Trend or No Trend: A Novel Nonparametric Method for Classifying Time Series

by

Stanislav Nikolov

S.B., Massachusetts Institute of Technology (2011)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2012

Author .... [illegible] ....
Department of Electrical Engineering and Computer Science
August 15, 2012

Certified by .... [illegible] ....
Prof. Devavrat Shah
Jamieson Career Development Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor
August 15, 2012

Certified by . [illegible] ....
Dr. Satanjeev Banerjee
Engineer, Twitter Inc.
Thesis Co-Supervisor
August 15, 2012

Accepted by . [illegible] ....
Prof. Dennis M. Freeman
Chairman, Masters of Engineering Thesis Committee

Trend or No Trend: A Novel Nonparametric Method for Classifying Time Series

by

Stanislav Nikolov

Submitted to the Department of Electrical Engineering and Computer Science
on August 15, 2012, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

In supervised classification, one attempts to learn a model of how objects map to labels by selecting the best model from some model space. The choice of model space encodes assumptions about the problem. We propose a setting for model specification and selection in supervised learning based on a *latent source model*. In this setting, we specify the model by a small collection of unknown *latent sources* and posit that there is a stochastic model relating latent sources and observations. With this setting in mind, we propose a nonparametric classification method that is entirely unaware of the structure of these latent sources. Instead, our method relies on the data as a proxy for the unknown latent sources. We perform classification by computing the conditional class probabilities for an observation based on our stochastic model. This approach has an appealing and natural interpretation — that an observation belongs to a certain class if it sufficiently resembles other examples of that class.

We extend this approach to the problem of online time series classification. In the binary case, we derive an estimator for online signal detection and an associated implementation that is simple, efficient, and scalable. We demonstrate the merit of our approach by applying it to the task of detecting *trending topics* on Twitter. Using a small sample of Tweets, our method can detect trends before Twitter does 79% of the time, with a mean early advantage of 1.43 hours, while maintaining a 95% true positive rate and a 4% false positive rate. In addition, our method provides the flexibility to perform well under a variety of tradeoffs between types of error and relative detection time.

Thesis Supervisor: Prof. Devavrat Shah
Title: Jamieson Career Development Associate Professor of Electrical Engineering and Computer Science

Thesis Supervisor: Dr. Satanjeev Banerjee
Title: Engineer, Twitter Inc.

Acknowledgments

This thesis would not have been possible without my advisor Devavrat Shah. I would like to express my deepest gratitude to Devavrat for his patience, for his breadth and depth of technical knowledge, for his infectious enthusiasm, and for teaching me the virtues of simplicity and optimism — all the way back to his warm response to my cold email.

I would also like to thank my co-advisor Bano Banerjee for invaluable brainstorming, discussions, and advice when I was starting out. His advice on how to choose a topic and how to conduct good research in general will stay with me. Thanks also for taking over my projects so I could focus on my thesis.

Many thanks are due to Judy Cheong, for always being cheerful, and for heroically handling the mountain of paperwork needed for me to stay at Twitter; to Ben Lee, for taking time out of fighting the good fight to make sure we don't do anything illegal; to Frank Lin, for emphasizing the "why" over the "what"; and to Ram Ravichandran for his intimate knowledge of the structure of retweet activity preceding a trend.

Special thanks are in order to Andy Schlaikjer and Jeff Dalton for many brainstorming sessions, discussions, coffee runs, and adventures. Their experience and technical knowledge helped me come up with a topic and influenced the way I think about research and academia.

A big thank you to Mark Tobenkin for his endless curiosity and for his uncanny ability to walk into my office at just the right time to help me with control theory problems. If not for him, I might still be finishing my final projects rather than this thesis.

Thank you to my friends, for always being there; to my mother, father and brother, for everything.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Detection, classification, and prediction of events in temporal streams of information are ubiquitous problems in science, engineering and society. From detecting malfunctions in a production plant, to predicting an imminent market crash, to revealing emerging popular topics in a social network, extracting useful information from time-varying data is fundamental for understanding the processes around us and making decisions.

In recent years, there has been an explosion in the availability of data related to virtually every human endeavor — data that demands to be analyzed and turned into valuable insights. Massive streams of user generated documents, such as blogs and tweets, as well as data from portable electronic devices, provide an amazing opportunity to study the dynamics of human social interaction online and face to face [16][17]. How do people make decisions? Who are they influenced by? How do ideas and behaviors spread and evolve? These are questions that have been impossible to study empirically at scale until recent times. In healthcare, records of over-the-counter medication sales [21] as well as search engine queries [22] can anticipate the outbreak of disease and provide insight into the most effective ways to limit its spread. Particle collision experiments at the Large Hadron Collider generate more than 15 petabytes of data [1] every year that promises to reveal fundamental physical truths.

Such large quantities of data present both opportunities and challenges. On the one hand, enough data can reveal the hidden underlying structure in a process of interest. On the other hand, making computations over so much data at scale is a challenge. Fortunately, in recent years, advances in distributed computing have made it easier than ever to exploit the structure in large amounts of data to do inference at scale.

All of the examples mentioned above share a common setting. There exists an underlying process whose observable properties generate time series. Using these time series, one may wish to do inference such as detecting anomalous events, classifying the current activity of the time series, or predicting the values of the time series at some future point.

This is difficult to do in general. Many real-world processes defy description by simple models. A quote from "The Unreasonable Effectiveness of Data" [9] by Halevy, Norvig, and Pereira sums this up:

> "*Eugene Wigner's article 'The Unreasonable Effectiveness of Mathematics in the Natural Sciences' examines why so much of physics can be neatly explained with simple mathematical formulas such as $f = ma$ or $e = mc^2$. Meanwhile, sciences that involve human beings rather than elementary particles have proven more resistant to elegant mathematics. Economists suffer from physics envy over their inability to neatly model human behavior. An informal, incomplete grammar of the English language runs over 1,700 pages. Perhaps when it comes to natural language processing and related fields, we're doomed to complex theories that will never have the elegance of physics equations. But if that's so, we should stop acting as if our goal is to author extremely elegant theories, and instead embrace complexity and make use of the best ally we have: the unreasonable effectiveness of data.*"

Like language, the behavior of complex systems rarely admits a simple model that works well in practice. Like machine translation and speech recognition, there is an ever growing amount of data "in the wild" about processes like epidemics, rumor-spreading in social networks, financial transactions, and more. The inadequacy of simple models for complex behavior requires an approach that embraces this wealth of data and it highlights the need for a unified framework that efficiently exploits the structure in that data to do detection, classification, and prediction in time series.

In this thesis, we study the problem of prediction in a complex system using large amounts of data. Specifically, we focus on binary classification of time series and ask whether we can tell apart "events" from "non-events" given sufficient historical examples. We apply this to the problem of *trending topic* detection on Twitter and show that we can reliably detect trending topics before they are detected as such by Twitter. At the same time, we aim to introduce a more general setting for doing inference in time series based on a large amount of historical data.

1.2 Previous Work

A popular approach to detecting emerging popular topics in document streams is to measure the deviation of topics' activity relative to some baseline. Ihler et al. [10] propose an event detection framework based on time-varying Poisson processes, in which a baseline Poisson rate is estimated in a sliding window and anomalies are considered to be deviations from a local the baseline. Becker et al. [5], Cataldi et al. [6], and Mathioudakis and Koudas. [13] all group terms together to form topics and use a combination of features including temporal activity and social authority and interaction to detect trending topics.

Many approaches to emergent topic detection specifically, and detecting outbreaks in networks more generally, involve explicit models of a spreading process over a network. Asur et al. [2] model the formation, persistence and decay of trending topics on Twitter using a branching process model. Shtatland and Shtatland [20] investigate outbreak phenomena based on an underlying SIR model for spreading. They train a stationary autoregressive model for spreading activity and declare anomalies when the model starts to become non-stationary. Gruhl et al. [8] use a cascade model to study the propagation of information through a social network. They posit that topic activity is composed of "spikes" and "chatter" and characterize individuals in the network in terms of different posting behaviors. They then use an EM algorithm to infer which edges may have been active in the spread of the topic. For modeling the activity of topics in a document stream, models not based on networks also exist. Kleinberg [11] models a stream of documents using an infinite state automaton and computes optimal state sequences based on the data to infer how the observed activity was generated.

A third class of methods operates on large collections of time series as a way to reason about the underlying hidden process without explicitly modeling that process. Along those lines a number of trajectory clustering methods have emerged recently. Gaffney and Smyth [7] propose a method for trajectory clustering, where each cluster is modeled as a prototype trajectory with some noise. They produce low dimensional representations of the trajectories by using regression models and train a finite mixture model on the regression model components. Munaga et al. [15] offer a more flexible approach that is able to identify regions of high density that are separated from one another by regions of low density as well as automatically determine the number of clusters. In the realm of supervised learning, McNames [14] uses a "nearest trajectory" strategy to do prediction in time series. Lenser and Veloso [12] propose a method for nonparametric time series classification in time series produced by a process consisting of different underlying states, in which pieces of the time series are classified as having been produced by a certain state.

1.3 Our Approach

Simple, parametric models prove ineffective at modeling many real-world complex systems. To resolve this, we propose a nonparametric framework for doing inference on time series. In this model, we posit the existence of a set of *latent source* time series, or signals, each corresponding to a prototypical event of a certain type, and that each observed time series is a noisy observation of one of the latent time series.

In the case of classification, an observed signal, is compared to two sets of *reference* signals — one consisting of positive examples and the other of negative examples. We posit that the observation belongs to the positive (resp. negative) class if it was generated by the same latent source as one of the positive (resp. negative) examples. To do classification, we compute the class probabilities conditioned on the observation. In our model, doing so involves a surprisingly simple computation — to see how likely it is that an observation belongs to a certain class, one simply computes the distances from the observation to the reference signals in that class. This allows us to infer the class in a nonparametric fashion directly from the data without specifying any model structure.

1.3.1 Application: Detecting Outbreaks of Popular Topics on Twitter

As an application, we apply the latent source model to the problem of detecting emerging popular topics (called *trends*) on Twitter. Twitter is a real-time messaging service and information network whose users can post short (140 characters or fewer) messages called *Tweets*. Tweets are public by default and broadcast to the users' *followers*. Users can engage in conversation with one another and join a potentially global conversation on a variety of topics being discussed. Inevitably, some topics, such as a breaking news event, gain sudden popularity on Twitter. Twitter surfaces such topics in the service as a list of *trending topics*. We apply our method to the task of detecting trending topics and show its effectiveness by comparing our results to the official topics detected by Twitter. Our method can detect trends in advance of Twitter 79% of the time, with a mean early advantage of 1.43 hours, while maintaining a 95% true positive rate and a 4% false positive rate. Furthermore, we are able to do this using only a sample — 10% — of the Tweets in a period of time. Lastly, we show that our method is flexible and can be tuned to reflect a wide variety of tradeoffs between false positive rate, true positive rate, and relative detection time.

Chapter 2

Classification Method

2.1 Motivation

Suppose we have a space of objects Ω, a set of class labels Z, and a probability distribution μ on $\Omega \times Z$. For simplicity, we take the classes to be $+$ and $-$. Based on objects X and labels Y drawn from μ, we would like to learn a classification function that maps each object to its correct class label. This is the standard supervised learning problem.

Typically, when one wants to learn a model of how objects map to labels one selects some model space and chooses the best model from that model space, preferring a model that fits the data but is not too complex. The choice of model space encodes assumptions about the problem. For example, in the class of methods known as Tikhonov Regularization [18][4], of which Support Vector Machines and Regularized Least Squares are special cases, this choice manifests itself in the choice of a kernel and its associated Reproducing Kernel Hilbert Space (RKHS).

There are many ways to specify a model and therefore many types of model spaces. Tikhonov Regularization specifies the model as a function. It defines model complexity using the norm of the function in RKHS. Then it searches over the RKHS to find the best function. However, the model need not be specified by a function in a function space at all. It could be specified by a neural network with a certain architecture, or a spline with a certain number of nodes, or a boolean expression with a certain number of terms, or any number of other objects. Each of these settings have corresponding ways to determine model complexity and to do model selection.

With this in mind, we propose a setting for model specification and selection in supervised learning based on a *latent source model.* In this setting, the model is specified by a small collection of unknown *latent sources.* We posit that the data were generated by the latent sources according to a stochastic model relating latent sources and observations. However, rather than encoding any assumptions about the data via a choice of model space (e.g. all sets of 10 latent sources) and searching over the model space for the best set of latent sources, we rely directly on the data itself as a proxy for the unknown latent sources.

In other words, we are entirely unaware of the structure of the classification function. To resolve this, we propose the following nonparametric model relating observed objects to their labels. We posit that there are a relatively small number of distinct *latent source* objects in each class that account for all observed objects in that class. Let us call them $\mathbf{t}_1, \ldots, \mathbf{t}_n$ for $+$ and $\mathbf{q}_1, \ldots, \mathbf{q}_\ell$ for $-$. Each observation labeled $+$ is assumed to be a noisy version of one of the latent sources $\mathbf{t}_1, \ldots, \mathbf{t}_n$. Similarly, each observation labeled $-$ is assumed to be a noisy version of one of the latent sources $\mathbf{q}_1, \ldots \mathbf{q}_\ell$. We do not know what the latent source objects are or even how many there are. We only know the stochastic model that relates an observation to its latent source object.

2.2 Stochastic Model

To make the presentation more concrete, let us focus for the rest of this chapter on time-varying *signals* — the main objects of concern in this thesis. In this context, an observed object is simply a signal in a time window of a certain length. A latent source object may be thought of as a signal corresponding to a prototypical type of event. If the same type of event were to happen many times, we suppose that the resulting observed signals are noisy versions of the latent source signal corresponding to that type of event.

We say an observation s from an infinite stream s_∞ is *generated* by a latent source $\mathbf{q}$ if s is a noisy version of $\mathbf{q}$. Accordingly, we propose the following stochastic model relating a latent source $\mathbf{q}$ and an observation s:

$$\mathbb{P}(\mathrm{s} \text{ generated by } \mathbf{q}) \propto \exp\left(-\gamma d(\mathrm{s}, \mathbf{q})\right) \tag{2.1}$$

where $d(\mathrm{s}, \mathbf{q})$ is the *distance* between s and $\mathbf{q}$ and γ is a scaling parameter. This coincides with the notion that the closer an observation is to a latent source, the more likely it is that the observation came from that source. For example, a choice of distance function might be

$$d(\mathrm{s}, \mathbf{q}) = \sum_{i=1}^{N_{obs}} (s_i - q_i)^2 \tag{2.2}$$

for digital signals s and $\mathbf{q}$ of length N_{obs}. However, any symmetric, positive definite, and convex d would work.

2.3 Detection

2.3.1 Class Probabilities

Suppose that s is an observed signal of length N_{obs}. We would like to compute the probability that s belongs to each class. We can then use those probabilities to compute an estimate of the class of s. To compute the probability that s belongs to each class, we make use of a set of *reference* signals for each class — a set $\mathcal{R}_+$ of signals sampled from + and a set $\mathcal{R}_-$ of signals sampled from −. Reference signals represent historical data about previous activity from each class to which we can compare our observation and draw conclusions about which class it belongs to. We will assume that reference signals have length $N_{ref} \geq N_{obs}$. We will deal with the case of $N_{ref} = N_{obs}$ first and generalize in the following section.

Under our model the observation must belong to + if it has the same latent source as one of the reference signals in $\mathcal{R}_+$. Similarly, the observation must belong to −

if it has the same latent source as one of the reference signals in $\mathcal{R}_-$. Hence, the probability that the observation belongs to $+$ is

$$\begin{aligned}
\mathbb{P}(+ \mid \mathrm{s}) &\propto \sum_{\mathbf{r}\in\mathcal{R}_+} \mathbb{P}(\mathrm{s} \text{ belongs to } +, \mathrm{s} \text{ shares a latent source with } \mathbf{r}) \\
&= \sum_{\mathbf{r}\in\mathcal{R}_+} \sum_{j=1}^{n} \mathbb{P}(\mathrm{s} \text{ generated by } \mathbf{t}_j, \mathbf{r} \text{ generated by } \mathbf{t}_j) \\
&= \sum_{\mathbf{r}\in\mathcal{R}_+} \sum_{j=1}^{n} \exp\left(-\gamma d(\mathrm{s}, \mathbf{t}_j)\right) \exp\left(-\gamma d(\mathbf{r}, \mathbf{t}_j)\right) \\
&= \sum_{\mathbf{r}\in\mathcal{R}_+} \sum_{j=1}^{n} \exp\left(-\gamma \left(d(\mathrm{s}, \mathbf{t}_j) + d(\mathbf{r}, \mathbf{t}_j)\right)\right) \qquad (2.3)
\end{aligned}$$

For large enough γ, the term with the smallest exponent will dominate the sum over the latent sources and we can write the approximation

$$\mathbb{P}(+ \mid \mathrm{s}) \propto \sum_{\mathbf{r}\in\mathcal{R}_+} \exp\left(-\gamma \min_j \left(d(\mathrm{s}, \mathbf{t}_j) + d(\mathbf{r}, \mathbf{t}_j)\right)\right). \qquad (2.4)$$

However, the expression so far still involves a minimization over the unknown latent sources $\mathbf{t}_j$. We would like to eliminate the $\mathbf{t}_j$ altogether and just end up with a sum over all reference signals $\mathbf{r}$. If we suppose that the space of signals is reasonably well-covered by the latent sources, then the minimizing source $\mathbf{t}_{j^*}$ should be close to the global minimizer $\mathbf{t}^*$ over all signals. Figure 2-1 illustrates the reference signal $\mathbf{r}$, the observation s, the latent source signals $\mathbf{t}_1, \ldots, \mathbf{t}_n$, and the minimizing latent source signal $\mathbf{t}_{j^*}$.

The global minimizer $\mathbf{t}^*$ is simply the mean of s and $\mathbf{t}$. To see this, first observe that since $d(\mathrm{s}, \mathbf{t})$ and $d(\mathbf{r}, \mathbf{t})$ are convex in $\mathbf{t}$, $d(\mathrm{s}, \mathbf{t}) + d(\mathbf{r}, \mathbf{t})$ is also convex in $\mathbf{t}$. Second, let us assume that the distance function $d(\mathrm{s}, \mathbf{t})$ is actually a *norm* and therefore has the functional form $d(\mathrm{s}, \mathbf{t}) = c(\mathrm{s} - \mathbf{t})$, which depends only on the difference between

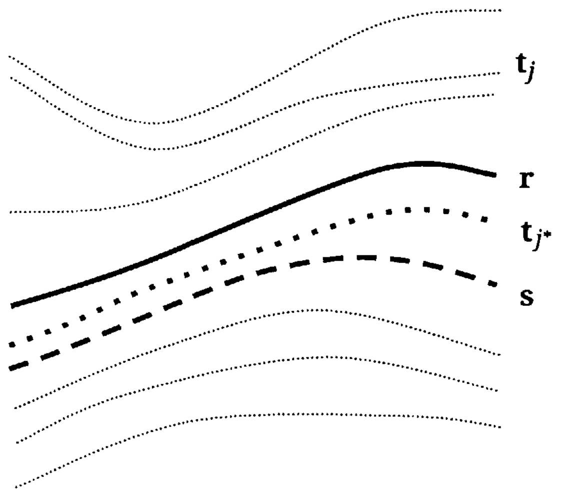


Figure 2-1: An illustration of the latent source signal $\mathbf{t}_{j^*}$ that minimizes $d(\mathrm{s},\mathbf{t}_j) + d(\mathbf{r},\mathbf{t}_j)$ in Eq. 2.4. Depicted are the reference signal $\mathbf{r}$, the observation s, the latent source signals $\mathbf{t}_1,\dots,\mathbf{t}_n$, and the minimizing latent source signal $\mathbf{t}_{j^*}$.

signals. Finally, observe that

$$\begin{aligned}\frac{\partial}{\partial \mathbf{t}}\left(d(\mathrm{s},\mathbf{t}) + d(\mathbf{r},\mathbf{t})\right)\Big|_{\mathbf{t}=\frac{\mathrm{s}+\mathbf{r}}{2}} &= \frac{\partial}{\partial \mathbf{t}}\left(c\left(\mathbf{t}-\mathrm{s}\right) + c\left(\mathbf{t}-\mathbf{r}\right)\right)\Big|_{\mathbf{t}=\frac{\mathrm{s}+\mathbf{r}}{2}} \\ &= c'\left(\frac{\mathrm{s}+\mathbf{r}}{2}-\mathrm{s}\right) + c'\left(\frac{\mathrm{s}+\mathbf{r}}{2}-\mathbf{r}\right) \\ &= c'\left(\frac{\mathbf{r}-\mathrm{s}}{2}\right) + c'\left(\frac{\mathrm{s}-\mathbf{r}}{2}\right) \\ &= 0 \end{aligned} \tag{2.5}$$

where in the last line, we have made use of the symmetry of c induced by the symmetry of d. Because $d(\mathrm{s},\mathbf{t}) + d(\mathbf{r},\mathbf{t})$ is convex in $\mathbf{t}$ and the derivative with respect to $\mathbf{t}$ at $\frac{\mathrm{s}+\mathbf{r}}{2}$ is zero, $\mathbf{t}^* = \frac{\mathrm{s}+\mathbf{r}}{2}$ is indeed the global minimizer.

Now, we can compute the corresponding global minimum $d(\mathrm{s},\mathbf{t}^*) + d(\mathbf{r},\mathbf{t}^*)$. The global minimum is

$$\begin{aligned} d(\mathrm{s},\mathbf{t}^*) + d(\mathbf{r},\mathbf{t}^*) &= d\left(\mathrm{s},\frac{\mathbf{r}+\mathrm{s}}{2}\right) + d\left(\mathbf{r},\frac{\mathbf{r}+\mathrm{s}}{2}\right) \\ &= c\left(\frac{\mathbf{r}-\mathrm{s}}{2}\right) + c\left(\frac{\mathrm{s}-\mathbf{r}}{2}\right) \end{aligned} \tag{2.6}$$

Let us also assume that for any reasonable distance function c, scaling the argument by a constant scales the distance according to

$$c(a\mathbf{x}) = g(a)c(\mathbf{x}) \tag{2.7}$$

where g is some positive definite function. Applying this to Eq. 2.6 gives

$$\begin{aligned} d(\mathbf{s}, \mathbf{t}^*) + d(\mathbf{r}, \mathbf{t}^*) &= g\left(\frac{1}{2}\right) c(\mathbf{r} - \mathbf{s}) + g\left(\frac{1}{2}\right) c(\mathbf{s} - \mathbf{r}) \\ &= 2 \cdot g\left(\frac{1}{2}\right) c(\mathbf{r} - \mathbf{s}) \\ &= 2 \cdot g\left(\frac{1}{2}\right) d(\mathbf{r}, \mathbf{s}) \\ &= C \cdot d(\mathbf{r}, \mathbf{s}). \end{aligned} \tag{2.8}$$

where C is a constant independent of $\mathbf{r}$ and $\mathbf{s}$.

Having done this, we can now approximate $\min_j(d(\mathbf{s}, \mathbf{t}_j) + d(\mathbf{r}, \mathbf{t}_j))$ in Eq. 2.4 by $C \cdot d(\mathbf{s}, \mathbf{r})$, assuming the actual minimizing latent source t_{j^*} is close to the global minimizer $\mathbf{t}^*$. This gives us the probability that the observation belongs to $+$ without having to minimize over the unknown $\mathbf{t}_j$:

$$\begin{aligned} \mathbb{P}(+ \mid \mathbf{s}) &\propto \sum_{\mathbf{r} \in \mathcal{R}_+} \exp\left(-\gamma \min_j \left(d(\mathbf{s}, \mathbf{t}_j) + d(\mathbf{r}, \mathbf{t}_j)\right)\right) \\ &\approx \sum_{\mathbf{r} \in \mathcal{R}_+} \exp\left(-\gamma d(\mathbf{s}, \mathbf{r})\right) \end{aligned} \tag{2.9}$$

where we have absorbed C into γ for convenience. We can similarly compute the probability that the observation belongs to $-$:

$$\mathbb{P}(- \mid \mathbf{s}) \propto \sum_{\mathbf{r} \in \mathcal{R}_-} \exp\left(-\gamma d(\mathbf{s}, \mathbf{r})\right). \tag{2.10}$$

2.3.2 Class Estimator

Our class estimation rule is simple: assign to the observation the class with the highest probability. In practice, we compute the ratio of $\mathbb{P}(+ \mid \mathbf{s})$ and $\mathbb{P}(- \mid \mathbf{s})$

$$R(\mathbf{s}) = \frac{\mathbb{P}(+ \mid \mathbf{s})}{\mathbb{P}(- \mid \mathbf{s})} = \frac{\sum\limits_{\mathbf{r} \in \mathcal{R}_+} \exp\left(-\gamma d(\mathbf{s}, \mathbf{r})\right)}{\sum\limits_{\mathbf{r} \in \mathcal{R}_-} \exp\left(-\gamma d(\mathbf{s}, \mathbf{r})\right)} \tag{2.11}$$

and check if it exceeds a threshold of $\theta = 1$. For a quadratic distance function, this becomes

$$R(\mathbf{s}) = \frac{\sum\limits_{\mathbf{r} \in \mathcal{R}_+} \exp\left(-\gamma \sum\limits_{i=1}^{N_{obs}} (s_i - r_i)^2\right)}{\sum\limits_{\mathbf{r} \in \mathcal{R}_-} \exp\left(-\gamma \sum\limits_{i=1}^{N_{obs}} (s_i - r_i)^2\right)} \tag{2.12}$$

The estimator for the class label L is therefore

$$\hat{L}(\mathbf{s}) = \begin{cases} + & \text{if } R(\mathbf{s}) > \theta \\ - & \text{if } R(\mathbf{s}) \leq \theta. \end{cases} \tag{2.13}$$

In practice, values other than 1 may also be used for the threshold θ. For example, if the benefits of true positives outweigh the costs of false positives, one may set θ to less than 1. On the other hand, if the costs of false positives outweigh the benefits of true positives, one may conservatively set θ to greater than 1. We explore this effect in Chapter 5.

Accumulating Evidence

In some cases, for example when dealing with noisy data, it could be advantageous to accumulate evidence over multiple time steps before determining which class the observation belongs to. A simple extension of our algorithm would be to require that the observation is judged to belong to a particular class for several consecutive time steps. We shall call this number of required time steps D_{req}. We study the effect of D_{req} in Chapter 5.

2.3.3 Online Classification

In the previous sections, we have assumed that the reference signals and the observations have the same length. In the online classification setting, it is convenient to extend this to reference signals of arbitrary length. At its core, our method compares an observation — recently observed measurements of some property of a system — to reference signals — sets of historical measurements of that property for each class. Recently observed measurements are judged to belong to the class whose reference signals they most resemble. For reference signals and observations of the same size, this resemblance is computed using the distance function d previously described. In practice, however, there are two complications. The first complication is that observations will generally be short, containing a small amount of recent samples, and reference signals will be long, containing large amounts of historical data. The second complication is that reference signals will often have unknown phase. That is, the events underlying the reference signals may have occurred at arbitrary time shifts with respect to one another. Furthermore, when comparing the observation to each reference signal, there is no temporal point of reference between the two. A natural solution to both of these complications is to check whether the observation resembles any *piece* of the reference signal of the same size as the observation. Figure 2-2 illustrates this.

We generalize the distance function to reflect this notion. Let us assume for simplicity that all observations are of length N_{obs} and all reference signals are of length $N_{ref} \geq N_{obs}$. We define the distance between a reference signal $\mathbf{r}$ and an observation $\mathbf{s}$ as the minimum distance between $\mathbf{s}$ and all contiguous subsignals of $\mathbf{r}$ of length N_{obs}.

$$d(\mathbf{r}, \mathbf{s}) = \min_{k=1,\ldots,N_{ref}-N_{obs}+1} d(\mathbf{r}_{k:k+N_{obs}-1}, \mathbf{s}) \tag{2.14}$$

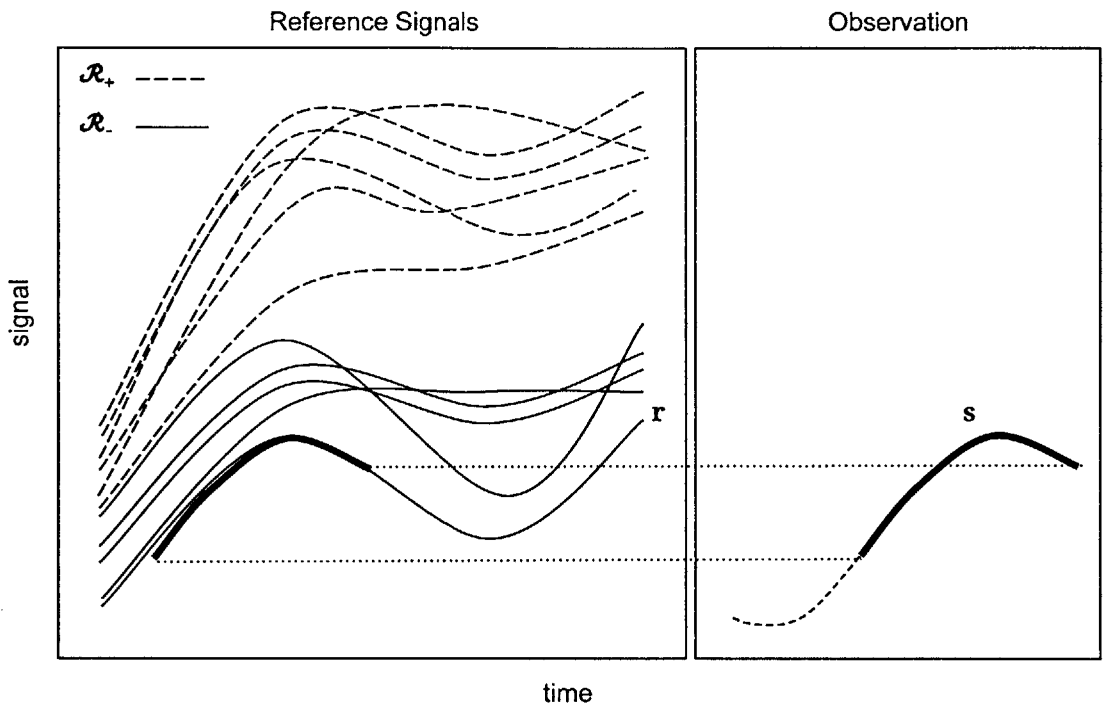


Figure 2-2: To compare a long reference signal to a short observation, we compute the distance between the observation (right) and the closest *piece* of a reference signal (left).

Conveniently, for $N_{ref} = N_{obs}$, this new distance function reduces to the distance function previously defined. Finally, using the generalized version of d, the ratio of class probabilities $R(\mathrm{s})$ from the previous section becomes

$$R(\mathrm{s}) = \frac{\displaystyle\sum_{\mathrm{r}\in\mathcal{R}_+} \exp\left(-\gamma \min_{k=1,\ldots,N_{ref}-N_{obs}+1} \sum_{i=1}^{N_{obs}} (s_i - r_{i+k-1})^2\right)}{\displaystyle\sum_{\mathrm{r}\in\mathcal{R}_-} \exp\left(-\gamma \min_{k=1,\ldots,N_{ref}-N_{obs}+1} \sum_{i=1}^{N_{obs}} (s_i - r_{i+k-1})^2\right)}. \tag{2.15}$$

Chapter 3

Algorithm

In this chapter, I describe the practical implementation of the method described in Chapter 2.

3.1 Overview

The goal of the algorithm is to perform online classification of an infinite stream of samples from an observed digital signal. We will focus on the case of binary classification, in which we have positive signals and negative signals, but the results can be extended to multiple classes. For binary classification, one could imagine that one class represents events and the other non-events, and that we would like to detect events as soon as they happen.

To predict which class the observed signal belongs to at a given point in time, we compute the probability that the recent samples of the observed signal were generated by a latent source from the positive class and the probability that they were generated by a latent source from the negative class, based on previously observed *reference signals* for each class. Recall from Chapter 2 that a signal is generated by a latent source of a particular class if it shares a latent source with some reference signal from that class. To compute the probability that the recently observed samples share the same latent source with a particular reference signal, we compute the distance between the trajectory consisting of the recently observed samples and all trajectories of the same size in the reference signal, and take the minimum over all such trajectories.

3.2 Implementation

In practice, the computation of conditional class probabilities amounts to nothing more than computing distances. To compute the probability that an observation belongs to a particular class, one simply computes the distance from the observation to each reference signal in that class in order to see how much the observation resembles the reference signals for that class.

Algorithm 1 contains the core detection logic. At each time step, it updates the

Algorithm 1 Perform online binary classification on the infinite stream $\mathbf{s}_\infty$ using sets of positive and negative reference signals R_+ and R_-.

```
DETECT($\mathbf{s}_\infty$, $\mathcal{R}_+$, $\mathcal{R}_-$, $\gamma$, $\theta$, $D_{req}$):
 1: $ConsecutiveDetections \leftarrow 0$
 2: loop
 3:   $\mathbf{s} \leftarrow$ UPDATEOBSERVATION($\mathbf{s}_\infty$, $N_{obs}$)
 4:   for $\mathbf{r}$ in $\mathcal{R}_+$ do
 5:     $PosDists$.APPEND(DISTTOREFERENCE($\mathbf{s}$, $\mathbf{r}$))
 6:   end for
 7:   for $\mathbf{r}$ in $\mathcal{R}_-$ do
 8:     $NegDists$.APPEND(DISTTOREFERENCE($\mathbf{s}$, $\mathbf{r}$))
 9:   end for
10:   $R$ = PROBCLASS($PosDists$, $\gamma$) / PROBCLASS($NegDists$, $\gamma$)
11:   if $R > \theta$ then
12:     if $ConsecutiveDetections > D_{req}$ then
13:       $DetectionTime \leftarrow$ CURRENTTIME()
14:       return $DetectionTime$
15:     else
16:       $ConsecutiveDetections \leftarrow ConsecutiveDetections + 1$
17:     end if
18:   else
19:     $ConsecutiveDetections \leftarrow 0$
20:   end if
21: end loop
```

observation $\mathbf{s}$ so that $\mathbf{s}$ contains the latest N_{obs} samples from the infinite stream $\mathbf{s}_\infty$ and computes the distances *PosDists* (resp. *NegDists*) to reference signals of the positive class (resp. negative class). A detection is declared whenever the ratio of class probabilities $R(\mathbf{s})$ exceeds the threshold θ for D_{req} consecutive time steps.

Algorithm 2 computes the distance between a reference signal $\mathbf{r}$ and an observation $\mathbf{s}$. Since the reference signal is generally longer than the observation, we compute the

Algorithm 2 Compute the minimum distance between $\mathbf{s}$ and all pieces of $\mathbf{r}$ of the same length as $\mathbf{s}$.

```
DistToReference(s, r):
1: $N_{obs} \leftarrow$ LENGTH(s)
2: $N_{ref} \leftarrow$ LENGTH(r)
3: $MinDist = \infty$
4: for $i = 1$ to $N_{ref} - N_{obs} + 1$ do
5:     $MinDist = $ Min($MinDist$, Dist($\mathbf{r}_{i:i+N_{obs}-1}$, $\mathbf{s}$))
6: end for
7: return $MinDist$
```

minimum distance (Algorithm 3) across all pieces of $\mathbf{r}$ of the same size as $\mathbf{s}$.

Algorithm 3 simply computes the Euclidean distance between two signals of the same size.

Algorithm 3 Compute the distance between two signals $\mathbf{s}$ and $\mathbf{t}$ of the same length

```
Dist(s, t):
1: $D \leftarrow 0$
2: for $i = 1$ to LENGTH(s) do
3:     $D \leftarrow D + (s_i - t_i)^2$
4: end for
5: return $D$
```

Using the distances from an observation to the reference signals of a class, we compute a number proportional the probability that the observation belongs to the class (Algorithm 4).

Algorithm 4 Using the distances of an observation to the reference signals of a certain class, compute a number proportional to the probability that the observation belongs to that class.

```
ProbClass(Dists, $\gamma$):
1: $P \leftarrow 0$
2: for $i = 1$ to Length(Dists) do
3:     $P \leftarrow P + \exp(-\gamma Dists_i)$
4: end for
5: return $P$
```

3.3 Performance and Scalability

To do detection on an infinite stream for T time steps, with $|\mathcal{R}_+|$ positive reference signals and $|\mathcal{R}_-|$ negative reference signals of length N_{ref}, and observations of length N_{obs}, our rudimentary implementation runs in $\mathcal{O}(TN_{ref}(|\mathcal{R}_-| + |\mathcal{R}_-|))$ time. In practice, the algorithm can be made faster by a constant factor by not performing detection on every time step, not computing distances based on the full N_{obs} samples, or not comparing the observation to every single slice of the reference signal.

Clearly, the computational cost of our implementation grows with the amount of data. Nevertheless, our approach is scalable, since one can compute in parallel the scores for each of the topics, as well as each of the reference signal distances for each topic.

A more sophisticated version of our algorithm would use an approach based on time series indexing. For instance, Rakthanmanon et al. have shown a way to efficiently search over trillions of time series subsequences [19]. Since our probability-based metric involves exponential decay based on the distance between signals, most reference signals that are far away from the observation can safely be ignored. Thus, instead of computing the distance to all reference signals, which could become costly, we can operate on only a very small fraction of them without significantly affecting the outcome.

Chapter 4

Application: Identifying Trending Topics on Twitter

In this chapter, we consider the application of the method and algorithm proposed in Chapters 2 and 3 toward detection of trending topics on Twitter. We discuss the Twitter service, the collection and pre-processing of data, and the experimental setup for the detection task.

4.1 Overview

4.1.1 Overview of Twitter

Twitter is a real-time messaging service and information network. Users of Twitter can post short (up to 140 characters) messages called *Tweets*, which are then broadcast to the users' *followers*. Users can also engage in conversation with one another. By default, Tweets are public, which means that anyone can see them and potentially join a conversation on a variety of topics being discussed. Inevitably, some topics gain relatively sudden popularity on Twitter. For example, a popular topic might reflect an external event such as a breaking news story or an internally generated inside joke or game. Twitter surfaces such topics in the service as a list of top ten trending topics.

4.1.2 Twitter-Related Definitions

Talking about Tweets, topics, trends and trending topics can be ambiguous, so here we make precise our usage of these and related terms.

Definition 1 (Topic). *We define a* **topic** *to be a phrase consisting of one or more* **words** *delimited by spacing or punctuation. A word may be any sequence of characters and need not be an actual dictionary word.*

Definition 2 (Tweet about topic). *A Tweet is* **about** *a topic if it contains the topic as a substring. Tweets can be about many topics.*

Example 1. *The following tweet by the author (handle @snikolov) contains the string "matlab." Hence, it is considered to be* about *the topic "matlab."*

> *"matlab symbolic eigendecomposition. expressions with 25000+ characters are not always the most interpretable thing."*

Definition 3 (Trending topic). *A* **trending topic** *is a topic that is currently on the list of trending topics on Twitter. If a topic was ever a trending topic during a period of time, we say that the topic* **trended** *during that time period.*

Definition 4 (Trend). *A trending topic will also occasionally be referred to as a* **trend** *for short.*

Definition 5 (Trend onset). *The* **trend onset** *is the time that a topic first trended during a period of time.*

Example 2. *If the topic "earthquake" is currently in the trending topics list on Twitter, we say that "earthquake" is* trending, *and that earthquake is a* trend. *The topic "earthquake" has a* trend onset, *which is the first time it was trending in a given period of time. This could, for example, correspond to when the earthquake happened. After "earthquake" is no longer trending, we say that "earthquake"* trended.

4.1.3 Problem Statement

At any given time there are many topics being talked about on Twitter. Of these, some will trend at some point in the future and others will not. We wish to predict which topics will trend. The earlier we can predict that a topic will trend, the better. Ideally, we would like to do this while maintaining a low rate of error (false detections and false non-detections).

4.1.4 Proposed Solution

Our approach to detecting trending topics is as follows. First, we gather examples of topics that trended and topics that did not trend during some period of time. Then, for each topic, we collect Tweets about that topic and generate a time series of the activity of that topic over time. We then use those time series as *reference signals* (cf. Chapter 2) and apply the classification method and algorithm described in Chapters 2 and 3.

4.2 Data

4.2.1 Data Collection

The online time series classification method detailed in Chapters 2 and 3 requires a set of reference signals corresponding to topics that trended and a set of reference signals corresponding to topics that did not trend during a time window of interest. These reference signals represent historical data against which we can compare our most recent observations to do classification.

The data collection process can be summarized as follows. First, we collected 500 examples of topics that trended at least once between June 1, 2012 and June 30, 2012 (hereafter referred to as the *sample window*) and 500 examples of topics that never trended during the sample window. We then sampled Tweets from the sample window and labeled each Tweet according to the topics mentioned therein. Finally, we constructed a reference signal for each topic based on the Tweet activity corresponding to that topic.

We obtained all data directly from Twitter via the MIT VI-A thesis program. However, the type as well as the amount of data we have used is all publicly available via the Twitter API.

Topics

We collected a list of all trending topics on Twitter from June 1, 2012 to June 30, 2012 (the *sample window*) as well as the times that they were trending and their rank in the trending topics list on Twitter. Of those, we filtered out topics whose rank was never better than or equal to 3. In addition, we filtered out topics that did not trend for long enough (the time of the first appearance to the time of the last appearance is less than 30 minutes) as well as topics that reappear multiple times during the sample window (the time of the first appearance to the time of the last appearance is greater than 24 hours). The former eliminates many topics that are spurious and only trend for a very short time. The latter eliminates topics that correspond to multiple events. For example, the name of a football player might trend every time there is an important game. We would like to avoid such ambiguity and restrict each trending topic to correspond to a single underlying event within the sample window.

We collected topics that did not trend during the sample window in two steps. First, we sampled a list of n-grams (phrases consisting of n "words") occurring on Twitter during the sample window for n up to 5. We filtered out n-grams that contain any topic that trended during the sample window, using the original, unfiltered list of all topics that trended during the sample window. For example, if "Whitney Houston" trended during the sample window, then "Whitney" would be filtered out of the list of topics that did not trend. We also removed n-grams shorter than three characters, as most of these did not appear to be meaningful topics. Lastly, we sampled 500 n-grams uniformly from the filtered list of n-grams to produce the final list.

Tweets

We sampled 10% of all public Tweets from June 1, 2012 to June 30, 2012 inclusive. We labeled each Tweet with the topic or topics contained therein using a simple regular expression match between the Tweet text and the topic text. In addition to the Tweet text, we recorded the date and time the Tweet was authored.

4.2.2 From Tweets to Signals

We discuss the process of converting the timestamped Tweets for a given topic into a reference signal. Each of the steps below is followed in order for each topic.

Tweet Rate

As a first step toward converting timestamped Tweets about a topic into a signal, we bin the Tweets into time bins of a certain length. We use time bins of length two minutes. Let $\rho[n]$ be the number of Tweets about a topic in the nth time bin. Let the cumulative volume of of Tweets up to time n be

$$v[n] = \sum_{m \leq n} \rho[m] \tag{4.1}$$

Thus, effectively $\rho[n]$ is the discrete derivative of the continuous cumulative volume $v(t)$ over time i.e. $\rho(t) = \dot{v}(t)$. Therefore, we shall call $\rho[n]$ the rate of the signal at time step n. Figure 4-1 shows this rate for a topic over the whole sample window.

Baseline Normalization

A first glance at the data reveals that many non-trending topics have a relatively high rate and volume of Tweets, and many trending topics have a relatively low rate and volume of Tweets. One important difference is that many non-trending topics have a high *baseline rate* of activity while most trending topics are preceded by little, if any, activity prior to gaining sudden popularity. For example, a non-trending topic such as 'city' is likely to have a consistent baseline of activity because it is a common

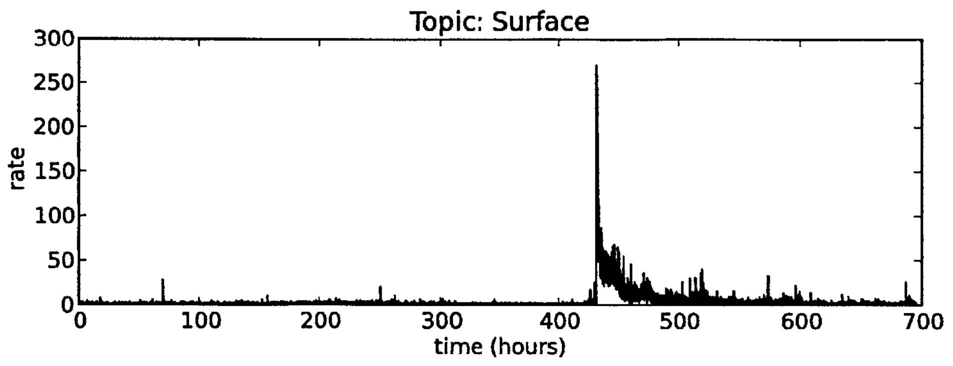


Figure 4-1: The rate of a topic over the sample window.

word. To emphasize the parts of the rate signal above the baseline and de-emphasize the parts below the baseline, we define a baseline b as

$$b = \sum_{n} \rho[n] \tag{4.2}$$

and a baseline-normalized signal ρ_b as

$$\rho_b[n] = \left(\frac{\rho[n]}{b}\right)^{\beta}. \tag{4.3}$$

The exponent β controls how much we reward and penalize rates above and below the baseline rate. In this thesis, we use $\beta = 1$. Figure 4-2 shows rate signals without any baseline normalization and their baseline-normalized versions.

Spike Normalization

Another difference between the rates of Tweets for trending topics and that of non-trending topics is the number and magnitude of spikes. The Tweet rates for trending topics typically contains larger and more sudden spikes than that of non-trending topics. We reward such spikes by emphasizing them, while de-emphasizing smaller spikes. To do so, we define a baseline-and-spike-normalized rate

$$\rho_{b,s}[n] = \left|\rho_b[n] - \rho_b[n-1]\right|^{\alpha} \tag{4.4}$$

in terms of the already baseline-normalized rate ρ_b. The parameter $\alpha > 1$ controls how much spikes are rewarded. We use $\alpha = 1.2$. Figure 4-2 shows the effect of this spike-based transformation.

Smoothing

Tweet rates, and the aforementioned transformations thereof, tend to be noisy, especially for small time bins. To mitigate this, we convolve the signal with a smoothing window of size N_{smooth}. Applied to the spike-and-baseline-normalized signal $\rho_{b,s}$, this yields the convolved version

$$\rho_{b,s,c}[n] = \sum_{m=n-N_{smooth}+1}^{n} \rho_{b,s}[m]. \tag{4.5}$$

Figure 4-3 shows the effect of smoothing with various window sizes.

Branching Processes and Logarithmic Scale

It is reasonable to think of the spread of a topic from person to person as a branching process. A branching process is a model of the growth of a population over time, in which each individual of a population in a given generation produces a random number of individuals in the next generation. While we do not know the details of how a topic spreads, we do know that in a wide generality of branching processes, the growth of the population is exponential with time, with the exponent depending on the details of the model [3]. It is reasonable, then, to measure the volume of tweets at a logarithmic scale to reveal these details. Asur et al. confirm that the spread of topics on Twitter can be modeled as a branching process and also propose a logarithmic scaling [2]. Therefore, as a final step, we take the logarithm of the signal constructed so far to produce the signal

$$\rho_{b,s,c,l}[n] = \log \rho_{b,s,c}[n]. \tag{4.6}$$

Figure 4-4 shows a sample of signals and their log-scaled versions.

Constructing a Reference Signal

The signal $\rho_{b,s,c,l}[n]$ resulting from the steps so far is as long as the entire time window from which all Tweets were sampled. Such a long signal is not particularly useful as a reference signal. Recall from chapter 3 that to see how much the recent trajectory of the observed signal resembles part of a reference signal, we have to traverse the full length of the reference signal in order to find the piece that most closely resembles the recent observed trajectory. If the reference signal for topic that trended spans too long of a time window, only a small portion of it will represent activity surrounding the onset of the trend. In addition, it is inefficient to compare the recently observed trajectory to a reference signal that is needlessly long. Hence, it is necessary to select a small slice of signal from the much longer rate signal. In the case of topics that trended, we select a slice that terminates at the first onset of trend. That way, we capture the pattern of activity leading up to the trend onset, which is crucial for recognizing similar pre-onset activity in the observed signal. We do not include activity after the true onset because once the topics is listed in the trending topics list on Twitter, we expect the predominant mode of spreading to change. For topics that did not trend, we assume that the rate signal is largely stationary and select slices with random start and end times. For simplicity, all slices are a fixed size.

Figures 4-2 through 4-4 show the samples of reference signals with various combinations of the transformations described in this section.

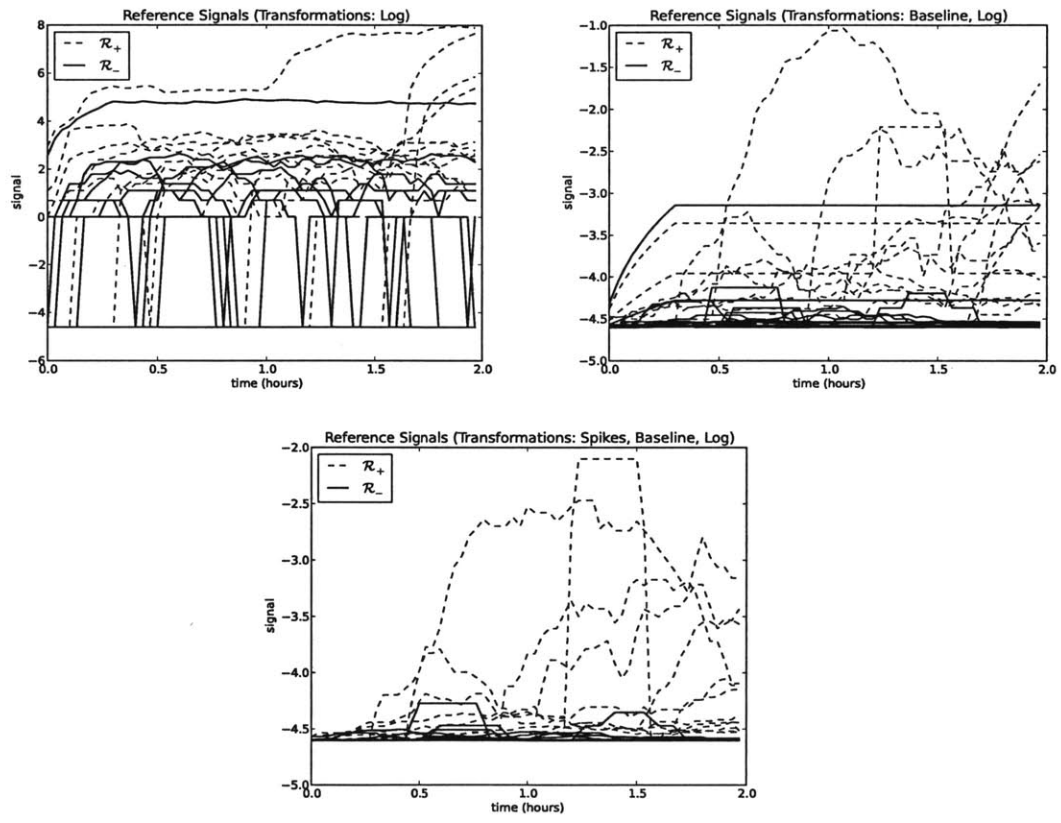


Figure 4-2: Reference signals of either class are hard to tell apart without normalization. **Top left**: no baseline or spike normalization. **Top right**: Baseline normalization. **Bottom**: Baseline and spike-based normalization.

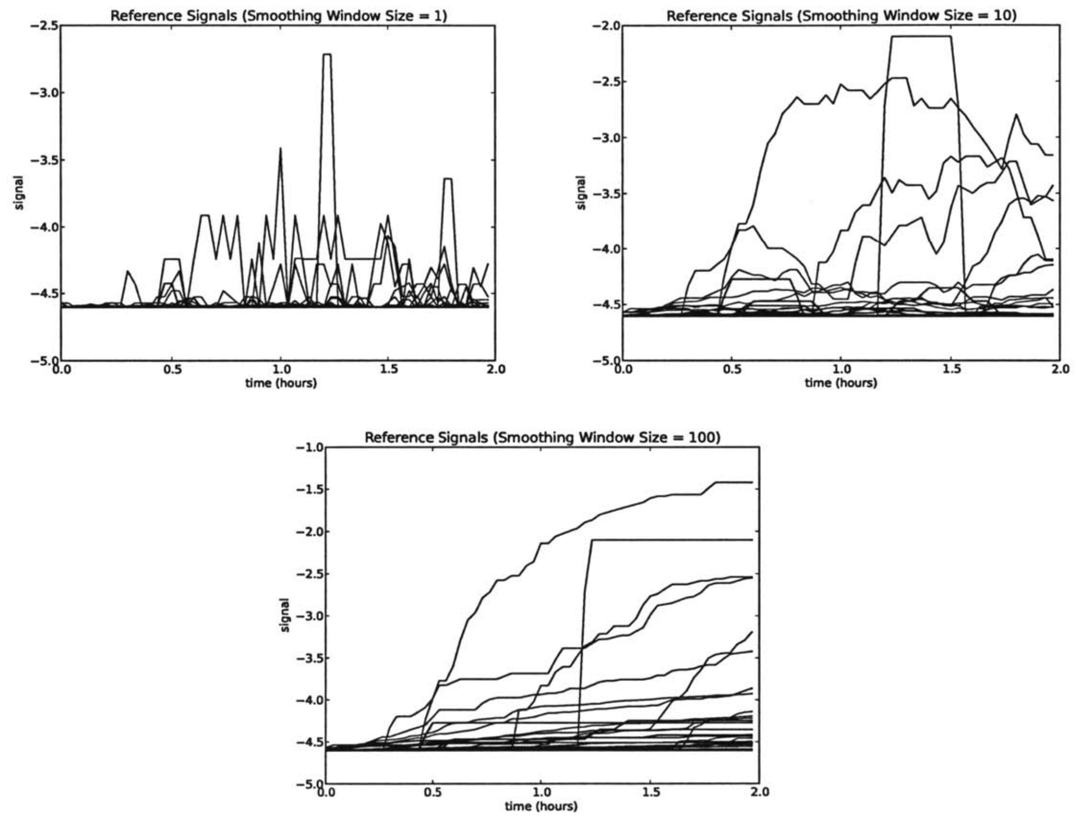


Figure 4-3: The results of smoothing the reference signals (with spike and baseline normalization previously applied) with windows of size 1 (2 minutes, i.e. no smoothing), 10 (20 minutes), and 100 (3 hours, 20 minutes).

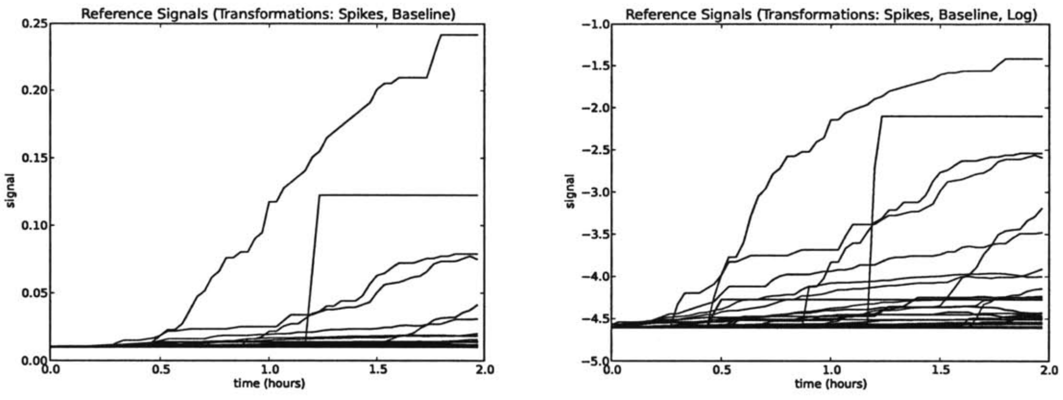


Figure 4-4: Logarithmically scaled reference signals (with spike and baseline normalization previously applied) allow one to make finer-grained distinctions between signals. **Left**: Not logarithmically scaled. **Right**: Logarithmically scaled.

4.3 Experiment

We propose an experiment to measure our algorithm's performance on two fronts: error rate and relative detection time. We divide the set of topics into a training set and a test set using a 50/50 split. For each topic in the test set, we wish to predict if the topic will trend. If the topic really did trend, we wish to detect it as early as possible relative to the true trend onset while incurring minimal error.

4.3.1 Detection Setup

In principle, to test the detection algorithm, one would step through the signal in the entire sample window for each topic in the test set and report the time of the first detection, or that there were no detections. In practice, we take a shortcut to avoid looking through the entire signal based on the following observations about the activity of topics that trended and topics that did not. First, for topics that trended, there is little, if any activity aside from that surrounding the true onset of the trend. In the rare event that a detection is made very far from the true onset, it is reasonable to assume that this corresponds to a completely different event involving that topic and we can safely ignore it. Thus, the only part of the signal worth looking at is the signal within some time window from the true onset of the trend. Second, topics that did not trend exhibit relatively stationary activity. That is, the signal usually looks roughly the same over the entire sample window. Therefore, it is reasonable to perform detection only on a piece of the signal as an approximation to the true detection performance.

We perform detection over a window of $2N_{obs}$ samples — twice the length of a reference signal. For convenience and future use, we define this in terms of hours.

Definition 6. *Let h_{ref} be the number of hours corresponding to N_{ref} samples. At 2 minutes per sample, h_{ref} is given by $N_{ref}/30$.*

For test topics that have trended, we do detection on the window spanning $2h_{ref}$ hours centered at the true trend onset. For topics that did not trend, we randomly choose a window of the desired size. Note that, although this seems to require *a priori* knowledge of whether the test topic ever trended or not, this is only a consequence of the shortcut we take to not do detection over the entire sample window.

4.3.2 Parameter Exploration and Trials

We explore all combinations of the following ranges of parameters, excluding parameter settings that are incompatible (e.g. $N_{obs} > N_{ref}$). For each combination, we conducted 5 random trials.

- γ: 0.1, 1, 10.
- N_{obs}: 10, 80, 115, 150.
- N_{smooth}: 10, 80, 115, 150.
- h_{ref}: 3, 5, 7, 9.
- D_{req}: 1, 3, 5.
- θ: 0.65, 1, 3.

4.3.3 Evaluation

To evaluate the performance of our method, we compute the false positive rate and true positive rate for each experiment, averaged over all trials. In the case of true detections, we compute the detection time relative to the true onset of the trending topic.

Before presenting the results in the following chapter, we note the following important difference between the general detection method employed herein and that of Twitter. Twitter produces a list of top ten trending topics, while we perform detection based on a score and a threshold, and do not limit the number of topics detected as trending at any given time. This could cause noticeable discrepancies between the topics detected. For example, an otherwise popular emerging topic might not be detected as a trend if there are many other important topics being discussed at the moment. Despite these differences, we show in the next chapter that our algorithm can achieve good performance relative to that of Twitter.

Chapter 5

Results and Discussion

In this chapter, we present the results of the trend detection experiment described in Chapter 4. We show the quality of the trend detection algorithm using ROC curves and distributions of detection time relative to the true trend onset. We analyze the effect of the algorithm parameters on the tradeoff between false positive rate, true positive rate, and relative detection time. Finally, we propose parameter regimes appropriate for three situations: 1) the cost of a false positive outweighs the cost of a false negative, 2) the cost of a false negative outweighs the cost of a false positive, and 3) the costs of a false positive and a false negative are comparable.

5.1 Summary of Results

In this section we show that we are able to detect trending topics before they are detected by Twitter while maintaining a low error rate. We showcase the flexibility of our algorithm and investigate the effects of parameters. In particular, we show that our algorithm accommodates a wide range of tradeoffs between true positive rate, false positive rate and relative detection time.

5.1.1 Main Result: Early Detection of Trending Topics

Our main result is that for a range of parameters, we are able to detect trending topics before they appear on Twitter's trending topics list. Figure 5-1 shows that for the given parameter setting, we are able to detect trending topics before Twitter does 79% of the time, and when we do, we detect them an average of 1.43 hours earlier. Furthermore, we achieve a low error rate: a true positive rate of 95% and a false positive rate of just 4%. Naturally, there are tradeoffs between false positive rate,

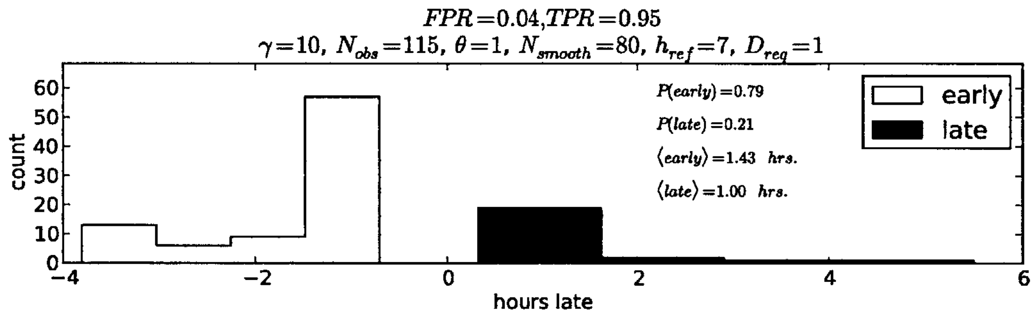


Figure 5-1: Our algorithm is able to achieve a low error rate while detecting trending topics in advance of Twitter a large percentage of the time.

true positive rate, and relative detection time. We explore these relationships in the following sections.

5.1.2 ROC Curve Envelopes

By varying a single parameter and keeping the rest fixed, we generate a Receiver Operating Characteristics (ROC) curve that describes the tradeoff between False Positive Rate (FPR) and True Positive Rate (TPR). Figures 5-3 and 5-2 show the ROC curves that result from varying each detection parameter, aggregated over all combinations of the remaining parameters. The left side of each plot shows all ROC curves for a given variable parameter overlaid on a single set of axes. The right side shows the upper-right-most envelope of those ROC curves, representing the best-case, or achievable ROC curve.

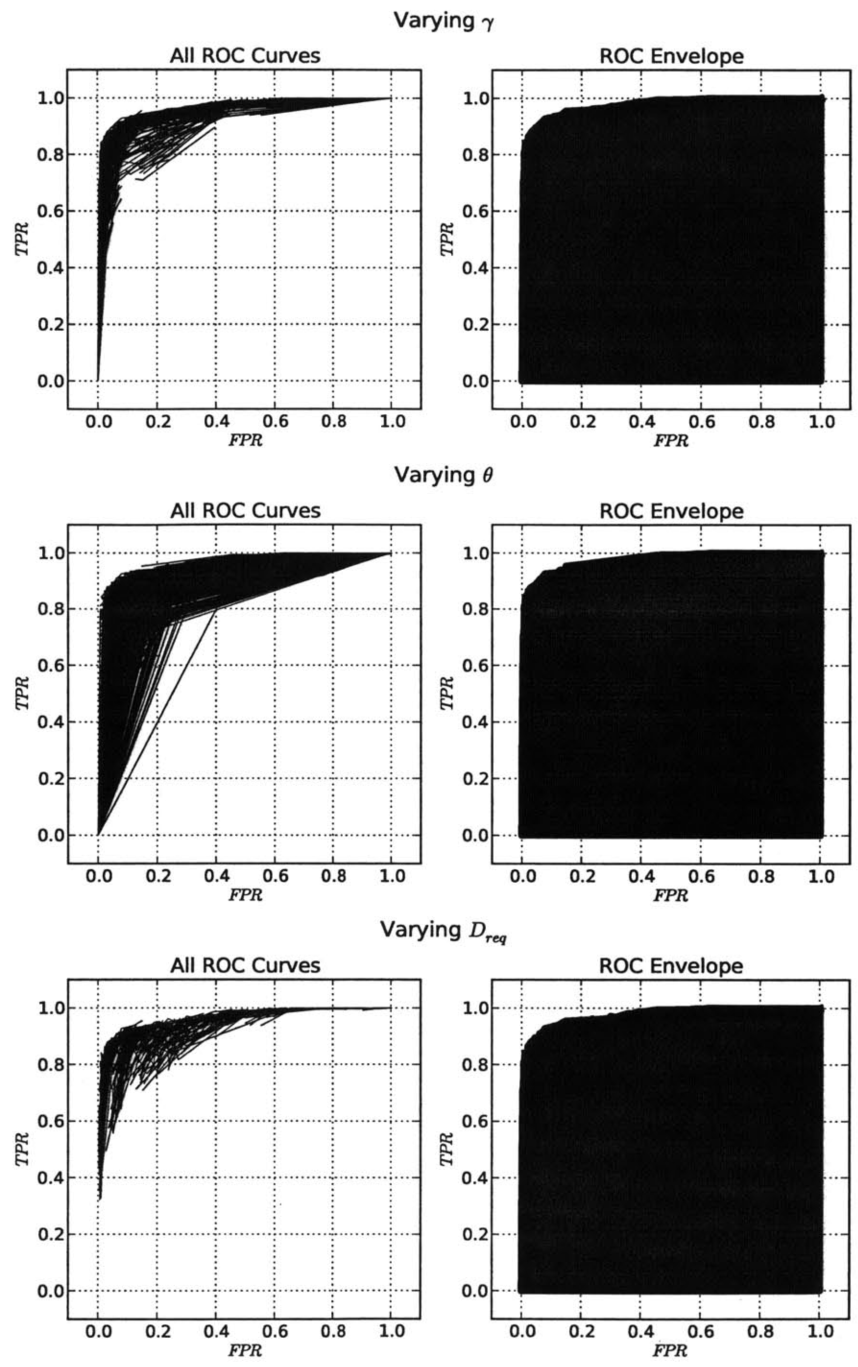


Figure 5-2: **Left**: All ROC curves for the given variable parameter overlaid on a single set of axes. **Right**: The upper-right-most envelope of those ROC curves, representing the best-case, or achievable ROC curve.

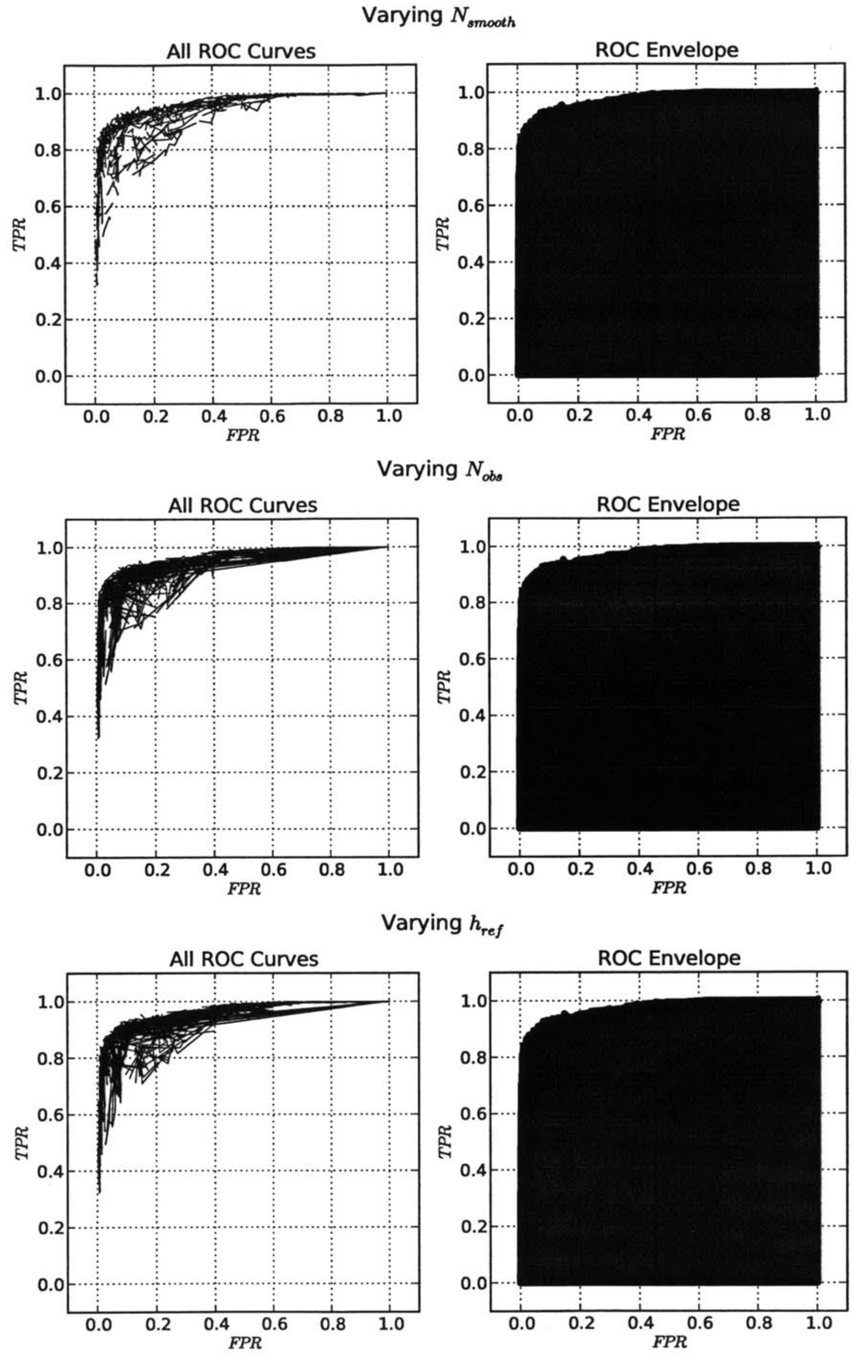


Figure 5-3: **Left**: All ROC curves for the given variable parameter overlaid on a single set of axes. **Right**: The upper-right-most envelope of those ROC curves, representing the best-case, or achievable ROC curve.

It is evident by the length of the ROC curves for each parameters that some parameters have a greater effect in trading off between TPR and FPR. In Section 5.2, we investigate these effects in detail. However, in aggregate, the ranges of the parameters studied allow a full range of tradeoffs between TPR and FPR, and show the flexibility of our approach.

5.1.3 Effect of FPR and TPR on Relative Detection Time

Figure 5-4 shows the effect of position along the ROC curve on the relative detection times of our algorithm compared to Twitter's trend detection algorithm. To simplify our analysis, we break the ROC curve into three regions: the *top* region, referring to the upper right corner of the curve, the *center* region, referring to the upper left corner of the curve, and the *bottom* region, referring to the bottom left corner of the curve. More precisely, we define the regions as follows.

Definition 7 (Top region). *(FPR,TPR) is in the top region if* $FPR > 0.25$ *and* $TPR > 0.75$.

Definition 8 (Center region). *(FPR,TPR) is in the top region if* $FPR \leq 0.25$ *and* $TPR > 0.75$.

Definition 9 (Bottom region). *(FPR,TPR) is in the top region if* $FPR \leq 0.25$ *and* $TPR \leq 0.75$.

In the top region, we accept the possibility of frequent false detections for the sake of rarely missing the chance to make a true detection. In the bottom region, we accept a lower chance of making a true detection for the sake of rarely making false detections. The center region lies in between these two extremes. Consequently, points in the top region correspond to earlier detection relative to the true onset of a trend as detected by Twitter, points in the bottom, correspond to predominantly late detection, and points in the center roughly balance being early and late. Figure 5-4 illustrates this.

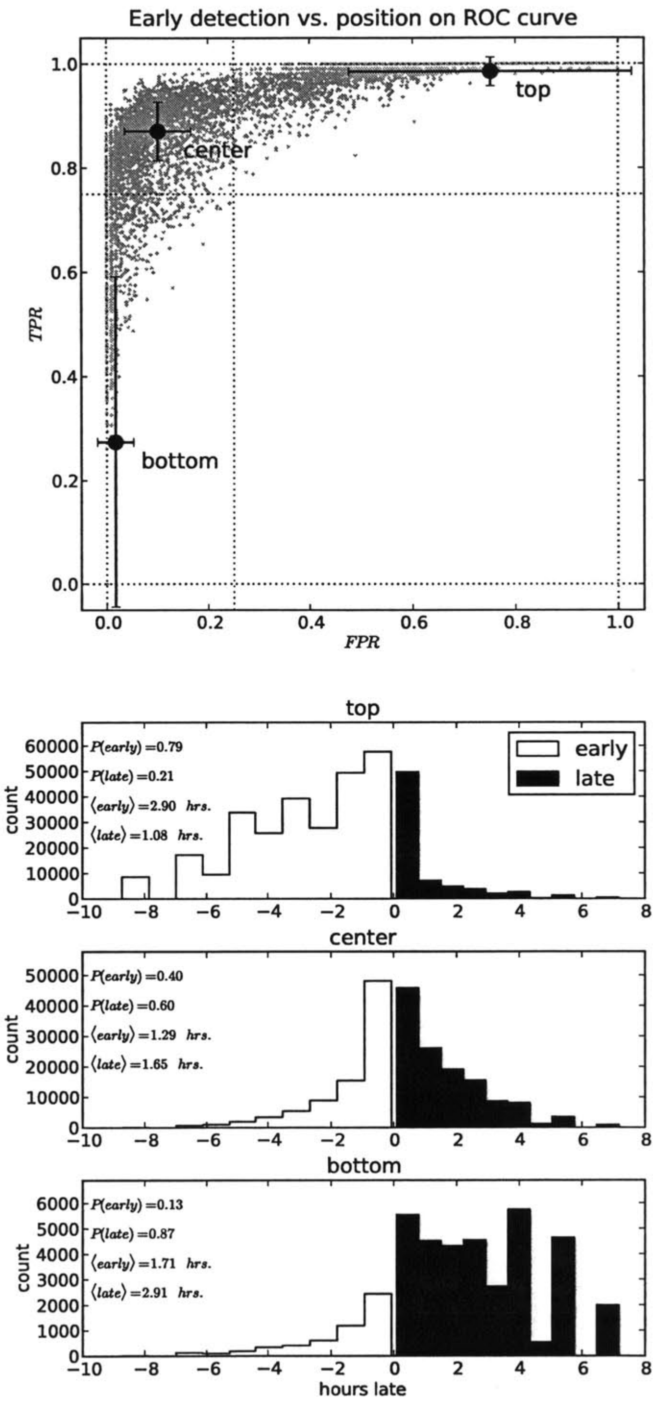
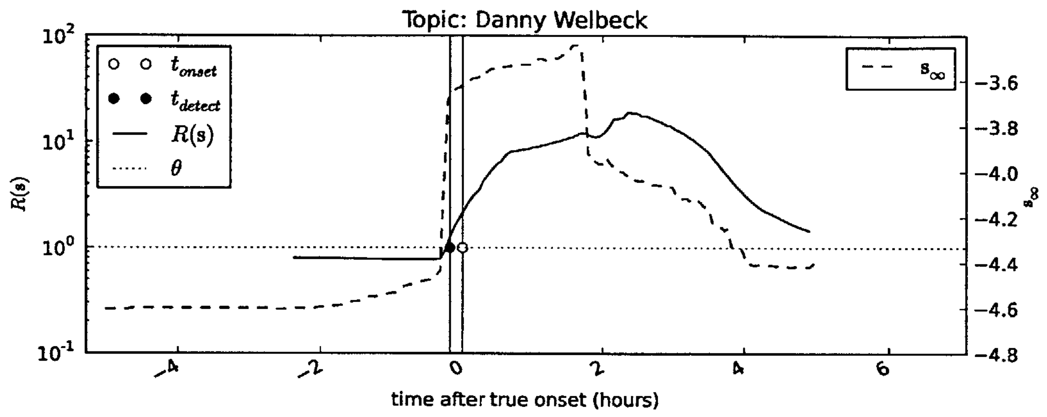


Figure 5-4: Effect of position along ROC curve on early and late detection.

5.1.4 Examples

In this section, we show examples of our detection algorithm in action on specific topics. Figure 5-5 shows the detection of fast-spreading and a slow-spreading topics. In general, it is harder to detect fast-spreading topics early.



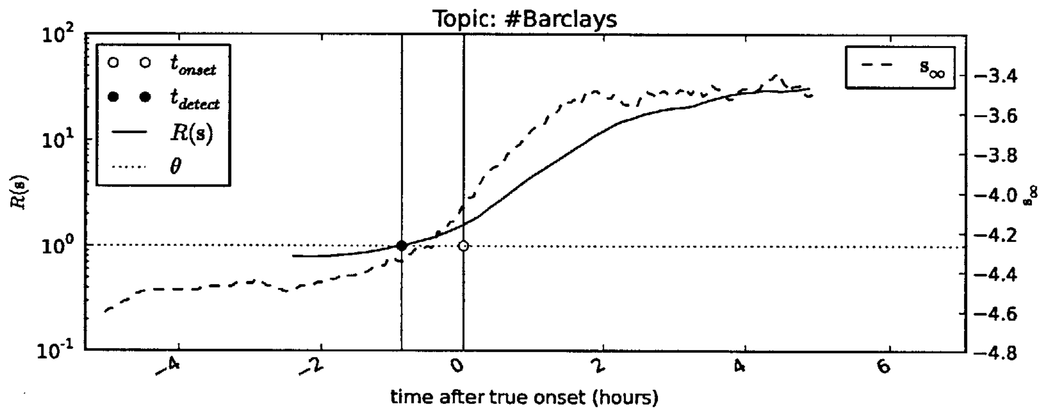
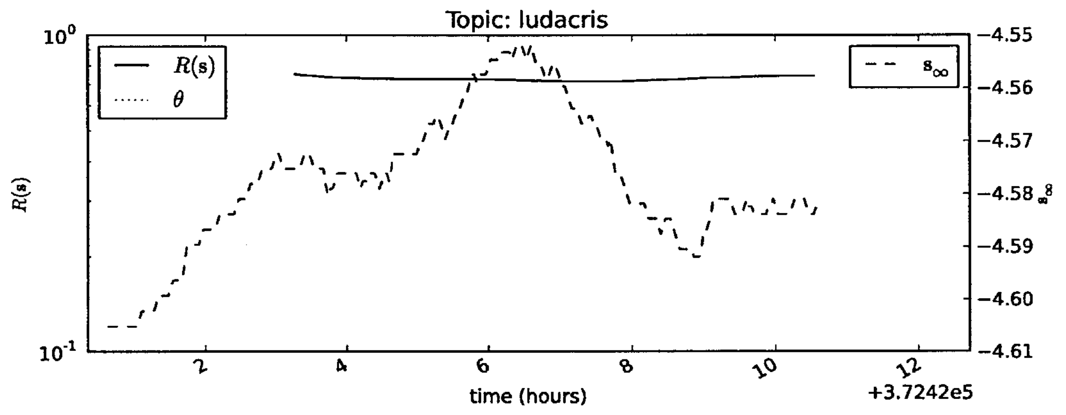


Figure 5-5: Fast-spreading vs. slow-spreading topics. **Top**: English football player Danny Welbeck scores late in the second half of the June 15th match between England and Sweden in the Euro 2012, securing a 3-2 victory for England. The reaction on Twitter is immediate. **Bottom:** Ed Miliband, leader of the UK's Labour Party, calls for a criminal investigation of Barclays, the global financial services provider, over involvement in the Libor fraud scandal. The story stimulates steadily growing discussion over the course of the day.

Figure 5-6 shows two true negative topics — topics that did not trend and were not detected as trending. In one case, the topic ("Ludacris") is a celebrity who, despite receiving consistently high attention on Twitter, is not involved in any rapidly breaking story, and hence never becomes a trending topic in the time period considered. The other topic, "tweetin," is presumed to be a common expression that is not associated with any trending topic.



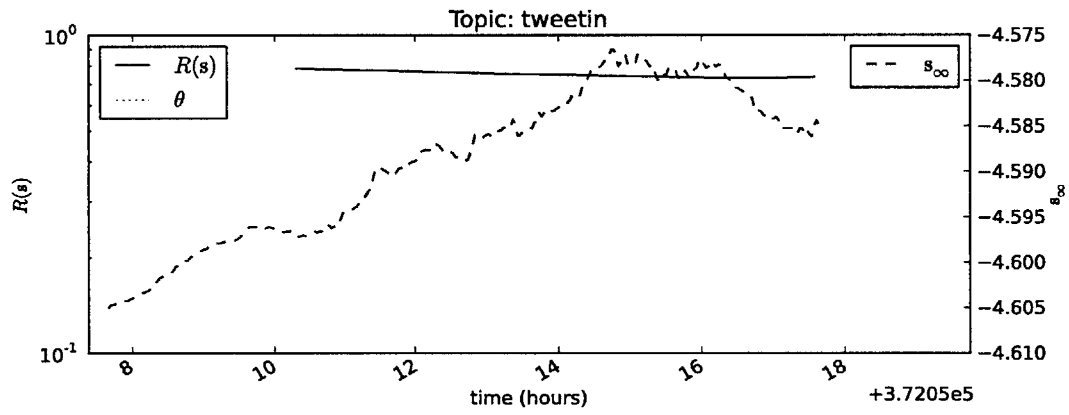
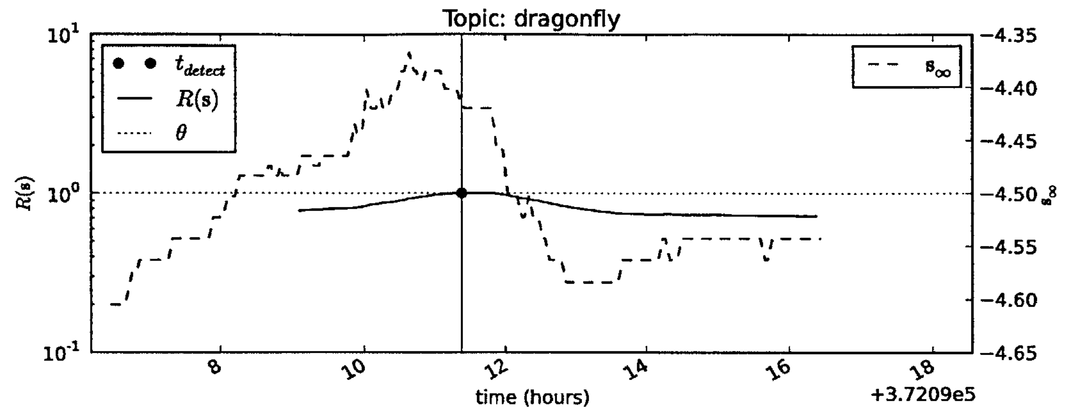


Figure 5-6: Examples of true negatives — topics that did not trend and were not detected as trending. **Top**: Although Ludacris, a well-known celebrity, receives constant attention on Twitter, there is no anomalous event involving Ludacris that would cause the topic to trend. **Bottom**: The word "tweetin" is being used as a part of regular speech to refer to the act of posting a message on Twitter, and does not constitute a trending topic.

In Figure 5-7, we show examples of false negatives — topics that were not trending, but were detected as such. It is interesting to note that some topics that did not trend, such as "redsn0w" in the bottom half of the figure, may be associated with emerging stories of smaller magnitude that almost became trending topics.



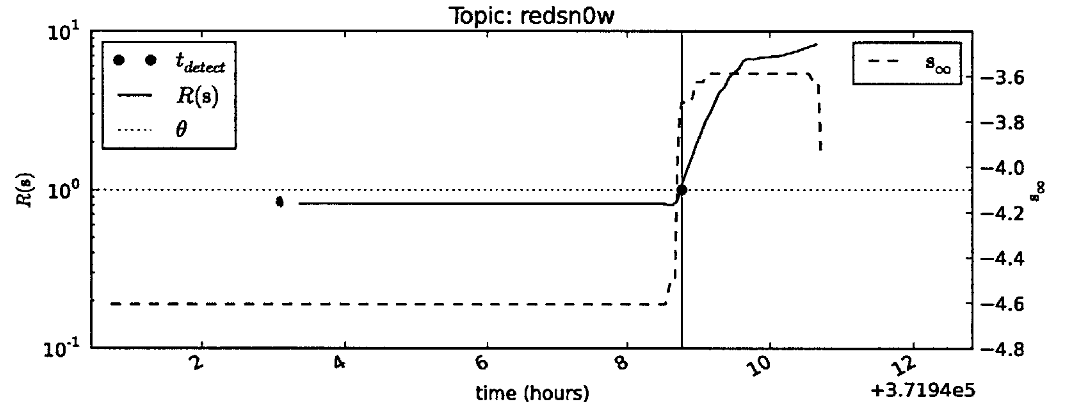


Figure 5-7: Examples of false positives — topics that did not trend but were detected as trending. **Top**: If the activity of a topic trends upward for a sufficiently long time, it may sufficiently resemble the activity of topics that trended and lead to a false detection. **Bottom**: Some false positives refer to actual breaking events that happened to not make the trending topics list on Twitter. The topic "redsn0w," for example, coincides with a new release of popular jailbreaking tool for iOS.

5.2 Effect of Parameters

5.2.1 Effect on Position Along ROC Curve

	$\langle N_{obs} \rangle$	$\langle h_{ref} \rangle$	$\langle \gamma \rangle$	$\langle D_{req} \rangle$	$\langle \theta \rangle$	$\langle N_{smooth} \rangle$
top	76.93	6.86	4.38	2.90	0.81	88.66
center	77.92	6.09	3.75	2.88	1.79	88.61
bottom	68.76	6.83	1.81	3.70	2.69	88.10

Table 5.1: The effect of parameters on position along the ROC curve.

In this section, we analyze the effect of each parameter on the position along the ROC curve. To simplify analysis, we again consider only the top, center and bottom regions of the curve. Table 5.1 shows the mean of the parameters responsible for the (FPR, TPR) points in each region. It is immediately clear that the mean threshold θ is dramatically different in each region. This coincides with our intuition that a low threshold leads to higher TPR and FPR (top region) and vice versa (bottom region). Similarly, a higher number of required consecutive detections D_{req} puts us in the bottom region of the curve, and a lower number puts us in the center and top regions. Another clear effect is that low values of γ put us in the bottom region and higher values put is in the center and top regions. N_{obs}, h_{ref}, and N_{smooth} do not appear to have a significant effect on the position along the curve.

5.2.2 Effect on Movement Along ROC Curve

For each ROC curve, we have a parameter that varies to produce the ROC curve, which we will call the variable parameter, and a fixed combination of the remaining parameters, which we will call the constant parameters.

As we vary the variable parameter, how do we move up or down the ROC curve?

We show how varying a given parameter p trades off FPR for TPR by computing the discrete derivative of FPR and TPR with respect to p. For each ROC curve, corresponding to the variable parameter p and some fixed combination of remaining parameters, we compute

$$\Delta_{p,i}^{FPR} = \frac{FPR(p_i) - FPR(p_{i-1})}{p_i - p_{i-1}} \tag{5.1}$$

$$\Delta_{p,i}^{TPR} = \frac{TPR(p_i) - TPR(p_{i-1})}{p_i - p_{i-1}} \tag{5.2}$$

for each ROC curve associated with p and for i ranging from the second to the last value of p in increasing order. If each point on the ROC curve is produced by multiple trials, we compute the above for all possible combinations of ROC curves. Finally, we compute the above across all combinations of fixed parameters.

The result is a distribution of discrete derivatives of FPR and TPR with respect to a variable parameter of interest p which highlights the effect of p on tradeoffs between FPR and TPR. We can refer this effect as moving "up" the ROC curve, or "down" the ROC curve. If most of the mass of Δ_p^{FPR} and Δ_p^{TPR} is at values greater than 0, then an increase in p causes a decrease in FPR at the expense of lower TPR, moving down the curve. If, on the other hand, most of the mass is at values less than zero, an increase in p causes an increase in TPR at the expensive of higher FPR, moving up the curve.

Sometimes, the curve moves neither toward $(0,0)$ ("down the curve") nor toward $(1,1)$ ("up the curve") but toward $(0,1)$ or $(1,0)$. The former represents an increase in TPR in addition to a decrease in FPR — a win-win situation. The latter represents the exact opposite of that — an increase in FPR and a decrease in TPR.

$\langle \Delta_p^{FPR} \rangle$

	N_{obs}	h_{ref}	γ	D_{req}	θ	N_{smooth}
top	-0.0023	0.0335	-0.0945	-0.0493	-0.9807	-0.0002
center	-0.0002	0.0413	0.0846	-0.0098	-0.0746	0.0001
bottom	0.0002	0.0061	0.0306	-0.0052	N/A	0.0001

$\langle \Delta_p^{TPR} \rangle$

	N_{obs}	h_{ref}	γ	D_{req}	θ	N_{smooth}
top	-0.0002	0.0019	-0.0126	-0.0228	-0.2298	-0.0001
center	0.0003	-0.0007	0.0227	-0.0358	-0.0238	0.0000
bottom	0.0016	-0.0168	0.3838	-0.0594	N/A	0.0004

Table 5.2: Movement along ROC curve caused by changes in each parameter, depending on which region in the *FPR*-*TPR* plane the ROC curve starts.

Note that we did not count Δ_p for consecutive points at $(0,0)$ or $(1,1)$ since the *TPR* and *FPR* are not free to move any further despite changes to the variable parameter.

In Table 5.2, we see the movement along the curve caused by changes in each parameter. The behavior is not uniform, however. The change in *FPR* and *TPR* depending on the current position in the *FPR*-*TPR* plane. To study the effect of initial position on the movement along the curve, we once again break the space up into top, center and bottom regions. This time, each ROC curve is assigned to a region based on where the ROC curve begins (starting with the lowest value of the variable parameter). The discrete derivatives resulting from that curve are then assigned to the appropriate region.

It is not surprising that θ, which has the most influence on the position along the curve, also has by far the most influence on the movement along the curve. A larger θ moves us down the curve no matter where we start, as expected. Similarly, a larger D_{req} always moves us down the curve, also as expected.

Also influential is γ. Interestingly, it moves us down the curve if we start in the top region, and up the curve otherwise.

An increase in N_{obs} causes us to move down the curve if we start in the top region and up the curve if we start in the bottom region. In the center, region, it causes a slight movement perpendicular to the curve — increasing *TPR* while decreasing

FPR.

The length of the reference signals in hours and half the time window for a single detection run, h_{ref} causes us to move down the curve if we start in the bottom and up the curve if we start in the top.

N_{smooth} has no significant effect for the range of smoothing widths explored. It is possible that a bigger difference may be seen outside of this range.

5.3 Recommended Parameter Settings

We propose parameter regimes appropriate for the following three situations: 1) the cost of a false positive outweighs the cost of a false negative, 2) the cost of a false negative outweighs the cost of a false positive, and 3) the costs of a false positive and a false negative are comparable. We make use of the results involving position and movement along a the ROC curve shown in the previous sections.

5.3.1 Cost(FP) $\ll$ Cost(TP)

We recommend the parameter settings in the third row of Table 5.1, corresponding to the bottom region, which give an average (FPR, TPR) equal to $(0.02, 0.27)$. For fine-tuning, we can increase h_{ref} or decrease γ or N_{obs} to move down the curve (or do the opposite to move up the curve.)

5.3.2 Cost(FP) $\gg$ Cost(TP)

We recommend the parameter settings in the first row of Table 5.1, corresponding to the top region, which give an average (FPR, TPR) equal to $(0.74, 0.98)$. For fine-tuning, we can decrease h_{ref} or increase γ, N_{obs} to move up the curve (or do the opposite to move down the curve.)

5.3.3 Cost(FP) $\sim$ Cost(TP)

We recommend the parameter settings in the second row of Table 5.1, corresponding to the center region, which give an average (FPR, TPR) equal to (0.10, 0.87). For fine-tuning, we can increase N_{obs} to simultaneously increase TPR and decrease FPR.

5.3.4 Out of the Box

Finally, if one wishes to use the algorithm "out of the box," we recommend, the parameter setting of $\gamma = 10$, $N_{obs} = 115$, $\theta = 1$, $N_{smooth} = 80$, $h_{ref} = 7$, $D_{req} = 1$ shown in Figure 5-1 at the beginning of this chapter, which achieves a TPR of 95%, a FPR of 4%, and is able to detect trending topics in advance of Twitter 97% of the time.

Chapter 6

Summary of Contributions and Future Work

6.1 Contributions

We have introduced a setting to do inference in the presence of large amounts of data governed by an underlying stochastic model. Within this setting, we have derived an online time series classification method and an associated implementation that is simple, efficient, and scalable. We have investigated the classification performance of our method by applying it to the task of predicting trending topics on Twitter. We have showed the method's flexibility by analyzing the effects of the algorithm parameters, and have quantified the tradeoffs between relative detection time, true positive rate, and false positive rate. Finally, we have demonstrated the method to be successful in detecting trending topics on Twitter before Twitter's algorithm does, while maintaining a low error rate.

6.2 Future Work

It remains to be established what the theoretical guarantees of our latent source method are. In particular, an important next step is to establish the statistical efficiency of our algorithm when the actual data is truly generated according to a latent source model. Another important step is to compare the classification performance of our algorithm relative to other supervised learning methods, e.g. ones based on Tikhonov Regularization. A third important step is to modify our algorithm to take advantage of the structure in large amounts of unlabeled data, as many large data sets are only sparsely labeled. A fourth and final important step is to evaluate the classification performance and computational efficiency of our algorithm on truly massive datasets. While we have designed our algorithm to be efficient and scalable, we have only used it on a relatively small data set and its full power remains to be seen.

Bibliography

[1] J Andreeva, S Belov, A Berejnoj, C Cirstoiu, Y Chen, T Chen, S Chiu, M D F D Miguel, A Ivanchenko, B Gaidioz, J Herrala, M Janulis, O Kodolova, G Maier, E J Maguire, C Munro, R P Rivera, R Rocha, P Saiz, I Sidorova, F Tsai, E Tikhonenko, and E Urbah. Dashboard for the lhc experiments. *Journal of Physics: Conference Series*, 119(6):062008, 2008.

[2] Sitaram Asur, Bernardo A. Huberman, Gábor Szabó, and Chunyan Wang. Trends in social media: Persistence and decay. In *ICWSM*, 2011.

[3] K.B. Athreya and PE Ney. *Branching processes.* Dover Publications, 2004.

[4] Frank Bauer, Sergei Pereverzev, and Lorenzo Rosasco. On regularization algorithms in learning theory. *J. Complex.*, 23(1):52–72, February 2007.

[5] Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. In *ICWSM*, 2011.

[6] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, MDMKDD '10, pages 4:1–4:10, New York, NY, USA, 2010. ACM.

[7] Scott Gaffney and Padhraic Smyth. Trajectory clustering with mixtures of regression models. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 63–72, New York, NY, USA, 1999. ACM.

[8] Daniel Gruhl, Ramanathan V. Guha, David Liben-Nowell, and Andrew Tomkins. Information diffusion through blogspace. In *WWW*, pages 491–501, 2004.

[9] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12, 2009.

[10] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 207–216, New York, NY, USA, 2006. ACM.

[11] Jon Kleinberg. Bursty and hierarchical structure in streams. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 91–101, New York, NY, USA, 2002. ACM.

[12] Scott Lenser and Manuela Veloso. Non-parametric time series classification, 2005.

[13] Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, SIGMOD '10, pages 1155–1158, New York, NY, USA, 2010. ACM.

[14] James McNames. A nearest trajectory strategy for time series prediction, 1998.

[15] Hazarath Munaga, M. D. R. Mounica Sree, and J. V. R. Murthy. Article: Dentrac: A density based trajectory clustering tool. *International Journal of Computer Applications*, 41(10):17–21, March 2012. Published by Foundation of Computer Science, New York, USA.

[16] Alex Pentland. Honest signals: how social networks shape human behavior. In *ACM Multimedia*, pages 583–584, 2011.

[17] Alex Pentland, Pamela Hinds, and Taemie Kim. Awareness as an antidote to distance: making distributed groups cooperative and consistent. In *CSCW*, pages 1237–1246, 2012.

[18] Tomaso Poggio and Steve Smale. The mathematics of learning: Dealing with data. *Notices of the American Mathematical Society*, 50:2003, 2003.

[19] Thanawin Rakthanmanon, Bilson J. L. Campana, Abdullah Mueen, Gustavo E. A. P. A. Batista, M. Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn J. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *KDD*, pages 262–270, 2012.

[20] Ernest S. Shtatl and Timur Shtatland. Another look at low-order autoregressive models in early detection of epidemic outbreaks and explosive behaviors in economic and financial time series.

[21] Michael M. Wagner, J. Michael Robinson, Fu-Chiang Tsui, Jeremy U. Espino, and William R. Hogan. Application of information technology: Design of a national retail data monitor for public health surveillance. *JAMIA*, 10(5):409–418, 2003.

[22] Danqing Xu, Yiqun Liu, Min Zhang, Shaoping Ma, Anqi Cui, and Liyun Ru. Predicting epidemic tendency through search behavior analysis. In *IJCAI*, pages 2361–2366, 2011.