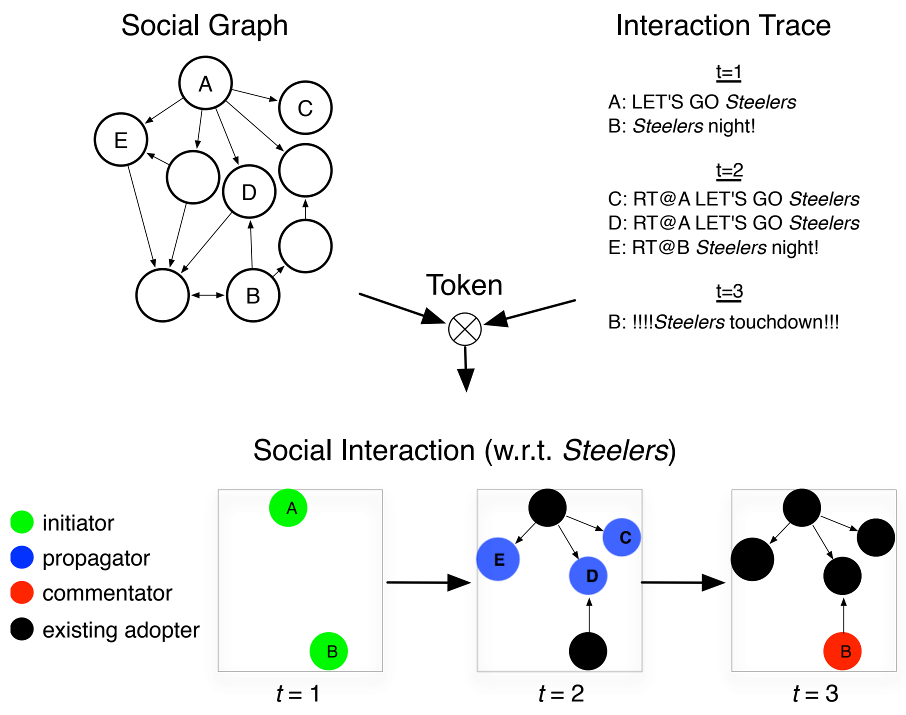

Social Graph
Interaction Trace
t=1
A: LET'S GO *Steelers*
B: *Steelers* night!
t=2
C: RT@A LET'S GO *Steelers*
D: RT@A LET'S GO *Steelers*
E: RT@B *Steelers* night!
t=3
B: !!!!*Steelers* touchdown!!!
Token
Social Interaction (w.r.t. *Steelers*)
initiator
propagator
commentator
existing adopter
t = 1
t = 2
t = 3

Figure 2.1: Illustration of Social Interaction. Based on a *social graph* and an *interaction trace*, each *token* can be used to define its corresponding *social interaction*.

2.2 Social Dynamics

The focus of this thesis, in contrast, is *social dynamics*, i.e., the time series that are extracted from the aforementioned evolving graph. For example, from the evolving graph on the top of Figure 2.2, we can extract the number of various types of users (e.g., initiators, propagators, and commentators). Besides, we can also calculate the size of the largest connected component (LCC) and the diameter of the evolving graph.

While this is true that the evolving graph contains strictly more information, the social dynamics (i.e., the time series being extracted) can still reveal insights that are not obvious by directly inspecting the evolving graph. For example, the social dynamics in Figure 2.2 shows that the token is driven first by a few initiators, then by several propagators, and finally by commentators. The LCC saturates in size and the diameter shrinks slowly, capturing the birth, growth, resurgence, and containment of such a social interaction. Such insights, as reported by the various authors who study evolving graphs [34, 14, 68, 10, 42, 30, 33], are useful in many applications.

In this thesis, we denote the social dynamics using $X \in R^{D \times T}$, where D denotes its dimensionality and T denotes its temporal length. For example, in the case of Figure 2.2, we have $D = 5$, considering the five features denoted in different colors, and $T = 5$, considering the five discrete time steps.

It is worth mentioning that in our experiments, we only use non-negative dynamics (i.e., $X \in R_{+}^{D \times T}$); however, this is not a necessary assumption in terms of modeling. Furthermore, in general, $X \in R^{D \times T}$ doesn't really need to be social dynamics. That is, the methods we propose in this thesis can apply generally to multi-dimensional time series. We will discuss this in Chapter 6.

2.3 Heterogeneity

As mentioned in Chapter 1 (and later in Chapter 3), Heterogeneity and Multi-scale Compositionality are two fundamental properties of social dynamics that are not yet explored in literature. Heterogeneity of social dynamics means that social dynamics cannot be captured by a single, homogeneous feature; instead, they need to be described by multiple heterogeneous ones. However, instead of considering the heterogeneous properties of social dynamics, most prior work focuses on a simplified version, i.e., the *usage dynamics*, defined by the number of usages over time. To illustrate, we plot the usage dynamics of two terms on Twitter, namely, *hospital* and *theonlinemom*, in the left panel of Figure 2.3. In this plot, the black lines represent actual per-minute usage, while the red lines represent the smoothed version of them.

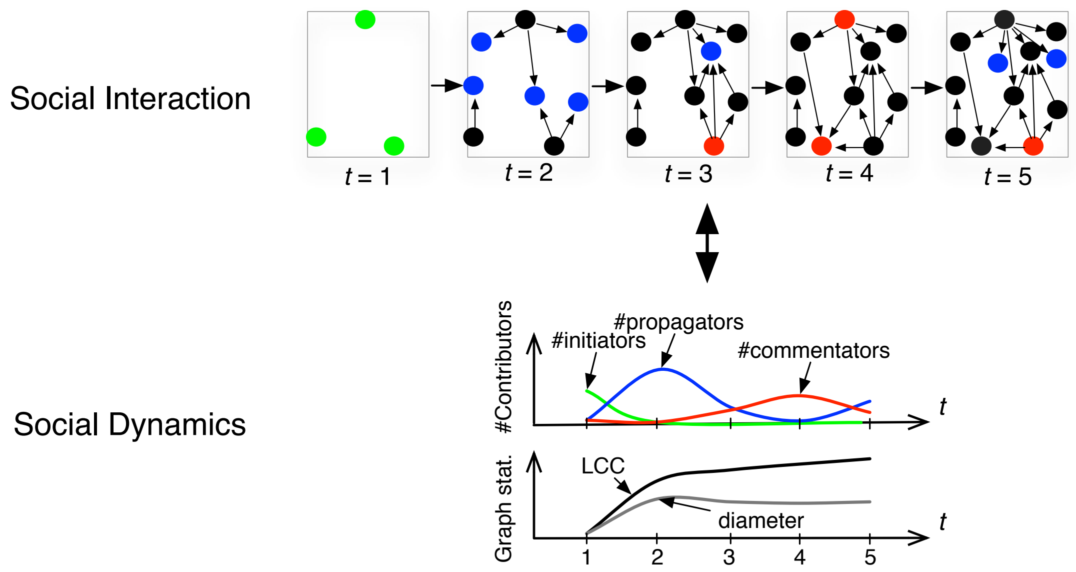


Figure 2.2: Illustration of Social Dynamics. (a) The social interactions among several types of users; (b) Time-series representation of such social interactions.

Looking at the usage dynamics, these two terms share a common signature characterized by two peaks: a smaller one followed by a stronger one. This is a common pattern that is reported by many authors [78, 29, 57][1].

To explore this pattern further, we break down the usages into original tweets (`orig`), propagation tweet/retweet (`propTw` and `propRT`), and repetitive tweet/retweet (`repTw` and `repRT`) and plot them in the right panel of Figure 2.3. A propagation tweet/retweet denotes the case when the author uses the term shortly after his friends did. Similarly, a repetitive tweet/retweet denotes the case when the author uses the term shortly after himself did.

Incorporating this additional information, the two terms now look surprisingly different. Indeed, *hospital* is retweeted intensively followed by several original tweets,

[1]This pattern is referred in Figure 7(e) of [78], the leftmost panel of Figure 1 in [29], and Figure 2 of [57].

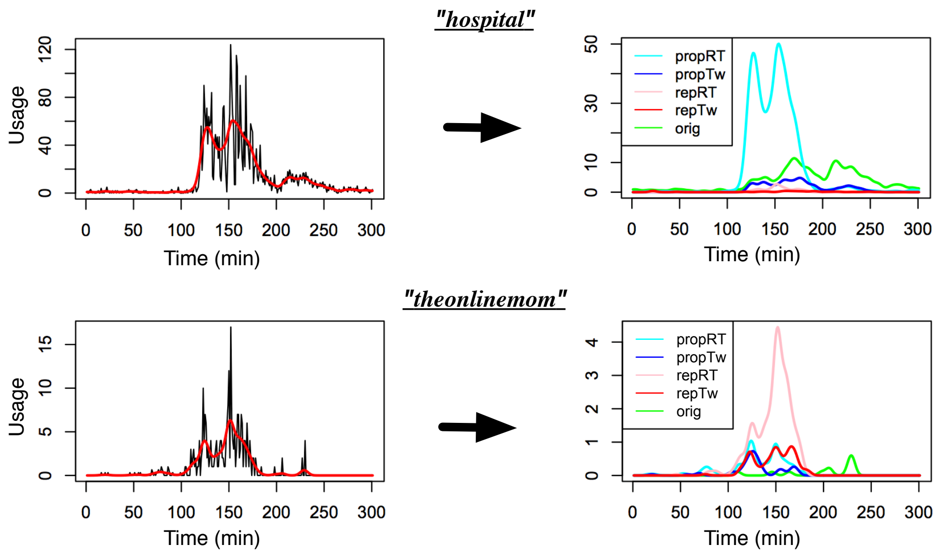


Figure 2.3: Need to Consider Heterogeneity of Social Dynamics. Left: before considering heterogeneity, the difference between the two social dynamics is not observable; right: after considering heterogeneity, their differences become obvious.

corresponding to a viral broadcast of the death of Michael Jackson. For *theonlinemom*, the usages are mainly contributed by repetitive tweets and retweets, corresponding to a back-and-forth discussion on parenting tips. Consequently, we see that quite similar usage dynamics can be actually driven by radically different social dynamics.

We note that the price of mixing up the two cases is high, because the course of *action* one would take to promote these two dynamics are very different. In the first case, one would focus on making the dynamics propagate broad and fast. In the second case, however, one would probably change tactics and focus on motivating and retaining a relatively smaller, core group of users. This clearly demonstrates the need to consider heterogeneity when studying social dynamics.

2.4 Multi-scale Compositionality

Studying the multi-scale compositionality of social dynamics consists of two components. First, the *identification of multi-scale signatures* consists of identifying distinct signatures across a wide range of time scales, as opposed to sticking with a single one. Second, the *mining of compositional interactions* consists of discovering the interaction among multiple such signatures that produce higher-level meanings. To illustrate these ideas, consider the case of human face recognition, where the first component includes recognizing the eyebrows, the cheeks, or the overall head shape. In contrast, the second component includes gauging the distance between the eyebrows, measuring the angle between the jaw and the ears, or recognizing the polygon formed by the lips, cheeks, and eyebrows. To recognize a human face, both components are equally important: one could make a mistake by either recognizing the wrong shape of an eyebrow, or by over/underestimating the distance between the eyebrows.

In the context of social dynamics, we find the same two components as being equally relevant. Indeed, social media exhibit distinct signatures at various time scales that range from seconds to days, whereas different combinations of such signatures can have totally different meanings and consequences. For example, an intense popularity of some keywords followed by a vibrant discussion may indicate a trendy event; however, the same popularity without any follow-up discussion can, on the contrary, indicate an internet scam. Clearly, being able to distinguish between the two cases can make a big difference. Therefore, multi-scale compositionality is another important property to consider when studying social dynamics.

2.5 The Three Main Tasks of This Thesis

By considering heterogeneity and multi-scale compositionality of social dynamics, we aim at addressing three basic tasks in this thesis: pattern mining, structure decomposition, and data-driven dynamics engineering.

2.5.1 Pattern Mining

Given a collection of N social dynamics $\{X^{(i)}\}_{i=1}^{N}$, *pattern mining*, aims at finding a collection of K *representative patterns* $\{X_k^*\}_{k=1}^{K}$, such that $X_k^* \in R^{D\times T}$ and $K \ll N$. This is essentially a clustering problem, in which the main challenges include defining a good distance measure and developing a good clustering algorithm.

2.5.2 Structure Decomposition

Given, again, a collection of N social dynamics $\{X^{(i)}\}_{i=1}^{N}$, *structure decomposition*, aims at finding a collection of *underlying structures* $\{W_{l,k}\}$, $l \in \{1,\cdots,L\}$, $k \in \{1,\cdots,K\}$, such that l indices different temporal scales (e.g., seconds, minutes, hours, etc.) and k indices different structures of a temporal scale. Both L and K are given, and typically, we have $L < K \ll N$. This can be regarded as a factorization problem, in which the main challenges include how to consider multiple temporal scales and how to solve the problem efficiently with a quality guarantee.

2.5.3 Data-driven dynamics engineering

The goal of *data-driven dynamics engineering* is rather different from the previous two tasks. We assume we are given some model that encapsulates the underlying patterns or structures of a collection of historical social dynamics. Also, we are given some observed dynamics $X \in R^{D \times T_x}$. The task is to find the best short-term *intervention dynamics* $U \in R^{D \times T_u}$ that leads to the best long-term *outcome dynamics* $V \in R^{D \times T_v}$ according to some pre-defined criteria (see Figure 7.1 for an illustration). This is essentially an inference / optimization problem, in which the main challenges consist of both the formulation and the solution.

2.6 Dataset Description

Twitter: We use the Twitter dataset from [78], which consists of 181M postings collected between June to December of 2009 from 40.1M users and 1.4B following relationships. To enumerate the information tokens that carry the social dynamics (as defined in Section 3.1), and in contrast to previous work based on hashtags [78] [29], we find that the discussion of many interesting events does not always include a hashtag. Therefore, we adopt a more general definition using *bursty keywords*, i.e., keywords that attract intense attention over short periods of time. We remove common terms (e.g., "the", "and", etc.) and apply the classic method in [26] to detect bursty keywords. Consequently, we end up with a 0.5M-sample dataset of Twitter social dynamics.

We characterize the Twitter social dynamics using seven features based on the type of users involved and certain graph statistics [52]. For features based on the types of users involved, we consider five types of users, namely: *Initiators* denote the users who use a particular keyword before any of their friends did. *First-time propagators* and *first-time commentators* denote the users who retweet and tweet, respectively, about this keyword after their friends did use the same keyword before. *Recurring propagators* and *recurring commentators* denote the users who retweet and tweet, respectively, the same keyword that himself or herself used it before. For graph statistics, we build the evolving graph corresponding to the uses of each keyword and calculate the graph *diameter* and the size of the *largest connected component LCC.*

Yelp: We also use the Yelp dataset from [79] that consists of 1.1M reviews made by 252K users (with 956K friendship edges among them) during the ten-year period from 2004 through 2014. The target of these reviews are 42K businesses in Las Vegas, Phoenix, Edinburgh, Madison, and Waterloo; each of these businesses is considered as an information token. For better representativeness, we select the businesses with at least 40 reviews (i.e. one review per season, on average), yielding a 5.3K dataset of social dynamics. We characterize the Yelp social dynamics using six evolving statistics of a business, namely, its numbers of reviews and tips, its average relative rating, the experience (measured by the number of previous reviews) and influence (measured by the number of friends) of the business's reviewers, and the amount of user responses (that tag each review as useful, funny, or cool).

Human Activity: To demonstrate the generality of our proposed methods beyond the social dynamics, we further use the Carnegie Mellon University Multi-Modal Activity Database (CMU-MMAC) [70]. This dataset contains the sensor readings from multiple accelerometers and gyroscopes when 46 participants were asked to cook five different recipes. These sensors are located all over the human body, including both forearms and upper arms, left and right calves and thighs, abdomen, and both wrists. For good interpretation, auxiliary modalities are included such as video, audio, and user annotations. Further, the dataset also includes abnormal cases such as phone calls, fire alarms, and armed robberies.

Macroeconomics: To further demonstrate the generality of our proposed methods, we use another dataset that is the macroeconomics dataset obtained from the World Bank [76]. We consider six macroeconomics indices, namely: inflation rate, total stock traded, exchange rate, lending interest rate, total goods traded, and domestic credit to private sector, and GDP growth. Further, we select the countries with all these indices for the past 22 years. This gives us a set of twelve countries, namely: Australia, Bangladesh, Colombia, India, Israel, Japan, Korea, Malaysia, Philippines, Singapore, Thailand, and USA.

2.7 Thesis Structure

In Chapter 3, we describe the prior work related to this thesis. Then we dedicate Chapter 4 to task T1, i.e., pattern mining, and Chapter 5 to task T2, i.e., structure decomposition. A few other applications are explored in Chapter 6. Then we describe task T3, i.e., data-driven dynamics engineering, in Chapter 7. Finally, we discuss future directions in Chapter 8 and draw a conclusion in Chapter 9.

Chapter 3

Related Work

There are six lines of research related to the topics presented here: (1) static and dynamic properties of social media, (2) modeling of dynamic social interactions, (3) pattern discovery of social dynamics, (4) time series modeling, (5) deep learning, and (6) human activity recognition.

3.1 Static and Dynamic Properties of Social Media

Several researches have studied the static properties of social media, such as short average path length [1], high clustering coefficient [47], and power-law degree distribution [4]. By the same token, many other researches have studied the dynamic properties of social media. In particular, the differences in the persistence of hashtags are studied in [2, 61]; endogenous vs. exogenous trends are studied in [44, 13]; the shape of the adoption dynamics are looked at in [78]; the relative proportions of adoption before, at, or after the peak are investigated in [29]. All this prior work confirms the rich and multi-faceted nature of of social-media properties, which motivates the present research for systematic understanding and engineering.

3.2 Social Dynamics Modeling

Another line of research is devoted to the modeling of social dynamics. Some of these models are generative in nature [26, 9, 3, 37], which define a probability distribution of social dynamics. There are also predictive models [71, 67, 57, 77, 18, 36, 38, 39], where a probability distribution can be indirectly defined, e.g., by introducing Gaussian noise. For a review, see [63]. The merit of these models is that they reveal insights into the underlying interactions and mechanisms behind the social dynamics. Our work herein is different as we are not trying to find a unifying model, but rather the systematic identification of the patterns and structures that exist in social media. Also, we explore how to steer the social dynamics purposefully given a set of identified patterns and structures.

3.3 Pattern Discovery in Social Dynamics

There are a few papers that have investigated methods of identifying patterns for social dynamics [44, 78, 29, 52]. In particular, endogenous vs. exogenous trends are studied in [44]; the shape of aggregate popularity is looked at in [78]; the proportions of readership before, at, and after the peak are investigated in [29]. Our work complements all these works by additionally considering the properties P1 (i.e., heterogeneity) and P2 (i.e., multi-scale compositionality) mentioned in Chapter 1, which are critically important for studying social dynamics. Indeed, we have demonstrated in [51] that not considering heterogeneity (P1) can mix up different types of social interactions that have radically different consequences and possible courses of action. Also, we have demonstrated in [52] that different signatures that characterize the social dynamics can manifest at multiple time scales and generate different meanings

(P2).

3.4 Time-Series Clustering and Dictionary Learning

The research in time-series clustering is conceptually related to our task T1 (i.e., pattern mining) in Chapter 1, which generally covers combinations of distance measures and clustering algorithms. For example, a classic combination is the Euclidean distance and the K-Means (KM) clustering [73]; the advantage of this combination is the availability of efficient symbolic approximation [32]. Dynamic Time Warping (DTW) [25] is another popular distance measure with the strength of finding the optimal alignment between two series. However, the clustering algorithms that work for DTW typically have very limited scalability, e.g, the Hierarchical Clustering (HC) [41]. The Scale-Invariant (SI) distance [11] received less attention compared to the two distances above. However, the SI distance has been recently shown to characterize dynamics of online activities very well [78], much better than Euclidean distance with proper normalization. Also, since the alignment of two series of online activities can be typically done by making them peak at the same time [78], DTW has little advantage, too. For the SI distance, the K-Spectral Centroid (KSC) clustering is developed by [78]. However, not only that KSC does not scale well for long times series, it is also difficult to derive other scalable clustering algorithms for the SI distance because it is *not* a metric. Finally, none of these methods provides a guaranteed solution quality. Particularly, since KM and KSC are both special cases of Expectation Maximization [41], they are typically *not* able to find the global optimal solution. As we show later, our proposed solution for task T1 can characterize online activities well, while exhibiting a high scalability and providing a guaranteed solution quality.

The research in time-series dictionary learning is conceptually related to our task T2, which targets the general problem of mining structures from time-series streams [62, 8, 58, 43, 59]. Along this line, the authors of [62, 8] make use of a fixed-length sliding window to extract the subsequences followed by conventional clustering methods. Also, in [58], a method based on *minimum description length* is proposed to consider variable scales. Also, the authors of [43, 59] introduce the concept of *shapelets* that does not rely on specific scales. Moreover, latent factor methods like [22, 49] model multivariate time series using hidden variables. Along this line, the State Space Model [22] builds a linear dynamical system assuming time-invariance and linearity. Also, Sparse Coding [49] can be used to discover global signatures for pre-aligned time series. Finally, the authors of [17, 82] combine a convolutional formation with sparse coding. While this prior work relates somehow to our problem, most of these approaches are not designed for social dynamics; moreover, none of them simultaneously considers the properties P1 and P2 mentioned in Chapter 1.

3.5 Pattern Mining Using Deep learning

Finally, the research in deep learning is conceptually related to our solutions for T2 and T3. Research in deep learning has recently gained much attention in supervised learning such as classification and regression [64, 12], as well as unsupervised learning such as feature extraction [69] and dimension reduction [19]. Moreover, there are also works that formulate deep learning using convolution [28, 24]. In particular, the authors of [28] propose the Convolutional Deep Belief Network (cDBN), which combines the Reduced Boltzmann Machine with the matrix convolution. Its sampling-based learning algorithm, however, is not efficient enough for practical use.

The Convolutional Autoencoder (cAE) proposed in [24] represents the current state-of-the-art among convolutional deep architectures. When applied to image recognition, it not only produces meaningful features that mimic the ones used by human's visual cortex area V2, but it also generates classification results that outperform the state-of-the-art.

When applied to social dynamics, our proposed methods for T2 and T3 have two advantages over cAE. First, cAE uses the conventional convolution operator that overlooks the heterogeneity inherent in social dynamics. Our proposed methods, in contrast, use specialized convolution operators that exploit the heterogeneity of social dynamics, and therefore offer higher-quality solutions, yet require much less runtime. Second, the higher-level layers of cAE have much more hidden variables compared to its lowest-level layer. In our proposed methods, on the contrary, the number of hidden variables of each layer remains roughly the same, which further enhances efficiency.

3.6 Human Activity Recognition

The ubiquitous availability of mobile devices with sensing capabilities has created many emerging applications, such as mobile advertisement, healthcare, personal lifelogging, etc [35]. For many of these applications, the challenge lies in how to understand and anticipate human activities from multiple heterogeneous sensors. [56, 20, 80, 21, 27, 40]. For example, authors of [56, 20] have proposed unsupervised algorithms to extract useful features from heterogeneous sensors. Also, authors of [27, 40] have proposed supervised methods to recognize basic activities. Moreover, authors of [80, 21] have focused on methods that detect abnormal activities. The main limitation of the previous work, however, is the absence of an unified framework that can achieve all these important tasks. In this thesis, we use what we propose for social dynamics and apply it directly to build a unified framework for human activity recognition.

3.7 Research Gap

In summary, although our work is conceptually related to the above five lines of prior research, our contribution is unique and novel in two ways. First, in terms of tasks T1 and T2 for social dynamics, we are the first to consider properties P1 and P2. As will be shown in Chapters 4 and 5, considering these two properties are critically important for studying social dynamics. Second, we are also the first to address the task T3, i.e., the *dynamics engineering* problem. As will be shown in Chapter 7, this can enable many new and exciting applications.

Chapter 4

Pattern Mining Using Angle-Shift

In this chapter, we present a novel framework called *Angle-Shift* (*ASH*) that enables the systematic and scalable identification of the representative patterns of social dynamics. ASH is based on a general data representation, involves a flexible distance measure that is a metric, exhibits a high scalability, and has a guaranteed solution quality. Experimental results show that the proposed method is significantly faster and achieves more accurate results compared to the state-of-the-art approaches. Consequently, by applying the proposed method to a large Twitter dataset, we identify new patterns of social dynamics, such as propagation of breaking news, advertisement, social mobilization, and interest group formation. Further, we investigate new applications enabled by the results produced by ASH, including decomposition of the intertwining collective-behaviors during a major real-world event, and clustering-augmented forecasting of social dynamics.

4.1 Dynamics Representation

Given social dynamics $X \in R^{D \times T}$, as defined at the end of Section 2.1, we begin by extracting useful features. In particular we consider two general types of features: time-series and scalar data.

The time-series features are meant to capture the *shape* of the evolution of important statistics. For example the five time series in Figure 2.2(b) represent the evolution of the numbers of different types of contributors and their graphical statistics. Without loss of generality (w.l.o.g.), we use the *shape vector* $\mathbf{x}_s$ to denote the concatenation of all the D time-series features $\mathbf{x}_s = (\mathbf{x}_{(s,1)}, \ldots, \mathbf{x}_{(s,D)})$, where $||\mathbf{x}_s|| = 1$, $||\mathbf{x}_{(s,d)}|| = 1/\sqrt{D}$ for all $d \in \{1, \ldots, D\}$, and $|| \cdot ||$ denotes the standard 2-norm. All notations used in this chapter are summarized in Table 4.1.

The scalar features, on the other hand, are meant to capture the *magnitude* of important statistics, e.g., the proportions of initiators and commentators. W.l.o.g, we use a *magnitude vector* $\mathbf{x}_m \in [0, 1]^{d_m}$ consisting of d_m scalar features. Considering both types of features, we use

$$\mathbf{x} = (\mathbf{x}_s, \mathbf{x}_m) \tag{4.1}$$

as the main representation of social dynamics in the chapter. While the particular choices of features depend heavily on the specific application, these two kinds of features cover all features used in previous works. For example, time-series features were employed by authors of [61, 78, 34, 14, 51], whereas scalar features were used by authors of [29, 44, 51, 34, 14].

Representation and Distance	
$\mathbf{x}$	vector of an collective behavior. $\mathbf{x} = (\mathbf{x}_s, \mathbf{x}_m)$
$\mathbf{x}_s$	time-series (shape) features of $\mathbf{x}$. $\mathbf{x}_s = (\mathbf{x}_{(s,d)})_{d=1}^{D}$
$\mathbf{x}_{(s,d)}$	the d-th time-series feature of $\mathbf{x}$
$\mathbf{x}_m$	scalar (magnitude) features of $\mathbf{x}$
d, d_s, d_m	dimensionality of $\mathbf{x}$, $\mathbf{x}_s$, and $\mathbf{x}_m$, respectively
$g^2(\mathbf{x}, \mathbf{y})$	distance between two collective behaviors $\mathbf{x}$ and $\mathbf{y}$
Clustering Algorithm	
$\{\mathbf{x}^{(i)}\}_{i=1}^{n}$	set of n sample collective behaviors
n	size of sample collective behaviors
$\mathcal{M}$	manifold (space of all possible collective behaviors)
$f(\mathbf{x})$	true p.d.f. where $\{\mathbf{x}^{(i)}\}_{i=1}^{n}$ is drawn from
$f_n(\mathbf{x})$	kernel density estimator of f
h	bandwidth parameter of $f_n(\mathbf{x})$
$\mathbf{x}^*$	true mode (i.e., local maximum) of $f(\mathbf{x})$
μ	estimated mode using $f_n(\mathbf{x})$ (i.e., identified pattern)
Convergence Guarantee	
$A(f, \delta, \mathbf{x})$	the δ-level set around $\mathbf{x}$ w.r.t. $f(\mathbf{x})$
diam A	diameter of set A
l^*	gap index of f
β	Hölder continuity of f
α	peak index of f

Table 4.1: Summary of all notations used in this chapter.

4.2 Distance Measure

Under the representation of Equation 4.1, our proposed distance measure is:

$$g^2(\mathbf{x}, \mathbf{y}) = \theta^2(\mathbf{x}_s, \mathbf{y}_s) + \gamma ||\mathbf{x}_m - \mathbf{y}_m||^2, \tag{4.2}$$

where $\theta(\mathbf{x}_s, \mathbf{y}_s)$ denotes the angle (in radians) between $\mathbf{x}_s$ and $\mathbf{y}_s$ and $\gamma = \frac{\pi^2}{d_m}$ is a normalizer. Note that this distance measure corresponds directly to the two types of features in Equation 4.1: The first component $\theta(\cdot)$ measures discrepancy about shape for time-series features, whereas the second component $|| \cdot ||$ measures discrepancy about magnitude for scalar features.

The design of the two components is geometrically appealing. While it is straightforward that the distance between $\mathbf{x}_m$ and $\mathbf{y}_m$ in magnitude can be measured by $||\mathbf{x}_m - \mathbf{y}_m||$, it is also intuitive that the distance between $\mathbf{x}$ and $\mathbf{y}$ in shape can be measured by $\theta(\mathbf{x}_s, \mathbf{y}_s)$. For example, if $\mathbf{x}_s$ and $\mathbf{y}_s$ are similar in shape but different in scale[1], the angle $\theta(\mathbf{x}_s, \mathbf{y}_s)$ between them will be close to zero. On the other hand, if $\mathbf{x}_s$ and $\mathbf{y}_s$ have nearly opposite shapes, the angle between them will be close to π. In other words, the distance in shape $\theta(\mathbf{x}_s,\mathbf{y}_s)$ can be obtained by mapping the two vectors onto a sphere and calculating the *geodesic path* between them, as illustrated by the dotted path between $\mathbf{x}$ and $\mathbf{y}$ in Figure 4.1.

Besides being geometrically intuitive, the proposed distance measure is also analytically convenient because it is a *metric*. Since it is straightforward to check $g^2(\cdot)$ for its non-negativity, symmetry, and identity of indiscernibles ($g^2(\mathbf{x}, \mathbf{y}) = 0 \iff \mathbf{x} = \mathbf{y}$), we derive its triangular inequality as follows. First, it can be observed that $\theta(\cdot)$ satisfies the triangular inequality $\theta(\mathbf{x}_s, \mathbf{z}_s) \leq \theta(\mathbf{x}_s, \mathbf{y}_s) + \theta(\mathbf{y}_s, \mathbf{z}_s)$, since $\theta(\cdot)$ is the shortest

[1]In this chapter, we assume that the all time series are pre-aligned to peak at the same time. As mentioned in Section 3.4, this is a common assumption that is often met for online activities.

geodesic path between two points lying on a sphere. Also, since $||\cdot||$ is a metric, the triangular inequality is obtained by:

$$\begin{aligned}
&(g(\mathbf{x},\mathbf{y}) + g(\mathbf{y},\mathbf{z}))^2 \\
&\geq \theta^2(\mathbf{x}_s,\mathbf{y}_s) + \theta^2(\mathbf{y}_s,\mathbf{z}_s) + \gamma(||\mathbf{x}_m - \mathbf{y}_m||^2 + ||\mathbf{y}_m - \mathbf{z}_m||^2) \\
&\geq \theta^2(\mathbf{x}_s,\mathbf{z}_s) + \gamma||\mathbf{x}_m - \mathbf{z}_m||^2 \\
&= g^2(\mathbf{x},\mathbf{z}).
\end{aligned} \tag{4.3}$$

4.3 Comparison Among Distances

Since the SI distance proposed in [78] also captures distances in shape, in principle, the first component of Equation 4.2 may be substituted by the SI distance. However, we show that the our angular distance $\theta(\cdot)$ is a better choice, both intuitively and analytically.

To start with, note that for two vectors that are already aligned, their SI distance can be written as:

$$\begin{aligned}
g_{SI}(\mathbf{u},\mathbf{v}) &= \min_{\alpha} \tfrac{||\mathbf{u}-\alpha\mathbf{v}||}{||\mathbf{u}||} \\
&= ||\dot{\mathbf{u}} - \dot{\mathbf{u}}^T\dot{\mathbf{v}}\,\dot{\mathbf{v}}|| \\
&= \sin\theta(\mathbf{u},\mathbf{v}),
\end{aligned} \tag{4.4}$$

where the second line is obtained by solving the optimal value of $\alpha^* = \frac{\mathbf{u}^T\mathbf{v}}{||\mathbf{v}||^2}$. Note that $\dot{\mathbf{u}}$ denotes $\frac{\mathbf{u}}{||\mathbf{u}||}$, and likewise for $\dot{\mathbf{v}}$. Geometrically, Equation 4.4 means that comparing the angular and the SI distances is equivalent to comparing $\theta(\mathbf{u},\mathbf{v})$ with $\sin\theta(\mathbf{u},\mathbf{v})$. Particularly, when $\theta \in [\frac{1}{2}\pi,\pi]$ and as $\theta(\mathbf{u},\mathbf{v})$ becomes larger (i.e., the shape of $\mathbf{u}$ and that of $\mathbf{v}$ becomes more *different*), $\sin\theta$, on the contrary, becomes *smaller*. Moreover, when $\mathbf{x}_s$ and $\mathbf{y}_s$ have nearly opposite shapes (i.e., $\theta(\mathbf{u},\mathbf{v}) \approx \pi$), their SI distance $\sin(\mathbf{u},\mathbf{v})$ will be close to *zero*.

While this is clearly counter-intuitive, this also shows that the SI distance is *not* a metric. More specifically, when $\theta(\mathbf{u},\mathbf{v}) = \pi$, we have:

$$g_{SI}(\mathbf{u},\mathbf{v}) = \sin\theta(\mathbf{u},\mathbf{v}) = 0 \text{ where } \mathbf{u} \neq \mathbf{v}, \tag{4.5}$$

which violates identity of indiscernibles. Note that being not a metric is a major disadvantage for a distance measure, since it prohibits the use of many mathematical tools to design scalable clustering algorithms for it.

4.4 Clustering Algorithm

4.4.1 ASH Clustering

We now present our Angle-SHift (ASH) clustering algorithm that identifies patterns of social dynamics with high scalability and guaranteed solution quality. We design ASH by generalizing the Mean-Shift clustering algorithm [41] to the Riemannian manifold [50] that is defined by the representation and distance metric that we propose. Under our representation (Equation 4.1), the space of any possible social dynamics $\mathbf{x}$ is given by:

$$\mathcal{M} = S^{d_s} \times [0,1]^{d_m} \tag{4.6}$$

where $S^{d_s} = \{||\mathbf{y}|| = 1\}$ is a d_s-dimensional sphere. Because our distance measure g (Equation 4.2) is a metric, the set $\mathcal{M}$ together with the distance g form a Riemannian manifold. All geometrical concepts in this section are illustrated in Figure 4.1, where a collection of seven sample social dynamics $\{\mathbf{x}^{(i)}\}_{i=1}^{7}$ lies in $\mathcal{M}$. A probability density function (p.d.f.) $f(\mathbf{x})$ can be defined over $\mathcal{M}$: In Figure 4.1, the corresponding $f(\mathbf{x})$ will have two bumps around the two clusters, $\{\mathbf{x}^{(i)}\}_{i=1}^{4}$ and $\{\mathbf{x}^{(i)}\}_{i=5}^{7}$.

Given a collection of social dynamics $\{\mathbf{x}^{(i)}\}_{i=1}^{n} \subseteq \mathcal{M}$ that are drawn from some p.d.f.

$f(\mathbf{x})$ defined over $\mathcal{M}$, the clustering problem is to find all local optimal solutions $\{\mathbf{x}^*\}$ (i.e., the *modes* of $f(\mathbf{x})$) of the problem:

$$\begin{aligned} \max_{\mathbf{x}} \quad & f(\mathbf{x}) \\ s.t. \quad & \mathbf{x} \in \mathcal{M}. \end{aligned} \tag{4.7}$$

Using the example of Figure 4.1, there will be two modes, denoted by $\star$, that lie around the centers of $\{\mathbf{x}^{(i)}\}_{i=1}^{4}$ and $\{\mathbf{x}^{(i)}\}_{i=5}^{7}$, respectively. Geometrically, it is intuitive that the modes $\{\mathbf{x}^*\}$ are representative for all sample events $\{\mathbf{x}^{(i)}\}_{i=1}^{n}$.

Since $f(x)$ is unknown, we maximize its estimator instead:

$$f_n(\mathbf{x}) = \frac{1}{nh^d}\sum_{i=1}^{n} K(\frac{g(\mathbf{x},\mathbf{x}^{(i)})}{h}) \propto \sum_{i=1}^{n} exp\left\{\frac{g^2(\mathbf{x},\mathbf{x}^{(i)})}{-2h^2}\right\}, \tag{4.8}$$

where h is the bandwidth parameter controlling how spiky the estimated density is. To maximize $f_n(\mathbf{x})$, we first need its derivative $\nabla f_n(\mathbf{x})$. However, because $f_n(\mathbf{x})$ is defined over $\mathcal{M}$, $\nabla f_n(\mathbf{x})$ always lies in the space $\mathcal{T}_{\mathbf{x}}$ that is tangent to $\mathcal{M}$ at $\mathbf{x}$. Since $\mathbf{x}+\nabla f_n(\mathbf{x})$ does not lie in $\mathcal{M}$, we need the *exponential operator* $Exp_{\mathbf{x}}(\cdot)$ to map it back to $\mathcal{M}$, such that the *geodesic* (the dotted) path between $\mathbf{x}$ and $\mathbf{y} = Exp_{\mathbf{x}}(\nabla f_n(\mathbf{x}))$ is of the same length as $||\nabla f_n(\mathbf{x})||$. The reverse mapping is the *log operator* $Log_{\mathbf{x}}(\cdot)$ that satisfies $\nabla f_n(\mathbf{x}) = Log_{\mathbf{x}}(\mathbf{y})$ and $\nabla_{\mathbf{x}} g^2(\mathbf{x},\mathbf{y}) = -2Log_{\mathbf{x}}(\mathbf{y})$.

Based on our representation and distance measure, we derive that:

$$\begin{aligned} Exp_{\mathbf{x}}(\boldsymbol{\Delta}) &= \left(\cos||\boldsymbol{\Delta}_s||\,\mathbf{x}_s + \tfrac{\sin||\boldsymbol{\Delta}_s||}{||\boldsymbol{\Delta}_s||}\boldsymbol{\Delta}_s\ ,\ \mathbf{x}_m + \boldsymbol{\Delta}_m\right) \\ Log_{\mathbf{x}}(\mathbf{y}) &= \left(\theta(\mathbf{x}_s,\mathbf{y}_s)\tfrac{(\mathbf{y}_s-\mathbf{x}_s^T\mathbf{y}_s\,\mathbf{x}_s)}{||(\mathbf{y}_s-\mathbf{x}_s^T\mathbf{y}_s\,\mathbf{x}_s)||}\ ,\ \mathbf{y}_m - \mathbf{x}_m\right). \end{aligned} \tag{4.9}$$

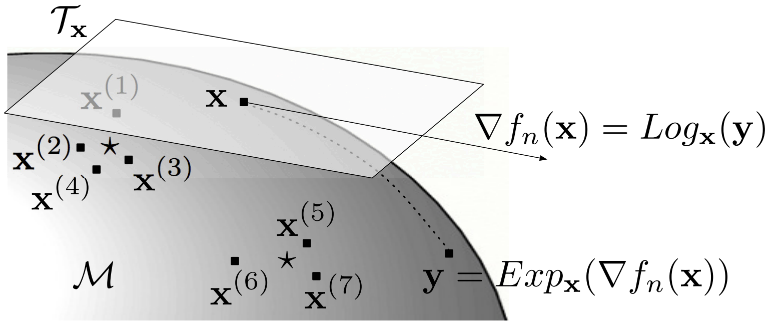


Figure 4.1: Geometric illustration of a set of social dynamics $\{\mathbf{x}^i\}_{i=1}^7$, the manifold $\mathcal{M}$, the tangent space $\mathcal{T}_{\mathbf{x}}$, the exponential operator $Exp_{\mathbf{x}}(\cdot)$, and the log operator $Log_{\mathbf{x}}(\cdot)$. Here the underlying p.d.f. $f(\mathbf{x})$ has two bumps around $\{\mathbf{x}^i\}_{i=1}^4$ and $\{\mathbf{x}^i\}_{i=5}^7$, corresponding to the two modes $\{\mathbf{x}^*\}$ denoted as $\star$'s.

Algorithm 1 ASH Clustering

```
Input: $\{\mathbf{x}^{(i)}\}_{i=1}^n$: collection of social dynamics
Input: $h$: bandwidth parameter
Output: $\{\mu^{(j)}\}$: set of distinct local optima
for $i \leftarrow 1$ to $n$ do
    $\mathbf{x} = \mathbf{x}^{(i)}$
    repeat
        $\mathbf{v} = \nabla_{\mathbf{x}} f_n(\mathbf{x})$
        $t$ = find_stepsize($\mathbf{x}, \mathbf{v}$)
        $\mathbf{x} = Exp_{\mathbf{x}}(t\mathbf{v})$
    until $\mathbf{x}$ converges;
    $\mu^{(i)} = \mathbf{x}$
end
return distinct local optima $\{\mu^{(j)}\}$
```

Accordingly, the gradient (∇) can be derived as:

$$\begin{aligned}
\nabla_{\mathbf{x}} f_n(\mathbf{x}) &= \textstyle\sum_{i=1}^n \frac{1}{-2h^2} e^{\frac{g^2(\mathbf{x},\mathbf{x}^{(i)})}{-2h^2}} \cdot \nabla_{\mathbf{x}} g^2(\mathbf{x},\mathbf{x}^{(i)}) \\
&= \textstyle\sum_{i=1}^n \frac{1}{h^2} e^{\frac{g^2(\mathbf{x},\mathbf{x}^{(i)})}{-2h^2}} \cdot Log_{\mathbf{x}}(\mathbf{x}^{(i)}) \\
&= \frac{1}{h^2} \textstyle\sum_{i=1}^n e^{\frac{g^2(\mathbf{x},\mathbf{x}^{(i)})}{-2h^2}} \cdot \\
&\qquad \left(\theta(\mathbf{x}_s, \mathbf{x}_s^{(i)}) \frac{\mathbf{x}_s^{(i)} - \mathbf{x}_s^T \mathbf{x}_s^{(i)} \mathbf{x}_s^{(i)}}{\|\mathbf{x}_s^{(i)} - \mathbf{x}_s^T \mathbf{x}_s^{(i)} \mathbf{x}_s^{(i)}\|} \ , \ \mathbf{x}_m^{(i)} - \mathbf{x}_m\right).
\end{aligned} \tag{4.10}$$

The ASH clustering algorithm is given in Algorithm 1. It takes as input a collection of social dynamics $\{\mathbf{x}^{(i)}\}_{i=1}^{n}$ and produces as output the set of distinct local optima $\{\mu^{(j)}\}$ that are representative social-dynamics patterns. In each iteration of the for-loop, the algorithm simply approximates Equation 4.7 using the i-th sample $\mathbf{x}^{(i)}$ as the initial condition. In the repeat-loop, the optimization (Equation 4.7) is solved iteratively in three steps: The derivative is taken (Equation 4.10), the stepsize is determined, and then the exponential mapping is evaluated (Equation 4.9). Technical steps of the `find_stepsize` function are detailed in Appendix A. At the end, samples that come from the same cluster will lead to the same local optimum and consequently be grouped together.

4.4.2 Performance Enhancement

We now introduce two techniques to improve the runtime complexity. For the for-loop, it is unnecessary to include all n samples as starting points. Since we are eventually interested in finding all distinct modes ($\mu^{(j)}$'s), only one sample per each cluster is really needed (e.g., in Figure 4.1, one from $\{\mathbf{x}^{i}\}_{i=1}^{4}$ and another from $\{\mathbf{x}^{i}\}_{i=5}^{7}$). Although the clustering information is unavailable beforehand, we can use a sampling technique. For example, under uniformly random sampling, suppose the smallest cluster occupies a portion r of all sample events. Then it can be shown that under a confidence level of p, the number of samples needed (n_0) to ensure that samples from all clusters are covered is:

$$n_0 = \left\lceil \frac{\log(1-p)}{\log(1-r)} \right\rceil. \tag{4.11}$$

For example, suppose n =1M, r = 0.1%, and p = 99.5%. In this case, only 0.5% (n_0 = 5296) of all samples are needed.

For the evaluation of $\nabla f_n(\mathbf{x})$, note that in Equations 4.10, the weight for a sample $\mathbf{x}^{(i)}$ drops exponentially fast when the distance $g^2(\mathbf{x}, \mathbf{x}^{(i)})$ increases. This means that all samples except for the close neighbors of $\mathbf{x}$ have little influence on $\nabla f_n(\mathbf{x})$. To dynamically retrieve the n_n closest neighbors of the current solution $\mathbf{x}$, a convenient data structure is the *kd-tree* [5]. Once the tree is built in $O(n \log n)$, to query n_n closest neighbors takes only $O(n_n + \log n)$.

Complexity: Incorporating both techniques, the runtime complexity becomes $O(n \log n + n_0(n_n d + \log n))$, or $O(n \log n + d)$ if n_0 and n_n are treated as constants. Of note, the for-loop in Algorithm 1 can be trivially decomposed into independent tasks and parallelized on a multiprocessor platform. Accordingly, the proposed method has excellent scalability.

4.5 Guarantee on Solution Quality

A significant challenge to evaluate any clustering algorithm is that it is usually difficult to obtain large-scale and high-quality ground truths to compare against. Therefore, it helps for a clustering algorithm to have formal guarantees. For example, the sample distribution $f(\mathbf{x})$ in Figure 4.1 has two bumps. But in general, $f(\mathbf{x})$ can be of arbitrary shape and an ideal clustering algorithm should still be guaranteed to perform well. In this section, we develop two theorems to ensure such a guarantee. Their proofs are given in Appendix A.

Definition 1. ***Local Maximum and Minimum.*** *Given $f(x)$, we call x^* a* local maximum *if:*

$$\exists \epsilon^* > 0 \text{ s.t. } f(x^*) \geq f(x) \;\; \forall x \in B_d(x^*, \epsilon^*) \cap \mathcal{M} \tag{4.12}$$

where $B_d(x^*, \epsilon^*)$ is a $d-$dimensional ball centered at x^* with radius ϵ^*. We denote the set of all local maxima as X^* and the set of all local minima *(substituting the "$\geq$" with "$\leq$" in Equation 4.12) as $\check{X}$. Moreover, the set of points obtained by substituting*

$f(x)$ with $f_n(\mathbf{x})$ in Equation 4.12 is denoted using $\hat{X}$.

Definition 2. ***Level set and Diameter.*** *Given a local maximum $\mathbf{x}^*$ and its corresponding ϵ^*, the* level set *$A(f, \delta, \mathbf{x}^*)$ is defined as:*

$$A(f, \delta, \mathbf{x}^*) = \{\mathbf{x} \in B_d(\mathbf{x}^*, \epsilon^*) \cap \mathcal{M},\ f(\mathbf{x}) > f(\mathbf{x}^*) - \delta\}. \tag{4.13}$$

Also, the diameter *of $A(f, \delta, \mathbf{x}^*)$ is defined as:*

$$\operatorname{diam} A(f, \delta, \mathbf{x}^*) = \sup_{\mathbf{x}, \mathbf{y} \in A(\mathbf{x}^*, \delta)} g(\mathbf{x}, \mathbf{y}). \tag{4.14}$$

Definition 3. ***Gap Index.*** *Given $f(\mathbf{x})$ and its set of local maxima (X^*) and minima ($\check{X}$), we define $l(\mathbf{x}^*)$ for each $\mathbf{x}^* \in X^*$ as:*

$$l(\mathbf{x}^*) = f(\mathbf{x}^*) - \operatorname*{arg\,min}_{\check{\mathbf{x}} \in \check{X}, f(\check{\mathbf{x}}) < f(\mathbf{x}^*)} g(\mathbf{x}^*, \check{\mathbf{x}})$$

where $l(\mathbf{x}^*)$ is defined as infinity if $\{\check{\mathbf{x}} \in \check{X}, f(\check{\mathbf{x}}) < f(\mathbf{x}^*)\} = \varnothing$. Then we define the gap index *l^* of f as:*

$$l^* = \operatorname*{arg\,min}_{\mathbf{x}^* \in X^*} l(\mathbf{x}^*) \tag{4.15}$$

Our guarantee is based on the following three assumptions:

(A1) $f(\mathbf{x})$ is Hölder-continuous of order β, that is:

$$\exists L, \beta \geq 0 \ \ s.t. \ \ |f(\mathbf{x}) - f(\mathbf{y})| \leq L\, g(\mathbf{x}, \mathbf{y})^{\beta} \ \ \forall \mathbf{x}, \mathbf{y} \in \mathcal{M}$$

(A2) $f(\mathbf{x})$ has a finite number of local maxima.

(A3) For each $\mathbf{x}^* \in X^*$, there exists an $\alpha^* > 0$ such that:

$$0 < \liminf_{\delta \to 0} \frac{\operatorname{diam} A(\mathbf{x}^*, \delta)}{\delta^{\alpha^*}} < \limsup_{\delta \to 0} \frac{\operatorname{diam} A(\mathbf{x}^*, \delta)}{\delta^{\alpha^*}} < \infty,$$

Intuitively, **(A3)** states that $f(\mathbf{x})$ is not infinitely sharp or flat at its local maxima, where α^* measures the sharpness of $f(\mathbf{x})$ at $\mathbf{x}^*$. For examples, $\alpha^* = 1/2$ when f is a Gaussian distribution centered at $\mathbf{x}^*$; similarly, $\alpha^* = 1$ when $f(\mathbf{x})$ is a Laplace distribution. We call the minimal value among all α^*'s (corresponding to all $\mathbf{x}^* \in X^*$) the *peak index* α of f.

Theorem 1. ***Convergence Rate.*** *Let $(\mathcal{M}, g)$ be a C^p Riemannian manifold* [45] *($p \geq 3$). Given a collection of samples $\{\mathbf{x}_i\}_{i=1}^n$ drawn from the p.d.f. $f(\mathbf{x})$ defined over $\mathbf{x} \in \mathcal{M}$ that satisfies* **(A1)** - **(A3)**. *Then under Definition 1 and with high probability (w.h.p.), we have that $\forall \hat{\mathbf{x}} \in \hat{X}$, there exists an $\mathbf{x}^* \in X^*$ that satisfies:*

$$\|\mathbf{x}^* - \hat{\mathbf{x}}\| \leq g(\mathbf{x}^*, \hat{\mathbf{x}}) = O(\sqrt{\log n}^{\alpha} n^{\frac{-\alpha\beta}{\beta + \frac{1}{2}m}}) \tag{4.16}$$

where $m = \frac{1}{2}n(3n + 11)$.

Theorem 2. ***Identifiability.*** *Incorporating Definition 3 and under the same setting as in Theorem 1, then there exists a constant $C > 0$ s.t. if $l^* > C\sqrt{\log n} n^{\frac{-\beta}{\beta + \frac{1}{2}m}}$, we have that $\forall \mathbf{x}^* \in X^*$, $\exists \hat{\mathbf{x}} \in \hat{X}$ that makes Equation 4.16 hold w.h.p.*

We make three final remarks about the above asymptotic guarantees. First, our two theorems are complementary: Theorem 1 ensures that all our solutions are close to the true modes, i.e., *all* solutions are of good quality. In contrast, Theorem 2 ensures that all true modes are covered by at least one solution, i.e., *all* true modes are identified. In other words, the set of our solutions converges asymptotically *one-to-one* to the set of the true modes. Second, note that although the true distribution

$f(\mathbf{x})$ of events $\mathbf{x}$ is unknown, the assumptions we use, namely, **(A1) - (A3)**, are very general and cover almost any mixture of known distributions. Finally, in both theorems, n needs to be large in order to provide a tight bound in Equation 4.16. This emphasizes again the importance of the scalability of the clustering algorithm.

4.6 Experiments

We conduct extensive experiments in the following four directions: (1) the relevance of distance measure, (2) the evaluation of ASH on scalability and solution quality using synthetic and real data, (3) social-dynamics patterns in Twitter, and (4) two new applications enabled by ASH.

4.6.1 Evaluation of Distance Measure using Ground Truth

We use the ground truth in [51], a carefully labeled hierarchical structure of social dynamics in Twitter, to evaluate various distance measures *per se* (i.e., while controlling the clustering algorithm). More specifically, we first conduct a complete-link hierarchical clustering [41] using each distance measure. Then, the similarity between each resulting hierarchical clustering and that of the ground-truth is measured by the *average* normalized mutual information ($\overline{MI}$) considering all possible number of clusters. Therefore, the distance measure with the largest $\overline{MI}$ provides the best match with the ground truth.

Both time-series and scalar features are used. For time series features, we use the seven time-series features shown in the legends of Figure 4.6. The first five time-series features consider different types of contributors. Besides the basic distinction among initiators, propagators, and commentators (as defined in Section 1), for non-initiators, we further distinguish whether a propagator (or commentator) contributes for the first time. The intuition is that the existence of recurring contributors shows that the information being disseminated is "sticky". As will be shown in Figure 7, this is the trait of certain specific types of social dynamics. For scalar features, we use the aggregate proportions of different types of contributors, as well as the peak values of the graph statistics in Figure 2.2.

Distance measures: We compare all the distance measures mentioned in Chapter 3.4. Moreover, to gain more insight, we extend the SI distance by substituting the first term of our proposed distance measure ($\theta(\cdot)$ in Equation 4.2) with the SI distance, and denote it using SI^*. For all these distance measures, both time-series and scalar features are used. We also tried to use only scalar features for the Euclidean distance and DTW, and only time-series features for the SI distance, but found the results very similar to what we present here.

Results: As presented in Figure 4.2, the results first confirm the findings in [78] that the SI distance characterizes online activities better than Euclidean and DTW distances do. Second, the fact that SI* and the proposed distances perform better than SI shows that carefully integrating scalar and time-series distances can characterize online social dynamics even better. Finally, although SI* and the proposed distances perform comparably in this small-scale experiment, we will soon show that the proposed distance measure performs much better in larger-scale settings.

4.6.2 Evaluation of Clustering Algorithm Using Synthetic Data

We then evaluate the proposed ASH clustering algorithm using synthetic data generated from Gaussian mixture. First, we sample $k = 10$ modes where each mode $\mathbf{x}^*$ is generated by sampling from $\mathbf{x}' \in [0.15\pi, 0.35\pi]^{d_s-1} \times [0.3, 0.7]^{d_m}$ uniformly at ran-

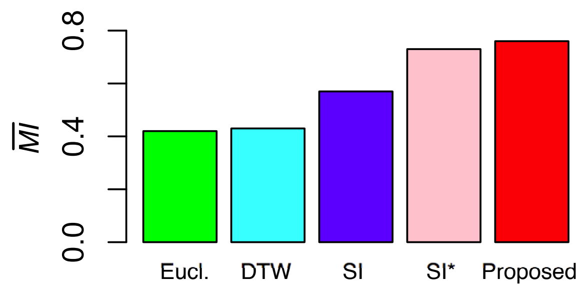


Figure 4.2: Evaluation of the state-of-the-art distance measures. `Eucl.`: Euclidean distance; `DTW`: dynamic time warping; `SI`: the distance given in [78]; `SI*`: the extension of [78] that incorporates multiple time series; `Proposed`: our proposed distance measure.

dom. Note that the first d_s elements are in Polar coordinates, where the mode $\mathbf{x}^*$ can be obtained by converting the first d_s elements of $\mathbf{x}'$ back to Cartesian coordinates. Next, we sample roughly an equal number of samples around each mode $\mathbf{x}^*$ by first drawing samples from:

$$\{\mathbf{x}^{(i)\prime}\} \sim N(\mathbf{x}', \Sigma) \tag{4.17}$$

and then converting them to Cartesian coordinates, where Σ is a diagonal matrix where the first $d_s - 1$ nonzero elements are set to 0.05π and the other nonzero elements are set to 0.1. Repeating this procedure for all k modes, we obtain our synthetic data.

Baseline: Our baseline method extends the KSC clustering to accommodate the SI* distance defined above; hence, it is called the *KSC* clustering algorithm*. Similar to KSC, KSC* has a runtime complexity of $O(nd_s^2 + kd_s^3 + kd_m n)$ due to the eigendecomposition required in the refinement step [78]. Also similar to KSC, the solution quality of KSC* is sensitive to initial conditions. Of note, we also experimented using the original KSC, which yields similar runtime but worse accuracy compared to the case of KSC*.

Scalability results: We first fix the dimensionality $d_s = 200$ and $d_m = 10$ and vary the sample size n from 1K to 3M; the results are presented in Figure 5.5(a).

We see that the runtime of ASH is faster than KSC* in all cases, and particularly so when n is large. To gain more insight about the scalability, we make the y-axis into a log-scale and re-plot the same figure in the inset. Under the log-scale, it can be observed that the slopes corresponding to ASH are smaller than that of KSC*, indicating that the speedup brought by ASH is greater than only a constant factor. For 3M events, it takes ASH 30 minutes to complete whereas it takes around 2 days for KSC*. This represents a speedup of two orders of magnitude even without the trivial parallelization mentioned at the end of Section 4.

Additionally, we fix the sample size at $n = 100K$ and vary d_s from 50 to 300 while keeping d_m fixed. The results are summarized in Figure 5.5(b) (and similarly, the y-axis is made into log-scale in the inset). Again, we see that not only ASH runs faster than KSC*, its slope in the inset is also smaller. It suggests that the advantage of ASH is even greater when the dimensionality is large. Indeed, at $d_s = 50$, it takes KSC* 152 seconds whereas it takes ASH 62 seconds (2.4X speedup); at $d_s = 300$, it takes KSC* 4.6 hours whereas it takes ASH 4.5 minutes (61X speedup). This shows that ASH is particularly useful in high-dimensional cases, which is important in analyzing social dynamics when their multifaceted properties need to be considered simultaneously.

Accuracy results: We then evaluate the capabilities of KSC* and ASH in recovering the true modes. For KSC*, we use $k = 10$. For ASH, we assign h such that the number of clusters equals to 10. In each case, the true and estimated modes are matched and the root mean-square error (RSME) is calculated. Again, we first fix the dimensionality $d_s = 200$ and $d_m = 10$ and vary the sample size n from 100 to 10K. The sample sizes are reduced because we now use 20 repetitions to evaluate the stability of the two algorithms' solutions.

As it can be seen from Figure 4.4(a), ASH consistently achieves significantly lower error and smaller variation compared to the KSC*. Also, for ASH, the error converges

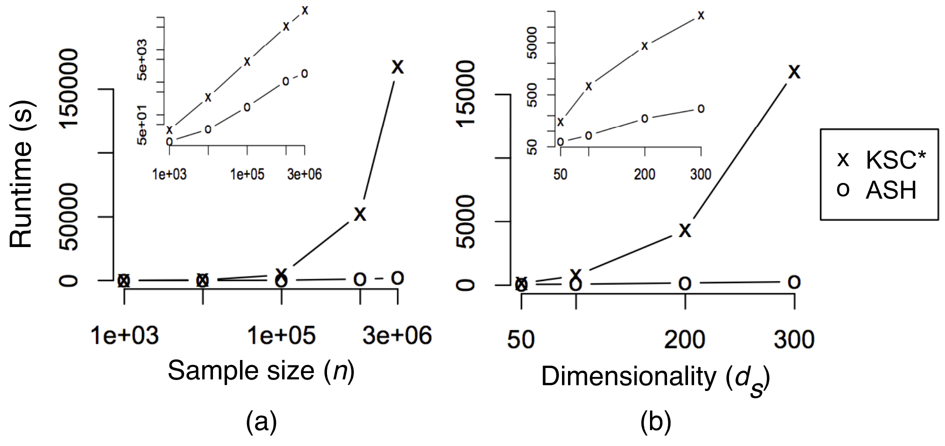


Figure 4.3: Scalability evaluation (via runtime) of KSC* vs. ASH, with varying (a) sample size and (b) dimensionality. Inset: y-axis plotted in log-scale.

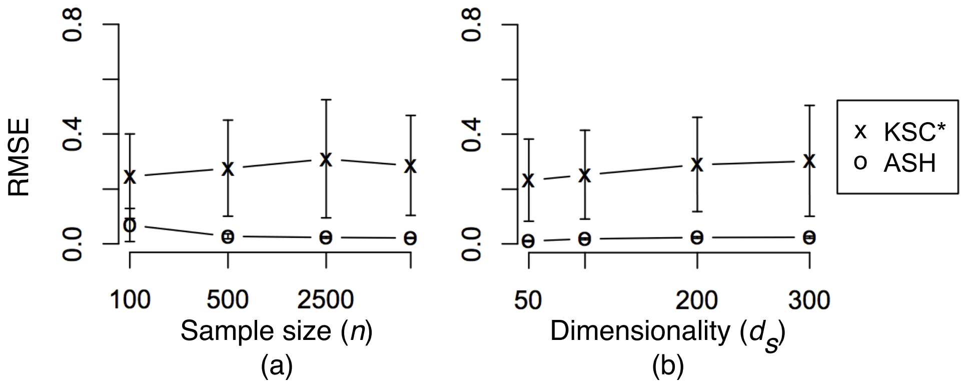


Figure 4.4: Accuracy evaluation (via RMSE) of KSC* vs. ASH, with varying (a) sample size and (b) dimensionality.

toward zero as n increases, which is not exactly the case for KSC*.

Additionally, in Figure 4.4(b), we fix the sample size at $n = 10K$ and vary d_s from 50 to 300 while keeping d_m fixed. We see that the error increases as the dimensionality goes up. Still, ASH consistently achieves both lower error and smaller variation. These results confirm the usefulness of the quality guarantees given in Section 5, and show that ASH is capable of reliably recovering the true modes with a moderate amount of data.

4.6.3 Evaluation of Clustering Algorithm using Twitter Data

We use the Twitter Dataset describe in Chapter 2.5 to evaluate the proposed algorithm. Since the number of clusters in this dataset is unknown, we determine the model parameters of ASH and KSC* as follows. For the bandwidth parameter h of ASH, we vary h from 0.01 to 300 and plot the resulting number of distinct modes k (namely, the number of clusters) on the left side of Figure 4.5. While k is large when h is small, it drops quickly and then stabilizes after $h \geq 10$, yielding $k \in \{7, 7, 6\}$. For the parameter k of KSC*, we calculate the the Average Silhouette [41] where a higher value corresponds to a better choice of k. From the results on the right side of Figure 4.5, it seems like $k \in \{6, 7, 8\}$ are all good choices. Therefore, we adopt $h = 10$ for ASH and $k = 7$ for KSC*.

Since the true modes for the dataset are unknown, we measure the clustering quality using:

$$F = \frac{\sum_k \sum_{i \in C_k} d^2(\mathbf{x}_i, \mu_k)}{\sum_{k,l} d^2(\mu_k, \mu_l)},$$

where $d^2(\cdot)$ denotes Equation 4.2 and SI* for ASH and KSC*, respectively. The numerator represents the sum of within-cluster distances and the denominator represents the sum of between-cluster distances. Since the objective of a clustering algorithm is to group similar samples in the same cluster and dissimilar samples in different clusters, a lower value of F indicates a better clustering solution.

We then conduct ten rounds of clusterings (for the 10% dataset) using both ASH and KSC*, and record their average and standard deviation in F. These results together with the average runtime are presented in Table 4.2. Again, ASH outperforms KSC* in terms of solution quality, both in the mean value and the standard deviation of F. Moreover, ASH runs significantly faster than KSC*. These results confirm again the superior scalability and solution quality of ASH.

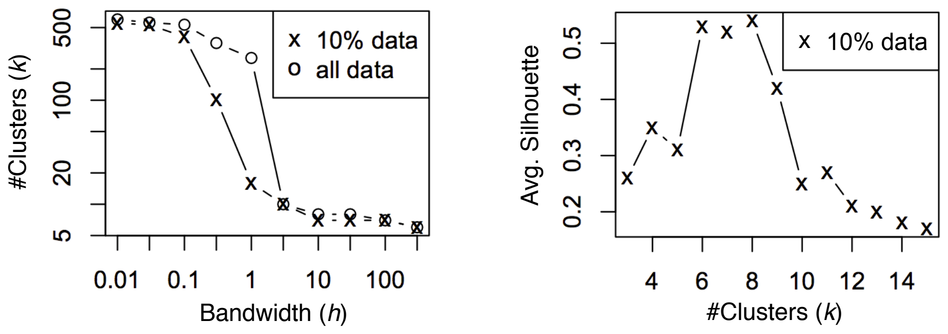


Figure 4.5: Parameter selection for ASH (left) and KSC* (right).

Method	μ_F	σ_F	Average runtime
KSC*	861.9	262.5	71 min.
ASH	**354.8**	**24.7**	**88 sec.**

Table 4.2: Evaluation on cluster quality and runtime using a 10% random subset (50K) of Twitter data.

4.6.4 Patterns of Social Dynamics in Twitter Data

Now we apply ASH to the full $0.5M$ Twitter dataset to identify patterns of social dynamics in Twitter. To determine the bandwidth parameter h, we repeat the parameter selection process in Section 6.3 and plot the result at the left side of Figure 4.5. Compared to the results of using the 10% dataset, although this time k remains high when $h \in \{0.3, 1\}$, it converges nearly identically to previous case after $h \geq 10$, yielding $k \in \{6, 7, 8\}$. This suggests that the social dynamics in Twitter can be reliably summarized by a relatively small number of patterns. We experimented with all cases for $h \in \{10, 30, 100, 300\}$ and found that the clusterings are quite stable. Accordingly, we adopt $h = 10$ and obtain $k = 8$ clusters, each representing a pattern of social dynamics.

Patterns: The identified patterns are summarized in Figure 4.6. The first three patterns (`P1 - P3`) are relatively similar; indeed, if we use $h = 300$ (in Figure 4.5) and obtain $k = 6$, then these three patterns will merge into one. The information

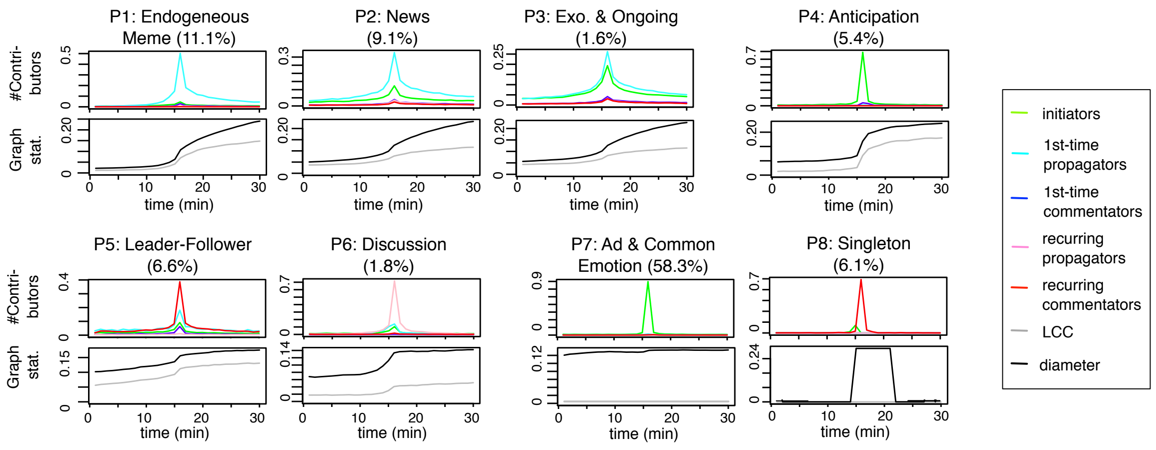


Figure 4.6: Patterns of social dynamics identified using our clustering algorithm: (a) choosing the bandwidth parameter; (b) the patterns identified. Significant differences among these patterns demonstrate the variety of social dynamics in Twitter.

carried by these social dynamics share a non-commentary trait characterized by the prominent light-blue line representing first-time propagators. Another common trait is the lack of recurring contributors (pink and red lines). For the graphical statistics, both the size of the largest connected component (LCC, represented by the black line) and diameter (represented by grey line) grow steadily. From the proportions annotated in Figure 4.6(b), `P1 - P3` represent a total of 21.8% of bursty keywords in our dataset.

Among `P1 - P3`, however, the important distinction is the amount of initiators (the green line), reflecting the relative strength of exogenous vs. endogenous propagation. The pattern `P1` has the smallest amount of initiators. By inspecting the corresponding keywords, we confirm that they are mostly associated with endogenous memes that come from *within* Twitter (e.g., *leak*, *iloveyou*, *fantasyfact*). Additionally, `P1` also include several successful viral marketing keywords (e.g., *xbox*, *rockband*). In contrast to `P2` that corresponds to conventional news (e.g., *madoff*, *fawcett*, *jackson*) with both endogenous and exogenous propagations, we found that P3 corresponds to the events that have broad exogenous sources and are still on-going (e.g., *mtv*, *gmailfail*, and *ireport*). As a consequence, more initiators introduce real-time updates from sources outside of Twitter.

Pattern P4 is associated with anticipation, carrying keywords such as *christmas*, *wwdc*, and *holiday*. It has a prominent peak of initiators followed by a minor amount of first-time commentators, represented by the minor bump of the blue line. These commentators are, however, important because they connect the group of participants, as shown by the increased diameter and LCC that differentiate P4 from P7.

Patterns P5 and P6 are both characterized by dominant recurring contributors, which are generally weaker in other patterns. In particular, P5 is associated with a leader-follower type of collective behavior, where opinion leaders contribute recurring commentaries while opinion followers contribute differently. From the diameter and LCC (gray and black lines), we observe that the community is relatively stable before and after the peak. Also, the diameter grows more comparably with LCC in contrast to the cases of P1 - P3. This shows that the participants are more loosely connected, which might be due to the fact that a common belief brings strangers together. Many keywords belonging to this pattern are related to some form of social mobilization, e.g., *iranelection*, *tehran*, and *brazilmissesjona*. P6, on the other hand, is associated with the interactions among a group of people having common hobbies or interests. It is characterized first by minor bumps of the green and light blue lines, and then a dominating peak of the pink line (recurring propagators). This characterizes a back-and-fourth discussion within a relatively small community, which is confirmed by the quickly saturated diameter and LCC. Example keywords following this pattern include *buzz140*, *cookbook*, *jazz*, and *theonlinemom*.

Finally, patterns P7 and P8 are both characterized by zero diameter, indicating the complete absence of interaction among contributors. Keywords corresponding to `P7` include conventional advertisements (e.g., *spree*), machine-generated messages (e.g., *free*), or common emotions generated exogenously (e.g., *sad*). Surprisingly, more than half of our collection of bursty keywords belong to this pattern. `P8`, on the other hand, is associated with the recurring use of keywords that involve only one user. For this uninteresting pattern, however, we note that if we consider only the usage dynamics (the left panels of Figure 2.3 and the cases in [78, 29]), its shape is often undistinguishable to that of `P1 - P6`. This also resonates to our findings in Figure 4.2.

4.6.5 Clustering-enabled Applications

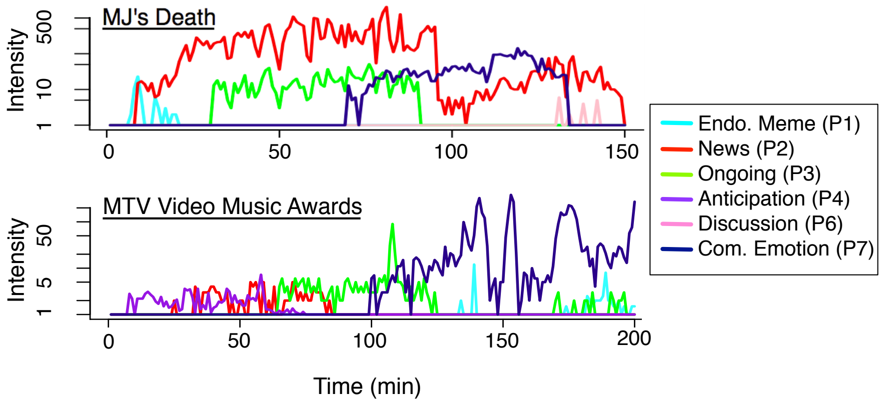
While the clustering results generated by ASH are meaningful on their own right, we further explore two novel applications that are enabled by these clustering results.

Decomposition of collective-behaviors: During major real-world events, the crowd often respond using a sequence of intertwining collective behaviors. However, because these collective behaviors are typically mixed together, observing them is quite difficult. To investigate this question, we study the bursty keywords during two different major events, namely, Michael Jackson's death in June 2009 and the MTV Video Music Awards in September 2009. For each event, we collect all bursty keywords around the time the event took place and manually ruled out the ones that are irrelevant. We then group these keywords according to which pattern in Figure 4.6 they belong to, and plot each pattern's dynamic intensity, i.e., the per-minute usage of all keywords sharing the same pattern.

In Figure 4.7, we present the intensity plots at the top and the list of bursty keywords along with the social-dynamics patterns they belong at the bottom. We first look at the event of Michael Jackson's death. Initially, an endogenous source (i.e., from within Twitter) reported that "*Michael Jackson rushed to hospital*". As shown in Figure 4.7, the unconfirmed message propagates rapidly using pattern `P1`. Soon enough, many websites started to report this news, generating an intense broadcasting using pattern `P2`. As sources exogenous to Twitter became increasingly available, a wave of P3 emerges around $t = 30$. After the source had been confirmed by the more reputed sources (e.g. LA Times and CNN), many users flooded to Twitter, not to learn more about this event, but to share their sorrow about it. This happens around $t = 75$ when pattern `P7` emerges. Note that there is no interaction among these users, since `P7` has zero diameter (see Figure 4.6). Finally, around $t = 135$, a small group of music lovers reminisced about Michael Jackson's masterpieces.

The event of MTV Video Music Awards, unlike the previous one, was already anticipated because all awards were generated through online voting. Such an anticipation is reflected by the initial attention around time $t = 5$ of pattern `P4`. As the pre-show began around time $t = 25$, a few online media started to report this event using pattern `P2`. As more and more celebrities showed up, people started to talk about them using pattern `P3`. But then, around $t = 100$ while Taylor Swift was making her speech for winning the Best Female Video, Kanye West got onto the stage, snatched her microphone, and said that "*Beyoncé had one of the best videos of all time*". Because of Kanye's inappropriate demeanor, the crowd's anger soon took over Twitter using pattern `P7`. Moreover, some endogenous meme also emerged around $t = 140$ using `P1`, including the most-retweeted posting (from Pink) during the show: "*Kanye West is the biggest piece of shit on earth. Quote me*". Finally, around $t = 180$, the show played the trailer of Michael Jackson's "This is it", triggering the crowd's reminiscence in forms of patterns `P3` and `P7`.

During both events, all these intertwining collective behaviors were originally



Pattern	MJ's Death	MTV Video Music Awards
P1	TMZ, rush, hospital, popstar, Michael, Jackson	Kanye, KanyeSucks, Pink, biggest, piece
P2	cardiac, arrest, died, confirmed, CNN, LA	mtva, vma, let, go, Janet, Jay, idol
P3	heart, attack, michaeljackson	Madonna, speech, Gaga, outfit, Shakira, live, king
P4	-	vmas, ready, excited, watch, homework, preshow, tonight
P6	song, thriller	-
P7	sad, shock, rip, shame, omg	West, rude, outburst, wtf, taylor, Beyoncé, trailer

Figure 4.7: Decomposition of intertwining collective behaviors during two major events: Michael Jackson's death and MTV Video Music Awards. Top: evolution of intensities of each type of collective behavior. Bottom: List of bursty keywords and the patterns they belong to.

lumped together and difficult to observe. Using the decomposition made available by using our clustering results, we get to visualize clearly the kinds of dynamic collective reactions among Twitter users.

Clustering-augmented forecasting: We now turn to investigate the hypothesis that knowing the clustering information of social dynamics can help predict its social dynamics. To test this hypothesis, we use the same seven time-series features as shown in Figure 4.6. Our data up to October 31, 2009 are used for training, while

Method	$t_{ob} = 12$	$t_{ob} = 15$	$t_{ob} = 18$
VARMA	0.296	0.237	0.191
Aug. VARMA	**0.219**	**0.141**	**0.105**

Table 4.3: Forecasting error (via RMSE) of conventional and augmented VARMA models.

the data afterwards are reserved for testing.

We train two kinds of predictors using the *Vector Autoregressive and Moving Average* (*VARMA*) model [7]: The *conventional VARMA* trains a unified model using all training data, whereas the *augmented VARMA* trains a separate model for each cluster in Figure 4.6. During testing, the dynamics of the seven time series are observed up to $t = t_{ob}$. For example, setting $t_{ob} = 12$ denotes that all dynamics up to three minutes before the peak at $t = 15$ (see Figure 4.6) are observed. The models are then used to forecast the values in $t_{ob} + 1$, and values $t_{ob} + 2$ based on the values predicted before, and so on. For the augmented VARMA to choose the right model to use, it needs to know which cluster the test case belongs to. Therefore, we also build an 1-nearest-neighbor (1-NN) classifier [41] based on the observed dynamics up to t_{ob} and the distance measure given in Equation 4.2. The prediction accuracy of these 1-NN classifiers for $t_{ob} = 12, 15, 18$ are 82.0%, 93.5%, and 96.1%, respectively.

Table 4.3 summarizes our results using conventional vs. augmented VARMA under different values of t_{ob}. In general, we see that the more data observed (i.e., higher value of t_{ob}), the lower error can be achieved. Moreover, the augmented VARMA outperforms the conventional VARMA in all cases. This is because the augmented model exploits the clustering information predicted by the 1-NN classifier. These results confirms our hypothesis that knowledge about social-dynamics patterns can help predict its associated social dynamics.

Chapter 5

Structure Decomposition using Recursive Convolutional Bayesian Model

In this chapter, we present the Recursive Convolutional Bayesian Model (RCBM) to address *multi-scale compositionality* of social dynamics. In particular, we focus on the identification of two things: (1) the signatures of social dynamics at different time scales, and (2) the way in which these signatures interact and form higher-level meanings. The key idea behind our approach consists of constructing a deep-learning framework using specialized convolution operators that are designed to exploit the inherent heterogeneity of social dynamics. RCBM's runtime and convergence properties are guaranteed by formal analyses. Experimental results show that the proposed method outperforms the state-of-the-art approaches both in terms of solution quality and computational efficiency. Indeed, by applying the proposed method on two social network datasets, Twitter and Yelp, we are able to identify the *compositional structures* that can accurately characterize the complex social dynamics from these

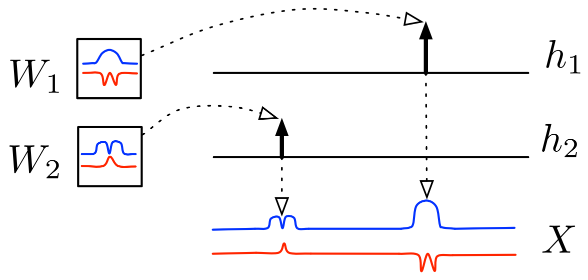


Figure 5.1: CBM's generation process. W: filter matrices; h: activation vectors; X: social dynamic. The filters W_1 and W_2 are activated differently depending on their corresponding activation vectors h_1 and h_2, respectively.

two social media. We further show that identifying these patterns can enable new applications such as anomaly detection and improved social dynamics forecasting.

5.1 Convolutional Bayesian Model (CBM)

5.1.1 Problem definition

As mentioned in Section 2.1, we use $X \in R^{D \times T}$ to represent the D-dimensional social dynamics corresponding to an information token (e.g., $D = 2$ for the X in Figure 5.1). Given a set of social dynamics $\{X^{(i)}\}_{i=1}^{(n)}$ (associated with n information tokens), our problem is defined as finding a set of D-dimensional *structures* (e.g., the W_1 and W_2 in Figure 5.1) that best characterize these dynamics. All notations in this chapter are summarized in Table 5.1.

5.1.2 Assumptions

We aim at solving the above problem under the following three assumptions:

A1. *Finite Structures*: the social dynamics can be characterized by a finite number of structures that are invariant to shifting in time and scaling in magnitude.

Base Model	
X	social-dynamic matrix. $X \in R^{D \times T}$
W_k	the k-th filter matrix. $W_k \in R^{D \times T_w}$
h_k	the k-th activation vector. $h_k \in R_+^{T+T_w-1}$
σ, β	parameters of $P(X\|h)$ and $P(h)$, respectively
K	the number of filters (in one level)
T_w	filter scale
Model Learning	
$\{X^{(i)}\}_{i=1}^n$	set of n sample social dynamics
n	size of sample social dynamics
$h_k^{(i)}$	the k-th activation vector of the i-th sample
t_{h_k}, t_{W_k}	step-sizes for updating h_k and W_k, respectively
$h_k^{[r]}$	the solution of h_k in the r-th optimization iteration
$W_k^{[r]}$	the solution of W_k in the r-th optimization iteration
Stacking Multiple Layers	
X_l	input dynamic at level l
$h_{l,k}$	the k-th activation vector at level l
$W_{l,k}$	the k-th filter matrix at level l
K_l	the number of filters at level l
c	the factor used for downsampling
L	the number of levels of an RCBM

Table 5.1: Summary of all notations in this chapter.

A2. *Burstiness*: the distribution for the magnitude of the social dynamics is right-skewed; it is typically small but can be occasionally very large.

A3. *Heterogeneity*: for each D-dimensional structure, all dimensions have different meanings and no one is an exact copy of another.

The validity of these assumptions has been reported by many previous authors. Indeed, A1 is discussed in [78, 29, 52], A2 in [2, 44, 61], and A3 in [78, 26].

5.1.3 CBM Generative Model

We postulate that each social dynamic X is generated by *random activations of filters*. For illustration, consider Figure 5.1 where W_1 and W_2 represent two *filter matrices*, while h_1 and h_2 represent their *activation vectors*, respectively. From the figure, the social dynamic X is generated by making *copies* of the filter matrices W_1 and W_2. Moreover, the activation vectors determine the time-shift and the magnitude of these two copies: h_2 is active earlier but weaker, hence the first weaker signal in X; h_1 is active later but stronger, hence the latter stronger signal.

Formally, given a set of K filters $\{W_k\}_{k=1}^K$, our generation process for a social dynamic X is:

$$\begin{array}{l} \text{1. Sample } \{h_k\}_{k=1}^K \text{ such that } h_k[t] \sim Exp(\beta)\ \forall k,t \\ \text{2. } X = \sum_k W_k \otimes h_k + \epsilon \text{ where } \epsilon \sim N(0,\sigma^2). \end{array} \tag{5.1}$$

Here $Exp(\cdot)$ and $N(\cdot)$ denote the Normal and Exponential distributions, respectively, with parameters β and σ. Also, $\otimes$ denotes our specialized convolutional operator that carries out the "scale-and-copy" task illustrated in Figure 5.1. It is defined as:

$$(W \otimes h)[d,t] = \sum_{s=1}^{T_w} h[t + T_w - s] \cdot W[d,s]\ \ \forall d,t. \tag{5.2}$$

Note that the $\otimes$ operator differs from the conventional matrix convolution used in [28, 24]. Effectively, $\otimes$ does D 1-D convolutions between each row of W and the entire h, and puts the results back to each row of the output matrix separately. Moreover, the above generation process implies a joint distribution $P(X,h) = P(X|h)P(h)$ where:

$$\begin{array}{lcl} P(X|h;W,\sigma) & = & \frac{\sqrt{2}}{\sigma\sqrt{\pi}} \exp\left(\frac{\|X - \sum_k W_k \otimes h_k\|_F^2}{-2\sigma^2}\right) \\ P(h;\beta) & = & \frac{1}{\beta} \exp\left(\frac{\sum_k \|h_k\|_1}{-\beta}\right). \end{array} \tag{5.3}$$

5.1.4 CBM Features

The design of CBM closely reflects our assumptions A1 ~ A3. To address A1, we use a convolutional formulation such that the structures (i.e., the filters W's) are invariant to shifting in time and scaling in magnitude. To address A2, we enforce burstiness by assuming that the magnitude of the activation vectors (i.e., h's) follows an exponential distribution, which is typically small but occasionally large. This will also enforce sparsity for activation vectors during model learning (see Section 3.2). Finally, to address A3, we consider heterogeneity using our specialized convolutional operator $\otimes$ instead of the conventional matrix convolution. As we will show in Section 4, this provides provable advantages in both runtime and solution quality.

5.1.5 CBM Model Learning

Since given W and h, the Maximum Likelihood Estimators (MLE) for σ and β (in Equation 5.3) can be calculated in closed form, the main challenge for learning a CBM lies in estimating W in presence of the hidden variables h_k's. Formally, the problem can be written as:

$$W^* = \arg\max_W \log P(X) = \arg\max_W \; log \int P(X|h)P(h)dh. \tag{5.4}$$

Assuming that $P(W,h)$ peaks with respect to (w.r.t.) h, we obtain the approximation:

$$\begin{aligned} W^* &\approx \arg\max_W max_h \log P(X|h)P(h) \\ &= \arg\max_{W,h} \tfrac{-1}{2}||X - \textstyle\sum_k W_k \otimes h_k||_F^2 - \tfrac{\sigma^2}{\beta} \textstyle\sum_k ||h_k||_1, \end{aligned} \tag{5.5}$$

where $||\cdot||_F$ denotes the Frobenius norm. Now, considering a set of n samples of social dynamics $\{X^{(i)}\}_{i=1}^n$ and their corresponding activation vectors $\{\{h_k^{(i)}\}_{k=1}^K\}_{i=1}^n$,

Equation 7.6 becomes:

$$\begin{array}{lll} \arg\min_{W,h} & \sum_i \left(\frac{1}{2} \|X^{(i)} - \sum_k W_k \circledast h_k^{(i)}\|_F^2 + \frac{\sigma^2}{\beta} \sum_k \|h_k^{(i)}\|_1 \right) & \\ s.t. & \|W_k\|_F \leq 1 \;\; \forall k & (5.6) \\ & h_k^{(i)} \geq 0 \;\; \forall k, i. & \end{array}$$

In Equation 5.6, two additional constraints are incorporated to improve the solution quality of W. Specifically, the first constraint prevents W_k from blowing up, because otherwise the objective function can be trivially improved by scaling up (and down) W_k (and h_k) by the same factor. Also, the second constraint helps ensure that the signs of W_k are not arbitrary and hence can be interpreted coherently. We note that Equation 5.6 is similar to *sparse coding* in [49] with two important distinctions. First, the conventional matrix multiplication is used in sparse coding whereas a convolutional formulation is used in Equation 5.6. Second, in sparse coding, the penalty strength (usually denoted as λ) needs to be tuned manually, whereas in Equation 5.6, the value of $\frac{\sigma^2}{\beta}$ can be assigned using MLE with a straightforward meaning.

To solve Equation 5.6, since the problem is convex w.r.t. each one of W and h (but not both), we alternate between optimizing over one of them while keeping the other one fixed. To start with, we first derive the derivatives of the smooth part of the objective function (i.e., $f_1(W,h) = \frac{1}{2}\|X^{(i)} - \sum_k W_k \circledast h_k^{(i)}\|_F^2$) w.r.t. h and W:

$$\begin{array}{lll} \nabla f_1(h_k^{(i)}) & = & \tilde{W_k} \odot \left(\sum_j h_j^{(i)} \circledast W_j - X \right) \\ \nabla f_1(W_k) & = & \sum_i \widetilde{h_k^{(i)}} \circledast \left(\sum_j h_j^{(i)} \circledast W_j - X \right). \end{array} \quad (5.7)$$

Here, the deconvolution operator $\odot$ is defined as:

$$(W \odot X)[t] = \sum_{d=1}^{D} \sum_{s=1}^{T_w} X[d, t-s+1] \cdot W[d,s]. \quad (5.8)$$

Again, the $\odot$ operator differs from the conventional matrix convolution. Effectively, it calculates the 1-D convolutions of individual rows of W and X separately, and then adds them together to form a single row. This brings the same advantages as $\otimes$ does as mentioned at the end of Section 3.1.

Stepsize assignment: Typically, one can use line search [6] to determine the stepsize in gradient-based methods. In our case, however, doing so would slow down the optimization considerably because the line search itself needs many additional convolutions. Therefore, we derive the following stepsize assignments for h and W, respectively:

$$\begin{aligned} t_{h_k} &= \frac{\alpha}{||W_k||_1^2} \\ t_{W_k} &= \frac{\alpha}{\sum_i ||h_k^{(i)}||_1^2}, \end{aligned} \tag{5.9}$$

where $\alpha \in (0, 2)$. In Section 4, we show that these stepsize assignments are essential to ensure good runtime and convergence properties.

Overall algorithm: Algorithm 2 provides the pseudocode for CBM learning. It takes as inputs a set of n sample social dynamics $\{X^{(i)}\}_{i=1}^n$, the scale of the filters T_w, and the number of filters K, and produces as outputs all model parameters including $\{W_k^{[r]}\}_{k=1}^K$, σ, and β. In each iteration of the main repeat loop of Algorithm 2, three tasks are executed in turn: Task 1 (the first for-loop) consists of solving Equation 5.6 w.r.t. h; Task 2 (the second loop) consists of advancing one step toward the solution of Equation 5.6 w.r.t. W; Task 3 (the remaining two lines) consists of calculating the MLE for σ and β.

The details of Task 1 are presented in Algorithm 3. This is basically designed based on the Nestrov acceleration [46] and the proximal method [6], where the function $S_\lambda^+(\cdot)$

Algorithm 2 Learning of CBM

```
Data: ${X^{(i)}}_{i=1}^{n}$: $n$ sample social dynamics
Data: $T_w$: scale of the filters
Data: $K$: number of filters
Result: $\{W_k\}_{k=1}^{K}$: solution filters
Result: $\sigma, \beta$: additional model parameters
$W_k^{[-1]} = W_k^{[0]} =$ random initialization $\forall k$
$\sigma = \beta = 1$
$\lambda = \sigma^2 t_{h_k} / \beta$
$r = 0$
repeat
    $r = r + 1$
    for $i$ = 1 to $n$ do
        for $k$ = 1 to $K$ do
            $\{h_k^{(i)}\}_{k=1}^{K} = \text{optimize\_over\_h}\Big(X^{(i)}, \{W_k^{[r-1]}\}_{k=1}^{K}, \sigma, \beta\Big)$
        end
    end
    for $k$ = 1 to $K$ do
        $t_{W_k} = \alpha / \sum_i \|h_k^{(i)}\|_1^2$
        $y = W_k^{[r-1]} + \frac{r-2}{r+1}(W_k^{[r-1]} - W_k^{[r-2]})$
        $W_k^{[r]} = \Pi(y - t_{h_k} \nabla f_1(y))$
    end
    $\sigma = \Big(\frac{1}{n} \sum_i \|X^{(i)} - \sum_k W_k^{[r]} \otimes h_k^{(i)}\|_F^2\Big)^{\frac{1}{2}}$
    $\beta = \frac{1}{n} \sum_{i,k} \|h_k^{(i)}\|_1$
until convergence;
return $\{W_k^{[r]}\}_{k=1}^{K}$, $\sigma$, and $\beta$
```

is an element-wise function defined as:

$$S_\lambda^+(u) = \begin{cases} u - \lambda & \text{if } u > \lambda \\ 0, & \text{otherwise.} \end{cases}$$

Algorithm 3 Optimization over h

```
Data: $X$: a sample social dynamic
Data: $\{W_k\}_{k=1}^K$: filter matrices
Data: $\sigma, \beta$: model parameters
Result: $\{h_k\}_{k=1}^K$: solution activation vectors
$h_k^{[-1]} = h_k^{[0]} =$ random initialization $\forall k$
$t_{h_k} = \alpha/(\|W_k\|_1^2) \ \forall k$
$\lambda = \sigma^2 t_{h_k}/\beta$
$r = 0$
repeat
    $r = r + 1$
    for $k$ = 1 to $K$ do
        $y = h_k^{[r-1]} + \frac{r-2}{r+1}(h_k^{[r-1]} - h_k^{[r-2]})$
        $h_k^{[r]} = S_\lambda^+(y - t_{h_k} \nabla f_1(y))$
    end
until convergence;
return $\{h_k^{[r]}\}_{k=1}^K$
```

Task 2 is conceptually similar to Task 1, where $\Pi(\cdot)$ is defined as:

$$\Pi(W) = \begin{cases} W/\|W\|_F & \text{if } \|W\|_F > 1 \\ W, & \text{otherwise.} \end{cases}$$

One distinction is that instead of solving h until convergence as in Task 1, only a single update is conducted here. Finally, Task 3 calculates the close-form solution of MLE for σ and β. Since the whole algorithm can be viewed as a case of Coordinate Descent [6], it is guaranteed to converge.

Specifying parameters: Algorithm 2 has two parameters, T_w and K, that need to be supplied by the user. The filter scale T_w can be conveniently specified as any small number (e.g., letting $T_w \approx D$) without the need to worry about overlooking the structures at larger scales. This is because the high-level structures with larger scales are meant to be captured by the CBM's at higher levels (that will be described later).

Regarding the number of filters K, since CBM has a natural corresponding probabilistic model (i.e., $P(X, h)$ according to Equation 5.3), a naive method is trying out a range of different K's and select the one that produces the highest Bayesian Information Criterion (BIC) [41], where the latter is a standard metric for model selection. Doing so, however, is very expensive because it requires training a large number of CBM's. Therefore, we propose the following three-step method for selecting K:

1. Pick a large K and train a CBM.

2. Sort all filters such that:

$$p \leq q \iff \sum_{i=1}^{n} \sum_{k=1}^{p} \|h_k^{(i)}\|_1 \geq \sum_{i} \sum_{k=1}^{q} \|h_k^{(i)}\|_1. \tag{5.10}$$

3. Plot the the *cumulative activation function* $F(m)$:

$$F(m) = \sum_{i=1}^{n} \sum_{k=1}^{m} \|h_k^{(i)}\|_1 \tag{5.11}$$

   and pick the new K as the position m^* such that $F(m^*)$ starts to saturate (i.e., when $\frac{dF}{dm} \leq \epsilon$ where $0 < \epsilon \ll 1$ is a small positive number).

The idea behind our method is that, since sparsity is enforced on h_k's using the one-norm in Equation 5.6, the irrelevant filters $\{W_{m^*+1}, ..., W_K\}$ will all have very low activations compared to that of the relevant filters $\{W_1, ..., W_{m^*}\}$. The advantage of this approach is that it requires training only *one* (instead of a large number of) CBM, and hence it is much more efficient. The effectiveness of this method is validated in Section 5.

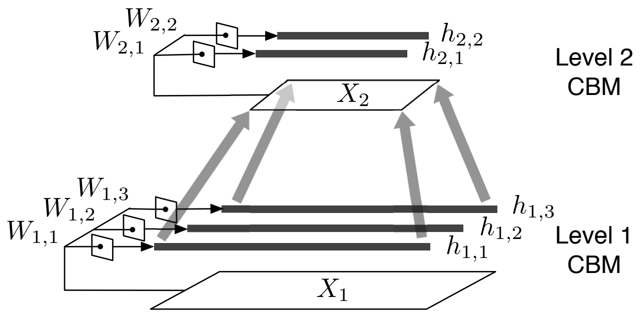


Figure 5.2: A two-level RCBM. W: filter matrices; h: activation vectors; X: input dynamics. The upper-level input dynamic is constructed by max-pooling the activation vectors that are one level below it.

5.2 Recursive Convolutional Bayesian Model (RCBM)

5.2.1 Recursive CBM

To capture the compositional structure of social dynamics across different scales, we now introduce RCBM, a hierarchical architecture constructed by stacking together multiple CBM's, as illustrated in Figure 5.2. For some new notations, we use l to represent any variable at the l-th level, including X_l (input dynamic of level l), $h_{l,k}$ (the k-th activation vector at level l), $W_{l,k}$ (the k-th filter matrix in level l) and K_l (the number of filters in level l). Also, L denotes the total number of levels of an RCBM.

Suppose we have trained a CBM with $K = 3$ following the procedures described in Section 3.2, like the Level 1 CBM in Figure 5.2. To raise the level of abstraction, we construct the input dynamics at level 2 (i.e., X_2) by down-sampling the lower-level activation vectors (i.e., $h_{1,1}$, $h_{2,1}$, and $h_{3,1}$) by a factor of c using a non-linear *max-pooling* [28, 24], which simply takes the maximum value among consecutive c values. For example, since $K_1 = 3$ in Figure 5.2, X_2 will have three rows of length $\lceil \frac{T+T_w-1}{3} \rceil$, where $T + T_w - 1$ is the length of $h_{k,1}$. Moreover, the values of X_2 will be assigned as

$X_2[d,t] = \max_{s \in \{1,\ldots,c\}} h_{d,1}[c(t-1)+s]$.

After doing max-pooling for each sample, we obtain a set of level-2 dynamics (i.e., X_2) for the whole dataset. We can then use these level-2 dynamics as if they are a set of new social dynamics and train another CBM as before, like the Level 2 CBM in Figure 5.2. Repeating this layer-wise training process for L times, we obtain an RCBM of L levels. Note that the number of filters K_l at each level can be different, e.g., in Figure 5.2, we have $K_1 = 3$ and $K_2 = 2$. Also, note that even if the filter scale T_w remains constant across different levels, the higher-level filters will still detect larger-scale dynamics, i.e., a level-l filter effectively looks at the dynamics of scale $c^{l-1}T_w$. Besides focusing at larger scales, a higher-level filter can also detect the dynamics of higher levels of abstraction, because it is trained using the lower-level activation vectors, which are themselves a non-linear transformation of their input dynamics. This is how RCBM can recognize the compositional structures of social dynamics across different scales and levels of abstractions.

5.2.2 RCBM features

While RCBM inherits all the features of CBM (in Section 3.1), it has two additional features that are reflected in its name. First, all levels of an RCBM share the same structure, hence the name "*recursive*". This ensures that the numbers of activation vectors remain roughly the same across different levels. This is in sharp contrast to other convolutional deep architectures like [28, 24], where the number of activation vectors becomes K^2 from the second level; this seriously limits the efficiency and scalability of previous algorithms. Second, by using Equation 5.3, we can decompose the joint probability of the entire RCBM using Bayes' rule:

$$\begin{aligned} P(X,h) &= \textstyle\prod_l P(X_l|h_l;W_l,\sigma_l)P(h_l;\beta_l) \\ &= \frac{1}{Z}\exp\left(\sum_l \frac{\|X_l-\sum_k W_{l,k}\otimes h_{l,k}\|_F^2}{-2\sigma_l^2} + \frac{\sum_k \|h_{l,k}\|_1}{-\beta_l}\right), \end{aligned} \tag{5.12}$$

hence the name "*Bayesian*". Moreover, we note that RCBM is normalized locally according to Equation 5.3. Therefore, the partition function Z in Equation 5.12 can be calculated efficiently using 5.3 and the first line of Equation 5.12; this makes various inferences of RCBM efficient. Finally, such a probabilistic formulation also enables many new applications such as conditional inferences and anomaly detection.

5.2.3 Model summary

To summarize, RCBM possesses three attractive properties:

- *Good solution quality*: under A1 ~ A3, RCBM is capable of identifying compositional structures of social dynamics that have provable convergence qualities. This is attributed to our specialized convolution operators ($\otimes$ and $\odot$) and stepsize assignment (Equation 5.9).

- *Efficiency*: the learning of RCBM is efficient and can scale much better than existing convolutional deep learning methods [28, 24]. This is attributed to our specialized convolution operators, stepsize assignment, and the recursive structure.

- *Wide applicability*: RCBM can be applied to a range of applications. For one, it can be used as the feature extractor for supervised tasks. For another, its probabilistic formulation (Equation 5.12) enables various conditional inferences and anomaly detection.

While all these properties are verified empirically, we formally establish the first two properties in the next section.

5.3 Convergence and Runtime Analysis

We now establish formally that under assumptions A1~A3, the specialized operators enable the learning of RCBM to produce good solutions efficiently, whereas the conventional one does not. The proof of all theorems in this section is given in Appendix B.

5.3.1 Convergence Properties

Theorem 3. ***Convergence using the proposed convolution.*** *Suppose a dataset* $\{X^{(i)}\}_{i=1}^{n}$ *is generated according to the process in Equation 5.1 using filters* W^*. *Also, suppose Algorithm 2 is used with the stepsize given in Equation 5.9, where* $W^{[0]}$ *denotes the initial condition and* $\hat{W}$ *denote the converged solution. Then we have that* $\forall W^*$, $\exists W^{[0]}$ *s.t.:*

$$\hat{W} \xrightarrow{p} W^*.$$

Theorem 4. ***Non-convergence using the conventional convolution.*** *With the same assumptions as in Theorem 3 but supposing that the conventional matrix convolution is used, then* $\forall W^*$, *we have that:*

$$\hat{W} \overset{p}{\nrightarrow} W^*.$$

The main message from these two theorems is that using the proposed convolution operators can lead to better convergence because it considers the heterogeneity (Assumption A3) in social dynamics. Although, in principle, not every initial condition ($W^{[0]}$) leads to the global optimum since Equation 5.6 is not jointly convex w.r.t. W and h. Practically, however, our experimental results (see Figure 5.5) show that a handful of random initializations suffice to produce good and reliable solutions.

5.3.2 Runtime Complexity

From Algorithm 2, the bottleneck of training an RCBM is the function `optimize_over_h` (i.e., Algorithm 3), because it is called repeatedly and that it is itself an iterative algorithm. Accordingly, we break down the runtime complexity analysis into two parts: (1) bounding the number of iterations r it takes to solve Algorithm 3, and (2) analyzing the overall runtime complexity while treating r as a constant. The first part is established using Theorem 5.

Theorem 5. ***Required Number of Iterations*** *Suppose an accelerated proximal method is used to solve Equation 5.6 w.r.t. either W or h but not both. Let $x^{[r]}$ denote the solution in the r-th iteration, x^* denote the optimal solution, and ϵ denote an error threshold. Then if the stepsizes given in Equation 5.9 are used, the number of iterations r that ensures $||x^* - x^{[r]}|| \leq \epsilon$ satisfies:*

$$r = O(\epsilon^{\frac{-1}{2}}).$$

We note that this represents the fastest convergence rate achievable using first-order methods [6], which is attributed to the careful design of stepsizes in Equation

5.9.

For the second part, both the $\otimes$ and $\odot$ operators take $O(DT_wT)$ to calculate. Considering L levels and K filters (and activation vectors) per level, the total complexity is bounded by $O(KDT_wTL)$. In contrast, previous works [28, 24] use the conventional matrix convolution that requires $O(D^2T_wT)$ per operation. Further, previous approaches need K^2 activation vectors to calculate from the second level up. Therefore, their total complexity is $O(K^2D^2T_wTL)$. Using $K = 10$ and $D = 10$, this represents a huge runtime overhead of two orders of magnitudes. Finally, we note that T and T_w are completely decoupled; typically, T (the length of dynamics) can be large but T_w (the length of the filters) is fixed.

5.4 Experimental Results

We conduct extensive experiments in the following three directions: (1) the evaluation of RCBM per se, (2) compositional structures in Twitter and Yelp discovered using RCBM, and (3) two new applications enabled by RCBM.

5.4.1 Evaluation of RCBM

Parametric forms: We first verify the distributional assumptions we made in Equation 5.3. To this end, we use each of the two datasets to train a 1-level CBM. For each sample X, we calculate the *per-sample error* $||X - \sum_k W_k \otimes h_k||_F$ and the *per-sample activation* $\sum_k ||h_k||_1$. We then compare their empirical distributions to their model distributions (i.e., according to Equation 5.3). From the results in Figures 5.3 (Twitter) and 5.4 (Yelp), the empirical distributions and the model distributions seem to match reasonably well. A close examination of the activation vectors confirms that sparsity is enforced effectively such that for each activation vector, most of its ele-

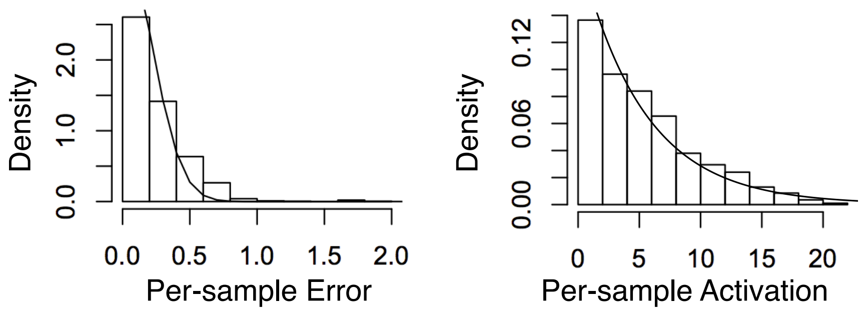


Figure 5.3: Empirical vs. fitted distributions of error and activation using the Twitter dataset. Left: per-sample error fitted with the Half-Normal distribution; Right: per-sample activation fitted with the Exponential distribution.

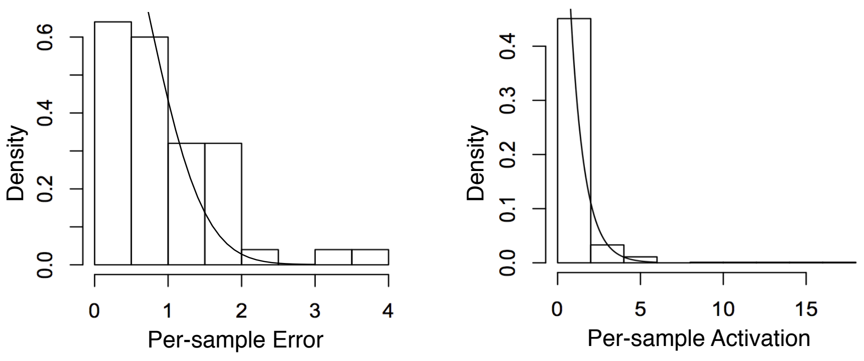


Figure 5.4: Empirical vs. fitted distributions of error and activation using the Yelp dataset. Left: per-sample error fitted with the Half-Normal distribution; Right: per-sample activation fitted with the Exponential distribution.

ments are exactly zero. These observation supports the validity of our formulations in Equations 5.3 and 5.6.

Runtime and solution quality: We then turn to evaluate the runtime performance and the solution quality of RCBM against deep-learning and non-deep-learning methods. For the baseline deep-learning method, we use cAE [24] as it represents the state-of-the-art convolutional deep learning algorithm. For the proposed method, we test two versions of RCBM: one determines the stepsizes using line search [6] (RCBM-LS); the other uses the proposed fixed stepsize in Equation 5.9 (RCBM-FS). Using each method, we vary the sample size in the range of 100 to 10000 and train a two-level model with 10 filters at each level. The solution quality of the learnt models is measured using perplexity [41] calculated using 3000 randomly sampled held-out test data. Intuitively, perplexity measures how closely the model distribution resembles the empirical distribution, where a lower value indicates a better model. All experiments are run using 10 repetitions, where both the means and the standard deviations are reported.

From the left panels of Figures 5.5 (Twitter) and 5.6 (Yelp), we first observe that RCBM-LS and RCBM-FS run significantly faster than cAE. Indeed, while cAE scales up to 500 samples, both RCBM-LS and RCBM-FS scale to 10,000 samples, which confirms our analysis in Section 4.2. Moreover, RCBM-FS runs much faster than RCBM-LS: while it may take more than 3 days to train an RCBM-LS with 10,000 samples, it takes around 17.5 hours using RCBM-FS. Accordingly, RCBM-FS achieves a 4X~6X speedup compared to RCBM-LS, or a 30X~100X speedup compared to cAE. Such a significant speedup is attributed to several of our carefully-designed features, including the fixed stepsizes, the specialized convolutions, and the recursive structure of RCBM.

For solution quality, we can observe from the right panels of Figures 5.5 and 5.6 that RCBM-LS and RCBM-FS perform comparably and both perform considerably better than cAE, which confirms Theorems 1 and 2. This is because they both incorporate our specialized convolution operators that exploit the heterogeneity of social dynamics, which is not considered by the conventional convolutions used in cAE.

To gain further insight, we compare our proposed method (i.e., RCBM-FS) against two non-deep-learning methods that also incorporate latent factors, i.e., SSM and SC (see Section 2). For a fair comparison, we setup SSM and SC such that each of them

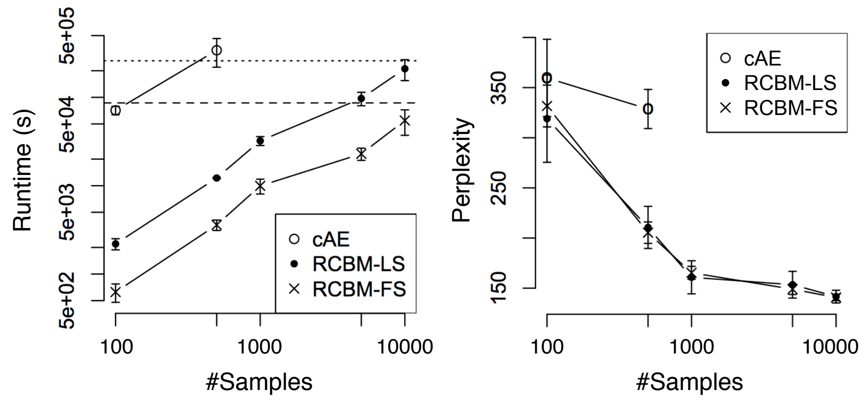


Figure 5.5: Runtime and perplexity comparisons among deep-learning methods using the Twitter dataset. Baseline: the convolutional autoencoder [24]; LS: RCBM with stepsizes determined by line-search; Proposed: RCBM with stepsizes determined by Equation 5.9. On the left panel, the dashed and dotted lines mark the runtimes of one and three days, respectively.

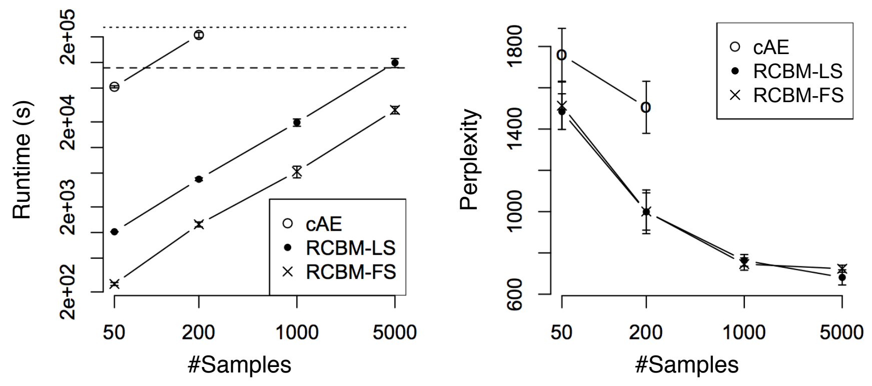


Figure 5.6: Runtime and perplexity comparisons among deep-learning methods using the Yelp dataset. Baseline: the convolutional autoencoder [24]; LS: RCBM with stepsizes determined by line-search; Proposed: RCBM with stepsizes determined by Equation 5.9. On the left panel, the dashed and dotted lines mark the runtimes of one and three days, respectively.

has an equal or slightly larger number of parameters compared to that of RCBM-FS. Similar to Figures 5 and 6, we train these models using the Twitter (Figure 5.7) and Yelp datasets (Figure 5.8) and present the the runtime and perplexity results.

In terms of runtime (i.e., the left panels of Figures 5.7 and 5.8), we observe that SC

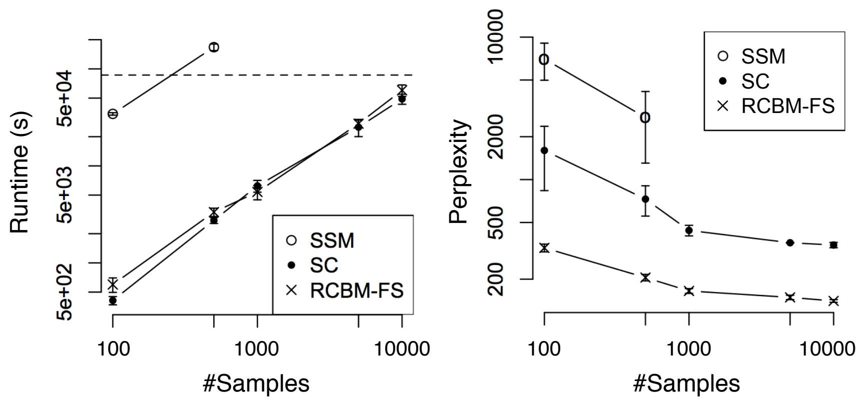


Figure 5.7: Runtime and perplexity comparisons against non-deep-learning methods using the Twitter dataset. SSM: State Space Model; SC: Sparse Coding; RCBM-FS: the proposed method. On the left panel, the dashed line marks the runtime of one day.

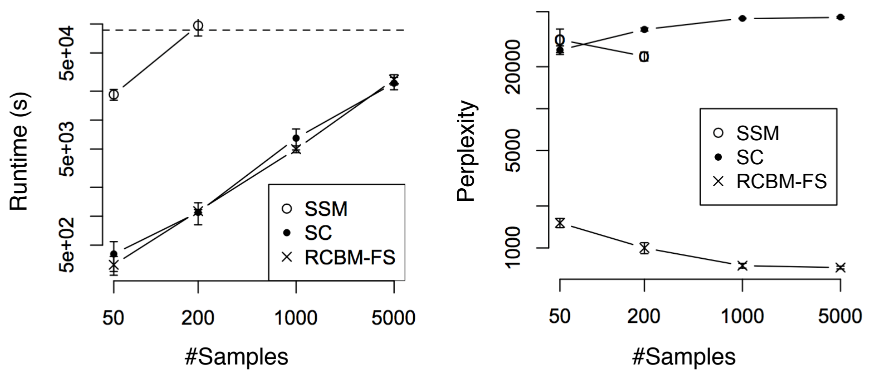


Figure 5.8: Runtime and perplexity comparisons against non-deep-learning methods using the Yelp dataset. SSM: State Space Model; SC: Sparse Coding; RCBM-FS: the proposed method. On the left panel, the dashed line marks the runtime of one day.

and RCBM-FS run much faster than SSM. This is because the standard expectation-maximization (EM) training of SSM involves multiplication and inversion of matrices [20]. Therefore, the complexity for one optimization step is $O(n(K^2T^2+D^2T^2)+K^3+D^3)$, which can be very high for large n, K, T, or D. Further, despite that SC is theoretically faster than RCBM by a constant factor T_w, we observe that they have comparable runtime in practice. This is attributed to our careful design of stepsize selection, which contributes to a 4~6X runtime speedup (see the runtime of RCBM-LS and RCBM-FS in Figures 5 and 6).

In terms of solution quality, (see the right panels of Figures 5.7 and 5.8), we observe that RCBM-FS performs much better than SSM and SC. The reason why SSM performs poorly is that it makes a rather strong modeling assumption that the dynamical transition of the hidden factors are both linear and time-invariant, which is typically not true in practice. As for SC, the reason is more involved. For the Twitter dataset where the majority of time series have a single peak and are aligned accordingly, SC performs better than SSM because it makes fewer assumptions about the time series dynamics. Still, SC performs poorly compared to RCBM, because it wastes the majority of its parameters in capturing the global signatures at the same scale. In contrast, RCBM-FS uses its parameters more efficiently by exploiting the local signatures of different scales. For the Yelp dataset where the majority of times series have multiple peaks that cannot be aligned, SC performs *worse* than SSM. Indeed, the perplexity even *increases* as the number of samples grows, showing that the incapability of SC to deal with time-shifts represents a serious issue when the time series are not pre-aligned.

Efficient selection of K: Next, we compare the naive and the proposed methods in selecting the best number of filters K; both methods are described at the end of Section 3.2. With each of our two datasets, we train two-level RCBM's with both methods. For the naive method based on Bayesian Information Criterion (BIC), we calculate BIC while fixing $K = 10$ for one of K_1 and K_2 and varying the other; this requires training 20 RCBM's in total. For the newly proposed method, we train only one RCBM using $K_1 = K_2 = 10$ while plotting the cumulative activation function $F(m)$ in Equation 5.11 for both levels. The results are summarized in Figures 5.9 (Twitter) and 5.10 (Yelp), where good choices of K are indicated by peak BIC's and

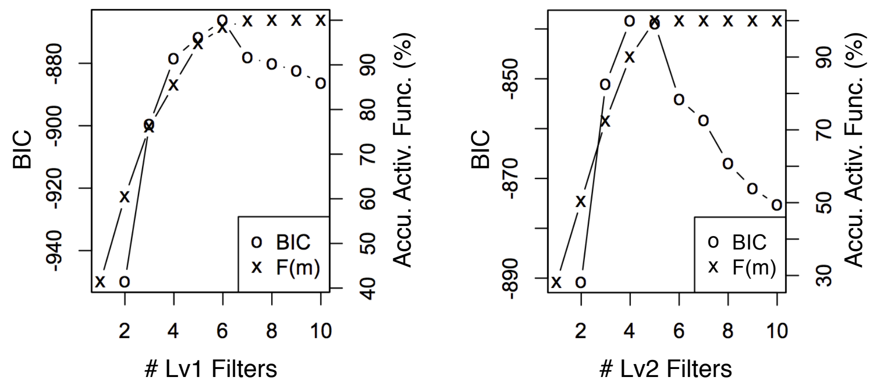


Figure 5.9: BIC and accumulated activation function of level-1 (left) and level-2 (right) filters using the Twitter dataset. The choices of K's using the two statistics match each other.

the points where $F(m)$ saturates. In both panels of both figures, we observe that the choices of K's suggested by BIC and $F(m)$ are nearly identical, although it requires training 20 RCBM's to obtain the BIC curves but only one to obtain the $F(m)$ curves. Moreover, manual inspection confirms that for the Twitter dataset, $W_1 \sim W_5$ of both levels consist of clearly interpretable filters, whereas $W_6 \sim W_{10}$ of both levels consist of plain noise; for the Yelp dataset, similarly, the first six (four) filters in level one (two) are clearly interpretable, whereas the last four (six) filters in level one (two) consist of plain noise. Accordingly, we conclude that the proposed method of choosing K is both efficient and effective.

5.4.2 Compositional Structures of Social Dynamics

We now investigate the compositional structures of social dynamics by inspecting the learnt filters (i.e., W's in Equation 5.12) in RCBM. We first note that this analysis is in sharp contrast with the ones given in [78, 29, 52] in two ways. First, the goal in [78, 29, 52] is finding representative samples, which is essentially *clustering*; our goal, on the other hand, is finding structures that best characterize social dynamics, which

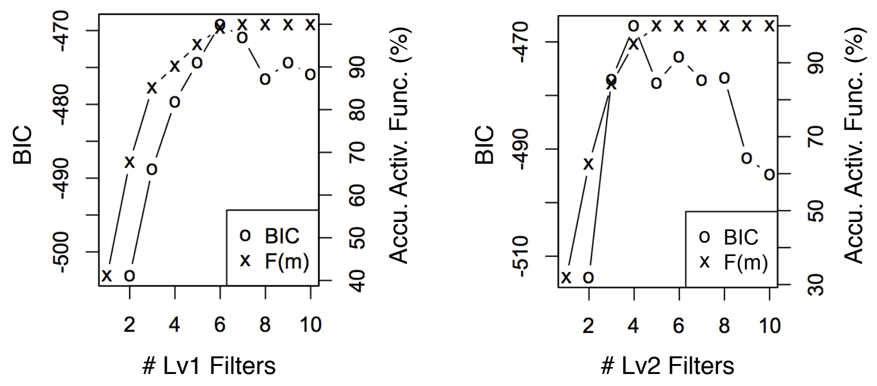


Figure 5.10: BIC and accumulated activation function of level-1 (left) and level-2 (right) filters using the Yelp dataset. The choices of K's using the two statistics match each other.

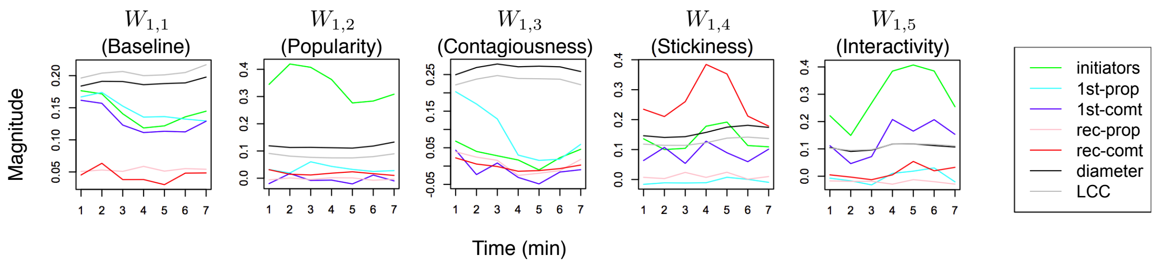


Figure 5.11: The level-1 compositional structures of Twitter social dynamics identified using a two-level RCBM. They represent the fine-grained signatures of Twitter social dynamics including the baseline, popularity, contagiousness, stickiness, and interactivity.

is essentially *decomposition*. Second, our method is compositional and scale-free.

Compositional Structures in Twitter

For the Twitter dataset, we use $K_1 = K_2 = 5$ according to the experiment in Figure 5.9 and train a two-level RCBM. The level-1 filters correspond to compositional structures of seven minutes, whereas the level-2 filters correspond to those of 30 minutes. All these filters are visualized in Figures 5.11 and 5.12. In both figures, the filters are ranked according to their corresponding activation strength (i.e., Equation 5.10).

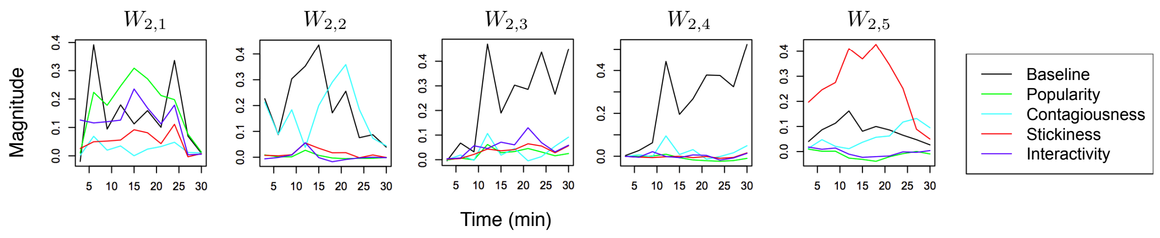


Figure 5.12: The level-2 compositional structures of Twitter social dynamics identified using a two-level RCBM. They represent the interactions among the fine-grained signatures in Figure 5.11.

Level-1 structures: The filter $W_{1,1}$ in Figure 5.11 represents the baseline of typical Twitter social dynamics. It corresponds to a strong community indicated by the black and grey lines of the graph diameter and LCC (largest connected component), respectively. Such a strong community is mainly attributed to the initiators (green), first-time propagators (light blue), and first-time commentators (blue), but not the recurrent propagators (pink) and commentators (red). Such a baseline structure matches Twitter's responsive and light-weighted nature. The filter $W_{1,2}$ characterizes the *popularity* of social dynamics. It mainly consists of the number of initiators, with minor first-time propagators, characterizing how popular a piece of information is from the external sources outside of Twitter, e.g., TV, web news, etc. The filter $W_{1,3}$ characterizes the *contagiousness* of social dynamics that consists of mainly first-time propagators (light blue) and the corresponding strong community indicated by the diameter (black) and the LCC (grey), despite only a small number of initiators (green). The filter $W_{1,4}$ characterizes the *stickiness* of social dynamics, which consists of mainly recurrent commentators (red) with smaller but proportional numbers of initiators (green) and first-time commentators (blue). It characterize the capability of a social dynamic to retain the attention of the users and keep commenting about it. We note that the community-related dynamics (diameter and LCC) are also weaker since the corresponding community is much smaller compared to that of $W_{1,3}$. The filter $W_{1,5}$ characterizes the *interactivity* of social dynamics, which has the largest magnitude of first-time commentators (blue) among all level-1 filters. It characterizes the capability of a social dynamic to motivate users to spend time and comment on it, instead of merely passing it along (i.e., propagating it) to other users.

Level-2 structures: We now turn to investigate the level-2 filters as visualized in Figure 5.12. Note that each individual component on the right of Figure 5.12 corresponds to one level-1 filter in Figure 5.11, and that the time scale now is 30 minutes instead of 7 minutes that is the case of Figure 5.11. This is because the level-2 filters are intended to capture how the level-1 structures interact with one another and form larger-scale structures with high-level meanings, which is a unique feature of RCBM.

The filter $W_{2,1}$ characterizes a three-stage structure that is driven mainly by popularity (the green line), but accompanied by different structures in each of its stages. It is accompanied firstly by contagiousness (light blue), secondly by interactivity (blue) and stickiness (red), and thirdly by combinations of the three. The contagiousness dips in the second stage, but gets enhanced again in the third stage, suggesting that contagiousness *alone* is not enough to sustain long-lasting social dynamics. The filter $W_{2,2}$ is mainly composed of strong contagiousness, which dips at around time $t = 12$, and is later continued and enhanced by interactivity and stickiness. Manual inspection shows that the contagiousness results from *reporting* some facts before $t = 12$, whereas it results from *commenting* about the facts, e.g., from famous bloggers, after $t = 12$. The filter $W_{2,3}$ and $W_{2,4}$ are also driven by contagiousness, but their corresponding contagiousness components have a smaller magnitude. The key difference among the two is that in $W_{2,3}$, strong interactivity and stickiness are generated as a result of the initial contagiousness, which is much weaker in the case of $W_{2,4}$. As a result, the dynamics with top 10% $W_{2,3}$ activations reaches more than three times larger audiences compared to the case of the dynamics with top 10% $W_{2,4}$ activations. Finally, the filter $W_{2,5}$ exhibits a clear two-stage structure. The second stage characterized by contagiousness (light blue) seems to result from the first stage that is characterized by strong stickiness. Manual inspection shows that such a structure consists of committed core users and responsive peripheral users, which is consistent with the *leader-follower* pattern reported in [52]. In the present work, however, the local structures of the pattern as well as the interaction among these structures are decomposed and analyzed in much greater detail.

To summarize, we find three representative ways where smaller-scale signatures can interact and form lager-scale structures with higher-level meanings. First, popularity can stimulate interactivity, stickiness, and contagiousness (i.e. $W_{2,1}$). Second, contagiousness can generate interactivity and stickiness, which, in turn, enhance contagiousness (e.g., $W_{2,2}$ and $W_{2,3}$). Finally, stickiness beyond a certain threshold can generate contagiousness (e.g., $W_{2,5}$).

Compositional Structures in Yelp

For the Yelp dataset, we use $K_1 = 6, K_2 = 4$ according to the experiment in Figure 5.10 and train a two-level RCBM. The level-1 filters correspond to compositional structures of one year, whereas the level-2 filters correspond to that of six years. Again, these filters are ranked according to their corresponding activation strength (i.e., Equation 5.10) and visualized in Figures 5.13 and 5.14.

Level-1 structures: Each level-1 structure indicate a particular level of rating (red lines in Figure 5.13) given by one of the two types of reviewers: a smaller number of *elite reviewers* who have a higher level of experience (green lines) and influence (blue lines), and a larger number of *average reviewers* who are the opposite of their

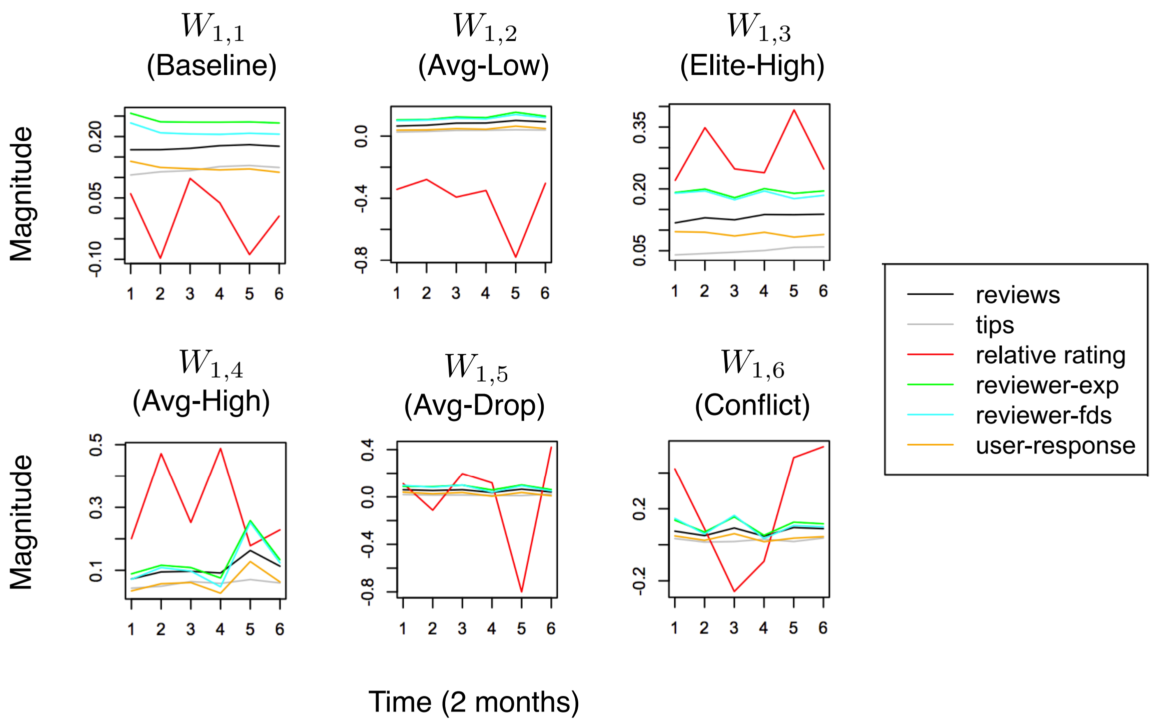


Figure 5.13: The level-1 compositional structures of Yelp social dynamics identified using a two-level RCBM. They represent the fine-grained signatures of Yelp social dynamics including high and low ratings given from reviewers with different levels of experience. The y- and x- axes in all plots are magnitude and time (in two-month units), respectively.

counterparts. The filter $W_{1,1}$ in Figure 5.13 represents the baseline of a typical Yelp social dynamic. It corresponds to neutral ratings (indicated by the small magnitude of the red line) given by *elite reviewers* who are more . Since this filter gets activated the most among all level-1 filters, it is consistent with the fact that the majority of Yelp contents are provided by a relatively small set of elite users. Moreover, it is consistent with the fact that most Yelp businesses have ratings close to the overall average (i.e., around 3.7 stars). The filter $W_{1,2}$ detect the cases when a business is given low ratings by average reviewers; The filter $W_{1,3}$ characterize the case when a business is given high ratings by elite reviewers; The filters $W_{1,4}$ and $W_{1,5}$ indicate the cases when high and low ratings are given by average users, respectively. Note

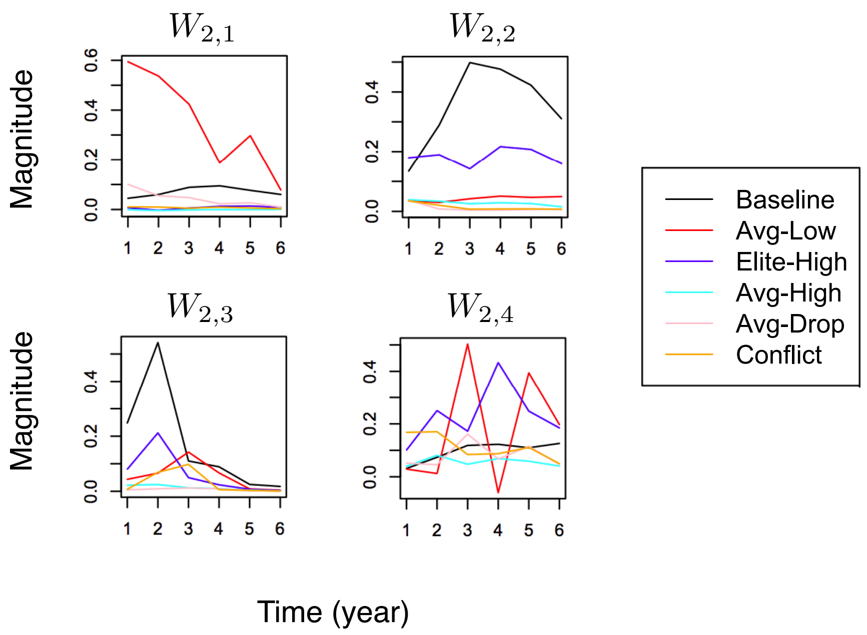


Figure 5.14: The level-2 compositional structures of Yelp social dynamics identified using a two-level RCBM. They represent the interactions among the fine-grained signatures in Figure 5.13. The y- and x- axes in all plots are magnitude and time (in one-year units), respectively.

that there is a difference between $W_{1,5}$ and $W_{1,2}$: in the former case, the rating for the business was neutral for several months, but drop suddenly at $t = 5$; In the latter case, the rating is low from the beginning, and become particularly so at $t = 5$. Finally, in $W_{1,5}$, the rating oscillates significantly, where the extreme values at $t = 1, 3, 5$ are all driven by elite users. This seem to characterize a conflict in rating a business among different groups of elite users, comparable to the edit wars in Wikipedia [66].

Level-2 structures: The level-2 structures of Yelp social dynamics are summarized in Figure 5.14. Note that the time scale now is six years instead of one year that is the case of Figure 5.13. Again, each individual component on the right of Figure 5.14 corresponds to one level-1 filter in Figure 5.13. The filters $W_{2,1}$ and $W_{2,2}$ indicates cases where a business is consistently given low and high ratings by average and elite users, respectively. While it is interesting to see these two common long-term dynamics that present on Yelp, it is equally informative to see that the opposite cases are *uncommon*. That is to say, from our data, it is uncommon to see a business that is consistently given high ratings by average users, or low ratings by elite users. Moreover, the filters $W_{2,3}$ and $W_{2,4}$ both show disagreement in the ratings of average versus elite reviewers. Indeed, in $W_{2,3}$, the high ratings from the elite reviewers (blue line) at $t = 2$ is substituted by the low ratings from the average reviewers (red line) at $t = 3$, accompanied by increased conflict (yellow line) among elite reviewers. Further, the situation becomes more dramatic in $W_{2,4}$, where multiple such transitions take place with one-year gaps.

To summarize, we find representative ways where the ratings from the average and the elite Yelp reviewers can interact in different time scales. Particularly, three common long-term structures seem to emerge: (1) low ratings by many average users, (2) high ratings by many elite users, and (3) sharp disagreement and transitions in the ratings between the average versus elite users. The cause and mechanism of these long-term structures are beyond the scope of this work and are left for future research.

5.4.3 Applications of RCBM

Anomaly detection

An advantage of RCBM is its probabilistic formulation (i.e., Equation 5.12) that assigns a probability to every sample social dynamic. Therefore, a natural application is to detect abnormal social dynamics with extremely-low probabilities. A list of such anomalies detected in Twitter is summarized in Table 5.2, where examples tweets are listed and the corresponding keywords underlined. Similarly, a list of such anomalies detected in Yelp is summarized in Table 5.3, where example reviews are listed with their corresponding ratings (i.e., in the parentheses).

Major Disaster
RT @BreakingNews: BULLETIN – TSUNAMI WARNING ISSUED FOR AMERICAN SAMOA, SAMOA, NIEU, WALLIS-FUTUNA
RT @BreakingNews Tsunami watch iss. for Indonesia, India, Thailand & Malaysia after a powerful 7.9-magnitude earthquake off Sumatra
RT @marcambinder: Breaking: Small plane and helicopter collide over Hudson River in Manhattan. 10-60 (major emergency) declared.
Urgent message
RT @SFChron_alert: Obama declares swine flu a national emergency. http://www.sfgate.com/ZILQ
RT from Iran - If you are outside Iran, change your location / timezone to Iran / Tehran to make it harder to track Iranians
#SaveBalloonBoy Colorado Boy Floats Away In Balloon, Frantic Search Under Way To Rescue Boy http://bit.ly/tsxWI
Major online service shutdown
And the world has come to an end... Gmail is down.
i hate @youtube for wrongfully banninag @ownagepranks account #youtubefail
Machine-generated message
omg!! is it true what they wrote about you in their twit blog? http://lila.twittersblogs.com
EVERYONE!!! Check this new dating site out! Totally Free! talk to mad local chicks that are down for anything! http://local-camz.com
300 new followers in a day - TOTALLY FREE - NO SALE - http://twittertrain.info
Hey everyeone. Just lost 32 lbs in 3 weeks. I wanted to say thanks to Rhonda and her awesome blog. www.rhondasweightloss.com
Hey #JonasOnUstream I LOVE IT (Jonas Brothers live ¿ http://ustre.am/2us4)
CHECK out this site, im a member of it, It gets you more followers: http://TwitTrain.info
Hello! , I just made $842 working a few hours this week from home for Google. You should really check this out! http://bit.ly/u7Rvz
I made an extra $80 today from using tips from http://EARNING-PROFIT3.com
I just took "how sexual are you?" and got: virgin! Try it ? http://bit.ly/zM3kl

Table 5.2: Sample Twitter keywords and tweets that are associated with abnormal social dynamics detected using RCBM. The keywords are underlined.

Adult Entertainment
First time visiting a male-dancing strip club. Never in my wildest dreams did I think I would enjoy this experience as much as I did! (5)
I get accosted and molested by this tall blond Eastern European girl who tried dragging me back to VIP. (1)
I am a strip club aficionado, and this place cannot be beat. (5)
Do you like being crammed into tight spaces and being yelled at by security wherever you stand? (1)
We call it heaven. Real life angels wiggling for our pleasure! (5)
My wife and I dropped $560 here tonight and got almost nothing out of it. (1)
Poor Service / Facility
Terrible customer service, hold times are outrageous, issues are rarely fixed in a timely manner. (1)
No stars, but its forcing me to at least do one star to do this review, worst customer service ever!!!! (1)
The store, the people who go there, the parking lot, the area, it is just all gross. (1)
It's dirty in there, and none of the employees are happy that they have a job. (1)
This mall is sad. You will actually feel bad for this mall. (1)
My son gripped my hand as if we were walking through a haunted house. My wife did the same. (1)
Consistently Outstanding Restaurants
The restaurant exceeded our expectation in both food and service. (5)
It is pricey, but the food and service is always consistently excellent. (5)
Loved everything about this place and was surprised it lived up to the hype. (5)
This place was incredible, and totally lived up to the hype. (5)
Thin Crust Pizza at its best. (5)
From start to finish, from wine to dessert and everything in between, this place lived up to all of my expectations and then some. (5)

Table 5.3: Sample Yelp businesses reviews that are associated with abnormal social dynamics detected using RCBM.

Anomalies in Twitter: The anomalies detected in the Twitter dataset (see Table 5.2) roughly consist of four groups. The anomalies in the first group correspond to major disasters, including the 2009 tsunamis in American Samoa and Indonesia, and the plane crash in Manhattan. The second group of anomalies corresponds to urgent messages, like the national emergency of swine flu and the 2009 Iran election. The property of the first two groups is that they are very contagious and can form large communities very quickly. However, there is relatively little interaction among users compared to other social dynamics with comparable level of contagiousness. The third group of anomalies corresponds to the shutdown or malfunctioning of major online services like gmail or youtube. The forth group are machine-generated messages, which typically correspond to tweets about some marketing promotion. The last two groups have a common characteristic of having a lot of popularity but barely any contagiousness, stickiness, and interactivity compared to typical social dynamics. Finally, we note that detection of all these four groups of anomalies has useful applications. Indeed, for the first three groups, although they happen rarely, detecting them early and being able to respond to them can have a huge impact. For the last group, it is useful in detecting the online scam.

When analyzing these anomalies, a legitimate question is whether these anomalies can be trivially detected by frequency-based rankings. It turns out that, the list in Table 5.2 is very *different* from the one generated by ranking keywords using their frequencies. Indeed, only twelve out of top-100 frequent keywords are considered to be associated with anomalies, which is, equivalently, 27 out of top 500 or 46 out of top 1000. To gain further insights, in Table 5.4, we list some tweets and keywords that are used the most frequently but are associated with normal social dynamics. They can be roughly divided into three groups: holidays, common emotion, and trendy events. It can be observed that each tweet in this table seems to be associated with more organic interactions compared to the cases in Table 5.2.

Anomalies in Yelp: The anomalies detected in the Yelp dataset (see Table 5.3) roughly consist of three groups. The anomalies in the first group correspond to adult entertainment businesses. The property of this group is the strong yet distinct preferences from individual reviewers, some calling it "heaven" and giving five starts, while others saying they "got almost nothing out of it" and give only one

Holiday
Merry Christmas! Anyone staying up to wait for *Santa*?
Happy Thanksgiving to all my friends in the US
Common emotion
#VMAs Taylor Swift is amazing. Kanye is so rude. @taylorswift13, you go girl. I'm proud of you :)
RT@newellhj Oh Nick. You are an idiot. This is why you should have been invited. You just show you're an idiot. #bbcqt
Trendy events
Yankees win! Thaaaaaaaaa Yankees win! #WorldSeries #Champs
Hooray! North Korea pardoned the detained US journalists!
So the balloon landed and the little boy isn't inside?! Where is he?? Ahh!
I wonder if Obama actually wrote this speech because it's really good.
Hi, i'm Madonna. I'm doing a tribute to Michael Jackson in which i ramble about myself the whole time because i am so very classy.
Watching the emmy's
Is it wrong that I cried at the glee finale? I wish it was like when I was in high school.

Table 5.4: Sample Twitter popular keywords and tweets associated with normal social dynamics.

star. Further, these radically different ratings are mixed uniformly in time, which is in sharp contrast to the transitions that present in $W_{1,6}$ of Figure 5.13 or in $W_{2,3}$ and $W_{2,4}$ of Figure 5.14 where each transition takes months or years. The second group of anomalies corresponds to exceptionally poor services or facilities for a prolonged period of time. The property of this group is that they consistently receive the lowest possible ratings from both average and experienced reviewers. While long-term negative ratings from average reviewers are common (i.e., see $W_{2,1}$ of 5.14), this group of anomalies get consistent negative ratings from elite reviewers as well. The third group corresponds exclusively to those restaurants that are constantly outstanding. They receive the highest possible ratings frequently, mostly from average users but also from experienced users. Common words that can be found from the reviews of this group of businesses include "consistently excellent", "lives up the hype", and "exceeds expectation". Moreover, unlike the case of $W_{2,2}$ of 5.14 where positive ratings are given by elite users, these business consistently receive top ratings from average reviewers. Finally, these anomalies, again, cannot be trivially detected using frequency-based rankings. Indeed, less then 15% of these anomalies appears in the lists of top-100 businesses in terms of the numbers of reviews, tips, and checkins. This confirms the advantage of RCBM in detecting anomalies according to their social dynamics, which is based on the common compositional structures learnt directly from a large quantity of unlabeled data.

Discussion about Anomaly Detection: We provide more in-depth discussion about anomaly detection along two lines. First, the type of anomalies that can be detected using RCBM are the ones that do not conform with the model filters (W's), which, by definition, are invariant to magnitude and time shift. In other words, if something is only "abnormal" in magnitude (e.g., extremely-papular social campaign, such as the 2009 Iran-election protect), or in time (e.g., celebrating Christmas in July), it will not be detected. Second, it is possible to preprocess the data with respect to anomaly. For example, a human expert may manually look for social dynamics that he or she considers as anomalies and explicitly remove those dynamics from the training data. This can help prevent the corresponding "abnormal patterns" from entering the model. On the other hand, a possible alternative is to feed the model *only* with the anomalous dynamics, and, at testing time, pick the high-likelihood samples as anomalies. To summarize, there are many things one can do if anomaly detection is the main application, which are not included in this thesis.

Feature extraction for forecasting

When deep learning is used as the unsupervised feature extraction module in Computer Vision and Natural Language Processing [64, 12, 69, 28, 24], it produces state-of-the-art results in various supervised learning tasks. Similarly, we explore RCBM's potential for supervised learning in social applications. For the Twitter dataset, we try to forecast the total number of users of a hashtag; for the Yelp dataset, we aim to forecast the average daily checkins of a business during 2014.

For each dataset, we build a two-level RCBM using a training set. Then, for each testing sample, we obtain its activation vectors using Algorithm 3. To prevent the use of unavailable information during forecasting, for the Twitter dataset, we use all samples up to November 31 as the training set, and all samples in December as the testing set. Also, for each test sample, only the data up to its peak usage time is used. Similarly, for the Yelp dataset, the prediction of the 2014 average checkins are made based on the information up to the end of December 2013. For the prediction models, we use the vector ARMA (VARMA) and Support Vector Regression (SVR) as representative linear and nonlinear models [41], respectively. For features, we use the seven-dimensional features in Figure 5.11 as the baseline, and the RCBM activation vectors in the first level (H1), the second level (H2), and in both levels (H1+H2). To gain further insights, we also use another 1-level RCBM with an equal number of parameters as the two-level RCBM (i.e., with doubled number of filters), and use its activation vectors ($H1^2$) as features.

The results are summarized in Tables 5.5 and 5.6. In general, we observe that SVR performs better than VARMA, whereas using the H1 / H2 / H1+H2 / $H1^2$ features also performs much better than using the baseline features. However, an interesting observation is that using the setting VARMA + H1 + H2 performs better than using the setting SVR + Baseline. It suggests that using activation vectors as features can perform reasonably well even when a simple linear model is used. Moreover, it can be observed that using H1 + H2 performs much better than using H1, H2, or $H1^2$, no matter whether VARMA or SVR is used. This indicates that exploiting compositional features across different time scales using a multi-layer structure is indeed helpful in forecasting social dynamics. We believe this is an important message with a lot of promising applications, and plan to study it in greater details in our future work.

Method	RMSE
VARMA + Baseline	451.1
VARMA + H1	246.7
VARMA + H2	397.6
VARMA + $H1^2$	313.8
VARMA + H1 + H2	**235.9**
SVR + Baseline	397.6
SVR + H1	231.0
SVR + H2	360.4
SVR + $H1^2$	281.5
SVR + H1 + H2	**184.2**

Table 5.5: Forecasting error (via RMSE) of various models and features using the Twitter dataset. The models used include VARMA and SVR, whereas the features used include the raw social dynamics (Baseline), level-1 activation vectors (H1), level-2 activation vectors (H2), and both levels of activation vectors (H1 + H2) of a 2-level RCBM. The forecasting accuracy using a 1-level RCBM with a doubled number of filters is denoted as ($H1^2$).

Method	RMSE
VARMA + Baseline	892.2
VARMA + H1	647.9
VARMA + H2	674.1
VARMA + $H1^2$	655.6
VARMA + H1 + H2	**639.8**
SVR + Baseline	744.5
SVR + H1	584.3
SVR + H2	582.7
SVR + $H1^2$	598.1
SVR + H1 + H2	**536.0**

Table 5.6: Forecasting error (via RMSE) of various models and features using the Yelp dataset. The models used include VARMA and SVR, whereas the features used include the raw social dynamics (Baseline), level-1 activation vectors (H1), level-2 activation vectors (H2), and both levels of activation vectors (H1 + H2) of a 2-level RCBM. The forecasting accuracy using a 1-level RCBM with a doubled number of filters is denoted as ($H1^2$).

Chapter 6

Other Applications

Up to now, we exclusively focused on social dynamics. The methods we propose, however, is general enough to deal with multi-dimensional time series. In this chapter, we demonstrate the generality of our proposed methods using two other (non-social) applications: human behavior modeling and macroeconomics.

6.1 Human Behavior Modeling

Using the human activity dataset mentioned in Section 2.5, we show that the proposed methods, in particular, RCBM, can be applied to design a unified framework for human behavior modeling. This includes several distinct tasks, namely, feature extraction, feature selection, movement prototype mining, activity recognition, and anomaly detection.

6.1.1 Feature Extraction and Selection

The preprocessing steps start from the raw input data consisting of 54 series of sensor reading, generated from nine accelerometer and nine gyroscope deployed over nine places on the subjects' bodies and sampled at 120 Hz. To smooth these readings, we take the average value over $\frac{1}{30}$-second windows.

We then proceed to feature extraction. Following the findings of prior work [56, 20, 80, 21, 27, 40], we extracted various features for each one-second window. We use 33 features, including the mean, standard deviation, 30 FFT coefficients, and energy (i.e., sum of squared FFT coefficients).

Since now the number of input dimensions is $33 \times 54 = 1728$, one potential problem is that most of them may be redundant. Therefore, we reduce the number of dimensions using *multi-task learning* [48]. The formulation of multi-task learning is given as:

$$\arg\min_{W} ||Y - XW||_F^2 + \alpha \sum_{k=1}^{K} ||\mathbf{w}_k||_2. \tag{6.1}$$

Here the $Y \in R^{N \times L}$ denotes the *label matrix* of L labels and N samples; $X \in R^{N \times K}$ denotes the *input matrix* of N samples and K features; W denotes the *weight matrix* of K features and L labels; $\mathbf{w}_k$ denotes the k-th row of W. The idea of Equation 6.1 is to enforce *row-sparsity* to the weight matrix W, such that considering all L labels, only a small subset of rows in W will be non-zero.

In Equation 6.1, the parameter α controls how sparse W is. Consequently, the selection of α has a major impact to feature selection. Here we choose α based on multi-class SVM and cross validation. In particular, we first solve Equation 6.1 using different values of $\alpha \in [1 \times 10^{-8}, 3 \times 10^{-2}]$. Using the result corresponding to each value of α, we select the non-zero features, train multi-class SVM's, and measure the cross-validation accuracy.

The results are summarized in Figure 6.1. It can be observed that there exists a sweet spot at $\alpha \approx 3 \times 10^{-4}$ that achieves the highest cross validation accuracy. Moreover, this value of α corresponds to only 35 non-zero features (out of 1782).

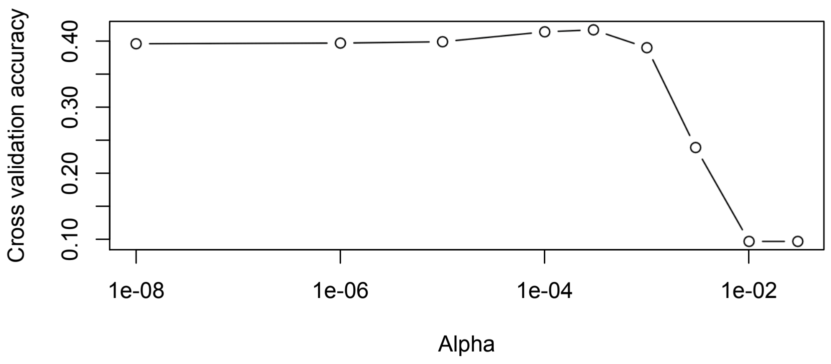


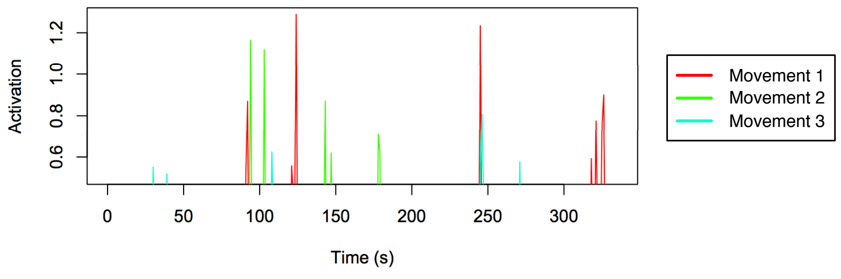
Figure 6.1: Determining the α value in feature selection based on multi-task learning. The best choice is $\alpha = 3 \times 10^{-4}$ (that corresponds to the highest cross-validation accuracy).

Manual inspection shows that the non-zero features are mean, standard deviation, energy, and FFT coefficients that correspond to the lowest and the highest frequencies. In other words, the FFT coefficients other than those corresponding to the lowest and the heist frequencies can be completely ignored. By ruling out these features, we select the 2% of features that are the most related to the activities labels.

6.1.2 Mining Movement Prototypes

We than train a 2-level RCBM following the procedure we described in Section 5.3. Similar to Figure 5.11 that summarizes the dynamical structures for Twitter, we use Figure 6.2 to summarize the top-three most prominent movement prototype of 46 subjects when they were instructed to make brownies.

Figure 6.2 includes these top-three movement prototypes with an activation plot (in the upper panel) indicating their activation strength over time. From the figure, we see that the first movement prototype corresponds to bending over the body, which, according to the activation plot, was activated in four distinct times (i.e., at $t = 87, 123, 248$, and around $t = 325$) during the cooking session. Moreover, we see that



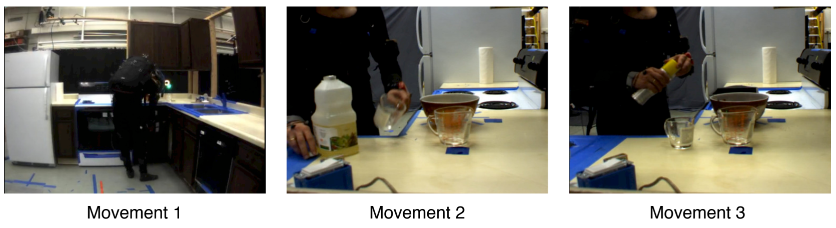


Figure 6.2: The most prominent movement prototypes. Left: bending over. Middle: reaching for appliances. Right: fine motor movements.

the second movement corresponds to reaching out for appliances using the left hand, which was activated in five distinct times (i.e., at $t = 89, 103, 145, 178$, and around $t = 248$). Finally, the third movement corresponds to finer motor movements of the hands, which are activated about six times. We note that the activations of different movements can overlap with one another. For example, movement 1 and movement 2 overlaps around $t = 90$. Manual inspection shows that this is when the subject was bending over while reaching for the oil. For another example, all three movements overlap around $t = 245$. Again, manual inspection shows that this is when the subject was bending over, opening the cabinet, search for the baking tray, and then reaching for it. Through these observations, we can see how mining movement prototypes can be used to analyze complicated movements, i.e., by decomposing them into simpler movement prototypes.

6.1.3 Activity Recognition

Similar to the cases of Table 5.5 and Table 5.6 to forecast Twitter and Yelp dynamics, we now use the activation vectors obtained from RCBM to recognize human activity. In particular, we use multi-class SVM as the classification model under four different sets of features. The baseline uses the 35 features mentioned at the end of Section 6.1.1. Then we use the activation vectors from the first level (H1), the second level (H2), and both levels (H1 + H2) of the trained RCBM as features. The performance of each setting is measured by classification accuracy using five-fold cross-validation.

The results are summarized in the upper part of Table 6.1. On one hand, we observe that using the second-level activation vector (H2) along does not improve the recognition result. On the other hand, the table also shows that using the first-level (H1) or both levels (H1 + H2) of activation vectors can improve the accuracy for activity recognition, by a noticeable margin. This is consistent with our observation in Table 5.5 and Table 5.6, i.e., that careful use of RCBM can improve prediction accuracy.

Method	Accuracy
SVM	41.7
SVM + H1	56.7
SVM + H2	39.1
SVM + H1 + H2	**58.9**
HMM	12.3
1-NN	48.6
3-NN	57.8

Table 6.1: Activity recognition accuracy using multi-class SVM with different features. The features used include the 37 features selected using multi-task learning (Baseline), level-1 activation vectors (H1), level-2 activation vectors (H2), and both levels of activation vectors (H1 + H2) of a 2-level RCBM.

Interestingly, the providers of the same dataset have also conducted activity recognition using various methods [65], including Hidden Markov Model (HMM) and k-Nearest Neighbors (k-NN). Their results are summarized in the lower part of Table 6.1. From the results, we observe that our method outperforms all the methods attempted in [65]. Also, the results they obtained using 3-NN is comparable to the accuracy we achieve. This is somewhat expected because nearest-neighbor-based methods are known as hard to beat; however, they are expensive in prediction time and therefore rarely used in practice.

6.1.4 Anomaly Detection

We next try to detect anomalies during the recorded cooking sessions (i.e., like the case of Section 5.4.3). In particular, we use a 30-second sliding window and calculate the log likelihood of each window using Equation 5.12. We then plot the log likelihood over time and single out the instances when the log likelihood is exceptionally low, which indicate anomalies.

The results are presented in Figure 6.3. The upper panel shows the log likelihood over time, whereas the instances of anomalies are marked using red circles. We then capture the video frames corresponding to these instances, and present those frames in the lower panels of Figure 6.3. From the figure, we can observe two anomalies that do not conform with the common structures of typical cooking sessions. The first anomaly happens at $t = 79$ when the subject received an unexpected phone call and paused the cooking to pick up the phone. Also, the second anomaly happens at $t = 308$ when the subject picked up the detergent and then realized that it is not an ingredient. Evidently, the common trait of these two anomalies is that they are not included in the typical procedure of preparing the designated recipe (i.e., brownies). Using our proposed method, these anomalies can be effectively identified.

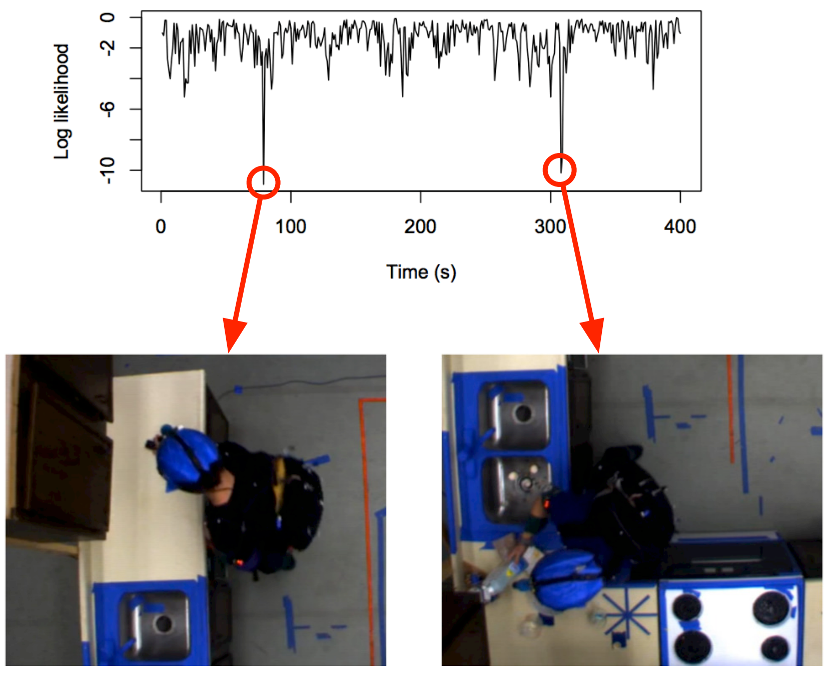


Figure 6.3: Anomaly detection during a cooking session. Left: subject answers to an unexpected incoming phone call. Right: subject picks up detergent and then realizes it is not an ingredient.

6.2 Macroeconomics Pattern Mining

We now switch to the second (non-social) dataset, namely, the macroeconomics dataset described in Section 2.5. The goal is to answer two questions. First, can we identify the underlying structures of economical indices that may affect GDP growth? Second, are these structures that we identify in data-driven way supported by Macroeconomics literature?

6.2.1 Setup

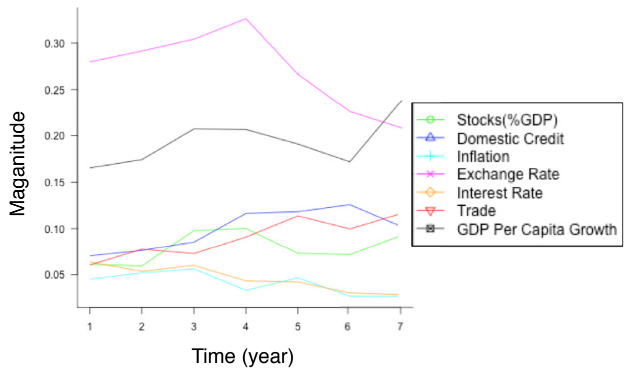
We take seven economical indices (mentioned in Section 2.5) of 12 countries over 22 years and apply RCBM described in Chapter 5. We conduct 20 independent runs and select the top five runs with the best model quality score (i.e., the lowest perplexity). In each of the top five runs, the filters are ranked according to Equation 5.10 in descending order. We then select the representative filters, namely, the top-ranking filters that are consistent among the runs (i.e., the observations which occur only in few runs are ignored).

6.2.2 Microeconomics Patterns

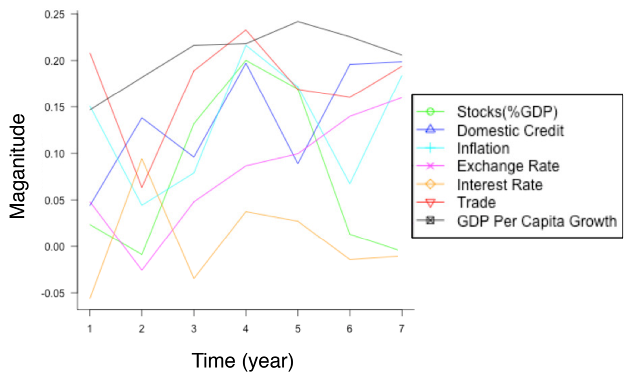
These representative filters are presented in Figure 6.4. The first filter presented in Figure 6.4(a) suggests that the exchange rate correlates to GDP growth. Also, it seems to relate to domestic credit and trade after a couple of timesteps. This behavior is supported in [60], where the author use several indices to measure real exchange rate and derive its relationship with economic growth. The various models used in [60] have also accounted for factors which could affect the relation being derived. The results proved that undervaluation of currency (a high real exchange rate) leads to a growth in the economy. We also note that the author states that, although the results suggest this causality, it is not backed by a generally accepted theory.

The second filter presented in Figure 6.4(b) suggests that the domestic credit is correlated with GDP per capita growth. We find that this observation is supported by [31]. In particular, the authors of [31] defined *bank credit* as the value of loans made by commercial banks and other deposit-taking banks to the private sector divided by GDP. This variable can be used as a measure of the banking development and corresponds to the "domestic credit" variable in our data. In this paper it has been shown that banks have statistically important relationships with future GDP growth. Even after accounting for many factors associated with growth, banking development is both positively and robustly correlated with current and future rates of GDP growth.

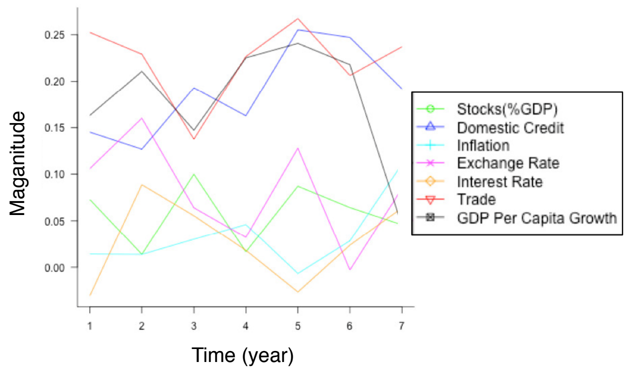
The third filter presented in Figure 6.4(c) suggests that when domestic credit is consistently high, GDP is correlated with stocks and trade. This observation is also supported by [31]. In particular, the authors of [31] put forth ways to measure the stock market and economic growth in order to fit models and find the relationship between the two. The stock market is measured by its size, liquidity, volatility and integration with world capital markets. One of the important measures defined to measure stock market liquidity is the *total value traded*, namely, the value of the trades of domestic shares divided by GDP, which corresponds to the "stock traded" variable in our data. Using per capita GDP growth as a measure of the economic growth, authors statistically show that stock markets facilitate long-term growth even when other factors have been accounted for. This is in accordance with what can be observed in Figure 6.4(c).



(a) Rank 1 - GDP and exchange rate are highly correlated



(b) Rank 2 - Domestic Credit and GDP are highly correlated



(c) Rank 3 - With Domestic Credit high, GDP and Trade are correlated

Figure 6.4: The hidden structures mined from the macroeconomics dataset.

Chapter 7

Data-driven Dynamics Engineering of Social Media

In this chapter, we tackle the data-driven *engineering* of social dynamics. More precisely, given a set of observations from the past, we aim at finding the best short-term intervention that can lead to predefined long-term outcomes. Toward this end, we propose a general formulation that covers two useful engineering tasks as special cases, namely, *pattern matching* and *profit maximization*. By incorporating a deep learning model, we derive a solution using convex relaxation and quadratic-programming transformation. Moreover, we propose a data-driven evaluation method in place of the expensive field experiments. Using a Twitter dataset, we demonstrate the effectiveness of our dynamics engineering approach for both pattern matching and profit maximization, and study the multifaceted interplay among several important factors of data-driven dynamics engineering, such as solution validity, pattern-matching accuracy, and intervention cost.

7.1 Problem Definition

As mentioned in Chapter 2, we use $X \in R^{D \times T}$ to represent the D-dimensional social dynamics corresponding to an information token. As a running example, the dissemination of a Twitter hashtag can be characterized by the evolution of its three types (D=3) of users [52]: *initiators* who bring in information from the outside world, *propagators* who forward the information as it is, and *commentators* who not only forward the information, but also provide their own comments about it. All notations used in this chapter are summarized using Table 7.1.

Dynamics-Engineering Problem (Informal): Given the *observation dynamics* $X \in R^{D \times T_x}$, find the best *intervention dynamics* $U \in R^{D \times T_u}$ such that some desired property of the *outcome dynamics* $V \in R^{D \times T_v}$ is optimized.

The problem is illustrated in Fig 7.1. In the figure, we split the dynamics (that correspond to, e.g., a hashtag) into three parts, X, U, and V, where the current time is at the end of X (i.e., immediately before U). The input of the dynamics engineering problem is X, plus the knowledge of the historical dynamics behavior (e.g., a model). The output of the problem is the best (recommended) intervention dynamics U^*, such some properties of V is optimized. Ideally, we also want to obtain the projected outcome dynamics V^*, i.e., as a result of U^*.

Let us revisit our Twitter example mentioned above. Under this context, an example engineering problem is as follows: Given the observed numbers of the initiators, propagators, and commentators of a particular hashtag within the past 30 minutes (i.e., X in Fig. 7.1), find the best possible intervention dynamics over the next 30 minutes (i.e., U in Fig. 7.1), e.g., using incentive programs or direct promotions, such that the total readership of the hashtag is maximized in the following hours (i.e., V in Fig. 7.1). Note that in the above problem definition, we assume that only the

Variables	
X	observation dynamics; $X \in R^{D \times T_x}$.
U	intervention dynamics; $U \in R^{D \times T_u}$.
V	outcome dynamics; $V \in R^{D \times T_v}$.
Y	$Y = [U\,V] \in R^{D \times T_y}$.
W_{ik}	the k-th filter matrix in the i-th layer; $W_k \in R^{D \times T_w}$.
h_{ik}	the k-th activation vector in the i-th layer; $h_{ik} \in R_+^{T+T_w-1}$.
D	the dynamics dimensionality.
T_x, T_w, etc.	the temporal length of X or W, etc.
σ_i, β_i	parameters of $P(X\|h)$ and $P(h)$, respectively.
K_i	the number of filters in i-th level.
$\mathbf{x}, \mathbf{y}$, etc.	vectorization of X or Y, etc. (Equation 7.1).
$\mathbf{h}_i$	vector concatenation of $\{h_{ik}\}_{k=1}^{K_i}$.
m_x, m_y, etc.	length of $\mathbf{x}$ or $\mathbf{y}$, etc.
B, d	parameters of the score function (Equation 7.2).
C_{cost}, C_{reward}	cost / reward parameters (Equations 7.3 and 7.4)
V_{ref}	pattern to be matched (Equations 7.3)
ρ	tradeoff parameter (Equations 7.3 and 7.4).
c	max-pooling parameter.
$S, \mathbf{s}$	max-pooling dummy variables; $\mathbf{s} = \text{vec}(S)$.
Q, p, A	canonical variables of Quadratic Programming
Special matrices and operations	
I_n	n-by-n identity matrix
$\mathbf{1}_{m \times n}$	n-by-m matrix with 1 in its elements.
$\mathbf{0}_{m \times n}$	n-by-m matrix with 0 in its elements.
	Subscripts might be omitted for simplicity.
$\otimes$	Specialized convolution in Equation 5.2
$\odot$	Kronecker product
$\text{vec}(\cdot)$	vectorization of a matrix

Table 7.1: Summary of notations used in this chapter.

observation (X) is given, while both the ideal intervention (U^*) and the projected outcome (V^*) are to be identified.

Of note, the goal of this work is to find what the best intervention dynamics U^* would *look like* given the input data. In other words, how to actually *implement* a particular intervention (e.g., using incentive programs, etc.) is a separate problem

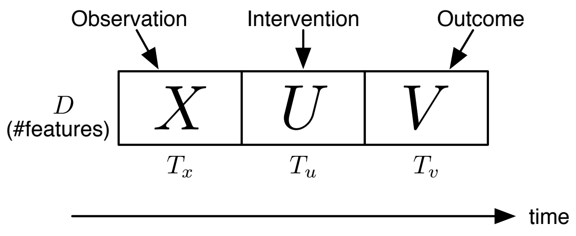


Figure 7.1: Illustration of the data-driven dynamics engineering problem. X: observation dynamics; U: intervention dynamics; V: outcome dynamics.

that is not covered in the present work.

Score function: To define the dynamics engineering problem formally, we first let $Y = [U\ V] \in R^{D\times(T_u+T_v)}$ denote the concatenation of the two matrices U and V. For example, if we have $U = [\begin{smallmatrix}1&2\\3&4\end{smallmatrix}]$ and $V = [\begin{smallmatrix}5&6\\7&8\end{smallmatrix}]$, then $Y = [\begin{smallmatrix}1&2&5&6\\3&4&7&8\end{smallmatrix}]$. Note that this concatenation is merely for mathematical convenience: U and V still differ in their meanings and in the kinds of properties we want their corresponding solutions (U^* and V^*) to satisfy.

Moreover, let $\mathbf{y} = \text{vec}(Y)$ denote its vectorization (i.e., its transformation into a column vector):

$$\mathbf{y} = \text{vec}(Y) = \text{vec}(\begin{bmatrix} -y_1^T- \\ \vdots \\ -y_D^T- \end{bmatrix}) = \begin{bmatrix} y_1 \\ \vdots \\ y_D \end{bmatrix}. \tag{7.1}$$

Using the same example above, we have $vec([\begin{smallmatrix}1&2&5&6\\3&4&7&8\end{smallmatrix}]) = [1\ 2\ 5\ 6\ 3\ 4\ 7\ 8\ ]^T$. Accordingly, we can reformulate the engineering problem as maximizing a *score function* defined as:

$$\text{score}(\mathbf{y}) = \mathbf{y}^T B\mathbf{y} + d^T\mathbf{y}. \tag{7.2}$$

where B and d define the quadratic and linear parts of the score function, respectively. We note that this quadratic score function is general, in the sense that different goals can be achieved using various special cases. In particular, two interesting special cases are:

- *Pattern matching*: To achieve an ideal outcome V_{ref} while minimizing the cost associated with the required intervention U, one can maximize the following score function:

$$\begin{aligned} \text{score}_{\text{match}}(Y) &= -(1-\rho)\|V - V_{ref}\|_f^2 - \rho\langle C_{cost}, U\rangle \\ &= \mathbf{y}^T B\mathbf{y} + d^T\mathbf{y}. \end{aligned} \tag{7.3}$$

  The first term denotes *mismatch* and will force V to match V_{ref}; the second term denotes *cost* and will typically force values in U to be small. Here $\rho \in [0,1]$ controls the relative importance of mismatch versus cost. Moreover, C_{cost} encodes the relative expense of controlling different features at different time, whereas $\langle U, C\rangle = \sum_{ij} U_{ij}C_{ij}$ denotes the dot product between U and C. For example, suppose $C_{cost} = \left[\begin{smallmatrix}1 & 1\\ 2 & 2\end{smallmatrix}\right]$ and $U = \left[\begin{smallmatrix}1 & 2\\ 3 & 4\end{smallmatrix}\right]$, then $\langle C_{cost}, U\rangle = 1\times 1 + 1\times 2 + 2\times 3 + 2\times 4 = 17$. Returning to our Twitter example above, suppose that the first row of $U = \left[\begin{smallmatrix}1 & 2\\ 3 & 4\end{smallmatrix}\right]$ represents the numbers of propagators (i.e., one propagator at $t = 1$ and two propagators at $t = 2$) and that the second row represents the numbers of commentators (i.e., three commentators at $t = 1$ and four at $t = 2$), then assigning $C_{cost} = \left[\begin{smallmatrix}1 & 1\\ 2 & 2\end{smallmatrix}\right]$ is equivalent to specifying that it is twice as expensive to grow the number of commentators (Twitter users who spend time to leave comments) than to control the number of propagators (who simply click "retweet"), regardless of time. Finally, we note that Equation 7.3 is a special case of Equation 7.2. To check this, we can rewrite the second line of Equation 7.3 using $B = (1-\rho)\hat{I}_v^T\hat{I}_v$, $d = \text{vec}([-\rho C_u \;\; 2(1-\rho)V_{\text{ref}}])$, and $\hat{I}_v = I_D \odot ([0_{T_v\times T_u} \;\; I_{T_v}])$. Here $\odot$ denotes the Kronecker product (see Table 7.1 for a summary of notations).

- *Profit maximization*: To maximize the reward associated with the outcome V while minimizing the cost associated with the intervention U, one can maximize the following score function:

$$\begin{aligned} \text{score}_{\text{profit}}(Y) &= -\rho\langle C_{cost}, U\rangle + (1-\rho)\langle C_{reward}, V\rangle \\ &= d^T\mathbf{y}, \end{aligned} \tag{7.4}$$

  The first term denotes *cost* and will typically force the values in U to be small; the second term denotes *reward* and will typically force the values in V to be large. Similarly to the above task, we use C_{cost} to encode the relative cost and use C_{reward} to encode the relative reward of different dimensions and time. Following the above Twitter example, assigning $C_{reward} = \left[\begin{smallmatrix}1 & 3\\ 1 & 3\end{smallmatrix}\right]$ is equivalent to specifying that it is three times more rewarding to acquire a user (either a propagator or a commentator) at $t = 2$ compared to acquiring a user at $t = 1$, regardless of the type of the user. Like the case of Equation 7.3, ρ controls the relative importance of cost versus reward. We note that Equation 7.4 is another special case of Equation 7.2. To check this, we can rewrite the second line of Equation 7.4 using $d = \text{vec}([-\rho C_{cost} \;\; (1-\rho)C_{reward}])$.

Formal Definition: While maximizing the score function, we make two implicit assumptions: (1) there exists a temporal dependency among X and $Y = [U\ V]$, and (2) the solution we come up with needs to follow that dependency. Accordingly, we propose the following formal definition of our problem:

Dynamics-Engineering Problem (Formal): Given observation X, a probabilistic model $P(\cdot)$, and a score function score(Y), find:

$$Y^* = [U^*\ V^*] = \arg\max_{Y}\ \log P(Y|X) + \lambda\ \text{score}(Y). \tag{7.5}$$

Here $P(\cdot)$ denotes the log-likelihood using a probabilistic model that captures the temporal dependencies of the social dynamics. In other words, while the second term (i.e., the score function) takes care of the specific engineering task, the first term (i.e., $\log P(Y|X)$) makes sure that the solution still conforms with the temporal dependency of the social dynamics. Moreover, $\lambda \geq 0$ is a balancing parameter that controls the relative importance of fitting the probability distribution $P(\cdot)$ versus maximizing the score. Of note, the selection of λ is crucial and will be described in detail later.

We note that our proposed problem definition is *general* yet *precise*. Indeed, it can incorporate any combination of $P(\cdot)$ and score(Y) functions, in which any different combination corresponds to a different engineering task. Also, once this combination is given, the engineering problem is mathematically precise.

7.2 RCBM-based Formulation

In principle, any probabilistic model of social dynamics can be plugged into the likelihood term $P(\cdot)$ in Equation 7.5. In this work, we use RCBM that we proposed in Chapter 5. As it will be shown in the experimental section, the choice of this model makes a big difference.

By writing down the conditional probability $P(Y|X)$ using the joint probability specified in Equation 5.12 and then plugging $P(Y|X)$ into the first term of Equation

7.5, the optimization problem in Equation 7.5 can be explicitly written as:

$$\begin{aligned}
\underset{Y,\, h_1,\, h_2,\, S}{\arg\min} \quad & \tfrac{1}{2}\|[X\ Y] - \textstyle\sum_k W_{1k} \otimes h_{1k}\|_F^2 + \tfrac{\sigma_1^2}{\beta_1} \textstyle\sum_{k=1}^{K_1} \|h_{1k}\|_1 \\
& + \tfrac{1}{2}\|S - \textstyle\sum_k W_{2k} \otimes h_{2k}\|_F^2 \\
& + \tfrac{\sigma_2^2}{\beta_2} \textstyle\sum_{k=1}^{K_2} \|h_{2k}\|_1 - \lambda\left(\mathbf{y}^T B \mathbf{y} - d^T \mathbf{y}\right) \\
\text{s.t.} \quad & S = \mathrm{MP}(\mathbf{h}_1), h_{1k} \geq 0,\ h_{2k} \geq 0 \text{ and } \mathbf{y} \geq 0.
\end{aligned} \tag{7.6}$$

Here, a two-level RCBM is presented for illustration purposes, though the optimization formulation for a multilevel RCBM can be similarly derived. The max-pooling operation MP$(\cdot)$ is defined as

$$\mathrm{MP}(\mathbf{h}_1)[k,t] = \max_{i \in 1,\, \ldots,\, c} h_{1k}[(t-1)c + i], \tag{7.7}$$

where $\mathbf{h}_1$ is the vector concatenation of $\{h_{ik}\}_{k=1}^{K_i}$. As mentioned in Section 3.2, MP$(\cdot)$ is the key that enables RCBM (or more generally, any convolutional deep-learning method) to learn the nonlinear features of the series. However, it also imposes significant difficulties in optimization by making the problem non-differentiable and non-convex. Consequently, the problem in Equation 7.6 is not only difficult to solve, but also prone to getting stuck at suboptimal solutions.

7.3 Solution based on Convex Relaxation and QP Transformation

7.3.1 Convex Relaxation

To solve the difficulty resulted from max-pooling, we propose the following convex relaxation:

$$\begin{aligned}
\underset{Y,\, h_{1k},\, h_{2k},\, S}{\arg\min} \quad & \tfrac{1}{2}\|[X\ Y] - \textstyle\sum_k W_{1k} \otimes h_{1k}\|_F^2 + \tfrac{\sigma_1^2}{\beta_1} \textstyle\sum_{k=1}^{K_1} \|h_{1k}\|_1 \\
& + \tfrac{1}{2}\|S - \textstyle\sum_k W_{2k} \otimes h_{2k}\|_F^2 + \tfrac{\sigma_2^2}{\beta_2} \textstyle\sum_{k=1}^{K_2} \|h_{2k}\|_1 \\
& -\lambda(y^T B y + d^T y) \\
\text{s.t.} \quad & h_{1k} \geq 0,\ h_{2k} \geq 0 \text{ and } Y \geq 0, \\
& h_{1k}[(t-1)c+i] \leq S[k,t] \\
& S[k,t] \leq \textstyle\sum_{i=1}^{c} h_{1k}[(t-1)c+i].
\end{aligned} \tag{7.8}$$

The idea behind this relaxation consists of introducing a new variable S as the surrogate of MP$(\cdot)$. Furthermore, we substitute the *equality* constraints specified in Equation 7.7 with two sets of *inequality* constraints, i.e.,

$$\max_{i \in 1,\, \dots,\, c} h_{1k}[(t-1)c+i] \leq S[k,t] \leq \sum_{i=1}^{c} h_{1k}[(t-1)c+i].$$

In other words, instead of forcing S to equate $S[k,t] = \max_{i \in 1,\, \dots,\, c} h_{1k}[(t-1)c+i]$, i.e., the maximal value among the consecutive c values, we now constrain it to be larger than or equal to the maximal value, but smaller than or equal to the sum of those c values.

We note that the problem in Equation 7.8 is now *jointly* convex in Y, h_1, h_2 and S, since the objective function is convex and all constraints are linear. Moreover, since the objective is differentiable, a possible approach to solve Equation 7.8 is using the proximal method [6]. It turns out, however, the projection functions corresponding to the constraints in Equation 7.8, which are required in the proximal method, are difficult to derive. Therefore, we propose an alternative solution based on a quadratic programming (QP) transformation.

7.3.2 QP transformation

Since the objective function of Equation 7.8 is quadratic with only linear constraints, in principle, there exists a QP problem that is equivalent to Equation 7.8. The explicit form of this QP transformation is given by the following theorem.

Theorem 6. *Let $z = [\mathbf{y}^T\ \mathbf{h}_1^T\ \mathbf{h}_2^T\ \mathbf{s}^T]^T$, then the quadratic programming problem*

$$\begin{array}{ll} \arg\min_z & \frac{1}{2} z^T Q z + p^T z \\ \text{s.t.} & Az \geq 0, \end{array}$$

is equivalent to the problem in Equation 7.8, where

$$\begin{aligned}
Q &= \begin{bmatrix} I_{m_y} - 2\lambda B & -\hat{I}_Y G_1 & \mathbf{0} & \mathbf{0} \\ -(\hat{I}_Y G_1)^T & G_1^T G_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & G_2^T G_2 & -G_2 \\ \mathbf{0} & \mathbf{0} & -G_2^T & I_{m_s} \end{bmatrix} \\
A &= \begin{bmatrix} I_{m_y} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{m_{h_1}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{m_{h_2}} & \mathbf{0} \\ \mathbf{0} & -I_{m_{h_1}} & \mathbf{0} & M \\ \mathbf{0} & M^T & \mathbf{0} & -I_{m_s} \end{bmatrix} \\
p &= [-\lambda d^T, \tfrac{\sigma_1^2}{\beta_1} \mathbf{1}_{1\times m_{h_1}} - ((\hat{I}_X G_1)^T \mathbf{x})^T, \tfrac{\sigma_2^2}{\beta_2} \mathbf{1}_{1\times m_{h_2}}, \mathbf{0}_{1\times m_s}].
\end{aligned}$$

As mentioned before, bold-faced letters (e.g., $\mathbf{y}$ and $\mathbf{s}$) denote the vectorization of matrices with the same capital letters (e.g., Y and S), while $\mathbf{h}_i$ denotes the vector concatenation of $\{h_{ik}\}_{k=1}^{K_i}$ for the i-th level. In the above theorem, we define $\hat{I}_X = I_D \odot [I_{T_x}\ \mathbf{0}_{T_x \times T_y}]$ and $\hat{I}_Y = I_D \odot [\mathbf{0}_{T_y \times T_x}\ I_{T_y}]$, where $\odot$ is the Kronecker product. Also, G_i denotes the matrix representation of the convolution operations $\sum_k W_{ik} \otimes h_{ik}$, which is defined as:

$$G_i = \begin{bmatrix} G_{11}^{(i)} & G_{12}^{(i)} & \dots & G_{1K_1}^{(i)} \\ \vdots & \vdots & \vdots & \vdots \\ G_{D1}^{(i)} & G_{D2}^{(i)} & \dots & G_{DK_1}^{(i)} \end{bmatrix}$$

where $G_{lk}^{(i)}$ is the Toeplitz matrix of the l-th column of the filter matrix W_{ik} [55]. Finally, $M \in R^{m_{h1} \times m_s}$ is defined as $M_{i,j} = 1$ if $j = \lceil i/c \rceil$ and $M_{i,j} = 0$ otherwise. The details and proof of this theorem can be found in Appendix C[1].

Since this proposed quadratic programming problem is convex, it is guaranteed to find an approximate solution in polynomial time. In our experiments, the vector z consists of around 1000 variables and even so the problem gets solved in just a few seconds.

7.4 Data-driven evaluation

For many methods in modeling and prediction, cross-validation [74] is the standard way for evaluating solutions and selecting parameters. However, cross-validation cannot be directly applied to our data-driven dynamics engineering problem, because the properties of a "good solution" for modeling and prediction is well-known. For example, a good modeling solution will have high data likelihood and a good prediction

[1]The author gratefully acknowledge Dr. Hao-Chih Lee's help in the proving the theorems in Appendix C

solution will be highly accurate. For our dynamics-engineering problem, however, such a property is less obvious.

For the dynamics-engineering problem, we argue that the key property of a good solution consists of the *combination* of a high score and a high *validity*, where the latter can be roughly defined as how well the solution is supported by historical samples that achieve high scores. To show that having a high score alone is not sufficient, consider the case when $\lambda \to \infty$ in Equation 7.5. In this case, the optimization will produce the highest possible score, while completely ignoring the likelihood term in Equation 7.5. As a result, the optimization will produce a solution that do not possess any inherent temporal dependency of the data. In this case, the projected outcome V^* would be unlikely to happen in the real world even if the suggested intervention U^* is implemented.

7.4.1 Solution Validity

As mentioned above, the informal definition of validity is how well the solution is supported by historical samples that achieve high scores. To formally define *validity* γ, two important components are: (1)$\hat{P}$ that denotes the density function capturing what the high-scoring dynamics look like in historical data, and (2) P_0 that denotes a carefully chosen threshold. More precisely, $\hat{P}(\cdot)$ and P_0 are constructed in four steps:

1. Evaluate the score function using all historical data and keep the N_0 top-scoring samples.

2. Use the first half $\{[X_i, Y_i]\}_{i=1}^{\lfloor \frac{N_0}{2} \rfloor}$ to construct a kernel density estimator [75]:

$$\hat{P}(X, Y; h) \propto \sum_i \exp \frac{||[X, Y] - [X_i, Y_i]||^2}{-2\omega^2}.$$

3. Use the second half $\{[X_i, Y_i]\}_{i=\lfloor \frac{N_0}{2} \rfloor + 1}^{N_0}$ to choose the value of ω that has the highest data likelihood.

4. Use the second half to calculate P_0, such that only a small fraction (e.g., 5%) of samples among the second half has $\hat{P}(X, Y; h) < P_0$.

With $\hat{P}(\cdot)$ and P_0 defined, we can define the *validity* γ of a solution as:

$$\gamma(\lambda) = \log \frac{\hat{P}([X, Y^*(\lambda)])}{P_0}. \tag{7.9}$$

Then we can use γ as a convenient measure, such that $\gamma \geq 0$ indicates that, according to historical high-scoring data, the solution is "realistic enough".

We note that in the construction of γ, and in particular, $\hat{P}(\cdot)$ and P_0, we do not use the entire training set. The underlying reason is that a realistically good solution can be very rare. In other words, it is by design that validity should measure how well the solution is supported by historical samples *that achieve high scores*, instead of historical samples in general.

7.4.2 Parameter Determination

With validity defined, we are now ready to select λ. As mentioned before, it should be the combination of high validity and high score. A key observation from Equation 7.5 is that one can make the score larger by making λ larger. Therefore, while there may be many potential ways to do it, we propose the following method:

$$\begin{aligned} &\arg\max_\lambda \quad && \lambda \\ &\text{s.t.} && \gamma(\lambda) \geq 0, \end{aligned} \tag{7.10}$$

where the idea is that conditioned on the solution being (sufficiently) valid, we want its score to be as high as possible. Finally, we use a Twitter dataset to demonstrate the interplay among λ, validity (γ), and score while engineering social dynamics in the next section.

7.5 Experimental Results

For experimental results, we first describe our dataset, the overall setup, and two baseline methods. Then, we present experimental results on two engineering tasks: pattern matching and profit maximization.

7.5.1 Dataset

Based on the Twitter dataset mentioned in Chapter 2.5, we use hashtags to enumerate the information tokens that carry social dynamics. We filter out "low-traffic" hashtags by selecting only the ones with at least 100 total usages around the 90 minutes during their peak times, yielding a 10K-sample dataset of social dynamics. We then sort these samples according to their peak time. The first 9K samples are used as the training set, i.e., for model training and the construction of $\hat{P}(\cdot)$ and P_0 (mentioned in Section 3.6), whereas the remaining 1K samples are reserved for testing. For all hashtag samples, we measure the dynamics in units of 3 minutes, where the first 30 minutes are the observation dynamics (X), the middle 30 minutes are the intervention dynamics (U), and the last 30 minutes are the outcome dynamics (V).

We characterize each social dynamic using its five types of users [52]. *Initiators* denote the users who use this keyword before any of his or her friends did. *First-time propagators* and *first-time commentators* denote the users who retweet and tweet, respectively, about this keyword after his or her friends using the same keyword before. *Recurring propagators* and *recurring commentators* denote the users who retweet and tweet, respectively, the same keyword that they used before.

7.5.2 Experimental Setup

We conduct experiments on two types of engineering tasks: pattern matching (Equation 7.3) and profit maximization (Equation 7.4). For pattern matching, we set $C_{cost} = \mathbf{1}_{D\times T_u}$ in Equation 7.3 to assume a uniform intervention cost in time and for different types of users. Similarly, for profit maximization, we set $C_{cost} = \mathbf{1}_{D\times T_u}$ and $C_{reward} = \mathbf{1}_{D\times T_v}$ in Equation 7.4.

In order to analyze the interplay and tradeoffs critical to real-world engineering applications, for each task, we conduct analyses along the following four directions:

1. Interplay between validity γ (Equation 7.9) and the optimization parameter λ (Equation 7.5).

2. Tradeoff of individual terms in the score functions. In particular, for pattern matching (Equation 7.3), it includes cost ($< C_{cost}, U >$) and mismatch ($||V - V_{ref}||_f^2$); for profit maximization (Equation 7.4), it includes cost ($< C_{cost}, U >$) and reward ($< C_{reward}, V >$).

3. Comparison between "real" vs. engineered cases, assuming that what happened in the real-world represents the "unconscious optimization" of the score functions. The motivation behind this analysis is to demonstrate the potential benefits as a result of purposeful engineering.

4. A case study that contrasts the "real" vs. engineered dynamics.

7.5.3 Baseline Methods

AR: Our first baseline is to substitute the likelihood term in Equation 7.5 with another one using the Autoregressive model (AR). AR is commonly used in time-series forecasting and is defined as:

$$x_t = \sum_{i=1}^{p} \Phi_i x_{t-i} + \epsilon_t \tag{7.11}$$

Here $x_t \in R^{D\times 1}$ denotes the multivariate features at time t; $\epsilon_t \sim \mathcal{N}(0,\ \Sigma)$ denotes the i.i.d. multivariate Gaussian noise; Φ_i's denote the matrices for modeling the dependency between the current dynamics and its history back to p steps, where we set $p = 10$. Details of solving Equation 7.5 with the first term using AR is given in the Appendix C. While this baseline fits perfectly in our proposed framework, its restrictive linear generative model may limit its performance.

NN: Our second baseline is based on the nearest-neighbor (NN) search. The idea is to search within the training data for the top 5% samples that are the most similar to the given observation X (using Euclidean distance). Then the solution Y^* is obtained using the $\{U, V\}$ part of the highest-scoring sample within that subset. The advantage of NN is that, unlike other methods, it doesn't have a concern about validity, i.e., whether the solution is realistic or not, because the solution is generated from real dynamics that happened in the past. However, its disadvantage is that not all historical dynamics matches the observation X and maximizes the score function at the same time. Consequently, the score of NN's solution may be low or unstable.

7.5.4 Experiment 1: Pattern Matching

In our first experiment, i.e., pattern matching, we are given the observation X_i of every test sample and we aim at matching a single V_{ref}. This V_{ref} is defined as the average outcome dynamics of the top 2% samples in the training set with the highest long-term popularity $||V||_1$. We conduct data-driven dynamics engineering using all test samples and analyze the resulting validity, cost, and mismatch.

Validity (γ) vs. λ

In Figure 7.2, we analyze the effect of different values of λ on the average solution validity γ. The dotted horizontal lines marks $\gamma = 0$, above which the solution is considered valid. Note that NN is not included here since it doesn't require the selection of λ. From the figure, the observations are twofold. First, there is indeed a range of λ that produces valid solutions. In particular, that range changes with the value of ρ: the lower ρ is, the larger the range is. This is because a lower ρ puts more emphasis on minimizing mismatch instead of cost (see Equation 7.3). While there is nothing unrealistic about the pattern that needs to be matched, matching it using an extremely low cost (i.e., using a large ρ) can be unrealistic. Second, the proposed method outperforms the AR baseline in terms of validity, since the results using the proposed method have a lot more cases above the dotted line (indicating $\gamma \geq 0$) compared to AR. This is because the proposed method incorporates RCBM that can effectively capture non-linear features, whereas AR is a linear model.

Cost-mismatch Tradeoff

In Figure 7.3, we further analyze the tradeoff of using different values of ρ where λ is selected using Equation 7.10. In cases when there is no λ that satisfies $\gamma(\lambda) \geq 0$,

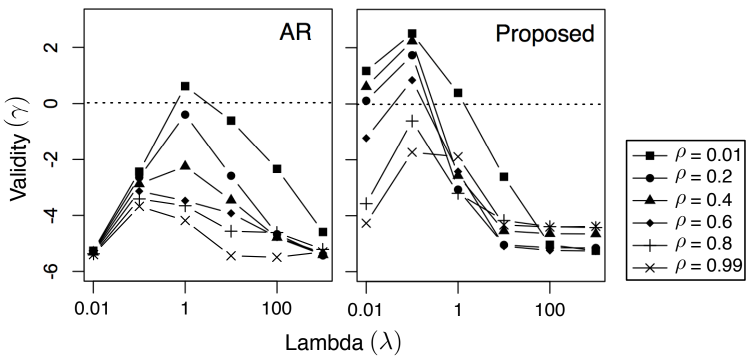


Figure 7.2: Influence of λ on solution validity γ using different values of ρ. The dotted lines mark $\lambda \geq 0$, above which the solution is considered valid.

we select $\arg\max_\lambda \gamma(\lambda)$ instead. The results of average cost versus mismatch using all three methods are summarized in Figure 7.3. For each method, the point in the lower-right corner corresponds to the case of $\rho = 0.01$, whereas the point in the upper-left corresponds to the case of $\rho = 0.99$. From the figure, the newly proposed method consistently makes the best tradeoff: with the same mismatch, it achieves a lower cost; with the same cost, it achieves the lower mismatch. The reason for this is twofolds: for AR, its linear model is too restrictive to reach either of the two objectives; for NN, the samples in the subset of training data that matches the given observation does not necessarily have a high score. Another interesting observation is that NN seems to make better tradeoffs compared to AR. This shows that the selection of a good generative model is crucial for data-driven dynamics engineering.

Constrained Cost Minimization

In order to demonstrate the potential benefits of purposeful engineering, we use a slightly different setting. While for each test sample i, we are still given the observation part X_i, we set $V_{ref} = V_i$, i.e, its own outcome dynamics. This setting allows us to compare the performance of the matching algorithms, in terms of cost, with what

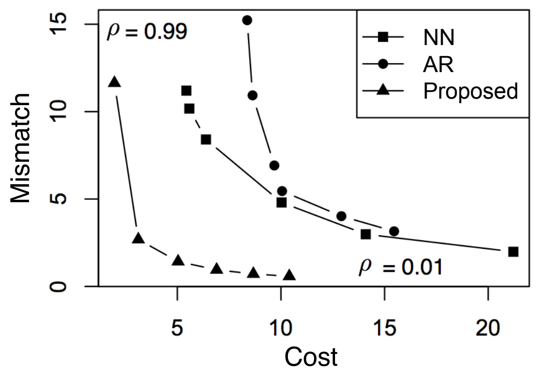


Figure 7.3: Tradeoff between cost and mismatch using different values of ρ. For each method, the point in the lower-right corner corresponds to the case of $\rho = 0.01$, whereas the point in the upper-left corresponds to the case of $\rho = 0.99$.

actually happened in reality, assuming that each test sample was actually performing a (perfect) matching task without consciously considering minimizing the cost.

Since the *real* case achieves a "perfect match", we need to constrain the engineering algorithms such that they can be compared on the same footing. Therefore, we enforce an additional constraint $||V^* - V_{ref}||_1 \leq pDT_v$ where $p = 5\%$. In other words, after going through every test sample, each algorithm will have its own fraction of valid answers, and only the valid answers will be compared to the same set of samples, in terms of cost, to the real case. For AR and the proposed method, a valid answer must satisfy this constraint *on top of* satisfying $\gamma \geq 0$.

Figure 7.4 summarizes the results where the fraction of valid answers are annotated at the top and the mean values are marked using red crosses. From the figure, we note that NN produces valid answers for 41% of the test samples, whereas the number is 9% for AR and 98% for the proposed method. Also, the mean cost among the valid solutions using NN is 4.34, compared to 4.78 for AR and 2.31 for the proposed method. In other words, the proposed method achieves not only the largest fraction of valid solutions, but also the lowest average cost for that larger fraction.

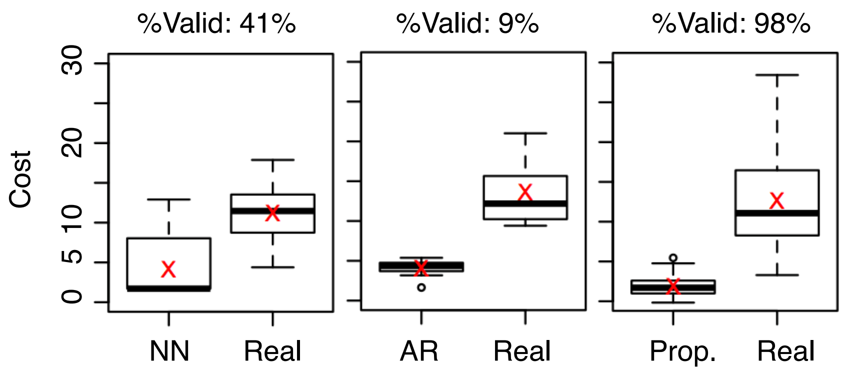


Figure 7.4: Cost distribution of valid answers using different methods. The cost distribution of each method is contrasted with that of the same set of samples in the real case. The mean values are marked using red crosses.

Note that the cost produced by NN has a high variation, confirming our expectation in Section 4.3. Finally, they all achieve lower cost than the corresponding samples in the real case, which is somewhat expected because the real cases were not consciously minimizing the cost. This further highlights the cost-saving potential of these dynamics-engineering methods.

Case Study

To gain further insights, we pick a test case where all three methods produce valid solutions from the experiment of Figure 7.4 and plot their suggested solutions in Figure 7.5. For AR and the proposed method, since their solution only cover the last 60 minutes, their first 30 minutes are copied from the real case. From the real case, we see that it is a rather sustained dynamics that seems to be full of interactions among different types of users. To compare among different solutions (i.e., NN, AR, and Proposed), we note that the ideal pattern-matching should achieve both low cost during $t \in [30, 60]$ and low mismatch during $t \in [60, 90]$.

The solution produced by NN, although seems to match the real case in its general shape, it produces a moderate mismatch. Further, the cost of its suggested intervention is the highest among the three. AR, on the other hand, produces a very smooth dynamics that does not match the general shape of (the third part of) the real case, although the mismatch is quantitatively comparable to that of NN. Moreover, although its cost is relatively low, the dynamics doesn't look real: in fact, its solution validity γ is 0.02, i.e., barely passes 0.

Finally, the proposed method produces a recommendation that best matches the third part of the real case, while also producing the lowest-cost intervention. A closer inspection shows that although the magnitude of the intervention dynamics (i.e., the second part) is generally low, it seems to consciously keep an interesting proportion and interaction among different types of users. This is because the proposed recommendation explicitly use the patterns (i.e., the filters W's in Equations 7.6 and 5.12) of different temporal scales that are learnt directly from data. Consequently, the proposed method is able to recommend low-cost, good-matching solutions while still making the suggested dynamics follows the intrinsic temporal dependencies from the data.

7.5.5 Experiment 2: Profit Maximization

In our second experiment, i.e., profit maximization, we are given the observation X_i of every test sample and aim at maximizing the long-term popularity (reward) $||V||_1$ with minimum cost $||U||_1$. Again, we conduct data-driven dynamics engineering using all test samples and analyze the resulting validity, cost, and reward.

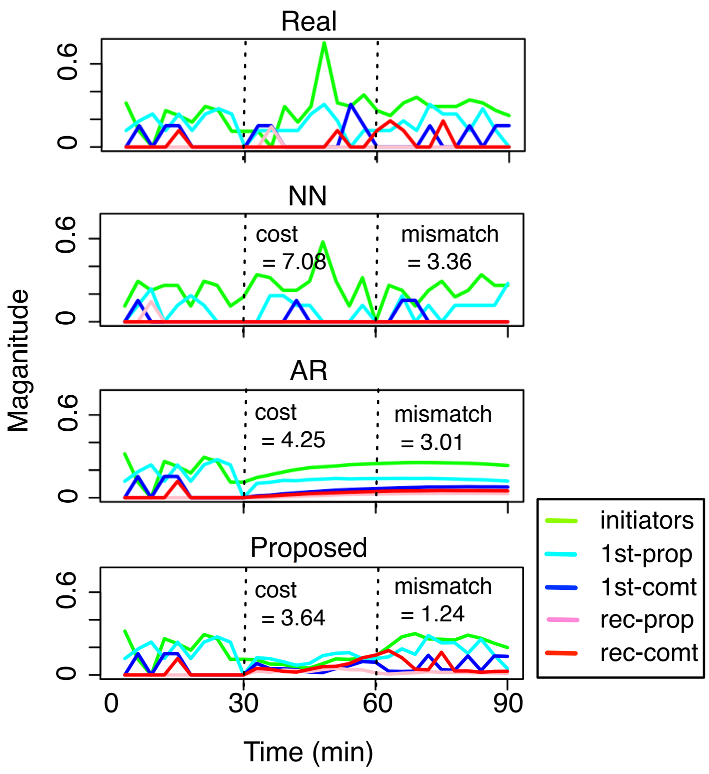


Figure 7.5: Case study: real versus suggested dynamics using different methods. A good solution is characterized by low cost (during $t \in [30, 60]$) and low mismatch (during $t \in [60, 90]$). The X-axis denotes time (in minutes) and the Y-axis denotes the normalized number of different types of users.

Validity (γ) vs. λ

In Figure 7.6, we present the effects of different values of λ on the average solution validity γ, where the dotted horizontal lines marks $\gamma = 0$ (above which the solution is considered valid). There are two observations in Figure 7.6 that are consistent with Figure 7.2. First, there is a range of λ that produces valid solutions; the lower ρ is, the easier to produce valid solutions. Since a lower ρ puts more emphasis on reward instead of cost (see Equation 7.4), it suggests that the key to produce good solutions is putting a low (numerical) weight on cost. Second, the proposed method outperforms the AR baseline in terms of validity, indicating that the proposed method produces a

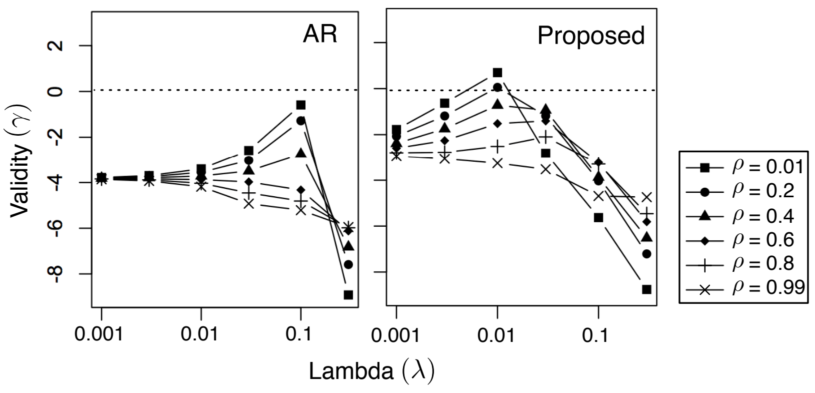


Figure 7.6: Influence of λ on solution validity γ using different values of ρ. The dotted lines mark $\lambda \geq 0$, above which the solution is considered valid.

lot more valid cases ($\gamma \geq 0$) compared to AR. This confirms that the proposed method incorporates RCBM that can effectively capture non-linear features, whereas AR is a linear model.

Interestingly, there are also three observations in Figure 7.6 that are different from that of Figure 7.2. First, the validity value γ is generally smaller, indicating that as a task, profit maximization is more challenging than pattern matching. Second, the best λ's that correspond to the highest γ's are also about $10X$ smaller than that of Figure 7.2. It suggests that in profit maximization, one must put more emphasis on the likelihood instead of the score in Equation 7.5. Finally, the range of λ that is above zero ($\lambda \approx 0.01$) is much more narrow than the case of Figure 7.2 ($\lambda \in [0.01, 1]$). This confirms that the task of profit maximization is more challenging than pattern matching.

Reward-cost Tradeoff

In Figure 7.7, we further analyze the tradeoff of using different values of ρ where λ is selected using Equation 7.10. Again, in cases when there's no λ that satisfies

$\gamma(\lambda) \geq 0$, we select $\arg\max_{\lambda} \gamma(\lambda)$ instead. The results of average reward versus cost using all three methods are summarized in Figure 7.7. For each method, the point in the upper-right corner corresponds to the case of $\rho = 0.01$, whereas the point in the lower-left corresponds to the case of $\rho = 0.99$. We note that, in general, the proposed method provides, again, that best overall tradeoff compared to NN and AR, which confirms that the selection of a good generative model is crucial for data-driven dynamics engineering. This is also consistent with the observations in Figure 7.3.

Interestingly, there are also two observations in Figure 7.7 that are somewhat different from the case of Figure 7.3. First, while NN seems to be slightly better than AR in Figure 7.3, it is significantly better in the case of Figure 7.7. It indicates that, due to the increased difficulty of profit maximization (compared to pattern matching), AR becomes no longer useful. Second, the reward produced by the proposed method is comparable to that of NN, although the proposed method requires much less cost. This suggests that in profit maximization, reducing cost is much easier than increasing reward. This also makes intuitive sense: while it is hard to beat the "Ice Bucket Challenge" in popularity, it might be possible to engineer its marketing campaign such that the cost can be reduced.

Constrained Reward Maximization

To demonstrate the potential benefits of purposeful engineering, we now use a slightly different setting. While for each test sample i, we are still given the observation part X_i, we enforce an additional constrain that the a solution must produce a cost that is at most *half* of the *actual* cost of sample i, i.e., $||U_i||_1$, on top of achieving $\gamma \geq 0$, to be considered a *valid* answer. This setting allows us to compare the performance of the profit-maximization algorithms, in terms of reward and cost, with what actually happened in reality, assuming that each test sample was actually performing a profit-

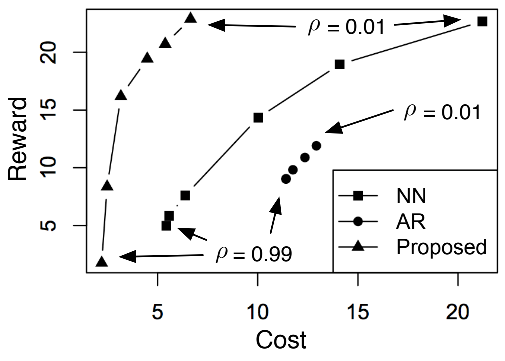


Figure 7.7: Tradeoff between cost and mismatch using different values of ρ. For each method, the point in the lower-right corner corresponds to the case of $\rho = 0.01$, whereas the point in the upper-left corresponds to the case of $\rho = 0.99$.

maximization task, unconsciously.

Figure 7.8 summarizes the results; again, the fraction of valid answers are annotated at the top and the mean values are marked using red crosses. From the figure, we can make three observations. First, the fractions of valid samples are significantly lower than the case of Figure 7.4. Indeed, NN produces valid answers for 36% of the test samples, whereas the number is 6% for AR and 44% for the proposed method. This confirms that profit maximization is, in some sense, harder than pattern matching. Second, while all valid solutions from each of the three methods have an average cost lower than half of the corresponding real cases (per our experimental design), these methods result in different reward distributions. Indeed, AR, NN, and the proposed method produce lower, comparable, and higher rewards compared to the real cases, respectively. This confirms that the proposed approach outperforms the two baseline methods and further highlights the profit-maximization potential of the proposed dynamics-engineering method.

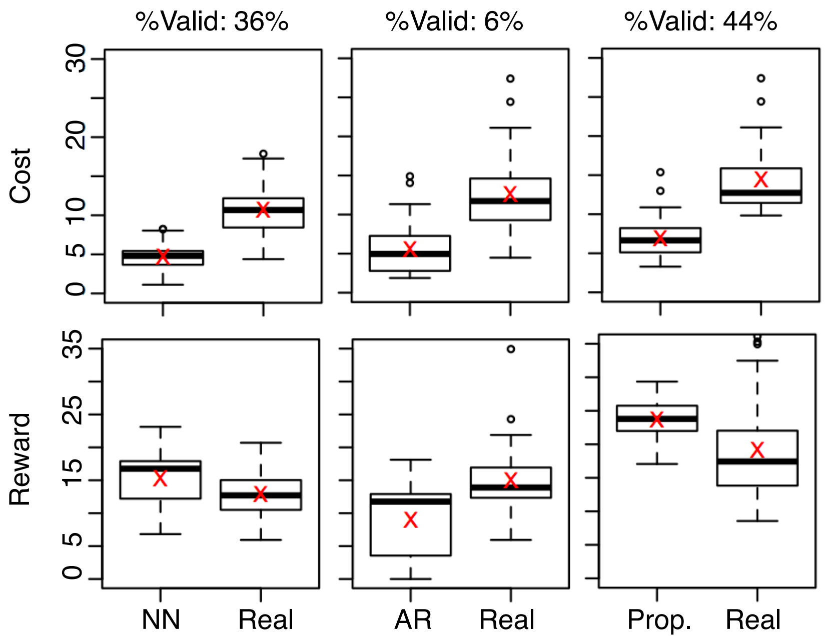


Figure 7.8: Cost distribution of valid answers using different methods. The cost distribution of each method is contrasted with that of the same set of samples in the real case. The mean values are marked using red crosses.

Case Study

To gain further insights, we pick a test case where all three methods produce valid solutions from the experiment of Figure 7.8 and plot their suggested solutions in Figure 7.9. All settings remain the same as the case of Figure 7.5. To compare among different solutions (i.e., NN, AR, and Proposed), we note that the ideal profit maximization should achieve both low cost during $t \in [30, 60]$ and high reward during $t \in [60, 90]$.

From Figure 7.8, we can see that, although the solutions from all three methods (NN, AR, and the proposed method) have costs lower than half of the real case, their rewards are quite different. For NN, the reward of its solution is comparable to that of the real case. Given that it also has a lower cost compared to the real case, this solution is not too bad. For AR, while the reward is even lower, the real issue is that the solution dynamics doesn't look real: in fact, its solution validity $\gamma \approx 0.004$, i.e., barely passes 0. Finally, the proposed method not only produces a recommendation that has a low cost, but also a reward higher than the real case. A closer inspection shows that although the magnitude of the intervention dynamics (i.e., the second part) is generally low, it seems to contain interesting interactions because the recommendation includes different key roles at different stages: recurring commentators (red) around $t = 35$, first-time propagators (dark blue) around $t = 50$, and then the dominating first-time propagators after $t \geq 60$. All these interactions reflect the patterns (i.e., the filters W's in Equations 7.6 and 5.12) of different temporal scales that are learnt directly from data. This is why the proposed method is able to recommend solutions with low cost and high reward, while still making the suggested dynamics follow the intrinsic temporal dependencies from the data.

7.6 Pattern Matching vs. Profit Maximization

The merits of the proposed pattern matching and profit maximization are quite different. Indeed, from Figure 7.4, the proposed pattern matching is capable of producing valid solutions for 98% of the test samples, while reducing the cost by an average of 82% with a minor mismatch within 5%. On the other hand, from Figure 7.8, the proposed profit minimization is capable of producing valid solutions for 44% of the test samples while improving the reward by an average of 27% with less than half of the original cost. Such a difference originates from the two tasks' different goals and formulations: pattern matching (Equation 7.3) aims at matching a given pattern with the lowest cost, whereas profit maximization aggressively maximizes reward and minimizes cost.

Further, such a difference in formulation implies a difference in the fundamental

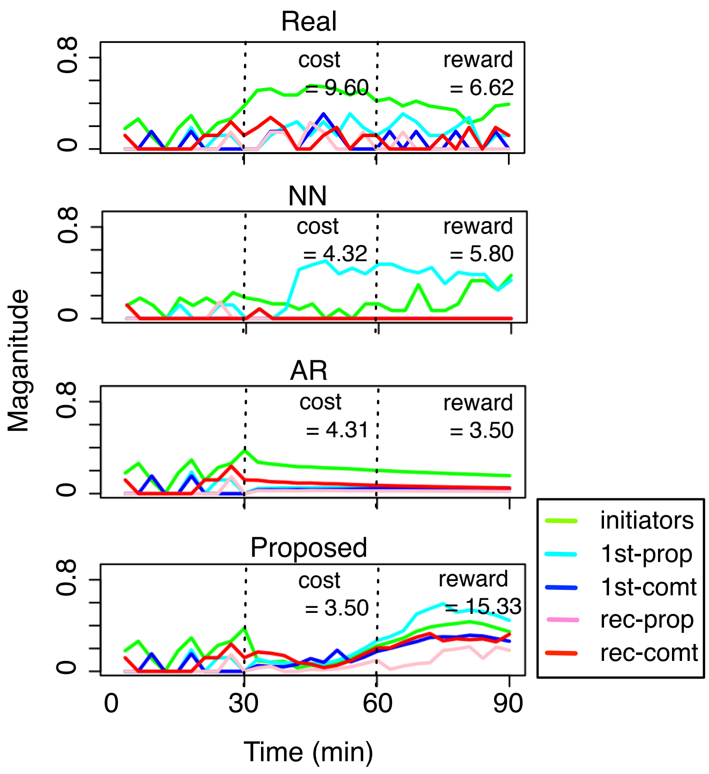


Figure 7.9: Case study: real versus suggested dynamics using different methods. A good solution is characterized by low cost (during $t \in [30, 60]$) and high reward (during $t \in [60, 90]$). The X-axis denotes time (in minutes) and the Y-axis denotes the normalized number of different types of users.

difficulties of two tasks. More importantly, profit maximization is significantly more difficult because while the "cost" has a natural lower bound (i.e., zero), the "reward", in principle, does not have any upper bound. In other words, unless the parameter λ is assigned perfectly, it is very easy to either obtain an invalid solution or a low-score solution. Therefore, in many engineering cases (e.g., marketing promotion), although profit maximization may be more desirable, in practice, pattern matching may be more useful.

The above analysis is confirmed by our experimental results in two ways. First, by comparing Figure 7.2 with Figure 7.6, we see that it is significantly harder to generate a valid solution in profit maximization. Indeed, compared to the case of pattern matching (Figure 7.2), the area above the horizontal line $\gamma \geq 0$ is much smaller in the case of profit maximization (Figure 7.6). Also, compared to the case of pattern matching, the range of λ that corresponds to $\gamma \geq 0$ ($\lambda \approx 0.01$) is much more narrow than the case of Figure 7.2 ($\lambda \in [0.01, 1]$). It suggests that it is harder to select a good value for the parameter λ in the case of profit maximization. Second, by comparing Figure 7.3 with Figure 7.7, we see that while pattern matching is capable of reducing both cost and mismatch, profit maximization is more capable of achieving a reasonable reward using low cost, compared to achieving a very high reward using moderate cost. Indeed, from Figure 7.7, we see that although the cost of the proposed solution is much lower than the case of the NN (i.e., nearest-neighbor) baseline, their highest possible rewards are only comparable.

These differences among the two tasks have practical implications on their real applications. First, if the ideal outcome pattern is given, pattern matching is the better option since from Figure 7.4, there is a 98% chance that the solution will be valid and of low cost and mismatch. Second, if the ideal pattern is not given, from Figure 7.4 there is a 44% chance that the solution is valid. In this case, a moderately high reward with a low cost can be expected.

Chapter 8

Future Work

There are many possible directions to extend the current work. We break it down into three directions.

8.1 Dynamics Understanding

Since our proposed algorithms address the two important properties, i.e., heterogeneity and multi-scale compositionality, of social dynamics with high scalability and guaranteed solution quality, they can be used as a convenient and reliable tool to aid research in social science. In particular, most existing theories in social science are based on manual discovery, explanation, and reasoning of patterns. With our algorithms, we can automate the discovery part and enable the social researchers to focus on the explanation and reasoning parts. Accordingly, existing theories can be efficiently confirmed, whereas new theories being explored, in a more efficient and data-driven way.

8.2 Data-driven Dynamics Engineering

What we presented in Chapter 7 is only the first step toward a new field with great potential, namely, data-driven dynamics engineering. While significant follow-up work can be built on the foundation this work offers, we would like to point out two directions that seem to hold the most promise. The first direction involves exploring different combinations of generative models and score functions as mentioned in Equation 7.5. Although we derive our solution based on a particular model (RCBM) and evaluate it using two specific score functions (Equations 7.3 and 7.4), other combinations can introduce equally, if not more, important engineering applications.

The second direction involves building a complete tool chain of dynamics engineering. In that sense, this work only accomplishes the very first component, which is figuring out what the ideal intervention should be. Two other important components in the tool chain are (1) how to *implement* that intervention most effectively and most efficiently and (2) how to efficiently *validate* the effectiveness of intervention given limited resources (i.e., using field experiments). We note that this work serves as the foundation for the other two components by providing a principled, data-driven method to offer an ideal intervention and its anticipated outcome. Using such information, the engineer gets to eliminate the need for trial-and-error among all possible interventions. Instead, he or she can focus on the implementation and validation perspectives of dynamics engineering, both of which justify an in-depth investigation on their own right.

8.3 More Applications

Although we look exclusively at social dynamics, the formulation proposed in this work applies generally to any multi-dimensional time series. Indeed, applications such as mobile context-aware computing, computational economics, and healthcare could potentially also use this framework to engineer their own dynamics problems. In this sense, we believe that developing a discipline for data-driven dynamics understanding and engineering is full of long-term potential.

Chapter 9

Conclusion

In this thesis, we have explored the data-driven understanding and engineering of social dynamics. We have addressed two unexplored properties of social dynamics, namely, heterogeneity and multi-scale compositionality. Consequently, we have proposed mining patterns (of social dynamics) using Angle-Shift (ASH) and decomposing hidden structures using Recursive Convolutional Bayesian Model (RCBM). For each of these proposed methods, we have derived formal guarantees in terms of both runtime and solution quality. These theoretical guarantees are then confirmed empirically: we achieve a 50~70X runtime speedup while delivering significantly better solution qualities compared to the prior state of the arts. Also, each of them is put to work in practical applications using real-world social-dynamics data, including Twitter and Yelp.

We are also the first to introduce the data-driven dynamics engineering problem, for which we propose a general framework that can incorporate various models and engineering objectives. Experiment results show that it can be used to do various useful engineering tasks, such as matching a specific pattern using minimum intervention cost, or maximizing the long-term reward while minimizing the short-term intervention. Finally, although we mainly focus on social dynamics in this thesis, our methods can potentially be applied to other multi-dimensional time series, which we demonstrate using two very different applications: human behavior modeling and macroeconomics.

Bibliography

[1] L. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley. Classes of small-world networks. *PNAS*, Oct. 2000.

[2] S. Asur, B. A. Huberman, G. Szabo, and C. Wang. Trends in social media : Persistence and decay. In *Proc. of the International Conference Web and Social Media (ICWSM)*, 2011.

[3] A. Barabasi. The origin of bursts and heavy tails in human dynamics. *Nature*, 2005.

[4] A.-L. Barabási. Scale-free networks: a decade and beyond. *Science*, July 2009.

[5] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 1975.

[6] S. P. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[7] C. Chatfield. *Time-series forecasting.* Chapman and Hall/CRC, 2002.

[8] J. R. Chen. Useful Clustering Outcomes from Meaningful Time Series Clustering. In *Proc. of the 6th Australasian conference on Data mining and Analytics*, 2007.

[9] W. Chen, Y. Yuan, and L. Zhang. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. *Proc. of the International Conference on Data Mining (ICDM)*, Dec. 2010.

[10] N. Christakis and James Fowler. The collective dynamics of smoking in a large social network. *New England journal of medicine*, 2008.

[11] K. K. W. Chu and M. H. Wong. Fast time-series searching with scaling and shifting. *Proc. of the ACM Symposium on Principles of Database Systems*, 1999.

[12] R. Collobert. Deep learning for efficient discriminative parsing. In *International Conference on Artificial Intelligence and Statistics*, 2011.

[13] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. In *Proc. of the National Academy of Science (PNAS)*, 2008.

[14] P. Dow, L. Adamic, and A. Friggeri. The Anatomy of Large Facebook Cascades. *Proc. of the International Conference Web and Social Media (ICWSM)*, 2013.

[15] P. Domingos and G. Hulten. Mining High-Speed Data Streams. *Proc. of Knowledge Discovery and Data Mining (KDD)*, 2000.

[16] P. Domingos and M. Richardson. Mining the Network Value of Customers. *Proc. of Knowledge Discovery and Data Mining (KDD)*, 2001.

[17] R. Grosse, R. Raina, H. Kwong, and A. Ng. Shift-invariant sparse coding for audio classofication. *Proc. of Uncertainty in Artificial Intelligence (UAI)*, 2012.

[18] M. Gupta, J. Gao, C. Zhai, and J. Han. Predicting future popularity trend of events in microblogging platforms. *Proc. of American Society of Information Science and Technology*, 2012.

[19] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006.

[20] T. Huynh and B. Schiele. Analyzing features for activity recognition. *Proc. of the joint conference on Smart objects and ambient intelligence*, (october):1–6, 2005.

[21] V. Jakkula and D. J. Cook. Anomaly detection using temporal data mining in a smart home environment. *Methods of information in medicine*, 2008.

[22] D. James and S. J. Koopman. *Time series analysis by state space methods.* Oxford University Press, 2012.

[23] B. Jansen and M. Zhang. Twitter power: Tweets as electronic word of mouth. *Journal of the American Society of Information Science and Technology.*, 2009.

[24] K. Kavukcuoglu and P. Sermanet. Learning convolutional feature hierarchies for visual recognition. *Advances in Neural Information Processing (NIPS)*, (1):1–9, 2010.

[25] E. Keogh and C. Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 2005.

[26] J. Kleinberg. Bursty and hierarchical structure in streams. *Proc. of Knowledge Discovery and Data Mining (KDD)*, 2002.

[27] J. Kwapisz, G. Weiss, and S. Moore. Activity recognition using cell phone accelerometers. *Proc. of SensorKDD*, 12(2):74–82, 2011.

[28] H. Lee, R. Grosse, R. Ranganath, and A. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. *International Conference on Machine Learning (ICML)*, pages 1–8, 2009.

[29] J. Lehmann and B. Gonçalves. Dynamic classes of collective attention in twitter. *Proc. of the World Wide Web Conference (WWW)*, 2012.

[30] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over Time : Densification Laws , Shrinking Diameters and Possible Explanations. In *Proc. of Knowledge Discovery and Data Mining (KDD)*, 2005.

[31] R. Levine and S. Zervos. Stock Markets, Banks and Economic Growth. *The American Economic Review*, 88(3), 1998.

[32] J. Lin, E. Keogh, L. Wei, and S. Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 2007.

[33] Y. Lin, Y. Chi, and S. Zhu. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. *Proc. of the World Wide Web Conference (WWW)*, 2008.

[34] Y. Lin, D. Margolin, and B. Keegan. # Bigbirds Never Die: Understanding Social Dynamics of Emergent Hashtag. *Proc. of the International Conference Web and Social Media (ICWSM)*, 2013.

[35] J. W. Lockhart, T. Pulickal, and G. M. Weiss. Applications of mobile activity recognition. In *Proc. of UbiComp*, 2012.

[36] Y. Matsubara, Y. Sakurai, B. Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: model and implications. *Proc. of Knowledge Discovery and Data Mining (KDD)*, 2012.

[37] Y. Matsubara, Y. Sakurai, and C. Faloutsos. AutoPlait: Automatic Mining of Co-evolving Time Sequences. *Proc. of ACM SIGMOD*, 2012.

[38] Y. Matsubara, Y. Sakurai, W. Panhuis, and C. Faloutsos. FUNNEL: Automatic Mining of Spatially Coevolving Epidemics. *Proc. of Knowledge Discovery and Data Mining (KDD)*, 2014.

[39] Y. Matsubara, Y. Sakurai, and C. Faloutsos. The Web as a Jungle: Non-Linear Dynamical Systems for Co-evolving Online Activities. *Proc. of World Wide Web (WWW)*, 2015.

[40] U. Maurer, A. Smailagic, D. Siewiorek, and M. Deisher. Activity Recognition and Monitoring Using Multiple Sensors on Different Body Positions. *International Workshop on Wearable and Implantable Body Sensor Networks (BSN)*, pages 113–116, 2006.

[41] T. Mitchell. *Machine Learning.* McGraw Hill, 1997.

[42] M. Moussaïd, J. E. Kämmer, P. P. Analytis, and H. Neth. Social influence and the collective dynamics of opinion formation. *PloS one*, Jan. 2013.

[43] J. Mueen and E. Keogh. Clustering time series using unsupervised-shapelets. In *International Conference on Data Mining (ICDM)*, 2012.

[44] M. Naaman, H. Becker, and L. Gravano. Hip and trendy: Characterizing emerging trends on twitter. *J. Am. Soc. Inf. Sci. and Tech.*, 2011.

[45] J. Nash. The imbedding problem for Riemannian manifolds. *Annals of Mathematics*, 1956.

[46] Y. Nesterov. Gradient methods for minimizing composite objective function, 2007.

[47] M. Newman. The Structure and Function of Complex Networks. *SIAM review*, 2003.

[48] G. Obozinski, B. Taskar, and M. Jordan. Multi-task feature selection. Technical report, Department of Statistics, University of California, Berkeley, 2006.

[49] B. A. Olshausen and D. J. Field. Emergence of Simple-cell Receptive Field Properties by Learning a sparse code for natural image. *Nature*, 1998.

[50] B. O'Neill. *Semi-Riemannian manifolds with applications to relativity.* Academic Press, San Diego, 1983.

[51] H.-K. Peng and R. Marculescu. Identifying Dynamics and Collective Behaviors in Microblogging Traces. In *Proc. of ASONAM*, 2013.

[52] H.-K. Peng and R. Marculescu. ASH : Scalable Mining of Collective Behaviors in Social Media using Riemannian Geometry. In *ASE International Conference on Social Computing*, 2014.

[53] H.-K. Peng and R. Marculescu. Multi-scale Compositionality: Identifying the Compositional Structures of Social Dynamics Using Deep Learning. *PloS One*, 2015.

[54] H.-K. Peng, H.-C. Lee, J.-Y. Pan, and R. Marculescu. Data-driven Engineering of Social Dynamics: Pattern Matching and Profit Maximization. *PloS One (in press)*, 2016.

[55] K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook.* Technical University of Denmark, 2008.

[56] S. Pirttikangas, K. Fujinami, and T. Nakajima. Feature selection and activity recognition from wearable sensors. *Ubiquitous Computing Systems*, 2006.

[57] K. Radinsky, K. Svore, S. Dumais, J. Teevan, A. Bocharov, and E. Horvitz. Modeling and predicting behavioral dynamics on the web. In *Proc. of the World Wide Web Conference (WWW)*, New York, New York, USA, 2012. ACM Press.

[58] T. Rakthanmanon. Time series epenthesis: clustering time series streams requires ignoring some data. *International Conference on Data Mining (ICDM)*, 2011.

[59] T. Rakthanmanon and E. Keogh. Fast shapelets: A scalable algorithm for discovering time series shapelets. *Proceedings of the thirteenth SIAM conference*, 2013.

[60] D. Rodrik. The Real Exchange Rate and Economic Growth. *Brookings Papers on Economic Activity*, 2008.

[61] D. Romero, B. Meeder, and J. Kleinberg. Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on Twitter. In *Proc. of the World Wide Web Conference (WWW)*, 2011.

[62] G. Smyth, D. K. L. H., M. G., and R. P. Rule Discovery from Time Series. In *Proc. of the 3rd Knowledge Discovery and Data Mining (KDD)*, 1998.

[63] T. Snijders, G. G. van de Bunt, and C. E. Steglich. Introduction to stochastic actor-based models for network dynamics. *Social Networks*, Jan. 2010.

[64] R. Sochard, C. C.-y. Lin, A. Y. Ng, and C. D. Manning. Parsing Natural Scenes and Natural Language. In *International Conference on Machine Learning (ICML)*, 2011.

[65] E. Spriggs, F. Torre, and M. Hebert. Temporal Segmentation and Activity Classification from First-person Sensing. In *IEEE Workshop on Egocentric Vision (CVPR)*, 2009.

[66] R. Sumi, T. Yasseri, A. Rung, A. Kornai, and K. Janos. Edit wars in Wikipedia. In *IEEE International Conference on Social Computing (SocialCom)*, 2011.

[67] G. Szabo and B. A. Huberman. Predicting the popularity of online content. *Communications of the ACM*, 2010.

[68] L. Tang and H. Liu. Toward predicting collective behavior via social dimension extraction. In *IEEE Intelligent Systems*, 2010.

[69] G. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. *European Conference on Computer Vision (ECCV)*, 2010.

[70] F. Torre, J. Hodgins, J. Montano, S. Valcarcel, R. Forcada, and J. Macey. Guide to the carnegie mellon university multimodal activity (CMU-MMAC) database. Technical report, Robotics Institute, Carnegie Mellon University, 2009.

[71] O. Tsur and A. Rappoport. What's in a Hashtag? Content based Prediction of the Spread of Ideas in Microblogging Communities. *Proc. of the International Conference on Web Search and Data Mining (WSDM)*, 2012.

[72] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen. Microblogging During Two Natural Hazards Events : What Twitter May Contribute to Situational Awareness. In *Proc. of CHI*, 2010.

[73] T. Warren Liao. Clustering of time series data: a survey. *Pattern Recognition*, Nov. 2005.

[74] L. Wasserman. *All of statistics: a concise course in statistical inference*. Springer, 2004.

[75] L. Wasserman. *All of nonparametric statistics.* Springer, 2005.

[76] Worldbank. The Worldbank Dataset, 2015.

[77] J. Yang and S. Counts. Predicting the Speed, Scale, and Range of Information Diffusion in Twitter. In *Proc. of the International Conference Web and Social Media (ICWSM)*, 2010.

[78] J. Yang and J. Leskovec. Patterns of temporal variation in online media. *Proc. of the International Conference on Web Search and Data Mining (WSDM)*, 2011.

[79] Yelp. Yelp Dataset Challenge, 2014.

[80] J. Yin, Q. Yang, S. Member, and J. J. Pan. Sensor-Based Abnormal Human-Activity Detection. *IEEE Transactions on Knowledge and Data Engineering*, 20(8):1082–1090, 2008.

[81] W. H. Young. On the multiplication of successions of Fourier constants. *Proceedings of the Royal Society A, JSTOR*, 1912.

[82] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus. Deconvolutional networks. *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 2528–2535, June 2010.

Appendix A

Mathematical Details of Chapter 4

A.1 Determination of stepsize

In the case when the optimization is to find the global optimum, the stepsize can be determined by techniques such as Backtracking [6]. These techniques cannot be used in our case, however, because we are now interested in finding the local optimum corresponding to each initial condition. Toward this end, the stepsize in our clustering algorithm is determined as follows. In the beginning, the stepsize t is initialized to ϵ, an user-specified threshold below which the error is considered negligible (e.g., $\epsilon = 10^{-6}$). Then, t is gradually increased by a factor of $\rho > 1$, until either the objective function ceases to increase, or when the stepsize pushes the solution out of the feasible space. Determining the stepsize this way ensures that Algorithm 1 is an ascent method and hence always converges.

A.2 Asymptotic Convergence Rate of ASH

Lemma 1. ***Convergence Rate under Euclidean Metric.*** *Let $f(\mathbf{x}) : \mathbf{R}^d \to \mathbf{R}$ be a pdf defined over $\mathbf{x} \in \mathbf{R}^d$ with the standard Euclidean metric $g(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$. Suppose that $f(\mathbf{x})$ satisfies* **(A1)** *-* **(A3)***. Then under Definition 1, we have that $\forall \hat{\mathbf{x}} \in \hat{X}$, $\exists \mathbf{x}^* \in X^*$ that satisfies:*

$$P(\|\mathbf{x}^* - \hat{\mathbf{x}}\| > C\sqrt{\log n}^{\alpha} n^{\frac{-\alpha\beta}{\beta + \frac{1}{2}d}}) \leq \Delta \;\; \forall \Delta > 0$$

where C depends on Δ and α.

Proof. Using the Uniform Bound of f_n [75], we have:

$$P(\sup_{\mathbf{x}} |f(\mathbf{x}) - f_n(\mathbf{x})| > C_0\sqrt{\log n} n^{\frac{-\beta}{\beta + \frac{1}{2}d}}) \leq \Delta \;\; \forall \Delta > 0$$

where C_0 depends on Δ. That is, with high probability (w.h.p.), we have:

$$|f(\mathbf{x}) - f_n(\mathbf{x})| \leq C_0\sqrt{\log n} n^{\frac{-\beta}{\beta + \frac{1}{2}d}} := \delta \tag{A.1}$$

Using geometric arguments, we know that with large enough n and w.h.p., we have that for each $\hat{\mathbf{x}} \in \hat{X}$:

$$\begin{aligned} \exists \mathbf{x}^* \in X^* \cap A_n(\hat{\mathbf{x}}, \delta) \quad s.t. \quad & |f(\mathbf{x}^*) - f_n(\hat{\mathbf{x}})| < \delta \\ \Rightarrow \quad & |f(\mathbf{x}^*) - f(\hat{\mathbf{x}})| < 2\delta \end{aligned}$$

From **(A3)**, this implies that w.h.p.:

$$\|\mathbf{x}^* - \hat{\mathbf{x}}\| = O((2\delta)^{\alpha}) = O(\sqrt{\log n}^{\alpha} n^{\frac{-\alpha\beta}{\beta + \frac{1}{2}d}}).$$

The proof follows. $\square$

Lemma 2. ***Nash's Embedding Theorem*** **[45].** *Let $\mathcal{M} \subseteq \mathbf{R}^d$ be a C^p Riemannian Manifold ($p \geq 3$) with a metric $g(\mathbf{x}, \mathbf{y})$. Then there exists a Euclidean space $\mathbf{R}^m$ with $m = \frac{1}{2}d(3d+11)$ that satisfies: (1) $\forall \mathbf{x} \in \mathcal{M}, \exists \Phi\mathbf{x} \in \mathbf{R}^m$ and (2) $\forall \mathbf{x}, \mathbf{y} \in \mathcal{M}, g(\mathbf{x}, \mathbf{y}) = \|\Phi\mathbf{x} - \Phi\mathbf{y}\|$.*

What Lemma 2 tells is that for our Manifold $\mathcal{M}$, there exists a higher dimensional space $\mathbf{R}^m$ where any event $\mathbf{x} \in \mathcal{M}$ has a unique mapping $\Phi\mathbf{x} \in \mathbf{R}^m$. It also says that the Euclidean distance between the two mapped events $\Phi\mathbf{x}, \Phi\mathbf{y} \in \mathbf{R}^m$ are identical to the distance $g(\cdot)$ between the two events in $\mathcal{M}$. With this Lemma, we can proof the convergence rate in the Manifold using Lemma 1.

Proof. (of Theorem 1) Using Lemma 2, let $\mathcal{M}_\Phi = \{\Phi\mathbf{x} \mid \mathbf{x} \in \mathcal{M}\}$ be the image of $\mathcal{M}$ in $\mathbf{R}^m$. Note that there is an one-to-one mapping between $\mathcal{M}$ and $\mathcal{M}_\Phi$. Then $\forall \Phi\mathbf{x} \in \mathcal{M}_\Phi$, define two functions:

$$f_\Phi(\Phi\mathbf{x}) := f(\mathbf{x}), \;\; f_{n\Phi}(\Phi\mathbf{x}) := \frac{1}{nh^m}\sum_{i=1}^{n} K(\frac{\|\Phi\mathbf{x} - \Phi\mathbf{x}_i\|)}{h}).$$

It is straightforward to check that f_Φ is also a pdf that satisfies **(A1)**, **(A2)**, and **(A3)**. Let ΦX^* and $\Phi\hat{X}$ be the sets of maxima of f_Φ and $f_{n\Phi}$, respectively. Since $\mathcal{M}_\Phi$ employes a standard Euclidean metric, from Lemma 1, we have that $\forall \hat{\Phi\mathbf{x}} \in \Phi\hat{X}$, $\exists \Phi\mathbf{x}^* \in \Phi X^*$ that satisfies:

$$\|\Phi\mathbf{x}^* - \hat{\Phi\mathbf{x}}\| = O(\sqrt{\log n}^{\alpha} n^{\frac{-\alpha\beta}{\beta + \frac{1}{2}m}}).$$

Finally, using Lemma 2, we have the proof. $\square$

Proof. (of Theorem 2) Using the uniform-bound of f_n [75], Equation A.1, and some geometric arguments, we see that letting $l^* > 2\delta$ is sufficient to ensure that $\forall \mathbf{x}^* \in X^*$, $\exists \hat{\mathbf{x}} \in \hat{X}$ that satisfies $\hat{\mathbf{x}} \in A(\mathbf{x}^*, \delta)$. Using **(A3)** and the same argument used in the proof of Theorem 1, we have the proof. $\square$

Appendix B

Mathematical Details of Chapter 5

B.1 Asymptotic Convergence of RCBM

Lemma 3. ***Lipschitz Continuity of*** f_1 ***w.r.t.*** W *: Let* $f_1(W,h) = \frac{1}{2}\|X^{(i)} - \sum_k W_k \otimes h_k^{(i)}\|_F^2$*. Then* $f_1(W,h)$ *is L-Lipschitz-continuous w.r.t.* W_k *with the constant:*

$$L = \sum_i \|h_k^{(i)}\|_1^2.$$

Proof. Using Equation 5.7 and let $\Delta W_k = W_k - W_k'$, we have:

$$\begin{aligned}
& \|\nabla f_1(W_k) - \nabla f_1(W_k')\| \\
= \; & \|\textstyle\sum_i \tilde{h}_k^{(i)} \otimes h_k^{(i)} \otimes \Delta W_k\| \\
\leq \; & \textstyle\sum_i \|\tilde{h}_k^{(i)} \otimes h_k^{(i)} \otimes \Delta W_k\| \\
\leq \; & \textstyle\sum_i \|h_k^{(i)}\|_1 \cdot \|h_k^{(i)} \otimes \Delta W_k\| \\
\leq \; & \textstyle\sum_i \|h_k^{(i)}\|_1^2 \cdot \|\Delta W_k\|
\end{aligned} \tag{B.1}$$

Note that the third line is obtained using triangular inequality, whereas the fourth and the fifth lines are obtained using Young's inequality [81]. $\square$

Lemma 4. ***Lipschitz Continuity of*** f_1 ***w.r.t.*** h *: Let* $f_1(W,h) = \frac{1}{2}||X^{(i)} - \sum_k W_k \otimes h_k^{(i)}||_F^2$. *Then* $f_1(W,h)$ *is L-Lipschitz-continuous w.r.t.* h_k *with the constant:*

$$L = ||W_k||_1^2.$$

Proof. Using Equation 5.7 and let $\Delta h_k = h_k - h'_k$, we have:

$$\begin{aligned} & ||\nabla f_1(h_k) - \nabla f_1(h'_k)|| \\ = \; & ||\tilde{W}_k \otimes W_k \otimes \Delta h_k|| \\ \leq \; & ||W_k||_1 \cdot ||W_k \otimes \Delta h_k|| \\ \leq \; & ||W_k||_1^2 \cdot ||\Delta h_k|| \end{aligned} \tag{B.2}$$

Again, the third and the fourth lines are obtained using Young's inequality [81]. $\square$

Proof. (of Theorem 3) We will show that $\hat{W} \xrightarrow{p} W^{mle} \xrightarrow{p} W^*$, where W^{mle} denotes the globally optimal solution of Equation 5.6. First, note that according to Equations 5.3 and 5.6, W^{mle} is the maximum likelihood estimator (MLE) of W^*. Since MLE is consistent [74], we have that:

$$W^{mle} \xrightarrow{p} W^*. \tag{B.3}$$

Now, suppose $W^{[0]} = W^{mle}$ and L denotes that Lipschitz constant given in Lemma 3. Since the stepsize t_W in Algorighm 2 (i.e., Equation 5.9) satisfies that $t_w < 2/L$, we have that [6]:

$$f(W^{[k]}) - f(W^{mle}) \leq \frac{||W^{[0]} - W^{mle}||^2}{(k+1)^2},$$

where $f(\cdot)$ denotes the objective function of Equation 5.6. Since that right-hand side

(RHS) is zero, trivially, we have:

$$W^{[k]} = \hat{W} \xrightarrow{p} W^*. \tag{B.4}$$

Combining Equations B.3 and B.4, we have the proof. $\square$

Remark: Although we use $W^{[0]} = W^{mle}$ in the proof, it can be assigned to any point in the ϵ-ball $B_\epsilon(W^{mle}) = \{W \mid \|W - W^{mle}\| \leq \epsilon\}, \epsilon > 0$, where the objective function is convex in the set $W \in B_\epsilon(W^{mle})$.

Definition 4. ***Heterogeneity***: *A filter W^* is called* heterogeneous *if $\forall i \neq j$, $\exists \epsilon > 0$ such that:*

$$\|w_i^* - w_j^*\|^2 \geq \epsilon, \tag{B.5}$$

where w_i^* denotes the i-th row of W^*.

Proof. (of Theorem 4) We prove a non-trivial special case where the general case can be established similarly. Suppose $D = 2, K = 1$, and h is the delta vector (i.e., one for the element in the middle and zero elsewhere). Let W^{mle} denote the global optimal solution of Equation 5.6 using the conventional matrix convolution. According to the generation process (that uses the proposed convolution operator) and the definition of the conventional matrix convolution, we have that:

$$\begin{aligned} w_1^{mle} = w_2^{mle} \quad \xrightarrow{p} \quad & \arg\min_w \|w - w_1^*\|^2 + \|w - w_2^*\|^2 \\ & = \tfrac{1}{2}(w_1^* + w_2^*), \end{aligned} \tag{B.6}$$

where w_i^{mle} and w_i^* denote the i-th row of W^{mle} and W^*, respectively. Since W^{mle} is the optimal solution, we have that:

$$\|\hat{W} - W^*\|^2 \geq \|W^{mle} - W^*\|^2, \tag{B.7}$$

where the RHS converges (in probability) to:

$$\begin{aligned}
&2\left(\|\tfrac{1}{2}(w_1^* + w_2^*) - w_1^*\|^2 + \|\tfrac{1}{2}(w_1^* + w_2^*) - w_2^*\|^2\right) \\
&= 2(\tfrac{1}{4}\|w_2^* - w_1^*\|^2 + \tfrac{1}{4}\|w_1^* - w_2^*\|^2) \qquad \text{(B.8)} \\
&\geq \epsilon > 0.
\end{aligned}$$

Here the last inequality is obtained using Assumption A3 and Equation B.5. Using Equations B.7 and B.8, we have the proof. □

Proof. (of Theorem 5) Using Lemma 3, Lemma 4, and a convergence result in [46], we know that there is a positive constant c such that:

$$\epsilon = \|x^* - x^{[r]}\| \leq c\,r^{-2}.$$

Equivalently, it follows that:

$$r \leq \sqrt{\frac{1}{c\epsilon}} = O(\epsilon^{\frac{-1}{2}})$$

□

Appendix C

Mathematical Details of Chapter 7

The author gratefully acknowledge Dr. Hao-Chih Lee's help in proving the theorems in this appendix.

C.1 Quadratic-Programming Transformation

Lemma 5. *Given $X \in R^{D\times T_x}$, $Y \in R^{D\times T_y}$, define*

$$\tilde{I}_X = [I_{T_x\times T_x}\ \mathbf{0}_{T_x\times T_y}],\ \tilde{I}_Y = [\mathbf{0}_{T_y\times T_x}\ I_{T_y\times T_y}],$$

and

$$\hat{I}_X = I_D \odot \tilde{I}_X,\ \hat{I}_Y = I_D \odot \tilde{I}_Y,$$

where $\odot$ is the Kronecker product [55], then we have

$$\hat{I}_X \text{vec}([X\ Y]) = \text{vec}(X),\ \text{ and } \hat{I}_Y \text{vec}([X\ Y]) = \text{vec}(Y),$$

and

$$\hat{I}_X^T\hat{I}_X + \hat{I}_Y^T\hat{I}_Y = I_{D\cdot(T_x+T_y)}.$$

of lemma 1. Using property of the Kronecker product that $B^T \odot A\text{vec}(X) = \text{vec}(AX^TB^T)$ [55], we have

$$\begin{aligned}
\hat{I}_X\text{vec}([X\ Y]) &= I_D \odot \tilde{I}_X\text{vec}([X\ Y]) \\
&= \text{vec}(\tilde{I}_X[X\ Y]^T I_D) \\
&= \text{vec}([I_{T_x\times T_x}\ 0_{T_x\times T_y}][X\ Y]^T) \\
&= \text{vec}(X).
\end{aligned}$$

For the second claim, we compute

$$\begin{aligned}
\hat{I}_X^T\hat{I}_X &= (I_D \odot \tilde{I}_X)^T(I_D \odot \tilde{I}_X) \\
&= (I_D \odot \tilde{I}_X^T)(I_D \odot \tilde{I}_X) \\
&= I_D \odot (\tilde{I}_X^T\tilde{I}_X).
\end{aligned}$$

Similarly $\hat{I}_Y^T\hat{I}_Y = I_D \odot \tilde{I}_Y^T\tilde{I}_Y$ and thus

$$\hat{I}_X^T\hat{I}_X + \hat{I}_Y^T\hat{I}_Y = I_D \odot (\tilde{I}_X^T\tilde{I}_X + \tilde{I}_Y^T\tilde{I}_Y) = I_D \odot I_{(T_x+T_y)} = I_{D\cdot(T_x+T_y)}.$$

□

Lemma 6. *Given filter matrices W_{ik}, then the vectorization of its convolution operation with respect to h_{ik} can be expressed as*

$$\text{vec}(\sum_{k=1}^{K_1} h_{ik} \otimes W_{ik}) = \begin{bmatrix} G_{11}^{(i)} & G_{12}^{(i)} & \dots & G_{1K_1}^{(i)} \\ \vdots & \vdots & \vdots & \vdots \\ G_{D1}^{(i)} & G_{D2}^{(i)} & \dots & G_{DK_1}^{(i)} \end{bmatrix} \begin{bmatrix} h_{i1}^T \\ h_{i2}^T \\ \vdots \\ h_{iK_1}^T, \end{bmatrix}$$

where $G_{lk}^{(i)}$ is the Toeplitz matrix representing the operation $\otimes$ w.r.t a single filter $w_{lk}^{(i)}$ in l-th row of the filter matrix W_{ik} [55], i.e., $G_{lk}^{(i)} h_{ik}^T = (w_{lk}^{(i)} \otimes h_{ik})^T$.

of lemma 2. Given that $w_{lk}^{(i)}$ is the l-th row of the filter matrix W_{ik}, we have

$$\begin{aligned}
\mathrm{vec}(\textstyle\sum_{k=1}^{K_1} h_{ik} \otimes W_{ik}) &= \textstyle\sum_{k=1}^{K_1} \begin{bmatrix} G_{1k}^{(i)} h_{ik}^T \\ G_{2k}^{(i)} h_{ik}^T \\ \vdots \\ G_{Dk}^{(i)} h_{ik}^T \end{bmatrix} \\
&= \begin{bmatrix} G_{11}^{(i)} & G_{12}^{(i)} & \dots & G_{1K_1}^{(i)} \\ \vdots & \vdots & \vdots & \vdots \\ G_{D1}^{(i)} & G_{D2}^{(i)} & \dots & G_{DK_1}^{(i)} \end{bmatrix} \begin{bmatrix} h_{i1}^T \\ h_{i2}^T \\ \vdots \\ h_{iK_1}^T \end{bmatrix}
\end{aligned}$$

□

of Theorem 1. Using Lemma 5 and 6, we can compute

$$\begin{aligned}
\|[X\ Y] - \textstyle\sum_{k=1}^{K_1} h_k \otimes W_{1k}\|_F^2 &= \|\mathrm{vec}([X\ Y]) - G_1 h_1\|_F^2 \\
&= \|\mathrm{vec}(X) - \hat{I}_X G_1 h_1\|_F^2 + \|\mathrm{vec}(Y) - \hat{I}_Y G_1 h_1\|_F^2 \\
&= \mathbf{h}_1^T G_1^T G_1 \mathbf{h}_1 + \mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \hat{I}_Y G_1 \mathbf{h}_1 - 2\mathbf{x}^T \hat{I}_X G_1 \mathbf{h}_1 + \mathbf{x}^T \mathbf{x}.
\end{aligned}$$

Similarly,

$$\begin{aligned}
\|S - \textstyle\sum_{k=1}^{K_2} h_{2k} \otimes W_{2k}\|_F^2 &= \|\mathrm{vec}(S) - G_2 h_2\|_F^2 \\
&= \mathbf{h}_2^T G_2^T G_2 \mathbf{h}_2 + \mathbf{s}^T \mathbf{s} - 2\mathbf{s}^T G_2 \mathbf{h}_2.
\end{aligned}$$

Expand these two terms in the objective function of equation 7.8 and sum up everything, we have the desired form in Theorem 6. □

C.2 AR baseline

The order-p ARMA model is defined by $x_t = \sum_{i=1}^{p} \Phi_i x_{t-i} + \epsilon_t,$, where $x_i \in R^{D\times 1}$ are multivariate features and $\epsilon_t \sim \mathcal{N}(0, \Sigma)$ is the i.i.d. Gaussian noise. We shall derive and solve the equation 7.5 based on order-p ARMA model. First we derive the log likelihood of order-p ARMA.

Lemma 7. *Let $\tilde{\Phi}_i = \Sigma^{-1/2}\Phi_i$, and define the Toeplitz matrix of $[-\tilde{\Phi}_p, -\tilde{\Phi}_{p-1}, \ldots, -\tilde{\Phi}_1, I_D]$ by shifting $\tilde{\Phi}$ $T_x - p$ times, i.e.,*

$$\hat{\Phi} = \begin{bmatrix} -\tilde{\Phi}_p & \ldots & -\tilde{\Phi}_1 & \Sigma^{-1/2} & & & \\ & -\tilde{\Phi}_p & \ldots & -\tilde{\Phi}_1 & \Sigma^{-1/2} & & \\ & & \ddots & & \ddots & \ddots & \\ & & & -\tilde{\Phi}_p & \ldots & -\tilde{\Phi}_1 & \Sigma^{-1/2} \end{bmatrix}.$$

Then

$$\sum_{t=p+1}^{T_x} \log P(x_t | x_{t-i}\ i = 1, \ldots, p) = -\|\hat{\Phi}[x_1^T, \ldots, x_{T_x}^T]^T\|_F^2/2.$$

Theorem 7. *Assume that the order-p AR time series is divided in to three sections 1) the given observation $x = [x_1^T\ x_2^T \ldots x_p]^T$ 2) the intervention $u = [x_{p+1}^T \ldots x_{p+T_u}^T]^T$ and the outcome $v = [x_{p+T_u+1}^T \ldots x_{p+T_u+T_v}^T]^T$ with a reference pattern v_{ref}. Let $Q = \hat{\Phi}^T\hat{\Phi} = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}$ where $Q_{11} \in R^{T_x \times T_x}$, $Q_{22} \in R^{T_y \times T_y}$ and $y^T = [u^T v^T]$. Then the problem*

$$\begin{array}{ll} \text{argmax}_y & \log P(y|x) - \lambda(\|v - v_{ref}\|_F^2 + \rho\|u\|_1) \\ \text{s.t.} & y \geq 0 \end{array}$$

is equivalent to a quadratic programming problem

$$\begin{array}{ll} \operatorname{argmin}_y & \frac{1}{2}y^T(Q_{22} + 2\lambda \begin{bmatrix} 0 & 0 \\ 0 & I_v \end{bmatrix})y + (Q_{21}x - \lambda d)^T y \\ \text{s.t.} & y \geq 0 \end{array}$$

where $d^T = [-\rho \mathbf{1}^T,\ 2v_{ref}^T]$.

Proof. Both the lemma and theorem are proved by direct calculation. □