Uncover Trending Topics on Data Stream by Linear Prediction Modeling

Carlos Enrique Gutierrez[†], Mohammad Reza Alsharif[†] and Katsumi Yamashita[††]

[†]Department of Information Engineering, University of the Ryukyus, Okinawa, Japan.
[††]Graduate School of Engineering, Osaka Prefecture University, Osaka, Japan.

Summary
The use of streaming data to analyze and discern patterns to make better decisions is becoming the basis for creating significant value for companies. Torrents of data flooding continuously force organizations to understand what information truly count, and analyze what they can do with that information. The aim of this paper is to explore the adaptation of a linear prediction model to discover trend topics on a news stream, by uncovering the most influential variables (words). Each input consists in a news classified within two dummy categories; and transformed into a numerical vector containing around 10,000 different features. We apply continuously a linear model to perform shrinkage on input vectors; as a result, variables with strongest characteristics show up, while those with negligible characteristics are removed. Due the dynamic and uninterrupted characteristics of the input, the output exposes the evolution of the most significant variables over the time. Firstly we provide details of linear prediction/regression model, secondly we introduce our proposed algorithm and finally simulation results and conclusions are shown.

Keywords:
Lasso, big data, data mining, features detection.

1. Introduction

In the past few years the tracking of breaking topics has become an active research area in computer science. Twitter introduced "Twitter Trends" in 2008 [1], defined them as the topics that are being talked about more right now than they were previously. The trends show the "most" breaking news of a set of breaking news generated from millions of tweets around the world. Algorithms to process such data stream must operate continuously, processing each new input in real time; using computational resources' memory and storage space up to a reasonable threshold. That is one of the main characteristic that differentiates data stream algorithms from batch algorithms.

Many algorithms are based on stream elements' frequency; showing as output the most frequent elements. A big issue related to counting is that it is needed a ranking of frequent elements extracted from a maintained full list of elements and their counters. Sampling techniques also are not enough efficient, they represent an approximate solution of what is a trending topic; the issue is related to whether samples include or not real trending tropics. Twitter's algorithm for determining trends is a private algorithm, but some information provided by the company indicates that topics break into trends when the volume associated to that topic at a given moment dramatically increases. In this case information's velocity increases quickly enough, compared to a baseline level and historical appearances/velocities associated.

In this paper we describe an algorithm to find the most important topics on a news stream, discovering trends and showing their evolution over the time. Our algorithm is based on a linear prediction model with shrinkage operators that retain the most important variables, to our knowledge this is the first time that linear prediction models are used for uncovering trending topics. With a large number of fixed features, around 10,000 in our experiments, we obtain automatically smaller subsets of the strongest variables over the time. In this paper we will refer to variables/features as potential trends. Important to clarify that our main goal is not to create a prediction model, but to uncover the strongest topics of a news stream.

The present work pretends to be a powerful alternative for processing data stream, where answers can be derived from a sequence of main features detected over the time, that approximate the output for the stream as a whole.

2. Linear Prediction Models

Linear predictors are still being used in a diverse area of applications, such as data forecasting, speech recognition, model-based spectral analysis, signal restoration, and others. In machine learning field, a linear predictor is a linear function of a set of coefficients and independent variables, whose value is used to predict the outcome of a dependent variable. Functions of this type are common in linear regression, where the coefficients are known as regression coefficients. They also appear in various types of linear classifiers, such as perceptrons, support vector machines, and linear discriminant analysis.

Manuscript received May 5, 2014
Manuscript revised May 20, 2014

A linear predictor model assumes its function as linear in the inputs $X_1, X_2,..., X_n$. We wish to predict an output $Y = f(X)$, so the model is defined as follows:

$$f(X) = \theta_0 + \sum_{j=1}^{d} X_j \theta_j \quad (1)$$

Where $\theta_0, \theta_1,..., \theta_d$ are unknown coefficients, and d is the dimension of input vectors X. Typically there is a set of training data $(X_1, y_1)...(X_n, y_n)$ from where parameters θ are estimated. The most popular estimation method is the least squares, in which the coefficients θ are defined in order to minimize the quadratic cost between the output training data and the model predictions.

$$J(\theta) = \sum_{i=1}^{n} (y_i - f(x_i))^2$$
$$J(\theta) = \sum_{i=1}^{n} (y_i - \theta_0 - \sum_{j=1}^{d} x_{ij}\theta_j)^2 \quad (2)$$

Each $x_i = (x_{i1}, x_{i2},..., x_{id})^T$ is a vector of feature measurements for the ith case. Let's denote by X the n x $(d+1)$ matrix, where each row is an input vector, the first column is filled with ones, and similarly let's denote as y to the $n-$vector of outputs in the training set. The quadratic cost can be written in a matrix notation, such as:

$$J(\theta) = (y - X\theta)^T (y - X\theta) \quad (3)$$

Optimization of (3) by differentiating with respect to θ results:

$$\frac{\partial J(\theta)}{\partial \theta} = -2X^T y + 2X^T X\theta$$
$$0 = -2X^T y + 2X^T X\theta$$
$$\hat{\theta} = (X^T X)^{-1} X^T y \quad (4)$$

Model's predictions are calculated as follows:

$$\hat{y} = X\hat{\theta} = X(X^T X)^{-1} X^T y \quad (5)$$

One of the most famous results in statistics asserts that the least squares estimates of the parameters have the smallest variance among all linear unbiased estimates [2].

In case of streaming text data, we have high-dimensional inputs coming continuously, if we consider each word as a single variable. Therefore, it is needed to estimate a high-dimensional vector of coefficients θ which is almost impractical for a streaming data processing system; we propose an automatic selection of the most influential variables using shrinkage methods.

3. Shrinkage methods

Ridge regression penalizes the size of coefficients θ, its solution is given by:

$$\hat{\theta}^{ridge} = \arg\min_{\theta} \left\{ \sum_{i=1}^{n} (y_i - \theta_0 - \sum_{j=1}^{d} x_{ij}\theta_j)^2 + \delta^2 \sum_{j=1}^{d} \theta_j{}^2 \right\} \quad (6)$$

Where $\delta^2 >= 0$ is a complexity parameter that controls the amount of shrinkage, for larger values of δ the amount of shrinkage increases. Ridge method compresses $\theta's$ toward zero and each other. Writing equation (6) in matrix form, we have:

$$J(\theta) = (y - X\theta)^T (y - X\theta) + \delta^2 \theta^T \theta \quad (7)$$

And the solution of the above regularized quadratic cost function is:

$$\hat{\theta} = (X^T X + \delta^2 I_d)^{-1} X^T y \quad (8)$$

Where I_d is the d x d identity matrix. The ridge coefficients minimize a penalized quadratic cost between the output training data and the model predictions. Equation (8) shows how ridge adds the penalty down the diagonal, introducing bias but reducing the variance of the estimate. This penalty was incorporated due issues with results for equation (4), if matrix X is not full rank $X^T X$ is not invertible and there is no unique solution; this problem does not occur with ridge regression, this was the main motivation for ridge method when it was introduced.

Another biased regression technique is Least Absolute Selection and Shrinkage Operator, also known as "lasso", it is like ridge method but with big differences in the penalty term. The estimation by lasso is as follows:

$$\hat{\theta}^{lasso} = \arg\min_{\theta} \left\{ \sum_{i=1}^{n} (y_i - \theta_0 - \sum_{j=1}^{d} x_{ij}\theta_j)^2 + \delta^2 \sum_{j=1}^{d} \left|\theta_j\right| \right\} \quad (9)$$

In this case, the ridge penalty in equation (6) is replaced by the lasso penalty $\delta^2 \sum_{j=1}^{d} \left|\theta_j\right|$ also known as

L_1 norm. Due to the new penalty form, the solutions for estimation of $\theta's$ are nonlinear in y. Lasso in matrix form is as follows:

$$J(\theta) = (y - X\theta)^T (y - X\theta) + \delta^2 \sum_{j=1}^{d} \left|\theta_j\right| \quad (10)$$

A ridge solution can be hard to interpret because it is not sparse, which means no $\theta's$ are set exactly to 0. Lasso penalty introduces a very important change; some of the coefficients may be shrunk exactly to zero [3]. If complexity parameter δ is sufficiently large, some $\theta's$ are driven to zero, leading to a sparse model.

This is the core of the proposed algorithm in this paper; lasso applied consecutively on a news stream, takes care of the variables selection for us removing irrelevant features. Computing lasso solution is a quadratic programming problem; it is needed to optimize equation (10) where several variables are subjected to linear constraints. For example, the derivative containing an absolute value. Next section describes our proposed solution for this problem.

4. Computing Lasso optimization

The Lasso has been studied and used in many applications over the last years. Least angle regression algorithm [4] and the solution path of the generalized lasso [5] are two relevant algorithms to solve lasso quadratic programming problems. Our proposed algorithm is a close form of lasso solution, designed with the purpose of provides scalability from batch data to streaming data.

Expression (10) can be expressed as:

$$J(\theta) = \sum_{i=1}^{n} (y_i - \theta_0 - \sum_{j=1}^{d} x_{ij}\theta_j)^2 + \delta^2 \sum_{j=1}^{d} \left|\theta_j\right| \quad (11)$$

Replacing $\sum_{j=1}^{d} x_{ij}\theta_j$ by its equivalent in matrix notation ${X_i}^T\theta$ we obtain:

$$J(\theta) = \sum_{i=1}^{n} (y_i - \theta_0 - {X_i}^T\theta)^2 + \delta^2 \sum_{j=1}^{d} \left|\theta_j\right| \quad (12)$$

Differentiating the 1st term of the above sum with respect to one generic feature coefficient θ_j we have:

$$\frac{\partial J(\theta)}{\partial \theta_j}^{1^{st}term} =$$

$$\sum_{i=1}^{n} 2(y_i - \theta_0 - {X_{i_{-j}}}^T\theta_{-j} - X_{i_j}\theta_j)(-X_{i_j})$$

Where ${X_{i_{-j}}}^T\theta_{-j}$ is same as ${X_i}^T\theta$ but excluding the feature j and its coefficient, and $X_{i_j}\theta_j$ are the variable and coefficient for feature j.

$$\frac{\partial J(\theta)}{\partial \theta_j}^{1^{st}term} = \quad (13)$$

$$= \sum_{i=1}^{n} 2{X_{i_j}}^2\theta_j - \sum_{i=1}^{n} 2(y_i - \theta_0 - {X_{i_{-j}}}^T\theta_{-j})(-X_{i_j})$$

Denoting $\sum_{i=1}^{n} 2{X_{i_j}}^2 = z_1$ and

$\sum_{i=1}^{n} 2(y_i - \theta_0 - {X_{i_{-j}}}^T\theta_{-j})(-X_{i_j}) = z_2$, expression (13) is expressed as:

$$\frac{\partial J(\theta)}{\partial \theta_j}^{1^{st}term} = z_1\theta_j - z_2 \quad (14)$$

The 2nd term $\delta^2 \sum_{j=1}^{d} \left|\theta_j\right|$ is differentiated in the following way:

$$\frac{\partial J(\theta)}{\partial \theta_j}^{2^{st}term} = \delta^2 \frac{\partial\left|\theta_j\right|}{\partial \theta_j} = \begin{cases} -\delta^2 & if \quad \theta_j < 0 \\ (-\delta^2, \delta^2) & if \quad \theta_j = 0 \\ \delta^2 & if \quad \theta_j > 0 \end{cases} \quad (15)$$

Written together equations (14) and (15) and equating to zero, we have:

$0 = z_1\theta_j - z_2 + \delta^2 \frac{\partial\left|\theta_j\right|}{\partial \theta_j}$, therefore estimation for the feature coefficient $\hat{\theta}_j$ will be:

$$\hat{\theta}_j = \frac{z_2 - \delta^2 \frac{\partial\left|\theta_j\right|}{\partial \theta_j}}{z_1} = \begin{cases} \frac{z_2 + \delta^2}{z_1} & if \quad \theta_j < 0 \\ (\frac{z_2 + \delta^2}{z_1}, \frac{z_2 - \delta^2}{z_1}) & if \quad \theta_j = 0 \\ \frac{z_2 - \delta^2}{z_1} & if \quad \theta_j > 0 \end{cases} \quad (15)$$

Based on equation (13) it is known that z_1 is always positive, therefore $\theta_j < 0$ is same as $z_2 < -\delta^2$, and equivalently $\theta_j > 0$ same as $z_2 > \delta^2$

Our algorithm that implements expression (15) iteratively until convergence is described below:

1 - Initialize coefficients by ridge method (equation 8)
$\hat{\theta} = (X^T X + \delta^2 I_d)^{-1} X^T y$
2 - Calculate z_1 *(equation 13)*
3 - Initialize a value for δ *(amount of shrinkage)*
4 - Repeat until coefficients $\hat{\theta}$*'s get stable:*
5 - For each feature j *perform:*
6-Calculate z_2 *(equation 13)*
7-In case $z_2 < -\delta^2$ *perform:*
8 -Update $\hat{\theta}_j = \frac{z_2 + \delta^2}{z_1}$
9 - In case $z_2 > \delta^2$ *perform:*
10 - Update $\hat{\theta}_j = \frac{z_2 - \delta^2}{z_1}$
11 -Otherwise, update $\hat{\theta}_j = 0$
12 - Change value of δ *and iterate from line 4*
13 - End

Above algorithm allows for identification of the relevant variables for a set of inputs; some modifications are implemented in next section to adapt its operation to streaming data.

5. Lasso for streaming data

The problem of discovering trend topics on news stream is thought as a dimension reduction and regression problem using lasso regression. Our algorithm works based on two important assumptions:

1 – Lasso regression is used for regression-prediction in linear models, therefore it is needed a training set with a defined output $Y = f(X)$. Every time an input comes we "classify" it on the fly assigning a dummy category at random; the category is selected from the set {-1, 1}. We assume a continue stream of data as a fictitious "training data".

2 – We assume a subset of sequential inputs vectors as a segment; its size is an important parameter in our trend topics detection system, due that it has close relation with the computing power and memory availability. During simulations explained in next section we used segments of size = 50; every time an input vector comes the system computes its results; when the vector number 50 comes the system restarts its parameters.

The algorithm described below detects main variables from a set of fixed variables that can be thought as a model's constrain, but if we guide the system's application to, for example, to discover trends topics during emergencies, it is viable to get satisfying insides from our proposed solution. The fixed variables set used throughout simulations was a predefined dictionary of 10,000 words related to natural disasters.

We want to emphasize, lasso is incorporated as part of the system not only to uncover main variables, but to comprise the "high-dimensional" problem of processing text. Reasonable to suppose that most of the coefficients of the words are exactly equal to 0.

Aside from segment's size, the shrinkage can be controlled; more or less variables can be shown depending on the problem, by varying complexity parameter δ.

1 – Initialize parameters: Select value of δ *, and segment's size.*
2 – Repeat until needed
3 - Receive input vector from data stream.
4 – Assign, at random, a dummy category from the set {-1, 1} to input vector.
5 – If size of segment is reached, initialize z_1 *and* z_2 *to zero (ready for next segment).*
6 - Calculate z_1 *(equation 13) (this calculation is accumulative over iterations, up to segment's size.)*
7 - For each feature j *perform:*
8-Calculate z_2 *(equation 13) (This calculation is performed for current segment; it is accumulative over iterations, up to segment's size)*
9-In case $z_2 < -\delta^2$ *perform:*
10 - Update $\hat{\theta}_j = \frac{z_2 + \delta^2}{z_1}$
11 - In case $z_2 > \delta^2$ *perform:*
12 -Update $\hat{\theta}_j = \frac{z_2 - \delta^2}{z_1}$
13 -Otherwise, update $\hat{\theta}_j = 0$
14 - Iterate from line 2 until needed.
15 - End

Above algorithm doesn't include considerations about velocity of the trend topics, but it shows how coefficients

θ for each important feature evolve over the time. This characteristic is analyzed at next section.

6. Results

Experiments were performed in a conventional laptop, with 4 GB of RAM and CPU @ 2.60 GHz. Segment's size was set to 50 inputs and amount of shrinkage $\delta = 7$. We reproduce a news stream environment using a huge collection of news linked to March 2011 earthquake / tsunami in Japan, sorted by date and time. News are entered into the system sequentially; for every new input, the system matches it to a set of 10,000 fixed features predefined. Thus, every news is transformed in a high-dimensional vector where each element is a number equal to the frequency of each term in the dictionary. Figure 1 shows simulation results up to 25 iterations; main features are immediately detected. Firstly, we can observe main topics such as "earthquake", "Japan", among others are exposed as significant in figure 1 major subplot. Horizontal axis enumerates the established 10,000 features, while vertical axis displays value of features coefficient $\theta's$. Lasso regression penalizes $\theta's$ for worthless topics, shrinking them gradually to zero, and at the same time rewards to trend topics coefficients stabilizing them.

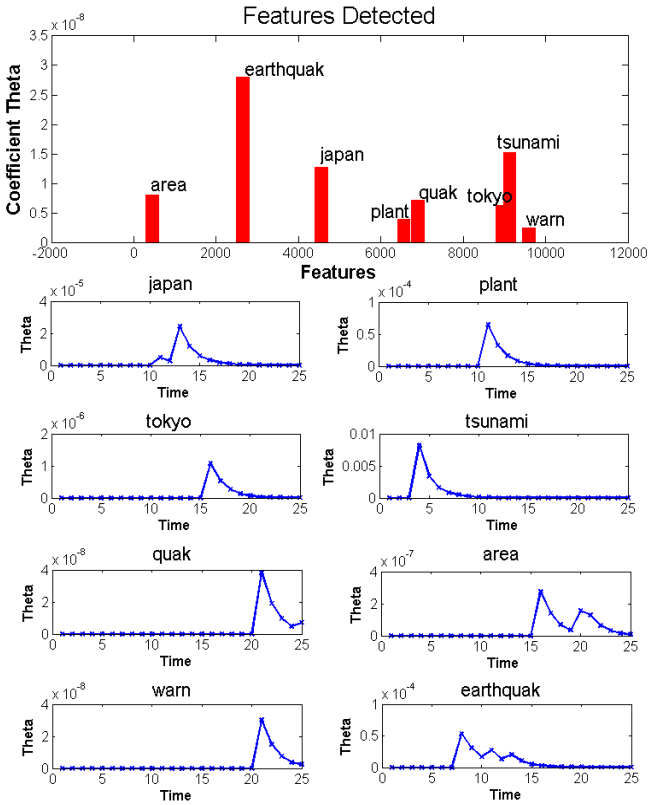

Fig. 1 Proposed algorithm results after 25 iterations.

In our system, coefficients $\theta's$ can be understood as a weighting value for each feature; in that sense, previous figure exposes "earthquake" as the most important topic, followed by "tsunami", "Japan" and others. Individual subplots show values of $\theta's$ for trend topics over the time; we believe that $\theta's$ model behavior of real trends topics, at the beginning when main issues are discovered they experiment a peak whose value gets stable after some time, that value may reach zero if the topic is not relevant anymore. For example, "tsunami" makes a peak at time 4; its weight $\theta_{tsunami}$ is strong because it just was discovered, through some interactions it becomes stable; although with a very small value it still pertinent.

Along our simulations, trend topics displayed peaks following a sequence comparable with how real facts occurred. At figure 1 our system detected the peaks' sequence: "tsunami", "earthquake", "plant", "Japan", "Tokyo", "area", "quake", "warn". Sequences analysis can be useful for future prevention plans and readiness.

Figure 2 below shows algorithm results up to iteration 420; in this case previously trends were replaced by weighty current issues, such as "nuclear", "plant", "people", "government".

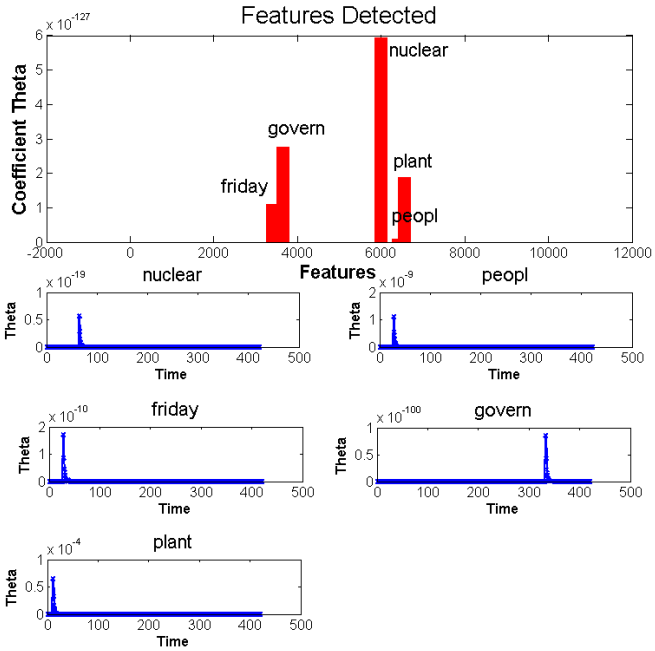

Fig. 2 Proposed algorithm results after 420 iterations.

As mentioned previously, some issues might require a deeper analysis, in that case the amount of shrinkage can be modified to lower values of δ to display more detailed trending topics.

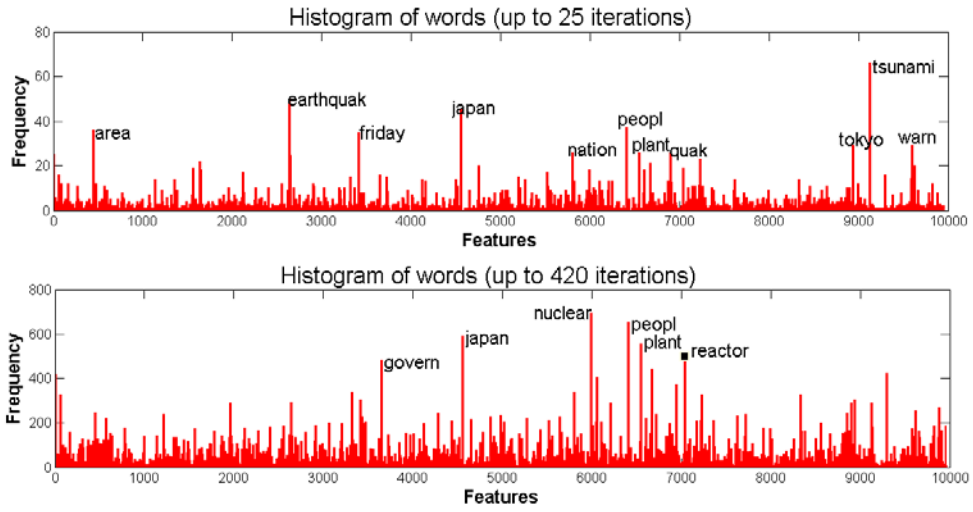

Fig. 3 Histograms of words.

Mainly, trending topics are found by counting inputs with repeated hashtags (#) tagged by users, as mentioned previously, Twitter's trend status is defined by a combination of velocity and volume of tweets containing the hashtag. Our proposal does not use hashtags; neither performs a systematic counting of terms. Experiments exposed at Fig. 1 and Fig. 2 are applied to a large set of sequential news including a major amount of terms; this characteristic demonstrates the scalability of our algorithm to different sizes of inputs vectors. Figure 3 shows histograms of words after 25 and 420 iterations. This plot can be compared with Fig. 1 and Fig. 2 respectively; counting of terms produces similar results, for example after 25 iterations our system delivers trending topics {"area", "earthquake", "Japan", "plant", "quake", "Tokyo", "tsunami", "warn"} which are comparable with most frequent words for 1st histogram {"area", "earthquake", "Friday", "Japan", "nation", "people", "plant", "quake", "Tokyo", "tsunami", "warn"}; equivalent conclusions can be obtained observing the results after 420 iterations. However, our solution incorporates a novel component, the weighting values $\theta's$ for each topic. This is an important issue with Twitter; its topics list constantly changes and does not apply a weight or relevancy to a particular topic. Our algorithm not only shows a significance value for discovered trends, but also how they behave over the time.

7. Conclusion

A system to uncover trend topics from a data stream was proposed; as far as we know it is the 1st time a linear regression / prediction model with shrinkage operators is used for detecting main topics on data stream. Our algorithm makes intense use of lasso model and adapts it to process streaming data; in a way that a trend is understood and modeled as a coefficient θ that serve as topic's weight. Coefficients θ and their evolution over time represents the "lifetime" of main topics detected. Although the amount of features to detect is bounded to a dictionary of 10,000 variables, our algorithm does not need special characters, like "#", to recognize a trend topic. The algorithm is not based on probability neither counting of variables occurrences. It is a pure application of a linear prediction model. As future work, we will adapt the algorithm to receive Twitter inputs.

References

[1] https://blog.twitter.com/2010/trend-or-not-trend
[2] T Hastie, R Tibshirani, J Friedman. The Elements of Statistical Learning. Data Mining, Inference and Prediction. Springer. 2nd edition. 2011.
[3] R Tibshirani. Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society. Volume 58, Issue 1 (1996), 267-288
[4] B Efron, T Hastie, I Johnstone, R Tibshirani. Least angle regression. Ann. Statist. 32. 407–499.
[5] R. Tibshirani, J Taylor. The solution path of the generalized lasso. Ann. Statis. 39. 1335-1371.
[6] C.E. Gutierrez, M. Alsharif, H. Cuiwei, M. Khosravy, R. Villa, K. Yamashita, H. Miyagi, Uncover news dynamic by principal component analysis. ICIC Express Letters, vol.7, no.4, pp.1245-1250, 2013.
[7] C.E. Gutierrez, M.R. Alsharif, H. Cuiwei, R. Villa, K. Yamashita, H. Miyagi, K. Kurata, Natural disaster online

news clustering by self-organizing maps. Ishigaki, Japan, 27th SIP symposium, 2012.
[8] C.E. Gutierrez, M.R. Alsharif, R. Villa, K. Yamashita, H. Miyagi, Data Pattern Discovery on Natural Disaster News. Sapporo, Japan, ITC-CSCC, ISBN 978-4-88552-273-4/C3055, 2012.

Carlos Enrique Gutierrez was born in Argentina. He received the B.Sc. from National University of Jujuy, Argentina in 2002, and M.Sc. degree from University of the Ryukyus, Japan in 2009. He is PhD candidate of Interdisciplinary Intelligent Systems at Department of Information Engineering, University of the Ryukyus, Japan. His research topics of interest are data mining, machine learning and big data.

Mohammad Reza Alsharif received the B.Sc. and M.Sc. degree in electrical engineering from the University of Tehran, in 1973 and 1974, respectively, and the Ph.D. degree in electrical engineering from the University of Tokyo in 1981. He was Head of Technical Department of IIRB College, Iran from 1981 to 1985. Then, he was a senior researcher at Fujitsu Labs. Co. Kawasaki, Japan from 1985 to 1992. Then, he was an assistant professor in the school of electrical and computer engineering, University of Tehran, Tehran, Iran from 1992 to 1997. From1997, Dr. Alsharif is a full professor at the Department of Information Engineering, University of the Ryukyus, Okinawa, Japan. He has developed an algorithm and implemented its hardware for real time T.V. Ghost canceling. He introduced a new algorithm for Acoustic Echo Canceller and he released it on VSP chips. His research topics of interest are in the field of Blind Source Separation, MIMO Speech and Image Processing, MIMO Communication systems, Echo Canceling, Active Noise Control and Adaptive Digital Filtering. He is a senior member of IEEE, and a member of IEICE.

Katsumi Yamashita received the B.E. degree from Kansai University, the M.E. degree from Osaka Prefecture University and the Dr. Eng. degree from Osaka University in 1974, 1976 and 1985, respectively, all in electrical engineering. In 1982, he became an assistant professor in University of the Ryukyus, where he became a professor in 1991. Now he is a professor in Osaka Prefecture University. His current interests are in digital communication and digital signal processing. Dr. Yamashita is a member of the IEEE, IEICE, and IEEJ.