TOPIC DETECTION AND COMPRESSED CLASSIFICATION IN TWITTER

Dimitris Milioris and Philippe Jacquet

Bell Labs, Alcatel-Lucent and École Polytechnique Paris
E-mail: {dimitrios.milioris, philippe.jacquet}@alcatel-lucent.com

ABSTRACT

In this paper we introduce a novel information propagation method in Twitter, while maintaining a low computational complexity. It exploits the power of Compressive Sensing in conjunction with a Kalman filter to update the states of a dynamical system. The proposed method first employs Joint Complexity, which is defined as the cardinality of a set of all distinct factors of a given string represented by suffix trees, to perform topic detection. Then based on the inherent spatial sparsity of the data, we apply the theory of Compressive Sensing to perform sparsity-based topic classification by recovering an indicator vector, while reducing significantly the amount of information from tweets, possessing limited power, storage, and processing capabilities, to a central server. We exploit datasets in various languages collected by using the Twitter streaming API and achieve better classification accuracy when compared with state-of-the-art methods.

Index Terms— Big Data, Text Classification, Joint Complexity, Combinatorics, Compressive Sensing, Kalman Filter

1. INTRODUCTION

Social Networks have undergone a dramatic growth in recent years and have changed the way we communicate with others, entertain and actually live. The communication between users has formed a new era with several research challenges, e.g. (a) real time search has to balance between quality, authority, relevance and timeliness of the content, (b) the relationship analysis between members of a social community can reveal the important teams which can be used for specific plans, (c) spam and advertisement detection to avoid the growth of irrelevant content. By extracting the relevant information from social networks in real time, we can address these challenges.

In this paper we use the theory of Joint Complexity (JC) to perform topic detection. The evaluation of the proposed method is based on the detection of real world topics like the categories of a mainstream news portal. We use large datasets, which are tweets from politics, economics, sport, technology and lifestyle. Then we classify new tweets into these categories with the power of Compressive Sensing (CS) by taking advantage of the inherent sparsity of the data, combined with a Kalman filter, as a refinement step for the update of the user's estimated class. The method is simple, context-free, with no grammar and no language assumptions, and does not use semantics. The sequence of text is decomposed in linear time into a memory efficient structure called Suffix Tree [1] (frequently used in DNA sequence analysis) and by overlapping two trees, in linear or sublinear average time, we obtain the JC defined as the cardinality of factors (subsequences) that are common in both trees. The method has been extensively tested for Markov sources of any order for a finite alphabet and gave good approximation for text generation and language discrimination, while being language-agnostic. Therefore there is no need to build any specific dictionary or stemming process.

The paper is organised as follows: Section 2 introduces the JC method, while Section 3 describes the classification via CS. Section 4 discusses the Kalman filter, while Section 5 evaluates the performance with real data obtained from Twitter. Finally, Section 6 summarises our main results and provides directions for future work.

2. JOINT COMPLEXITY

Several attempts have been made to capture mathematically the concept of complexity of a sequence, *i.e.*, the number of distinct factors contained in a sequence. If X is a sequence and $I(X)$ its set of factors, then $|I(X)|$ is the complexity of the sequence. For example if $X = “sunny''$ then $I(X) = \{$s, u, n, y, su, un, nn, ny, sun, unn, nny, sunn, unny, sunny, $v\}$ and $|I(X)| = 15$ (v denotes the empty string). The complexity of a string is called the $I-$complexity, and is connected with deep mathematical properties, including the rather elusive concept of randomness in a string [2], [3].

In general, a reference string is required to measure the information contained in a second string. In [4] the concept of JC of two strings was introduced as the number of common distinct factors of two sequences, i.e. the JC of sequence X and Y is equal to $J(X,Y) = |I(X) \cap I(Y)|$. Our previous experiments [5] showed that the mentioned complexity estimate for memoryless sources converges very slowly. Furthermore, memoryless sources are not appropriate for mod-

Dimitris Milioris would like to thank the “Alliance Doctoral Mobility Grant”. The work presented in this paper has been partially carried out at Columbia University, NY and at Bell Labs, Murray Hill, NJ, USA.

elling text generation. In a prior work, we extend JC estimate to Markov sources of any order on a finite alphabet. Markov models are more realistic and have a better approximation for text generation than memoryless sources [5], [6]. We derived a second order asymptotics for JC of the following form

$$\gamma \frac{n^{\kappa}}{\sqrt{\alpha \log n + \beta}}, \tag{1}$$

for some $\beta > 0$, $\kappa < 1$ and $\gamma, \alpha > 0$ which depend on the parameters of the two sources. This new estimate has a faster convergence, and is preferred for texts of order $n \approx 10^2$; For some Markov sources our analysis indicates that JC oscillates with n. This is outlined by the introduction of a periodic function $Q(\log n)$ in the leading term of our asymptotics. This additional term even further improves the convergence for small values of n and therefore JC is an efficient method to capture the similarity degree of short texts, *e.g.* tweets.

Up to now, the main methods used for text classification are based on *keywords* detection and Machine Learning techniques, described extensively in [7]. Using keywords in tweets will often fail because of the distorted usage of the words – which also need lists of keywords for every language to be built – or because of implicit references to previous texts or messages. Machine Learning techniques are generally heavy and complex and therefore are not good candidates for real time text classification, especially in the case of Twitter where we have natural language and thousands of tweets per second to process. Furthermore Machine Learning processes have to be manually initiated by tuning parameters, and it is one of the main drawbacks for that kind of application. Some other methods are using information extracted by visiting the specific URLs on the text, which makes them a heavy procedure, since one may have limited or no access to the information, *e.g.* because of access rights, or data size.

In this work, we construct the training databases (DBs) by using Twitter's Streaming API while filtering for specific keywords. For example, we build a class about politics by sending a request to Twitter API for tweets that contain the word "politics". Using these requests we build C classes on different topics. Assume that each class contains N tweets (*eg.* $C = 5$, i.e. Classes: politics, economics, sports, technology, lifestyle of $N = 12,000$ tweets). We allocate a number of keywords to each class (*e.g.* the keywords used to populate the class). The tweets come in the basic *.json* format delivered by the Twitter API.

Then, assume that we have a dataset of S timeslots with $s = 1 \dots S$, and each timeslot is a 15 minutes request in Twitter API. For every tweet x_i, where $i = 1 \dots N$, with N being the total number of tweets, in the s-th timeslot, *i.e* x_i^s, we build a Suffix Tree, $ST(x_i^s)$, as described in Section 2. Building an ST costs $O(m \log m)$ and takes $O(m)$ space in memory, where m is the length of the tweet, in comparison of m^3 needed by algorithms based on semantic analysis.

Then we compute the JC metric, $JC(x_i^s, x_j^s)$ of the tweet x_i^s with every other tweet x_j^s of the s-th timeslot, where $j = 1 \dots N$, and $j \neq i$ (by convention we choose $JC(x_i^s, x_i^s) = 0$). The JC between two tweets can be computed efficiently in $O(m)$ operations (sublinear on average) by ST superposition. For the S timeslots we store the JC scores in the matrices $S_1, S_2, \dots, S_S$ of $N \times N$ dimensions.

We represent timeslots by fully-connected edge weighted graphs, where each tweet is a node in the graph and S_n holds the weight of each edge [8]. Then, we calculate the score for each node by summing weights of all the edges that are connected to that node. The node that gives the highest score is the most representative and central tweet of the timeslot. Most of the timeslots have $N = 12,000$ tweets, so matrices $S_1, S_2, \dots, S_S$ have approximately 144×10^6 entries for each timeslot. Since the score matrices are symmetric, only half of these entries could be used, *i.e* the upper/lower triangular, in order to reduce the complexity to $\frac{N^2}{2}$.

We then assign a new tweet to the class that maximises the JC metric inside that class. In order to limit the size of each reference class we delete the oldest tweets or the least significant ones (e.g. the ones which obtained the lowest JC score). This ensures the *low cost* and *efficiency* of the method.

3. COMPRESSIVE SENSING

Let us first describe the main theoretical concepts of CS [9] as applied in the context of classification. Let $\mathbf{x} \in \mathbb{R}^N$ denote the signal of interest. Such signal can be represented as a linear combination of a set of basis $\{\psi_i\}_{i=1}^N$. By constructing a $N \times N$ basis matrix $\mathbf{\Psi} = [\psi_1, \psi_2, \dots, \psi_N]$, the signal x can be expressed as $x = \sum_{i=1}^N s_i \psi_i = \mathbf{\Psi} s$. In fact the signal is represented as $x = \mathbf{\Psi} s + \theta$, with $\theta \in \mathbb{R}^N$ being the noise, where $\mathbb{E}(\theta) = 0$ and $var(\theta) = O(|\Psi s|)$.

The efficiency of a CS method for signal approximation or reconstruction depends highly on the sparsity structure of the signal in a suitable transform domain associated with an appropriate sparsifying basis $\mathbf{\Psi}$. The measurement model in the original space-domain is expressed as $\mathbf{g} = \mathbf{\Phi x}$, where $\mathbf{g} \in \mathbb{R}^M$ is the measurement vector and $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ denotes the measurement matrix. The measurement model has the following equivalent transform-domain representation

$$\mathbf{g} = \mathbf{\Phi \Psi s} + \mathbf{\Phi} \boldsymbol{\theta} \; . \tag{2}$$

In fact when the length of the sequence $n \to \infty$ and $N \to \infty$, $\mathbb{E}(\mathbf{\Psi} \boldsymbol{s}) = O(nN)$, with $var(\theta) = O(nN)$, $std(\theta) = O(\sqrt{|\Phi| n})$ and $\mathbb{E}(\mathbf{\Phi}\boldsymbol{\theta}) = 0$. The second part of (2), $\mathbf{\Phi}\boldsymbol{\theta}$ is of relative order $O(\frac{1}{\sqrt{nN}})$, and is negligible compare to $\mathbf{\Phi \Psi s}$ due to the law of large numbers.

Examples of measurement matrices $\mathbf{\Phi}$, which are incoherent with any fixed transform basis $\mathbf{\Psi}$ with high probability (universality property [11]), are random matrices with independent and identically distributed (i.i.d.) Gaussian or Bernoulli entries.

In the framework of CS, the problem of classifying a tweet is reduced to a problem of recovering the one-sparse vector $\mathbf{s}$. Of course in practice we do not expect an exact sparsity, thus, the estimated class corresponds simply to the largest-amplitude component of $\mathbf{s}$. According to [10, 11], $\mathbf{s}$ can be recovered perfectly with high probability by solving the following optimization problem

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \Big(\|\mathbf{s}\|_1 + \tau \|\mathbf{g} - (\mathbf{\Phi}\mathbf{\Psi}\mathbf{s})\|_2 \Big) , \tag{3}$$

where τ is a regularization factor that controls the trade-off between the achieved sparsity and the reconstruction error.

3.1. Preprocessing phase

During the Preprocessing phase, we built our classes as described in Section 2 and for each class we extract the most representative tweet(s) (CTs) based on the Joint Complexity method. The vector $\mathbf{\Psi}_T^i$ consists of the highest JC scores of the i-th CT. The matrix $\mathbf{\Psi}_T$ is used as the appropriate sparsifying dictionary for the training phase. Moreover, a measurement matrix $\mathbf{\Phi}_T^i$ is associated with each transform matrix $\mathbf{\Psi}_T^i$, while T denotes the preprocessing phase.

The matrix $\mathbf{\Psi}_T^i \in \mathbb{R}^{N_i \times C}$ is used as the appropriate sparsifying dictionary for the i-th CT, since in the ideal case the vector of tweets at a given class j received from CT i should be closer to the corresponding vectors of its neighboring classes, and thus it could be expressed as a linear combination of a small subset of the columns of $\mathbf{\Psi}_T^i$. Moreover, a measurement matrix $\mathbf{\Phi}_T^i \in \mathbb{R}^{M_i \times N_i}$ is associated with each transform matrix $\mathbf{\Psi}_T^i$, where M_i is the number of CS measurements. In the proposed algorithm, a standard Gaussian measurement matrix is employed, with its columns being normalized to unit ℓ_2 norm. A random matrix or a PCA matrix could be also used.

3.2. Run phase

A similar process is followed during the runtime phase. More specifically, we denote $\mathbf{x}_{c,R}$ as the Joint Complexity score of the incoming tweet with the CT_i classified at the current class c, where R denotes the runtime phase. The runtime CS measurement model is written as

$$\mathbf{g}_c = \mathbf{\Phi}_R \mathbf{x}_{c,R} , \tag{4}$$

where $\mathbf{\Phi}_R^i \in \mathbb{R}^{M_{c,i} \times N_{c,i}}$ denotes the corresponding measurement matrix during the runtime phase. In order to overcome the problem of the difference in dimensionality between the preprocessing and run phase, while maintaining the robustness of the reconstruction procedure, we select $\mathbf{\Phi}_R^i$ to be a subset of $\mathbf{\Phi}_T^i$ with an appropriate number of rows such as to maintain equal measurement ratios.

The measurement vector $\mathbf{g}_{c,i}$ is formed for each CT i according to (4) and transmitted to the server, where the reconstruction takes place via the solution of (3), with the training matrix $\mathbf{\Psi}_T^i$ being used as the appropriate sparsifying dictionary. We emphasize at this point the significant conservation of the processing and bandwidth resources of the wireless device by computing only low-dimensional matrix-vector products to form $\mathbf{g}_{c,i}$ $(i = 1, \ldots, P)$ and then transmitting a highly reduced amount of data ($M_{c,i} \ll N_{c,i}$). Then, the CS reconstruction can be performed remotely (*e.g.*, at a server) for each CT independently.

In this work, we are based on the assumption that the CS-based classification method involves the mobile device that collects the tweets from the Twitter API and a server that performs the core CS algorithm.

4. TRACKING VIA KALMAN FILTER

Focusing on the problem of classification, the user tweets periodically, and we check that information with the CTs at a specific time interval Δt.

Then, the classification system estimates the user's class at time t, which is denoted by $p^*(t) = [x^*(t)]^T$. Following a Kalman filtering approach [12], we assume that the process and observation noises are Gaussian, and also that the motion dynamics model is linear. The process and observation equations of a Kalman filter-based model are given by $\mathbf{x}(t) = \mathbf{F}\mathbf{x}(t-1) + \boldsymbol{\theta}(t)$ and $\mathbf{z}(t) = \mathbf{H}\mathbf{x}(t) + \mathbf{v}(t)$, where $\mathbf{x}(t) = [x(t), v_x(t)]^T$ is the state vector, with x being the correct class in the space (user's tweets) and $v_x(t)$ the tweeting frequency, $\mathbf{z}(t)$ is the observation vector, while matrices $\mathbf{F}$ and $\mathbf{H}$ define the linear motion model. The process noise $\boldsymbol{\theta}(t) \sim N(\mathbf{0}, \mathbf{S})$ and the observation noise $\mathbf{v}(t) \sim N(\mathbf{0}, \mathbf{U})$ are assumed to be independent zero-mean Gaussian vectors with covariance matrices $\mathbf{S}$ and $\mathbf{U}$, respectively. The current class of the user is assumed to be the previous one plus the information provided by the JC metric, which is computed as the time interval Δt multiplied by the current tweeting speed/frequency.

The steps to update the current estimate of the state vector $\mathbf{x}^*(t)$, as well as its error covariance $\mathbf{P}(t)$, during the prediction and update phase are given by the following equations

$$\begin{aligned}
\mathbf{x}^{*-}(t) &= \mathbf{F}\mathbf{x}^*(t-1) && (5)\\
\mathbf{P}^-(t) &= \mathbf{F}\mathbf{P}(t-1)\mathbf{F}^\mathbf{T} + \mathbf{S} && (6)\\
\mathbf{K}(t) &= \mathbf{P}^-(t)\mathbf{H}^T(\mathbf{H}\mathbf{P}^-(t)\mathbf{H}^T + \mathbf{U})^{-1} && (7)\\
\mathbf{x}^*(t) &= \mathbf{x}^{*-}(t) + \mathbf{K}(t)(\mathbf{z}(t) - \mathbf{H}\mathbf{x}^{*-}(t)) && (8)\\
\mathbf{P}(t) &= (\mathbf{I} - \mathbf{K}(t)\mathbf{H})\mathbf{P}^-(t) && (9)
\end{aligned}$$

where the superscript "$-$" denotes the prediction at time t, and $\mathbf{K}(t)$ is the optimal Kalman gain at time t.

The proposed Kalman system exploits not only the highly reduced set of compressed measurements, but also the previous user's class to restrict the classification set. The Kalman filter is applied on the CS-based classification, described briefly in Section 3, to improve the estimation accuracy of the mobile user's path. More specifically, let $\mathbf{s}^*$ be the reconstructed position-indicator vector. Of course in practice $\mathbf{s}^*$ will be not truly sparse, thus the current estimated position $[x_{CS}]$, or equivalently, cell c_{CS}, corresponds to the highest-amplitude index of $\mathbf{s}^*$. Then, this estimate is given as an input to the Kalman filter by assuming that it corresponds to the previous time $t-1$, that is, $\mathbf{x}^*(t-1) = [x_{CS}, v_x(t-1)]^T$, and the current position is updated using (5). At this point, we would like to emphasize the computational efficiency of the proposed approach, since it is solely based on the use of the very low-dimensional set of compressed measurements given by (2), which are obtained via a simple matrix-vector multiplication with the original high-dimensional vector. Finally, Figure 1 presents the flowchart of the classification and tracking model based on Joint Complexity, Compressive Sensing and Kalman filter.

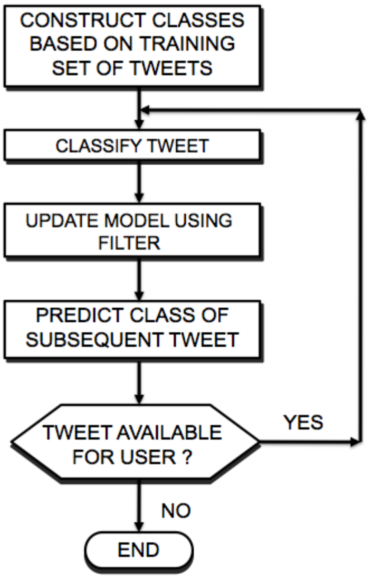
CONSTRUCT CLASSES BASED ON TRAINING SET OF TWEETS
CLASSIFY TWEET
UPDATE MODEL USING FILTER
PREDICT CLASS OF SUBSEQUENT TWEET
TWEET AVAILABLE FOR USER ?
YES
NO
END

Fig. 1. Flowchart of the proposed topic detection and classification method.

5. EXPERIMENTAL RESULTS

The efficiency of the proposed classification method is evaluated on sets of tweets acquired from the Twitter API. The classification accuracy of the tested methods was measured with the standard *Precision*, *Recall*, and *balanced F-score* metrics, using a Ground Truth (GT) on more than $1,041,000$ tweets [13]. Then we kept track of which tweet was classified in which class in order to compare this with four classification methods along with many different optimisation techniques for the signal reconstruction mentioned in Section 3.

We selected the Document-Pivot (DP) method to compare with our method, since it outperformed the other state-of-the-art techniques in a Twitter context as shown in [7]. The most important difference of DP method is that instead of building suffix trees, this time the method constructs a *tf-idf* bag of words), and then classifies each tweet of the Run phase by selecting the category containing the tweet *closest* to our test tweet. The notion of *closest* is because we used Locality Sensitive Hashing based on the Cosine Similarity in a vector space where each possible word is a dimension and its *tf-idf* score is the coordinate in that dimension. In such a space when the cosine between the two vectors is close to one, it means that the vectors are pointing in the roughly same direction, in other words the two tweets represented by the vectors should share a lot of words and thus should probably speak about or refer to the same subject.

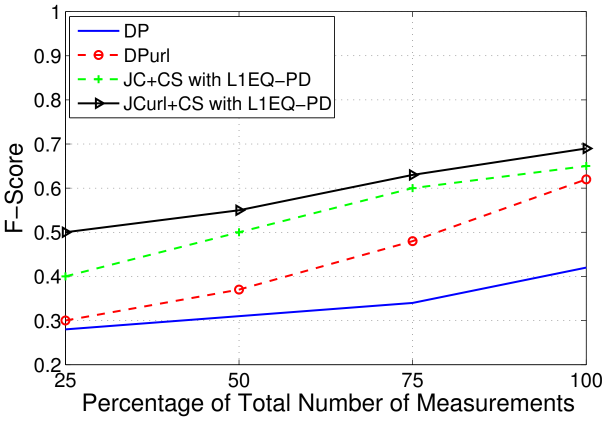

Fig. 2. Classification accuracy measured by F-Score for the DP, DPurl and JC+CS, JCurl+CS method as a function of the number of measurements (%) by using the ℓ_1-norm min.

5.1. Classification performance based on Ground Truth

The classification performance is compared for: (a) Document Pivot (DP), (b) Joint Complexity with Compressive Sensing (JC+CS), (c) Document Pivot with URL (DPurl), (d) Joint Complexity and Compressive Sensing with URL (JCurl+CS), where (c) and (d) include the information of the compressed URL of a tweet concatenated with the original tweet's text (extracted from the *.json* file).

Fig. 2 compares the classification accuracy (increased by 27%) of the DP, DPurl and JC+CS, JCurl+CS method as a function of the number of measurements by using the ℓ_1-norm minimization. Fig. 3 compares the reconstruction performance between several widely-used norm-based techniques and Bayesian CS algorithms. More specifically, the following methods are employed[1]: 1) ℓ_1-norm minimization using the primal-dual interior point method (L1EQ-PD), 2) Orthogonal Matching Pursuit (OMP), 3) Stagewise Orthogonal Matching Pursuit (StOMP), 4) LASSO, 5) BCS, and 6)

[1] For the implementation of methods 1)-5) the MATLAB codes can be found in: `http://sparselab.stanford.edu/`, `http://www.acm.caltech.edu/l1magic`, `http://people.ee.duke.edu/~lcarin/BCS.html`

BCS-GSM [14, 15]. Fig. 3 shows that BCS and BCS-GSM outperform the introduced reconstruction techniques, while Fig. 4 shows that we a achieve better performance of 11% when using the Kalman filter.

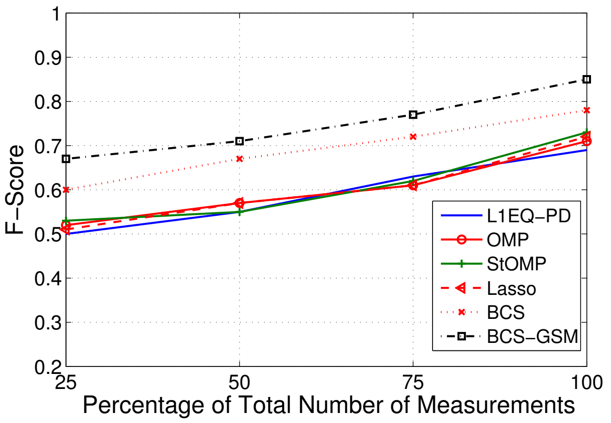

Fig. 3. Classification accuracy measured by F-Score as a function of the number of measurements (%) by using several reconstruction techniques, for the JCurl+CS method.

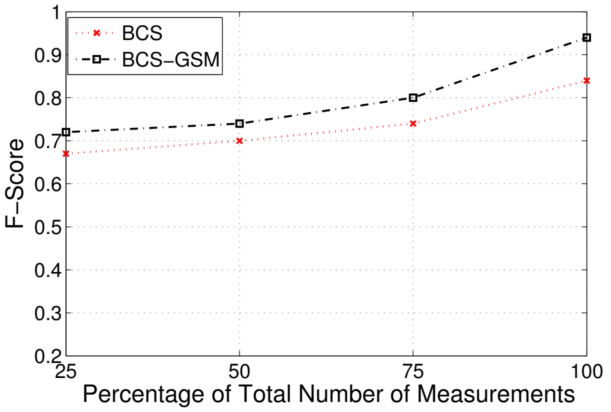

Fig. 4. Classification accuracy measured by F-Score as a function of the number of measurements (%) by using Kalman, for the JCurl+CS method.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we performed topic detection based on Joint Complexity and low dimensional classification based on Compressive Sensing with the accuracy of a Kalman filter as a refinement step. The experimental evaluation revealed better performance, while maintaining a low computational complexity. As a future work, we intend to exploit the joint sparsity structure of the indicator vector among the several representative tweets, improving the reconstruction accuracy, while investigating the inherent encryption properties of CS for potential employment in classification on cloud services.

REFERENCES

[1] S. Tata, R. Hankins and J. Patel, “Practical Suffix Tree Construction”, *in Proceedings of the 30th VLDB Conference*, 2004.

[2] M. Li and P. Vitanyi, “Introduction to Kolmogorov Complexity and its Application”, Springer–Verlag, Berlin, Aug. 1993.

[3] T. Hellseth and H. Niederreiter, “Some computable complexity measures for binary sequences”, *in Sequences and Their Applications*, Eds. C. Ding, Springer–Verlag, 67–78, 1999.

[4] P. Jacquet, “Common words between two random strings”, *in IEEE International Symposium on Information Theory (ISIT)*, Nice, France, June 2007.

[5] P. Jacquet, D. Milioris and W. Szpankowski, “Classification of Markov Sources Through Joint String Complexity: Theory and Experiments”, *in IEEE International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, July 2013.

[6] D. Milioris and P. Jacquet, “Joint Sequence Complexity Analysis: Application to Social Networks Information Flow”, *in Bell Labs Technical Journal*, Vol. 18, No. 4, 2014 (doi: 10.1002/bltj.21647).

[7] L. M. Aiello *et al.*, “Sensing Trending Topics in Twitter”, *in IEEE Transactions on Multimedia*, Vol. 15, Iss. 6, pp. 1268–1282, Oct 2013.

[8] G. Burnside, D. Milioris and P. Jacquet. “One Day in Twitter: Topic Detection Via Joint Complexity”, *Proceedings of SNOW 2014 Data Challenge, WWW’14, Seoul, South Korea*, April 2014.

[9] D. Milioris *et al.*, “Low-dimensional signal-strength fingerprint-based positioning in wireless LANs”, *in Ad Hoc Networks Journal, Elsevier*, Vol. 12, pp. 100–114, Jan. 2014.

[10] E. Candés, J. Romberg, and T. Tao, “Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information,” *in IEEE Transactions on Information Theory*, Vol. 52, pp. 489–509, Feb. 2006.

[11] D. Donoho, “Compressive sensing”, *in IEEE Transactions on Information Theory*, Vol. 52, No. 4, pp. 1289–1306, April 2006.

[12] M. Grewal and A. Andrews, “Kalman filtering: Theory and practice using MATLAB”, 2nd Edition, 2001.

[13] S. Papadopoulos, D. Corney, and L. M. Aiello. “Snow 2014 data challenge: Assessing the performance of news topic detection methods in social media,” in *Proceedings of the SNOW 2014 Data Challenge*, WWW’14, Seoul, Korea, Apr. 2014.

[14] G. Tzagkarakis and P. Tsakalides, “Bayesian compressed sensing imaging using a Gaussian scale mixture”, in *in Proc. 35th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Dallas, TX, USA*, March 2010.

[15] G. Tzagkarakis, D. Milioris and P. Tsakalides, “Multiple-Measurement Bayesian Compressive Sensing using GSM Priors for DOA Estimation”, *in Proc. 35th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Dallas, TX, USA*, March 2010.