Trend Detection and Information Propagation in Dynamic Social Networks

Dimitrios Milioris

▶ **To cite this version:**

Dimitrios Milioris. Trend Detection and Information Propagation in Dynamic Social Networks. Document and Text Processing. École Polytechnique, 2015. English. NNT : . tel-01152275

HAL Id: tel-01152275
https://pastel.hal.science/tel-01152275

Submitted on 15 May 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A DISSERTATION PRESENTED BY

DIMITRIOS MILIORIS

TO

THE COMPUTER SCIENCE DEPARTMENT

IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF SCIENCE

IN THE SUBJECT OF

TREND DETECTION AND INFORMATION PROPAGATION IN DYNAMIC SOCIAL NETWORKS

ÉCOLE POLYTECHNIQUE

prepared at Bell Laboratories, Alcatel-Lucent,
Mathematics of Complex and Dynamic Networks Team

Paris, April, 2015

Submitted to the
Department of Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Science

© Copyright, Dimitrios S. Milioris, April, 2015
All Rights Reserved

Author:

Dimitrios MILIORIS
Department of Computer Science

Committee:

Advisor

Philippe JACQUET
Bell Labs & École Polytechnique

Advisor

Paul MÜHLETHALER
INRIA

Rapporteur

Laurent VIENNOT
INRIA

Rapporteur

Bernard MANS
Macquarie University, Sydney

Rapporteur

Valérie ISSARNY
UC Berkeley & INRIA

Examinateur

James ROBERTS
SystemX

École Polytechnique, Paris, April 2015

Acknowledgments

It is a pleasure to thank the many people who made this thesis possible.
I would like to express my gratitude to my supervisors, Dr. Philippe Jacquet and Dr. Paul Mühlethaler, who were abundantly helpful and offered invaluable assistance, support and guidance.

Deepest gratitude are also due to Prof. Wojciech Szpankowski and Prof. Panagiotis Tsakalides, without whose knowledge and assistance this study would not have been successful.

I would like also to thank Dr. Iraj Saniee for hosting me at Bell Labs in Murray Hill, New Jersey, and Prof. Augustin Chaintreau for hosting me at Columbia University in the city of New York, during my Alliance Fellowship.

I wish to offer my regards and blessings to all of my best friends in undergraduate and graduate level for helping me get through the difficult times, and for all the emotional support, camaraderie, entertainment, and caring they provided.

I would like also to convey thanks to the École Polytechnique, Inria, Bell Laboratories and Columbia University for providing the financial means and laboratory facilities.

Lastly, and most importantly, I wish to thank my sister, Maria, and my parents, Milioris Spyridonas and Anastasiou Magdalini. They bore me, raised me, supported me, taught me, and loved me. To them I dedicate this Doctorate.

Milioris S. Dimitrios

Trend detection and information propagation in dynamic social networks

Abstract: During the last decade, the information within *Dynamic Social Networks* has increased dramatically. The ability to study the interaction and communication between users in these networks can provide real time valuable prediction of the evolution of the information.

The study of social networks has several research challenges, *e.g.* (a) real time search has to balance between quality, authority, relevance and timeliness of the content, (b) studying the information of the correlation between groups of users can reveal the influential ones, and predict media consumption, network and traffic resources, (c) detect spam and advertisements, since with the growth of social networks we also have a continuously growing amount of irrelevant information over the network. By extracting the relevant information from online social networks in real time, we can address these challenges.

In this thesis a novel method to perform topic detection, classification and trend sensing in short texts is introduced. Instead of relying on words as most other existing methods which use *bag-of-words* or n-gram techniques, we introduce *Joint Complexity*, which is defined as the cardinality of a set of all distinct common factors, subsequences of characters, of two given strings. Each short sequence of text is decomposed in linear time into a memory efficient structure called Suffix Tree and by overlapping two trees, in linear or sublinear average time, we obtain the cardinality of factors 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. The proposed method is language-agnostic since we can detect similarities between two texts in any loosely character-based language. The method is not based on a specific grammar or semantics, therefore there is no need to build any specific dictionary or stemming technique. The proposed method can be used to capture a change of topic within a conversation, as well as the style of a specific writer in a text.

In the second part of the thesis, we take advantage of the nature of the data, which motivated us in a natural fashion to use of the theory of *Compressive Sensing* driven from the problem of target localization. *Compressive Sensing* states that signals which are sparse or compressible in a suitable transform basis can be recovered from a highly reduced number of incoherent random projections, in contrast to the traditional methods dominated by the

well–established Nyquist–Shannon sampling theory. Based on the spatial nature of the data, we apply the theory of Compressive Sensing to perform topic classification by recovering an indicator vector, while reducing significantly the amount of information from tweets. The method works in conjunction with a Kalman filter to update the states of a dynamical system as a refinement step.

In this thesis we exploit datasets collected by using the Twitter streaming API, gathering tweets in various languages and we obtain very promising results when comparing to state-of-the-art methods.

Keywords: Dynamic Social Networks, Joint Sequence Complexity, Analytic Combinatorics, Compressive Sensing, Sparse Representation, Kalman Filter

Détection des tendances et la propagation des informations dans les réseaux sociaux dynamiques

Résumé: Au cours de la dernière décennie, la dissémination de l'information au travers des réseaux sociaux a augmenté de facon spectaculaire. L'analyse des interactions entre les utilisateurs de ces réseaux donne la possibilité de la prédiction en temps réel de l'évolution de l'information.

L'étude des réseaux sociaux présentent de nombreux défis scientifiques, comme par exemple : (a) peut on trouver un compromis entre la qualité, l'autorité, la pertinence et l'actualité du contenu ? (b) Peut on utiliser les interactions entre les groupes d'utilisateurs pour révéler les utilisateurs influents, pour prédire les pics de trafic ? (c) la publicité, les spams, et autres trafics non pertinent peuvent ils être détectés et écartés ?

Dans cette thèse, nous proposons une nouvelle méthode pour effectuer la détections dans les textes courts des sujets et des tendances, et leur classification. Au lieu de découper les textes en mots ou en n-grames comme le font la plupart des autres méthodes qui utilisent des *sac-de-mots*, nous introduisons la *Complexité Jointe*, qui est définie comme le cardinal de l'ensemble des facteurs communs distincts entre les deux textes, un facteur étant une chaîne de caractères consécutifs. L'ensemble des facteurs d'un texte est décomposé en temps linéaire en une structure efficace de mémoire appelée arbre suffixe et on obtient par le superposition des deux arbres, en temps moyen sous-linéaire, la complexité jointe des deux textes. La méthode a été largement testée à grande échelle pour des sources de texte de Markov d'ordre fini et permet en effet une bonne discrimination des sources (langue, etc). La simulation de la production des textes par processus de Markov est une approximation satisfaisante de la génération de textes en langage naturel. La méthode de la complexité jointe est indépendante de la langue agnostique puisque nous pouvons détecter les similitudes entre deux textes sans avoir recours à l'analyse sémantique. Elle ne nécessite pas une analyse sémantique sur la base d'une grammaire spécifique, par conséquent, il ne est pas nécessaire de construire un dictionnaire spécifique. La méthode proposée peut aussi être utilisé pour détecter un changement de thème dans une conversation, ainsi qu'un changement de style d'un écrivain dans un texte.

Dans la deuxième partie de la thèse, nous profitons de la faible densité de l'espace des données, ce qui nous a motivé de facon naturelle à appliquer la théorie de *Compressive Sensing* extrapolée du problème de la localisation

des objets physiques. Le *Compressive Sensing* stipule que les signaux qui sont rares ou compressibles peuvent être récupérés à partir d'un nombre très réduit de projections aléatoires incohérentes dans une base appropriée, contrairement aux méthodes traditionnelles dominées par la théorie classique de Nyquist-Shannon de l'échantillonnage. Grâce à la faible densité spatiale des sujets, nous appliquons la théorie pour récupérer un vecteur d'indicateur, à partir de l'ensemble des tweets. Le procédé fonctionne en conjonction avec un filtre de Kalman pour mettre à jour des états d'un système dynamique comme étape de raffinement.

Dans cette thèse, nous exploitons des ensembles de données recueillies en utilisant le flux de l'API de Twitter, sur des tweets collectés en plusieurs langues et nous obtenons des résultats très prometteurs lorsque l'on compare ces méthodes au meilleur de l'existant.

Mots-clés: Réseaux sociaux dynamiques, complexité jointe, analyse combinatoire, Compressive Sensing, Sparse Representation, Filtre Kalman

Publications, Technical Reports and Patents during this thesis

Publications

1. **D. Milioris** and P. Jacquet, "Topic Detection and Compressed Classification in Twitter", **submitted to** *European Signal Processing Conference (EUSIPCO'15), Nice, France*, September 2015.

2. **D. Milioris** and P. Jacquet, "Classification in Twitter via Compressive Sensing", *in IEEE International Conference on Computer Communications (INFOCOM'15), Hong Kong, SAR China*, Short Paper, April 2015.

3. **D. Milioris**, M. Bradonjić and P. Mühlethaler, "Building Complete Training Maps for Indoor Location Estimation", *in IEEE International Conference on Computer Communications (INFOCOM'15), Hong Kong, SAR China*, Short Paper, April 2015.

4. P. Jacquet and **D. Milioris**, "Information in Strings: Enumeration of Eulerian Paths and Eulerian Components in Markov Sequences", *in International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA'14), Paris, France*, June 2014.

5. **D. Milioris** and P. Jacquet, "SecLoc: Encryption System Based on Compressive Sensing Measurements for Location Estimation", *in IEEE International Conference on Computer Communications (INFOCOM'14), Toronto, Canada*, short paper, May 2014.

6. G. Burnside, **D. Milioris** and P. Jacquet, "One Day in Twitter: Topic Detection Via Joint Complexity", *in Snow Data Challenge, WWW'14, Seoul, South Korea*, April 2014.

7. P. Mirowski, **D. Milioris**, P. Whiting and T. Kam Ho, "Probabilistic RF Fingerprinting and Localization on the Run", *in Bell Laboratories Technical Journal, Issue on Data Analytics*, Vol. 18, No. 4, 2014.

8. **D. Milioris** and P. Jacquet, "Joint Sequence Complexity Analysis: Application to Social Networks Information Flow", *in Bell Laboratories Technical Journal, Issue on Data Analytics*, Vol. 18, No. 4, 2014.

9. **D. Milioris**, G. Tzagkarakis, A. Papakonstantinou, M. Papadopouli and P. Tsakalides, "Low-dimensional Signal-Strength Fingerprint-based Positioning in Wireless LANs", *in Elsevier Ad Hoc Networks*, Vol. 12, pp. 100–114, January 2014.

10. **D. Milioris**, G. Tzagkarakis, P. Tsakalides and P. Jacquet, "WLAN-based Indoor Path-Tracking using Compressive RSS Measurements", *in 21st European Signal Processing Conference (EUSIPCO'13), Marrakech, Morocco*, September 2013.

11. P. Jacquet, **D. Milioris** and P. Mühlethaler, "A novel energy efficient broadcast leader election", *in IEEE 21st International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'13), San Francisco, CA, USA*, August 2013.

12. 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'13), Istanbul, Turkey*, July 2013.

Technical Reports

1. P. Jacquet and **D. Milioris**, "Stealth Encryption Based on Eulerian Circuits", *Alcatel-Lucent, Bell Laboratories*, February 2014.

Patents

1. **D. Milioris**, "Text Classification Based on Joint Complexity and Compressed Sensing", **US 14/540770**, Filed November 13, 2014.

2. P. Jacquet and **D. Milioris**, "HoneyPot Design via Random Eulerian Paths", **EP 14306779.1**, Filed November 6, 2014.

3. P. Jacquet, **D. Milioris** and G. Burnside, "Textual Steganography - Undetectable Encryption based on N-Gram Rotations", **EP 14306482.2**, Filed September 25, 2014.

4. **D. Milioris**, "Method, User Equipment, System and Computer Readable Medium for Localizing a User Equipment", **EP 14306453.3**, Filed September 22, 2014.

5. P. Jacquet and **D. Milioris**, "Undetectable Encryption based on M-grams Permutations", **EP 14305064.9**, Filed January 17, 2014.

6. **D. Milioris** and P. Jacquet, "Method and Device for Classifying a Message", **EP 13306222.4**, Filed September 6, 2013.

Contents

Chapter 1

Introduction

Contents

1.1 Dynamic Social Networks

Social networks have undergone a dramatic growth in recent years. Such networks provide an extremely suitable space to instantly share multimedia information between individuals and their neighbours in the social graph. Social networks provide a powerful reflection of the structure, the dynamics of the society and the interaction of the Internet generation with both people and technology. Indeed, the dramatic growth of social multimedia and user generated content is revolutionising all phases of the content value chain including production, processing, distribution and consumption. It also originated and brought to the multimedia sector a new underestimated and now critical

aspect of science and technology, which is social interaction and networking. The importance of this new rapidly evolving research field is clearly evidenced by the many associated emerging technologies and applications, including (a) online content sharing services and communities, (b) multimedia communication over the Internet, (c) social multimedia search, (d) interactive services and entertainment, (e) health care and (f) security applications. It has generated a new research area called social multimedia computing, in which well established computing and multimedia networking technologies are brought together with emerging social media research.

Social networking services are changing the way we communicate with others, entertain and actually live. Social networking is one of the primary reasons why more people have become avid Internet users, people who until the emergence of social networks could not find interests in the Web. This is a very robust indicator of what is really happening online. Nowadays, users both produce and consume significant quantities of multimedia content. Moreover, their behaviour through online communities, is forming a new Internet era where multimedia content sharing through Social Networking Sites (SNSs) is an everyday practice. More than 200 SNSs of worldwide impact are known today and this number is growing quickly. Many of the existing top web sites are either SNSs or offer some social networking capabilities.

Except for the major social networks with hundreds of millions of users that span in the entire world, there are also many smaller SNSs which are equally as popular as the major social networks within the more limited geographical scope of their membership, *e.g.* within a city or a country. There

are also many vertically oriented communities that gather users around a specific topic and have many dedicated members on all continents.

Facebook is ranked among the most visited sites in the world, with over than 1.3 billion subscribed users to date. Moreover, Friendster is popular in Asia, Orkut in Brazil and Vkon-takte in Russia. On top of that, there are dozens of other social networks with vibrant communities, such as Vznet, Xing, Badoo, Netlog, Tuenti, Barrabes, Hyves, Nasza Klasa, LunarStorm, Zoo, Sapo, Daily-Motion and so on. There are also many vertically oriented communities which gather users around a specific topic, such as music, books, etc. LinkedIn with over 330 million users or Viadeo with 50 million users and Xing with 13.5 million users are mostly oriented in establishing professional connections between their users and initiate potential business collaborations.

The rapid growth in popularity of social networks has enabled large numbers of users to communicate, create and share content, give and receive recommendations, and, at the same time, it opened new challenging problems. The unbounded growth of content and users pushes the Internet technologies to its limits and demands for new solutions. Such challenges are present in all SNSs to a greater or lesser extent. Considerable amount of effort has already been devoted worldwide for problems such as content management in large scale collections, context awareness, multimedia search and retrieval, social graph modelling analysis and mining, etc.

1.1.1 The Twitter Social Network

Twitter is an online social networking service that enables users to send and read short messages of up to 140 characters called "tweets". Registered users can read and post tweets, but unregistered users can only read them. Users access Twitter through the website interface, SMS, or through a mobile device application. Twitter is one of the most popular social networks and micro-blogging service in the world, and according to its website it has more than 640 million users connected by 24 billion links. In Twitter, "following" someone means that a user will have in his personal timeline other people's tweets (Twitter updates). "Followers" are people who receive other people's Twitter updates. Approximately 99.89% of the Twitter accounts have less than $3,500$ followers and followings. There are approximately 40 million accounts with less than 10 followers and followings, that is between 6% to 7% of all Twitter accounts. It is a social trend to ask followed accounts to follow back in Twitter. There is a limit at $2,000$ followings that starts growing after $1,800$ followers, which is the number of followings set by Twitter to prevent users monitoring too many accounts whereas they have no active role in Twitter. Approximately 40% of accounts have no followers and 25% have no followings. Twitter is interesting to be studied because it allows the information spread between people, groups and advertisers, and since the relation between its users is unidirectional, the information propagation within the network is similar to the way that the information propagates in real life.

1.2 Research and Technical Challenges

This section lists the main research challenges in social networks, which are currently being investigated by the research community.

- The analysis of relations and communications between members of a community can reveal the most influential users from a social point of view.

- As social networks will continue to evolve, the discovery of communities, users' interests [1], and the construction of specific social graphs from large scale social networks will continue to be a dynamic research challenge [2]. Research in dynamics and trends in social networks may provide valuable tools for information extraction that may be used for epidemic predictions or recommender systems [3, 4, 5].

- The information extracted from social networks proved to be a useful tool towards security. One example of an application related to security is the terrorism analysis, *e.g.* the analysis of the 9-11 terrorist network [6]. This study was done by gathering public information from major newspapers on the Internet and analyzed it by means of social networks [7]. Therefore, cyber surveillance for critical infrastructure protection is another major research challenge on social network analysis.

- Searching in blogs, tweets and other social media is still an open issue since posts are very small in size but numerous, with little contextual information. Moreover, different users have different needs when it comes to the consumption of social media. Real time search has to balance between quality, authority, relevance and timeliness of the content [8].

- Crowdsourcing systems gave promising solutions to problems that were unsolved for years. The research community nowadays is working by leveraging human intelligence to solve critical problems [9, 10], since social networks contain immense knowledge through their users. However, it is not trivial to extract that knowledge [11].

- Traffic prediction based on media consumption may be correlated between groups of users. This information can be used to dimension media servers and network resources to avoid congestion and improve the quality of experience and service.

  Content sharing and distribution needs will continue to increase. Mobile phones, digital cameras and other pervasive devices produce huge amounts of data which users want to distribute if possible in real time [12].

- Since users population and data production increase, spam and advertisements will continue growing [13]. In addition, the importance of social networks to influence the opinions of the users has to be protected with a mechanism that promotes trustworthy opinions that are relevant to businesses.

- As in every human community, online social communities face also critical social and ethical issues that need special care and delicate handling. Protection of personal information and many other problems need special attention [14].

  In order to address these challenges, we need to extract the relevant information from online social media in real time.

1.3 Problem Statement and Objectives

Topic detection and trend sensing is the problem of automatically detecting topics from large text documents and extracting the main opinion without any human intervention. This problem is of great practical importance given the massive volume of documents available online in news feeds, electronic mail,

digital libraries, and social networks.

Text classification is the task of assigning predefined labels or categories to texts or documents. It can provide conceptual views of document collections and has important applications in real world problems. Nowadays, the documents which can be found online are typically organized by categories according to their subject, *e.g.* topics. Some widespread applications of topic detection and text classification is community detection, traffic prediction, dimensioning media consumption, privacy, and spam filtering, as mentioned in Section 1.2.

By performing topic detection on social network communities, we can regroup users in teams and find the most influential ones, which can be used to build specific and strategic plans. Public information in social networks can be extracted by topic detection and classification and used for cyber surveillance in an automatic way in order to avoid the overload. Extracting an opinion from social networks is difficult, because users are writing in a way which does not have correct syntax or grammar and contains many abbreviations. Therefore, mining opinions in social networks can benefit by on automatic topic detection on really short and tremendous posts. By grouping users and adding labels to discussions or communities we are able to find their interests and tag people that share information very often. This information can be used to dimension media servers and network resources to avoid congestion and improve the quality of experience and service. Finally, by performing topic classification we can find similarities between posts of users that spread irrelevant information into the network and enable a spam filter to defend against that.

In this thesis we present a novel method to perform topic detection, clas-

sification and trend sensing in short texts. The importance of the proposed method comes from the fact that up to now, the main methods used for text classification are based on keywords detection and machine learning techniques. By using keywords or bag-of-words in tweets will often fail because of the wrongly or distorted usage of the words – which also needs lists of keywords for every language to be built – or because of implicit references to previous texts or messages. In general, machine learning techniques are 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 thesis we are trying to address the discussed challenges and problems of other state-of-the-art methods and propose a method which is not based on keywords, language, grammar or dictionaries, in order to perform topic detection, classification and trend sensing.

Instead of relying on words as most other existing methods which use bag-of-words or n-gram techniques, we introduce Joint Complexity (JC), which is defined as the cardinality of a set of all distinct common factors, subsequences of characters, of two given strings. Each short sequence of text is decomposed in linear time into a memory efficient structure called Suffix Tree and by overlapping two trees, in linear or sublinear average time, we obtain the JC defined as the cardinality of factors that are common in both trees. The method

has been extensively tested for text generation by Markov sources of finite order for a finite alphabet. The Markovian generation of text gives a good approximation for natural text generation and is a good candidate for language discrimination. One key take-away from this approach is that JC is language-agnostic since we can detect similarities between two texts without being based on grammar and vocabulary. Therefore there is no need to build any specific dictionary or stemming process. JC can also be used to capture a change in topic within a conversation, as well as a change in the style of a specific writer of a text.

On the other hand, the inherent sparsity of the data space motivated us in a natural fashion the use of the recently introduced theory of *Compressive Sensing* (CS) [15, 16] driven by the problem of target localization [17]. More specifically, the problem of estimating the unknown class of a message is reduced to a problem of recovering a sparse position-indicator vector, with all of its components being zero except for the component corresponding to the unknown class where the message is placed. CS states that signals which are sparse or compressible in a suitable transform basis can be recovered from a highly reduced number of incoherent random projections, in contrast to the traditional methods dominated by the well-established Nyquist-Shannon sampling theory. The method works in conjunction with a Kalman filter to update the states of a dynamical system as a refinement step.

In this thesis we exploit datasets collected by using the Twitter streaming API, getting tweets in various languages and we obtain very promising results when comparing to state-of-the-art methods.

1.4 Scope and Plan of the Thesis

In this thesis, a novel method for topic detection, classification and trend sensing in Dynamic Social Networks is proposed and implemented. Such system is able to address the research and technical challenges mentioned in Section 1.2. The structure of this thesis is organized as follows.

First, Chapter 2 overviews the state-of-the-art of topic detection, classification and trend sensing techniques for online social networks. First, it describes the *document–pivot* and *feature–pivot* methods, along with a brief overview of the pre-processing stage of these techniques. Six state-of-the-art methods: LDA, Doc-p, GFeat-p, FPM, SFPM, BNgram are described in details, as they serve as the performance benchmarks to the proposed system.

In Chapter 3, we introduce the Joint Sequence Complexity method. This chapter describes the mathematical concept of the complexity of a sequence, which is defined as the number of distinct subsequences of the given sequence. The analysis of a sequence in subcomponents is done by suffix trees, which is a simple, fast, and low complexity method to store and recall subsequences from the memory. We define and use Joint Complexity for evaluating the similarity between sequences generated by different sources. Markov models well describe the generation of natural text, and their performance can be predicted via linear algebra, combinatorics, and asymptotic analysis. We exploit Markov sources trained on different natural language datasets, for short and long sequences, and perform automated online sequence analysis on information streams in Twitter.

Then, Chapter 4 introduces the Compressive Sensing based classification method. Driven by the methodology of indoor localization, the algorithm converts the classification problem into a signal recovery problem, so that CS theory can be applied. First we employ Joint Complexity to perform topic detection and build signal vectors. Then we apply the theory of CS to perform topic classification by recovering an indicator vector, while reducing significantly the amount of information from tweets. Kalman filter is introduced as a refinement step for the update of the process, and perform users and topics tracking.

Chapter 5 presents the extension of Joint Complexity and Compressive Sensing on two additional research subjects that have been studied during this thesis, (a) localization and path tracking in indoor environments, and (b) cryptography based on the Eulerian circuits of original texts. It has to be noted that Section 5.2 describes the use of Compressive Sensing, which was first exploited in the problem of indoor localization and path tracking, and motivated us to use the theory for text classification.

Finally, Chapter 6 presents the concluding remarks and gives directions for future work and perspectives.

Chapter 2

Background and Related Work

Contents

2.1 Introduction

Topic detection and tracking aims at extracting topics from a stream of textual information sources, or documents, and to quantify their "trend" in real time [18]. These techniques apply on pieces of texts, *i.e.* posts, produced within social media platforms. Topic detection can produce two types of complementary outputs: cluster output or term output are selected and then clustered. In the first method, referred to as *document–pivot*, a topic is represented by a cluster of documents, whereas in the latter, commonly referred to as *feature–pivot*, a cluster of terms is produced instead. In the following, we review several popular approaches that fall in either of the two categories. Six state-of-the-art methods: Latent Dirichlet Allocation (LDA), Document–Pivot Topic Detection (Doc-p), Graph–Based Feature–Pivot Topic Detection (GFeat-p), Frequent Pattern Mining (FPM), Soft Frequent Pattern Mining (SFPM), BNgram are described in details, as they serve as the performance benchmarks to the proposed system.

2.2 Document–pivot methods

Simple *document–pivot* approaches cluster documents by leveraging some similarity metric between them. A recent work [19] follows this direction to provide a method for "breaking news" detection in Twitter. Tweets retrieved using targeted queries or hashtags are converted into a *bag-of-words* representation weighted with boosted *tf-idf* (term frequency–inverse document frequency) emphasizing important entities such as names of countries or public figures. *Bag-of-words* of a text is its representation as the set of its words, disregarding grammar and even word order but keeping multiplicity. Tweets are then incrementally merged in clusters by considering the textual similarity between

incoming tweets and existing clusters. Similar approaches based on textual similarity and *tf-idf* can be found in literature [20, 21]. Among them, the method discussed in [21] classifies tweets as referring to real–world events or not. The classifier is trained on a vast variety of features including social aspects (e.g., number of mentions) and other Twitter–specific features. An important drawback of the method is the need for manual annotation of training and test samples.

Dimensions other than text can also be used to improve the quality of clustering. TwitterStand [22] uses a "leader–follower" clustering algorithm that takes into account both textual similarity and temporal proximity. Each cluster center is represented using a centroid *tf-idf* vector and the average post–time of the tweet in the cluster. A similarity metric based on both elements and on the number of shared hashtags allows incremental merging of new tweets with existing clusters. The main disadvantages of this method is the sensitivity to noise (which is a known problem for *document–pivot* methods [23]) and fragmentation of clusters. It needs a manual selection of trusted information providers and periodic defragmentation runs to mitigate such effects. The goal in a large corpus is to detect the first document discussing a given topic like in [24]. A new story is created by a document having low similarity with all previously detected clusters. Locality sensitive hashing is used for fast retrieval of nearest neighbors for the incoming document.

In conclusion, document-pivot methods have two main problems, which are: (a) segmentation of classification groups, and (b) depend on arbitrary thresholds for the classification of an incoming tweet.

2.3 Feature–pivot methods

Feature–pivot methods are closely related to topic models in natural language processing, namely statistical models used to build sets of terms which are representative of the topics occurring in a corpus of documents. Most of the state-of-the-art static topic models are based on the Latent Dirichlet allocation (LDA) [25], which is described in Section 2.4.3. Even though LDA extensions for dynamic data have been proposed [26], alternative approaches trying to capture topics through the detection of term burstiness have been studied [27], mainly in the context of news media mining. The idea behind those methods is that "breaking news", unlike other discussion topics, happen to reach a fast peak of attention from routine users as soon as they are tweeted or posted [28, 29]. Accordingly, the common framework which underlies most of the approaches in this category first identifies bursty terms and then clusters them together to produce topic definitions.

The diffusion of the services over social media and detection of bursty events had been studied in generic document sets. The method presented in [23], for instance, detects bursty terms by looking their frequency in a given time window. Once the bursty terms are found, they are clustered using a probabilistic model of cooccurrence. The need for such a global topic term distribution restricts this approach to a batch mode of computation. Similar methods were tested for topic detection in social media, such as in the Twitter, but with additional emphasis on the enrichment of the obtained topics with non–bursty but relevant terms, URLs and locations [30].

Graph–based approaches detect term clusters based on their pairwise similarities. The algorithm proposed in [31] builds a term cooccurrence graph, whose nodes are clustered using a community detection algorithm based on betweenness centrality, which is an indicator of a node's centrality in a network and is equal to the number of shortest paths from all vertices to all others that pass through that node. Additionally, the topic description is enriched with the documents which are most relevant to the identified terms. Graphs of short phrases, rather than of single terms, connected by edges representing similarity have also been used [32]. Graph–based approaches have also been used in the context of collaborative tagging systems with the goal of discovering groups of tags pertaining to topics of social interest [33].

Signal processing approaches have also been explored in [34], which compute *df-idf* (a variant of *tf-idf*) for each term in each considered time slot, and then apply wavelet analysis on consecutive blocks. The difference between the normalised entropy of consecutive blocks is used to construct the final signal. Relevant terms which are bursty are extracted by computing the autocorrelation of the signal and heuristically learning and determining a threshold to detect new bursty terms. Also in this case, a graph between selected terms is built based on their cross–correlation and it is then clustered to obtain event definitions. The Discrete Fourier Transform is used in [35], where the signal for each term is classified according to its power and periodicity. Depending on the identified class, the distribution of appearance of a term in time is modeled using Gaussian distributions. The Kullback–Leibler divergence (as a relative entropy) between the distributions is then used to determine clusters and increase the computational complexity of the method.

The knowledge of the community leads to even more sophisticated approaches. In a recent work [36] a PageRank–like measure is used to identify important users on the Twitter social network. Such centrality score is combined with a measure of term frequency to obtain a measure for each term. Then, clustering on a correlation graph of bursty terms delineates topic boundaries.

These methods are based on the analysis of similarities between terms and often give wrong correlation of topics, with their main disadvantage being the use of dictionaries and stemming processes.

2.4 Related Work

In general the topic detection methods use preprocessing techniques. Next, we define all the components of the topic detection process. In Section 2.4.1, we present the problem statement and define some basic terminology. Then in Section 2.4.2, we describe the data preprocessing and in the following sections we present six methods that take in as input the preprocessed data and output the detected topics.

2.4.1 Problem Definition

We address the task of detecting topics in real–time from social media streams. To keep our approach general, we consider that the stream is made of short pieces of text generated by social media users, *e.g.* posts, messages, or tweets in the specific case of Twitter. Posts are formed by a sequence of words or terms, and each one is marked with the timestamp of creation. A plethora of methods

have a user–centered scenario in which the user starts up the detection system by providing a set of seed terms that are used as initial filter to narrow down the analysis only to the posts containing at least one of the seed terms. Additionally, there exists an assumption that the time frame of interest (can be indefinitely long) and a desired update rate are provided (*e.g.* detect new trending topics every 15 minutes). The expected output of the algorithm is a topic, defined as a headline and a list of terms, delivered at the end of each time slot determined by the update rate. This setup fits well many real–world scenarios in which an expert of some domain has to monitor specific topics or events being discussed in social media [3, 37]. For instance, this is the case for computational journalism in which the media inquirer is supposed to have enough knowledge of the domain of interest to provide initial terms to perform an initial filtering of the data stream. Even if it requires an initial human input, this framework still remains generic and suitable to any type of topic or event.

2.4.2 Data Preprocessing

The content of user generated messages could be unpredictably noisy. In many works, in order to reduce the amount of noise before the proper topic detection is executed, the raw data extracted through the seed terms filter is subjected to three preprocessing steps.

- Tokenization: In a raw post, terms can be combined with any sort of punctuation and hyphenation and can contain abbreviations, typos, or conventional word variations. The Twokenizer tool [20] is used to extract bags of cleaner terms from the original messages by removing stopwords and punctuation, compressing redundant character repetitions, and removing

mentions, *i.e.*, IDs or names of other users included in the text for messaging purposes.

- Stemming: In information retrieval, stemming is the process of reducing inflected words to their root (or stem), so that related words map to the same stem. This process naturally reduces the number of words associated to each document, thus simplifying the feature space. Most techniques use an implementation of the Porter stemming algorithm [38].

- Aggregation: Topic detection methods based on word or n–grams cooccurrences, or any other type of statistical inference, suffer in the absence of long documents. This is the case of social media, where user–generated content is typically in the form of short posts. In information retrieval it is a common practice to partially address this problem by concatenating different messages together to produce documents of larger size. Large documents construction is based on two strategies. The first strategy involves temporal aggregation that concatenates together N messages, whose generation date is contiguous. The second strategy involves a similarity–based aggregation which attaches to a message all the near–duplicate messages posted in the same time slot, identified through an efficient document clustering method [24], which is also also used by one of the examined topic detection algorithms presented in Section 2.4.4.

Determining the effect of such preprocessing algorithms on the quality of the final topic detection is difficult to assess, and it has not been much investigated so far. For instance, the aggregation of posts in super–documents could on the one hand help to improve the word cooccurrence statistic but on the other hand introduces the risk of grouping terms related to different topics, and to reveal

false cooccurrence.

2.4.3 Latent Dirichlet Allocation

Topic extraction in textual *corpus* can be addressed through probabilistic topic models. In general, a topic model is a Bayesian model which associates with each document a probability distribution over the topics, where each topic is in turn a probability distribution. The Latent Dirichlet Allocation (LDA) [25] is the best known and most widely used topic model. According to LDA, every document is considered as a bag of terms, which are the only observed variables in the model. The topic distribution per document and the term distribution per topic are instead hidden variable and have to be estimated through Bayesian inference. The Collapsed Variational Bayesian inference algorithm [39], which is an LDA variant, is computationally efficient, more accurate than standard variational Bayesian inference for LDA, and has given rise to many independent implementations already available in the literature. LDA requires the expected number of topics k as a input. The estimation of the optimal k, although possible through the use of non-parametric methods [40], falls beyond the scope of this thesis.

2.4.4 Document–Pivot Topic Detection

The second method discussed here, is an instance of a classical Topic Detection and Tracking method which uses a document–pivot approach (Doc-p). It works as follows:

- First, the method performs online clustering of posts. It computes the cosine similarity of the *tf-idf* [41] representation of an incoming post with

all other posts processed so far. If the best cosine similarity is above some threshold θ_{tf-idf}, it assigns the item to the same cluster as its best match; otherwise it creates a new cluster with the new post as its only item. The best matching tweet is efficiently retrieved by Locality Sensitive Hashing (LSH).

- Then, it filters out clusters with item count smaller than θ_n.
- For each cluster c, it computes a score as follows:

$$score(c) = \sum_{doc \in C} \sum_{word \in doc} \exp(-p(word))$$

The probability of appearance of a single term $p(word)$ is estimated from a reference dataset that has been collected from Twitter, mentioned in Section 2.4.5. Thus, less frequent terms contribute more to the score of the cluster.

- Clusters are sorted according to their score and the top clusters are returned. LSH can rapidly provide the nearest neighbours with respect to cosine similarity in a large collection of documents. An alternative to LSH is to use inverted indices on the terms which appear in the tweets and then compute the cosine similarity between the incoming document and the set of documents that have a significant term overlap with it; however, the use of LSH is much more efficient as it can provide the nearest neighbours with respect to cosine similarity directly.

In practice, when posts are as short as tweets, the similarity of two items is usually either close to zero or close to one (between 0.8 to 1.0). This observation makes setting θ_{tf-idf} relatively easy to 0.5. Due to this phenomenon, items

grouped together by this procedure are usually, but not always, near–duplicates (*e.g.* ReTweets). Therefore, it is clear that topics produced by this method will be fragmented, *i.e.* the same topic may be represented by different sets of near duplicate tweets. To begin dealing with this issue, we present methods for aggregating as described in Section 2.4.2.

2.4.5 Graph–Based Feature–Pivot Topic Detection

The Graph–Based Feature–Pivot Topic Detection method (GFeat-p) is a first of a series of *feature–pivot* methods, where the clustering step is performed via the Structural Clustering Algorithm for Networks (SCAN) [42], which is in general applied to network analysis. Apart from detecting communities of nodes, SCAN provides a list of hubs (vertices that bridge many clusters), each of which may be connected to a set of communities. For SCAN applied to topic detection, the nodes in the network correspond to terms and the communities correspond to topics. The detected hubs are terms which are related to more than one topic, and effectively provides an explicit link between topics. That would not be possible to achieve with a common partition clustering algorithm. The terms to be clustered, is a subset of terms present in the *corpus*, applied to independent reference corpus selected with randomly collected tweets [20]. For each of the terms in the reference corpus, the likelihood of appearance $p(w|corpus)$ is estimated as follows:

$$p(w|corpus) = \frac{N_w(corpus) + \delta}{N(corpus)}$$

where $N_w(corpus)$ is the number of appearances of term w in the corpus, N(corpus) is the number of terms in corpus (including repetition), and δ is a

constant (typically set to 0.5) that is included to regularize the probability estimate (*i.e.* to ensure that a new term that does not appear in the corpus is not assigned a probability of 0). The most important terms in the new corpus, are determined by computing the ratio of the likelihoods of appearance in the two corpora for each term, as follows:

$$\frac{p(w|corpus_{new})}{p(w|corpus_{ref})}$$

The terms with the highest ratio are expected to be related to the most actively discussed topics in the *corpus*. Once the high ratio terms are identified, a term graph is constructed and the SCAN graph–based clustering algorithm is applied to extract groups of terms, each of which is considered to be a distinct topic. More specifically, the algorithm steps are the following:

- Selection: The top K terms are selected using the ratio of likelihoods measure and will be used as the nodes for a graph G.

- Linking: The nodes of G are connected using a term linking strategy. According to a preselected similarity measure for pairs of terms all pairwise similarities are computed.

  Moreover, each term is linked with its k nearest neighbours (kNN approach) or all pairs of nodes that have similarity higher that ε (ε-based approach) are linked.

- Clustering: The SCAN algorithm is applied to the graph; a topic is generated for each of the detected communities. A hub may be linked to more than one topic or community.

The term linking step is crucial for the success of the method. Unfortunately,

there is no straightforward method for determining a priori the best similarity measure or node linking strategy to be used. Additionally, it can be expected that the graph construction tuning parameters will need to vary for datasets with different topic granularities and levels of internal topic connectivity.

2.4.6 Frequent Pattern Mining

As it was mentioned in the previous section, a problem with the *feature–pivot* methods is that terms clustering relies on pairwise similarities which are based on cooccurrences number only. In the case of closely interconnected topics which share a relatively large number of terms, this procedure most likely produces general topics. An option to deal with this issue is to take into account the simultaneous cooccurrence between more than two terms. This motivation leads naturally to consider the use of frequent set of items, a well–defined technique in transaction mining for topic detection to determine which items are likely to cooccur in a set of transactions [43].

In the social media context, an item is any term w mentioned in a post (excluding stop words, punctuation tokens, etc.). The transaction is the post, and the transaction set are all posts that occur in a time slot T. The number of times that any given set of terms occurs in the time slot is defined as its support, and any set of items that meets a minimum support is called a pattern. The initial challenge is to apply highly-scalable Frequent Pattern (FP) detection to each time slot to identify the most frequent terms or patterns. These terms are used to characterize the topics that best illustrate the underlying social interactions. Below, we describe these two processing steps, FP detection and ranking.

1. FP detection: First, the algorithm constructs a term list according to their frequency [44, 45]. Then, for each time slot an FP-Tree sorts the patterns

according to their cooccurrences and their support. Finally, the FP-tree structures are aggregated and analysed to produce association rules on the transaction set.

2. FP ranking: Once a set of frequent patterns has been extracted from the dataset, they are ranked and the top N results are returned as candidate topics. The challenge is to rank patterns such that terms in the candidate topics are sufficiently related and with enough diversity to cover the different underlying subjects of conversation in the social interactions. A common way to rank patterns is to simply use the support of a given pattern; the more often a set of terms cooccurs, the more likely it can be considered relevant as a topic.

2.4.7 Soft Frequent Pattern Mining

In Section 2.4.6 a frequent pattern mining approach for topic detection was described. It provided an elegant solution to the problem of *feature–pivot* methods that takes into account only pairwise cooccurrences between terms in the case of corpus with densely interconnected topics. Section 2.4.5 examined only pairwise cooccurrences, where frequent pattern mining examines cooccurrences between any number of terms, typically larger than two. A question that naturally arises is whether it is possible to formulate a method that lies between these two extremes. Such a method would examine cooccurrence patterns between sets of terms with cardinality larger that two, like frequent pattern mining does, but it would be less strict by not requiring that all terms in these sets cooccur frequently. Instead, in order to ensure topic cohesiveness, it would require that large subsets of the terms grouped together, but not necessar-

ily all, cooccur frequently, resulting in a "soft" version of frequent pattern mining.

The proposed approach (SFPM) works by maintaining a set of terms S, on which new terms are added in a greedy manner, according to how often they cooccur with the terms in S. In order to quantify the cooccurrence match between a set S and a candidate term t, a vector D_S for S and a vector D_t for the term t are maintained, both with dimension n, where n is the number of documents in the collection. The i-th element of D_S denotes how many of the terms in S cooccur in the i-th document, whereas the i-th element of D_t is a binary indicator that represents if the term t occurs in the i-th document or not. Thus, the vector D_t for a term t that frequently cooccurs with the terms in set S, will have a high cosine similarity to the corresponding vector D_S. Note that some of the elements of D_S may have the value $|S|$, meaning that all items in S occur in the corresponding documents, whereas other may have a smaller value indicating that only a subset of the terms in S cooccur in the corresponding documents. For a term that is examined for expansion of S, it is clear that there will be some contribution to the similarity score also from the documents in which not all terms cooccur, albeit somewhat smaller compared to that documents in which all terms cooccur. The "soft" matching between a term that is considered for expansion and a set S is achieved. Finding the best matching term can be done either using exhaustive search or some approximate nearest neighbour scheme such as LSH. As mentioned, a greedy approach that expands the set S with the best matching term is used, thus a criterion is needed to terminate the expansion process. The termination criterion clearly has to deal with the cohesiveness of the generated topics, meaning that if not properly set, the resulting topics may either end up having too few terms or

really being a mixture of topics (many terms related to possibly irrelevant topics). To deal with this, the cosine similarity threshold $\theta(S)$ between S and the next best matching term is used. If the similarity is above the threshold, the term is added, otherwise the expansion process stops. This threshold is the only parameter of the proposed algorithm and is set to be a function of the cardinality of S. In particular a sigmoid function of the following form is used:

$$\theta(S) = 1 - \frac{1}{1 + \exp((|S| - b)/c)}$$

The parameters b and c can be used to control the size of the term clusters and how soft the cooccurrence constraints will be. Practically, the values of b and c are set so that the addition of terms when the cardinality of S is small is easier (the threshold is low), but addition of terms when the cardinality is larger is harder. A low threshold for the small values of $|S|$ is required so that it is possible for terms that are associated to different topics and therefore occur in more documents rather than to ones corresponding to the non–zero elements of D_S to join the set S. The high threshold for the larger values of $|S|$ is required so that S does not grow without limit. Since a set of topics is required – rather than a single topic – the greedy search procedure is applied as many times as the number of considered terms, each time initializing S with a candidate term. This will produce as many topics as the set of terms considered, many of which will be duplicates, thus a post-process of the results is needed to remove these duplicates. To limit the search procedure in reasonable limits the top n terms with the highest likelihood-ratio are selected by 2.4.5.

When the "soft" frequent pattern matching algorithm runs for some time,

the vector D_S may include too many non–zero entries filled with small values, especially if some very frequently occurring term has been added to the set. This may have the effect that a term may be deemed relevant to S because it cooccurs frequently only with a very small number of terms in the set rather than with most of them. In order to deal with this issue, after each expansion step, any entries of D_S that have a value smaller than $|S|/2$ are reset to zero. The most relevant documents for a topic can be directly read from its vector D_S: the ones with the highest document counts.

2.4.8 BNgram

Both the frequent itemset mining and soft frequent itemset mining approaches attempted to take into account the simultaneous cooccurences between more than two terms. However, it is also possible to achieve a similar result in a simpler way by using n–grams. This naturally groups together terms that cooccur and it may be considered to offer a first level of term grouping. Using n–grams makes particularly sense for Twitter, since a large number of the status updates in Twitter are just copies or retweets of previous messages, so important n–grams will tend to become frequent.

Additionally, a new feature selection method is introduced. The changing frequency of terms over time as a useful source of information to detect emerging topics is taken into account. The main goal of this approach is to find emerging topics in post streams by comparing the term frequencies from the current time slot with those of preceding time slots. The $df - idf_t$ metric which introduces time to the classic *tf–idf* score is proposed. Historical data to penalize those topics which began in the past and are still popular in the present, and which therefore do not define new topics have been used.

The term indices, implemented using Lucene, are organized into different time slots. In addition to single terms, the index also considers bigrams and trigrams. Once the index is created, the $df - idf_t$ score is computed for each n–gram of the current time slot i based on its document frequency for this time slot and penalized by the logarithm of the average of its document frequencies in the previous t time slots:

$$score_{df-idf_t} = \frac{df_i + 1}{\log(\frac{\sum_{j=1}^{t} df_{i-j}}{t} + 1) + 1}$$

In addition, a boost factor is considered to raise the importance of proper nouns (persons, locations and organizations, in our case) using a standard named entity recognizer [46], as they are essential terms in most discussed stories. The use of this factor is similar to [19], where the authors highlight the importance of such words for grouping results. The selected values for this factor are based on the best values from the experiments of the previous work, being boost=1.5 in case the n-gram contains a named entity and boost=1 otherwise. As a result of this process, a ranking of n–grams is created sorted by their df-idft scores. A single n–gram is often not very informative, but a group of them often offers interesting details of a story. Therefore, a clustering algorithm to group the most representative n–grams into clusters, each representing a single topic is used. The clustering is based on distances between n–grams or clusters of n–grams. From the set of distances, those not exceeding a distance threshold are assumed to represent the same topic. The similarity between two n–grams is defined as the fraction of posts that contain both of them. Every n–gram is initially assigned to its own singleton cluster, then following a standard "group average" hierarchical clustering algorithm [47] to iteratively find and merge

the closest pair of clusters. When an n–gram cluster is joined to another, the similarities of the new cluster to the other clusters are computed as the average of the similarities of the combined clusters. The clustering is repeated until the similarity between the nearest un-merged clusters falls below a fixed threshold θ, producing the final set of topic clusters for the corresponding time slot.

The main contribution of this approach is the use of the temporal dimension of data to detect emerging stories. There are other similar approaches on term weighting considering the temporal dimension of data but most of them suffer from several shortcomings. For instance, in [27] two methods of finding "peaky topics" are presented. They find the peak terms for a time-slot compared to the rest of the corpus, whereas each slot is compared to the immediately previous time slots. If some topic is discussed at several different times, their approach could miss this since the defining words would be highly frequent in the whole corpus. In addition, their approach only uses single words which often seem to be too limited to identify stories. Finally, their use of the whole corpus is less suitable for real-time analysis.

2.5 Chapter Summary

This chapter overviewed the state-of-the-art of topic detection, classification and trend sensing techniques for online social networks. First, it described the *document–pivot* and *feature–pivot* methods, along with a brief overview of the pre-processing stage of these techniques. Six state-of-the-art methods: LDA, Doc-p, GFeat-p, FPM, SFPM, BNgram were described in details, as they serve as the performance benchmarks to the proposed system.

Chapter 3

Joint Sequence Complexity: Introduction and Theory

Contents

3.1 Introduction

In this thesis we study joint sequence complexity and we introduce its applications for topic detection and text classification, in particular source discrimination. The mathematical concept of the complexity of a sequence is defined as the number of distinct factors of it. The Joint Complexity is thus the number of distinct common factors of two sequences. Sequences containing many common parts have a higher Joint Complexity. The extraction of the factors of a sequence is done by suffix trees, which is a simple and fast (low complexity) method to

store and retrieve them from the memory. Joint Complexity is used for evaluating the similarity between sequences generated by different sources and we will predict its performance over Markov sources. Markov models describe well the generation of natural text, and their performance can be predicted via linear algebra, combinatorics, and asymptotic analysis. This analysis follows in this chapter. We exploit datasets from different natural languages, for both short and long sequences, with promising results on complexity and accuracy. We performed automated online sequence analysis on information streams, such as Twitter.

3.2 Sequence Complexity

In the last decades, several attempts have been made to capture mathematically the concept of the "complexity" of a sequence. In [56], the sequence complexity was defined as the number of different factors contained in a sequence. If X is a sequence and $I(X)$ its set of factors (distinct substrings), then the cardinality $|I(X)|$ is the complexity of the sequence. For example, if $X = aabaa$ then $I(X) = \{v, a, b, aa, ab, ba, aab, aba, baa, aaba, abaa, aabaa\}$, and $|I(X)| = 12$, where v denotes the empty string. Sometimes the complexity of the sequence is called the I–Complexity (IC) [50]. The notion is connected with quite deep mathematical properties, including the rather elusive concept of randomness in a string [51, 52, 53].

3.3 Joint Complexity

In general, the information contained in a string cannot be measured in absolute and a reference string is required. The concept of Joint Complexity (JC) has been introduced in [54], as a metric of similarity between two strings. JC method is the number of different factors, which are common in two sequences. In other words the JC of sequence X and Y is equal to $J(X,Y) = |I(X) \cap I(Y)|$. We denote $J_{n,m}$ the average value of $J(X,Y)$ when X and Y have length n and m respectively. In this work we study its growth when the lengths are the same, $n = m$.

JC method is an efficient way of evaluating the degree of similarity between two sequences. For example, the genome sequences of two dogs will contain more common factors than the genome sequences of a dog and a cat. Similarly, two texts written in the same language have more common factors than texts written in very different languages. JC method is also greater when languages have similarities (*e.g.*, French and Italian), than when they differ significantly (*e.g.*, French and English). Furthermore, texts written in the same language but on different topics (*e.g.*, law and cooking) have smaller JC than texts on the same topic (*e.g.*, medicine). Therefore, JC method is a pertinent tool for automated monitoring of social networks. This requires a precise analysis of the JC method, discussed in this work as well as in [64], together with some experimental results, confirming usefulness of the joint string complexity for short text discrimination.

In [54] it is proved that the JC of two texts of length n built from two different binary memoryless sources grows like $\gamma \frac{n^\kappa}{\sqrt{\alpha \log n}}$, for some $\kappa < 1$ and $\gamma, \alpha > 0$ which depend on the parameters of the two sources. When the sources are identical, *i.e.*, when their parameters are identical, but the text still being

independently generated, then the JC growth is $O(n)$, hence $\kappa = 1$. When the texts are identical (*i.e.*, $X = Y$), then the JC is identical to the I–Complexity and it grows as $\frac{n^2}{2}$ [56]. Therefore JC method can already be used to detect "copy–paste" portion between the texts. Indeed the presence of a common factor of length $O(n)$ would inflate the JC by a term $O(n^2)$.

We should point out that experiments demonstrate that for memoryless sources the JC estimate $\gamma \frac{n^k}{\sqrt{\alpha \log n}}$ converges very slowly. Therefore, JC is not really meaningful even when $n \approx 10^9$. In this work we derive second order asymptotics for JC of the following form $\gamma \frac{n^k}{\sqrt{\alpha \log n + \beta}}$ for some $\beta > 0$. Indeed it turns out that for text where $n < 100$ and $\log n < \beta$, this new estimate converges more quickly than the estimate $\gamma \frac{n^k}{\sqrt{\alpha \log n}}$, thus it can be used for short texts, like tweets. In fact, our analysis indicates that JC can be refined via a factor for $P(\frac{1}{\alpha \log n + \beta})$ appearing in the JC, where P is a specific polynomial determined via saddle point expansion. This additional term further improves the convergence for small values of n, and also same periodic factors of small amplitude appear when the source parameters satisfy some specific and very unfrequent conditions.

In this work we extend the JC estimate to Markov sources of any order on a finite alphabet. Although Markov models are no more realistic than memoryless sources, say, for a DNA sequence, they seem to be fairly realistic for text generation [64]. An example of Markov simulated text for different order is shown in Table 3.1 from "The Picture of Dorian Gray".

In view of these facts, we can use the JC to discriminate between two identical/non-identical Markov sources [55]. We introduce the discriminant function as follows $d(X, Y) = 1 - \frac{1}{\log n} \log J(X, Y)$ for two sequences X and Y of length n. This discriminant allows us to determine whether or not X and Y are gener-

Table 3.1: Markov simulated text from the book "The Picture of Dorian Gray"

Markov order	Text simulated by the given Markov order
3	"Oh, I do yourse trought lips whose-red from to his, now far taked. If Dorian, Had kept has it, realize of him. Ther chariten suddenial tering us. I don't belige had keption the want you are ters. I am in the when mights horry for own that words
4	"Oh, I want your lives. It is that is words the right it his find it man at they see merely fresh impulses. But when you have you, Mr. Gray, a fresh impulse of mine. His sad stifling round of a regret. She is quite devoted forgot an arrowed."
5	"Oh, I am so sorry. When I am in Lady Agatha's black book that the air of God, which had never open you must go. I am painting, and poisons us. We are such a fresh impulse of joy that he has done with a funny looked at himself unspotted"

ated by the same Markov source by verifying whether $d(X,Y) = O(1/\log n) \to 0$ or $d(X,Y) = 1 - \kappa + O(\log\log n/\log n) > 0$, respectively when the length of X and Y are both equal to n. In this work we concentrate mainly on the analysis of the JC method, however, we also present some experimental evidence of how useful our discriminant is for real texts.

In Figure 3.1, we compared the JC of a real English text with simulated texts of the same length written in French, Greek, Polish, and Finnish (all language is transcribed in the Latin alphabet, simulated from a Markov source of order 3). It is easy to see that even for texts smaller in length than a thousand words, one can discriminate between these languages. By discriminating, we mean that the JC between texts of different languages drops significantly in comparison to JC for texts of the same language. The figure shows that Polish, Greek, and Finnish are farther from English than French is. On the other hand, in Figure 3.2, we plot the similarity between real and simulated texts in French, Greek, Polish, English, and Finnish.

In Polish, the second part of the text shifts to a different topic, and we can

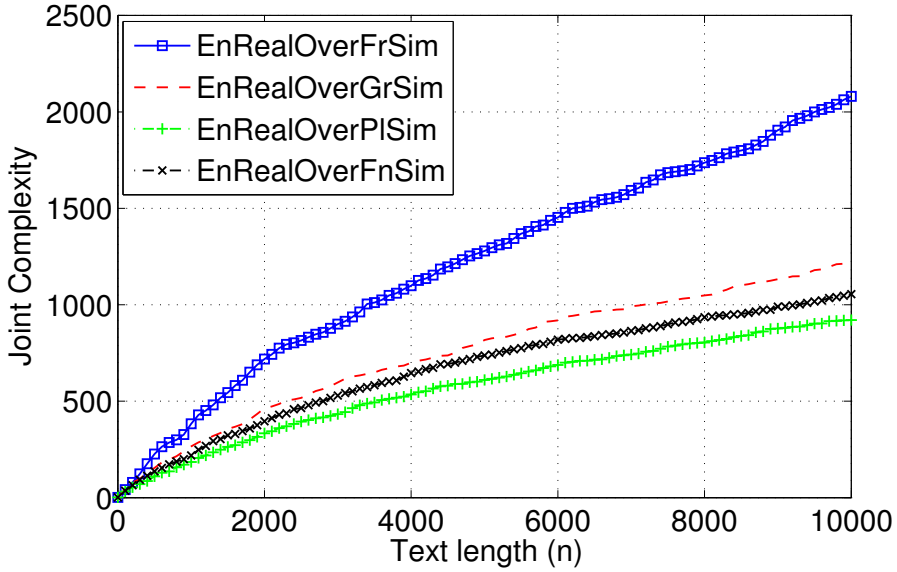

Figure 3.1: Joint Complexity of real English text versus simulated texts in French, Greek, Polish, and Finnish

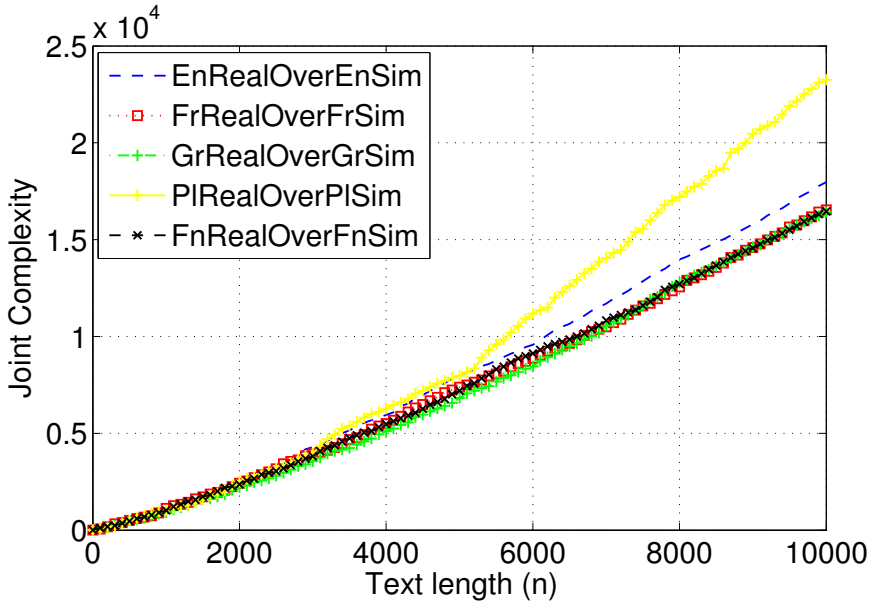

Figure 3.2: Joint complexity of real and simulated texts (3rd Markov order) in the English, French, Greek, Polish, and Finnish languages.

see that the method can capture this difference. Clearly, the JC of such texts grows like $O(n)$ as predicted by theory. In fact, computations show that with Markov models of order 3 for English versus French we have $\kappa = 0.44$; versus Greek: $\kappa = 0.26$; versus Finnish: $\kappa = 0.04$; and versus Polish: $\kappa = 0.01$, which is consistent with the results in Figure 3.1, except for the low value of κ where the convergence to the asymptotics regime is slower. In fact, they agree with the actual resolution of equation (3.7), which contains the transition to an asymptotics regime. A comparison between different topics or subjects is presented in Figure 3.3. We test four texts from books on constitutional and copyright law as well as texts extracted from two cookbooks. As we can see, the method can well distinguish the differences, and shows increased similarity for the same topic.

The complexity of a single string has been studied extensively. The literature is reviewed in [56] where precise analysis of string complexity is discussed for strings generated by unbiased memoryless sources. Another analysis of the same situation was also proposed in [54] which was the first work to present the joint string complexity for memoryless sources. It is evident from [54] that a precise analysis of JC is quite challenging due to the intricate singularity analysis and an infinite number of saddle points. Here, we deal with the joint string complexity applied to Markov sources, which to the best of our knowledge has never been tackled before. The analysis requires two-dimensional asymptotic analysis with two variable Poisson generating functions. In the following sections, we begin with a discussion of models, and notations, and we present a summary of the main results. Next, we present an extended overview of the theoretical analysis, and apply the results in the context of the Twitter social network. We present a study on expending asymptotics and periodic terms.

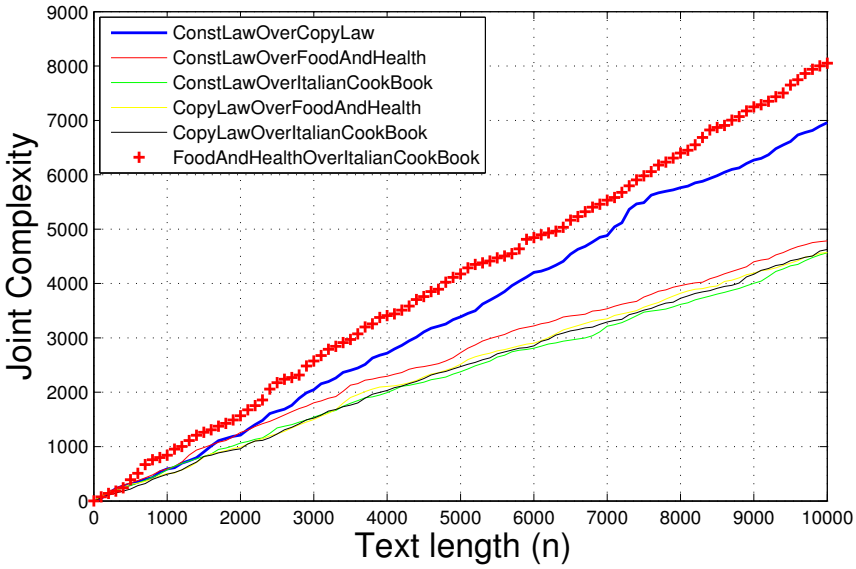

Figure 3.3: Joint complexity of real text from a variety of books spanning constitutional and copyright law to healthy cooking and recipes from the cuisine of Italy.

3.4 Main Results

We detail our main results below. We introduce some general notations and then present a summary.

3.4.1 Models and Notations

Let ω and σ be two strings over a finite alphabet $\mathcal{A}$ (*e.g.* $\mathcal{A} = \{a, b\}$). We denote by $|\omega|_\sigma$ the number of times σ occurs as a factor in ω (e.g., $|abbba|_{bb} = 2$). By convention $|\omega|_\nu = |\omega| + 1$, where ν is the empty string, because the empty string is prefix and suffix of ω and also appears between the characters, and $|\omega|$ is the length of the string.

Throughout we denote by X a string (text) and we plan to study its complexity. We also assume that its length $|X|$ is equal to n. Then the string complexity

is $I(X) = \{\omega : \ |X|_\omega \geq 1\}$. Observe that

$$|I(X)| = \sum_{\sigma \in \mathcal{A}^*} 1_{|X|_\sigma \geq 1},$$

where 1_A is the indicator function of a Boolean A. Notice that $|I(X)|$ is equal to the number of nodes in the associated suffix tree of X [56, 57, 58]. We will come back on the suffix tree in Section 3.8.

Let X and Y be two sequences (not necessarily of the same length). We have defined the *Joint Complexity* as the cardinality of the set $J(X,Y) = I(X) \cap I(Y)$. We have

$$|J(X,Y)| = \sum_{\sigma \in \mathcal{A}^*} 1_{|X|_\sigma \geq 1} \times 1_{|Y|_\sigma \geq 1} \ .$$

We now assume that the strings X and Y are respectively generated by two *independent Markov sources* of order r, so called source 1 and source 2. We will only deal here with Markov of order 1, but extension to arbitrary order is straightforward. We assume that source i, for $i \in \{1,2\}$ has the transition probabilities $P_i(a|b)$ from term b to term a, where $(a,b) \in \mathcal{A}^r$. We denote by $\mathbf{P}_1$ (resp. $\mathbf{P}_2$) the transition matrix of Markov source 1 (resp. source 2). The stationary distributions are respectively denoted by $\boldsymbol{\pi}_1(a)$ and $\boldsymbol{\pi}_2(a)$ for $a \in \mathcal{A}^r$.

Let X_n and Y_m be two strings of respective lengths n and m, generated by Markov source 1 and Markov source 2, respectively. We write $J_{n,m} = \mathbf{E}(|J(X_n, Y_m)|) - 1$ for the joint complexity, *i.e.* omitting the empty string.

3.4.2 Summary of Main Results

Definition: We say that a matrix $\mathbf{M} = [m_{ab}]_{(a,b)\in\mathcal{A}^2}$ is *rationally balanced* if $\forall (a,b,c) \in \mathcal{A}^3$: $m_{ab} + m_{ca} - m_{cb} \in \mathbb{Z}$, where $\mathbb{Z}$ is the set of integers. We say that a positive matrix $\mathbf{M} = [m_{ab}]$ is *logarithmically rationally balanced* when the matrix

$\log^*(\mathbf{M}) = [\ell_{ab}]$ is rationally balanced, where $\ell_{ab} = \log(m_{ab})$ when $m_{ab} > 0$ and $\ell_{ab} = 0$ otherwise. Furthermore, we say that two matrices $\mathbf{M} = [m_{ab}]_{(a,b)\in\mathcal{A}^2}$ and $\mathbf{M}' = [m'_{ab}]$ are *logarithmically commensurable* when matrices $\log^*(\mathbf{M})$ and $\log^*(\mathbf{M}')$ are commensurable, that is, there exist a nonzero pair of reals (x, y) such that $x \log^*(\mathbf{M}) + y \log^*(\mathbf{M}')$ is rationally balanced.

We now present our main theoretical results in a series of theorems each treating different cases of Markov sources. Most of the mathematics were primary introduced in [64] and then developed in [72] and [73].

Theorem 1 *Joint Complexity over same sources: Consider the average joint complexity of two texts of length n generated by the same general stationary Markov source, that is, $\mathbf{P} := \mathbf{P}_1 = \mathbf{P}_2$.*

(i) *[Noncommensurable Case.] Assume that $\mathbf{P}$ is not logarithmically rationally balanced. Then*

$$J_{n,n} = \frac{2\log 2}{h} n + o(1) \tag{3.1}$$

where h is the entropy rate of the source.

(ii) *[Commensurable Case.] Assume that $\mathbf{P}$ is logarithmically rationally balanced. Then there is $\varepsilon > 0$ such that:*

$$J_{n,n} = \frac{2\log 2}{h}(1 + Q_0(\log n)) + O(n^{-\varepsilon})$$

where $Q_0(.)$ is a periodic function of small amplitude.

Now we consider different sources, *i.e.* $\mathbf{P_1} \neq \mathbf{P_2}$ that are not the same and have respective transition matrices $\mathbf{P}_1$ and $\mathbf{P}_2$. The transition matrices are on $\mathcal{A}^r \times \mathcal{A}^r$. For a tuple of complex numbers (s_1, s_2) we write $\mathbf{P}(s_1, s_2)$ for a matrix whose (a, b)-th coefficient is $(\mathbf{P}_1(a|b))^{-s_1}(\mathbf{P}_2(a|b))^{-s_2}$, with the convention that

if one of the $P_i(a|b) = 0$ then the (a, b) coefficient of $\mathbf{P}(s_1, s_2) = 0$.

We first consider the case when matrix $\mathbf{P}(s_1, s_2)$ is nilpotent [61], that is, for some K the matrix $\mathbf{P}^K(s_1, s_2) = 0$.

Theorem 2 *If $\mathbf{P}(s_1, s_2)$ is nilpotent, then there exists γ_0 and $\varepsilon > 0$ such that $\lim_{n\to\infty} J_{n,n} = \gamma_0 := \langle \mathbf{1}(\mathbf{I} - \mathbf{P}(0, 0))^{-1}|\mathbf{1}\rangle$ where $\mathbf{1}$ is the unit vector and $\langle\cdot|\cdot\rangle$ is the inner product.*

This result is not surprising since the common factors are uniformly bounded in length, therefore form a finite set.

Indeed the common factors can only occur in a finite window of size K at the beginning of the strings and this will be developed in the proofs.

Throughout, now we assume that $\mathbf{P}(s_1, s_2)$ is not nilpotent. We denote by $\mathcal{K}$ the set of real tuple (s_1, s_2) such that $\mathbf{P}(s_1, s_2)$ has the main eigenvalue $\lambda(s_1, s_2)$ equal to 1. Let

$$\begin{aligned} \kappa &= \min_{(s_1,s_2)\in\mathcal{K}} \{-s_1 - s_2\} \\ (c_1, c_2) &= \arg \min_{(s_1,s_2)\in\mathcal{K}} \{-s_1 - s_2\}. \end{aligned}$$

with $\kappa < 1$.

Lemma 1 *We have either $c_1 > 0$ or $c_2 > 0$ and $(c_1, c_2) \in [-1, 0]^2$.*

Theorem 3 *Assume $\mathbf{P}(s_1, s_2)$ is not nilpotent and either $c_1 > 0$ or $c_2 > 0$. We only handle $c_2 > 0$, the case $c_1 > 0$ being obtained by symmetry.*
(i) *[Noncommensurable Case.] We assume that $\mathbf{P}_2$ is not logarithmically balanced. Let $c_0 < 0$ such that $(c_0, 0) \in \mathcal{K}$. There exists γ_1 and $\varepsilon > 0$ such that*

$$J_{n,n} = \gamma_1 n^{-c_0}(1 + O(n^{-\varepsilon})) \tag{3.2}$$

(ii) *[Commensurable Case.] Let now* $\mathbf{P}_2$ *be logarithmically rationally balanced. There exists a periodic function* $Q_1(.)$ *of small amplitude such that*

$$J_{n,n} = \gamma_1 n^{-c_0}(1 + Q_1(\log n) + O(n^{-\varepsilon})).$$

The case when both c_1 and c_2 are between -1 and 0 is the most intricate to handle.

Theorem 4 *Assume that* c_1 *and* c_2 *are between* -1 *and* 0.
(i) *[Noncommensurable Case.] When* $\mathbf{P}_1$ *and* $\mathbf{P}_2$ *are not logarithmically commensurable matrices, then there exist* α_2, β_2 *and* γ_2 *such that*

$$J_{n,n} = \frac{\gamma_2 n^{\kappa}}{\sqrt{\alpha_2 \log n + \beta_2}}(1 + O(\frac{1}{\log n})) \ . \tag{3.3}$$

(ii) *[Commensurable Case.] Let* $\mathbf{P}_1$ *and* $\mathbf{P}_2$ *be logarithmically commensurable matrices. Then there exists a double periodic function* $Q_2(.)$ *of small amplitude such that:*

$$J_{n,n} = \frac{\gamma_2 n^{\kappa}}{\sqrt{\alpha_2 \log n + \beta_2}}(1 + Q_2(\log n) + O(\frac{1}{\log n})).$$

3.5 Proof of Main Results

In this section we present proofs of our main results.

3.5.1 An important asymptotic equivalence

We have the identity:

$$J_{n,m} = \sum_{w \in \mathcal{A}^* - \{\nu\}} P(w \in I(X_n)| \geq 1) \cdot P(w \in I(Y_n) \geq 1) \ . \tag{3.4}$$

We know that from [62, 58] there is a closed formula for

$$\sum_n P(|X_n|_w \geq 1)z^n = \frac{P_1(w)z}{(1-z)D_w(z)}$$

which is, in the memoryless case,

$$D_w(z) = (1-z)(1+A_w(z)) + P_1(w)z^{|w|}$$

where $P(w)$ is the probability that w is a prefix of X'_n and $A_w(z)$ is the "autocorrelation" polynomial of word w [58]. For the Markov source, we omit the expression which carries extra indices which track to the Markov correlations for the starting symbols of the words. A complete description of the parameters can be found in [62, 58].

Although it is a closed formula, this expression is not easy to manipulate. To make the analysis tractable we notice that $w \in I(X_n)$ is equivalent to the fact that w is at least a prefix of one of the n suffices of X_n.

If the suffices would have been n independent infinite strings, then $P(w \in I(X_n))$ would be equal to $1-(1-P_1(w))^n$ whose generating function is $\frac{P1(z)z}{(1-z)(1-z+P_1(w)z)}$ which would be the same as $\frac{P_1(w)z}{(1-z)(D_w(z))}$ if we set in $D_w(z)$ with $A_w(z) = 0$ and identify $z^{|w|}$ to z.

We define $I_1(n)$ (resp. $I_2(n)$) as the set of prefixes of n independent strings built on source 1 (resp. 2). Let

$$C_{n,m} = \sum_{w \in \mathcal{A}^* - \{\nu\}} P(w \in I_1(n))P(w \in I_2(n))$$

As it is claimed in [62, 58] we prove that

Lemma 2 *There exists* $\varepsilon > 0$

$$J_{n,n} = C_{n,n}\left(1 + O(n^{-\varepsilon})\right) + O(1). \tag{3.5}$$

The proof is not developed in [62, 58] and seems to be rather complicated. It can be found in [65].

3.5.2 Functional Equations

Let $a \in \mathcal{A}$. We denote

$$C_{a,m,n} = \sum_{w \in a\mathcal{A}^*} P(w \in I_1(n))P(w \in I_2(n))$$

where $w \in a\mathcal{A}^*$ for $a \in \mathcal{A}$ means that w starts with symbol a. Notice that $C_{a,m,n} = 0$ when $n = 0$ or $m = 0$. Using the Markov nature of the string generation, the quantity $C_{a,n,m}$ for $n, m \geq 1$ satisfies the following recurrence for all $a, b \in \mathcal{A}$

$$\begin{aligned} C_{b,n,m} &= 1 + \sum_{a \in \mathcal{A}} \sum_{n_a, m_a} \binom{n}{n_a}\binom{m}{m_a} \\ &\quad \times (P_1(a|b))^{n_a}(1 - P_1(a|b))^{n-n_a} \\ &\quad \times (P_2(a|b))^{m_a}(1 - P_2(a|b))^{m-m_a} C_{a,n_a,m_a} \ , \end{aligned}$$

where n_a is an integer smaller or equal to n (resp. m_a) and denotes the number of strings among n (resp. m), independent strings from source 1 (resp. 2) which starts with symbol b, n strings starting with b and n_a tracks the number of such string that starts with ba. Quantity m_a is the counterpart for source 2. The

unconditional average $C_{n,m}$ satisfies for $n, m \geq 2$

$$\begin{aligned} C_{n,m} &= 1 + \sum_{a\in\mathcal{A}} \sum_{n_a,m_a} \binom{n}{n_a}\binom{m}{m_a} \pi_1^{n_a}(a)(1-\pi_1(a))^{n-n_a} \\ &\quad \times \pi_2^{m_a}(a)(1-\pi_2(a))^{m-m_a} C_{a,n,m}. \end{aligned}$$

since $\pi_i(a)$ is the probability that a string from source i starts with symbol a.

We introduce the double Poisson transform of $C_{a,n,m}$ as

$$C_a(z_1, z_2) = \sum_{n,m\geq 0} C_{a,n,m} \frac{z_1^n z_2^m}{n!m!} e^{-z_1-z_2} \tag{3.6}$$

which translates the recurrence (in the formula above n_a tracks the number) into the following functional equation:

$$\begin{aligned} C_b(z_1, z_2) &= (1-e^{-z_1})(1-e^{-z_2}) \\ &\quad + \sum_{a\in\mathcal{A}} C_a\left(P_1(a|b)z_1, P_2(a|b)z_2\right). \end{aligned} \tag{3.7}$$

Furthermore, the cumulative double Poisson transform

$$C(z_1, z_2) = \sum_{n,m\geq 0} C_{n,m} \frac{z_1^n z_2^m}{n!m!} e^{-z_1-z_2} \tag{3.8}$$

satisfies

$$\begin{aligned} C(z_1, z_2) &= (1-e^{-z_1})(1-e^{-z_2}) \\ &\quad + \sum_{a\in\mathcal{A}} C_a(\pi_1(a)z_1, \pi_2(a)z_2) \ . \end{aligned} \tag{3.9}$$

3.5.3 Double DePoissonization

The asymptotics of the coefficient $C_{n,m}$ are extracted from the asymptotics of function $C(z_1, z_2)$ where $\Re(z_1, z_2) \to \infty$. This is an extension of DePoissonization theorems of [63, 64, 57], and are used to prove the following lemma.

Lemma 3 (DePoissonization) *When n and m tend to infinity:*

$$C_{n,m} = C(n, m)(1 + O(n^{-1}) + O(m^{-1})) \ .$$

This equivalence is obtained by proving some growth properties of $C(z_1, z_2)$ when (z_1, z_2) are complex numbers; such properties are stated in [64].

3.5.4 Same Markov sources

We first present a general result when the Markov sources are identical: $\mathbf{P}_1 = \mathbf{P}_2 = \mathbf{P}$. In this case (3.7) can be rewritten with $c_a(z) = C_a(z, z)$:

$$c_b(z) = (1 - e^{-z})^2 + \sum_{a \in \mathcal{A}} c_a \left(P(a|b)z\right). \tag{3.10}$$

This equation is directly solvable by the Mellin transform $c_a^*(s) = \int_0^\infty c_a(x)x^{s-1}dx$ defined for $-2 < \Re(s) < -1$. For all $b \in \mathcal{A}$ we find [57]

$$c_b^*(s) = (2^{-s} - 2)\Gamma(s) + \sum_{a \in \mathcal{A}} (P(a|b))^{-s} c_a^*(s) \ . \tag{3.11}$$

Introducing $c^*(s) = \int_0^\infty C(z, z)z^{s-1}dz$ [66], and the property of Mellin transform $\int f(ax)x^{s-1} = a^{-s} \int f(x)x^{s-1}dx$. The definition domain of $c^*(s)$ is $\Re(s) \in$

$(-2,-1)$, and the Mellin transform $c^*(s)$ of $C(z,z)$ becomes

$$c^*(s) = (2^{-s}-2)\Gamma(s) + \sum_{a\in\mathcal{A}} (\pi(a))^{-s} c_a^*(s) \ .$$

Thus

$$c^*(s) = (2^{-s}-2)\Gamma(s)\left(1 + \langle \mathbf{1}(\mathbf{I}-\mathbf{P}(s))^{-1}|\boldsymbol{\pi}(s)\rangle\right) \tag{3.12}$$

where $\mathbf{1}$ is the vector of dimension $|\mathcal{A}|$ made of all 1's, $\mathbf{I}$ is the identity matrix, and $\mathbf{P}(s) = \mathbf{P}(s,0) = \mathbf{P}(0,s)$, $\boldsymbol{\pi}(s)$ is the the vector made of coefficients $\pi(a)^{-s}$ and $\langle .|. \rangle$ denotes the inner product.

By applying the methodology of Flajolet [67, 57], the asymptotics of $c(z)$ for $|\arg(z)| < \theta$ is given by the residues of the function $c^*(s)z^{-s}$ occurring at $s=-1$ and $s=0$. They are respectively equal to $\frac{2\log 2}{h} z$ and $-1-\langle \mathbf{1}(\mathbf{I}-\mathbf{P}(0,0))^{-1}\boldsymbol{\pi}(0)\rangle$. The first residues comes from the singularity of $(\mathbf{I}-\mathbf{P}(s))^{-1}$ at $s=-1$. This led to the formula expressed in Theorem 1(i). When $\mathbf{P}$ is logarithmically rationally balanced then there are additional poles on a countable set of complex numbers s_k regularly spaced on the same imaginary axes containing -1 and such that $\mathbf{P}(s_k)$ has eigenvalue 1. These poles contributes to the periodic terms in Theorem 1(ii).

Computations on the trained transition matrix show that a Markov model of order 3 for English text has entropy of 0.944221, while French text has an entropy of 0.934681, Greek text has an entropy of 1.013384, Polish text has an entropy of 0.665113, and Finnish text has an entropy of 0.955442. This is consistent with Figure 3.2.

3.5.5 Different Markov Sources

In this section we identify the constants in Theorem 3 and Theorem 4 with the assumption $\mathbf{P}_1 \neq \mathbf{P}_2$. We cannot obtain a functional equation for $C_a(z,z)$'s, and we thus have to deal with two variables z_1 and z_2. We define the double Mellin transform $C_a^*(s_1,s_2) = \int_0^\infty \int_0^\infty C_a(z_1,z_2) z_1^{s_1-1} z_2^{s_2-1} dz_1 dz_2$ and similarly the double Mellin transform $C^*(s_1,s_2)$ of $C(z_1,z_2)$. And thus we should have the identity

$$\begin{aligned} C_b^*(s_1,s_2) &= \Gamma(s_1)\Gamma(s_2) \\ &\quad + \sum_{a\in\mathcal{A}} (P_1(a|b))^{-s_1} (P_2(a|b))^{-s_2} C_a^*(s_1,s_2) \end{aligned} \tag{3.13}$$

which leads to

$$C^*(s_1,s_2) = \Gamma(s_1)\Gamma(s_2)\left(1 + \langle \mathbf{1}(\mathbf{I} - \mathbf{P}(s_1,s_2))^{-1} | \boldsymbol{\pi}(s_1,s_2)\rangle\right) \tag{3.14}$$

where $\boldsymbol{\pi}(s_1,s_2)$ denotes the vector composed of coefficients $\pi_1(a)^{-s_1}\pi_2(a)^{-s_2}$. In fact to define the Mellin transform we need to apply it to $C(z_1,z_2) - \frac{\partial}{\partial z_1} C(0,z_2) z_1 e^{-z_1} - \frac{\partial}{\partial z_2} C(z_1,0) z_2 e^{-z_2}$ which leads to exponentially decaying but we omit this technical detail, which is fully described in [64]. The original value $C(z_1,z_2)$ is obtained via the inverse Mellin transform

$$C(z_1,z_2) = \frac{1}{(2i\pi)^2} \int\int C(s_1,s_2) z_1^{-s_1} z_2^{-s_2} ds_1 ds_2 \tag{3.15}$$

thus

$$C(z,z) = \frac{1}{(2i\pi)^2} \int_{\Re(s_1)=\rho_1} \int_{\Re(s_2)=\rho_2} C^*(s_1,s_2) z^{-s_1-s_2} ds_1 ds_2 \ . \tag{3.16}$$

where (ρ_1, ρ_2) belongs to the definition domain of $C^*(s_1, s_2)$; $\rho \in (-2,-1) \times (-2,-1)$.

We denote $L(s)$ be the function of complex s such that $\mathbf{P}(s, L(s))$ has eigenvalue 1 or equivalently such that $(\mathbf{I} - \mathbf{P}(s_1, s_2))^{-1}$ ceases to exist. The function $L(s)$ is meromorphic and has several branches; one branches describes the set $\mathcal{K}$ when s is real. Now to evaluate the double integral (3.16) we move the line of integration with respect to s_2 from ρ_2 to some $M > 1$ while fixing the value of s_1 and collecting on the way the residues on all the poles encounterned. In particular, the dominant residue at $s_2 = L(s)$ contributes

$$\begin{aligned} C(z,z) &= \frac{1}{2i\pi} \int_{\Re(s_1)=\rho_1} \mu(s_1)\Gamma(s_1)\Gamma(L(s_1)) z^{-s_1-L(s_1)} ds_1 \\ &\quad + O(z^{\rho_1 - M}) \end{aligned} \tag{3.17}$$

where $\mu(s)$ is the residue of $\langle \mathbf{1}(\mathbf{I} - \mathbf{P}(s,s_2))^{-1} \boldsymbol{\pi}(s_1, s_2) \rangle$ at point $(s, L(s))$, that is,

$$\mu(s_1) = \frac{1}{\frac{\partial}{\partial s_2} \lambda(s_1, s_2)} \langle \mathbf{1} | \boldsymbol{\zeta}(s_1, s_2) \rangle \langle \mathbf{u}(s_1, s_2) | \boldsymbol{\pi}(s_1, s_2) \rangle \Big|_{s_2 = L(s_1)} \ .$$

where $\lambda(s_1, s_2)$ is the eigenvalue which has value 1 at $(s, L(s))$ and $\mathbf{u}(s_1, s_2)$ and $\boldsymbol{\zeta}(s_1, s_2)$ are respectively the left and right eigenvectors with the convention that $\langle \boldsymbol{\zeta}(s_1, s_2) | \mathbf{u}(s_1, s_2) \rangle = 1$.

The expression is implicitly a sum since the function $L(s)$ is meromorphic, but we retain only the branch where $\lambda(s_1, s_2)$ is the main eigenvalue of $\mathbf{P}(s_1, s_2)$ that contributes to the leading term in the expansion of $C(z, z)$. For more details

see [64] where the analysis is specific to a case where one of the sources, namely source 2, is memoryless uniform, *i.e.* $\mathbf{P}_2 = \frac{1}{|\mathcal{A}|}\mathbf{1} \otimes \mathbf{1}$.

The next step consists in moving the integration line for s_1 from ρ_1 to c_1 which corresponds to the position where function $-s_1 - L(s_1)$ (actually equal to κ) attains the minimum value. We only consider the case when $L(c_1) = c_2 < 0$ (the other case is obtained by symmetry). The poles are due to the function $\Gamma(.)$. The first pole encountered is $s_1 = -1$ but this pole cancels with the technical arrangement discussed earlier.

We do not work on the simple case, *i.e.* when $c_1 > 0$. We meet the second pole at $s = 0$ and the residue is equal to $\mu(0)\Gamma(c_0)z^{-c_0}$ since $L(0) = c_0$. This quantity turns out to be the leading term of $C(z, z)$ since the integration on $\Re(s_1) = c_1$ is $O(z^\kappa)$. This proves Theorem 3. When $\mathbf{P}_2$ is logarithmically balanced, there exists ω such that $\lambda(s, L(s) + ik\omega) = 1$ for $k \in \mathbb{Z}$ and the terms $z^{c_0+ik\omega}$ lead to a periodic contribution.

The difficult case is when $-1 < c_1 < 0$. In this case, $C(z, z) = O(z^\kappa)$ but to find precise estimates one must use the saddle point method [67], at $s = c_1$ since the integration is of the form $\int_{\Re(s)} = c_1 f(s) \exp(-(s + L(s))A)ds$, where $f(s) = \mu(s)\Gamma(s)\Gamma(L(s))$, and $A = \log z \to \infty$. We naturally get an expansion when $\Re(z) \to \infty$

$$C(z, z) = \frac{e^{\kappa \log z}\mu(c_1)}{\sqrt{(\alpha_2 \log z + \beta_2)}} \left(1 + O(\frac{1}{\sqrt{\log z}})\right)$$

with $\alpha_2 = L''(c_1)$ and $\beta_2 = \frac{\mu'(c_1)}{\mu(c_1)}$. In fact, the saddle point expansion is extendible to any order of $\frac{1}{\sqrt{\log z}}$. This proves Theorem 4 in the general case. However, in the case when $\mathbf{P}_1$ and $\mathbf{P}_2$ are logarithmically commensurable, the line $\Re(s_1) = c_1$ contains an infinite number of saddle points that contribute in a double periodic

additional term.

Example: Assume that we have a binary alphabet $A = \{a, b\}$ with memory 1, and transition matrices $P_1 = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$ and $P_2 = \begin{bmatrix} 0.2 & 0.8 \\ 0.8 & 0.2 \end{bmatrix}$

The numerical analysis gives $C_{n,n} = 13.06 \frac{n^{0.92}}{\sqrt{1.95 \log n + 56.33}} \Longrightarrow C_{n,n} = 3.99(\log 2)n$.

Inspired from the general results about the asymptotic digital trees and suffix tree parameters distribution we conjecture the following [58, 59, 60].

Conjecture 1

- *(i) The variance $V_{n,n}$ of the joint complexity of two random texts of same length n generated by two Markov sources is of order $O(n^\kappa)$ when $n \to \infty$.*
- *(ii) The normalised distribution of the joint complexity $\mathbf{J}_{n,n}$ of these two texts tends to the normal distribution when $n \to \infty$.*

Remark By "normalized distribution" we mean the distribution of $\frac{\mathbf{J}_{n,n} - J_{n,n}}{\sqrt{V_{n,n}}}$.

3.6 Expending Asymptotics and Periodic Terms

The estimate $J_{n,n} = \gamma \frac{n^k}{\sqrt{\alpha \log n + \beta}} (1 + Q(\log n) + O(\frac{1}{\log n}))$ which appears in the case of a different Markov source comes from a saddle point analysis. The potential periodic terms $Q(\log n)$ occur in a case where the Kernel $\mathcal{K}$ shows an infinite set of saddle points. It turns out that the amplitude of the periodic terms is of the order of $\Gamma(\frac{2i\pi}{\log |A|})$, *i.e.* of the order of 10^{-6} for binary alphabet, but it rises when $|A|$ increases. For example when $|A| \geq 26$ such as in the Latin alphabet used

in English (including spaces, commas, and other punctuation) we get an order within 10^{-1}.

Figure 3.4 shows the number of common factors from two texts generated from two memoryless sources. One source is a uniform source over the 27 Latin symbols (such source is so-called monkey typing), the second source takes the statistic of letters occurrence in English. The trajectories are obtained by incrementing each text one by one. Although not quite significant, the logarithmic oscillations appear in the trajectories. We compare this with the expression $\gamma \frac{n^k}{\sqrt{\alpha \log n + \beta}}$ without the oscillation terms which are actually $13.06 \frac{n^{0.92}}{\sqrt{1.95 \log n + 73.81}}$.

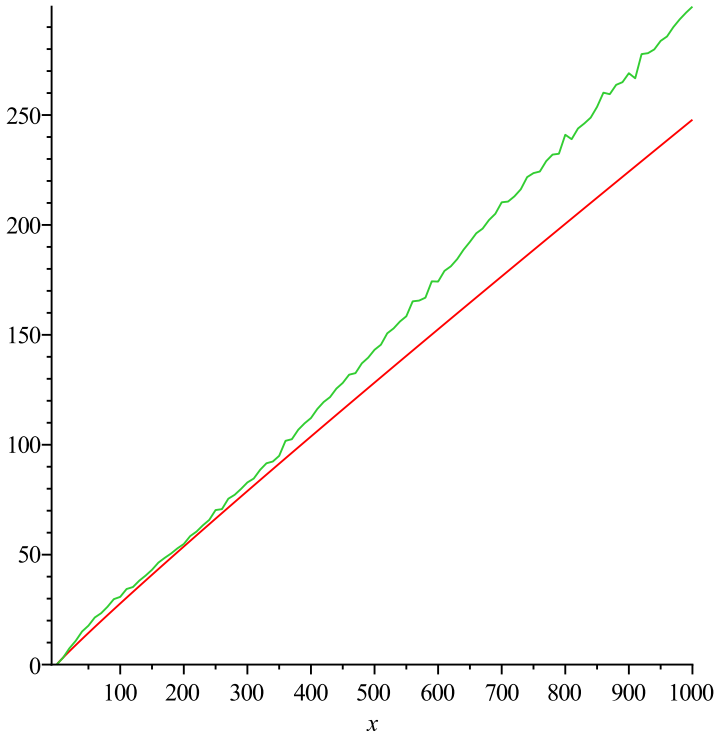

Figure 3.4: Joint Complexity (y axis) of memoryless English text (x axis) versus monkey typing. The first order theoretical average is shown in red (cont. line).

In fact it turns out that the saddle point expression has a poor convergence term since the $O(\frac{1}{\log n})$ is indeed in $\frac{1}{\alpha \log n + \beta}$ made poorer since the latter does not make less than $\frac{1}{\beta}$ for the text length range that we consider. But the saddle

point approximation leads to the estimate factor $P_k((\alpha \log n + \beta)^{-1})$ of

$$J_{n,n} = \gamma \frac{n^k}{\sqrt{\alpha \log n + \beta}}(1 + P_k((\alpha \log n + \beta)^{-1})) + O(\frac{1}{(\log n)^{k+\frac{1}{2}}}) + Q(\log n) \quad (3.18)$$

where $P_k(x) = \sum_{j=1}^{k} A_i x^j$ is a specific series polynomial of degree k. The error term is thus in $(\alpha \log n + \beta)^{-k-\frac{1}{2}})$ but is not uniform for k. Indeed, the expansion polynomial P_k diverges when $k \to \infty$. Therefore for a given value of x there is an optimal value of k which minimizes the error term whose relative order is given by $|A_{k+1}x^{k+1} + A_{k+2}x^{k+2}|$, since the series of the A_i's have alternated signs.

Figure 3.5 shows the different error terms versus k for $x = \frac{1}{\beta}$. The optimal is reached for $k = 5$ with a relative error of 10^{-3}. Figure 3.6 shows the new theoretical average with $P_5((\alpha \log n + \beta) - 1)$ as correcting term. The estimate is now well centered but does not include the periodic terms, which are shown in Figure 3.7. As we can see in Figure 3.7 the fluctuation confirms the Conjecture 1 about the variance of Joint Complexity.

3.7 Numerical Experiments in Twitter

In this Section we apply Joint Complexity in Twitter in order to perform topic detection and extract similarities between tweets and communities. We consider four sets of tweets from different sources – the New York Times, BBC Business, CNN, and BBC Breaking. In Twitter the maximum length of the messages is 140 characters. We take the hypothesis that the sources are Markov sources of finite order. Individual tweets are of arbitrary length. The alphabets of the different languages of tweet sets, are converted on ASCII.

We compute the JC value for pairs of tweet sets in Figure 3.8. We used tweets

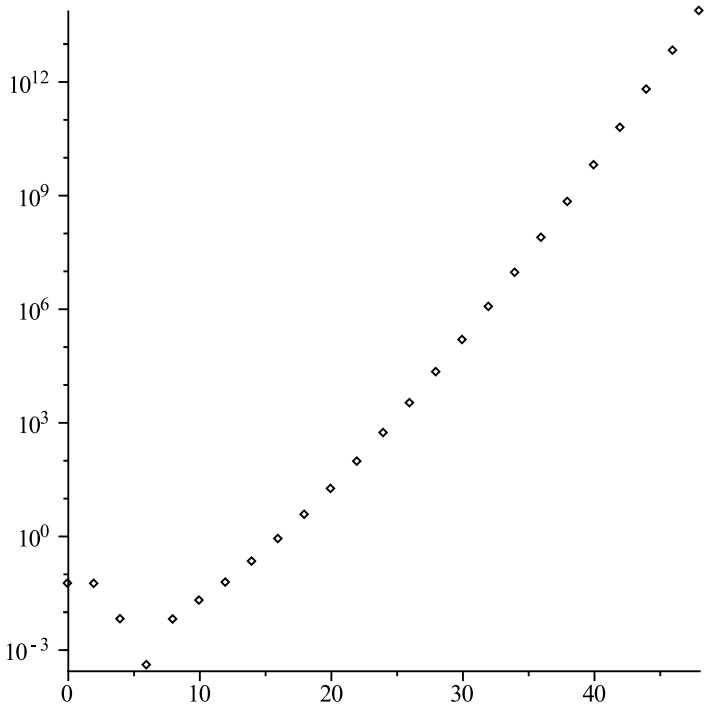

Figure 3.5: Relative error in the saddle point expansion versus order for $x = 1/\beta$.

from the 2012 Olympic Games and 2012 United States elections. We took two sets from each of these tweet sets to run our experiments, but first we removed the tags similar to the topic, such as #elections, #USelections, #USelections2012, #Olympics, #Olympics2012, #OlympicGames and so on. As we can see in Figure 3.8, the JC is significantly high when we compare tweets in the same subjects, for both real and simulated tweets (simulated tweets are generated from a Markov source of order 3 trained on the real tweets). We observe the opposite when we compare different subjects. In the US elections topic, we can see that the JC increases significantly when the number of characters is between 1700 and 1900. This is because users begin to write about and discuss the same subject. We can observe the same in the Olympic Games topic between 6100 and 6300 and between 9500 and 9900. This shows the applicability of the method

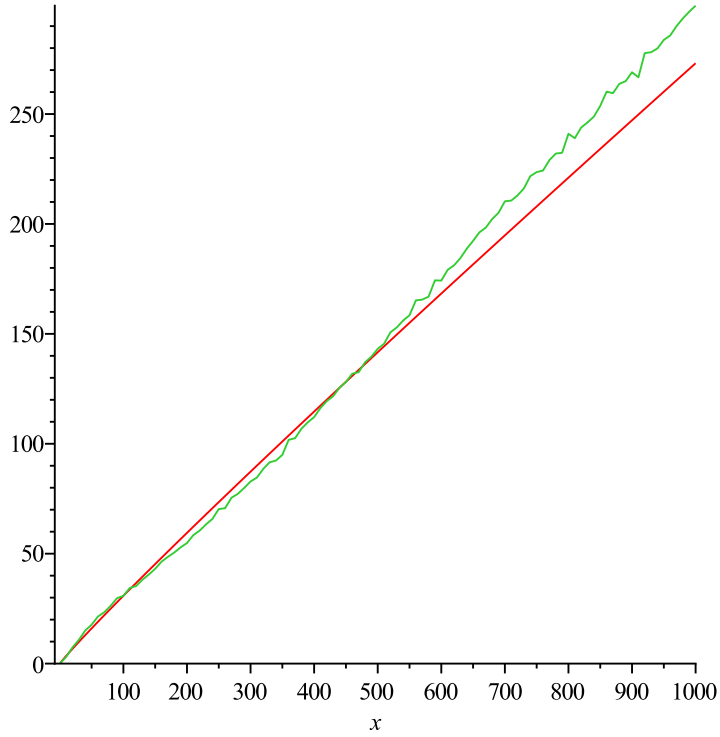

Figure 3.6: Joint Complexity (y axis) of memoryless English text (x axis) versus monkey typing. The optimal fifth order theoretical average is shown in red (cont. line).

to distinguish information sources. In Figure 3.9, we plot the JC between the simulated texts and compare with the theoretical average curves expected by the proposed methodology.

3.8 Suffix Trees

A Suffix Tree [70] is a compressed trie [71] containing all the suffixes of the given text as their keys and positions in the text as their values. We may refer as PAT tree or, in an earlier form, position tree. The suffix tree allows particularly fast implementations of many important string operations.

The construction of such a tree for a string S of length n takes on average

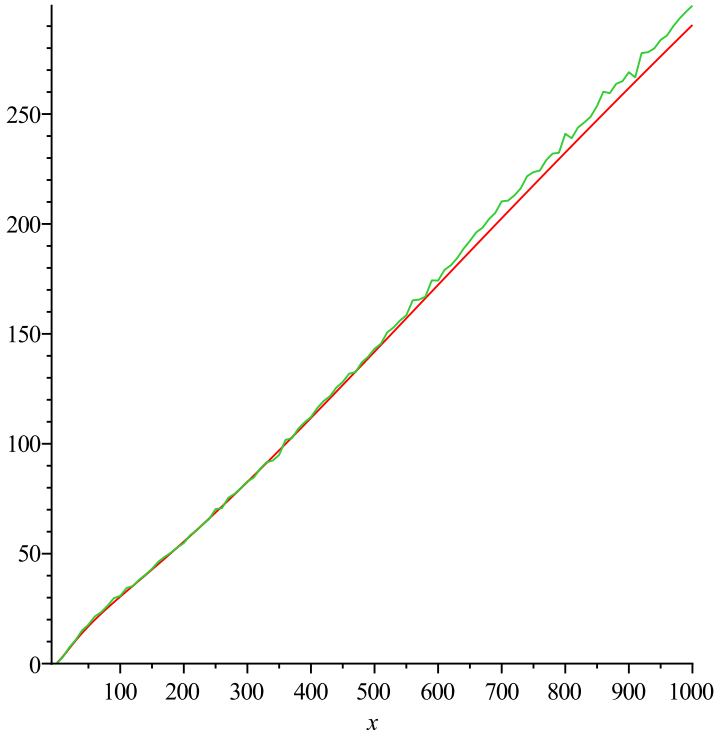

Figure 3.7: Average Joint Complexity (y axis) of memoryless English text versus monkey typing. The optimal fifth order theoretical (in red, cont. line) plus periodic terms.

$O(n \log n)$ time and space linear in the length of n. Once constructed, several operations can be performed quickly, for instance locating a substring in S, locating a substring if a certain number of mistakes are allowed, locating matches for a regular expression pattern and other useful operations.

Every node has outgoing edges labeled by symbols in the alphabet of S. Thus every node in the Suffix Tree can be identified via the word made of the sequence of labels from the root to the node. The Suffix Tree of S is the set of the nodes which are identified by any factor of S.

Suffix Tree Compression: An unitary sequence of nodes is a chain where nodes have all degree 1. If an unitary chain ends to a leaf then it correspond to a factor which appears only once in S. The chain can be compressed into a single

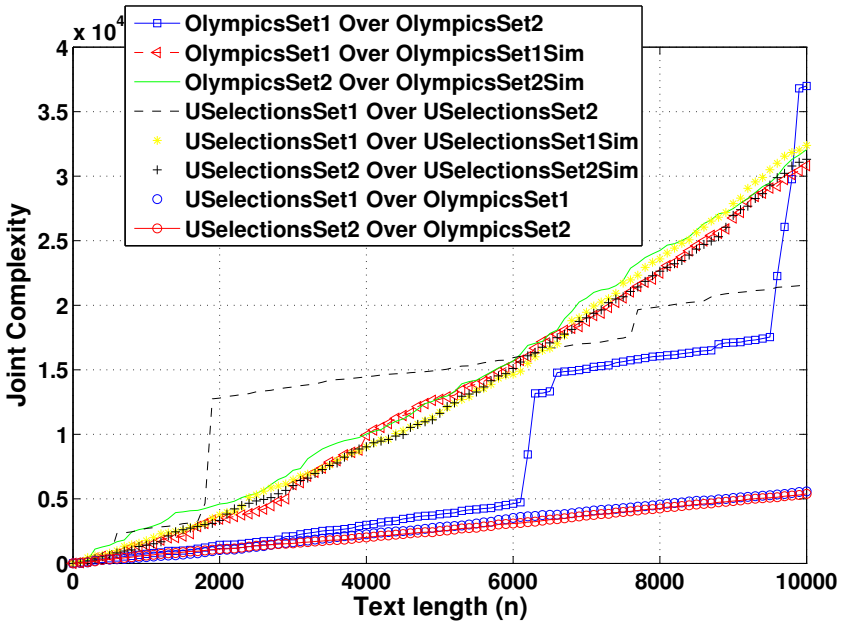

Figure 3.8: Joint Complexity of four tweet sets from the 2012 United States elections and Olympic Games. The text is an incremental aggregation of tweets from these sets.

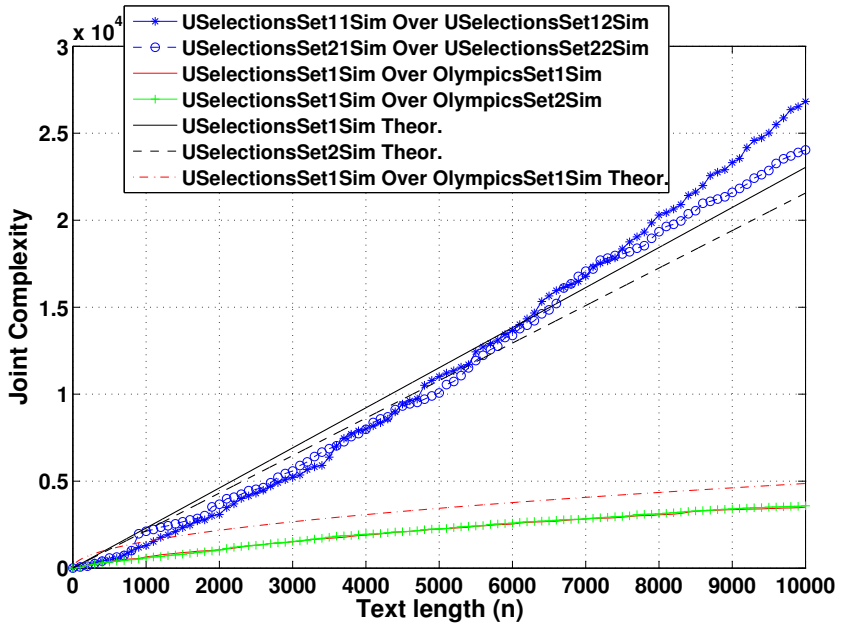

Figure 3.9: Joint Complexity of tweet sets from the 2012 United States elections and Olympic games, in comparison with theoretic average curves. The text is an incremental aggregation of tweets from these sets.

node. In this case the concatenation of all the labels of the chain correspond to a suffix of S and the compressed leaf will contain a pointer to this suffix. The other (internal) nodes of the Compressed Suffix Tree correspond to the factors which appear at least twice in S. This is the Compressed Suffix Tree version whose size is $O(n)$ in average, otherwise the uncompressed version is $O(n^2)$.

Similarly any other unitary chain which does not go to a leaf can also be compressed in a single node, the label of the edges to this node is the factor obtained buy concatenating all the labels. This is called the Patricia compression and in general gives very small reduction in size.

The Suffix Tree implementation and the comparison process (ST superposition) between two Suffix Trees in order to extract the common factors of the text sequences can be found in the Appendix A.

3.8.1 Examples of Suffix Trees

The proposed tree structure in the Appendix A needs $O(n)$ time to be stored and sub linear time for the superposition (finding overlaps). Two main examples with graph tree representation follow in Figure 3.10 and 3.11 for the sequence "apple" and "maple", respectively, which have 9 common factors. Figure 3.12 and 3.13 show the Suffix Tree for the sequence "healthy" and "sealed", which have 7 common factors. The construction of the Suffix Tree for the sequence "apple" and "maple", as well as the comparison between them (ST superposition) is shown in Figure 3.14.

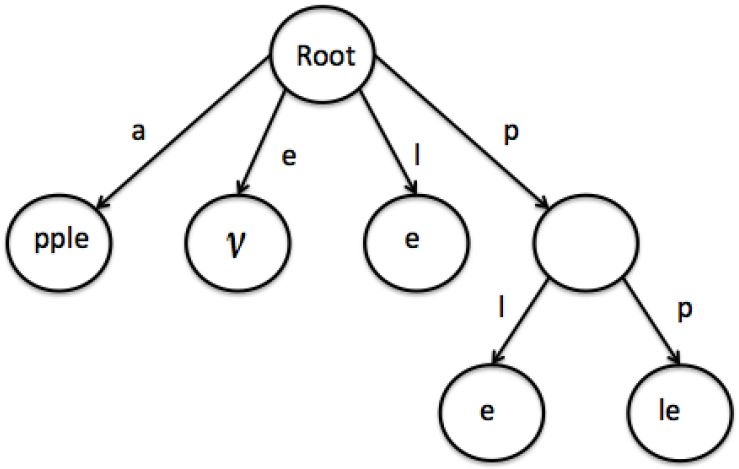

Figure 3.10: Suffix Tree for the sequence "apple", where ν is the empty string.

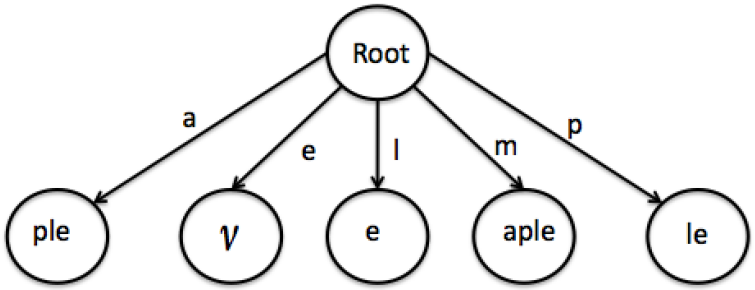

Figure 3.11: Suffix Tree for the sequence "maple", where ν is the empty string.

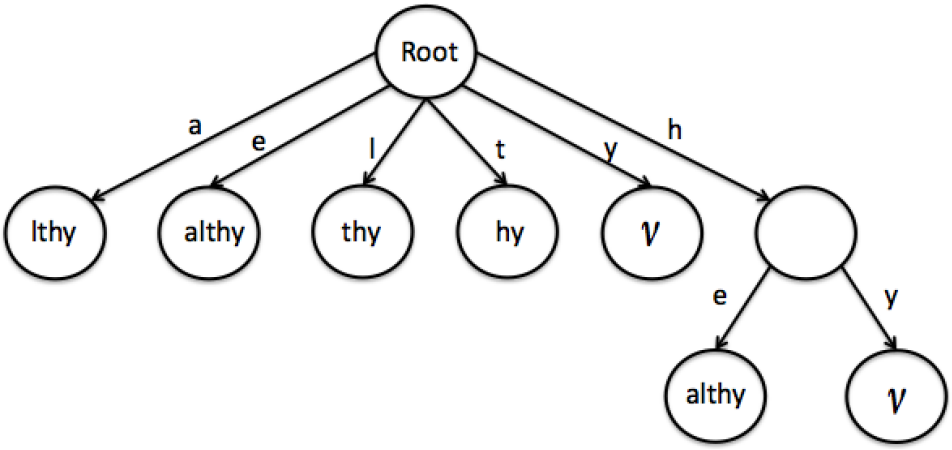

Figure 3.12: Suffix Tree for the sequence "healthy", where ν is the empty string

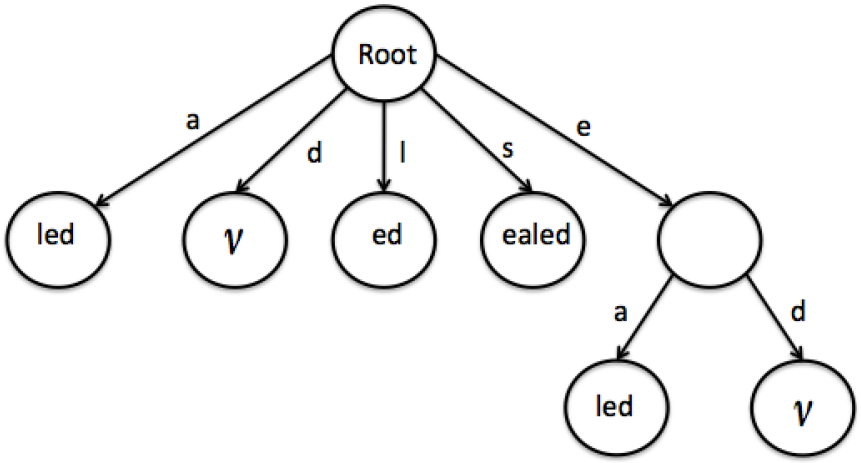

Figure 3.13: Suffix Tree for the sequence "sealed", where ν is the empty string.

3.9 Snow Data Challenge

In February 2014 the Snow Data Challenge of the World Wide Conference (WWW'14) was announced. Every year the WWW community organize a different challenge. In 2014 the challenge was about extracting topics in Twitter. The volume of information in Twitter is very high and it is often difficult to extract topics in real time. The task of this challenge was to automatically mine social streams to provide journalists a set of headlines and complementary information that summarize the most important topics for a number of timeslots (time intervals) of interest.

The Snow Challenge organization provided a common framework to mine the Twitter stream and asked to automatically extract topics corresponding to known events (*e.g.*, politics, sports, entertainment). The crawled data were divided in timeslots and we had to produce a fixed number of topics for selected timeslots. Each topic should be in the form of a short headline that summarizes a topic related to a piece of news occurring during that timeslot, accompanied by a set of tweets, URLs of pictures (extracted from the tweets), and a set of keywords.

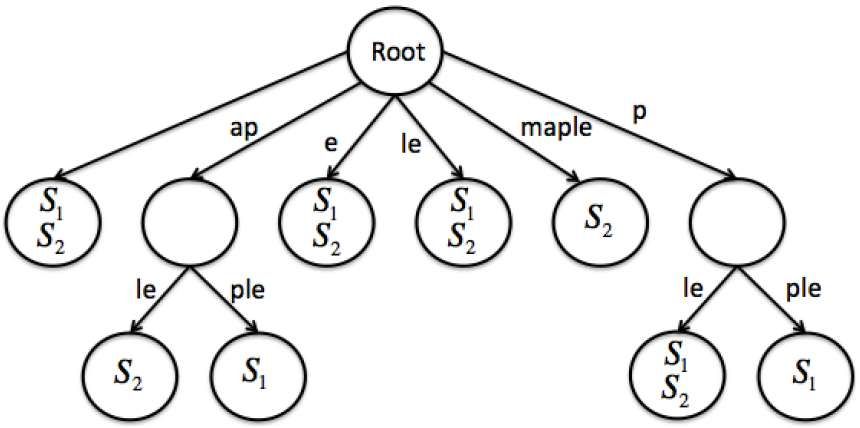

Figure 3.14: Suffix Tree superposition for the sequences S_1 = *apple and* S_2 = *maple.*

The expected output format was the following: *[headline, keywords, tweetIds, picture urls].*

We got the third Prize in the Challenge, while our method was discussed to receive the first Prize. The main advantage of the method was its language agnostics and we were able to report topics in many different languages other than English, *e.g.* French, Spanish, Korean, *etc.* The Challenge organization restricted us to report topics only in English, since the evaluation was decided to be done in that language, but we decided to report the exact output of the method [68].

First, we collected tweets for 24 hours; between Tuesday Feb. 25, 18:00 and Wednesday Feb. 26, 18:00 (GMT). The crawling collected more than $1,041,062$ tweets between the Unix timestamps 1393351200000 and 1393437600000 and was conducted through the use of the *Twitter* streaming API by following $556,295$ users and also looking for four specific keywords: *Syria; terror; Ukraine; bitcoin.* The dataset was split into 96 timeslots, where each timeslot contains tweets for every 15 minutes, starting at 18:00 on Tuesday 25th 2014. The challenge then

consisted in providing a minimum of one and a maximum of ten different topics per timeslot, along with a headline, a set of keywords and a URL of a relevant image for each detected topic. The test dataset activity and the statistics of the dataset crawl are described more extensively in [68].

3.9.1 Topic Detection Method

Until the present, the main methods used for text classification are based on *keywords* detection and machine learning techniques as was extensively described in Chapter 2. Using keywords in tweets has several drawbacks because of wrong spelling or distorted usage of the words – it also requires lists of stop-words for every language to be built – or because of implicit references to previous texts or messages. The machine learning techniques are generally heavy and complex and therefore may not be good candidates for real time text processing, 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 the kind of application, where we want minimum if any human intervention. 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 and throughput.

In our method [69] we use the Joint Complexity (computed via Suffix Trees) as a metric to quantify the similarity between the tweets. This is a significant achievement because we used a general method adapted to the Snow Data Challenge.

According to the dataset described in Section 3.9 and in [68] we have $N = 96$ timeslots with $n = 1 \dots N$. For every tweet t_i^n, where $i = 1 \dots M_n$, with M_n being the total number of tweets, in the n-th timeslot, we build a Suffix Tree, $ST(t_i^n)$, as described in Section 3.3. Building a Suffix Tree is an operation that costs linear time and takes $O(m)$ space in memory, where m is the length of the tweet.

Then we compute the Joint Complexity metric as mentioned earlier, $JC(t_i^n, t_j^n)$ of the tweet t_i^n with every other tweet t_j^n of the n-th timeslot, where $j = 1 \dots M_n$, and $j \neq i$ (by convention we choose $JC(t_i^n, t_i^n) = 0$). For the N timeslots we store the results of the computation in the matrices $T_1, T_2, \dots, T_N$ of $M_n \times M_n$ dimensions.

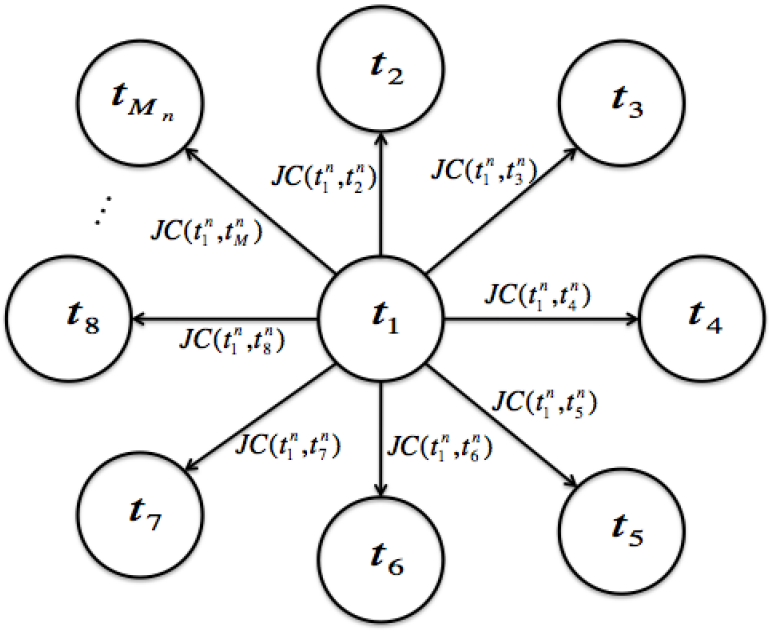

Figure 3.15: Representation of the first row of the n-th timeslot via weighted graph.

We represent each matrix T_n by fully-connected weighted graphs. Each tweet is a node in the graph and the two-dimensional array T_n holds the weight of each

edge, as shown in Fig. 3.15. Then, we calculate the score for each node in our graph by summing all the edges which are connected to the node. The node that gives the highest score is the most representative and central tweet of the timeslot.

$$\mathbf{T}_n = \begin{pmatrix} 0 & JC(t_1^n, t_2^n) & JC(t_1^n, t_3^n) & \cdots & JC(t_1^n, t_M^n) \\ JC(t_2^n, t_1^n) & 0 & JC(t_2^n, t_3^n) & \cdots & JC(t_2^n, t_M^n) \\ JC(t_3^n, t_1^n) & JC(t_3^n, t_2^n) & 0 & \cdots & JC(t_3^n, t_M^n) \\ \vdots & \vdots & & \ddots & \vdots \\ JC(t_M^n, t_1^n) & JC(t_M^n, t_2^n) & JC(t_M^n, t_3^n) & \cdots & 0 \end{pmatrix}$$

Most of the timeslots have $M = 5{,}000$ tweets, so matrices $T_1, T_2, \ldots, T_N$ have approximately $25M$ entries for every timeslot. Since they are symmetric, only half of these entries could be used, *i.e* the upper triangular of matrices $T_1, T_2, \ldots, T_N$.

The whole Joint Complexity computation was run in a multithreaded way on a 24 processor machine: $k = 23$ threads are started and each thread works on a disjoint set of rows.

This implementation allowed the program to run in on average 90 seconds in order to compute a 15-minutes timeslot, on a simple computer. These computations were only run once, as soon as the data was properly divided into the 15-minutes timeslots, and the results were saved in files which were subsequently used to perform the rest of implementation.

When we finally get the scores of the Joint Complexity metric, we try to find the R most representative and central tweets of every timeslot as required in the Challenge. At first we get the sum of the Joint Complexity, $S_i^n = \sum_{j=1\ldots M, j\neq i} JC_{t_i^n, t_j^n}$, of the i-th tweet with every other tweet j in the n-th

timeslot, and finally we get the vector $\mathbf{S}^n = [S_1^n, S_2^n, \ldots, S_{M_n}^n]$ for every timeslot.

We sort the elements of each vector $\mathbf{S}^n$ in descending order and we get the R most representative and central tweets in the following way: The best-ranked tweet is chosen unconditionally, the second one is picked only if its JC score with the first one is below a chosen threshold Thr_{low}, otherwise it is added to the list of related tweets of the first tweet; similarly, the third one is picked only if its JC score with the first two is below Thr_{low}, etc. This ensures that the topics are dissimilar enough and it classifies best ranked tweets into topics at the same time.

3.9.2 Headlines

In order to produce *headlines* as requested from the Data Challenge, we removed punctuation, special characters, *etc.*, of each selected tweet. We could use a better selection but due to the lack of time in preparation for the Challenge we used this simple selection method. We constructed the headlines of each topic and we run through the list of related tweets to keep only tweets that are different enough from the selected one (ensures no duplicates), we did so by keeping only the tweets whose JC score with the selected tweet and all previous related tweets was above a chosen threshold Thr_{max}. We first chose empirical values 400 and 600 for Thr_{low} and Thr_{max} respectively, but then we noticed that many topics had only one related tweet (all the others were retweets), so we decided to lower that threshold to $Thr_{low} = 240$. Due to the lack of time during the contest, we did not recompute the results on the whole dataset so only a handful of timeslots benefited from this better Thr_{low}. The final plan was to come up with a formula to have the system determine those thresholds automatically depending on the number of characters of each tweet.

While running through the list of related tweets we computed the *bag-of-words* used to construct the list of keywords and we also checked the original *.json* data to find a URL pointing to a valid image related to the topic.

We chose to print the first top eight topics for each timeslot, which are the heads of the first eight lists of related tweets.

3.9.3 Keywords Extraction

In order to produce a list of *keywords* per topic as requested from the Data Challenge, we first removed articles (stop-words), punctuation, special characters, etc., from the *bag-of-words* constructed from the list of related tweets of each topic. We got a list of words, and then we ordered them by decreasing frequency of occurrence. Finally, we reported the k most frequent words, in a list of keywords $\mathbf{K} = [K^1_{1\ldots k}, K^2_{1\ldots k}, \ldots, K^N_{1\ldots k}]$, for the N total number of timeslots.

3.9.4 Media URLs

As requested from the Data Challenge, we provided a representative Media URL per topic. The body of a tweet (in the *.json* file format), contains a URL information for links to media files such as pictures or videos, when available this information is stored in the following subsection of the $json$ object: $entities \rightarrow media \rightarrow media\_url$. While reporting the most representative and central tweets, we scan the original json format in order to retrieve such a URL, from the most representative tweet or any of its related tweets, pointing to valid photos or pictures in a *.jpg*, *.png* or $.gif$ format. Then, we report these pictures along with the headlines and the set of keywords, as shown in Algorithm 1.

Algorithm 1 Topic detection based on Joint Complexity

```
// N = # timeslots, M = # tweets in the n-th timeslot
for n = 1 to N do
    for t = 1 to M do
        t ← t_json.getText();
        t_ST ← suffixTreeConstruction(t);
        JCScores ← JCMetric();
    end for
    // Find the most representative & central tweets
    S^n ← sum(JCScores);
    // Get headlines for the central tweets
    R^n ← descendingOrder(S^n);
    // Get set of keywords
    K^n ← keywords(R^n);
    // Get URLs of pictures from the .json file
    P^n ← mediaURL(R^n);
    // Print the results in appropriate format
    Print(R^n);
    Print(K^n);
    Print(P^n);
end for
```

Almost half of the headlines (47%) produced by our method had a picture retrieved in the *.json* file. When no such URL is provided within the collected *json* objects we planned to visit the URLs of websites and retrieve images, which were online, in a way that were enforced to be relevant for the topic, but we did not follow that strategy because of the study of the Google API and due to lack of time. It is important to point out that the goal of the challenge was to detect newsworthy items before they hit mainstream news websites, so it was decided that parsing images from such websites was not interesting in that context.

3.9.5 Evaluation of Topic Detection

Apart from the specific implementation for the Snow Data Challenge, the main benefits of our method are that we can both classify the messages and identify the growing trends in real time, without having to manually set up lists of keywords for every language. We can track the information and timelines within a social network and find groups of users which agree on the same topics.

The official evaluation results of our method in the Snow Data Challenge are included in [68]. Although the dataset that was used for this challenge did not allow to show this properly, one key advantage of using Joint Complexity is that it can deal with languages other than English [72, 73] without requiring any additional feature.

3.10 Tweet Classification

3.10.1 Tweet augmentation

Joint Complexity was also used for classification in Twitter [74]. The innovation brought by the method is in the use of the information contained in the redirected URLs of tweets. We use this information to augment the similarity measure of JC, which we call *tweet augmentation*. It must be noted that this method does not have access to the redirected URLs as described above about the prior art existing solution.

The method proceeds in two phases: (3.10.2) Training phase, and (3.10.3) Run phase.

3.10.2 Training Phase

During the *Training phase* we construct the training databases (DBs) by using Twitter's streaming API with filters for specific keywords. For example, if we want to build a class about politics, then we ask the Twitter API for tweets that contain the word "politics". Using these requests we build M classes on different topics. Assume that each class contains N tweets (*eg.* $M = 5$ Classes: politics, economics, sports, technology, lifestyle of $N = 5000$ tweets). To each class we allocate K keywords (*e.g.* the keywords used to populate the class; their set is smaller than the *bag-of-words*). The tweets come in the *.json* format which is the basic format delivered by the Twitter API.

Then we proceed to the URL extraction and tweet augmentation. The body of a tweet (in the *.json* file format), contains a URL information if the original author of the tweet has inserted one. In general Twitter applies a hashing code in order to reduce the link size in the tweets delivered to users (this is called *URL shortening*). However the original URL comes in clear in the *.json* format provided by the Twitter API. While extracting the tweet itself, we get both the hashed URL and the original URL posted by the user. Then, we replace the short URL in the tweet's text by the original URL and we get the augmented tweet.

In the next step, we proceed with the Suffix Tree construction of the augmented tweet. Building a suffix tree is an operation that costs $O(n \log n)$ operations and takes $O(n)$ space in memory, where n is the length of the augmented tweet. The tweet itself does not exceed 140 characters, so the total length of the augmented tweet is typically smaller than 200 characters.

3.10.3 Run Phase

During the *Run phase* (shown in Algorithm 2), we get tweets from the Twitter Streaming Sample API. For every incoming tweet we proceed to its classification by the following operations: At first, we augment the tweet as described in the 3.10.2 Training phase. Then, we compute the matching metric of the augmented incoming tweet with each class. The score metric is of the form:

$$MJC * \alpha + PM * \beta \tag{3.19}$$

where MJC is the max of Joint Complexity (JC) of the augmented incoming tweet over the tweets already present in the class, and PM is the pattern matching score of the incoming tweet over the class keywords. Quantities α and β are weight parameters, which depend on the average Joint Complexity, JC_i^{avg}, of the i-th class, and the maximum JC (best fitted), JC_i^{max}. We construct those as follows:

$$\begin{aligned} \beta &= \frac{JC_i^{max} - JC_i^{avg}}{JC_i^{max}} \\ \alpha &= 1 - \beta \end{aligned}$$

When the average Joint Complexity, $JC_i^{avg} = \frac{JC_i^{max}}{2}$ the weight $\alpha = \beta = 0.5$, and if the pattern matching on the URL returns zero, then $\beta = 0$ and $\alpha = 1$.

The Joint Complexity between two tweets is the number of the common factors defined in language theory and can be computed efficiently in $O(n)$ operations (sublinear on average) by Suffix Tree superposition. We also compute the Pattern Matching score with the keywords of the i-th class *i.e.* as the number of keywords actually present in the augmented tweet URL. The metric is a

combination of the MJC and PM.

We then assign an incoming tweet to the class that maximizes the matching metric defined at (3.19) and we also link it to the best fitted tweet in this class, *i.e.* the tweet that maximises the Joint Complexity inside this class.

In the case described above where newly classified tweets are added to the reference class (which is useful for *trend sensing*), then 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 got the lowest JC score). This ensures the low cost and efficiency of our method.

The main benefits of our method are that we can both classify the messages and identify the growing trends in real time, without having to manually identify lists of keywords for every language. We can track the information and timeline within a social network and find groups of users that agree or have the same interests, *i.e*, perform trend sensing.

3.10.4 Experimental Results on Tweet Classification

The efficiency of the proposed classification method is evaluated on sets of tweets acquired from the Twitter API. The classification accuracy of five tested methods was measured with the standard *Precision*, *Recall*, and *F-score* metrics (detailed in the section below), using a Ground Truth (GT) shown in Fig. 3.16. The experiments were run on more than 1M tweets [68].

Instead of retrieving live tweets from the Twitter Streaming API as described in Algorithm 2, we stored a list of random tweet IDs in a file so that each algorithm would work on the same set of tweets.

We selected the Document-Pivot (DP) method to compare with our new method, since it outperformed most of the other state-of-the-art techniques in a

Algorithm 2 Tweet classification based on Joint Complexity and pattern matching

Training phase:

$constructClasses(M, N)$;
for $i = 1$ *to* M **do**
  for $i = 1$ *to* N **do**
    $t_{i,j}^{URL} \leftarrow extractURL(t_{ij})$;
    $t_{i,j}^{aug} \leftarrow tweetAugmentation(t_{ij}, t_{i,j}^{URL})$;
    $t_{i,j}^{ST} \leftarrow suffixTreeConstruction(t_{ij}^{aug})$;
  end for
end for

Run phase:

while $(t_{json} \leftarrow TwitterAPI.getSample() \; != \; null)$ **do**
  $t \leftarrow t_{json}.getText()$;
  $t_{URL} \leftarrow extractURL(t_{json})$;
  $t_{aug} \leftarrow tweetAugmentation(t, t_{URL})$;
  $t_{ST} \leftarrow suffixTreeConstruction(t_{aug})$;
  for $i = 1$ *to* M **do**
    $PM_i(t) \leftarrow patternMatching(t_{URL})$;
    $JC_i^{avg} \leftarrow averageJC(t_{aug})$;
    $JC_i^{max} \leftarrow maximum(JC(t_{aug}))$;
    $\beta \leftarrow \frac{JC_i^{max} - JC_i^{avg}}{JC_i^{max}}$
    $\alpha \leftarrow 1 - \beta$
  end for
  $D(t) \leftarrow arg\max_{i=1}^{M}\{\max_{j=1}^{N}(JC(t_{i,j}, t) * \alpha) + PM_i * \beta\}$
  $classifyTweet(t, D(t))$;
end while
$UpdateByDeletingOldestTweets()$;

Twitter context as shown in [75].

In order to run the experiment, we modified an existing implementation of Document-Pivot which was developed for topic detection in [75]. The modification consisted in setting up almost the same Training phase as for the Joint

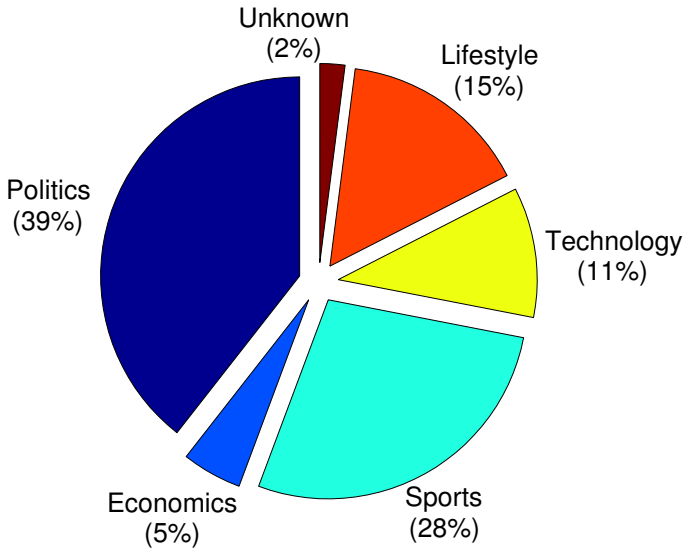

Figure 3.16: Ground Truth distribution of the tweets into the different categories.

Complexity implementation (*i.e.* use the same reference tweets), with a notable exception for the *tweet augmentation*. For the latter implementation, instead of placing the original URL, we first decompose the URL into a list of pseudo-words by replacing the '/' character by a space. This method, named DPurl, will prove useful for the classification as quite often URLs contain specific keywords such as *sport*, *politics*, *etc.*

The other and more important difference with DP (without changing the output of the 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 *closest* tweet 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 1, 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.

3.10.4.1 Classification Performance based on Ground Truth

The classification performance is compared for five methods, which are:

1. Document Pivot (DP), without tweet augmentation,
2. Joint Complexity (JC), without tweet augmentation,
3. Document Pivot with URL (DPurl) described above,
4. Joint Complexity with URL (JCurl) described above, without the Pattern Matching,
5. Joint Complexity with URL and Pattern Matching (JCurlPM) described in Algorithm 2 in Section 3.10.

The standard *Precision*, *Recall*, and *F-score* metrics were used to evaluate the different classification output, they are described below:

$$Precision = \frac{true\ positives}{true\ positives + false\ positives}$$

$$Recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

where, for a class $\mathcal{C}$, *true positives* are tweets that were classified in $\mathcal{C}$ by both the algorithm and the Ground Truth, *false positives* are tweets that were classified in $\mathcal{C}$ by the algorithm but in some other class by the Ground Truth and *false negatives* are tweets that were classified in $\mathcal{C}$ by the Ground Truth but in some other class by the algorithm.

We also computed the *F-score* in order to combine into a single metric both precision and recall (for faster comparison at a glance):

$$F\text{-}score = 2 * \frac{Recall * Precision}{Recall + Precision}$$

A global overview of the results is presented in Table 4.1 where we can see that, on average, JC outperforms DP, JCurl outperforms DPurl, and JCurlPM clearly outperforms them all.

Metric	DP	JC	DPurl	JCurl	JCurlPM
Precision	0.47	**0.60**	0.68	**0.71**	**0.86**
Recall	0.38	**0.48**	0.57	**0.63**	**0.86**
F-score	0.42	**0.53**	0.62	**0.67**	**0.86**

Table 3.2: Average of *precision, recall* and *F-score* for the used classification methods for all classes. The methods DP, JC, Purl, JCurl and JCurlPM are used.

Looking in more details for each category, the global tendency is confirmed except for a couple of categories like *Technology* where DP has a slightly better Precision but a worse Recall. In the *Sports* category on the other hand the situation is reversed as DP seems to provide a slightly better Recall. In both cases the differences are too small to be really significant and what can be noted is that JCurlPM always outperforms all other methods. The mediocre precision obtained by DP and JC in Fig. 3.18 can be explained by the fact that the *Economics* category was under-represented in the Ground Truth dataset and given the fact that *Politics* and *Economics* are often very close subjects, both methods classified a few *Politics* tweets into the *Economics* category thus lowering the Precision. It can be noted that the Recall on the other hand is quite good for both methods.

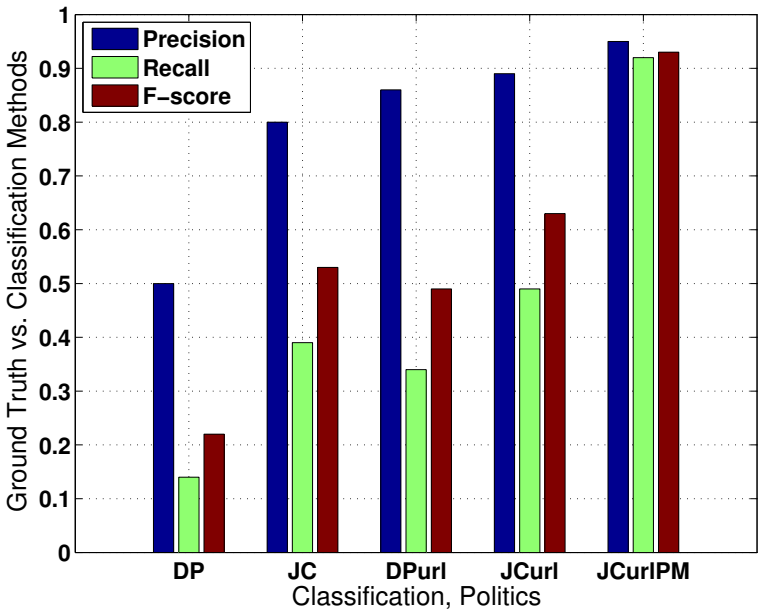

Figure 3.17: Precision (left), recall (middle) and F-score (right) for the classified tweets in the class Politics.

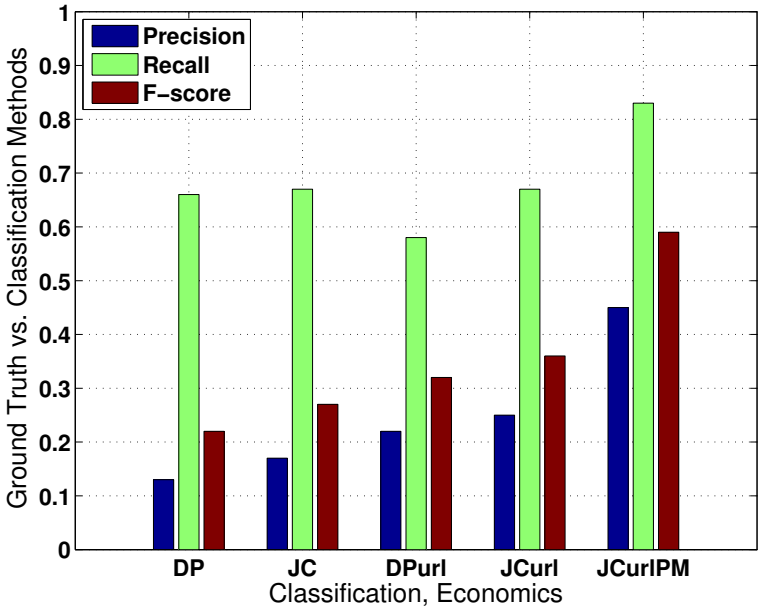

Figure 3.18: Precision (left), recall (middle) and F-score (right) for the classified tweets in the class Economics.

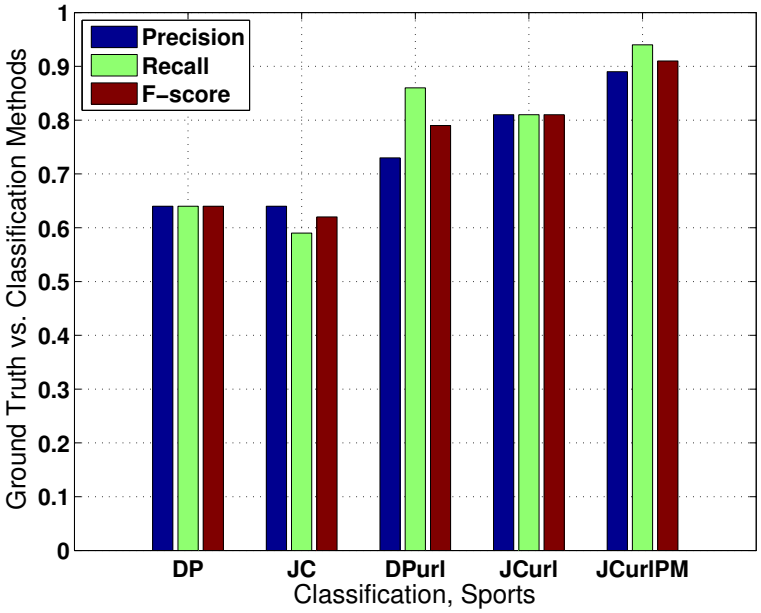

Figure 3.19: Precision (left), recall (middle) and F-score (right) for the classified tweets in the class Sports.

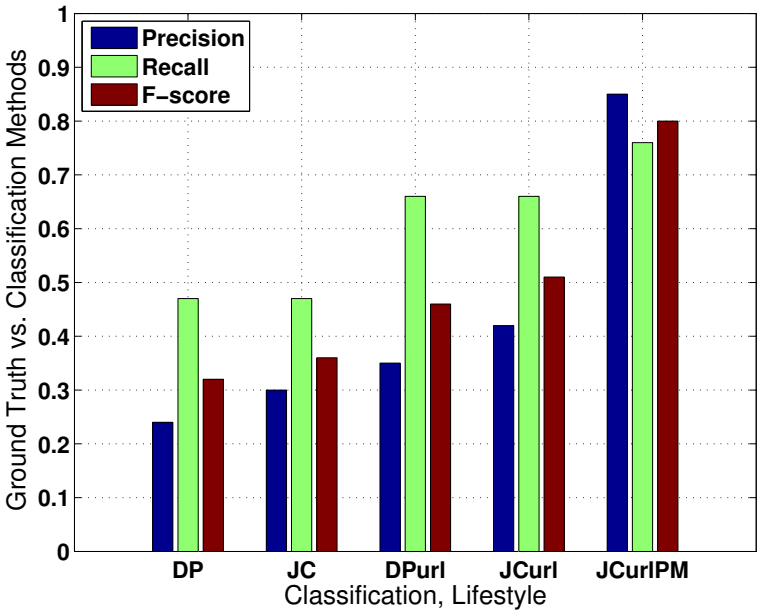

Figure 3.20: Precision (left), recall (middle) and F-score (right) for the classified tweets in the class Lifestyle.

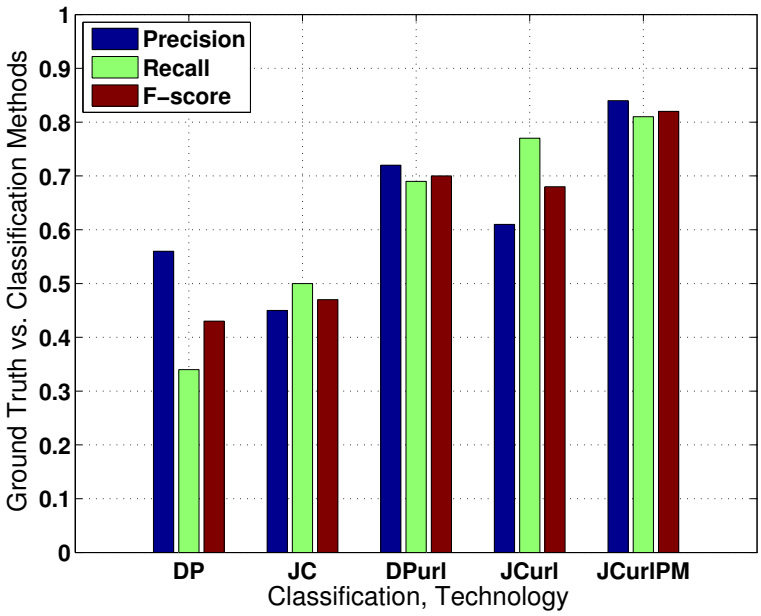

Figure 3.21: Precision (left), recall (middle) and F-score (right) for the classified tweets in the class Technology

3.11 Chapter Summary

In this Chapter we studied the Joint Sequence Complexity and its applications, which range from finding similarities between sequences to source discrimination. Markov models well described the generation of natural text, and we exploited datasets from different natural languages using both short and long sequences. We provided models and notations, and presented the theoretical analysis. A study on expending asymptotics and periodic terms was also mentioned. We applied our methodology to real messages from Twitter in the Snow Data Challenge of the World Wide Web Conference in 2014, where we evaluated our proposed methodology on topic detection, classification and trend sensing in Twitter in real time. Our proposed method based on Joint Complexity was praised by a committee of experts and we won the third Prize.

Chapter 4

Text Classification via Compressive Sensing

Contents

4.1 Introduction

According to Compressive Sensing (CS) theory [16], signals that are sparse or compressible in a suitable transform basis can be recovered from a highly reduced number of incoherent linear random projections, which overcomes the

traditional signal processing methods. Traditional methods are dominated by the well-established Nyquist-Shannon sampling theorem, which requires the sampling rate to be at least twice the maximum bandwidth.

We introduce a hybrid classification and tracking method, which extents our recently introduced Joint Complexity method [72, 73], which was tailored to the topic detection and trend sensing of user's tweets. More specifically, we propose a two-step detection, classification and tracking method:

First we employ the Joint Complexity, already described in detail in the previous Chapter, as the cardinality of a set of all distinct factors of a given string represented by suffix trees, to perform topic detection. Second, based on the nature of the data, we apply the methodology of Compressive Sensing to perform topic classification by recovering an indicator vector. Finally, we combine the Kalman filter, as a refinement step for the update of the tracking process.

4.2 Compressive Sensing Theory

Let us first describe the main theoretical concepts of CS [48, 15, 16] and how it is applied on the problem classification [78]. Consider a discrete-time signal x in $\mathbb{R}^N$. Such signal can be represented as a linear combination of a set of basis $\{\psi_i\}_{i=1}^N$. Constructing a $N \times N$ basis matrix $\mathbf{\Psi} = [\psi_1, \psi_2, \ldots, \psi_N]$, the signal x can be expressed as

$$x = \sum_{i=1}^{N} s_i \psi_i = \mathbf{\Psi} s \tag{4.1}$$

where $s = (s_1, s_2, \dots s_N) \in \mathbb{R}^N$ and is an equivalent representation of x in a basis Ψ.

In fact the signal is represented as

$$x = \mathbf{\Psi} s + \theta \tag{4.2}$$

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 $\mathbf{\Psi} \in \mathbb{R}^{N \times N}$. It has been demonstrated [15, 16] that if $\mathbf{x}$ is K-sparse in $\mathbf{\Psi}$ (meaning that the signal is exactly or approximately represented by K elements of this basis), it can be reconstructed from $M = rK \ll N$ non-adaptive linear projections onto a second measurement basis, which is incoherent with the sparsity basis, and where r is a small overmeasuring factor ($r > 1$).

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. By noting that $\mathbf{x}$ can be expressed in terms of the basis $\mathbf{\Psi}$ as in(4.2) the measurement model has the following equivalent transform-domain representation

$$\mathbf{g} = \mathbf{\Phi \Psi s} + \mathbf{\Phi \theta} \ . \tag{4.3}$$

In fact when the length of the sequence (*i.e.* tweet) $n \to \infty$ and $N \to \infty$, $\mathbb{E}(\mathbf{\Psi s}) = O(nN)$, with $var(\theta) = O(nN)$, $std(\theta) = O(\sqrt{|\Phi|n})$ and $\mathbb{E}(\mathbf{\Phi \theta}) = 0$. The second part of (4.3), $\mathbf{\Phi \theta}$ 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 [16]), are random matrices with independent and identically distributed (i.i.d.) Gaussian or Bernoulli entries. Two matrices $\mathbf{\Psi}$, $\mathbf{\Phi}$ are incoherent if the elements of the first are not represented sparsely by the elements of the second, and vice versa. Since the original vectors of signals, $\mathbf{x}$, are not sparse in general, in the following study we focus on the more general case of reconstructing their equivalent sparse representations, $\mathbf{s}$, given a low-dimensional set of measurements $\mathbf{g}$ and the measurement matrix $\mathbf{\Phi}$.

By employing the M compressive measurements and given the K-sparsity property in basis $\mathbf{\Psi}$, the sparse vector $\mathbf{s}$, and consequently the original signal $\mathbf{x}$, can be recovered perfectly with high probability by taking a number of different approaches. In the case of noiseless CS measurements the sparse vector $\mathbf{s}$ is estimated by solving a constrained ℓ_0-norm optimisation problem of the form,

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \|\mathbf{s}\|_0, \quad s.t. \quad \mathbf{g} = \mathbf{\Phi}\mathbf{\Psi}\mathbf{s} \ , \tag{4.4}$$

where $\|\mathbf{s}\|_0$ denotes the ℓ_0 norm of the vector $\mathbf{s}$, which is defined as the number of its non-zero components. However, it has been proven that this is an NP-complete problem, and the optimization problem can be solved in practice by means of a relaxation process that replaces the ℓ_0 with the ℓ_1 norm,

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \|\mathbf{s}\|_1, \quad s.t. \quad \mathbf{g} = \mathbf{\Phi}\mathbf{\Psi}\mathbf{s} \ . \tag{4.5}$$

which will give s with a relative error of $O(\frac{1}{\sqrt{nN}})$. In [15, 16] it was shown that these two problems are equivalent when certain conditions are satisfied by the two matrices $\mathbf{\Phi}$, $\mathbf{\Psi}$ (restricted isometry property (RIP)).

The objective function and the constraint in (4.5) can be combined into a single objective function, and several of the most commonly used CS reconstruction

methods solve the following problem,

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \Big(\|\mathbf{s}\|_1 + \tau \|\mathbf{g} - (\mathbf{\Phi}\mathbf{\Psi}\mathbf{s})\|_2 \Big) \ , \tag{4.6}$$

where τ is a regularization factor that controls the trade-off between the achieved sparsity (first term in (4.6)) and the reconstruction error (second term). Commonly used algorithms are based on linear programming [79], convex relaxation [15, 80], and greedy strategies (*e.g.,* Orthogonal Matching Pursuit (OMP) [81, 82]).

4.3 Compressive Sensing Classification

4.3.1 Training Phase

During the training phase, we built our classes as described in Section 3.3 and for each class we extract the most central/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 $\mathbf{\Psi}_T^i$. In the proposed algorithm, a standard Gaussian measurement matrix is employed, with its columns being normalized to unit ℓ_2 norm.

Figure 4.2 shows a flowchart of the preprocessing phase classification based on Compressive Sensing in conjunction with the part of Joint Complexity shown in the flowchart of Figure 4.1.

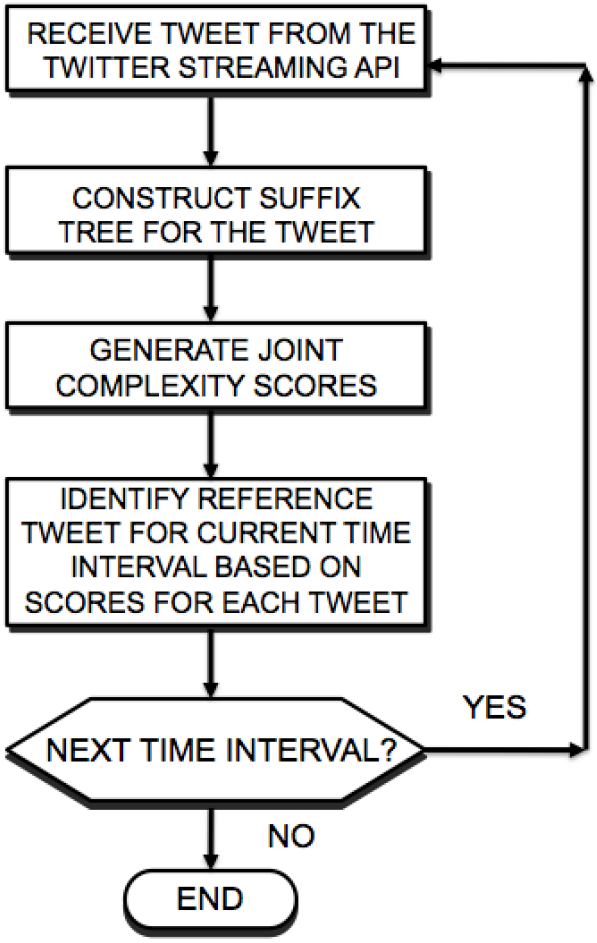

Figure 4.1: Flowchart of the preprocessing phase of Joint Complexity.

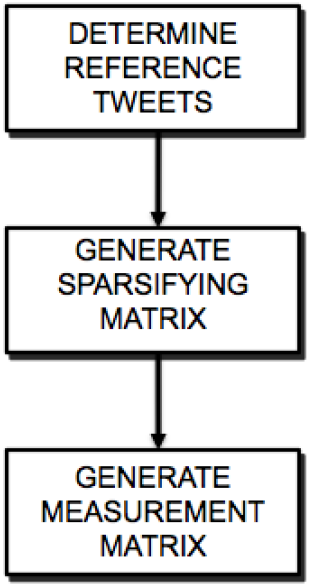

Figure 4.2: Flowchart of the preprocessing phase of Compressive Sensing.

4.3.2 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

$$\mathbf{g}_c = \mathbf{\Phi}_R \mathbf{x}_{c,R} \ , \tag{4.7}$$

where $\mathbf{\Phi}_R$ denotes the corresponding measurement matrix during the runtime phase.

The measurement vector $\mathbf{g}_c$ is formed for each CT_i according to (4.7) and the reconstruction takes place via the solution of (4.6), with the training matrix $\mathbf{\Psi}_T$ being used as the appropriate sparsifying dictionary.

Figure 4.3 shows the flowchart for the runtime phase of the classification

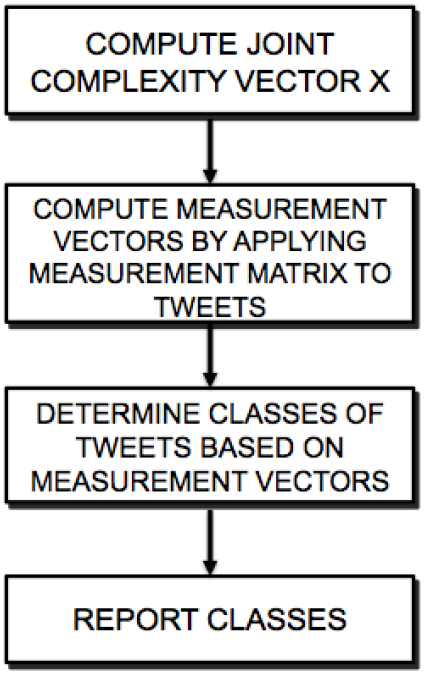

Figure 4.3: Flowchart of the runtime phase of the Compressive Sensing based classification.

based on Compressive Sensing.

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 performs the core CS algorithm. The performance analysis, described in Section 4.5, reveals an increased accuracy of the proposed CS-based classification algorithm when compared with other methods described in Section 2.4 and 4.5.

4.4 Tracking via Kalman Filter

Most of the tracking methods use past state estimates and motion dynamics to refine the current state estimate determined by the above topic detection and classification methods. In addition, the dynamic motion model can also be used

Algorithm 3 Tweet classification based on JC, CS and Kalman

Training phase:

// Build Classes according to JC
$constructClasses(M, N)$;

Run phase:

while $(t_{json} \leftarrow TwitterAPI.getSample() \; != \; null)$ **do**
// Classify t by running the CS classification algorithm
Give the estimated class as input to the Kalman filter
end while
Update by deleting oldest tweets

in conjunction with the current state estimate to predict the future possible states.

We are based on the assumption that the Compressive Sensing based classification method involves the mobile device that collects the tweets from the Twitter API and performs the core Joint Complexity and Compressive Sensing algorithm.

If we had a model of Joint Complexity to detect the change of topics, we could use the Kalman filter to track a user according to his tweets. In this work, we assume that the change of topics is uniform.

Kalman filtering is a well-established method for estimating and tracking mobile targets. A typical Kalman filter [49] is applied recursively on a given dataset in two phases: i) *Prediction* and ii) *Update*. The main advantage of this algorithm is that it can be executed in real time, since it is only based on the currently available information and the previously estimated state.

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, 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

$$\mathbf{x}(t) = \mathbf{F}\mathbf{x}(t-1) + \boldsymbol{\theta}(t) \tag{4.8}$$

$$\mathbf{z}(t) = \mathbf{H}\mathbf{x}(t) + \mathbf{v}(t) \tag{4.9}$$

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

$$\mathbf{x}^{*-}(t) = \mathbf{F}\mathbf{x}^*(t-1) \tag{4.10}$$

$$\mathbf{P}^-(t) = \mathbf{F}\mathbf{P}(t-1)\mathbf{F}^\mathbf{T} + \mathbf{S} \tag{4.11}$$

$$\mathbf{K}(t) = \mathbf{P}^-(t)\mathbf{H}^T(\mathbf{H}\mathbf{P}^-(t)\mathbf{H}^T + \mathbf{U})^{-1} \tag{4.12}$$

$$\mathbf{x}^*(t) = \mathbf{x}^{*-}(t) + \mathbf{K}(t)(\mathbf{z}(t) - \mathbf{H}\mathbf{x}^{*-}(t)) \tag{4.13}$$

$$\mathbf{P}(t) = (\mathbf{I} - \mathbf{K}(t)\mathbf{H})\mathbf{P}^-(t) \tag{4.14}$$

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 [83], described briefly in Section 4.3, 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 (4.10). 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 (4.3), which are obtained via a simple matrix-vector multiplication with the original high-dimensional vector. Given the limited memory and bandwidth capabilities of a small mobile device, the proposed approach can be an effective candidate to achieve accurate information propagation, while increasing the device's lifetime. Since $M \ll N$ we have a great complexity improvement given by Compressive Sensing, which reduces the overall complexity of the Kalman filter. Algorithm 3 shows the combination of JC and CS method in conjunction with the Kalman filter, and summarizes the proposed information propagation system. Finally, Figure 4.4 presents the flowchart for generating a tracking model and predicting classes of tweets.

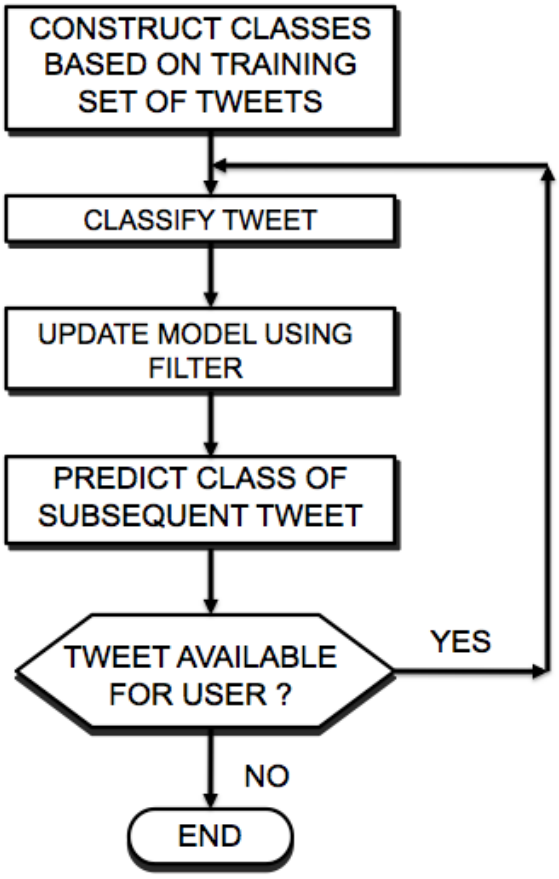

Figure 4.4: Flowchart of the classification and tracking model based on Kalman filter and Compressive Sensing.

4.5 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 *F-score* metrics, using a Ground Truth (GT) on more than 1M tweets [68].

The GT was computed by averaging the values returned by users and 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 4.3.

We selected the Document-Pivot (DP) method to compare with our new method, since it outperformed most of the other state-of-the-art techniques in a Twitter context as shown in [75]. 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 1, 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.

4.5.1 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 and extracted from the *.json* file.

An overview of the results is presented in Table 4.1 where we can see that, on average, JC with CS outperforms DP, and JCurl with CS outperforms DPurl.

Table 4.1: Precision, recall and F-score for the used classification methods for all classes.

Metric	DP	JC+CS	DPurl	JCurl+CS
Precision	0.47	**0.78**	0.68	**0.89**
Recall	0.38	**0.56**	0.57	**0.81**
F-score	0.42	**0.65**	0.62	**0.85**

Fig. 4.5 compares the classification accuracy 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. 4.6 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) BCS-GSM [84]. Fig. 4.6 shows that BCS and BCS-GSM outperform the introduced reconstruction techniques, while Fig. 4.7 shows that we a achieve better performance of 10% when using the Kalman filter.

[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

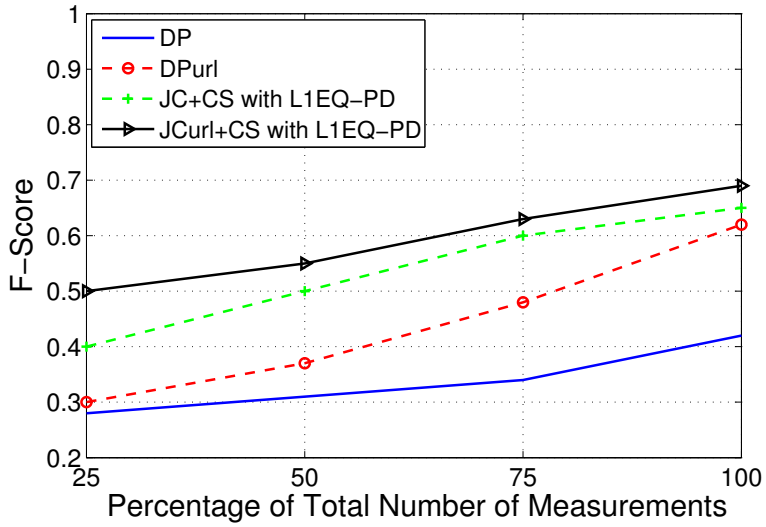

Figure 4.5: Classification accuracy measured by F-Score for the DP, DPurl and JC+CS, JCurl+CS method as a function of the number of measurements (%) by using the ℓ_1-norm minimization.

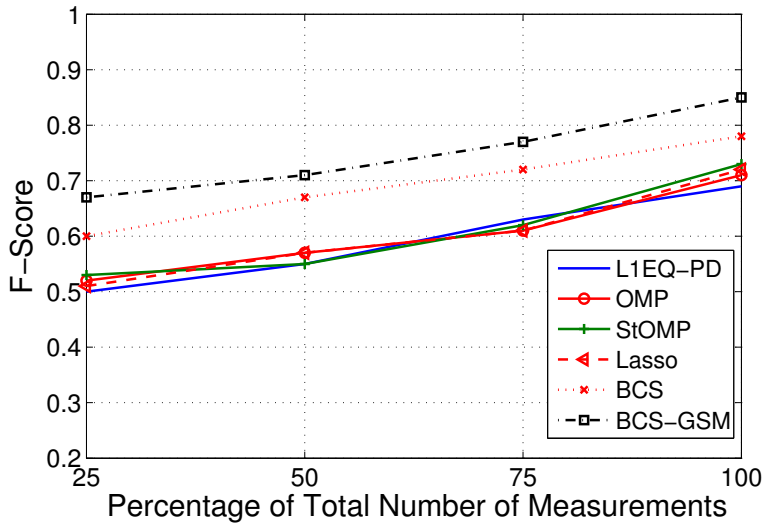

Figure 4.6: Classification accuracy measured by F-Score as a function of the number of measurements (%) by using several reconstruction techniques, for the JCurl+CS method.

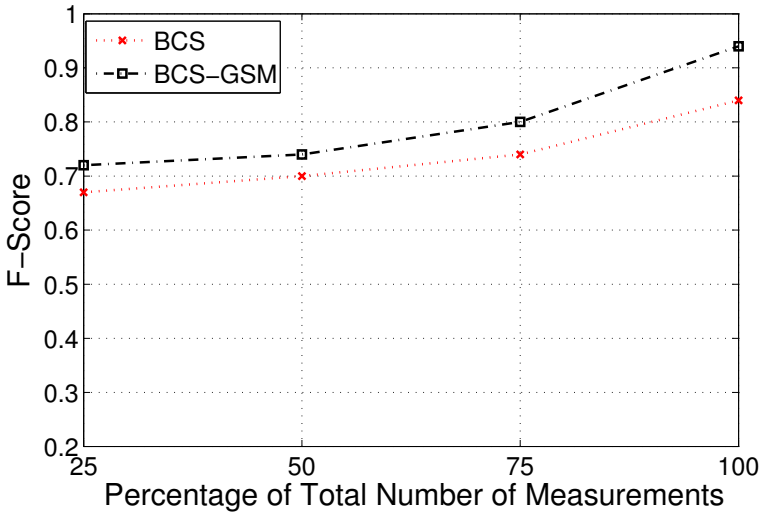

Figure 4.7: Classification accuracy measured by F-Score as a function of the number of measurements (%) by using Kalman, for the JCurl+CS method, with the BCS and BCS-GSM reconstruction techniques.

4.6 Chapter Summary

In this Chapter an information propagation method was introduced. First, 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 with a large set of tweets revealed a better performance, when compared with previous state-of-the-art methods, while being language-agnostic, without any need of grammar, dictionaries or semantics.

Chapter 5

Extension of Joint Complexity and Compressive Sensing

Contents

5.1 Introduction

During this thesis, the theory of Joint Complexity and Compressive Sensing has been extended to three research subjects, (a) localization and path tracking in indoor environments, (b) encryption based on compressive measurement vectors, and (c) encryption based on the Eulerian circuits of original texts.

5.2 Indoor Path-Tracking Using Compressive RSS Measurements

In this work, a hybrid path-tracking system is introduced, which exploits the power of compressive sensing (CS) to recover accurately sparse signals, in conjunction with the efficiency of a Kalman filter to update the states of a dynamical system. The proposed method first employs a hierarchical region-based approach to constrain the area of interest, by modeling the signal-strength values received from a set of wireless access points using the statistics of multivariate Gaussian models. Then, based on the inherent spatial sparsity of indoor localization, CS is applied as a refinement of the estimated position by recovering an appropriate sparse position-indicator vector. The experimental evaluation with real data reveals that the proposed approach achieves increased localization accuracy when compared with previous methods, while maintaining

a low computational complexity, thus, satisfying the constraints of mobile devices with limited resources.

During the last decade, location estimation and navigation systems emerged as important areas of research in the fields of pervasive and mobile computing. Transportation, security, entertainment, and medicine are just a few examples where accurate location estimation is a key ingredient. Focusing on the problem of *indoor localization*, numerous solutions have been proposed based on distinct technologies, such as IEEE802.11 [85], infrared [86], ultrasonic [87], bluetooth [88], or even a combination of optical, acoustic, and received signal-strength (RSS) information along with motion attributes [89]. In an other work [90], the RSS was used for service discovery in mobile ad hoc networks.

Based on the wide deployment of wireless local area networks (WLANs) using IEEE802.11 infrastructures, most indoor positioning systems employ the RSS values obtained directly from a set of access points (APs) by any mobile device which is connected to the network. However, the nature and structure of indoor environments pose significant challenges, since phenomena, such as shadowing and multipath fading, result in radio channel obstructions and variations of the RSS values. This makes the design of accurate positioning systems a difficult and challenging task.

On the other hand, the inherent spatial sparsity, which characterizes a location estimation problem, motivates naturally the use of the novel mathematical framework of *compressive sensing* (CS) [16]. CS states that signals that are sparse or compressible in a suitable transform basis can be recovered from a

highly reduced number of incoherent linear random projections.

Motivated by the need to locate and track accurately a mobile user who holds a device with potentially limited power, processing, and bandwidth resources, in this work we introduce a hybrid path-tracking method, which extents our recently introduced positioning approach [76, 77], which was tailored to the localization of static users. More specifically, we propose a two-step path-tracking method: First, we employ a region-based multivariate Gaussian model to restrict the search space of candidate cells; then, for each region, we perform CS reconstruction of an appropriate sparse position-indicator vector, combined with a Kalman filter, as a refinement step for the update of the mobile user's estimated position.

The rest of the work is organized as follows: Section 5.2.1 overviews the current state-of-the-art on indoor path-tracking methods, while Section 5.2.2 describes in brief our recent CS-based localization method for static users, introduced in [76]. Section 4.4 analyzes in detail the proposed algorithm for tracking the location of a mobile user in an indoor environment, while Section 5.2.4 evaluates experimentally and compares the performance of our approach with previous state-of-the-art localization methods.

5.2.1 Prior Work on RSS-based Path Tracking

RSS-based location estimation methods can be classified roughly in two categories, namely, the fingerprint- and prediction-based ones. Fingerprint-based methods consist of two individual phases, that is, the *training* and the *runtime* phase. During the training phase, a wireless device records the RSS values at

known predefined positions on a discretized grid partition of the physical space and constructs the training signature map [85, 91]. During the runtime phase, the system also records the RSS values at an unknown position to construct a runtime signature, which is then compared with the training signature map to estimate the user's location.

On the other hand, prediction-based techniques use the RSS values and radio propagation models to predict the distance of a wireless device from an AP [92]. The main challenge of these techniques is the difficulty to formulate a reliable radio propagation model due to multipath fading, shadowing, floor layout, and moving objects.

In the following, we focus on fingerprint-based localization techniques. Current state-of-the-art methods are reviewed in brief, which were shown to be efficient in several indoor environments, and with which we compare the performance of the proposed path-tracking architecture.

A common approach in location estimation problems is the use of the k-Nearest Neighbor algorithm (kNN) [93], where an RSS map is constructed by averaging separately the RSS values received from each AP. It computes a signature vector of the unknown runtime cell and a signature vector of the cell extracted during the training phase. Then, the algorithm calculates the Euclidean distance between the runtime and all the training cells, and reports the k closest neighbors by sorting the distances in increasing order. The final estimated position is given by computing the centroid of these k closest neighbors.

In [94], the problem of indoor path tracking is also treated in a probabilistic framework. In particular, a reduced number of locations is sampled to construct a radio map, in conjunction with an interpolation method, which is developed to patch effectively the radio map. Furthermore, a Hidden Markov Model (HMM)

that exploits the user traces to compensate for the loss of accuracy is employed to achieve further improvement of the radio map due to motion constraints, which could confine possible location changes. The HMM-based localization technique requires several iterations to converge, while in each iteration several model parameters have to be estimated. The major benefit of our proposed algorithm, when compared with the HMM-based approach, is the significantly reduced computational complexity and implementation simplicity, as well as the high accuracy in several specific environments (obstacle-free, robust measurements) as it was revealed by the experimental evaluation. On the other hand, the HMM-based approach can be proven to be more robust in case of system failures, but at the cost of requiring increased computational resources.

In another work introduced by Guvenc *et al.* [95], the Kalman filter is used without considering the time complexity of the algorithm, especially in case of runtime performance, which introduces large delays in estimating the location. This is also a major drawback of the path-tracking methods proposed in [96, 97].

In a recent work [98], Au *et al.* introduced a tracking system analyzed in two stages. During the coarse localization stage the appropriate cells are chosen, while during the second stage the Kalman filter is used to refine the location estimates. In particular, the localization algorithm is carried out on the mobile device by using the average RSS values in order to construct the transform basis. Our proposed work differs from the previous one in several aspects, from the way we acquire the compressed set of measurements to the way we perform the location estimation. For instance, in contrast to [98], where the estimation is performed onboard by the wireless device with the potentially limited resources, in our system the computational burden is put on the server, where increased storage and processing resources are available. Besides, in the proposed localiza-

tion technique the CS approach is applied directly on the raw RSS measurements and not on their average as in [98], and thus exploiting their time-varying behavior. Moreover, in [98], the lack of a random measurement matrix required when working in the framework of CS may decrease the system's performance under unpredictable environmental conditions, while also the communication of the projected measurements from the wireless device to the APs, where the localization takes place, could pose undesired security issues. In our work, there is no insight of the physical space during runtime experiments where in [98] a map information of the area is provided.

Except for the Kalman filter, particle filters [99] have been also very popular in the design of positioning systems. However, the main disadvantage of a particle filter lies in its high computational cost [101]. For instance, for an indoor space of 70 m^2 we need almost 5000 particles for each filter update. This is against the power constraints of mobile devices, such as, cell phones, making particle filters unsuitable for indoor localization in case of lightweight mobile devices with limited resources.

In [102], a localization via random field differentiation is applied in order to track a continuous trajectory between sampling points. Our approach is complementary to this, since the field differentiation is used as a refinement after the CS algorithm identifies the best candidate cell as if the tracking node were static. The field differentiation uses the variation in the random field and is more appropriate to track motion trajectory between cells.

One of the main advantages of our proposed approach, is that it succeeds to run in real time with a significantly reduced computational complexity, and thus, satisfying the constraints of devices with limited power, memory, and bandwidth resources, which was not addressed completely in these earlier studies. Moreover,

in a recent work [100] we exploited the spatial correlation structure of the fingerprints and used the framework of Matrix Completion to build complete training maps from a small number of random sample fingerprints.

5.2.2 CS-based Location Estimation

In this section, we review briefly the main concepts of our previous work on localization of static users, based on the statistical modeling of the RSS values using multivariate Gaussian (MvG) models [103, 104], in conjunction with a spatial sparsity assumption exploited in the framework of CS [105].

5.2.2.1 Statistical Modeling of RSS Values using MvG Models

We start by considering that the physical space is discretized as a grid consisting of cells with known coordinates. Then, during the training phase, a statistical signature is extracted for each cell by modeling the RSS values received from a set of APs using a multivariate Gaussian (MvG) distribution. During the runtime phase, a statistical signature is generated at the unknown position in a similar way, which is then compared with the training signatures by means of a statistical similarity measure, namely, the *Kullback-Leibler divergence* (KLD). The estimated location is found by minimizing the following KLD ($D(\cdot\|\cdot)$) between two MvGs,

$$j^* = \arg \min_{j=1,\ldots,C} D(f_R||f_{j,T}) \ , \tag{5.1}$$

where C is the number of cells in the grid representing the physical space, f_R denotes the estimated MvG for the runtime cell, and $f_{j,T}$ is the estimated MvG for the j-th training cell.

A hierarchical region-based approach [103] is applied as an initial step to

restrict the space of candidate cells, as follows: First, the space is divided into regions (groups of cells) and then, the process is repeated iteratively by dividing the selected region into sub-regions and applying the algorithm on them, until we end up with the final estimated cell. This process reduces the likelihood of selecting a single false region/cell over the correct one. The closest region is found by minimizing the following KLD between two MvGs,

$$s^* = \arg \min_{s=1,\ldots,S} D(f_R||G_{s,T}) \ , \tag{5.2}$$

where S is the number of regions and $G_{s,T}$ denotes the MvG whose parameters are estimated from the RSS values over all cells of the s-th region during the training phase.

5.2.2.2 Exploiting Inherent Spatial sparsity using CS

The spatial sparsity, which is inherent in a localization problem, motivated us to extend our previous statistical-based localization scheme in the framework of CS [105]. More specifically, the problem of estimating the unknown position is reduced to a problem of recovering a sparse position-indicator vector, with all of its components being zero except for the component corresponding to the unknown cell where the user is placed.

Let $\mathbf{\Psi} \in \mathbb{R}^{P \times N}$ $(P \geq N)$ be a matrix whose columns correspond to a, possibly overcomplete, transform basis. In terms of signal approximation it has been demonstrated [16] that if a signal $\mathbf{x} \in \mathbb{R}^N$ is K-sparse in basis $\mathbf{\Psi}$ (meaning that the signal is exactly or approximately represented by K elements of this basis), then it can be reconstructed from $M = rK \ll N$ non-adaptive linear projections onto a second measurement basis, which is incoherent with the sparsity basis,

and where $r > 1$ is a small overmeasuring constant.

The measurement model generating these projections in the original space-domain is denoted as $\mathbf{g}$, described more extensively in Chapter 4.2, where $\mathbf{g} \in \mathbb{R}^M$ is the vector of compressed measurements, $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ denotes the measurement matrix, and $\mathbf{w}$ is the sparse vector of transform coefficients.

In our indoor positioning application, the training measurement model associated with the $i-$th AP is given by

$$\mathbf{g_i} = \boldsymbol{\Phi}_{\boldsymbol{T}}^{\boldsymbol{i}} \boldsymbol{\Psi}_{\boldsymbol{T}}^{\boldsymbol{i}} \mathbf{w}, \tag{5.3}$$

and the runtime measurement model for cell c is expressed as

$$\mathbf{g_{c,i}} = \boldsymbol{\Phi}_{\boldsymbol{R}}^{\boldsymbol{i}} \boldsymbol{\psi}_{\boldsymbol{R,c}}^{\boldsymbol{i}} \ , \tag{5.4}$$

where the subscripts T and R are used to denote the variables (matrices and vectors) generated in the training and runtime phase, respectively, and $\boldsymbol{\psi}_{\boldsymbol{R,c}}^{\boldsymbol{i}}$ is the vector of runtime RSS values collected at cell c from AP i.

For the localization problem, let $\mathbf{w} = [0 \ 0 \ \cdots \ 0 \ 1 \ 0 \ \cdots \ 0]^T \in \mathbb{R}^C$ be a *position-indicator vector* whose j-th component is equal to "1" if the mobile device is located in the j-th cell. The inherent sparsity in the problem of location estimation comes from the fact that the device to be localized can be placed in exactly one of these cells. Thus, in the framework of CS, the problem of localization is reduced to a problem of recovering the 1-sparse vector $\mathbf{w}$.

5.2.3 CS-Kalman Filter

Kalman filtering as described in Chapter 4.4 is used to estimate and track the position of mobile targets. Focusing on the problem of indoor localization, the

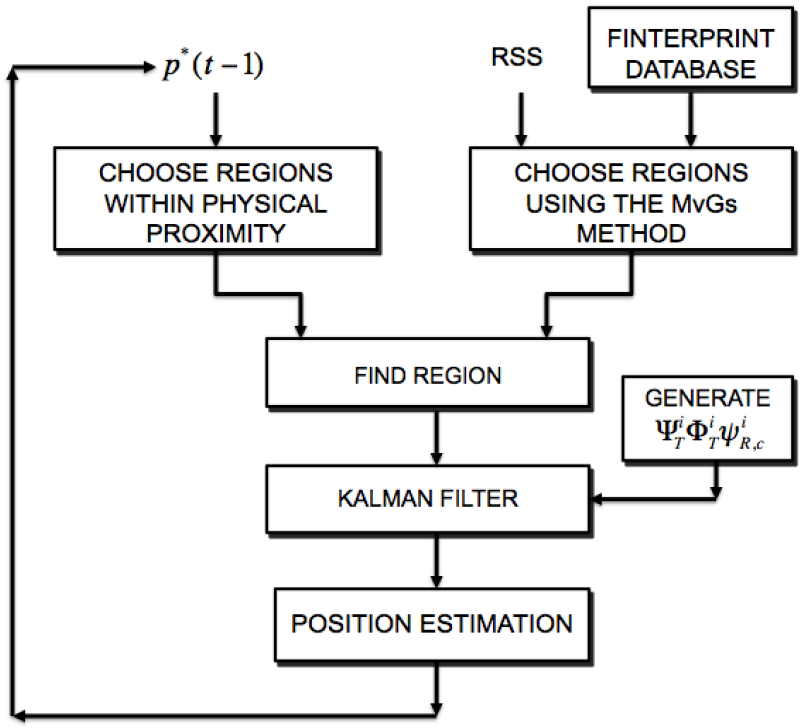

Figure 5.1: Flow diagram of the proposed path-tracking system.

device collects periodically the RSS values from each AP at a specific time interval Δt. Then, the indoor tracking system estimates the user's position at time t, which is denoted by $p^*(t) = [x^*(t), y^*(t)]^T$. Following a Kalman filtering approach, 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 tracking model are described in Section 4.4.

The proposed CS-Kalman tracking system exploits not only the highly reduced set of compressed RSS measurements, but also the previous user's position estimate to restrict the set of candidate training regions based on physical proximity. The Kalman filter is applied on the CS-based positioning system [106, 107, 108], described briefly in Section 5.2.2.2, to improve the estimation accuracy of the mobile user's path. More specifically, let $\mathbf{w}^*$ be the reconstructed

position-indicator vector. Of course in practice $\mathbf{w}^*$ will be not truly sparse, thus the current estimated position $[x_{CS}, y_{CS}]$, or equivalently, cell c_{CS}, corresponds to the highest-amplitude index of $\mathbf{w}^*$. 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, $\mathbf{x}^*(t-1) = [x_{CS}, y_{CS}, v_x(t-1), v_y(t-1)]^T$, and the current position is updated using (4.10). 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 RSS measurements $\mathbf{g}$, which are obtained via a simple matrix-vector multiplication with the original high-dimensional RSS vector. Given the limited memory and bandwidth capabilities of a small mobile device, the proposed approach can be an effective candidate to achieve accurate location estimation, while increasing the device's lifetime. Fig. 5.1 summarizes the proposed indoor path-tracking system, while Algorithm 4 summarizes the main steps of the proposed approach in a pseudocode form.

5.2.4 Experimental Results

The efficiency of the proposed tracking system is evaluated on sets of real data acquired in INRIA at Rocquencourt campus[1] and Bell Labs, in Murray Hill, NJ[2]. The estimation accuracy of the tested methods is evaluated in terms of the localization error, which is defined as the Euclidean distance between the centers of the estimated and the true cell at time t, where the mobile user is located at runtime.

[1]The data were collected at Hipercom team at INRIA, for which it is highly acknowledged.

[2]P. Mirowski and the Statistics and Learning Research department of Bell Labs in Murray Hill, NJ, are highly acknowledged for sharing the dataset.

Algorithm 4 Proposed CS-Kalman tracking algorithm

1. During training phase: collect RSS measurements at each cell from all APs.
2. During runtime phase: collect RSS measurements at the unknown cell from each AP.
3. During runtime phase: execute the following steps for each cell c:
   - Given the previously estimated position, extract a set of candidate regions E_{prox} within physical proximity.
   - Import the runtime RSS values and the training map to the MvG-based method to get a set of regions E_{MvG} (Section 5.2.2.1).
   - The intersection, $E_I = E_{MvG} \cap E_{prox}$, is the common region of interest.
   - Use the CS-based algorithm (Section 5.2.2.2) by generating $\boldsymbol{\Phi}_{\boldsymbol{R}}^{\boldsymbol{i}}$, $\boldsymbol{\Psi}_{\boldsymbol{T}}^{\boldsymbol{i}}$ and $\boldsymbol{\psi}_{\boldsymbol{R,c}}^{\boldsymbol{i}}$ for AP i.
   - CS reconstruction of the position-indicator vector is performed in region E_I, estimated position (cell) is reported and given as input to the Kalman filter.
4. The estimated position (cell) is given as an input to the system in order to obtain new regions within physical proximity.

5.2.4.1 Evaluation in INRIA, Paris

A detailed description of the physical space can be found in the experimentations section of [102]. The wireless coverage is achieved by using an infrastructure consisting of five IEEE802.11 APs. The physical space is discretized in cells of equal dimensions 0.76 m $\times$ 0.76 m, while the RSS map consists of measurements from different cells and for an average number of five APs per cell.

The reconstruction performance is compared for two widely-used CS algorithms, thus working with the much lower-dimensional compressed RSS measurements, as well as with methods working directly with the original full-dimensional RSS vectors. In the *CS domain* we employ and test: 1) ℓ_1-norm minimization

using the primal-dual interior point method (L1EQ-PD)[3]and 2) BCS-GSM [84]. In the *original RSS domain* we evaluate: 3) a kNN-based approached [93], 4) our previous method based on MvGs [103], 5) a typical Kalman filter, and 6) a method employing a particle filter.

Fig. 5.2 shows the cumulative distribution functions (CDFs, $P(X < x)$) of the localization error of the kNN and MvG fingerprint-based methods working in the original RSS domain, together with the CDFs corresponding to the L1EQ-PD and BCS-GSM implementations of the proposed CS-Kalman approach. As it can be seen, the CS-based methods obtain an improved position estimation accuracy compared to standard fingerprint-based methods achieving median errors of 1.71 m (L1EQ-PD) and 1.36 m (BCS-GSM), as opposed to a median error of 1.90 m for the kNN and 1.69 m for the MvG. We emphasize that in this experimental setup the compression ratio r (ref. Section 5.2.2.2) of the runtime RSS measurements vectors employed by the CS-Kalman methods is equal to $r = \frac{M}{N} = 0.25$. In other words, the CS-based approach achieves better positioning results with a significantly reduced amount of data by exploiting the inherent spatial sparsity property of the localization problem.

Fig. 5.3 compares the localization error of the proposed CS-Kalman filter approach using BCS-GSM to solve the sparse reconstruction problem, with the typical Kalman and particle filters. Again, we observe that our proposed approach achieves a higher estimation accuracy with a significantly reduced computational complexity, when compared to the Kalman and particle filters.

[3] http://sparselab.stanford.edu/

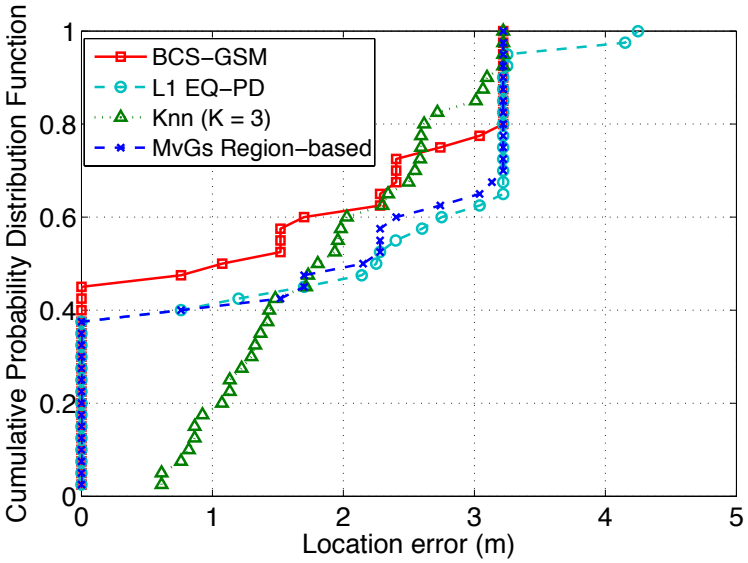

Figure 5.2: Performance evaluation of the proposed path-tracking method (L1EQ-PD and BCS-GSM), compared with methods based on kNN and MvGs.

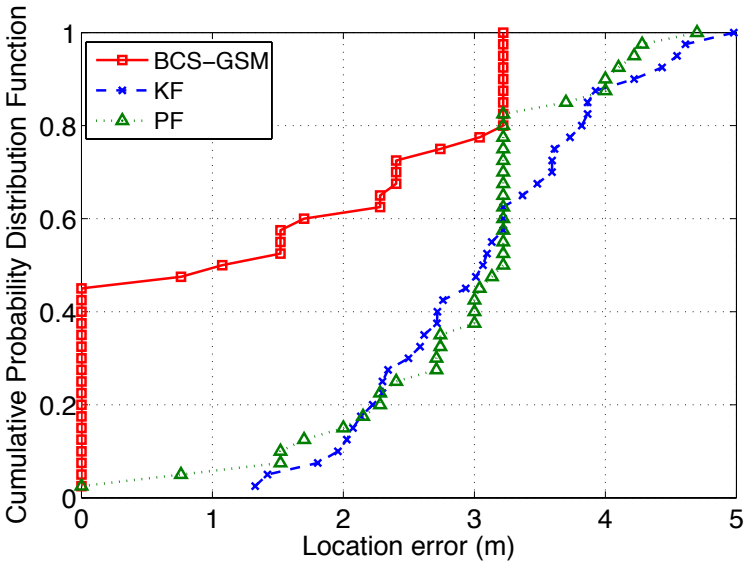

Figure 5.3: Performance evaluation of the proposed path-tracking method (BCS-GSM), compared with the typical Kalman and particle filters.

5.2.4.2 Evaluation in Bell Labs, Murray Hill, NJ

Table 5.1 illustrates three different trajectories acquired both in an office space, in a large corridor with a high, slanted ceiling and in the 5-story atrium. The signal-

strength was captured by a robot which covered an area of about 40 m $\times$ 50 m, on a single floor, with an installed mobile phone on it. Every channel of an AP is considered as a different AP, and we have 72 channels. One of the trajectories, with multiple small loops, is used for defining the fingerprints. Three other trajectories (so–called Track 1, Track 2 and Track 3) were acquired later the same day, towards the end of the business day when there were occasional people moving around the robot, which was a challenge. In order to define fingerprint cells, we simply subdivide the trajectory of the robot according to a regular grid defined using the building coordinates, which was 1 m$\times$1 m. In order to assess the impact of the fingerprint grid size and the trade-off between a denser fingerprint grid (with fewer RF samples) and a coarser fingerprint grid (with more RF samples in each cell), we have explored various resolutions for the fingerprint grid size in the Bell Labs atrium dataset: 1 m, 2 m, 3 m, 4 m, 5 m, 7.5 m and 10 m. We expect that richer fingerprints containing more received signal strength (RSSI) samples per cell would enable better discrimination between RSSI distributions in different fingerprint cells.

The MvG algorithm takes advantage of correlations of the RSSI at certain positions from various APs. Due to the non-visibility of many APs (some of them are not frequently heard) in the positions during training and runtime, the MvG algorithm computes only the correlation between the common APs which are visible in both training and runtime cell. The proposed framework in Section 5.2.2 performs better in general, with best results for grid size of 2 m $\times$ 2 m.

Table 5.1: Results in Bell Labs, Murray Hill, NJ

Grid size (m)	1	2	3	4	5	7.5	10
Track 1 MvG	2.09	1.85	2.11	3.10	3.78	3.87	4.32
CS	1.73	1.57	1.92	2.23	2.32	3.30	3.80
Track 2 MvG	1.81	2.04	2.60	3.31	3.89	4.06	4.86
CS	1.33	1.23	1.46	2.61	2.45	2.59	3.74
Track 3 MvG	2.12	2.42	2.59	3.11	3.28	3.77	4.09
CS	1.92	1.61	1.63	2.51	2.79	3.01	3.47

5.3 Encryption System based on Compressive Sensing Measurements

In this work, the inherent property of CS acting as a weak encryption module combined with an extra *key* (the combination of number of false measurement vectors with the correct one) is exploited to guarantee with high probability that the communication between the device and the server is secured against potential intrusions of an unauthorized entity.

CS-based encryption [109] provides both signal compression and encryption guarantees, without the additional computational cost of a separate encryption protocol and thus it could be useful in location estimation, where the implementation of an additional software layer for cryptography could be costly.

5.3.1 SecLoc System Description

The method consists of two parts: (5.3.1.1) Privacy system, and (5.3.1.2) Key description.

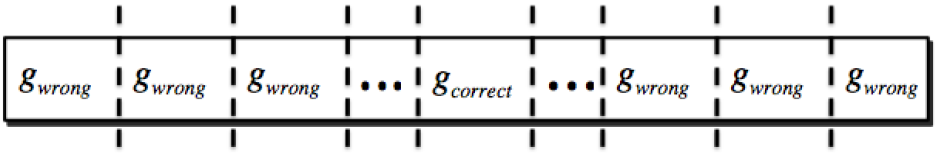

Figure 5.4: N-1 false vectors plus the correct one. This *key*, i.e., the sequence of the measurement vectors reaches the server.

5.3.1.1 Privacy System

Due to their acquisition process, CS measurements can be viewed as *weakly encrypted* for a malicious user without knowledge of the random matrices Φ^i, where i corresponds to the i-th AP, which have independent and identically distributed (i.i.d.) Gaussian or Bernoulli entries [16].

The encryption property of a CS approach relies on the fact that the matrix Φ is unknown to an unauthorized entity, since Φ can be generated using a (time-varying) cryptographic key that only the device and the server share.

More specifically, the server extracts the runtime sub-matrix Φ_R^i from the training Φ_T^i (since the runtime phase lasts significantly less time than training). The lines of Φ_R^i are permuted and the *key* of the merge of the false measurement vectors and the correct one is used, as shown in Fig. 5.5 and described extensively in [76].

5.3.1.2 Key Description

The wireless device sends the measurement vector $\mathbf{g}$ to the server along with $N-1$ false vectors, where the reconstruction takes place. Then, the server uses the information of the physical topology, AP characteristics, previous position, etc., and estimates the location.

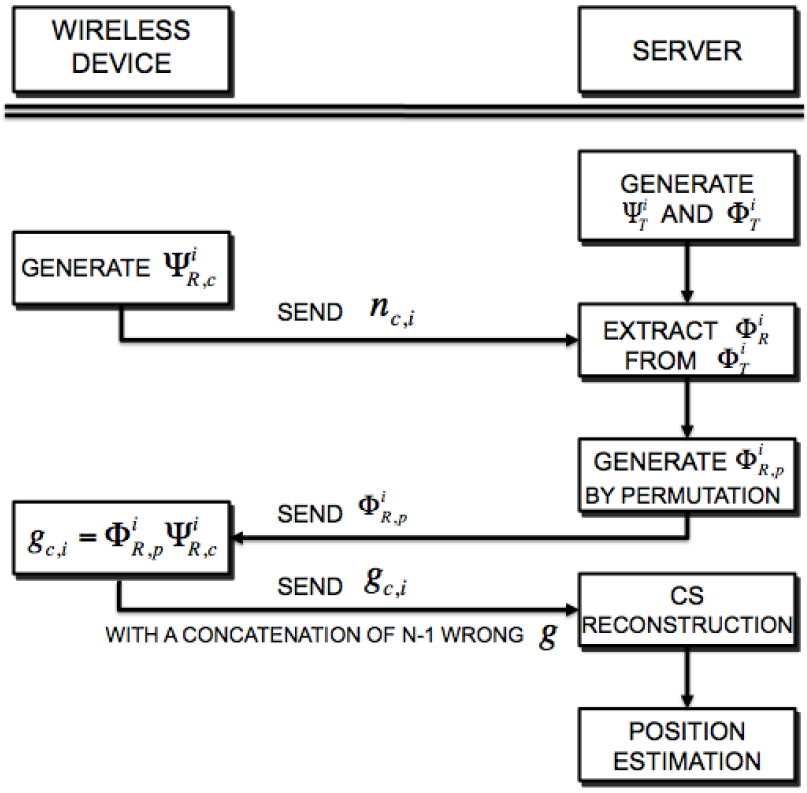

Figure 5.5: SecLoc system's security architecture

5.3.2 Possible Attacks from Malicious Users

An attack could follow three directions:

1. Find Φ matrix by intercepting the server (modern network cryptographic protocols could guarantee that the decryption of Φ_p – where p denotes the permutation of the lines – is almost infeasible in practice due to the combinatorial nature of the inverse problem).

2. Find $\mathbf{g}$ by intercepting the opposite direction (the exact knowledge of $\mathbf{g}$ is insufficient, resulting in a significantly increased estimation error, when the attacker does not achieve the exact estimate of Φ_R^i).

3. Find the correct measurement vector $\mathbf{g}$, which increases the estimation error, without exact knowledge.

5.3.3 Experimental Results

Fig. 5.6 shows the encryption capability of the method for the Bayesian Compressive Sensing (BCS) [110] and Bayesian Compressive Sensing – Gaussian Scale Mixture (BCS-GSM) [84, 111] reconstruction algorithms.

In particular, the average localization error, over 100 Monte-Carlo runs, is shown as a function of the percentage of permuted lines from 0% to 100%, of the true matrices Φ_R^i, where the reconstruction is performed by considering exact knowledge of the measurement vectors $\mathbf{g}$. The results agree with the intuition that as the complexity of the permutation increases, the estimation accuracy decreases without an exact estimate of the true measurement matrix.

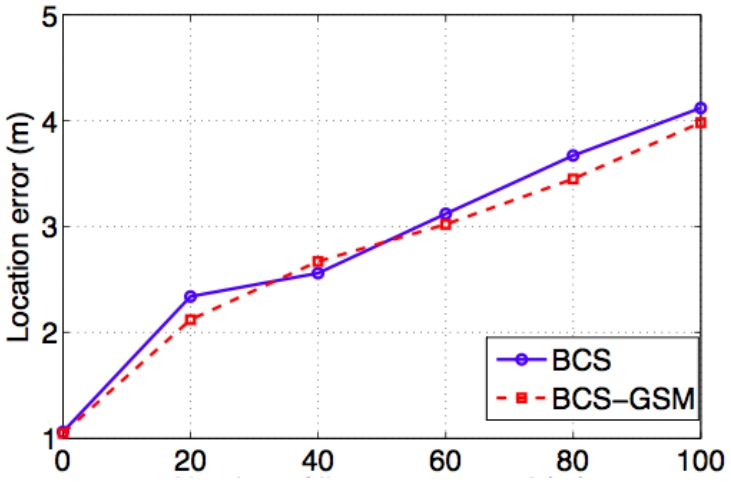

Figure 5.6: Evaluation of the encryption property using BCS and BCS-GSM, for a varying number (percentage) of permuted lines of Φ_R^i (x axis).

5.4 Stealth Encryption based on Eulerian Circuits

In this work, we introduce an encryption system that destructs the semantics of a text while retaining it almost in correct syntax. The encryption is almost undetectable, since the text is not able to be identified as different to a regular text. This makes the system resilient to any massive scanning attack. The system is based on the Eulerian circuits of the original text. We provide the asymptotic estimate of the capacity of the system when the original text is a Markovian string.

In the seventies the US Air Force set up a program for a stealth fighter (so–called Nighthawk F-117). The principle was to design a fighter aircraft that would be almost undetectable to radar. The idea was in a specifically shaped hull, inside which the aircraft would be of a classical design. Our aim is to design an equivalent for text encryption so that an encrypted text would hardly detectable as such by an automated scanning process.

The practice and study of techniques for secure communication between users has become emergent and important in every day life. The meaning of cryptology is huge in the disciplines of computer systems security and telecommunications. The main goal is to provide mechanisms such that two or more users or devices can exchange messages without any third party intervention. Nowadays it is used in wireless networks, information security in banks, military purposes, biometric recognition, smart cards, VPN, WWW, satellite TV, databases, VOIP and plethora of systems.

In the very early stages, cryptographic mechanisms were dealing with the language structure of a message, where nowadays, cryptography has to deal with numbers, and is based on discrete mathematics, number theory, information theory, computational complexity, statistics and combinatorics.

Cryptography consists of four basic functions: (a) confidentiality, (b) integrity, (c) non-repudiation and (d) certification. The encryption and decryption of a message is based on a cryptographic algorithm and a cryptographic key. Usually the algorithm is known, so the confidentiality of the encrypted transmitted message is based on the confidentiality of the cryptographic key. The size of that key is counted in number of bits. In general, the larger the cryptographic key, the harder the decryption of the message.

There are two main categories of crypto-systems: (a) classic crypto-systems, which are divided in substitution ciphers and transposition ciphers, and (b) modern crypto-systems, which are divided in symmetric (share a common key)

and asymmetric (use public and private key). Systems based on symmetric cryptography are divided into block ciphers and stream ciphers. However, users based on asymmetric cryptographic systems, know the public key, but the private key is secret. The information being encrypted by one of the keys, can be decrypted only by the other key.

Up to now, the main methods used for text encryption are sophisticated algorithms that transform original data into encrypted binary stream. The problem is that such streams are easily detectable under an automated massive attack, because they are in very different format and aspect of non encrypted data, *eg* texts in natural language. This way any large scale data interception system would very easily detect the encrypted texts. The result of this detection is twofold. First detected encrypted texts can be submitted to massive decryption processes on large computing resources, and finally be deciphered. Second, even when the texts would not eventually be deciphered, the source or the destination of the texts are at least identified as hiding their communications and therefore can be subject to other intrusive investigations. In other words encryption does not protect against a massive spying attack if encrypted texts are easily detectable.

In our method we use a permutation of the symbols *i.e.* n-grams of the original text. Doing so leads to the apparent destruction of the semantic information of the text while keeping the text quasi correct in its syntax, and therefore undetectable under an automated syntactic interception process. To retrieve the original information the n-grams would be reordered in their original

arrangement.

5.4.1 Background

In the following terminology, let T be a text written on an alphabet $\mathcal{A}$ of size V. Let r be an integer, which is used to denote a sequence of r consecutive symbols appearing in a text, *i.e* r-gram, of text T.

5.4.1.1 Syntax Graph

According to the above terminology, we define the *syntax graph* G of text T. We assume a fixed integer r and we denote $G_r(T) = (V, E)$ the directed *syntax graph* of the r-grams of text T. There is an edge between two r-grams a and b, if b is obtained by the translation by one symbol of r-gram a in the text T. For example, the 3-gram $b =$ "*mpl*" follows the 3-gram $a =$ "*amp*" in the text $T =$ "*example*".

The graph $G_r(T)$ is a multi-graph, since several edges can exist between two r grams a and b, as many as b follows an instance of a in the text T.

Fig. 5.7 shows the syntax graph of the famous Shakespeare's text $T =$ "*to be or not to be that is the question*", with $|V| = 13$ (for 1-grams) and $|E| = 39$.

5.4.1.2 Eulerian Path and Circuit

An Eulerian path in a multi-graph graph is a path which visits every edge exactly once. An Eulerian circuit or cycle is an Eulerian path which starts and ends on the same vertex and visits every edge exactly once. The number of Eulerian paths in a given multi-graph is easy to compute (in a polynomial time) as a sequence of binomials and a determinant; or can be adapted from BEST theorem [117, 118],

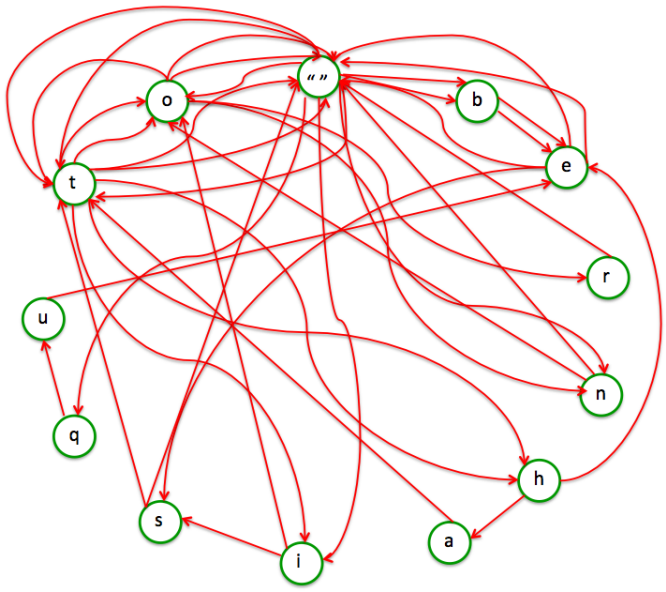

Figure 5.7: Syntax Graph (1-grams) of the famous Shakespeare's sentence $T =$ "*to be or not to be that is the question*", where " " denotes the space.

to enumerate the number of Eulerian circuits [119, 120], [112] which will be explained in section 5.4.3.

5.4.2 Motivation and Algorithm Description

5.4.2.1 Motivation

The originality of the system is that the appearance of the transmitted information, although encoded, is not changed, so that malicious software applications cannot determine that a flow of information under study is encoded or not. Consequently, being undetected, the encoded information will not be subject to massive deciphering techniques. The probability of the transmitted information being deciphered is therefore dramatically higher, regardless of the encoding scheme.

When a malicious deciphering application has detected an emitter or a

receiver as transmitting encoded information (regardless of whether or not it managed at deciphering it), it may tag it, so as to trigger further actions, like systematically analyzing of its transmitted information, trying to infiltrate it to steal further information. Therefore, an additional and un-addressed technical problem solved by this system, consists in avoiding being determined by such an application the fact that encoded information is transmitted.

The system applies in particular to text transmission, but can also apply to other types of information. The encoded text comprises of only a few grammatical or lexicographic errors, so that any detecting mechanisms based on first level errors will not detect anything and will consider such an encoded text as "badly written normal text". Accordingly, the encoded text will not be detected as encoded and no deciphering mechanism will be triggered. Any malicious application analyzing such information will detect a text, with a typical text format (words, sentences, etc.), and detecting that it is encoded text would require a complex semantic analysis. In addition, such a semantic analysis would become more difficult by having no information about the language used to write such a text.

Let the text "*Up to now, the main ... easily detectable.* " written in the Introduction, be the text T. We design the Syntax Graph $G(T)$ with the 4-grams of T, which gives 3.42×10^{42} discrete Eulerian circuits. One possible circuit is the following:

T' = "In In other words encryptions and finally be deciphered, therefore can be submitted binary stream. This way any large communication does not

protectable under an automated massive investination is twofold. First detect the texts can be decrypted texts are in very easily detection of this detectable. Up to now, they are easily detected encryption processes on large scale data into encrypted to massive deciphered. Second, even when the main methods used form original data intrusive spying resources, and as hiding attack if encrypted data, eg text encrypted aspect of non encrypted texts are attack, because the source or texts would very different for the encryption are easily detect against a massive at least identified algorithms that transformat and their computing the texts in natural language. The problem is that such streams are sophisticated texts. The result of the destigation system would not eventually be subject to other interceptions. In "

In this example, it is clear that the encoded text looks like an English text, and that it would require either manual intervention to determine that it is not a normal text, or a very complex automatic semantic analysis. Even the small text presented in Fig. 5.7 gives 1.19×10^{10} discrete Eulerian circuits.

5.4.2.2 Algorithm Description

In general, the block ciphers method, breaks into blocks the initial message, which is going to be encrypted, and encrypts every block separately. Usually these blocks's size is 64 or 128 bits. This way the encryption can be adapted to the length of the information to be coded. The encryption is based on a mathematical encryption or cryptographic function $f(x, k_0)$ where x is the block and k_0 is a shared key. The result is a new cryptographic block y that usually

has the same length with the initial block.

The length of every block has to be large enough, in order to avoid dictionary attacks. If the length of every block is small, then a malicious user, can find a pair of clean and respectfully encrypted block to design a dictionary that corresponds every clean block to a encrypted block. Based on that dictionary, every text that is encrypted with the particular key could be decrypted.

The decryption process is the inverted encryption process $g(y, k_0)$ such that $g(f(x, k_0), k_0) = x$. In that case, a decryption function is used instead of the encryption function. Some classic block cipher algorithms are Data Encryption Standard [121], Triple DES which uses three keys instead of one, Advanced Encryption Standard [122], Tiny Encryption Algorithm [123] and others [124].

In our algorithm we assume that there is an encryption function $f(x, N, k_0)$ where k_0 is a shared key that produces an integer $y \in (0, N-1)$, when x is an integer, and $\in (0, N-1)$. Let $g(y, N, k_0)$ be an inverse function, such that $x = g(f(x, N, k_0), N, k_0)$. The function can be made of the concatenation of m blocks encryption of size ℓ (64 or 128 bits) with $m = \lceil \frac{\log_2 N}{\ell} \rceil$. In this case $f(x, N, k_0) = x_1 y_2 \dots y_m$ if x is made of the concatenation of blocks $x_1 \dots x_m$ with $y_i = f(x_i, k_0)$ for all $1 < i < m$. The first block is left unencrypted in order to have $y < N$ with very high probability.

Let r_0 be an integer, the proposed scheme is described in the following algorithm:

Algorithm 5 Encryption system based on Eulerian path

Suppose that Alice is the transmitter with text T, Bob is the receiver, and Neysa is the malicious user.

1. Transmitting Information:

   - Alice builds the syntax graph $G_{r_0}(T)$
   - Computes the number N of Eulerian paths. *Assume that Alice and Bob share the same indexing method of the Eulerian paths in* $G_{r_0}(T)$.
   - *Since text* T *itself is a specific Eulerian path, assume that* $I(T)$ *is the index of text* T *as Eulerian path.* Alice builds the text T' being the Eulerian path with index $f(I(T), N, k_0)$, and then transmits the text T' to Bob.

   According to the malicious user Neysa, the text T' seems to have correct syntax, but the original semantic is destroyed.

2. Receiving Information:

   - Bob builds the syntax graph $G_{r_0}(T')$ which is the same as $G_{r_0}(T)$.
   - Computes the number of Eulerian paths N and is able to recover the original text index $I(T) = g(I(T'), N, k_0)$.

5.4.3 Performance in Markov Models

In this section we consider a string $T = X_n$ of length n generated by a Markov process of memory 1 over an alphabet $\mathcal{A}$ of size V. Notice that there is no loss of generality with the r-grams description of our model, in this case $\mathcal{A}$ is the set of all r-grams. We denote $\mathbf{P}$ the Markov transition matrix: for $(a, b) \in \mathcal{A}^2$ we denote p_{ab} the coefficients of matrix $\mathbf{P}$, *ie* the probability that symbol b follows symbol a.

The Euler code length of a string is the logarithm in base 2 of the number of eulerian circuits that can be build from string X_n while making it cyclic. We denote this quantity L_n it reflects the number of bits needed to index the eulerian number and indeed the encoding capacity of our scheme.

Theorem 5 *The average code length L_n of X_n is asymptotically equivalent to the entropy $H(X_n)$ of the string X_n when $n \to \infty$.*

This theorem has an important consequence. Let denote Y_n the string obtained by the encryption algorithm. We assume that the encryption algorithm selects an Euler circuit uniformly in the set of Eulerian circuits that can be built in X_n, then we have the following corollary:

Corollary 1 *The mutual information rate $\frac{1}{n}I(X_n, Y_n)$ tends to zero when $n \to \infty$.*

Proof 1 *We already know that $H(Y_n) = H(X_n)$ since X_n and Y_n have same probability in the Markov process, thus $I(X_n, Y_n) = H(X_n) - H(Y_n|X_n)$, where the last term is the conditional entropy of Y_n with respect to X_n. Since $H(Y_n|X_n) = L_n$ the results holds.*

Remark The order of magnitude of $I(X_n, Y_n)$ that is obtained in the proof of theorem 5 is $O(\log^3 n\sqrt{n})$ but the right order should be $O(\log n)$ with a more careful handling of the error terms. Anyhow this proves that our scheme is efficiently erasing the information of X_n while keeping the text in a formal shape.

We will also evaluate the average number E_n of Eulerian circuits in X_n. To this end, for $s \geq 0$ we denote $\mathbf{P}(s)$ the matrix made of coefficients $(p_{ab})^s$ (if $p_{ab} = 0$ we assume $(p_{ab})^s = 0$). We denote $\lambda(s)$ the main eigenvalues of matrix $\mathbf{P}(s)$.

Theorem 6 *The average number of Eulerian circuits of string X_n of length n is equivalent to $\frac{\alpha}{n}\lambda^{2n}(\frac{1}{2})$ for some $\alpha > 0$ that can be explicitly computed.*

Remark This average is purely theoretical, in fact it is impossible to simulate this result when n is large like in the example text, since the most important and decisive contributions come from strings with extremely low probabilities. In order to prove our theorem we need some notations and lemmas.

Let $\mathbf{k}$ be a $V \times V$ integer matrix defined on $\mathcal{A} \times \mathcal{A}$ which is the adjacency matrix of syntax graph $G(X_n)$, *i.e.* the coefficient k_{ab} of $\mathbf{k}$ is equal to the number of time symbol b follows symbol a in X_n, we say that $\mathbf{k}$ is the *type* of string X_n as defined in [112]. For $(a, b) \in \mathcal{A}^2$ we also denote δ_{ab} the type of the string ab.

For $c \in \mathcal{A}$ we denote $k_c = \sum_{d\in\mathcal{A}} k_{cd}$ and $k^c = \sum_{d\in\mathcal{A}} k_{dc}$ respectively the outdegree and indegree of symbol c in the syntaxic graph. Let $\mathcal{F}_n$ the set of *balanced* types, *i.e.* such that $\forall c \in \mathcal{A} : k_c = k^c$, and such $\sum_{(c,d)\in\mathcal{A}^2} k_{cd} = n$.

Lemma 4 *The set $\mathcal{F}_n$ is a lattice of dimension $V^2 - V$ [112] . Its size $a(n) = O(n^{V^2-V+1})$ and we denote ω the volume of its elementary element.*

Proof 2 *The set of matrix is embedded in the vector space of real matrices which is a dimension V^2. The V balance equations are in fact $V - 1$ since any of them*

can be deduced from the sum of the other. There is a last equation to specify that all coefficients sum to n.

We denote $\mathcal{F}(1)$ the set of balanced real matrices with positive coefficients that sum to 1. For $\mathbf{y}$ a real non negative matrix and $s \geq 0$, we denote $L(\mathbf{y}, s) = \sum_{(c,d)\in\mathcal{A}^2} y_{c,d} \log(\frac{y_c}{y_{cd}} p_{cd}^s)$

We have the following technical lemma.

Lemma 5 *Let $s > 0$, the maximal value of $L(\mathbf{y}, s)$ for $\mathbf{y} \in \mathcal{F}(1) - \frac{\delta_{ba}}{n}$ is the matrix $\tilde{\mathbf{y}}_n(s)$ which converges to a matrix $\tilde{\mathbf{y}}(s) \in \mathcal{F}(1)$ whose (c, d) coefficient is $v_c(s)u_d(s)\frac{p_{cd}^s}{\lambda(s)}$ with $(u_c(s))_{c\in\mathcal{A}}$ and $(v_c(s))_{c\in\mathcal{A}}$ being respectively the right and left main eigenvectors of $\mathbf{P}(s)$ (with $\sum_{c\in\mathcal{A}} u_c(s)v_c(s) = 1$). For instance $L(\tilde{\mathbf{y}}(s), s) = \log \lambda(s)$ and $L(\tilde{\mathbf{y}}_n(s), s) = L(\tilde{\mathbf{y}}(s), s) - \frac{1}{n}(\log \lambda(s)\frac{u_b(s)}{u_a(s)})$.*

Proof 3 *We have for all $(c, d) \in \mathcal{A}^2$ the gradient matrix is*

$$\frac{\partial}{\partial y_{cd}} L(\mathbf{y}, s) = \log \frac{y_c}{y_{cd}} p_{cd}^s \ . \tag{5.5}$$

The maximum on $\mathcal{F}(1)$ or $\mathcal{F}(1) - \frac{1}{n}\delta_{ba}$ must be member of the vector space generated by the matrix $\mathbf{1}$ (made of all one), and the matrices $\mathbf{A}_j$, $j \in \mathcal{A}$, the coefficients of A_j are all zeros excepted 1 on the jth column and -1 on the jth row, and zero on the diagonal. These matrices are the orthogonal matrices that define $\mathcal{F}(1)$ (or a translation of it). Membership to this vector space is equivalent to the fact that $\frac{\partial}{\partial y_{cd}} L(\mathbf{y}, s)$ must be of the form $\alpha + z_c - z_d$ for some α and $(z_c)_{c\in\mathcal{A}}$, which is equivalent to the fact that

$$\frac{y_{cd}}{y_c} = \frac{x_d}{x_c}\frac{p_{cd}^s}{\lambda} \tag{5.6}$$

for some λ and $(x_c)_{c\in\mathcal{A}}$. From the fact that $\sum_{d\in\mathcal{A}} \frac{y_{cd}}{y_c} = 1$ we get $\lambda = \lambda(s)$ and

$(x_c)_{c\in\mathcal{A}} = (u_c(s))_{c\in\mathcal{A}}$, *i.e.* $\lambda x_c = \sum_{d\in\mathcal{A}} p^s_{cd} x_d$. *Consequently*

$$\begin{aligned} L(\mathbf{y}, s) &= \sum_{(c,d)\in\mathcal{A}^2} y_{cd} \log(\lambda \frac{x_c}{x_d}) && (5.7)\\ &= \log(\lambda) \sum_{(c,d)\in\mathcal{A}^2} y_{cd} && (5.8)\\ &\quad + \sum_{c\in\mathcal{A}} (y_c - y_c) \log(x_c) && (5.9) \end{aligned}$$

Thus $L(\tilde{\mathbf{y}}(s), s) = \log\lambda(s)$ *and* $L(\tilde{\mathbf{y}}_n(s), s) = (1-\frac{1}{n})\log\lambda(s) + \frac{1}{n}\log\frac{u_a(s)}{u_b(s)}$.

To simplify our proofs we will assume in the sequel that all strings start with a fixed initial symbol a. The strings starting with a having type $\mathbf{k}$ have probability $\prod_{(c,d)\in\mathcal{A}^2} p_{cd}^{k_{cd}}$ that we denote $\mathbf{P}^{\mathbf{k}}$. We denote

$$B_{\mathbf{k}} = \prod_{c\in\mathcal{A}} \binom{k_c}{(k_{cd})_{b\in\mathcal{A}}} . \tag{5.10}$$

For a matrix $\mathbf{M}$ and $(c,d)\in\mathcal{A}^2$ we denote $\det(\mathbf{M})$ the (c,d) cofactor of $\mathbf{M}$. We know that the number of eulerian circuits that share the same balanced type $\mathbf{k}$ is equal to $B_{\mathbf{k}}\frac{\det_{cd}(\mathbf{I}-\mathbf{k}^*)}{k_c}$ for any pair of symbols $(c,d)\in\mathcal{A}^2$, defining $\mathbf{k}^*$as the matrix $\frac{k_{cd}}{k_c}$ for $(c,d)\in\mathcal{A}^2$ and $\mathbf{I}$ the identity matrix.

The number of strings that share the same type $\mathbf{k}$ depends on their terminal symbol b. Let denote $N^b_{\mathbf{k}}$ this number. When $b\neq a$, the type is not balanced but $\mathbf{k}+\delta_{ba}\in\mathcal{F}_n$. As described in [112],

$$N^b_{\mathbf{k}} = B_k \det_{bb}(\mathbf{I} - (\mathbf{k}+\delta_{ba})^*) . \tag{5.11}$$

In passing $N^b_{\mathbf{k}}\mathbf{P}^{\mathbf{k}}$ is the probability that string X_n has type $\mathbf{k}$ and we have

the identity

$$\sum_{b\in\mathcal{A}}\sum_{\mathbf{k}+\delta_{ba}\in\mathcal{F}_n} N_{\mathbf{k}}^b\mathbf{P}^{\mathbf{k}} = 1\ . \tag{5.12}$$

Similarly the number of eulerian circuits $E_{\mathbf{k}}^b$ corresponding to the string X_n assuming it ends with b and is made cyclic satisfies:

$$E_{\mathbf{k}}^b = \frac{1}{k_{ba}+1}B_k\det{}_{bb}(\mathbf{I}-(\mathbf{k}+\delta_{ba})^*)\ . \tag{5.13}$$

For convenience we will handle only natural logarithms. We have $L_n = \sum_{b\in\mathcal{A}} L_n^b$ with

$$L_n^b = \sum_{\mathbf{k}+\delta_{ba}\in\mathcal{F}_n} N_{\mathbf{k}}^b\mathbf{P}^{\mathbf{k}}\log E_{\mathbf{k}}^b\ . \tag{5.14}$$

Proof 4 (Proof of Theorem 5) *Using the Stirling approximation:* $k! = \sqrt{2\pi k}k^k e^{-k}(1+O(\frac{1}{k}))$ *and defining* $\ell(\mathbf{y}) = \sum_{cd} y_{cd}\log\frac{y_c}{y_{cd}}$ *we have*

$$\begin{aligned} L_n^b &= (n+O(1))\frac{1}{n^{(V-1)V/2}}\sum_{\mathbf{k}+\delta_{ba}\in\mathcal{F}_n} r_b(\mathbf{y})\exp(nL(\mathbf{y},1))\ell(\mathbf{y}) \\ &\quad +O(\log n) \end{aligned} \tag{5.15}$$

where $r_b()$ *is a rational function and* $\mathbf{y}$ *is the matrix of coefficients* $y_{cd} = \frac{k_{cd}}{n}$.

According to lemma 5 the maximum value of $L(\mathbf{y},1)$ *is* $\log\lambda(1) = 0$ *and thus we already have* $L_n^b = O(a(n)) = O(n^{V^2-V})$. *for a matrix* $\mathbf{M}$ *we denote* $\|\mathbf{M}\|$ *the cartesian norm, i.e. the square root of the squared coefficients. Since* $\tilde{\mathbf{y}(1)}$ *is the maximum of* $L(\mathbf{y},1)$ *over* $\mathcal{F}(1)$ *there exists* $A>0$ *such that* $\forall\mathbf{y}\in\mathcal{F}(1)$*:* $L(\mathbf{y},1)\le L(\tilde{\mathbf{y}}(1),1) - A\|\mathbf{y}-\tilde{\mathbf{y}}(1)\|^2$, *and when* $\mathbf{y}+\frac{1}{n}\delta_{ba}\in\mathcal{F}(1)$

$$L(\mathbf{y},1)\le L(\tilde{\mathbf{y}}_n(1)) - A\|\mathbf{y}-\tilde{\mathbf{y}}_n(1)\|^2 \tag{5.16}$$

Let $B > 0$: we define

$$L_n^b(B) = \frac{n}{n^{(V-1)V/2}} \sum_{\substack{\mathbf{k}+\delta_{ba} \in \mathcal{F}_n \\ |\mathbf{y}-\tilde{\mathbf{y}}_n| < B \log n/\sqrt{n}}} r_b(\mathbf{y}) \exp(nL(\mathbf{y}))\ell(\mathbf{y}) \tag{5.17}$$

From (5.16) we have $L_n^b = L_n^b(B)(1+O(n^{-1})$. Since function $L(\mathbf{y},1)$ is infinitely derivable and attains its maximum in $\mathcal{F}(1)$, which is zero, on $\tilde{\mathbf{y}}_n(1)$ we have for all $\mathbf{y} \in \mathcal{F}(1) - \frac{1}{n}\delta_{ba}$

$$L(\mathbf{y},1) = L(\tilde{\mathbf{y}}_n(1)) - D_n(\mathbf{y} - \tilde{\mathbf{y}}_n(1)) + O(\|\mathbf{y} - \tilde{\mathbf{y}}_n(1)\|^3) \ , \tag{5.18}$$

where $D_n()$ is a positive quadratic form obtained from the second derivative of $L(\mathbf{y},1)$ on $\tilde{\mathbf{y}}_n(1)$. The value of L_n will be attained in the vicinity of the maximum since $\exp(nL(\mathbf{y},1))$ behaves like a Dirac when $n \to \infty$. Indeed we are in a kind of Saddle point application. Thus, since $L(\tilde{\mathbf{y}}_n(1)) = \frac{1}{n}\log\lambda(1)\frac{u_b(1)}{u_a(1)} = 0$ ($\lambda(1) = 1$ and $\forall c \in \mathcal{A}$: $u_a(1) = 1$)

$$\begin{aligned} L_n^b(B) &= \frac{1}{n^{(V-1)V/2}}(1 + O(\frac{\log^3 n}{\sqrt{n}})) \\ &\quad \sum_{\substack{\mathbf{k}+\delta_{ba} \in \mathcal{F}_n \\ |\mathbf{y}-\tilde{\mathbf{y}}_n| < B \log n/\sqrt{n}}} \tilde{r}(\mathbf{y}) e^{nD_n(\mathbf{y}-\tilde{\mathbf{y}}_n(1))} \ . \end{aligned} \tag{5.19}$$

Since $\frac{1}{n}\mathcal{F}_n$ is a lattice of degree $(V-1)V$ with elementary volume ωn^{-V^2+V} and $D_n()$ converge to some the non negative quadratic form $D()$ on $\mathcal{F}(1)$:

$$\begin{aligned} L_n^b(B) &= n\ell(\tilde{\mathbf{y}}(1))\frac{r(\tilde{\mathbf{y}}(1))}{n^{(V-1)V/2}}\frac{n^{V^2-V}}{\omega}(1 + O(\frac{1}{\sqrt{n}})) \\ &\quad \int_{\mathcal{F}(1)} e^{-nD(\mathbf{y}-\tilde{\mathbf{y}}(1))} d\mathbf{y}^{V(V-1)} \end{aligned} \tag{5.20}$$

We use the property that

$$\int_{\mathcal{F}(1)} \exp(-nD(\mathbf{y}-\tilde{\mathbf{y}}))d\mathbf{y}^{V(V-1)} = \frac{1}{\sqrt{n^{V^2-V}\det(\pi D)}} \tag{5.21}$$

where $\det(\pi D)$ *must be understood as the determinant of the quadratic operator* $\pi D()$ *on the vector space*$\mathcal{F}(1)$*. Therefore*

$$L_n = n\ell(\mathbf{y})\frac{1}{\sqrt{\det(\pi D)}}\sum_{b\in\mathcal{A}} r_b(\tilde{\mathbf{y}})(1+O(\frac{\log^3 n}{\sqrt{n}})) \ . \tag{5.22}$$

The same analysis can be done by removing $\log E^b_{\mathbf{k}}$ *and since via (5.12) we shall get* $\sum_{b\in\mathcal{A}} \frac{r_b(\tilde{\mathbf{y}}(1))}{\sqrt{\det(\pi D)}} = 1$*, we get*

$$L_n = n\ell(\tilde{\mathbf{y}}(1))(1+O(\frac{\log^3 n}{\sqrt{n}})) \ . \tag{5.23}$$

We terminate the proof of theorem 5 by the fact that $n\ell(\tilde{\mathbf{y}}(1)) = H(X_n)$ *(since* $\forall c \in \mathcal{A}$*:* $u_c(1) = 1$ *and* $(v_c(1))_{c\in\mathcal{A}}$ *is the Markov stationary distribution).*

Proof 5 (Proof of Theorem 6) *The proof of theorem 6 proceeds equivalently except that* $E_n = \sum_{b\in\mathcal{A}} E^b_n$ *with*

$$E^b_n = \sum_{\mathbf{k}+\delta_{ba}\in\mathcal{F}_n} N^b_{\mathbf{k}}\mathbf{P}^{\mathbf{k}}E^b_{\mathbf{k}} \ . \tag{5.24}$$

The main factor $\mathbf{P}^{\mathbf{k}}B^2_{\mathbf{k}}$ *leads to a factor* $\exp(2nL(\mathbf{y},\frac{1}{2}))$*. Consideration on the order of the* n *factors leads to the estimate in* $An^{-1}\lambda^{2n}(\frac{1}{2})$*.*

More extensive studies can be found in our later works in [113, 114, 115, 116].

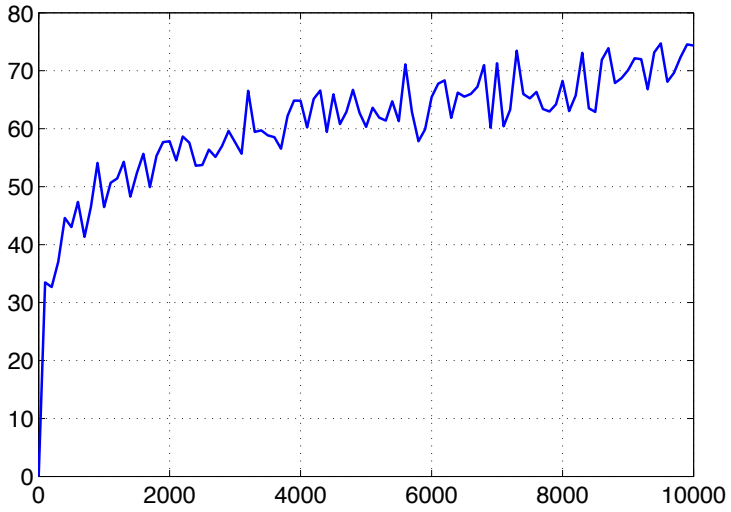

Figure 5.8: Mutual information $I(X_n, Y_n)$ versus n for the Markov process of figure 5.7. Y axe is the mutual information (in bit), X axe is the length of the string up to 10,000.

5.4.4 Experimental Results

Figure 5.8 shows $I(X_n, Y_n)$, the discrepancy between L_n and $H(X_n)$, more precisely the mean value of $\log N_{\mathbf{k}} + \log \mathbf{P}^{\mathbf{k}}$ versus the string length n, when X_n is generated by a Markov process of memory 1 based on the statistics of the syntax graph of the sentence *to be or not to be that is the question* (depicted on figure 5.7) [113]. As predicted the conjectured it is quite sub-linear and seems to be in $\log n$ as conjectured. Each point has been simulated 100 times.

5.5 Chapter Summary

In this Chapter we presented our work in three additional research subjects, (a) localization and path tracking in indoor environments, (b) encryption based on compressive measurement vectors, and (c) encryption based on the Eulerian circuits of original texts.

In the first additional research subject we proposed a path-tracking method for indoor localization by exploiting the efficiency of a CS framework with the accuracy of a Kalman filter. Using our previous method of MvG-based modeling as an initial step to constrain the area of interest, CS was applied then as a refinement step by recovering an appropriate sparse position-indicator vector. The experimental evaluation with a set of real datasets revealed an increased localization accuracy, when compared with previous state-of-the-art methods, while operating at a significantly reduced computational cost by using only a small number of compressed RSS measurements.

In the second additional research subject, we presented an efficient encryption system based on Compressive Sensing, without the additional computational cost of a separate encryption protocol, when applied to indoor location estimation problems. The breakthrough of the method is the use of the weakly encrypted measurement matrices (which are generated when solving the optimization problem to localize the source), along with an alternative key to secure the system.

In the third additional research subject, we presented an encryption system based on Eulerian circuits, that destructs the semantics of a text while retaining it in correct syntax. We studied the performance on Markov models, and presented experiments on real text.

Chapter 6

Conclusions and Perspectives

This thesis introduced and compared two novel topic detection and classification methods based on Joint Complexity and Compressive Sensing. In the first case, the joint sequence complexity and its application was studied, towards finding similarities between sequences up to the discrimination of sources. We exploited datasets from different natural languages using both short and long sequences. We provided models and notations, and presented the theoretical analysis. We applied our methodology to real messages from Twitter, where we evaluated our proposed methodology on topic detection, classification and trend sensing, and we performed automated online sequence analysis.

In the second case, the classification problem was reduced in a sparse reconstruction problem in the framework of Compressive Sensing. The dimensionality of the original measurements was reduced significantly via random linear projections on a suitable measurement basis, while maintaining an increased classification accuracy.

The empirical experimental evaluation revealed that the methods outperform previous approaches based on bag-of-words and semantic analysis, while the Compressive Sensing based approach achieved a superior performance when

compared to state-of-the-art techniques. We performed an evaluation of various datasets in Twitter and compete in the Data Challenge of the World Wide Web conference 2014, where we got the 3rd Prize.

Finally, a hybrid tracking system is presented, which exploits the efficiency of a Kalman filter in conjunction with the power of Compressive Sensing to track a Twitter user, based on his tweets. The experimental evaluation reveals an increased classification performance, while maintaining a low computational complexity.

There is a growing interest in statistical methods that exploit various spatio-temporal statistical properties of the datasets to form robust maps of the space. In general, a user's tweets are described by very transient phenomena (frequency, language, list of followers) and is highly time-varying. At the same time, the collection and analysis of data is subject to the quality and relevance of real time search, the correlation information between groups of users, and the analysis of the relationship between members of a group or a community. Thus, the general problem of building a theoretical framework to analyze these data taking into consideration the above limitations opens up exciting research opportunities.

Regarding the theoretical analysis of the Joint Complexity method, we gave a conjecture inspired from the general results about the asymptotic digital trees and suffix tree parameters distribution. We omitted the proof of the conjecture on its variance, which will be studied in a future work.

Regarding the part of Compressive Sensing, in the present work, the un-

known class was estimated by performing separate reconstruction for each representative tweet. A straightforward extension will be the use of the joint sparsity structure of the indicator vector $\mathbf{s}$ among the reference points for the simultaneous classification. Moreover, the random nature of the measurement vectors associated with a representative tweet vector could be exploited in order to enhance the encryption capabilities of the proposed CS classification approach, without the additional computational cost of a separate encryption protocol. Besides, a more thorough study should be carried out for the robustness of the inherent encryption property in terms of the several network parameters. The choice of appropriate sparsifying and measurement bases is crucial for an increased classification accuracy. The design of new transform and measurement bases, $\mathbf{\Psi}$, $\mathbf{\Phi}$, respectively, being adaptive to the specific characteristics of the data is a significant open problem.

Another interesting use of Joint Complexity method, would be the design of a source finding model, based on topic generation processes and the statistics of the obsolescence rates of the topics. We may assume that topic sources are generated on each Twitter user like an i.i.d Poisson process of a specific rate per timeslot studied. Then we have to obtain an obsolescence rate, which defines the number of tweets posted in a timeslot based on the difference between a tweet's time stamp and the time stamp of its source. The source finding model will be also explored in a future work.

Bibliography

[1] N. Hegde, L. Massoulié and L. Viennot, "Self-Organizing Flows in Social Networks", in *Structural Information and Communication Complexity, Lecture Notes in Computer Science*, Vol. 8179, pp. 116–128, 2013. (Cited on page 5.)

[2] B.A. Prakash, M. Seshadri, A. Sridharan, S. Machiraju and C. Faloutsos, "EigenSpokes: Surprising Patterns and Scalable Community Chipping in Large Graphs", in *IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 290-295, Dec. 2009. (Cited on page 5.)

[3] T. Sakaki, M. Okazaki, and Y. Matsuo,"Earthquake Shakes Twitter Users: Real-Time Event Detection by Social Sensors", in *Proc. of the 19th ACM International Conference on World Wide Web (WWW T10)*, New York, NY, USA. (Cited on pages 5 and 19.)

[4] V. Lampos and N. Cristianini, "Tracking the Flu Pandemic by Monitoring the Social Web", in *2nd International Workshop on Cognitive Information Processing (CIP)*, pp. 411-416, June 2010. (Cited on page 5.)

[5] A. Toninelli, A. Pathak and V. Issarny, "Yarta: A Middleware for Managing Mobile Social Ecosystems", in *International Conference on Grid and Pervasive Computing (GPC)*, pp. 209–220, Oulu, Finland, May 2011. (Cited on page 5.)

[6] U. K. Wiil, J. Gniadek and N. Memon, "Measuring Link Importance in Terrorist Networks", in *International Conference on Advances in Social Networks Analysis and Mining*, August 9-11, 2010. (Cited on page 5.)

[7] K. Valdis, "Uncloaking Terrorist Networks", in *First Monday 7*, no. 4, March 2002. (Cited on page 5.)

[8] T. White, W. Chu and A. Salehi-Abari, "Media Monitoring Using Social Networks", in *IEEE Second International Conference on Social Computing*, pp.661-668, August 2010. (Cited on page 5.)

[9] E.M. Rodrigues, N. Milic-Frayling, B. Fortuna, "Social Tagging Behaviour in Community-Driven Question Answering", in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp.112-119, Dec. 2008. (Cited on page 5.)

[10] G. Li, H. Li, Z. Ming, R. Hong, S. Tang and T. Chua, "Question Answering over Community Contributed Web Video", in *Transactions on Multimedia*, no.99, 2010. (Cited on page 5.)

[11] J. Jiao, J. Yan, H. Zhao and W. Fan, "ExpertRank: An Expert User Ranking Algorithm in Online Communities", in *International Conference on New Trends in Information and Service Science*, pp. 674-679, June 30 - July 2, 2009. (Cited on page 5.)

[12] M. Conti, F. Delmastro and A. Passarella, "Social-aware Content Sharing in Opportunistic Networks", in *6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops*, pp.1-3, June 2009. (Cited on page 6.)

[13] B. Markines, C. Cattuto and F. Menczer; "Social spam detection", in *Proc. of the 5th ACM International Workshop on Adversarial Information Retrieval on the Web*, pp. 41-48, New York, NY, USA, 2009. (Cited on page 6.)

[14] M.A. Garcia-Ruiz, M.V. Martin, A. Ibrahim, A. Edwards, R. Aquino-Santos, "Combating Child Exploitation in Second Life", in *IEEE International Conference on Science and Technology for Humanity*, pp.761-766, Sept. 2009. (Cited on page 6.)

[15] E. Candés, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information", *IEEE Trans. on Inf. Th.*, Vol. 52, pp. 489–509, Feb. 2006. (Cited on pages 9, 84, 85, 86 and 87.)

[16] D. Donoho, "Compressive sensing", *IEEE Trans. on Inf. Th.*, Vol. 52, No. 4, pp. 1289–1306, April 2006. (Cited on pages 9, 83, 84, 85, 86, 101, 107 and 116.)

[17] C. Feng, S. Valaee and Z. Tan, "Multiple target localization using compressive sensing," in *Proc. IEEE GLOBECOM'09*, Hawaii, USA, 4 Nov.–5 Dec. 2009. (Cited on page 9.)

[18] J. Allan, "Topic detection and tracking: event-based information organization", in *Kluwer Academic Publishers*, Norwell, MA, USA, 2002. (Cited on page 14.)

[19] S. Phuvipadawat and T. Murata, "Breaking news detection and tracking in Twitter", in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 3, pp. 120-123, 2010. (Cited on pages 14 and 30.)

[20] B. O'Connor, M. Krieger and D. Ahn, "TweetMotif:ExploratorySearch and Topic Summarization for Twitter", in *4th International AAAI Conference on*

Web and Social Media, W. W. Cohen, S. Gosling, W. W. Cohen, and S. Gosling, Eds. The AAAI Press, 2010. (Cited on pages 15, 19 and 23.)

[21] H. Becker, M. Naaman and L. Gravano, "Beyond Trending Topics: Real-World Event Identification on Twitter", in *5th International AAAI Conference on Web and Social Media*, 2011. (Cited on page 15.)

[22] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman and J. Sperling, "TwitterStand: News in Tweets", in *17th ACM International Conference on Advances in Geographic Information Systems*, pp. 42P51, New York, NY, USA, 2009. (Cited on page 15.)

[23] G. P. C. Fung, J. X. Yu, P. S. Yu and H. Lu, "Parameter free bursty events detection in text streams", in *31st International Conference on Very Large Data Bases*, VLDB Endowment, pp. 181–192, 2005. (Cited on pages 15 and 16.)

[24] S. Petrovic, M. Osborne and V. Lavrenko, "Streaming First Story Detection with Application to Twitter", in *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 181P189, Stroudsburg, PA, USA, 2010. (Cited on pages 15 and 20.)

[25] D. M. Blei, A. Y. Ng and M. I. Jordan, "Latent Dirichlet Allocation", in *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, March 2003. (Cited on pages 16 and 21.)

[26] D. M. Blei and J. D. Lafferty, "Dynamic topic models," in *23rd ACM International Conference on Machine Learning*, pp. 113–120, New York, NY, USA, 2006. (Cited on page 16.)

[27] D.A. Shamma, L. Kennedy and E.F. Churchill,"Peaksandpersistence: Modeling the Shape of Microblog Conversations", in *ACM Conference on Computer Supported Cooperative Work*, pp. 355–358, New York, NY, USA, 2011. (Cited on pages 16 and 31.)

[28] J. Yang and J. Leskovec, "Patterns of temporal variation in online media", in *4th ACM international conference on Web search and data mining*, pp. 177P186, New York, NY, USA, 2011. (Cited on page 16.)

[29] J. Lehmann, B. Goncalves, J. J. Ramasco and C. Cattuto, "Dynamical Classes of Collective Attention in Twitter", in *21st ACM International Conference on World Wide Web (WWW)*, pp. 251–260, New York, NY, USA, 2012. (Cited on page 16.)

[30] M. Mathioudakis and N. Koudas, "TwitterMonitor: Trend Detection over the Twitter Stream", in *International Conference on Management of Data (SIGMOD)*, pp. 1155–1158, New York, NY, USA, 2010. (Cited on page 16.)

[31] H. Sayyadi, M. Hurst and A. Maykov, "Event detection and tracking in social streams", in *3rd International AAAI Conference on Web and Social Media*, E. Adar, M. Hurst, T. Finin, N. S. Glance, N. Nicolov, and B. L. Tseng, Eds. The AAAI Press, 2009. (Cited on page 17.)

[32] J. Leskovec, L. Backstrom and J. Kleinberg, "Meme-Tracking and the Dynamics of the News Cycle", in *15th ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 497P506, New York, NY, USA, 2009. (Cited on page 17.)

[33] S. Papadopoulos, Y. Kompatsiaris and A. Vakali, "A graph-based clustering scheme for identifying related tags in folksonomies", in *12th Interna-*

tional Conference on Data Warehousing and Knowledge Discovery, pp. 65–76, Berlin, Heidelberg, 2010. (Cited on page 17.)

[34] J.Weng and B.-S.Lee,"EventDetectioninTwitter", in *5th International Conference on Weblogs and Social Media*. The AAAI Press, 2011. (Cited on page 17.)

[35] Q. He, K. Chang and E.-P. Lim, "Analyzing feature trajectories for event detection", in *30th Annual International ACM Conference on Research and Development in Information Retrieval*, pp. 207–214, New York, NY, USA, 2007. (Cited on page 17.)

[36] M. Cataldi, L. Di Caro and C. Schifanella, "Emerging Topic Detection on Twitter Based on Temporal and Social Terms Evaluation", in *10th International Workshop on Multimedia Data Mining*, pp. 4:1–4:10, New York, NY, USA, 2010. (Cited on page 18.)

[37] S. Diplaris, G. Petkos, S. Papadopoulos, Y. Kompatsiaris, N. Sarris, C. Martin, A. Goker, D. Corney, J. Geurts, Y. Liu and J.-C. Point, "SocialSensor: Surfacing real-time trends and insights from multiple social networks", in *Proc. of the 2012 NEM Summit*, pp. 47P52, Oct 2012. (Cited on page 19.)

[38] M. F. Porter, "Readings in Information Retrieval", in *Morgan Kaufmann Publishers Inc., Eds. San Francisco* , ch. An algorithm for suffix stripping, pp. 313P316, 1997. (Cited on page 20.)

[39] Y. W. Teh, D. Newman and M. Welling, "A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation", in *Advances in Neural Information Processing Systems*, vol. 19, 2007. (Cited on page 21.)

[40] Y. W. Teh, M. I. Jordan, M. J. Beal and D. M. Blei, "Hierarchical Dirichlet processes", in *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1566–1581, 2006. (Cited on page 21.)

[41] G. Salton and M. J. McGill, "Introduction to Modern Information Retrieval", in *McGraw-Hill*, New York, NY, USA, 1986. (Cited on page 21.)

[42] X. Xu, N. Yuruk, Z. Feng and T. A. J. Schweiger, "SCAN: a Structural Clustering Algorithm for Networks", in *13th ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 824–833, New York, NY, USA, 2007. (Cited on page 23.)

[43] B. Goethals, "Frequent Set Mining", pp. 377–397, 2005. (Cited on page 25.)

[44] C. Gyorodi and R. Gyorodi, "A Comparative Study of Association Rules Mining Algorithms", 2004. (Cited on page 25.)

[45] H. Li, Y. Wang, D. Zhang, M. Zhang and E. Y. Chang, "PFP: Parallel FP-growth for Query Recommendation", in *ACM Conference on Recommender Systems*, pp. 107–114, New York, NY, USA, 2008. (Cited on page 25.)

[46] J. R. Finkel, T. Grenager and C. Manning, "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling", in *43rd Annual Meeting on Association for Computational Linguistics*, pp. 363–370, Stroudsburg, PA, USA, 2005. (Cited on page 30.)

[47] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms", in *The Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983. (Cited on page 30.)

[48] R. Baraniuk, "Compressive sensing", *IEEE Signal Processing Magazine*, vol. 24, pp. 118–121, July 2007. (Cited on page 84.)

[49] M. Grewal and A. Andrews, "Kalman filtering: Theory and practice using MATLAB", 2nd Edition, 2001. (Cited on page 91.)

[50] V. Becher and P. A. Heiber, A better complexity of finite sequences, Abstracts of the 8th *Int. Conf. on Computability and Complexity in Analysis* and 6th *Int. Conf. on Computability, Complexity, and Randomness* , Cape Town, South Africa, January 31, February 4, 2011, p. 7. (Cited on page 35.)

[51] Ilie, L., Yu, S., and Zhang, K. Repetition Complexity of Words In *Proc. COCOON* 320–329, 2002. (Cited on page 35.)

[52] Li, M., and Vitanyi, P. *Introduction to Kolmogorov Complexity and its Applications.* Springer-Verlag, Berlin, Aug. 1993. (Cited on page 35.)

[53] Niederreiter, H., Some computable complexity measures for binary sequences, in *Sequences and Their Applications*, Eds. C. Ding, T. Hellseth and H. Niederreiter Springer Verlag, 67-78, 1999. (Cited on page 35.)

[54] P. Jacquet, "Common words between two random strings", *IEEE International Symposium on Information Theory*, 1495-1499, 2007. (Cited on pages 36 and 40.)

[55] J. Ziv, "On classification with empirically observed statistics and universal data compression", *IEEE Transactions on Information Theory*, 34, 278-286, 1988. (Cited on page 37.)

[56] S. Janson, S. Lonardi and W. Szpankowski, "On Average Sequence Complexity", *Theoretical Computer Science*, 326, 213-227, 2004. (Cited on pages 35, 37, 40 and 42.)

[57] W. Szpankowski, *Analysis of Algorithms on Sequences*, John Wiley, New York, 2001. (Cited on pages 42, 49 and 50.)

[58] P. Jacquet, and W. Szpankowski, "Autocorrelation on Words and Its Applications. Analysis of Suffix Trees by String-Ruler Approach", *J. Combinatorial Theory Ser. A*, 66, 237–269, 1994. (Cited on pages 42, 46, 47 and 54.)

[59] P. Jacquet, W. Szpankowski and J. Tang, "Average profile of the Lempel-Ziv parsing scheme for a Markovian source", in *Algorithmica*, 31(3), 318–360, 2001. (Cited on page 54.)

[60] R. Neininger and L. Rüschendorf, "A general limit theorem for recursive algorithms and combinatorial structures", in *The Annals of Applied Probability*, 14(1), 378-418, 2004. (Cited on page 54.)

[61] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985. (Cited on page 44.)

[62] J. Fayolle and M. D. Ward, "Analysis of the Average Depth in a Suffix Tree Under a Markov Model", in *International Conference on the Analysis of Algorithms (AofA)*, pp. 95–104, Barcelona, Spain, 2005. (Cited on pages 46 and 47.)

[63] P. Jacquet, and W. Szpankowski, "Analytical DePoissonization and Its Applications", *Theoretical Computer Science*, 201, 1–62, 1998. (Cited on page 49.)

[64] P. Jacquet and W. Szpankowski, "Joint String Complexity for Markov Sources", *23rd International Meeting on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms*, AofA'12, *DMTCS Proc.*, 303-322, Montreal, 2012. (Cited on pages 36, 37, 43, 49, 51 and 53.)

[65] P. Jacquet, and W. Szpankowski, "Analytic Pattern Matching: From DNA to Twitter", *Cambridge University Press*, 2015. (Cited on page 47.)

[66] P. Flajolet, X. Gourdon, P. Dumas, "Mellin Transforms and Asymptotics : Harmonic Sums", in *Theoretical Computer Science*, Vol. 144 (1-2), pp. 3-58, 1995. (Cited on page 49.)

[67] P. Flajolet and R. Sedgewick, *Analytic Combinatorics*, *Cambridge University Press*, Cambridge, 2008. (Cited on pages 50 and 53.)

[68] S. Papadopoulos, D. Corney and L. Aiello, "SNOW 2014 Data Challenge: Assessing the Performance of News Topic Detection Methods in Social Media", *Proceedings of the SNOW 2014 Data Challenge*, 2014. (Cited on pages 64, 65, 66, 71, 74 and 95.)

[69] G. Burnside, **D. Milioris** and P. Jacquet, "One Day in Twitter: Topic Detection Via Joint Complexity", *in Snow Data Challenge, International World Wide Web Conference (WWW'14), Seoul, South Korea*, April 2014. (Cited on page 65.)

[70] S. Tata, R. Hankins and J. Patel, "Practical Suffix Tree Construction", in *30th VLDB Conference*, 2004. (Cited on page 58.)

[71] S. Nilsson and M. Tikkanen, "Implementing a Dynamic Compressed Trie", in *Algorithm Engineering*, 1998. (Cited on page 58.)

[72] 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. (Cited on pages 43, 71 and 84.)

[73] **D. Milioris** and P. Jacquet, "Joint Sequence Complexity Analysis: Application to Social Networks Information Flow", *in Bell Laboratories Technical Journal, Issue on Data Analytics*, Vol. 18, No. 4, 2014. (Cited on pages 43, 71 and 84.)

[74] **D. Milioris** and P. Jacquet, "Method and Device for Classifying a Message", **European Patent No. 13306222.4**, Filed September 6, 2013. (Cited on page 71.)

[75] L. M. Aiello *et al.*, "Sensing Trending Topics in Twitter", in *IEEE Transactions on Multimedia*, Vol. 15, Iss. 6, pp. 1268–1282, Oct 2013. (Cited on pages 75 and 95.)

[76] **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. (Cited on pages 102 and 116.)

[77] P. Mirowski, **D. Milioris**, P. Whiting and T. Kam Ho, "Probabilistic RF Fingerprinting and Localization on the Run", *in Bell Laboratories Technical Journal, Issue on Data Analytics*, Vol. 18, No. 4, 2014. (Cited on page 102.)

[78] **D. Milioris** and P. Jacquet, "Classification in Twitter via Compressive Sensing", *in IEEE International Conference on Computer Communications (INFOCOM'15), Hong Kong, SAR China*, Short Paper, April 2015. (Cited on page 84.)

[79] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. on Sci. Comp.*, Vol. 20, No. 1, pp. 33–61, 1999. (Cited on page 87.)

[80] R. Tibshirani,"Regression shrinkage and selection via the lasso," *J. of the Royal Stat. Soc. Series B (Methodological)*, Vol. 58, No. 1, pp. 267–288, 1996. (Cited on page 87.)

[81] J. Tropp, and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. on Inf. Th.*, Vol. 53, pp. 4655–466, Dec. 2007. (Cited on page 87.)

[82] D. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," Tech. Rep., Stanford 2006 [on-line: `http://www-stat.stanford.edu/~donoho/Reports/2006/StOMP-20060403.pdf`] (Cited on page 87.)

[83] **D. Milioris**, "Text Classification Based on Joint Complexity and Compressed Sensing", **United States Patent No. 14/540770**, Filed November 13, 2014. (Cited on page 93.)

[84] G. Tzagkarakis and P. Tsakalides, "Bayesian compressed sensing imaging using a Gaussian scale mixture", in *35th Int. Conf. on Acoustics, Speech, and Sig. Proc. (ICASSP'10)*, Dallas, TX, Mar. 2010. (Cited on pages 96, 112 and 118.)

[85] P. Bahl and V. Radmanabhan, "An in-building RF-based user location and tracking system", in *19th IEEE Int. Conf. on Computer Com. (INFOCOM'00)*, Tel-Aviv, Israel, Mar. 2000. (Cited on pages 101 and 103.)

[86] R. Want *et al.*, "The active badge location system", *ACM Trans. on Information Systems*, Vol. 10, No. 1, pp. 91–102, Jan. 1992. (Cited on page 101.)

[87] N. Priyantha, A. Chakraborty and H. Balakrishnan, "The cricket location-support system", in *6th ACM Int. Conf. on Mob. Comp. and Net. (MOBICOM'00)*, Boston, MA, Aug. 2000. (Cited on page 101.)

[88] U. Bandara *et al.*, "Design and implementation of a bluetooth signal strength based location sensing system", in *IEEE Radio and Wireless Conf. (RAWCON'04)*, Atlanta, GA, Sep. 2004.

[89] M. Azizyan, I. Constandache and R. Choudhury, "SurroundSense: Mobile phone localization via ambience fingerprinting", in *15th ACM Int. Conf. on Mob. Comp. and Net. (MOBICOM'09)*, Beijing, China, Sep. 2009. (Cited on page 101.)

[90] J. Liu and V. Issarny, "Signal strength based service discovery (S3D) in mobile ad hoc networks", in *IEEE Personal Indoor and Mobile Radio Communications (PIMRC)* pp. 811–815, 2005. (Cited on page 101.)

[91] A. Ladd *et al.*, "Robotics-based location sensing using wireless ethernet", in *8th ACM Int. Conf. on Mob. Comp. and Net. (MOBICOM'02)*, Atlanta, GA, Sep. 2002. (Cited on page 101.)

[92] C. Fretzagias and M. Papadopouli, "Cooperative location sensing for wireless networks", in *IEEE Conf. on Pervasive Comp. and Com. (PERCOM'04)*, Orlando, FL, Mar. 2004. (Cited on page 103.)

[93] B. Li *et al.*, "Indoor positioning techniques based on wireless LAN", in *Int. Conf. on Wireless Broadband and Ultra Wideband Com. (AUSWIRELESS'06)*, Sydney, Australia, Mar. 2006. (Cited on page 103.)

[94] X. Chai and Q. Yang, "Reducing the calibration effort for probabilistic indoor location estimation", in *IEEE Transactions on Mobile Computing*, Vol. 6 , No. 6, pp. 649–662, June 2007. (Cited on pages 103 and 112.)

[95] I. Guvenc *et al.*, "Enhancements to RSS based indoor tracking systems using Kalman filters", in *Proc. Global Signal Processing Exposition (GSPx)*, Dallas, TX, Apr. 2003. (Cited on page 103.)

[96] A. Bekkali, H. Sanson and M. Matsumoto, "RFID indoor positioning based on probabilistic RFID map and Kalman filtering", in *3rd IEEE Int. Conf. on Wireless and Mob. Comp., Net. and Com. (WIMOB'07)*, New York, NY, Oct. 2007. (Cited on page 104.)

[97] A. Paul and E. Wan, "RSSI-based indoor localization and tracking using sigma-point Kalman smoothers", in *Journal on Selected Topics in Sig. Proc.*, Vol. 3, No. 5, pp. 860–873, 2009. (Cited on page 104.)

[98] A. Au *et al.*, "Indoor tracking and navigation using received signal strength and compressive sensing on a mobile device", in *IEEE Transactions on Mobile Computing*, Is. 99, Aug. 2012 (DOI: 10.1109/TMC.2012.175). (Cited on page 104.)

[99] C. H. Chao *et al.*, "Location-constrained particle filter human positioning and tracking system", in *3rd IEEE Workshop on Sig. Proc. Syst. (SiPS'08)*, Washington, DC, Oct. 2008. (Cited on pages 104 and 105.)

[100] **D. Milioris**, M. Bradonjić and P. Mühlethaler, "Building Complete Training Maps for Indoor Location Estimation", *in IEEE International Conference on Computer Communications (INFOCOM'15), Hong Kong, SAR China*, Short Paper, April 2015. (Cited on page 105.)

[101] H. Wang *et al.*, "WLAN-based pedestrian tracking using particle filters and low-cost MEMS sensors", in *4th Work. on Posit., Nav. and Com. (WPNC'07)*, Hanover, Germany, Mar. 2007. (Cited on page 106.)

[102] Y. Mezali, P. Jacquet and G. Rodolakis, "A path tracking algorithm using the IEEE 802.11 Infrastructure", in *14th Int. Symp. on Wireless Personal Multimedia Communications (WPMC'11)*, Brest, France, Oct. 2011. (Cited on page 105.)

[103] **D. Milioris** *et al.*, "Empirical evaluation of signal-strength fingerprint positioning in wireless LANs", in *13th ACM Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'10)*, Bodrum, Turkey, Oct. 2010. (Cited on pages 105 and 111.)

[104] **D. Milioris**, "Method, User Equipment, System and Computer Readable Medium for Localizing a User Equipment", **European Patent No. 14306453.3**, Filed September 22, 2014. (Cited on pages 106 and 112.)

[105] **D. Milioris**, G. Tzagkarakis, P. Jacquet and P. Tsakalides, "Indoor positioning in wireless LANs using compressive sensing signal-strength fingerprints", in *19th Europ. Sig. Proc. Conf. (EUSIPCO'11)*, Barcelona, Spain, Aug. 2011. (Cited on page 106.)

[106] C. Feng *et al.*, "Indoor navigation system on handheld devices for the visually impaired ", in *Canadian National Institute Conference for the Blind (CNIB'11)*, Canada, 2011. (Cited on pages 106 and 107.)

[107] **D. Milioris**, G. Tzagkarakis, P. Tsakalides and P. Jacquet, "WLAN-based Indoor Path-Tracking using Compressive RSS Measurements", *in 21st Eu-*

ropean Signal Processing Conference (EUSIPCO'13), Marrakech, Morocco, September 2013. (Cited on page 109.)

[108] **D. Milioris** and P. Jacquet, "A Statistical Region-based Compressive Sensing Indoor Path-Tracking System", *in Proc. 12th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'11), Short Paper, Paris, France*, May 2011. (Cited on page 109.)

(Cited on page 109.)

[109] **D. Milioris** and P. Jacquet, "SecLoc: Encryption System Based on Compressive Sensing Measurements for Location Estimation", *in IEEE International Conference on Computer Communications (INFOCOM'14), Toronto, Canada*, Short Paper, May 2014. (Cited on page 115.)

[110] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing", in *IEEE Transactions on Signal Processing*, 2008. (Cited on page 118.)

[111] G. Tzagkarakis, **D. Milioris** and P. Tsakalides, "Multiple-Measurement Bayesian Compressive Sensing using GSM Priors for DOA Estimation", in *Proc. 35th IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP'10)*, Dallas, TX, USA, March 2010. (Cited on page 118.)

[112] P. Jacquet and W. Szpankowski, *Markov Types and Minimax Redundancy for Markov Sources*, in IEEE Transactions on Information Theory, Vol. 50, No. 7, July 2004. (Cited on pages 123, 129 and 131.)

[113] P. Jacquet and **D. Milioris**, "Information in Strings: Enumeration of Eulerian Paths and Eulerian Components in Markov Sequences", *in International Conference on Probabilistic, Combinatorial and Asymptotic Methods*

for the Analysis of Algorithms (AofA'14), Paris, France, June 2014. (Cited on pages 134 and 135.)

[114] P. Jacquet and **D. Milioris**, "HoneyPot Design via Random Eulerian Paths", **European Patent No. 14306779.1**, Filed November 6, 2014. (Cited on page 134.)

[115] P. Jacquet, **D. Milioris** and G. Burnside, "Textual Steganography - Undetectable Encryption based on N-Gram Rotations", **European Patent No. 14306482.2**, Filed September 25, 2014. (Cited on page 134.)

[116] P. Jacquet and **D. Milioris**, "Undetectable Encryption based on M-grams Permutations", **European Patent No. 14305064.9**, Filed January 17, 2014. (Cited on page 134.)

[117] W. T. Tutte, C. A. B. Smith, *On unicursal paths in a network of degree 4*, American Mathematical Monthly, Vol. 48, pp. 233–237, 1941, JSTOR 2302716.1941. (Cited on page 122.)

[118] T. van Aardenne-Ehrenfest, N. G. de Bruijn, *Circuits and trees in oriented linear graphs"*, Simon Stevin Vol. 28, pp. 203–217, 1951. (Cited on page 122.)

[119] B. McKay and R. W. Robinson, *Asymptotic enumeration of Eulerian circuits in the complete graph*, Combinatorica, no. 4, 10, pp. 367–377, 1995. (Cited on page 123.)

[120] M. I. Isaev, *Asymptotic number of Eulerian circuits in complete bipartite graphs*, in Proc. of 52nd MFTI Conference, Moscow, 2009. (Cited on page 123.)

[121] National Bureau of Standards, *Data Encryption Standard*, FIPS-Pub.46., U.S. Department of Commerce, Washington D.C., January ,1977. (Cited on page 126.)

[122] J. Daemen, V. Rijmen, *The Design of Rijndael: AES P The Advanced Encryption Standard*, Springer, 2002. (Cited on page 126.)

[123] D. J. Wheeler, R. M. Needham, "TEA, a tiny encryption algorithm", in *2nd International Workshop on Fast Software Encryption, Lecture Notes in Computer Science*, Leuven, Belgium, 1008: pp. 363–366, 1994. (Cited on page 126.)

[124] C. Paar, J. Pelzl, *Understanding Cryptography, A Textbook for Students and Practitioners*, Springer, 2009. (Cited on page 126.)

Appendix A

Suffix Trees

A.1 Suffix Tree Construction

A suffix tree is composed of nodes, which can have three types:

- internal to the tree, in which case they contain only a list of children (or equivalently outgoing edges), leaves, in which case they contain a pointer to their beginning position in the original string (to save memory space as opposed to storing the whole substring for each leaf). In order to reconstruct the suffix, we simply build a string from the noted position to the end of the original string.
- a special kind of leaf, noted epsilon, which is an empty leaf denoting the end of the string.
- the list of children can be represented as a Map<Character, Node>, where the Character key is the label of the outgoing edge and the value Node is the other extremity of the outgoing edge.

The construction of a Suffix Tree follows:

- a root node is created. We start from the end of the string and add suffixes from each position until we reach the beginning of the string after which time the Suffix Tree construction is complete.

- adding a suffix takes two parameters, the position of the suffix and the character leading to this position, and may have two outcomes:
  - if there was no previous edge with the given label, we simply create the edge and make it point to the given position.
  - if there was already an edge with the same label, then we need to start walking down this edge (and moving along also in the suffix we are trying to add) until we reach a position where the child we are trying to add is not already present, in which case we simply add the edge at the correct sublevel. If at any point during this process we reach a leaf node, we need to expand it and create the corresponding internal nodes as long as the two substrings coincide and sprout new leaves as soon as they differ.

A.2 Suffix Trees Superposition

In order to compute the Joint Complexity of two sequences, we simply need their two respective Suffix Trees.

Comparing Suffix Trees can be viewed as a recursive process which starts at the root of the trees and walks along both trees simultaneously. When comparing subparts of the trees we can face three situations:

- both parts of the trees we are comparing are leaves. A leaf is basically a string representing the suffix. Comparing the common factors of two strings can be done easily by just incrementing a counter each time the characters of both strings are equal and stop counting as soon as they differ. For example comparing the two suffixes “nalytics” and “nanas” would give a

result of 2 as they only share the first two characters; while comparing suffixes "nalytics" and "ananas" would return 0 as they do not start with the same character.

- one subpart is a leaf while the other subpart is an internal node (*i.e.* a branch). Comparing a non-leaf node and a leaf is done by walking through the substring of the leaf and incrementing the score as long as there is an edge whose label corresponds to the current character of the leaf. Note that the keys (edge's labels) are sorted so that we can stop looking for edges as soon as an edge is sorted after the current character (allowing average sublinear computation). When we reach a leaf on the subtree side, we just need to compare the two leaves (starting from where we are at in the substring and the leaf that we just reached).

- both parts of the trees we are comparing are internal nodes (*i.e.* we are trying to compare two branches). Comparing two non-leaf nodes is done by initializing a current counter to zero and walking through both subtrees while doing the following at each step:

  - first check whether we have reached the end (epsilon) of one of the subtrees, in which case we can stop comparing and return the current counter.

  - then check whether we have reached two leaves at the same time, in which case we add the current counter to the output of the previously described method for comparing two leaves.

  - check whether we have reached a leaf on one side only, in which case we add the current counter to the output of the previously described method for comparing a leaf with an internal node.

– keep walking through subtrees as long as we find identical edges (and call ourselves recursively), each time incrementing our internal counter.