A Latent Source Model for Online Time Series Classfication

Zhang Zhang
zsquared@math.umd.edu
University of Maryland, College Park

Advisor
Prof. Aravind Srinivasan
srin@cs.umd.edu
Department of Computer Science
University of Maryland, College Park

October 14, 2013

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. In the project: 1. We will investigate the theoretical performance guarantees of the latent source model; 2. We will implement the model by python; 3. We will investigate the strategy to estimate the values of different parameters; 4. We will test our implementation and use the model to predict which news topics on Twitter will go viral to become trends and analyze the results.

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. 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. Particle collision experiments at the Large Hadron Collider generate more than 15 petabytes of data 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. 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 project, 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" "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 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 the 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.

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 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, 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).

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 $t_1, \ldots, t_n$ for + and $q_1, \ldots, q_m$ for . Each observation labeled + is assumed to be a noisy version of one of the latent sources $t_1, \ldots, t_n$. Similarly, each observation labeled is assumed to be a noisy version of one of the latent sources $q_1, \ldots, q_m$. 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

In this project, we will focus on time-varying *signals*. 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 $\mathbf{s}$ from an infinite stream s_∞ is *generated* by a latent source $\mathbf{q}$ if $\mathbf{s}$ is a noisy version of $\mathbf{q}$. Accordingly, we propose the following stochastic model relating a latent source $\mathbf{q}$ and an obervation $\mathbf{s}$:

$$\mathbb{P}(\mathbf{s} \text{ generated by } \mathbf{q}) \propto \exp(-\gamma d(\mathbf{s}, \mathbf{q})) \tag{1}$$

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

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

for digital signals $\mathbf{s}$ 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 $\mathbf{s}$ is an observed signal of length N_{obs}. We would like to compute the probability that $\mathbf{s}$ belongs to each class. We can then use those probabilities to compute an estimate of the class of $\mathbf{s}$. To compute the probability that $\mathbf{s}$ 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

$$\begin{aligned}
\mathbb{P}(+|\mathbf{s}) &\propto \sum_{\mathbf{r}\in\mathcal{R}_-} \mathbb{P}(\mathbf{s} \text{ belongs to } +,\, \mathbf{s} \text{ shares a latent source with } \mathbf{r}) \\
&= \sum_{\mathbf{r}\in\mathcal{R}_-} \sum_{j=1}^{n} \mathbb{P}(\mathbf{s} \text{ generated by } t_j,\, \mathbf{r} \text{ generated by } t_j) \\
&= \sum_{\mathbf{r}\in\mathcal{R}_-} \sum_{j=1}^{n} \exp(-\gamma d(\mathbf{s}, t_j))\exp(-\gamma d(\mathbf{r}, t_j)) \\
&= \sum_{\mathbf{r}\in\mathcal{R}_-} \sum_{j=1}^{n} \exp(-\gamma d(\mathbf{s}, t_j) + d(\mathbf{r}, t_j))
\end{aligned} \tag{3}$$

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

$$\mathbb{P}(+|\mathbf{s}) \propto \sum_{\mathbf{r}\in\mathcal{R}_+} \exp(-\gamma \min_j (d(\mathbf{s}, t_j) + d(\mathbf{r}, t_j))) \tag{4}$$

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 1 illustrates the reference signal $\mathbf{r}$, the observation $\mathbf{s}$, the latent source signals $t_1, \ldots, t_n$ and the minimizing latent source signal $\mathbf{t}_{j^*}$

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

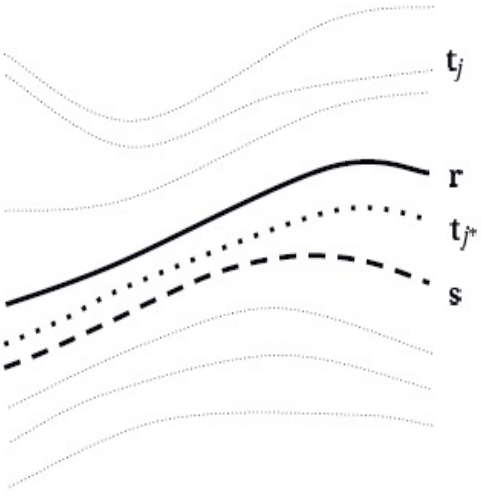

Figure 1: An illustration of the latent source signal $\mathbf{t}_{j^*}$ that minimize $d(\mathbf{s}, t_j)+d(\mathbf{r}, t_j)$ in Eq.4. Depicted are the reference signal $\mathbf{r}$, the observation $\mathbf{s}$, the latent source signals $t_1, \ldots, t_n$ and the minimizing latent source signal $\mathbf{t}_{j^*}$

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{t}}(d(\mathbf{s},\mathbf{t}) + d(\mathbf{r},\mathbf{t}))|_{t=\frac{\mathbf{s}+\mathbf{r}}{2}} &= \frac{\partial}{\partial \mathbf{t}}(c(\mathbf{t}-\mathbf{s}) + c(\mathbf{t}-\mathbf{r}))|_{t=\frac{\mathbf{s}+\mathbf{r}}{2}} \\
&= c'(\frac{\mathbf{s}+\mathbf{r}}{2} - \mathbf{s}) + c'(\frac{\mathbf{s}+\mathbf{r}}{2} - \mathbf{r}) \\
&= c'(\frac{\mathbf{r}-\mathbf{s}}{2}) + c'(\frac{\mathbf{s}-\mathbf{r}}{2}) \\
&= 0
\end{aligned} \tag{5}$$

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

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

$$\begin{aligned}
d(\mathbf{s},\mathbf{t}^*) + d(\mathbf{r},\mathbf{t}^*) &= d(\mathbf{s}, \frac{\mathbf{r}+\mathbf{s}}{2}) + d(\mathbf{r}, \frac{\mathbf{r}+\mathbf{s}}{2}) \\
&= c(\frac{\mathbf{r}-\mathbf{s}}{2}) + c(\frac{\mathbf{s}-\mathbf{r}}{2})
\end{aligned} \tag{6}$$

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

$$c(ax) = g(a)c(x) \tag{7}$$

where g is some positive definite function. Applying this to Eq.6 gives

$$
\begin{aligned}
d(\mathbf{s},\mathbf{t}^*) + d(\mathbf{r},\mathbf{t}^*) &= g(\frac{1}{2})c(\mathbf{r}-\mathbf{s}) + g(\frac{1}{2})c(\mathbf{s}-\mathbf{r}) \\
&= 2 \cdot g(\frac{1}{2})c(\mathbf{r}-\mathbf{s}) \\
&= 2 \cdot g(\frac{1}{2})d(\mathbf{r},\mathbf{s}) \\
&= C \cdot d(\mathbf{r},\mathbf{s})
\end{aligned} \tag{8}
$$

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

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

$$
\begin{aligned}
\mathbb{P}(+|\mathbf{s}) &\propto \sum_{\mathbf{r}\in\mathcal{R}_+} \exp(-\gamma \min_j(d(\mathbf{s},t_j) + d(\mathbf{r},t_j))) \\
&\approx \sum_{\mathbf{r}\in\mathcal{R}_+} \exp(-\gamma d(\mathbf{s},\mathbf{r}))
\end{aligned} \tag{9}
$$

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

$$
\mathbb{P}(-|\mathbf{s}) \propto \sum_{\mathbf{r}\in\mathcal{R}_-} \exp(-\gamma d(\mathbf{s},\mathbf{r})) \tag{10}
$$

2.3.2 Class Estimator

Our class estimation rule is simple: assign to the observation the class with the highest probability:

$$
R(\mathbf{s}) = \frac{\mathbb{P}(+|\mathbf{s})}{\mathbb{P}(-|\mathbf{s})} = \frac{\sum_{\mathbf{r}\in\mathcal{R}_+} \exp(-\gamma d(\mathbf{s},\mathbf{r}))}{\sum_{\mathbf{r}\in\mathcal{R}_-} \exp(-\gamma d(\mathbf{s},\mathbf{r}))} \tag{11}
$$

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

$$
R(\mathbf{s}) = \frac{\sum_{\mathbf{r}\in\mathcal{R}_+} \exp(-\gamma \sum_{i=1}^{N_{obs}} (s_i - r_i)^2)}{\sum_{\mathbf{r}\in\mathcal{R}_-} \exp(-\gamma \sum_{i=1}^{N_{obs}} (s_i - r_i)^2)} \tag{12}
$$

The estimator for the class label L is therefore

$$
\hat{L}(\mathbf{s}) = \begin{cases} + & \text{if} \quad R(\mathbf{s}) > \theta \\ - & \text{if} \quad R(\mathbf{s}) \leq \theta \end{cases} \tag{13}
$$

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 will explore it in Section 4.

2.3.3 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 explore the effect of D_{req} in Section 4.

2.3.4 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 illustrates this.

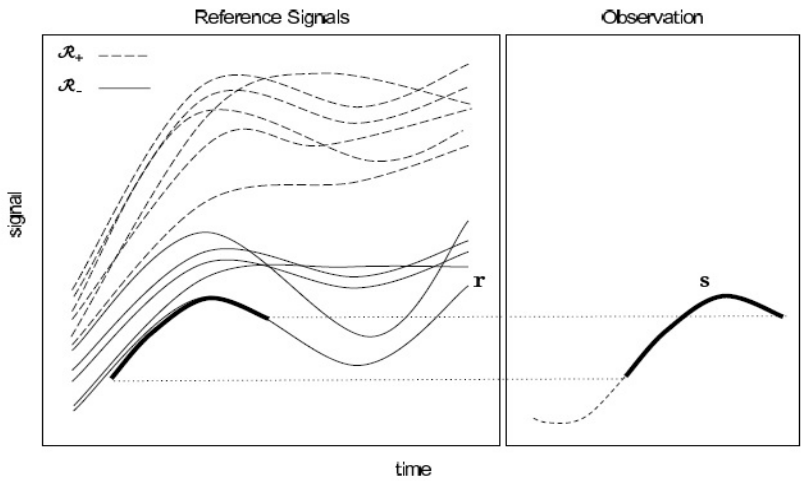

Figure 2: To compare a long reference signal to a short observation, we compute the distance between the observation and the closest *piece* of a reference signal

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}.

$$d(\mathbf{r}, \mathbf{s}) = \min_{k=1,\ldots,N_{ref}-Nobs+1} d(\mathbf{r}_{k:k+N_{obs}-1}, \mathbf{s}) \tag{14}$$

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 $R(\mathbf{s})$ from the previous section becomes

$$R(\mathbf{s}) = \frac{\sum_{\mathbf{r}\in\mathcal{R}_+} \exp(-\gamma \min_{k=1,\ldots,N_{ref}-Nobs+1} (\sum_{i=1}^{N_{obs}} (s_i - r_{i+k-1})^2)}{\sum_{\mathbf{r}\in\mathcal{R}_-} \exp(-\gamma \min_{k=1,\ldots,N_{ref}-Nobs+1} (\sum_{i=1}^{N_{obs}} (s_i - r_{i+k-1})^2)} \tag{15}$$

3 Implementation on Twitter Topics Detection

In this section, we consider the application of the method in section 2 toward detection of trending topics on Twitter. We discuss the Twitter service, the collection and preprocessing of data, and the experimental setup for the detection task.

3.1 Overview

3.1.1 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.

3.1.2 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 3.1. *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 3.2. *A Tweet is* ***about*** *a topic if it contains the topic as a substring. Tweets can be about many topics.*

Example 3.1. *The following tweet by the author (handle @Zsquared) contains the string "AMSC663" Hence, it is considered to be about the topic "AMSC663".*

"AMSC663 Project drives me crazy."

Definition 3.3. *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 3.4. *A trending topic will also occasionally be referred to as a **trend** for short.*

Definition 3.5. *The **trend onset** is the time that a topic first trended during a period of time.*

Example 3.2. *If the topic "government shutdown" is currently in the trending topics list on Twitter, we say that "government shutdown" is trending, and that is a trend. The topic "government shutdown" 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 "shutdown" happened. After "government" is no longer trending, we say that "government shutdown" trended.*

3.1.3 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).

3.1.4 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 and apply the classification method we discussed in previous sections.

3.2 Data

3.2.1 Data Collection

The online time series classification method detailed 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 October 1, 2013 and October 31,2013 (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.

Topics

We collected a list of all trending topics on Twitter from Oct. 1, 2013 to Oct 31, 2013 (the *sample window*) as well as the times that they were trending and their rank in the trending topics list on Twitter. 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.

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 "Kobe Bryant" trended during the sample window, then "Kobe" 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 Oct 1, 2013 to Oct 31, 2013 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.

3.2.2 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 n^{th} time bin. Let the cumulative volume of of Tweets up to time n be

$$v[n] = \sum_{m \leq n} \rho[m] \tag{16}$$

Thus, effectively $\rho[n]$ is the discrete derivative of the continuous cumulative volume $v(t)$ over time i.e. $\rho(t) = v'(t)$. Therefore, we shall call $\rho[n]$ the rate of the signal at time step n.

Baseline Normalization

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. For example, a non-trending topic such as "city" is likely to have a consistent baseline of activity because it is a common 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

$$b = \sum_{n} \rho[n] \tag{17}$$

and a baseline normalized signal ρ_b as

$$\rho_b[n] = (\frac{\rho[n]}{b})^{\beta} \tag{18}$$

The exponent β controls how much we reward and penalize rates above and below the baseline rate. We use $\beta = 1$ in this project.

Spike Normalization

Another difference between the rates of Tweets for trending topics and that of nontrending 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.

$$\rho_{b,s}[n] = |\rho_b[n] - \rho_b[n-1]|^{\alpha} \tag{19}$$

in terms of the already baseline-normalized rate ρ_b. The parameter $\alpha > 1$ controls how much spikes are rewarded. We use $\alpha = 1.2$. We will check the graph of signals to see if the value of α is good enough.

Smooth

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

$$\rho_{b,s,c}[n] = \sum_{m=n-N_{smooth}+1}^{n} \rho_{b,s}[m] \tag{20}$$

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. It is reasonable to measure the volume of tweets at a logarithmic scale to reveal these details. Therefore, as a final step, we take the logarithm of the signal constructed so far to produce the signal

$$\rho_{b,s,c,l}[n] = \log \rho_{b,s,c}[n] \tag{21}$$

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 Section 2, we try 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 a 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 l eading 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 topic 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 of N_{ref}.

3.3 Experiment

We propose an experiment to measure our algorithms 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.

3.3.1 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. 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 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, we define this in terms of hours.

Definition 3.6. *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.

3.3.2 Parameter Exploration and Trials

We explore all combinations of the following ranges of parameters, excluding param- eter settings that are incompatible (e.g. $N_{obs} > N_{ref}$). For each combination, we conducted 10 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_{ref}: 1, 3, 5
- θ: 0.65, 1, 3

3.3.3 Validation of Implementation

Currently, we proved that the probability of misclassification of the signal is bounded by a function of parameters. We could run my codes and see if my result is bounded. We will discuss the proof in the winter report.

$$\begin{aligned} &\mathbb{P}(\text{misclassification of } \mathbf{s} \text{ using its first T samples}) \\ &\leq (\theta + \frac{1}{\theta})(2\Delta_{max} + 1)n \exp(-(\gamma - 4\sigma^2\gamma^2)G^{(T)}(R_+, R_-)) + m^{-\beta+1} \end{aligned} \tag{22}$$

3.3.4 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. We analyze the effect of the algorithm parameters on the tradeoff between false positive rate, true positive rate, and relative detection time. 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.

We should 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.

4 Project Schedule and Milestones

Oct 1,2013 - Oct 31,2013
Learn Programming language: Python

Nov 1,2013 - Nov 30,2013
Write codes to classify data as different topics

Dec 1,2013 - Dec 30,2013
Write codes of the model

Jan 1,2014 - Jan 31,2014
Write codes to simulate ROC curves

Feb 1,2014 - Feb 28,2014
Testing Implementation

Mar 1,2014 - Mar 31,2014
Write Documentations

5 Deliverables

The deliverables for this project are the Python codes that implement the Latent Source Model to dection twitter trending topics and any code used for testing. The code will be optimized for performance and effective memory management, as well as being fully documented. Report at various stages throughout the course will detail the approach, implementation, validation, testing, and extensions of the algorithm. With this information, a researcher will be able to reproduce any results present in our reports.

References

[1] George Chen, Stanislav Nikolov, Devavrat Shah *A Latent Source Model for Online Time Series Classification*, CIPS Conference, 2013

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

[3] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella *Emerging topic detection on twitter based on temporal and social terms evaluation*, In Proceedings of the Fifth International Conference on Weblogs and Social Media, 2011

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

[5] Alon Halevy, Peter Norvig, and Fernando Pereira *The unreasonable effectiveness of data*, IEEE Intelligent Systems, 2011

[6] Yen-Hsien Lee, Chih-Ping Wei, Tsang-Hsiang Cheng, and Ching-Ting Yang *Nearest neighbor based approach to time series classification*,Decision Support Systems, 2012

[7] Juan Rodriguez and Carlos Alonso *Interval and dynamic time warping based decision trees*, In proceedings of the 2004 ACM Symposium on Applied Computing