***Mining user activity data in
social media services***

Alceu Ferraz Costa
Tese de Doutorado

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Alceu Ferraz Costa

Mining user activity data in social media services¹

Doctoral dissertation submitted to the Instituto de Ciências Matemáticas e de Computação - ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Profa. Dra. Agma Juci Machado Traina
Coadvisor: Prof. Dr. Christos Faloutsos

USP – São Carlos
July 2017

¹ This PhD work was funded by FAPESP – Process 2012/00005-0

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

F368m Ferraz Costa, Alceu
Mining User Activity Data in Social Media
Services / Alceu Ferraz Costa; orientadora Agma
Juci Machado Traina; coorientador Christos
Faloutsos. -- São Carlos, 2017.
93 p.

Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2017.

1. Social Media. 2. Data Mining. 3. User
Modeling. I. Juci Machado Traina, Agma, orient. II.
Faloutsos, Christos, coorient. III. Título.

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Alceu Ferraz Costa

Mineração de dados de atividade de usuários em serviços de mídia social

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Agma Juci Machado Traina
Co-Orientador: Prof. Dr. Christos Faloutsos

USP – São Carlos
Julho de 2017

Agradecimentos

Gostaria de agradecer, primeiramente, à minha esposa Lourdes, que compartilhou comigo todos os momentos desse doutorado. *Gracias por todo.* Agradeço também a minha família: pai, mãe e meus irmãos Lucas e Tiago, que sempre apoiaram e incentivaram meus estudos. Essa tese é resultado de toda atenção e carinho que sempre recebi. Agradeço à minha orientadora, Profa. Agma Traina, que me apresentou à pesquisa quando eu ainda era aluno de graduação. Esse trabalho só foi possível por seu apoio, ensinamentos e orientação. Agradeço também ao meu coorientador, Prof. Christos Faloutsos, por todas as reuniões, conselhos e por ter me recebido tão bem na CMU. Também agradeço aos meus amigos do ICMC-USP e do GBdI, não somente pelas conversas e sugestões ao longo desses anos, mas também por terem proporcionado muitos momentos alegres que certamente irei guardar para sempre comigo. Por fim, agradeço a FAPESP, pelo apoio financeiro.

Abstract

Social media services have a growing impact in our society. Individuals often rely on social media to get their news, decide which products to buy or to communicate with their friends. As consequence of the widespread adoption of social media, a large volume of data on how users behave is created every day and stored into large databases. Learning how to analyze and extract useful knowledge from this data has a number of potential applications. For instance, a deeper understanding on how legitimate users interact with social media services could be explored to design more accurate spam and fraud detection methods.

This PhD research is based on the following hypothesis: data generated by social media users present patterns that can be exploited to improve the effectiveness of tasks such as prediction, forecasting and modeling in the domain of social media. To validate our hypothesis, we focus on designing data mining methods tailored to social media data. The main contributions of this PhD can be divided into three parts. First, we propose Act-M, a mathematical model that describes the timing of users actions. We also show that Act-M can be used to automatically detect bots among social media users based only on the timing (i.e. time-stamp) data. Our second contribution is VnC (Vote-and-Comment), a model that explains how the volume of different types of user interactions evolve over time when a piece of content is submitted to a social media service. In addition to accurately matching real data, VnC is useful, as it can be employed to forecast the number of interactions received by social media content. Finally, our third contribution is the MFS-Map method. MFS-Map automatically provides textual annotations to social media images by efficiently combining visual and metadata features. Our contributions were validated using real data from several social media services. Our experiments show that the Act-M and VnC models provided a more accurate fit to the data than existing models for communication dynamics and information diffusion, respectively. MFS-Map obtained both superior precision and faster speed when compared to other widely employed image annotation methods.

Keywords: Social Media, Data Mining, User Modeling

Resumo

O impacto dos serviços de mídia social em nossa sociedade é crescente. Indivíduos frequentemente utilizam mídias sociais para obter notícias, decidir quais os produtos comprar ou para se comunicar com amigos. Como consequência da adoção generalizada de mídias sociais, um grande volume de dados sobre como os usuários se comportam é gerado diariamente e armazenado em grandes bancos de dados. Aprender a analisar e extrair conhecimentos úteis a partir destes dados tem uma série de potenciais aplicações. Por exemplo, um entendimento mais detalhado sobre como usuários legítimos interagem com serviços de mídia social poderia ser explorado para projetar métodos mais precisos de detecção de spam e fraude.

Esta pesquisa de doutorado baseia-se na seguinte hipótese: dados gerados por usuários de mídia social apresentam padrões que podem ser explorados para melhorar a eficácia de tarefas como previsão e modelagem no domínio das mídias sociais. Para validar esta hipótese, foram projetados métodos de mineração de dados adaptados aos dados de mídia social. As principais contribuições desta pesquisa de doutorado podem ser divididas em três partes. Primeiro, foi desenvolvido o Act-M, um modelo matemático que descreve o tempo das ações dos usuários. O autor demonstrou que o Act-M pode ser usado para detectar automaticamente *bots* entre usuários de mídia social com base apenas nos dados de tempo. A segunda contribuição desta tese é o VnC (*Vote-and-Comment*), um modelo que explica como o volume de diferentes tipos de interações de usuário evolui ao longo do tempo quando um conteúdo é submetido a um serviço de mídia social. Além de descrever precisamente os dados reais, o VnC é útil, pois pode ser empregado para prever o número de interações recebidas por determinado conteúdo de mídia social. Por fim, nossa terceira contribuição é o método MFS-Map. O MFS-Map fornece automaticamente anotações textuais para imagens de mídias sociais, combinando eficientemente características visuais e de metadados das imagens. As contribuições deste doutorado foram validadas utilizando dados reais de diversos serviços de mídia social. Os experimentos mostraram que os modelos Act-M e VnC forneceram um ajuste mais preciso aos dados quando comparados, respectivamente, a modelos existentes para dinâmica de comunicação e difusão de informação. O MFS-Map obteve precisão superior e tempo de execução reduzido quando comparado com outros métodos amplamente utilizados para anotação de imagens.

Palavras-chave: Mídia Social, Mineração de Dados, Modelagem de Usuários

List of Symbols

General

$P(A)$	Probability of an event A.
$f(x)$	Probability density function (PDF).
$F(x)$	Cumulative distribution function (CDF).
$\mathcal{X}$	Training set.
c_{FP}	False Positive error cost.
c_{FN}	False Negative error cost.
$\mathcal{U}$	Social media user.

Communication Dynamics

$T = \{t_1, t_2, \dots\}$	Sequence of time-stamps.
Δ	Inter-arrival time (IAT).
$\Delta = \{\Delta_1, \Delta_2, \dots\}$	Sequence of inter-arrival times.
$\Delta_{\mathcal{U}} = \{\Delta_1, \Delta_2, \dots\}$	Sequence of IAT from the social media user $\mathcal{U}$.
$\hat{\Delta} = \{\Delta_1, \Delta_2, \dots\}$	Synthetic sequence of inter-arrival times
θ	Set of model parameters.
ρ	SCorr correlation parameter.
λ_o	SCorr base rate parameter.
δ	State duration of the Act-M model.
p_a	Probability of entering the active state (Act-M).
p_r	Probability of entering the rest state (Act-M).
λ_A, ρ_A	SCorr parameters for the active state.
λ_R, ρ_R	SCorr parameters for the rest state.
t_{wake}	Act-M wake-up time.
t_{sleep}	Act-M sleep time.

t_{clock}	Act-M clock variable.
r_{sleep}	Fraction of day considered sleep time by Act-M.
D	Consecutive IAT pair-counting matrix.

Social Voting

$v_+(t)$	Number of up-votes received exactly at time t .
$v_-(t)$	Number of down-votes received exactly at time t .
$c(t)$	Number of comments received exactly at time t .
$V_+(t), V_-(t), C(t)$	Up-votes, down-votes and comments accumulated at time t .
Δ_R	Reaction time.
P_{Like}	Probability that a user will like a submission.
P_{React}	Probability that a user will react to a submission.
ξ	Independent coefficient (VnC).
β	Cascading coefficient (VnC).
N	Population of potential voters (VnC).
t_s	Time instant in which a submission is shared (VnC).

Image Annotation and Association Rules

$\mathcal{I}$	Set of all items that may appear in a database.
$\{j_1, j_2, \dots\}$	Itemset (a set of items such that $j_i \in \mathcal{I}$).
I	Image.
$L = \{\ell_1, \ell_2, \dots\}$	Set of textual tags.
$\mathcal{I} = (I, L)$	Social media image tuple (image I and textual tags L).
$A = \{a_1, a_2, \dots\}$	Set of image annotations.
ε	Extraction function
$\mathbf{v} = (v_1, v_2, \dots)$	Feature vector.
$\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \dots\}$	Set of feature vectors.

Acronyms

Act-M Activity Model (model)

AIA Automatic Image Annotation

API Application Programming Interface

CAR Classification Association Rules

CDF Cumulative Distribution Function

CNPP Cascading Non-homogeneous Poisson Process

IAT Inter Arrival Time

KDD Knowledge Discovery in Databases

MFS-Map Multi-Feature Space Map

PDF Probability Distribution Function

RMSE Root-mean-square Error

SCorr Self-Correlated Process

SI Susceptible-Infected (model)

TID Transaction Identifier

VnC Vote-and-Comment (model)

Contents

Agradecimientos	i
Abstract	iii
Resumo	v
List of Symbols	vii
Acronyms	xi
1 Introduction	1
1.1 Motivation	1
1.2 Problem Statement	2
1.3 Contributions	3
1.4 Summary	3
2 Background and Related Work	5
2.1 Social Media Services	5
2.2 Modeling User Behavior	9
2.3 Knowledge Discovery in Social Media Data	13
2.4 Automatic Image Annotation	17
2.5 Conclusions	18
3 Dynamics of Social Media Communications	21
3.1 Introduction	21
3.2 Problem Definition	23
3.3 Observations on the Dynamics of Social Media Communications	23
3.4 The Act-M Model	30
3.5 Experiments	34
3.6 Detecting Bots	43
3.7 Conclusions	46
4 Coevolution of User Interactions	49
4.1 Introduction	49
4.2 Definitions	50
4.3 Activity in Social Voting Services: Observations	51

4.4	The VnC Model	55
4.5	Experiments	57
4.6	Applications	64
4.7	Conclusions	69
5	Automatic Annotation of Social Media Images	71
5.1	Introduction	71
5.2	Problem Definition	73
5.3	Proposed Method: MFS-Map	73
5.4	Experiments	76
5.5	Conclusions	80
6	Conclusions	81
6.1	Contributions of this PhD Work	81
6.2	Future Work	83
6.3	List of Produced Publications	84
	Bibliography	85

Chapter 1

Introduction

Estimates indicate that in 2016, over 68% of US adults [41] and over 47% of Brazilians [56] are users of at least one social media service. This growing popularity of social media has changed many aspects of our society. For example, among young US adults, TV news consumption is decreasing while a growing part of the population rely on social media feeds to get their news [75]. The advertising industry has also been affected. Since social media services often have detailed information on their users, advertisers can target a specific audience based on age, location and interests. As a result, the money spent on paid advertisement in social media is projected to rise significantly, from \$8.5 billion in 2014 to \$14 billion in 2018 [6]. Additionally, content submitted to social media has the potential of becoming viral and companies often try explore this when marketing their brands or products. Social media has also played a significant role in the Brazil protests of 2013 and 2016 [13, 85]. Movements often employed social media to recruit participants and increase population mobilization. Social media has also changed the way individuals communicate [111], how companies market their products [1], and even how professionals find their jobs [9, 105].

1.1 Motivation

The pervasiveness of social media has created challenges and opportunities. Undesired activity such as spam and spread of false news can be used by attackers to obtain illegal monetary gain, to deceive users or to damage the reputation of individuals, products or organizations [82, 92]. At the same time, an unprecedented volume of human behavior data is readily available to researchers. Learning how to understand, analyze and extract useful knowledge from all of this data has a number of potential applications. For example, if we were able to understand how legitimate users' interact with social media services, this knowledge could be used to improve spam and fraud detection methods. Popularity of content in social media services strongly depends on user interaction. For instance, when users share news articles with their Twitter followers, those articles are more likely to attract even more attention. Learning how content attracts these users' interactions could potentially allow online advertisers to design more effective viral marketing campaigns.

In this PhD research, we aim at answering the following research question: *“How can we accurately mine useful information from social media data?”* The main challenge towards solving

this problem is that social media data is *complex* and *diverse*. The data is complex because objects are often composed of distinct parts, having variable size and are represented by a large number of attributes. For example, an image uploaded to a social media service such as Flickr often has metadata such as title, description and comments. Additionally, when mining information from images, a very large number of numerical features are used to represent visual content, increasing even more the complexity of the data. The data is also diverse because it can be represented by images, time-series, graphs, videos and textual information.

1.2 Problem Statement

As we discussed in the previous section, the main goal of this thesis is to analyze and propose methods to extract useful information from social media services. Due to the complexity and diversity of social media data, in this PhD research, we propose the following thesis:

Thesis. *Data from social media services obey patterns that can be explored to design specialized methods for mining the raw data and improve the accuracy of prediction, anomaly detection and forecasting tasks when compared to traditional non-specialized data mining methods.*

In order to support the stated thesis, this PhD research focused on analyzing social media data in order to find useful patterns. The discovered patterns were then used to design data mining and modeling techniques tailored to the domain of social media services. More specifically, we worked on the following research problems:

- **Analyzing the timing of social media users' actions.** Users interact with social media services in multiple ways. For example, they can send messages, post pictures or watch videos. The timing of all these interactions is often stored by social media services, resulting in large databases of time-stamps. When we aggregate this temporal data by user, we get sequences of time-stamps showing the temporal behavior of individuals. Based on this data, we formulated the following research questions: (i) What patterns can be discovered from the temporal activities of social media users? (ii) Can these patterns be described by a mathematical model? (iii) Is it possible to use this data to detect anomalous behavior?
- **Modeling how the volume of users' interactions evolves over time.** When a piece of content such as a picture or a video is submitted to a social media service, users can interact with it in a number of ways. For instance, Facebook allows users to click the like button to indicate that they approve or enjoy a picture and to post textual comments to discuss it. The goal of this research problem is to analyze how the volume of these different types of user interactions evolve over time after a piece of content is submitted to a social media service. For example, if a picture receives 2,000 "likes", how many comments is it expected to receive?
- **Automatically annotating social media images.** Many social media allow users to submit and interact with multimedia content. Due to the volume of the data, allowing users to effectively find relevant content is an important aspect that can improve the usability of social media services. Allowing users to search for images using textual queries such as "find pictures of my family during our trip to the beach in 2014" is a way to address this

challenge. Since many images uploaded to social media services do not have reliable textual metadata, automatic image annotation can be used to enable textual queries. The goal of this research problem is to develop an automatic image annotation method capable of combining both visual and contextual data when annotating images.

1.3 Contributions

This PhD research resulted in three main contributions, each of which addressing one of the previously listed research problems. We summarize the contributions as follows:

1. **The Act-M Model:** In order to model the timing of users' interactions with social media services, we analyzed time-stamp data from Reddit and Twitter users. We show that the distribution of inter-arrival times between interactions is characterized by four patterns: (i) heavy-tails, (ii) periodic-spikes, (iii) correlation between consecutive values and (iv) bimodality. Based on these observations, we proposed the Act-M model, a model that accurately fits the distribution of inter-arrival times from social media users. Finally, we used Act-M to develop a method that detects if users are bots based only on the timing of their postings.
2. **The MFS-Map Method:** MFS-Map is a method that automatically annotates images from social media services. In addition to the visual content, these images have textual metadata. MFS-Map combines the images' content with their textual metadata to improve annotation accuracy when compared to methods that rely only on visual or textual features. MFS-Map is also efficient, using an algorithm that mines classification rules that represent the relationship between features (both visual and textual) with annotations. In our experiments, we show that MFS-Map can be used to search images from social media services.
3. **The VnC Model:** VnC is a mathematical model that describes how the volume of different user interactions evolve over time when a multimedia content is submitted to a social media service. We analyzed data from social voting services, a type of social media service in which users use up-votes to indicate that they approve content and down-votes to indicate disapproval in addition to textual comments. Given a submission, VnC describes: how the number of votes evolves over time; how the difference between up-votes and down-votes evolves over time and how the number of comments grows with respect to the number of votes. We validated VnC using more than 20,000 submissions from three social voting services: Reddit, Imgur and Digg. Compared to existing models for information diffusion, VnC provided a more accurate fit and forecast of the volume of user interactions. We also show that VnC can be used to forecast the number of votes and comments received by a submission. Finally, we used VnC parameters to detect submissions with unusual activity and to cluster different types of content (e.g. pictures and news articles).

1.4 Summary

This chapter summarized the motivations, objectives and main contributions of this PhD work. The remainder of this PhD thesis is organized as follows.

Chapter 2 provides an analysis of the relevant literature and background. In Chapter 3 we analyze time-stamp data describing the timing of user interactions in social media services. Based on our analysis, we propose the Act-M which can be used to detect users' anomalous behavior. In Chapter 4 we propose the VnC, a mathematical model that describes the coevolution of user interactions in social voting services. Chapter 5 presents the MFS-Map, a method for automatic annotation of images. Finally, in Chapter 6, we present the conclusion and future work.

Chapter 2

Background and Related Work

In this chapter we discuss the background and related work that are relevant to this PhD thesis. We start in Section 2.1 by discussing social media services. Section 2.2.1 summarizes existing efforts to extract knowledge from the timing of users' interactions in social media. In Section 2.2.2, we provide a review on the problem of analyzing how the popularity of content submitted to social media evolves over time. We discuss the problem of anomaly detection in the context of social media in Section 2.3.1. Finally, in Section 2.4, we present an overview of the task of automatically providing textual annotations to images based on their visual content.

2.1 Social Media Services

Social Media are Internet-based applications that allow the creation and exchange of user generated content [54]. User generated content is diverse, ranging from textual comments, pictures and any other media content created by social media users.

Social media plays an important and growing role in our society. A survey conducted by Pew Research Center [41] between March and April 2016 observed that more than 68% of U.S. adults use at least one social media service. This popularity has changed the way individuals communicate [111], how companies market their products [1], and even how professionals find their jobs [9, 105]. As consequence of this popularity, today we have access to an unprecedented volume of human behavior data generated by the operation of social media services. As a result, understanding how to extract useful information from all of this data has attracted the attention of the research community [84, 104].

Human behavior data obtained from social media can be represented by using different data types such as time-series, graphs and textual information. In the remainder of this section, we provide an overview of the modalities of social media data that were analyzed in this dissertation.

2.1.1 Social Voting Time-Series

We define *social voting services* as a type of social media service that allows users to use votes to curate content. Reddit¹ is an example of a social voting service. Figure 2.1 shows a screenshot of the Reddit front page: it consists of a list of submissions (i.e. links). Left of each link, Reddit

¹<https://www.reddit.com/>

Figure 2.1: Screenshot of the front-page of the social voting service Reddit. Social voting services allow users to curate content by casting up-votes and down-votes as well as posting textual comments. The difference between up-votes and down-votes combined with other factors determine the ranking of a submission.

Table 2.1: User interactions in social voting services.

Service	Up-votes	Down-votes	Comments
Reddit	✓	✓	✓
Imgur	✓	✓	✓
Digg	✓		
Hacker News	✓		✓
Slashdot	✓	✓	✓

shows an up and down arrow. Users can employ these arrows to cast up-votes and down-votes to express whether they like or not links submitted by other users. The difference between up-votes and down-votes and other factors, such as time since submission, are then used by a ranking algorithm to determine the popularity of a submission. Popular links are displayed on Reddit’s front page where they attract the attention of more users. As a result, the few links that reach the front page attract a significant part of user activity [66]. In addition to voting, social voting services such as Reddit often allow users to post textual comments to discuss submitted content.

Examples of social voting services that have attracted the attention of the scientific community include not only Reddit [61, 38, 103, 101], but also Slashdot [53, 60] and Digg [66, 91, 101]. Table 2.1 summarizes the types of user interactions available to rate and discuss links in different social voting services. Up-votes, down-votes and comments are present in many social voting services. While services such as Reddit, Imgur or Slashdot allow users to up-vote or down-vote a link, other services such as Digg or Hacker News only present the up-vote option.

Many authors have studied social voting services to predict the popularity of a submission

Figure 2.2: The activity of social media users' generates a sequence of time-stamps. (a) Each black bar corresponds to the time a Twitter user posted a tweet. (b) Histogram of activity level over a weekly interval built using the timing of the tweets. (c) Distribution of the inter-arrival times (IAT) between tweets.

based on its early popularity [91, 112] or on comment data [64]. Kunegis et al. [60] focused on mining the social network of users in Slashdot. In [61], the authors analyze how the title, submission time and content affect a link popularity.

A way to analyze social voting services data is to use a time-series for each type of user interaction. Given a submission such as a picture or a link to a news article, we can create a time-series for the number of up-votes, down-votes and comments received over time. In [66], the authors propose a method for estimating the popularity of submissions to the social voting service Digg² by using a self-exciting point process to model the up-vote time-series.

An aspect of social voting services not often explored in the literature is the correlation between the volume of user interactions received by user generated content. For example, if an image receives 10,000 up-votes, how many down-votes or comments is that image expected to receive? In Chapter 4 we address this limitation by analyzing how the number up-votes, down-votes and comments evolve as the popularity of a submission changes over time.

2.1.2 Temporal Sequences

When users interact with social media services, the timing of their actions are stored as time-stamps. For instance, each textual comment submitted to a social media service has an associated time-stamp. As a result, the temporal activity of a user in a social media service can be represented by a sequence of time-stamps. This is illustrated in Figure 2.2(a) where each bar corresponds to the time-stamp of a tweet created by a Twitter user.

A widely used approach to analyze sequences of time-stamps is creating histograms of activity levels over daily or weekly cycles. For example, Figure 2.2(b) shows the histogram of activity

²<http://digg.com>

levels over a weekly interval using the same data shown in Figure 2.2(a). Several authors have used histograms of activity levels to analyze the activity of social media users. Golder et al., for example, concluded in [40] that the daily activity of Facebook users could be clustered into two distinct patterns: weekdays and weekend days. The type of content also seems to affect the volume of activity over the week. For instance, it was observed in [68] that users of social media services focused on photo-sharing, such as Instagram, are less active during work hours when compared to users of text-based services such as Twitter. In addition to analyzing activity over cyclic intervals, in [79] the authors used the time-stamps of users' activity to create time-series of activity level across different social media services over several weeks.

Although histograms of activity levels are a useful analysis tool, part of the information contained in the raw time-stamp sequence data is lost in the summarization process. An example of lost information is the interval between activities. These intervals, also known as inter-arrival times (IAT), have been widely studied in the literature [71, 98]. Figure 2.2(c) shows the distribution of inter-arrival times using the same data from Figures 2.2(a) and 2.2(b). The analysis of inter-arrival times in social media has several applications. As we discuss in Sections 2.2.1 and 2.3.1, it can be used to study and analyze human behavior and to detect anomalous behavior.

2.1.3 Image Data

Many social media services such as Flickr, Twitter and Facebook allow users to share multimedia such as images. These images are often associated to context data such as textual information and geographic coordinates. For this reason, social media services constitute a valuable source from which we can mine information about image data.

While machine learning provides a set of useful tools to analyze image data, most traditional methods expect the input data to be in a tabular format. For most classifiers, for example, the rows of the dataset correspond to observations while the columns correspond to attributes [80]. As a result, when analyzing image data using machine learning methods, we often have to use *extraction functions* that take as input the matrix of pixels and outputs a *feature vector*.

A feature vector, denoted by $\mathbf{v} = (v_1, v_2, \dots, v_n)$, is defined as a point in a $\mathbb{R}^n$ space, where n is the dimension of the feature vector [93]. We define an image, denoted by I , as a pair $(P_I, \vec{I})$ where $P_I \subset \mathbb{N}^2$ is a finite set of pixel coordinates and $\vec{I}$ is a function that maps each pixel coordinate in P_I to the pixel value. Based on these concepts, we can define a feature extractor (i.e. the extraction function) as follows:

Definition 2.1 (Feature Extraction Function). *A feature extraction function, or simply extractor, is a function $\varepsilon : I \rightarrow \mathbb{R}^n$ which takes as input an image I and returns a feature vector $\mathbf{v} = (v_1, v_2, \dots, v_n)$.*

Feature extraction functions often capture particular visual aspects from images. For example, RGB and HSV histograms are feature extraction functions that capture global color content of the images. In contrast, Gist features [77] are employed to capture global spatial structure content. Local features can be represented, for example, by bag-of-visual-words feature vectors using SIFT [70].

2.2 Modeling User Behavior

The design of mathematical models capable of describing the most important properties of human behavior is a widely studied topic [57, 10, 55, 7]. There are models capable of describing different phenomena affected by human behavior, such as movie watching behavior [94], product ratings in crowdsourced review services [42], and human mobility [107].

In the context of data mining, mathematical models can be used to summarize and interpret data [87]. For instance, consider a very simple example for analyzing a dataset storing the height of every Brazilian person. That would correspond to over 200 million measurements. If we model the height of the Brazilian population using a normal distribution, then we can summarize the data by using two parameters: the mean and the standard deviation of the height. This simple model could be considered interpretable since, using only two parameters, it allows us to draw insights such as the probability of an individual being taller than, for example, two meters.

Despite the extensive work on modeling human behavior, the growing volume and variety of data, especially in social media, allows the scientific community to discover new patterns and improve existing models as well as modeling novel aspects of human behavior [34, 104, 18].

In this section, we discuss existing works on modeling user behavior in social media. We focus, in Section 2.2.1, on how to model the timing of users' actions in social media. Finally, in Section 2.2.2, we discuss existing work on modeling the popularity of content shared in social media services.

2.2.1 Communication Dynamics in Social Media

Understanding the dynamics of human activity has attracted the attention of the research community [57, 79, 55] as it has implications that range from efficient resource management [50], human event recognition [59] and clustering [71, 32] to anomaly detection [63, 95, 42].

A classical model for human dynamics assumes that arrival times follow a homogenous Poisson Process [47, 23, 88]. In a homogenous Poisson Process, new postings are submitted at a constant rate λ . In this case, the inter-arrival times (IAT) are distributed following an exponential distribution with density function $f(x) = \lambda \cdot e^{-x\lambda}$, and the expected number of postings per unit of time is given by λ . Previous works, however, have shown that human communication often has long periods of inactivity followed by bursts of activity [57, 32, 78]. This often yields heavy tailed inter-arrival times distributions that a Poisson Process is not able to explain.

In [10], Barabási proposes a model for human activity that assumes that individuals decide when to perform a task based on a priority-queue. This model generates an inter-arrival times distribution that follows a power law $f(x) = k \cdot x^{-\alpha}$ where k is the scale parameter and α is the shape parameter. Other similar approaches that have generated power-law distributions include [46]. While these models are able to explain the tail part of the inter-arrival times distribution, they fail to match the whole distribution.

In [71], Malmgren et al. show that the daily cycle of user activity can affect the distribution of IAT. They propose a Cascading Non-homogeneous Poisson Process (CNPP) model that takes into account users' activity levels. The CNPP model is a non-homogeneous Poisson Process, where the rate $\lambda(t)$ at which the postings are submitted changes over time according to two mechanisms:

Table 2.2: Patterns of communication dynamics described by the models presented in this Section.

Pattern	Poisson-Process	Log-Logistic	Power-Law	CNPP	SFP
Heavy Tails		✓	✓		✓
Bimodality				✓	
IAT Correlation					✓

(i) the model state, which can be either passive or active, and (ii) time of day and week. A disadvantage of the CNPP model is that it fails to match the heavy tailed IAT distributions which are common in social media communication [99, 98].

The Poisson-Process, as well as the power-law and CNPP models, assume that consecutive IAT are independent and identically distributed (i.i.d.). Recent works [55, 99, 98], however, have shown that consecutive IAT are often correlated. In [99], Vaz de Melo et al. showed that human communication violates the i.i.d. assumption and proposed a model named Self-Feeding Process (SFP) in which a synthetic IAT Δ_i is generated as a function of the previous IAT Δ_{i-1} :

$$\Delta_i \sim \text{Exp} \left[\lambda = \left(\Delta_{i-1} + \frac{1}{\lambda_o \cdot e} \right)^{-1} \right]^a \quad (2.1)$$

where $X \sim \text{Exp}[\lambda]$ denotes an exponential random variable with rate λ , λ_o is the starting rate of the Self-Feeding Process, and a is a shape parameter. Even though SFP is able to match the correlation between consecutive IAT, we show in Chapter 3 that this correlation is too strong when compared to that of real data from social media services.

Although there is no closed equation for the IAT probability distribution function of SFP, the authors showed that it can be approximated by a log-logistic distribution. The log-logistic distribution has been used to model income distribution [37], phone call duration [97] and job arrival at a data center [52]. Even though the log-logistic distribution is not able to generate correlated consecutive IAT as the SFP, it matches the heavy tail distribution and provides a closed equation for its PDF.

Table 2.2 summarizes the capabilities of the models for communication dynamics presented in this section. None of the discussed models is able to capture all the statistical properties – heavy tails, bimodality and consecutive IAT correlation – that are often observed in the timing of users’ actions in social media services.

2.2.2 Content Popularity in Social Media

In the previous section we discussed how mathematical models can be used to analyze the timing of users actions in social media services. Also relevant to this PhD research are models that describe how the popularity of an object submitted to a social media service – such as a picture or a news article – evolves over time. Many authors have studied how information or memes spread over a network of users such as bloggers or friends [67, 29, 62, 113]. These works are relevant to us because they could be used to explain how the popularity of an object submitted to a social media service evolves over time.

A well-known approach to model information diffusion is the Susceptible-Infected (SI) model. In the SI model, information diffusion is modeled as the spread of a disease in a network of N people. SI model assumes that a node can be in one of two states: susceptible S or infected I . An infected node can infect its neighbors with a probability β . Once infected, a node remains in that state.

Consider, for instance, a social voting service where users can vote on submitted content. If we assume that infected nodes correspond to users that have voted on a submission, then we can use the SI model to describe the number of votes $v_+(t)$ received by a link at time-tick t :

$$v_+(t+1) = \beta(N - V_+(t))V_+(t) \quad (2.2)$$

where $V_+(t)$ is the number of votes accumulated by the link at time-tick t , and β is the infectivity parameter. This example illustrates how a model such as SI can model content popularity in social media.

A model that is closely related to SI is the Bass Model [12]. It was originally proposed to describe the adoption of a new product in a population, but has shown to be useful in several other areas. Consider again the problem of modeling the number of votes received by a submission in a social voting service. Given a population of N potential voters, we can apply the Bass Model to describe the number of votes $v_+(n)$ received by a link over time:

$$v_+(t+1) = (N - V_+(t))\frac{V_+(t)}{N}q + (N - V_+(t))p \quad (2.3)$$

where q and p are, respectively, the coefficients of imitation and innovation. The Bass Model assumes that there are two types of consumers (in our example, voters): the imitators who are influenced by the word-of-mouth effect, and innovators who will buy a product (vote) with a probability given by the coefficient of innovation p .

In [36], Figueiredo et al. introduced Phoenix-R, a model that uses a peak finding algorithm in order to make multiple fits using the SIR model. Phoenix-R, as well as the SI and Bass models, yields a curve for $v_+(t)$ that decays exponentially after it reaches a peak. However, our analysis of real data from social voting Web sites indicates that the vote time-series have a heavy-tailed decay (Chapter 4). This agrees with previous results which indicate that the popularity of multi-media items such as blog posts, tweets [67, 74], Youtube videos [29] as well as user activity [78] decays as a power law. Moreover, with the exception of rare outliers, that we also discuss in Chapter 4, we find that the votes time-series can be modeled as a single peak.

In [74], Matsubara et al. proposed a model called Spike-M to describe how the popularity of news events evolves over time in a network of bloggers. Unlike Bass Model, Spike-M is able to explain the heavy-tail decay observed in real data. However, evaluating Spike-M for a time-series with n data points requires $O(n^2)$ operations.

Additionally, Spike-M assumes that the infectivity of a node decays over time. While it is reasonable to expect the popularity of a blog post to decay over time, in the context of social voting this assumption reduces the ability of a vote to attract new votes over time. However, ranking algorithms in social voting services usually do not take into account the time a vote was casted.

Figure 2.3: Comparison of models for content popularity. For each model we show the generated time-series using both linear vs. linear and log vs. log axes. While the SI, Bass and Phoenix-R models have an exponential decay, the Phoenix-R model has a heavy-tailed decay.

To conclude this section, Figure 2.3 compares the time-series generated by each model for content popularity that we discussed. We started by generating a time-series with the SI model by setting the infectivity parameter $\beta = 3.1 \cdot 10^{-5}$ and the population $N = 10^4$. For the remaining models, we selected the parameters that minimized the root-mean-square error (RMSE) with the SI time-series. For each model, Figure 2.3 shows the time-series in both linear vs. linear and log vs. log axes.

When compared using linear vs. linear axes, the time-series of all models are considerably similar, implying that the Bass, Phoenix-R and Spike-M models are able to mimic the rise-and-fall pattern of the SI model. However, when we analyze the time-series in log vs. log axis we notice significant differences. The most important is the concavity of the tail part of the time-series: the Spike-M produces a heavy-tailed decay that can be roughly approximated by a straight line. The tail part of the Phoenix-R time-series, on the other hand, has a very steep fall – which can be explained by an exponential decay – followed by a straight line caused by a constant term parameter. We set, for comparison purposes, the vertical axis for the log-log plots to the range 0.01 to 1,000. Since the SI and Bass models have an exponential decay and do not have a constant term – see Equations 2.2 and 2.3 – the tail part of the time-series fall below the visible part of the plot³.

³If we had set the vertical axis in Figure 2.3 to show the minimum value of the log-log time-series for the Bass and SI models, the rise part of time-series would be compressed and barely visible

2.3 Knowledge Discovery in Social Media Data

As we discussed in the beginning of this chapter, the popularity of social media services in combination with advances in data acquisition (e.g. mobile devices) and storage technologies, resulted in an explosion in the availability of human behavior data. Instead of left unused in large databases, this growing volume of data can be used in a number of applications such as detecting fraud [2, 51], and allowing users to effectively search multimedia data [16]. Considering this demand, the research area of Knowledge Discovery in Databases (KDD) has as goal the development of techniques to assist humans to analyze, extract and understand knowledge from data [80]. In [33], Fayyad et al. defined KDD as: “The nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data.”

KDD tasks can be categorized as predictive or descriptive [80]. The goal of predictive tasks are to predict the value of a given target attribute based on the value of the other attributes. Descriptive tasks, on the other hand, seek to extract patterns such as correlations and trends that can be used to describe and understand the data being analyzed. Some of the key data mining tasks are:

- **Classification:** This is a prediction task used when the target attribute are discrete. Classification could be used, for example, to predict the diagnosis of a patient (e.g. sick vs. healthy).
- **Regression:** Like classification, regression is a predictive task. The main difference is that the target attribute is continuous. Predicting the price of a property given attributes such as the floor area and number of bedrooms is an example of a scenario in which regression could be used.
- **Cluster Analysis:** The purpose of this task is to find groups of objects such that objects in the same group are more similar to each other than objects in different groups [39]. Cluster analysis could be used, for example, to find social media users with similar behavior.

Two KDD tasks that we explore in more detail in this Dissertation are anomaly detection and association rule mining. In Chapter 3, we propose a method for detecting social media users with anomalous temporal behavior. For this reason, Section 2.3.1 gives an overview of the task of anomaly detection in the context of social media. Additionally, we employed association rule mining to design a method that automatically suggests textual annotations to images based on their visual and contextual content (Chapter 5). Therefore, in Section 2.3.2 we discuss the problem of mining association rules.

2.3.1 Anomaly Detection in Social Media

Despite its impacts, the popularity of social media has also introduced negative effects into our society. Undesired activity such as spam and spread of false news can be used by attackers to obtain illegal monetary gain, to deceive users or to damage the reputation of an individual, product or organization [82, 92]. As a result, anomaly detection methods capable of identifying such undesired behavior are highly important.

Anomaly detection, also known as outlier detection, exception mining or novelty recognition, is the task of finding observations that are significantly different from the remaining data [2].

Figure 2.4: Box-plot of the number of posts made per hour by Twitter, Google+, Reddit and Hacker-News users. Points outside the whiskers (the vertical lines extending from the boxes) are considered outliers.

Anomalies may arise for different reasons including, for example, measurement errors [89]. However, in the context of this Dissertation we are interested in anomalies caused by the unusual behavior of the data generating process. This type of anomaly often provides useful information about abnormal characteristics of the observed system and can be used to solve problems such as detecting fraud [48], social media bots [106] and computer networks intrusion [115]. The study of anomaly detection is also relevant to our work, since we use our proposed Act-M model in Chapter 3 to detect social media bots.

There are several taxonomies for anomaly detection methods. In [21], the authors consider two categories: supervised vs. unsupervised anomaly detection. Examples of unsupervised methods may range from statistical tests for extreme values to density-based approaches. For example, Tukey’s box plot [96] flags as outliers the measures outside the range $(Q1 - 1.5 \cdot IQR, Q3 + 1.5 \cdot IQR)$, where $Q1$, $Q3$, and IQR denote, respectively, the first quartile, third quartile and interquartile range of the data. Figure 2.4 shows several box-plots comparing the number of posts made per hour by 223, 23, 222, and 60 Twitter, Google+, Reddit and Hacker-News users, respectively. The boxes are delimited by the quartiles of each dataset while the whiskers (the lines extending vertically from the boxes) are delimited by the lowest and highest value within the range $(Q1 - 1.5 \cdot IQR, Q3 + 1.5 \cdot IQR)$. Measures outside the whiskers, depicted as diamonds, are considered outliers.

When labeled data regarding anomalies is available, one-class classification methods can be employed [90]. Examples of other approaches for detecting anomalies include nearest neighbors [81], density based [19] and graph based methods [30, 5, 72].

In the context of using time-stamp data to detect bots (as we do in Chapter 3), Zhang and Paxson proposed a method [108] that consists of constructing a scatter-plot of the minute vs. the second for all comment time-stamps of a user. The plot is then used to visually assess whether a user is a bot or not. However, the plot is not used to automatically test whether users are bots or not.

In [24], the authors use the entropy of the histogram of tweets’ inter-arrival times as a feature to detect Twitter bots. These temporal features are then combined to textual and account properties.

Table 2.3: Example of a dataset in which the data is organized as transactions. Each row consists of a transaction identifier (TID) and an itemset of products.

TID	Transaction
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Soda}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Soda}

In Chapter 3 we show that it is possible to detect bots on social media services such as Twitter even when textual features or account properties are missing. More specifically, we propose a method that can detect bots using only the timing of users' actions.

2.3.2 Association Rule Mining

The task of mining association rules was first proposed in [3] to automatically discover which products are frequently bought together. Consider the example of Table 2.3. It illustrates – in a simplified way – data typically stored by supermarket chains and online stores: sets of products bought together in a single purchase. In Table 2.3, each row corresponds to a *transaction*, consisting of a transaction identifier (TID) and an *itemset* (i.e. a set of items) bought by a client. In this example, three out of five purchases contain the products diaper and beer. Association rule mining could be used to detect this and other similar patterns allowing shopkeepers to find out which products are commonly purchased together.

When mining association rules, an important concept is that of itemset, defined as follows:

Definition 2.2 (Itemset). *Let $\mathcal{I}$ denote the set of all items that may appear in a database. An itemset is a set of items $\{j_1, j_2, \dots\}$ such that $j_i \in \mathcal{I}$.*

Considering the example of Table 2.3, the set $\mathcal{I}$ of all items appearing in the database is {Bread, Milk, Diapers, Beer, Eggs, Soda}. Another concept is the *support* of an itemset. Let X denote an itemset and $\mathbf{E} = \{E_1, E_2, \dots\}$ the set of all transactions (i.e. rows) of a database. The support of X is the number of transactions $E_i \in \mathbf{E}$ that contain X . Formally, the support of an itemset is given by the following expression:

$$\text{sup}(X) = |\{E_i | X \subseteq E_i, E_i \in \mathbf{E}\}| \quad (2.4)$$

where $|\cdot|$ denotes the number of elements in a set. In the example of Table 2.3, the itemset {Bread, Milk} has a support of three, since three out of five transactions contain both items.

One way to represent patterns from itemset data consists in using *association rules*. Association rules are implication expressions presented as $X \Rightarrow Y$, where X and Y are two disjoint itemsets defined, respectively, as the antecedent and consequent of the rule. A rule such as {Milk, Diapers} $\Rightarrow$ {Beer} could be used to indicate that customers who buy milk and diapers are also likely to buy beer.

When mining association rules, there are several measures of significance to estimate the usefulness of a rule. Two of such measures are the support and confidence. Support measures the frequency of a rule in a database, while confidence is an indication of how frequently a rule is found to be true. Let $|\mathbf{E}|$ denote the total number of transactions in a database. The support $\text{sup}(X \Rightarrow Y)$ and confidence $\text{conf}(X \Rightarrow Y)$ of the rule $X \Rightarrow Y$ are given by:

$$\text{sup}(X \Rightarrow Y) = \frac{\text{sup}(X \cup Y)}{|\mathbf{E}|} \quad (2.5)$$

$$\text{conf}(X \Rightarrow Y) = \frac{\text{sup}(X \cup Y)}{\text{sup}(X)} \quad (2.6)$$

Using as example the data from Table 2.3 and the rule $\{\text{Diapers}\} \Rightarrow \{\text{Beer}\}$, the antecedent of the rule is $\{\text{Diapers}\}$ and the consequent is $\{\text{Beer}\}$. Since the number of transactions (rows) in Table 2.3 is five, the support and confidence of the rule are given by:

$$\text{sup}(\{\text{Diapers}\} \Rightarrow \{\text{Beer}\}) = \frac{\text{sup}(\{\text{Diapers}, \text{Beer}\})}{|\mathbf{E}|} = \frac{3}{5} \quad (2.7)$$

$$\text{conf}(\{\text{Diapers}\} \Rightarrow \{\text{Beer}\}) = \frac{\text{sup}(\{\text{Diapers}, \text{Beer}\})}{\text{sup}(\{\text{Diapers}\})} = \frac{3}{4} \quad (2.8)$$

The problem of mining association rules consists in finding in a database all rules whose support and confidence are higher than a minimum threshold. This procedure can be divided into two phases:

1. Find all frequent itemsets whose support are higher than a minimum support threshold.
2. Generate, from the set of frequent itemsets, rules whose confidence are higher than a minimum threshold.

In terms of computational cost, generating the frequent itemsets dominates the execution time of the association rule mining algorithms. Consider, for example, a database with d different items. In this scenario, the number of itemsets that can be generated (excluding the empty itemset) is $2^d - 1$. Since the number of itemsets grows exponentially on the number of items in the database, computing the support of all itemsets is unfeasible even for databases of moderate size.

To address this problem, most association rule mining algorithm employ strategies to avoid generating all possible itemsets in a database. For example, the Apriori algorithm [4] explores the fact that the support of an itemset never exceeds the support of its subsets. This allows the Apriori algorithm to reduce the number of generated itemsets when mining association rules. In addition to reducing the number of generated itemsets, the FP-Growth [43] method uses a prefix-tree to represent itemsets and reduce memory use of the mining process. An alternative approach, used by the SON algorithm [86], consists in dividing the database into chunks to give an approximate answer to the problem of mining frequent itemsets. From each of these chunks, the SON algorithm mines frequent itemsets that are merged into a single set in a final phase.

Although originally proposed with the goal of analyzing supermarket purchases, the task of mining of association rules is not limited to this application. In Chapter 5 we employ *classification association rules* (CARs) to design a method that automatically suggests textual annotations to images. CARs are a special type of association rules used to solve classification problems.

Specifically, the consequent of CARs represents class attributes while the antecedent is composed of items representing the predictor attributes [69].

Given a collection of CARs, there are two main approaches to determine the class of a given unlabeled object. The first approach, employed in [69, 11], consists in ordering all the generated rules based on their support and confidence. The predicted class is the one given by the rule with the highest precedence among all rules that apply to the test object. A second strategy, adopted in [44, 8, 83], assigns a score value to each class. The class score is computed based on measures such as the confidence of the rules that are applicable to the object being classified. Finally, the classifier returns the class with the highest score.

2.4 Automatic Image Annotation

Research on image retrieval traditionally considers two types of queries: by example and by keywords. When querying by example, the user provides a sample image and the retrieval system returns a set of similar images regarding a given criterion. It is often impractical for the user to provide the sample image to express the query. Additionally, as we discussed in Section 2.1.3, visual similarity is defined by color, texture and shapes and these low level features present a gap regarding the query semantics.

In keyword-based queries, images are retrieved by employing their textual annotations, as is done with text retrieval. However, for keyword-based queries to work images should contain textual annotations. Since manual annotation is a very tiresome task, it can be impractical for huge numbers of images. In addition, human annotations are subjective and can be ambiguous. An alternative to manual annotation is to employ automatic image annotation (AIA).

Automatic image annotation can be modeled as a multi-label classification problem [16, 76]. In multi-label classification, each object can be associated with a set of labels. In the context of image annotation, the objects are the images and the labels are the annotations.

Cross-training is a widely employed approach for multi-label classification in image annotation [16]. It consists in training a binary classifier for each annotation. The classifiers are then employed to predict which annotations are relevant to a non-annotated image. Each classifier is trained using images that contain their respective annotation as positive examples and the remaining images as negative examples. Annotations are scored based on the probability returned by the respective binary classifier.

Finding classifiers' optimal meta-parameters for cross-training annotation can be potentially time consuming. Most classifiers have a set of user-defined meta-parameters that directly affects accuracy. For instance, in order to train a Support Vector Machine (SVM), it is necessary to set the kernel function type, the kernel function parameters (e.g. the exponent of a radial basis function) and the misclassification cost. In order to obtain acceptable results, it is necessary to find the set of parameters that optimize the classifier performance. This is achieved by cross-validation using different combinations of parameter values.

Another disadvantage of cross-training is that images in the training phase must be divided into positive and negative examples. Since the number of examples in the positive class is usually significantly smaller than the number of negative examples, the classification problem is imbalanced, degrading the classification accuracy [45].

Another approach to image annotation consists in using nearest-neighbors methods to annotate images based on annotations of visually similar images [110, 109]. Visual similarity is estimated by calculating the distance between images' feature vectors. Weighted nearest-neighbors models have shown to provide state of the art results in image annotation [15, 100].

A simple method can be formulated by suggesting annotations to an image based on the annotations of its k nearest-neighbors in the training set. For example, a one nearest-neighbor (1-NN) model annotates images with the existing annotations of the most similar image from the training set.

Most of the current automatic image annotation methods [110, 15, 100, 109] employ multiple visual feature extraction algorithms to analyze images. This is necessary because none of existing extraction algorithms is capable of describing the large visual variability of images. In the following section, we discuss strategies to combine these different visual features with context features when annotating images.

2.4.1 Combining Visual and Context Features

Social media images often present metadata such as geographical coordinates, time-stamps and textual comments. These metadata are potentially useful when annotating social media images as they can be used as *context features* alongside visual content to improve annotations' quality. In this section we discuss existing approaches for combining visual features and context features.

There are two main strategies when combining visual and context features: early-fusion and late-fusion [25]. Given a set of feature vectors representing the content and context of an image, the early-fusion strategy consists in concatenating all these feature vectors into a large feature vector. The main advantage of this approach is its simplicity. For example, this large feature vector can be feed in a straightforward manner to a classifier [100]. However, this simplicity has a disadvantage: the resulting feature vector may have a very high dimensionality which has the potential to degrade the performance of the annotation methods due to the dimensionality curse [58].

The late-fusion strategy consists in first analyzing each feature vector individually. The results of these individual analyses are then fused together to decide which annotations are relevant to the input image. In [114], the authors adopted a late-fusion strategy in which they trained three SVM classifiers: one for a feature vector representing all visual features and two classifiers for two different representations of context data. The classifiers returns three scores that are then fed to a final SVM classifier.

2.5 Conclusions

The goal of this chapter was to present the background and related works that are relevant to this Dissertation. We started with Section 2.1 by providing an overview of several types of human behavior data that can be obtained from social media services. Next, in Section 2.2, we discussed how mathematical models can be employed to summarize and interpret several aspects of human behavior in social media. Section 2.3 examined how knowledge discovery in databases can be applied to analyze social media data. Finally, we concluded with Section 2.4, where we presented the background on the task of automatically annotating images.

The literature covered in this chapter is quite broad in the subjects presented. It was not the author's intent to exhaustively discuss all of these subjects. Rather, the objective was to present the reader the required background and knowledge to understand the contributions of this PhD.

Chapter 3

Dynamics of Social Media Communications

Social media services allow users to communicate by posting content such as photos, comments and videos. Postings are often annotated with time-stamps specifying when they were created. As a result, when users communicate, they create a sequence of time-stamps that can be used to study the properties of human communication in social media services. In this chapter, we use time-stamp sequences to analyze communication dynamics in social media.

3.1 Introduction

As discussed in Section 2.2.1, there are many works devoted to studying the timing of human actions. Results on the area of human communication dynamics include learning that inter-arrival times have a heavy-tailed distribution [10] and that they are not independent and identically distributed (iid) [98]. In this chapter, we use time-stamps sequences generated by the activity of social media users as a proxy to study human communication dynamics.

Our first goal is to analyze the distribution of postings' inter-arrival times to find patterns common to different users and social media services. By using data from Reddit, Twitter, Stack-Overflow and Hacker-News, we show that: (i) the inter-arrival time Δ_i between two postings of the same user depends on the previous inter-arrival time Δ_{i-1} ; (ii) the distribution has periodic spikes at every 24 hours; (iii) the distribution is bimodal and (iv) the distribution is heavy tailed.

Figure 3.1 illustrates the discovered temporal patterns using an example one of the studied social media services: Hacker-News. Later, in Section 3.5, we show that these patterns also apply to the other studied datasets.

The logarithmic-binned histogram of inter-arrival times of Figure 3.1(a) has spikes at every 24 hours and two modes: the first occurring near 100s and the second occurring near 10,000s. Figure 3.1(b) depicts a heat-map of consecutive inter-arrival times. The concentration of points along the diagonal of the heat-map is caused by the correlation between consecutive inter-arrival times. Finally, Figure 3.1(d) shows, using the complementary cumulative distribution function (CCDF), that the inter-arrival time distribution is heavy-tailed.

Figure 3.1: The distribution of inter-arrival times (IAT) in seconds, of postings in social media services is characterized by several patterns. (a) the logarithmic-binned histogram showing periodic spikes and bimodal inter-arrival time distribution. (b) The diagonal of the heat-map shows the positive correlation between consecutive inter-arrival times. (d) Heavy-tailed inter arrival time distribution. Our proposed Act-M model, indicated by a solid blue line in (a) and (d), matches the real data. Additionally, the heat-map (c) generated by Act-M matches the correlation between consecutive inter-arrival times.

Our second goal is to answer the following research question: is it possible to design a model capable of explaining all the four discovered temporal patterns? Since existing literature on human communication dynamics fail to explain all these patterns with a single mathematical model, we introduce Act-M (Activity Model). Act-M is a model that generates synthetic time-stamps whose distribution of inter-arrival times matches all the four temporal patterns that we observed in the analyzed data. In Figures 3.1(c,g) and 3.1(d,h), Act-M is indicated by a solid blue line which accurately matches the real data. The heat-maps of Figures 3.1(b,f) were created using synthetic time-stamps generated by Act-M.

Finally, the third contribution discussed in this chapter is showing that Act-M can be used to decide whether users are controlled by humans or computer programs based only on the timing of their postings.

We validate Act-M using data from over 55 million postings from more than 35,000 Reddit, Twitter, Stack-Overflow and Hacker-News users. The experiments show that Act-M consistently provides a more accurate fit to real data than existing models for human dynamics. Additionally, Act-M is able to detect bots in highly imbalanced datasets with a precision and sensitivity of 77% and 70% respectively.

The remaining of this chapter is organized as follows. We start in Section 3.2 by introducing the problem definition. In Section 3.3 we report the discovered patterns of user communications.

The Act-M model is described in Section 3.4. In Sections 3.5 and 3.6 we evaluate Act-M. Finally, we conclude this chapter in Section 3.7.

3.2 Problem Definition

The data we analyze in this chapter consists of postings time-stamps from a set of users $\{\mathcal{U}_1, \mathcal{U}_2, \dots\}$. Each user $\mathcal{U}_i$ produces a sequence of postings time-stamps $\mathbf{T}_i = (t_1, t_2, \dots)$ where $t_i \geq t_{i-1}$. A posting represents an event in which a user submits a comment, tweet, question, answer or a vote to a social media service. Each posting has a time-stamp t which indicates the time at which it was submitted. From each sequence of time-stamps $\mathbf{T}_i = (t_1, t_2, \dots)$ we are able to compute the postings inter-arrival times (IAT) as follows:

Definition 3.1 (Inter-Arrival Time (IAT)). *An IAT, denoted by Δ_i , corresponds to the time interval between two consecutive postings t_i and t_{i-1} from the same user.*

A sequence $\mathbf{T}_i$ of time-stamps from a given user $\mathcal{U}_i$ yields a corresponding sequence of IAT $\Delta_i = (\Delta_1, \Delta_2, \Delta_3, \dots)$. We also analyze *consecutive inter-arrival times*, which are pairs of two IATs (Δ_i, Δ_{i+1}) from the same user.

Given the time-stamp data from social media services, our goal is to solve the following problems:

Problem 3.1 (Pattern-Finding). *Given the time-stamps data from different social media services, analyze the inter-arrival times distribution and find patterns that are common to all services.*

Problem 3.2 (Time-Stamp Generation). *Design a model that is able to generate synthetic time-stamps whose inter-arrival times fits the real data distribution and matches all the patterns found in Problem 3.1.*

Problem 3.3 (Bot-Detection). *Given time-stamp data from a set of users $\{\mathcal{U}_1, \mathcal{U}_2, \mathcal{U}_3, \dots\}$ where each user $\mathcal{U}_i$ has a sequence of postings time-stamps $\mathbf{T}_i = (t_1, t_2, t_3, \dots)$, decide if user $\mathcal{U}_i$ is a human or a bot.*

With respect to Problem 3.3, we consider users to be bots if their postings are created in an automated manner by using, for example, a computer program.

3.3 Observations on the Dynamics of Social Media Communications

In this section we analyze time-stamp data generated by user activity in four different social media services. We start in Section 3.3.1 by describing the datasets that we studied and report, in Section 3.3.2, our findings.

3.3.1 Datasets

In order to analyze temporal patterns of user activity we collected data from four social media services: Twitter, Reddit, Stack-Overflow and Hacker-News. Table 3.1 summarizes the collected data. For all datasets, the time-stamps have a resolution of one second.

Twitter Dataset: Twitter is a social media service that enables users to share short messages (140 characters) and pictures. For this work, we initially collected data from over 9,000 verified accounts. Verified accounts are manually checked to be authentic by Twitter. For each user we collected the 3,000 most recent postings. Users with less than 800 postings were discarded, resulting in 6,790 users. We also used the Twitter lists features to search for accounts that were suspicious of being a bot. Twitter lists are collections of Twitter accounts that are curated by users. Using the Twitter API, we searched for suspicious users that were in lists that contained the term “bot” in its description or title. We manually inspected the suspicious accounts by reading their recent tweets and selected 64 bots.

Reddit Dataset: Reddit is a Web site that allows users to submit and rate content by voting and commenting on them. We randomly selected a set of over 200,000 users that commented on Reddit at least once between December 6th and December 29th 2013. Due to Reddit API restrictions, we collected the 1,000 most recent comments for each user. Similarly to the Twitter data, users with less than 800 comments were discarded, resulting in a set of 21,198 users. Additionally, we searched the Web using queries such as “reddit bots users” to collect a list of accounts that were suspicious of being bots. We inspected each suspicious account by reading the content of the most recent comments and manually selected 32 bots.

Stack-Overflow: Stack-Overflow is a questions and answers Web site focused on computer programming. Users can post questions, vote and comment on them. Using publicly available data from Stack Overflow, we collected the time-stamps of all postings and comments made by all users on the Web site. We discarded users with less than 800 time-stamps.

Hacker News: Hacker News is a social voting Web site focused on technology news. The dataset is publicly available¹ and corresponds to all comments posted to Hacker News from december 2009 to May 2014. Users with less than 800 comments were discarded.

Table 3.1: Summary of the datasets.

Dataset	# Users	# Bots	# Time-stamps
Reddit	21,198	32	20 Million
Twitter	6,790	64	16 Million
Stack-Overflow	8,755	0	19 Million
Hacker-News	1,368	0	2 Million

3.3.2 Temporal Patterns in Social Media Communication

In this section we analyze the postings’ inter-arrival times using data from the datasets described in Section 3.3.1. Our goal is to find temporal patterns common to all datasets. We start by answering the following question: is the distribution of inter-arrival times heavy-tailed? We define a distribution as heavy tailed if it is exponentially bounded:

Definition 3.2. A random variable X with probability density function $f(x)$ has a heavy tailed

¹<https://github.com/sytelus/HackerNewsData>

Figure 3.2: Heavy tailed IAT distribution. The black dots correspond to the empirical complementary cumulative distribution function (CCDF) of the IAT by all users in each dataset. The exponential distribution fit (red line) does not match the heavy tail of the empirical distribution.

distribution if:

$$\lim_{x \rightarrow \infty} \frac{e^{kx}}{f(x)} = 0, \text{ for } k < 0 \quad (3.1)$$

To answer the question, we plot in Figure 3.2 the empirical complementary cumulative distribution function (CCDF) of the IAT of postings from all users in each dataset (black dots). The solid red line corresponds to the exponential distribution fit obtained using maximum likelihood estimation (MLE). From visual inspection of Figure 3.2, it is possible to conclude that the exponential distribution does not provide an accurate fit to the empirical IAT CCDF. Specifically, the tail part of the empirical distribution is always above the fitted curve in all datasets.

A similar pattern arises when we analyze the distribution of individual users. In Figure 3.3, the black dots correspond to the empirical CCDF of the inter-arrival times of selected users while the solid red line corresponds to the exponential distribution fit. For all datasets, the CCDF of individual users also has a heavier tail than the exponential distribution fit. Although Figure 3.3 only shows the CCDF for the selected users on the first, second and third quartile positions with respect to the number of comments, the same pattern is observed for the majority of the users in all datasets analyzed. Based on the results from Figures 3.2 and 3.3, we make the following

Figure 3.3: The distribution of IAT from individual users is also heavy-tailed. The empirical CCDF (black dots) has a heavier tail than the exponential distribution (red line). The exponential distribution was fit to each user's data employing maximum likelihood estimation. All plots are in log-log scale.

observation:

Observation 3.1. *The distribution of the postings' IAT is heavy-tailed.*

This heavy-tail pattern agrees with the previous studies in human activity that we discussed in Section 2.2.1. This also shows that classical Poisson statistics are not adequate to model the interval between users' postings. A consequence of the heavy-tail pattern is that a regular user can be inactive for long periods of time.

Figure 3.4: Bimodal IAT distribution. The gray area corresponds to the logarithmic binned histogram of IAT and the solid black line to the kernel density estimation (KDE). The plots were created using data from all users in each dataset.

In Figure 3.4, we analyze how the inter-arrival times of postings are distributed. We plotted the logarithmically binned histogram of the inter-arrival times for all users in each dataset. The histogram was computed by varying the width of bins, such that i -th is wider than the $(i-1)$ -th bin by a fixed multiple. That is, the sequence of bin widths follows a geometric series. As a result, the bins in the tail of the distribution receive more data points than they would if the bin widths were constant. For all datasets, the log-binned histogram has two modes. The first mode is between one and six minutes while the second mode is between one and three hours.

For individual users, the number of postings time-stamps is not large enough to compute a histogram as we did for the aggregate data in Figure 3.4. In order to analyze the distribution for individual users, we show in Figure 3.5 the kernel density estimation (KDE) of the inter-arrival times. The bandwidth was automatically selected by minimizing the mean integrated squared error using the normal distribution approximation over the logarithm values of the inter-arrival times. From each dataset, we selected three users: on the first, second and third quartile positions with respect to the number of comments. For the majority of the users, the distribution of inter-arrival times has two clear modes. Based on the distribution of the inter-arrival times from aggregated as well as individual users data, we make the following observation:

Observation 3.2. *Excluding the daily periodicities, the distribution of the IAT of postings is bimodal.*

Observation 3.2 reinforces the hypothesis that classical Poisson statistics are not adequate to model the distribution of the time intervals between users' postings.

Figure 3.5: Bimodal inter-arrival times distribution for individual users. The black line depicts the kernel density estimation (KDE) of the inter-arrival times from selected users.

The circadian rhythm also affects the users' communication patterns. Figure 3.6 shows the histogram of the inter-arrival times of postings by zooming into the interval from five hours to ten days. The histograms for all studied datasets have periodic peaks at every 24-hour intervals. As we discuss in Section 3.4, when we introduce our proposed Act-M model, the periodic spikes can be attributed to the users' daily intervals of inactivity (i.e. sleeping). This finding is summarized as follows:

Observation 3.3. *The distribution of inter-arrival times has periodic spikes at every 24 hours.*

Finally, we analyze how consecutive inter-arrival times are correlated. For this purpose, we create a heat-map of the distribution of pairs of consecutive inter-arrival times (Δ_i, Δ_{i+1}) . In order to construct the heat-map, we divided the Δ_i vs. Δ_{i+1} space into a logarithmic spaced grid. Next, we counted the pairs of consecutive inter-arrival times in each grid cell. Finally, we used color-coding to represent the grid cell counts, resulting in a heat-map visualization. Figure 3.7 shows the heat-maps for the analyzed datasets.

Figure 3.6: The distribution of IAT has periodic peaks at 24-hour intervals. The gray area corresponds to the log-binned histogram of IAT from all users in each dataset.

There is a concentration of consecutive inter-arrival times along the diagonal of the heat-maps for all the datasets. This indicates that there is a positive correlation between consecutive inter-arrival times and that the distribution is not independent and identically distributed (i.i.d). Another pattern present in the heat-maps are crests for $\Delta_i \approx 1$ day or $\Delta_{i+1} \approx 1$ day. The crests match the periodic spikes at 24-hour intervals (Observation 3.3) from the PDF of postings' inter-arrival times (Figure 3.6). Based on the analysis of the heat-maps, we make the following observation:

Observation 3.4. *Consecutive inter-arrival times are positively correlated.*

Figure 3.7: Heat-maps depicting the joint distribution between consecutive inter-arrival times. The distribution of consecutive inter-arrival times has a concentration along the diagonal $\Delta_i = \Delta_{i+1}$ indicating that consecutive inter-arrival times are correlated.

3.4 The Act-M Model

In this Section we introduce the Act-M (Activity Model) model. We designed Act-M to generate synthetic time-stamps matching the dynamics of users' communication in social media services. Specifically, the distribution of inter-arrival times produced by Act-M matches the following patterns:

1. Heavy tails (Observation 3.1);
2. Bimodality (Observation 3.2);
3. Periodic spikes centered at 24 hour intervals (Observation 3.3);
4. Correlation between consecutive inter-arrival times (Observation 3.4);

Our description of the Act-M model starts in Section 3.4.1 by proposing the SCorr (Self-Correlated Process), a stochastic process that generates events whose consecutive inter-arrival times are correlated. In Section 3.4.2, we use SCorr as a building block to explain the complete Act-M model. Finally, in Section 3.4.3, we describe the Act-M's parameter estimation algorithm.

3.4.1 The Self-Correlated Process

The Self-Correlated Process (SCorr) is a stochastic process that generates a sequence of random variables $(X_1, X_2, \dots)$ in which two consecutive random variables X_i and X_{i+1} are correlated. This is different from a Poisson Process or a queue-based model in which the random variables are independent and identically distributed (i.i.d.). The motivation to generate non i.i.d. random

Figure 3.8: State diagram of the Act-M model. Act-M can be either in the rest or active state. The transition probabilities are given by $p_a(t)$ and p_r . The probability $p_a(t)$ changes over time while p_r is constant. The duration δ of each state is sampled from a SCorr process.

variables lies in our observation that consecutive time-tamps from social media postings are correlated. The SCorr Process is defined as follows:

Definition 3.3. Let δ_i be the inter-arrival time between the events i and $i - 1$. A stochastic process is a Self-Correlated Process, with base rate λ_o and correlation ρ if:

$$\delta_1 \sim \text{Exp}\left(\frac{1}{\lambda_o}\right) \quad (3.2)$$

$$\delta_i \sim \text{Exp}\left(\rho \cdot \delta_{i-1} + \frac{1}{\lambda_o}\right) \quad (3.3)$$

where $X \sim \text{Exp}(1/\lambda)$ denotes an exponentially distributed random variable with rate λ .

In SCorr, the duration δ_i between two events is sampled from an exponential distribution with rate λ depending on the previous inter-event time δ_{i-1} . SCorr uses the *correlation parameter* ρ to control the dependency between consecutive inter-event times. Therefore, the Poisson-Process is a special case of SCorr:

Theorem 3.1 (SCorr Equivalence). When $\rho \rightarrow 0$, SCorr reduces to a Poisson Process with rate λ_o . When $\rho = 1$, SCorr reduces to a Self-Feeding Process.

Proof. For $\rho \rightarrow 0$, Equation 3.3 yields $\delta_i \sim \text{Exp}(1/\lambda_o)$ which corresponds to the IAT distribution of a Poisson-Process. For $\rho = 1$, Equation 3.3 results in $\delta_i \sim \text{Exp}(\delta_{i-1} + 1/\lambda_o)$, which corresponds to the definition of the Self-Feeding Process. $\square$

3.4.2 Time-stamp Generation

The SCorr process generates a heavy-tailed distribution with correlated consecutive inter-arrival times. The SCorr alone, however, is not sufficient to generate a bimodal distribution. To solve this problem, the Act-M model uses a two state Markov chain whose transition probabilities change over time the users' circadian rhythm. As illustrated in the diagram of Figure 3.8, Act-M has two states: active and rest. If Act-M is in the active state, there is a probability p_r to transition to the rest state and a probability $1 - p_r$ to remain in the active state. Similarly, if Act-M is in the rest state, the transition probability to the active state is $p_a(t)$ and the probability of remaining in the rest state is $1 - p_a(t)$. As discussed later in this section, the transition probability $p_a(t)$ changes over time.

In order to generate the synthetic time-stamps, Act-M performs the following actions after each state transition:

1. **Active:** After Act-M transitions to the active state, it waits a time interval δ and generates a posting event. The time interval δ is generated by using the SCorr process with base rate λ_A and correlation parameter ρ_A .
2. **Rest:** After Act-M transitions to the rest state, it waits a time interval δ generated by using the SCorr process with base rate λ_R and correlation parameter ρ_R . As result, the rest state only contributes to increment the postings inter-arrival times.

As result of the Act-M model formulation, there can be more than one state transition between two posting events. For example, after a posting event is generated in the active state, Act-M could transfer and remain in the rest state for more than one transition. Act-M also assumes that the state duration of the rest state is larger, on average, than the state duration of the active state. This is achieved by ensuring that the base rate of the rest state λ_R is smaller than the base rate λ_A of the active state.

Users often modulate their activity following a daily cycle. For instance, users are less likely to become active during typical sleep hours. To reflect this, and match the periodic spikes that we observed in the empirical inter-arrival times distribution, Act-M assumes that the probability p_a of a user becoming active changes over time. We use a clock variable t_{clock} to keep track of the current time of day, where $0:00:00h < t_{\text{clock}} < 23:59:59h$. The clock variable t_{clock} is advanced after each state transition. For example, when an interval δ is generated in the rest or active states, the clock variable is advanced by δ . Additionally, Act-M uses the variables t_{wake} and t_{sleep} to define the time of the day the model finishes and starts its sleep mode. The probability p_a is given by the following equation:

$$p_a = \begin{cases} p_{\text{awake}} & \text{if } t_{\text{wake}} < t_{\text{clock}} < t_{\text{sleep}}, \\ p_{\text{sleep}} & \text{otherwise.} \end{cases} \quad (3.4)$$

where p_{awake} and p_{sleep} denote the transition probability to the active state when Act-M is in the awake and sleep modes and $p_{\text{awake}} > p_{\text{sleep}}$.

Act-M can fit the distribution of inter-arrival times without finding the exact values of the t_{wake} and t_{sleep} variables. If we assume that $t_{\text{wake}} = 0$, we can replace the parameters t_{wake} and t_{sleep} by a single parameter r_{sleep} that corresponds to the fraction of the day that is considered sleep time by Act-M. In this case, $t_{\text{sleep}} = r_{\text{sleep}} \cdot 24h$ when $t_{\text{wake}} = 0$.

Figure 3.9 illustrates the key aspects of the Act-M model. Figure 3.9(a) shows how the probability p_a (solid red line) changes over time. Notice that the curve for the probability p_a corresponds to a square wave in which valleys indicate daily intervals of low activity. The transition probability p_r , in contrast, is constant over time. In Figure 3.9(b), the duration of the rest and active states are indicated by dashed and solid arcs, and the state transitions are shown as black circles. Figure 3.9(c) shows the generated synthetic time-stamps which occur after each active state.

Algorithm 1 describes the procedure to generate time-stamps using the Act-M model. The inputs are the model parameters and the desired number of time-stamps N that we want to generate. This algorithm assumes that the transition probability to the active state during the sleep mode is

Figure 3.9: Time-stamps generated by the Act-M model. (a) The transition probability to the active state p_a is indicated by a solid red line and has a daily periodicity. (b) The duration of the active and rest states are indicated by solid and dashed lines, respectively. On average, the active state is shorter than the rest state. (c) Postings events (time-stamps) generated during the active state.

zero, that is, $p_{sleep} = 0$. This is based on our empirical evaluation when fitting real data (Section 3.5). The variable *currentState* stores the current state of Act-M. In our implementation, we make Act-M start on the active state. This decision is arbitrary and if the number of time-stamps generated is sufficiently large, this will not affect the distribution of inter-arrival times. We use an auxiliary variable δ to store the last Self-Correlated Process random generated value. The δ variable is reseted whenever there is a state transition. The procedure RAND generates a uniformly distributed number in the interval $[0, 1]$. Finally, the procedure ISAWAKE returns true if Act-M is not in sleep mode.

Algorithm 1 Algorithm to generate time-stamps using the Act-M model.

Input Parameters $\theta = \{p_r, p_a, \lambda_A, \rho_A, \lambda_R, \rho_R, r_{sleep}\}$ and desired size N of the time-stamp sequence.

Output Sequence of time-stamps $T = [t_1, t_2, \dots, t_N]$.

```

1: CurrentState  $\leftarrow$  Active,  $\Delta \leftarrow 0$ ,  $i \leftarrow 1$ ,  $T \leftarrow [0]$ ,  $\delta \leftarrow 1/\lambda_A$ 
2: while  $i \leq N$  do
3:   if CurrentState = Active then
4:      $\delta \leftarrow \text{SCorr}(\lambda_A, \rho_A, \delta)$ ,  $\Delta \leftarrow \Delta + \delta$ 
5:      $i \leftarrow i + 1$ ,  $T[i] \leftarrow T[i - 1] + \Delta$ ,  $\Delta \leftarrow 0$ 
6:     if RAND()  $< p_r$  then
7:       CurrentState  $\leftarrow$  Rest,  $\delta \leftarrow 1/\lambda_R$ 
8:     end if
9:   else if CurrentState = Rest then
10:     $\delta \leftarrow \text{SCorr}(\lambda_R, \rho_R, \delta)$ ,  $\Delta \leftarrow \Delta + \delta$ 
11:    if ISAWAKE( $T$ ,  $\Delta$ ,  $r_{sleep}$ ) and RAND()  $< p_a$  then
12:      CurrentState  $\leftarrow$  Active,  $\delta \leftarrow 1/\lambda_A$ 
13:    end if
14:  end if
15: end while

```

3.4.3 Parameter Estimation

In order to estimate the parameters of Act-M, we propose an algorithm that minimizes the squared error between the empirical and synthetic logarithmic binned histogram of inter-arrival times. The algorithm starts by computing the empirical inter-arrival times histogram. The first bin of the histogram counts all samples in the interval $[\Delta_{\min}, \Delta_{\min} + w_1)$ where $\Delta_{\min}$ is the minimum inter-arrival value in the dataset and w_i denotes the width of the i -th histogram bin. The width w_i of the i -th bin is wider than the previous $(i - 1)$ -th bin by a fixed factor k . That is, $w_i = w_{i-1} \cdot k$. We denote the normalized counts in each histogram bin as c_i . Since the counts c_i are normalized, we have $\sum_i c_i = 1$.

The next step of the algorithm consists in computing the inter-arrival times histogram for the synthetic data. We use the Act-M model to generate M synthetic time-stamps. By using the same bin widths w_i from the empirical histogram, we count the number of synthetic samples $\hat{c}_i(\theta)$ in each bin, where θ denotes the set of Act-M parameters. Finally, we use the *Levenberg-Marquardt* algorithm [73], to search the parameter values θ that minimize the squared error between synthetic and real data bin counts:

$$\min_{\theta} \sum_i [c_i - \hat{c}_i(\theta)]^2 \quad (3.5)$$

Estimation for Small Sequences

Employing the logarithmic binned histogram allows our estimation algorithm to capture the periodic peaks of the inter-arrival times distribution. However, when the sample size of the data is not sufficiently large, the histogram is affected by noise. Although this is not a problem when fitting aggregate data from all users in a dataset, the histogram approach is not adequate when there is only the sequence of inter arrival times from a single user.

To overcome the problem of noise in the histogram when fitting small sequences of inter-arrival times, we propose an alternative approach when fitting data from individual users. Instead of minimizing the squared error between the empirical and synthetic histograms, we minimize the squared error between the empirical and synthetic kernel density estimates (KDE) of the inter-arrival times logarithms. The KDE from both real and synthetic data was computed using a standard normal kernel with mean $\mu = 0$ and standard deviation $\sigma = 1$. The bandwidth of the kernel function was selected to minimize the mean integrated squared error (MISE) under the assumption that the KDE could be approximated by a normal distribution [17].

The motivation for using the KDE is that, compared to the histogram, it is more robust to noise. However, there is a drawback when using the KDE-based parameter estimation: the relatively large kernel bandwidth – required to smooth the noise – may hide the periodic peaks of the inter-arrival times distribution.

3.5 Experiments

In this section we analyze how well Act-M is able to fit real data from social media services. We compare Act-M against two other state-of-the-art models: the cascading non-homogeneous Poisson Process (CNPP) proposed in [71] and the Self-Feeding Process proposed in [99]. We

Table 3.2: Estimated Act-M parameters for the datasets. The estimated Act-M parameters were used to plot all figures in Section 3.5.1.

Dataset	p_r	p_a	$\frac{1}{\lambda_A}$ (min)	ρ_A	$\frac{1}{\lambda_R}$ (min)	ρ_R	$r_{\text{sleep}} \cdot 24\text{h}$
Reddit	0.53	0.06	4'54"	1.2	18'56"	0.9	7h 11m
Twitter	0.59	0.13	1'27"	1.6	25'34"	1.0	9h 15m
Stack-Overflow	0.44	0.10	10'16"	0.7	20'33"	1.0	8h 30m
Hacker-News	0.64	0.08	8'28"	1.1	36'35"	1.0	7h 42m

estimate the parameters of all models using the algorithm described in Section 3.4.3. We start in Section 3.5.1 by fitting aggregate data, that is, the IAT of the postings for all users in each dataset. Next, in Section 3.5.2 we use Act-M to fit data from individual users.

3.5.1 Fitting Aggregate Data

In Section 3.3 we showed that the distribution of postings' inter-arrival times has the following properties: (i) heavy tails; (ii) bimodality; (iii) periodic spikes and (iv) correlation between consecutive values. Our goal in this section is to show that our Act-M model is able to match all these four patterns. Table 3.2 shows the estimated Act-M parameters (obtained using the algorithm described in Section 3.4.3) that we use for all figures in this section.

We start by analyzing whether our Act-M model is able to recreate the periodic spikes and the bimodal distribution that we observed in real data (Observations 3.2 and 3.3). Figure 3.10 shows the logarithmic binned histogram of IAT for all users in each dataset. We compare the real data distribution (gray area) to the distribution of synthetic IAT generated by Act-M, SFP and CNPP, which were presented in Chapter 2. For each dataset, we also show the root-mean-square error (RMSE) between the models' and the data's histograms. Accurately fitted results are indicated by lower RMSE, while a RMSE value equals to zero indicates a perfect fit. The SFP model fails to match both the periodic spikes and the bimodal distribution from the data as it lacks a mechanism that accounts to the fact that users may sleep or have bursts of activity. The CNPP model matches the bimodal distribution, but fails to generate the periodic spikes. In contrast, our Act-M model accurately matches the real data histogram from all datasets, including the periodic spikes and the bimodal distribution.

In order to investigate if Act-M generates a heavy tailed distribution as we observed in the social media data in Section 3.3, we plot in Figure 3.11 the complementary cumulative distribution function (CCDF) of real and synthetic inter-arrival times. Only the Act-M and SFP models generate heavy-tailed inter-arrival times distributions while CNPP fails to match the tail part of the empirical distribution. This indicates that the non-homogeneous Poisson-Process proposed by the CNPP model is not able to match the tail-part of the IAT distribution. Moreover, while SFP produces heavy-tailed distributions, it fails to fit the slope of the data as Act-M does.

Finally, we verify if Act-M produces correlated consecutive IAT as observed in real data (Observation 3.4). Figure 3.12 compares the heat-maps of consecutive pairs of IAT of real and syn-

Figure 3.10: Act-M (solid blue line) matches both the periodic spikes and bimodal IAT distribution of the real data. Both the gray area (real data) and the lines (synthetic data) correspond to the logarithmic binned histogram of IAT. For all fits, we also show the root mean squared error (RMSE) obtained by each model.

Figure 3.11: Act-M (solid blue line) matches the heavy tailed distribution of the real data (gray dots). Due to the accuracy of the Act-M fit, the blue line is often invisible due to occlusion.

thetic time-stamps. The procedure to generate the heat-maps was described in Section 3.3. Act-M provides the best match for the real data when compared to the SFP and CNPP models. In particular, the heat-maps generated by Act-M have a concentration of pairs along the diagonal, which indicates that it is able to model the correlation between consecutive IAT as observed in the real data. Act-M also matches the crests centered at $\Delta_i \approx 1$ day or $\Delta_{i+1} \approx 1$ day which are analogous to the periodic spikes observed in the logarithmic binned histogram (Figure 3.10). Since the CNPP relies on Poisson Processes to generate IAT, it fails to produce both the crests and correlation of consecutive IAT, while the SFP model generates a correlation between consecutive IAT that deviates significantly from the real data.

3.5.2 Fitting Individual Users Data

The number of postings generated by most of social media users is often not large enough to plot a logarithmic binned histogram as we show in Figure 3.10 when we analyze the aggregate data from all users in each dataset. Therefore, we use the kernel density estimation (KDE) to analyze how well Act-M and its competitors are able to fit data from individual users. Figure 3.13 compares the KDE of the real data (black dots) and the models. We selected, from each dataset, the users at the first, second (median) and third quartile positions with respect to the number of postings. Act-M obtained the most accurate fit and was closely followed by the CNPP model, while the

Figure 3.12: The heat-maps depict the distributions of pairs of consecutive IAT. Notice that both Act-M (second column) and the real data (first column) have a concentration of pairs along the diagonal of the heat-maps. This shows that Act-M generates correlated consecutive IAT as observed in real data.

Figure 3.13: Fits for the IAT kernel density estimation (KDE) of individual users' postings. The black dots correspond to the data. From each dataset we selected the users in the median and first and third quartile positions with respect to the number of postings.

SFP model was not able to fit the data. Although Act-M and CNPP obtained similar results when fitting the KDE of individual users, CNPP is only able to fit the IAT distribution when there is a small number of postings. On the other hand, Act-M has the advantage of also explaining temporal patterns, such as periodic spikes and heavy tails which are only observable when we have a large number of postings.

To provide a quantitative comparison of the models when fitting users' IAT kernel density estimation, we computed the root mean squared error (RMSE) between the users' and the models' KDE. To compute the RMSE, we compared the users' and the models' KDE values over 100 logarithmic-spaced inter-arrival times. Table 3.3 shows the median RMSE obtained by the models in each dataset. Act-M consistently obtained the smallest median RMSE for all datasets.

Table 3.3: Median Root Mean Squared Error (RMSE) when fitting users' IAT kernel density estimation.

Dataset	SFP	CNPP	Act-M
Reddit	$2.63 \cdot 10^{-3}$	$0.37 \cdot 10^{-3}$	$0.33 \cdot 10^{-3}$
Twitter	$3.63 \cdot 10^{-3}$	$0.49 \cdot 10^{-3}$	$0.35 \cdot 10^{-3}$
Stack-Overflow	$4.39 \cdot 10^{-3}$	$0.40 \cdot 10^{-3}$	$0.31 \cdot 10^{-3}$
Hacker-News	$2.81 \cdot 10^{-3}$	$0.38 \cdot 10^{-3}$	$0.32 \cdot 10^{-3}$

Since Act-M can accurately model the IAT distribution of individual users, we use it to analyze the distribution of users in a social media service with respect to the Act-M parameters. In Figure 3.14 we show the distribution of the transition probabilities p_r and p_a for a sample of 1,000 users from each dataset. For all datasets, the transition probabilities are distributed as a single cluster, which we indicate as a red ellipse. However, for some outliers the values of the transition probabilities deviate from the cluster. For each dataset we selected a user, indicated by a colored diamond in the p_r vs. p_a scatter plot and plotted the KDE fit obtained by the Act-M model.

By analyzing users whose parameters deviate from the distribution observed in its respective dataset, we found interesting patterns. First, the KDE of the postings' IAT often had a single mode which can be attributed to the high value of the transition probability p_a . For the Reddit dataset (Figure 3.14(a)), most of the users located outside the red ellipse are bots with user names such as *moderator_bot*, *bitcointip* and *QualityEnforcer*. In the Stack-Overflow dataset, we observed that outliers included Web site administrators such as the user Marc Gravell (ID 23345) or the user Jon Skeet (ID 22656) who has the highest reputation score in Stack-Overflow. A similar pattern was observed for the Hacker-News dataset, where the outlier highlighted in Figure 3.14(d) is the Web site developer (user *dang*).

3.5.3 Running Time Analysis of Parameter Estimation

The parameter estimation algorithm (Section 3.4.3) uses M synthetic time-stamps to approximate Act-M's inter-arrival time distribution (Equation 3.5). As we increase M , the approximation of Act-M's inter-arrival time distribution becomes more accurate. However, as the optimization algorithm changes Act-M's parameters to minimize the squared error of Equation 3.5, the synthetic time-stamps must be generated again with new parameters. As a result, the running time of the parameter estimation algorithm grows as we increase the number M of synthetic time-stamps used to approximate the IAT distribution.

The left column of Figure 3.15 shows the relationship between the parameter M and running time of Act-M's parameter estimation. In the right column of Figure 3.15, we plot the root-mean-square error (RMSE) between the Act-M's and the data IAT histogram. The markers and the error bars indicate the mean and standard deviation of the measures over ten runs of the fitting algorithm for each dataset. The running times of the Act-M fitting algorithm were measured in a MacBook Pro computer with a 2.6GHz Intel Core i5 processor, 8GB of RAM with the Mac OS X 10.10. The Act-M model was implemented and executed using Matlab R2013.

Figure 3.14: Act-M parameters can be used to detect anomalies. The Act-M transition probabilities p_r and p_a are distributed as a single cluster for the majority of the users as indicated by the red ellipsis. We found that users that deviate from this pattern are often bots or Web site administration staff.

Figure 3.15: Act-M fitting time. As the number M of synthetic time-stamps used to approximate Act-M's distribution increases, the running time of the estimation algorithm increases while the fitting error (RMSE) decreases. The markers and the error bars indicate, respectively, the mean and standard deviation of the measures over 10 runs of the fitting algorithm.

3.6 Detecting Bots

In this section, we show that the Act-M can also be used to detect bots. As previously discussed (Section 3.2), we consider all users controlled by computer programs as bots. Specifically, we want to solve the following problem: given time stamp data from a set of users $\{\mathcal{U}_1, \mathcal{U}_2, \dots\}$ where each user $\mathcal{U}_i$ has a sequence of postings inter-arrival times $\Delta_i = (\Delta_1, \Delta_2, \Delta_3, \dots)$ we want to decide if user $\mathcal{U}_i$ is a *human* or a *bot*.

To solve this problem, we propose ACTM-SPOTTER, a method for bot detection that extracts temporal features from users' time stamp data and uses a cost-sensitive classification scheme to decide if users are bots. The ACTM-SPOTTER method employs three temporal features: Act-M-Distance, D-Score and L-Score which are discussed in Section 3.6.1. In Section 3.6.2 we describe the cost-sensitive classification scheme used by ACTM-SPOTTER. Finally, in Section 3.6.3 we evaluate ACTM-SPOTTER.

3.6.1 Temporal Features

The first feature used by ACTM-SPOTTER is the Act-M-Distance, which uses the Act-M model to compare the distribution of inter-arrival times of each user against the aggregate distribution of all users in the dataset. The rationale of Act-M-Distance feature is that users whose inter-arrival times distribution are significantly different from the aggregate distribution are suspicious. To extract the Act-M-Distance feature, we start by estimating the Act-M parameters θ using the aggregate inter-arrival time from a social media service. By using the estimated parameters, we generate for each user $\mathcal{U}_i$ a sequence of synthetic inter-arrival times of size M , where M denotes the size of the inter-arrival sequence for user $\mathcal{U}_i$. Finally we compute the logarithmic binned histogram of inter-arrival times for the user and synthetic data. The Act-M-Distance feature is given by the distance between the two histograms.

Algorithm 2 describes the procedure employed to compute the Act-M-Distance feature. The algorithm takes as input a sequence of inter-arrival times from a user, denoted by $\Delta_{\mathcal{U}} = (\Delta_1, \Delta_2, \dots, \Delta_M)$ and the estimated Act-M parameters θ . The procedure LOGBINHIST(A, b) computes the logarithmic binned histogram of sequence of values A with b bins. The procedure ACTM-GENERATE (Algorithm 1) generates a synthetic sequence of inter-arrival times $\hat{\Delta}$ using the Act-M model and is described in Section 3.4.2.

Algorithm 2 Algorithm to compute the Act-M-Distance feature.

Input User inter-arrival times $\Delta_{\mathcal{U}} = (\Delta_1, \Delta_2, \dots, \Delta_M)$ and Act-M parameters θ .

Output Act-M-Distance feature.

- 1: $(c_1, c_2, \dots, c_K) \leftarrow \text{LOGBINHIST}(\Delta_{\mathcal{U}}, K)$
 - 2: $\hat{\Delta} \leftarrow \text{GENERATE}(\theta, M)$
 - 3: $(\hat{c}_1, \hat{c}_2, \dots, \hat{c}_K) \leftarrow \text{LOGBINHIST}(\hat{\Delta}, K)$
 - 4: Act-M-Distance $\leftarrow \sum_{j=1}^K |c_j - \hat{c}_j|$
-

The two remaining features used by ACTM-SPOTTER (D-Score and L-Score) are computed by using a pair-counting matrix D similar to that used to build the heat-maps of consecutive IAT in Section 3.3. The D matrix is generated by dividing the Δ_i vs. Δ_{i+1} space into a logarithmic

spaced grid of dimensions $m \times m$. The entries $D_{i,j}$ of the matrix counts the number of pairs (Δ_i, Δ_{i+1}) of consecutive inter-arrival times in each grid cell. The D-Score feature corresponds to the fraction of the entries in the pair-counting matrix diagonal:

$$\text{D-Score} = \frac{\sum_{i=1}^m D_{i,i}}{\sum_{i=1}^m \sum_{j=1}^m D_{i,j}} \quad (3.6)$$

where $0 \leq \text{D-Score} \leq 1$. A D-Score close to one indicates that the user is posting comments at regular intervals (e.g. every 15 minutes). Figure 3.16(b) shows the pair-counting matrix for a Reddit user with a large D-Score (0.45). In this particular case, the user is posting comments at every 45 minutes indicating automated behavior. For comparison, Figure 3.16(a) shows a user with organic behavior.

The L-Score feature is the largest sum of the entries located at the k -th line and column of the pair-counting matrix:

$$\text{L-Score} = \max_k \left\{ \frac{\sum_{i=1}^m D_{i,k} + \sum_{i=1}^m D_{k,i} - 2 \cdot D_{k,k}}{\sum_{i=1}^m \sum_{j=1}^m D_{i,j}} \right\} \quad (3.7)$$

where $0 \leq \text{L-Score} \leq 1$. A large L-Score indicates that a user posts comments at alternated regular intervals and is also an indication of automated behavior. Figure 3.16(c) shows the pair-counting matrix for the user with the second highest L-Score (0.50) from the Reddit dataset. The user shown in Figure 3.16(c) alternates between posting comments at every 4 seconds and 5 minutes.

Figure 3.16: Reddit users with different commenting patterns captured by the ACTM-SPOTTER features. Each figure shows the pair-counting matrix D for different Reddit users. (a) Pair-counting matrix for a human user. (b) Pair-counting matrix of a bot that posts at regular intervals (every 45 minutes). (c) Pair-counting matrix of a bot that alternates between posting at every 4 seconds and 5 minutes.

3.6.2 Cost Sensitive Classification

After ACTM-SPOTTER extracts the set of three features (Act-M-Distance, D-Score and L-Score) from each user, it is necessary to identify which users are bots. Given a training set of users labeled either as bots (positive examples) or humans (negative examples), we train a Naive-Bayes classifier to estimate the posterior probability p_{bot} that a user is a bot given the extracted features. If p_{bot} is higher than a decision threshold p_{thresh} , then the user is classified as a bot.

Based on [31], we estimate the decision threshold p_{thresh} by assigning a cost c_{FN} to false negative (FN) errors and a cost c_{FP} to false positive (FP) errors. A false negative error occurs when a bot is classified as a human. A false positive error occurs when a human is classified as a bot. We select the threshold p_{thresh} that minimizes the F_β -Measure on the training set:

$$F_\beta = \frac{(1 + \beta^2) \cdot \text{TP}}{(1 + \beta^2) \cdot \text{TP} + \beta^2 \cdot \text{FN} + \text{FP}} \quad (3.8)$$

where $\beta = \sqrt{c_{\text{FN}}/c_{\text{FP}}}$.

3.6.3 ACTM-SPOTTER Evaluation

We evaluated ACTM-SPOTTER using a random sample of 1,963 Reddit users manually labeled as humans in addition to 37 bots. We also randomly selected 1,353 Twitter users that were identified by Twitter as humans (e.g. music celebrities and politicians) in addition to a set of 64 Twitter bots. To classify the users as humans or bots, we randomly split the datasets into train and test subsets of the same size while preserving the class distribution. The train subset is used to train a Naive-Bayes classifier, estimate p_{thresh} and the Act-M parameters θ . Each experiment was repeated ten times. In our experiments, we set the number K of bins of the log-binned histogram (Algorithm 2) to 30. Although there are many methods that try to automatically select the number of bins of a histogram, for simplicity, our decision was based on the rule $K = \sqrt{n}$, where n is the sample size (i.e. number of postings). We compared ACTM-SPOTTER against the following features extracted from the sequence of time-stamps of each user:

1. *IAT Histogram*: Logarithmic binned histogram of postings inter-arrival times;
2. *Entropy*: Entropy of the IAT histogram, proposed in [24].

Figure 3.17 compares the precision vs. sensitivity (recall) curve obtained by ACTM-SPOTTER and competitors. A good performance is indicated by a curve closer to the top part of the plot. For the Twitter dataset (Figure 3.17(b)), ACTM-SPOTTER obtained the highest precision for all sensitivity values, indicating that for any configuration of FP and FN costs, it is better than the other features. For the Reddit dataset (Figure 3.17(a)), ACTM-SPOTTER obtained considerably higher precision for sensitivity values smaller than 70%, and precision values closer to the other methods for sensitivity values larger than 70%.

Even though the datasets are strongly imbalanced, with significantly more humans than bots, ACTM-SPOTTER obtained a precision of 96.5% and 94.7% for sensitivity values of 47.9% and 70.3% for the Reddit and Twitter datasets, respectively.

Figure 3.17: Precision vs. sensitivity (recall) curve for the task of bot detection. ACTM-SPOTTER obtained the highest precision for the majority of sensitivity (recall) values in both datasets. A good performance is indicated by a curve closer to the top part of the plot.

3.7 Conclusions

In this chapter, we presented Act-M (Activity Model), a model that accurately fits the distribution of inter-arrival times from social media users' postings. Act-M was designed based on our observations of data from several social media services. Specifically, we found that the inter-arrival time between postings is characterized by four activity patterns:

1. **Heavy-Tailed Distribution:** The distribution of inter-arrival times is heavy-tailed (Observation 3.1).
2. **Bimodal Distribution:** The distribution of inter-arrival times has two modes (Observation 3.2).
3. **Periodic Spikes:** The circadian rhythm affects users' postings times, generating periodic peaks in the inter-arrival times distribution at every 24-hour interval (Observation 3.3).
4. **Positive Correlation:** There is a dependency between the arrival time of consecutive postings (Observation 3.4).

Part of the temporal patterns that we have observed – such as the heavy tailed distribution – agrees with previous studies in human communication [10, 99]. However, to the best of our knowledge, the existence of all four discovered patterns in several social media services is a novel finding that we presented in this chapter.

In Chapter 2, we used Table 4.5 to compare the capabilities of different models for communication dynamics. We now update Table 4.5 to also include the Act-M model. The result is Table 3.4, which shows that only Act-M is capable of explaining all the discovered patterns using a single mathematical model. Moreover, Act-M was able to provide an accurate fit to the inter-arrival times distribution of the real data.

Finally, we also show that Act-M can be used to spot bots in social media. We proposed ACTM-SPOTTER, a method that uses Act-M to tell if users are humans or bots based solely on

Table 3.4: Communication patterns matched by different models. Only Act-M is able to describe all patterns.

Pattern	Poisson-Process	Log-Logistic	Power-Law	CNPP	SFP	Act-M
Heavy Tails		✓	✓		✓	✓
Bimodality				✓		✓
Periodic Spikes						✓
Correlation					✓	✓

the timing of their postings. In our experiments, ACTM-SPOTTER obtained a higher precision than the competitors for different values of sensitivity (recall). Additionally, even though the datasets for the bot detection task were heavily imbalanced, with less than 5% of positive examples (bots), ACTM-SPOTTER obtained a precision higher than 93% and 77% for a sensitivity of 70% considering the Twitter and Reddit datasets, respectively.

Chapter 4

Coevolution of User Interactions

Social media services allow users to interact with content in different ways. For instance, when a user uploads an image, other users can share, comment or express approval by using features such as a like button. In this chapter we use data from social voting services to investigate the relationship between different types of user interactions in social media.

4.1 Introduction

As we discussed in Section 2.1.2, social voting services are a type of social media service in which users can employ votes and comments to curate content. For example, consider [Imgur](http://imgur.com/)¹, a social voting service which focuses on image content. Imgur users can indicate that they like or dislike an image by clicking on up-vote or down-vote buttons. Additionally, users can post short textual comments related to each image. Images that gather more up-votes are shown more prominently on the Imgur website and mobile application and consequently attract even more user interactions. Other social voting services behave similarly to Imgur. [Reddit](http://www.reddit.com/)², for example, also allows users to use up-votes, down-votes and comments. Differently from Imgur, however, Reddit users can submit other types of content in addition to images, such as links to news articles or videos.

In this Chapter, our goal is to study how the volume of different types of user interactions received by a piece of content submitted to a social media service evolve over time. Since we focus on social voting services, the types of user interactions that we consider are up-votes, down-votes and comments.

We analyzed data from three popular social voting services: Reddit and Imgur, which are, respectively, the 18th and 50th most visited websites in the world³ as well as historical data from Digg [65]. We show in this chapter that user interactions (up-votes, down-votes and comments) are characterized by three patterns: (i) the count of comments received by a submission over time grows as a power-law on the count of votes; (ii) for popular submissions, the time for reaching the peak of their popularity decays with respect to the number of votes received; and (iii) user reaction times – i.e. the time interval between a submission creation and a user interaction – can be accurately modeled by a log-logistic distribution.

¹<http://imgur.com/>

²<http://www.reddit.com/>

³Source: www.alexacom.com, accessed in March 2017.

Figure 4.1: Examples of how VnC can fit the three curves: (a) up-votes time-series, (b) up-votes vs. down-votes and (c) votes vs. comments.

Based on the discovered patterns, we propose the Vote-and-Comment (VnC), a mathematical model that describes how the up-votes, down-votes and comments change over time for a given submission. In our experiments (Section 4.5), VnC was consistently more accurate than state-of-the-art baselines. Figure 4.1 shows the up-votes, down-votes and comment data from a link submitted to Reddit. VnC, indicated by a solid line, is able to accurately fit three curves: up-votes time-series (Figure 4.1(a)), number of up-votes vs. down-votes (Figure 4.1(b)) and number of comments vs. number of votes (Figure 4.1(c)).

Finally, as the final contribution of this chapter, we show that VnC can forecast up-vote, down-vote and comment time-series. VnC provided a more accurate prediction for the final number of user interactions when compared to competitors. We also used VnC to spot outliers and cluster different types of content (pictures vs. discussions) based on user activity.

The remainder of this chapter is organized as follows. In Section 4.2 we define the concepts used throughout this chapter. In Section 4.3 we present observations from our analysis of social voting services data. The VnC model is introduced in Section 4.4. The experiments and applications involving the VnC are presented in Sections 4.5 and 4.6. Finally, we present the concluding remarks for this chapter in Section 4.7.

4.2 Definitions

We define a *submission* as any item that a user can upload to a social voting service. Examples of items that may be submitted to a social voting service include images and URLs to news articles. A *user interaction* is the act of up-voting, down-voting or commenting on a submission. For each submission, there are three time-series: $V_+(t)$, $V_-(t)$, $C(t)$ which denote, respectively, the number of up-votes, down-votes and comments accumulated by a submission over time. The derivatives of

the time-series $V_+(t)$, $V_-(t)$, $C(t)$ are denoted by corresponding lower-case letters: $v_+(t)$, $v_-(t)$, $c(t)$.

4.3 Activity in Social Voting Services: Observations

In this section, we analyze data from three social voting services: Reddit, Imgur and Digg. The combined datasets correspond to over 300,000 submissions. Our goal is to find patterns between different user interactions (up-votes, down-votes and comments).

4.3.1 Datasets

We developed a crawler to collect data from two social voting services: Reddit and Imgur. We tracked Reddit and Imgur submissions over a period of 20 and 34 days respectively. Every ten minutes, our crawler added to its tracking list the first 150 submissions appearing on Reddit news sections and the first 50 submissions appearing on Imgur news section. We used Reddit's and Imgur's API to collect every 20 minutes a snapshot of each submission data: the number of up-votes, down-votes and comments. Each submission was tracked for 33 hours and 20 minutes, starting from within 10 minutes from its submission. Our crawler stopped tracking a submission after 33 hours and 20 minutes due to the APIs limitation on the number of requests per hour and because the submissions usually did not receive a significant number of votes and comments after 33:20h of their creation. Finally, we added to our datasets all submissions with at least 100 up-votes. The selected submissions attracted over 97% of the votes and comments. Table 4.1 summarizes the number of submissions and user interactions (votes and comments) for each dataset.

For each submission, we used the collected data to generate time-series for the number of up-votes, down-votes and comments. Each time-series contains 100 data-points, where each data point (time-tick) corresponds to an interval of 20 minutes. We also analyze publicly available historical Digg data⁴ that was originally studied in [65]. The Digg data consists of 3,553 up-votes time-series and does not contain down-vote and comment data.

4.3.2 Relationship Between Comments and Votes

We start the exploration of the social voting services datasets by analyzing the correlation between the total number of votes and comments received by submissions. For this analysis, we included

⁴Digg Dataset: <http://www.isi.edu/~lerman/downloads/digg2009.html>

Table 4.1: Summary of the social voting services datasets.

Dataset	# Submissions	# User Interactions
		(Votes and Comments)
Reddit	17,205	113,331,266
Imgur	724	2,107,576
Digg	3,553	5,149,170

Figure 4.2: Joint distribution of votes (up-votes plus down-votes) vs. comments accumulated by Reddit and Imgur submissions. The number of votes and comments received by submissions is correlated, with Pearson coefficients of 0.39 and 0.78 for the Reddit and Imgur submissions, respectively.

only submissions with at least 10 comments. Since the Digg dataset does not contain comment data, we do not use it for the analysis presented in this section. The heat-maps from Figure 4.2 depict the joint distribution between the total number of comments $C(t)$ and votes $V_+(t) + V_-(t)$ accumulated by submissions of the Reddit and Imgur datasets. The two types of user interactions are correlated for both datasets. More specifically, the Pearson correlation between the total votes and comments are 0.39 and 0.78 for the Reddit and Imgur submissions, respectively.

While the heat-maps from Figure 4.2 depict the correlation between the final number of votes and comments received by submissions, they do not explain how the relationship between the number of comments and votes evolve over time. We evaluated two hypotheses to describe how the number of comments $C(t)$ accumulated by a submission grows with respect to the number of votes $V_+(t) + V_-(t)$ accumulated by a submission:

- **H1 - Linear Relationship:** The number of comments $C(t)$ grows linearly with respect to the number of votes $V_+(t) + V_-(t)$, that is: $C(t) = a [V_+(t) + V_-(t)]$;
- **H2 - Power-law Relationship:** The number of comments $C(t)$ grows as a power-law on the number of votes $V_+(t) + V_-(t)$, that is: $C(t) = k [V_+(t) + V_-(t)]^\alpha$.

We fitted the two hypotheses to data from all submissions and computed the coefficient of determination R^2 . Accurately fitted results are indicated by R^2 values closer to 1, while $R^2 = 1$ corresponds to a perfect fit. Table 4.2 shows the mean R^2 obtained by each model. The power-law model obtained statistically significantly larger R^2 values than the linear model for both datasets with $p = 0.01$. Based on this result, we propose the Vote-Comment (VC) Law:

Observation 4.1. VC Law: *The relationship between the number of votes and comments received by a submission can be accurately described by a power-law.*

In Figure 4.3, we plot the comments vs. votes trajectory for four selected submissions from the Reddit and Imgur datasets. The VC Law is indicated by the solid blue line and accurately

Table 4.2: VC Law: the relationship between the number of comments and votes received by a submission can be accurately described by a power-law. The values are mean R^2 and standard deviation obtained by fitting real data from Reddit and Imgur.

Model	Reddit	Imgur
Power-Law Relationship	0.95 ± 0.07	0.93 ± 0.10
Linear Relationship	0.87 ± 0.40	0.82 ± 0.27

Figure 4.3: The number of comments can be modelled as a power-law on the number of votes (VC Law).

captures the power law relationship in the growth rate of votes and comments. The α parameter, which captures the slope of the comments and votes' trajectory in a log-log scale plot, is also indicated. For comparison, we also included the fit obtained by the linear relationship hypothesis, indicated by a red dashed line. The linear relationship fails to match the slope of the comments vs. votes' trajectory.

4.3.3 Peak Time

In this section, we analyze how different aspects of a submission affect the peak time of the time-series of user interactions. We define the peak-time t_p of a time-series as the time it takes to reach its maximum value. For example, the peak-time t_p of the up-vote times-series $v_+(t)$ is given by:

$$t_p = \operatorname{argmax}_t v_+(t) \quad (4.1)$$

Figure 4.4: Submissions with a larger number of up-votes reach their peak faster. We built the plot by grouping submissions into logarithmic buckets based on the number of total up-votes. The gray dots indicate the median peak-time of the up-vote time-series of the submissions in each bucket. The blue line corresponds to the Urgency Law (Observation 4.2).

We start by answering the following question: a submission that receives a large number of up-votes will peak faster or slower than a less popular submission? Figure 4.4 shows the median peak times for submissions with a different number of total up-votes. We built the plot by grouping submissions into logarithmic buckets based on the number of total up-votes. Next, for each bucket, we computed the median peak times. Figure 4.4 shows that the final number of up-votes V_{total} of a submission is related to its peak time. We name this relationship “Urgency Law”:

Observation 4.2. Urgency Law: *the peak time t_p of the up-votes time-series of popular submissions (with at least 3,000 up-votes) decays approximately linearly with respect to the logarithm of the total number of up-votes: $t_p \approx b - a \log V_{total}$.*

Figure 4.4 indicates that popular submissions reach the peak of their popularity faster. The fit obtained by our proposed Urgency Law is indicated by a solid blue line which matches the real data. The values obtained for the coefficients a and b were $a = 2.46$ and $b = 29.5$.

4.3.4 Reaction Times

We also analyzed how long it takes for users to interact with submissions after they are created. We start by defining *reaction times*:

Definition 4.1 (Reaction Time). *The reaction time Δ_R of a given user interaction corresponds to the time interval between the instant t_o in which a submission is created and the instant t_r in which an interaction (vote or comment) occurs.*

Figure 4.5 shows the distribution of reaction times of up-votes, down-votes and comments for all submissions from the Reddit, Imgur and Digg datasets. The solid red line corresponds to the log-logistic fit to the data whose PDF is given by:

$$f(x) = \frac{(s/\mu)(x/\mu)^{s-1}}{(1 + (x/\mu)^s)^2} \quad (4.2)$$

where μ and s denote the scale and shape parameters, respectively. We estimated the parameters μ and s of the log-logistic distribution using maximum likelihood estimation (MLE).

Figure 4.5: Reaction times can be described by a log-logistic distribution. The parameters μ and s were estimated using MLE and the log-logistic fit is indicated by the solid red line.

A common assumption of previous works is that reaction times in human communication follow power-law distributions [10, 74]. While power-laws are able to match the tail part of the reaction time distribution, they do not explain the distribution for small reaction times. Figure 4.5 shows that the log-logistic is a more robust model as it can be used to model the distribution of small reaction times in addition to the heavy-tail:

Observation 4.3. *The distribution of reaction times for votes and comments in social voting services can be accurately modeled by the log-logistic distribution.*

4.4 The VnC Model

In this section, we present VnC (Vote-and-Comment), a model that describes the coevolution of up-votes, down-votes and comments received by a submission. VnC was designed to accurately match the user interaction time-series from social voting services by taking into account the Observations 4.1 to 4.3 presented in Section 4.3. VnC is composed of three sub-models that describe the following relationships:

1. Number of up-votes over time;
2. Up-votes vs. down-votes;
3. Comments vs. votes.

First, in Section 4.4.1, we describe how VnC models the number of votes received by a submission over time. In Section 4.4.2 we describe how the number of up-votes and down-votes evolves after a submission is created. Finally, in Section 4.4.3, we show that the number of comments can be explained as a power law of the number of votes.

4.4.1 Modeling Up-Votes Over Time

VnC assumes that the probability $P(t)$ of a user up-voting a submission at time-tick t is given by the probability $P_{\text{Like}}(t)$ that the user will like the submission with the condition that the user has already observed the submission. Denoting the probability that the user will observe (i.e. react) to the submission at the time tick t as $P_{\text{React}}(t)$ and assuming that $P_{\text{React}}(t)$ and $P_{\text{Like}}(t)$ are independent, we have:

$$P(t) = P_{\text{Like}}(t) \cdot P_{\text{React}}(t) \quad (4.3)$$

The probability $P_{\text{Like}}(t)$ that a user will like a submission in VnC is governed by a *cascading mechanism*. That is, $P_{\text{Like}}(t)$ increases as the submission accumulates up-votes. We model this cascading mechanism as a linear fraction of the users that have already voted on that submission:

$$P_{\text{Like}}(t; \beta_+, \xi_+) = \xi_+ + \beta_+ \cdot \frac{V_+(t)}{N_+}, \quad (4.4)$$

where ξ_+ is the independent coefficient, β_+ is the cascading coefficient, N_+ is the population of potential voters and $V_+(t) = \sum_{i=1}^t v_+(i)$ is the number of votes accumulated by the submission at time-tick t .

The cascading coefficient β_+ reflects how strong the influence exerted by users on each other is. Larger values of β_+ mean that users are more likely to up-vote a submission based on the number of up-votes accumulated by that submission.

Based on Observation 4.3, we model the distribution P_{React} of reaction times using the log-logistic distribution with PDF given by Equation 4.2 and scale and shape parameters denoted by μ and s , respectively. The shape parameter μ corresponds to median reaction time while the scale parameter is related to the decay rate of P_{React} .

Since users on social voting services can vote only once on a submission, the number of users that can cast a vote on a submission is given by the difference $N_+ - V_+(t)$. As a result, the expected number of up-votes $v_+(t+1)$ received by a submission at time-tick $t+1$ is given by the product $P(t) \cdot [N_+ - V_+(t)]$:

$$v_+(t+1) = [N_+ - V_+(t)] \cdot P_{\text{Like}}(t; \beta_+, \xi_+) \cdot P_{\text{React}}(t; \mu, s) \quad (4.5)$$

where the initial conditions are $v_+(0) = 0$ and $V_+(0) = 0$.

Equation 4.5 assumes that a submission starts receiving votes as soon as it is created at time-tick $t = 1$. However, in certain social voting services a user can create a submission at time $t = 1$ but only share it later at time $t = t_s$. In this case, the probability of a user voting on a submission is $P(t) = 0$ if $t \leq t_s$ and $prob(t) = P_{\text{Like}}(t; \beta_+, \xi_+) \cdot P_{\text{React}}(t - t_s; \mu, s)$ for $t > t_s$.

Parameter Estimation: Given a time-series $v_+(t)$ of real data corresponding to the number of up-votes received by a submission and the estimated values $\hat{v}_+(t; \theta)$, we learn the parameters $\theta = \{N, \beta_+, \xi_+, \mu, s, t_s\}$ for our model by minimizing the sum of squared errors between $v_+(t)$ and $\hat{v}_+(t; \theta)$:

$$\min_{\theta} \sum_{t=1}^{t_m} [v_+(t) - \hat{v}_+(t; \theta)]^2 \quad (4.6)$$

We solve the least-squares problem from Equation 4.6 by using the *Levenberg-Marquardt* algorithm [73].

4.4.2 Modeling Up-Votes and Down-Votes

VnC assumes that the cascading and log-logistic reaction times mechanisms, which govern the evolution of up-votes, are the same for the down-votes time-series. That is, users are more likely to down-vote a submission as it accumulates down-votes (Equation 4.4), and the probability of a submission receiving a down-vote is modulated by the users' reaction time distribution. This results in the following equation for the down-votes time-series:

$$v_-(t+1) = [N_- - V_-(t)] \cdot P_{\text{Like}}(t; \beta_-, \xi_-) P_{\text{React}}(t; \mu, s) \quad (4.7)$$

with initial conditions $v_-(0) = 0$ and $V_-(0) = 0$.

The parameters μ and s that control the reaction times distribution as well as the start time t_s are shared between the up-votes (Equation 4.5) and down-votes time-series. The cascading coefficient β_- , independent coefficient ξ_- and the population of potential down-voters N_- are allowed to be different for the time-series $v_+(t)$ and $v_-(t)$.

We fit the parameters $\theta = \{N_+, \beta_+, \xi_+, N_-, \beta_-, \xi_-, \mu, s, t_s\}$ by solving the following least-squares problem:

$$\min_{\theta} \sum_{t=1}^{t_m} [v_+(t) - \hat{v}_+(t; N_+, \beta_+, \xi_+, \mu, s, t_s)]^2 + [v_-(t) - \hat{v}_-(t; N_-, \beta_-, \xi_-, \mu, s, t_s)]^2 \quad (4.8)$$

where $v_+(t)$ and $v_-(t)$ denote the real data and $\hat{v}_+(t)$ and $\hat{v}_-(t)$ are the estimated values for the up-votes and down-votes time-series.

4.4.3 Modeling comments

Based on the VC Law exhibited by real data (Observation 4.1), we model the number of comments $C(t)$ accumulated by a submission as a power-law on the number of votes $V_+(t) + V_-(t)$:

$$C(t) = k[V_+(t) + V_-(t)]^\alpha \quad (4.9)$$

Given real data from the time-series of accumulated votes $V_+(t) + V_-(t)$ and comments $C(t)$, we find the parameters k and α by minimizing the squared error between $C(t)$ and $k[V_+(t) + V_-(t)]^\alpha$.

Definition 4.2. *The Equations 4.5, 4.7 and 4.9 constitute the VnC model.*

4.5 Experiments

In this section, we evaluate how well VnC fits real data from the Reddit, Imgur and Digg social voting services. The Reddit and Imgur datasets were obtained by our crawlers and are described in Section 4.3.1. The Digg dataset is publicly available and was originally studied in [65]. We start in Section 4.5.1 by analyzing the VnC accuracy when fitting up-vote time-series and comparing the results to other models from the literature. Next, in Section 4.5.2 we analyze VnC ability to match the tail part of the time-series. Finally, Section 4.5.3 shows VnC results when fitting the coevolution of different types of user interactions.

4.5.1 Vote Time-Series

We start by comparing the VnC accuracy against the SI, Bass, Phoenix-R and Spike-M models, which we discussed in Chapter 2. SI and Bass are widely employed in the literature to model information cascades while Phoenix-R and Spike-M are state-of-the-art models. Figure 4.6 shows the scatter plots of the R^2 values obtained by VnC and a competitor. Accurately fitted results are indicated by R^2 values closer to 1, while $R^2 = 1$ corresponds to a perfect fit. Each point

corresponds to an up-votes time-series. In each scatter plot, points below the diagonal, in the yellow area, correspond to time-series that were best fitted by VnC. VnC obtained better results than its competitors for all datasets. More specifically, VnC provided a better fit than the SI, Bass, Phoenix-R and Spike-M models for over 98%, 90%, 95% and 88% of the time-series, respectively. This is indicated by a larger number of time-series (black points) below the diagonal of the scatter plots of Figure 4.6.

Figure 4.7 compares the root mean squared error (RMSE) obtained by VnC, SI, Bass, Phoenix-R and Spike-M models when fitting up-votes time-series from Reddit, Imgur and Digg. A lower RMSE indicates a better fit, and points below the diagonal in the yellow area correspond to time-series that were best fit by VnC. VnC obtained a smaller RMSE for over 91%, 99%, 96% and 90% of the time-series when compared to the Bass, SI, Phoenix-R and Spike-M models, respectively.

To conclude the quantitative analysis of VnC accuracy, Table 4.3 compares the mean R^2 values obtained by VnC, SI, Bass, Phoenix-R and Spike-M models for all up-vote time-series from the Reddit, Digg and Imgur datasets. For all datasets VnC obtained a statistically significantly higher R^2 when compared to its competitors with $p = 0.01$.

Table 4.3: Fitting accuracy: our VnC model obtained the highest mean R^2 values when fitting up-votes time-series from links submitted to Reddit, Imgur and Digg. The maximum possible R^2 value is 1, which indicates a perfect fit. In addition to the mean R^2 , the Table also shows the standard deviation.

Model	Reddit	Imgur	Digg
VnC	0.80 ± 0.21	0.88 ± 0.13	0.77 ± 0.16
Spike-M	0.76 ± 0.23	0.66 ± 0.18	0.67 ± 0.15
Phoenix-R	0.65 ± 0.30	0.55 ± 0.23	0.58 ± 0.22
Bass Model	0.75 ± 0.21	0.78 ± 0.15	0.55 ± 0.17
SI	0.69 ± 0.26	0.51 ± 0.19	0.31 ± 0.39

Finally, in Figure 4.8 we show the VnC fit for up-vote time-series. We selected the submissions to plot in Figure 4.8 based on the final number of up-votes. The first row, starting from the top, corresponds to the most up-voted submission from each dataset. The second, third and fourth rows correspond, respectively, to the submissions in the 3rd quartile, median and 1st quartile with respect to the number of up-votes. Although different submissions may have different rise and fall patterns, VnC, indicated by a solid blue line, accurately fitted the time-series from the different social voting services.

4.5.2 Heavy-Tail Decay

The left column of Figure 4.9 shows up-vote time-series from Reddit, Imgur and Digg in log-log scale. The yellow areas of the time-series correspond to the region after the peak-time.⁵ Like most of the up-vote time-series from Reddit, Imgur and Digg, the time-series depicted in Figure 4.9

⁵As discussed in Section 4.3, we define the peak-time of a time-series as the time it takes to reach its maximum value.

Figure 4.6: Comparison of VnC accuracy to existing models from the literature. Points below the diagonal in the yellow area correspond to up-votes time-series that were best fit by VnC. The bars above each plot show the fraction of time-series that were best fit by VnC. All time-series with at least 10 up-votes were selected.

Figure 4.7: VnC obtained the lowest fitting error (RMSE). Points below the diagonal in the yellow area correspond to time-series from Reddit, Imgur and Digg that were best fit by VnC.

have a heavy-tailed decay as their tail part (yellow area) can be approximated by a straight line. In the right column of Figure 4.9, we zoom into the tail part of the time-series and show the fit obtained by VnC and its competitors. While VnC model is able to match the heavy tail decay in the up-votes time-series, the Bass and Phoenix-R models generate a non-realistic exponential decay in the number of votes received over time.

Although the Spike-M model also generates heavy-tailed decays, VnC has the advantage of being *scalable* on the length (number of time-ticks) of the time-series. That is, VnC requires $O(n)$ operations to evaluate a time-series, while Spike-M has $O(n^2)$ runtime on the time-series length. This is shown in Figure 4.10, where we plot the time in seconds required to evaluate time-series of different lengths for VnC and Spike-M. Evaluation time was obtained in a computer with an Intel Core i5 3.2GHz processor, 16 GB of RAM running Mac OS X 10.9. Both methods were implemented in Matlab R2013.

4.5.3 Up-votes, Down-votes and Comments

In Figure 4.11, we analyze how well VnC fits the coevolution of up-votes and down-votes. We show the trajectory in the up-votes vs. down-votes plane of the most voted submission and the submission with the median number of votes in each dataset (Reddit and Imgur). We do not analyze the Digg dataset in this section as it does not contain down-vote or comment data. For both datasets, VnC (indicated by a solid blue line) closely fits the data. We also observe from Figure 4.11 that the trajectories in the up-vote vs. down-vote plane of a submission do not follow a simple straight line. This shows that although the total number of up-votes and down-votes are correlated for a given submission (see Figure 4.2), the relationship between up-votes and down-votes may change over time and VnC is able to accurately capture this behavior.

Figure 4.8: VnC fits for up-vote time-series from different social voting services: Reddit, Imgur and Digg. Submissions were selected from each dataset based on total number of up-votes received. Top row corresponds to the most voted submissions. The second, third and fourth rows correspond, respectively, to the 3rd quartile, median and 1st quartile with respect to the number of up-votes received.

In Figure 4.12 we use data from the same submissions depicted in Figure 4.11 to analyze the VnC fitting for the coevolution between comments and votes accumulated by submissions. The plots are in log-log scale, and the data can be approximated by a straight line, indicating a power-law relationship, as we discussed in Section 4.3. Again, the VnC fitting is indicated by a solid blue line that accurately captures the difference in the growth rate between votes and comments.

Figure 4.9: Heavy-tailed decay in up-vote-time series. The plots in the left column show up-vote time-series from Reddit, Imgur and Digg in log-log scale. The right column show the tail part of the same time-series in log-log scale. VnC model, indicated by a solid blue line, matches the heavy-tailed decay while the Bass and Phoenix-R models have unrealistic exponential decays.

Figure 4.10: Evaluation of VnC is linear on the length of time-series while Spike-M requires $O(n^2)$ operations.

Figure 4.11: VnC (blue line) fits the relationship up-votes and down-votes. Submissions to the left and to the right received, respectively, the median and the largest number of up-votes on each dataset.

Figure 4.12: VnC fits the relationship votes and comments (log-log scale). Submissions to the left and to the right received, respectively, the median and the largest number of up-votes on each dataset. For each submission, we also show the parameters

4.6 Applications

In the previous section, we showed that VnC accurately fits the up-vote, down-vote and comment time-series of different social voting services (Reddit, Imgur and Digg). Now we focus our analysis on how VnC can be employed in different applications: forecasting the number of user interactions received by submissions, detecting outliers and clustering submissions based on their content type (image vs. discussions).

4.6.1 Forecasting

The problem of forecasting the tail part of the up-vote, down-vote and comment time-series of a given submission can be stated as follows:

Problem 4.1. *Given the initial parts of the up-vote $v_+(t)$, down-vote $v_-(t)$ and comment time-series $c(t)$ of a submission for $t < t'$, predict the tail part of the time-series for $t > t' + 1$.*

In order to forecast the tail part of the time-series, we learn the VnC parameters using only data from the first six hours and 40 minutes after a submission is created (20 time-ticks). We start by using VnC and to predict the final number of up-votes, down-votes and comments received by a submission. Table 4.4 shows the median absolute percentage error (APE), given by $|A - F|/A$, where A denotes the actual final number of up-votes, down-votes or comments and F the forecasted values. The best results for each combination of dataset and time-series (including statistical ties) are indicated in bold. We used a Wilcoxon rank-sum test at 5% significance level to compare the medians.

Table 4.4: Comparison of VnC forecasting error. The table shows the median forecasting absolute percentage error (APE). Best results (including statistical ties) are shown in bold.

		Bass	SI	Phoenix-R	Spike-M	VnC
Reddit	v_+	0.57	0.57	0.82	0.42	0.39
	v_-	0.67	0.64	0.86	0.55	0.53
	c	0.62	0.64	0.86	0.73	0.39
Imgur	v_+	0.69	0.65	0.81	0.98	0.71
	v_-	0.65	0.68	0.84	0.98	0.65
	c	0.59	0.58	0.84	1.15	0.47
Digg	v_+	0.87	0.89	0.98	0.52	0.77

VnC provided the most accurate forecast – or tied with the best competitor – for almost all combinations of user interaction and datasets. The exception was the up-vote time-series from the Digg dataset for which Spike-M obtained the lowest error. VnC obtained the most striking results for the comments time-series, where the median APE (error) was considerably lower than all competitors. The volume of comments received by a submission is, on average, lower than the number of votes. As a result, forecasting the comment time-series is a challenging task. VnC solves this problem by using data from the up-vote and down-vote time-series to forecast the comment time-series (see Equation 4.9).

Figure 4.13 shows the VnC forecast results for the up-vote time-series of the most voted submissions from each dataset. We also include, for comparison purposes, the results obtained by Spike-M and Bass models. We omit forecasts from SI and Phoenix-R to avoid occlusion and because the Spike-M and Bass models obtained the best results after VnC. Only the data points to the left of the vertical dashed line (indicated by a solid black line) were used to train the models. Both Spike-M and VnC obtained accurate forecasts while the Bass model did not predict correctly the tail, which is expected, since it yields a curve that decays exponentially while real data has a heavier tail. While Spike-M was able to match VnC forecasting capability, VnC is scalable and Spike-M has a quadratic run-time complexity on the length of the time-series (see Figure 4.10).

Figure 4.13: Forecasting using VnC. We trained the models using only data from the first seven hours after a submission was created. The vertical axis (up-votes) is in log scale. The interval between data points is 20 minutes.

4.6.2 Detecting Outliers

Our goal in this section is to show how VnC can be used to detect submissions that attracted an anomalous volume of user interactions. We divide the outliers that we observed in the Reddit and Imgur datasets into three categories: (i) late night outliers, (ii) flat down-vote outliers and (iii) “one-shot users” outliers.

Late night outliers: Figure 4.14(a) shows a scatter plot that compares the R^2 values obtained by VnC when fitting the up-votes and down-votes time-series of all Reddit and Imgur submissions. As expected from the results of Section 4.5, VnC obtained high R^2 values for most of the submissions. However, submissions inside the dashed blue ellipsis are outliers, as they have low R^2 values for both the up-votes and down-votes time-series and represent less than 2% of the dataset. Figure 4.14(c) shows two of these outliers, which are characterized by having double peaked time-series and are depicted as triangles in the R^2 scatter plot of Figure 4.14(a). For comparison, the normal submissions shown in Figure 4.14(b), as well as the majority of the time-series, have a single peak.

In order to understand what causes the double peak behavior, we analyzed the time of the day of occurrence of the valleys between the two peaks. We find the time-series’ two peaks by computing all local maxima in a 5.5h window and selecting the two local maxima with the largest up-vote count. Our analysis of the 35 submissions with the lowest R^2 values showed that the valleys are concentrated around 3h to 10h in US Eastern Standard Time (EST), coinciding with time interval in which Reddit users are less active. This indicates that the two peak behavior may be caused by submissions that receive a first peak of up-votes by users active at late night and a second peak of votes by users active by the morning.

Flat down-vote outliers: Figure 4.14(s) shows the up-votes and down-votes time-series for submissions that are located in the red ellipsis area in Figure 4.14(a). These submissions represent less than 2% of the dataset. Compared to the normal submissions, the outliers located in the red ellipsis area have a flat down-vote time-series and a regular (single peaked) up-vote time-series. More than half of the flat down-vote outliers are pictures of animals such as dogs and cats, which seems to repel down-votes from Reddit and Imgur users.

“One shot users” outliers: Finally, VnC can also detect anomalous outliers by analyzing the fit accuracy of the votes vs. comments’ curve (Equation 4.9). We selected all submissions with at least 100 comments from the Reddit and Imgur datasets. We computed VnC’s R^2 values for the votes vs. comments fit. Figure 4.15(a) shows the distribution of R^2 values with the outliers indicated by a red ellipsis. Figure 4.15(c) shows the time-series for the votes (up-votes and down-votes) and comments received by the two suspicious submissions with the smallest R^2 values. Both submissions received a large number of comments just after they were created. Notice that this pattern does not occur for regular submissions such as in Figure 4.15(b).

We inspected each suspicious submission in order to find comments made by “one-shot users”. We define “one-shot users” as users who only commented on a single submission. For the left submission in Figure 4.15(c), over 55% of the users who posted comments are suspicious: i.e. they only posted comments on that submission. Similarly, for the right submission of Figure 4.15(c), over 10% of the users who posted comments are suspicious. For comparison, we also inspected

Figure 4.14: Outliers detected by VnC. (a) VnC R^2 values obtained by fitting the up-votes and down-votes time-series of submissions with at least 1,000 up-votes. Two regions of outliers – blue and red ellipsis – are shown. (b) Normal submissions for which VnC provided an accurate fit (over 95% of the dataset). (c) Late night outliers (dashed blue ellipsis, less than 2% of the dataset) have a low R^2 for the up-votes and down-votes time-series. (d) Flat down-vote outliers (solid red ellipsis, less than 2% of the dataset) have a high R^2 for the up-votes time-series but low R^2 for the down-votes time-series. More than half of the flat down-votes outliers are pictures of animals.

Figure 4.15: VnC spotted submissions suspected of using artificial users to boost their popularity. (a) Distribution of R^2 values for the comments vs. up-votes fit. (b) Normal submissions. (c) Suspicious submissions which received a large number of comments just after their creation.

the submissions with the highest and median R^2 values, and we were not able to find a single one-shot user.

Both suspicious submissions were posted to the Reddit’s AMA (ask me anything) section, which focuses on interviews. In the AMA section, a user (the interviewee) makes a submission and other users (commenters) can leave questions (comments) and vote on other questions and answers. The AMA section is often used as a marketing tool and has attracted the participation of celebrities such as Bill Gates, Barack Obama and Arnold Schwarzenegger. As a result, the outliers detected by VnC could be attributed by individuals that recruited users to boost the number of comments – and consequently, the popularity – of their submission.

4.6.3 Clustering

While social voting services such as Imgur are focused only on image data, other social voting services such as Reddit allow users to submit links to different types of content (e.g. pictures, news, movies). A question one could ask is whether the comment vs. votes behavior is the same for distinct types of submissions. To answer this question we make use of the fact that Reddit links are categorized into *sub-reddits*. Sub-reddits are sub-communities focused on different interests. We used VnC to fit the votes vs. comments curve of links submitted to two sub-reddits: *pics* and *AskReddit*. In *pics* the sub-reddit users can only post links to pictures while in *AskReddit* users submit questions asking for other users’ comments. In Figure 4.16 we plot the parameters k and α of VnC Equation 4.9.

Observation 4.4. *Links to pictures and to discussions generate two separated clusters when projected into the parameter space α and k of the VnC model (Figure 4.16).*

Figure 4.16: Clustering submissions using the VnC model parameters. Submissions were selected from two sub-reddits: *pictures* and *AskReddit*.

Figure 4.16 shows that there is a clear difference in the parameter k between links submitted to the sub-reddits *pics* and *AskReddit*, resulting in a separation of the submissions into two clusters. This indicates that the coefficient k changes significantly for different types of submissions (pictures and questions). The difference in the exponent α , however, is not as pronounced. While submissions to the *pics* sub-reddit have a mean $\alpha = 0.68 \pm 0.12$, submissions to *AskReddit* have a mean $\alpha = 0.75 \pm 0.09$.

4.7 Conclusions

In this chapter we proposed Vote-and-Comment (VnC), a model for user activity on social voting services. Given a submission, VnC describes: (i) how the number of votes evolves over time; (ii) how the difference between up- and down-votes evolves over time and (iii) how the number of comments grows with respect to the number of votes. The VnC model was designed based on observations taken from more than 20,000 submissions from three social voting Web sites: Reddit and Imgur – each attracting over 110 million unique visitors per month⁶ – as well as publicly available data from Digg [65].

We compared VnC against representative models for information diffusion from the literature. VnC consistently provided the most accurate fit and forecast to real data for all analyzed datasets. In particular, for the comment time-series VnC performed significantly better than the baselines (Table 4.4). Table 4.5 compares the capabilities of VnC against existing models from the literature. VnC is the only model designed to describe the relationship between the different types of user interactions found in social voting services: up-votes, down-votes and comments. Additionally, VnC is able to fit the heavy tail observed in real data without requiring quadratic evaluation cost as Spike-M does.

The contributions presented in this chapter are summarized as follows:

- **Discoveries:** We discovered the following patterns by analyzing different modalities of user interactions: (i) relationship between peak time and total number of votes (*Urgency Law*),

⁶Source: <https://www.reddit.com/about>, accessed on February 2016.

Table 4.5: Capabilities of VnC compared with existing models from the literature.

	SI	Bass	Ph.-R	Sp.-M	VnC
Votes over time	✓	✓	✓	✓	✓
Up- vs. Down-votes					✓
Votes vs. Comments					✓
Heavy Tail				✓	✓
Linear Evaluation	✓	✓	✓		✓

(ii) relationship between votes and comments (*VC Law*) and (iii) log-logistic distribution of reaction times.

- **Model:** Based on the discovered patterns, we proposed VnC, a parsimonious, accurate and scalable model that describes the relationship between different types of activity on social voting Web sites. VnC consistently obtained superior results with respect to fitting accuracy when compared to competitors. Specifically, VnC provided a better fit than the closest competitor for over 92% of the more than 20,000 submissions studied. Moreover, VnC is also able to explain the relationship between up-votes vs. down-votes and votes vs. comments, while existing models such as Spike-M and SI can only be used to fit the up-votes, down-votes and comment time-series individually.
- **Usefulness:** VnC can be used to forecast the number of votes received by a submission, spot anomalies and cluster different types of content, based only on user activity.

Chapter 5

Automatic Annotation of Social Media Images

Due to their large user base, social media services such as Facebook, Flickr, Instagram and Twitter generate a large volume of images. Due to the volume of the data, providing tools for users to effectively find relevant image content is important to improve the usability of these social media services. Keyword-based queries, in which users search for content using textual keywords, could be used to address the problem of searching for images in such large collections. However, in order to use keyword-based queries with images, they need to contain textual annotations. Since manually annotating such large volumes of images is not always a feasible solution, we tackle in this chapter the problem of automatically annotating images from social media services.

5.1 Introduction

As we discussed in Section 2.4, most of the current automatic image annotation methods employ multiple extraction algorithms to analyze images considering local or global features under different visual features. This is necessary because most of the extraction algorithms are not capable of describing the large visual variability of images. Additionally, in the context of social media, the visual features are often fused with metadata such as textual comments. However, the use of a large number of features also falls in the problem known as the *dimensionality curse* [58], in which the significance and information content of each feature decreases, reducing the annotation accuracy.

Therefore, if we knew beforehand which features are useful to determine the relevance of a given annotation, we would be able to discard irrelevant features, improving the annotation accuracy. However, it is not always clear which features are appropriate to predict the relevance of a given annotation.

To tackle this problem, we propose MFS-Map (Multi-Feature Space Map), a method that combines visual and textual features to annotate social media images. MFS-Map design is based on the following assumption: images with a given textual annotation are more likely to have a lower visual variability. To illustrate this assumption, consider Figure 5.1. It shows pictures with manual annotations from the MIR Flickr dataset [49]. Figure 5.1(a) presents images with the annotation “sky” while Figure 5.1(b) shows images with the annotation “structures”. For the

Figure 5.1: Pictures from the MIR Flickr dataset with annotations (a) “sky” and (b) “structure”. Color information may be correlated to annotation “sky” but will probably be less meaningful to annotation “structure”.

annotation “sky”, we observe the prevalence of the color blue. On the other hand, the images with the annotation “structure” are characterized by the presence of sharp edges.

MFS-Map takes advantage of the lower visual variability among images with the same annotation to automatically identify which visual features are more useful to annotate an image. For example, in Figure 5.1, color information may be appropriate to identify the presence of the annotation “sky” because of the prevalence of blue color. However, for the annotation “structures” there is no prevalent color. Thus, color may not be an adequate feature.

In order to identify useful visual and textual features when annotating images, we divide the features into subspaces. This allows MFS-Map to discretize the visual features from each subspace by using clustering algorithms. Finally, MFS-Map mines association rules between annotations and the discretized subspace features. The rules are then employed to suggest annotations for an input non-annotated image. This is possible because the feature vectors resulting from the extraction algorithms are not concatenated into a single large feature vector. Rather, we divide the features into a number of feature subspaces. This allows MFS-Map to find relationships between annotations and regions of the subspaces. The useful relationships are automatically selected and represented as rules that are employed to predict annotations for an input non-annotated image.

We evaluated MFS-Map by employing two publicly available datasets: MIR Flickr and Image CLEF 2011. We compared our results with widely employed annotation approaches, and MFS-Map almost always obtained both significantly superior precision and faster training and testing times.

5.2 Problem Definition

The data that we analyze in this chapter consists of images from the social media service Flickr¹. Images from Flickr may contain one or more textual tags assigned by users. These tags may be missing and are not necessarily related to the visual content of the image.

The problem that we want to solve is the following: given an image uploaded to Flickr – which may contain textual tags – find a set of textual annotations that describe the content of that image. Formally, this problem can be stated as follows:

Problem 5.1 (Automatic Image Annotation). *Given a social media image $\mathcal{I}$, consisting of a tuple (I, L) , where I corresponds to the image (i.e. a matrix of pixels) and a set of textual tags $L = \{\ell_1, \ell_2, \dots\}$, we want to find a set $A = \{a_1, a_2, \dots\}$ of textual image annotations to describe the input social media image.*

Here we want to emphasize the difference between textual annotations and tags. Tags are provided by Flickr users and used alongside with the image content as input to solve the problem of automatic image annotation. These tags are potentially noisy (e.g. wrong, missing or unrelated to the image visual content). On the other hand, the annotations should describe the visual content of the images and are the output that we aim to find when solving the problem.

5.3 Proposed Method: MFS-Map

MFS-Map (Multi-Feature Space Map) is an automatic image annotation method that, given a non-annotated image, returns a list of the annotations that best describes its visual content. Each returned annotation is scored based on its predicted relevance.

In order to predict the annotation relevance, MFS-Map generates rules that describe relationships between annotations and regions in the visual feature subspaces. The rules are generated from a training set of images where each image may contain one or more annotations. The rule generation is carried out by the following steps:

1. **Feature Extraction:** Extract visual features from a training set of images employing a set of extraction algorithms. Each extraction algorithm yields a feature space where the images are represented by feature vectors;
2. **Discretization:** Convert the sets of feature vectors that represent the training set images into sets of *feature items*. Each feature item provides a discrete representation of a feature vector.
3. **Rule Generation:** Combine the images feature items with their textual tags and annotations and generate rules in the format: $\{\text{feature item}\} \rightarrow \{\text{annotation}\}$ or $\{\text{textual tag}\} \rightarrow \{\text{annotation}\}$.

The MFS-Map is described as follows. Section 5.3.1 details how MFS-Map generates feature items from visual features. Section 5.3.2 presents the algorithm employed to obtain rules from the discrete representation of the image. Finally, Section 5.3.3 describes MFS-Map’s annotation algorithm.

¹<https://www.flickr.com/>

5.3.1 Visual Feature Discretization

Before it generates the annotation rules, MFS-Map discretizes the visual features of the training set images. To discretize these visual features, MFS-Map learns a map function from a training set of images. Specifically, MFS-Map solves the following problem:

Problem 5.2 (Visual Feature Discretization). *Let $\mathbb{Z}$ denote the set of all integers. Given a training set of images $\mathcal{X} = \{I_1, I_2, \dots, I_n\}$ and the image annotations for each training image $\{A_1, A_2, \dots, A_n\}$, we want to learn a map function $m(I) \rightarrow \{f_1, f_2, \dots, f_k\}$ that maps the input image I into a set of discrete feature items $f_i \in \mathbb{Z}$.*

Algorithm 3 describes how MFS-Map learns the map function. The algorithm takes as input the set of training images $\mathcal{X} = \{I_1, I_2, \dots\}$ and a set of feature extraction functions $\{\varepsilon_1, \varepsilon_2, \dots, \varepsilon_k\}$. As discussed in Section 2.4, each feature extraction function ε_i takes as input an image I and returns a feature vector $\mathbf{v}$.

Algorithm 3 Algorithm to learn MFS-Map’s mapping function.

Input Set of training images $\mathcal{X} = \{I_1, I_2, \dots, I_n\}$, annotations of the training images $\mathbf{A} = \{A_1, A_2, \dots, A_n\}$, and a set of feature extraction functions $\{\varepsilon_1, \varepsilon_2, \dots, \varepsilon_k\}$.

Output Map function $m(I)$.

```

for  $\varepsilon_i \in \{\varepsilon_1, \varepsilon_2, \dots, \varepsilon_k\}$  do
     $\mathbf{V}_i \leftarrow \emptyset$ 
    for  $I_j \in \mathcal{X}$  do
         $\mathbf{V}_i \leftarrow \mathbf{V}_i \cup \{\varepsilon_i(I_j)\}$ 
    end for
     $\mathbf{C}_i \leftarrow \text{K-MEANS}(\mathbf{V}_i, \mathbf{A})$ 
end for
 $m(I) \leftarrow f(I) = \{\text{NCL}(\varepsilon_1(I), \mathbf{C}_1), \dots, \text{NCL}(\varepsilon_k(I), \mathbf{C}_k)\}$ 

```

For each extraction function ε_i , Algorithm 3 creates a set of feature vectors $\mathbf{V}_i$ by applying ε_i to all images in the training set (lines 2 to 5). In line 6, the procedure K-MEANS applies a k-means clustering on the feature vectors in $\mathbf{V}_i$. When running the k-means algorithm, we set the number of clusters to correspond to the number of different annotation values present in the training set. The output of the k-means clustering is a set of centroids. For all centroids, we store in $\mathbf{C}_i$ the coordinates and a centroid label. The centroid label is an integer and we implement the procedure K-MEANS to assign distinct labels to all centroids. Finally, in line 8, we assign a mapping function to $m(I)$. This map function assigns a feature item to each feature vector $\mathbf{v}_i$ representing the image I . For this purpose, we use the nearest centroid label procedure, denoted by $\text{NCL}(\mathbf{v}_i, \mathbf{C}_i)$, which searches for the centroid in $\mathbf{C}_i$ that is nearest to the feature vector $\mathbf{v}_i$ and returns its label.

Figure 5.2 illustrates how MFS-Map assigns centroids labels. In this example, there are two feature extraction functions: A and B. The part (a) of Figure 5.2 shows how MFS-Map extract feature vector representation of images. In this example, the feature spaces are represented as three-dimensional spaces, and each feature vector is represented by a point. However, in the general case there is no restriction to the feature spaces dimensionality and they do not need to share the number of dimensions among themselves.

Figure 5.2: MFS-Map maps images to a set of discrete feature items. (a) Feature vector representations are extracted from the images. (b) Resulting centroids obtained by applying k -means in feature subspaces A and B separately. The resulting set of feature items for an image is obtained by finding the label of nearest centroid in each feature space.

Figure 5.2(b) shows the resulting centroids obtained by applying k -means in feature space A and B separately. For the feature space A, two centroids were obtained (c_1 , and c_2) and for the feature space B three centroids were obtained (c_3 , c_4 and c_5). If we used the map function on the image represented by a black circle in Figure 5.2(b), the resulting set of feature items would be $\{c_1, c_5\}$.

5.3.2 Rule Generation

In the previous section we described how MFS-Map learns a function $m(I)$ that given an image I returns a set $\{f_1, f_2, \dots\}$ of feature items to describe it. In the rule generation phase, MFS-Map takes as input the *itemset representations* of the training images and finds a set of rules that are employed to predict annotation relevance. The itemset representation of an image is defined as follows:

Definition 5.1 (Itemset Representation). *The itemset representation of an image I is the set formed by the union of the sets of feature items $\{f_1, f_2, \dots\}$, textual tags $L = \{\ell_1, \ell_2, \dots\}$ and annotations $A = \{a_1, a_2, \dots\}$ of I .*

Before presenting the rule generation algorithm, we also define the confidence $\text{conf}(f_i, a_j)$ of a feature item f_i and annotation a_j pair:

Definition 5.2. *The confidence $\text{conf}(f_i, a_j)$ of a feature item f_i and annotation a_j pair is given*

by:

$$\text{conf}(f_i, a_j) = \text{freq}(\{f_i, a_j\}) / \text{freq}(\{f_i\}), \quad (5.1)$$

where $\text{freq}(\{f_i, a_j\})$ is the number of times that the pair feature item f_i and annotation a_j occurs in the same itemset and $\text{freq}(\{f_i\})$ is the number of times f_i appears in an itemset.

The rule generation algorithm calculates a confidence value for each pair $\{f_i, a_j\}$, $f_i \in \mathcal{F}$ and $a_j \in \mathcal{A}$ where $\mathcal{F}$ and $\mathcal{A}$ denote, respectively, all possible values of feature items and annotations in the training set. MFS-Map uses the confidence value of a pair (f_i, a_j) as an estimate of the usefulness of the feature item f_i in predicting the relevance of annotation a_j . The rules are generated from all pairs $\{f_i, a_j\}$ whose confidence value is higher than a minimum confidence threshold, denoted by minConf . The format of a rule is $f_i \rightarrow a_j$, where f_i is defined as the antecedent and a_j is defined as the consequent of the rule.

5.3.3 Prediction of Annotation Relevance

The annotation phase of MFS-Map takes as input the set of mined rules and predicts annotations for an input non-annotated image I' and its – possibly empty – set of textual tags $L' = \{\ell_1, \ell_2, \dots\}$. MFS-Map starts by applying the map function to the image I' to obtain its itemset representation.

The next step of the annotation phase consists in selecting all mined rules whose antecedent contains an item that is also in the itemset of the input image. If an annotation a_i appears in at least one of the selected rules, then its relevance score is given by the mean confidence of all selected rules whose consequent contains a_i . Otherwise, if an annotation does not appear in any rule, then its relevance with respect to the input image is zero. For example, if the itemset representation of the input image was $\{f_2, f_4\}$ and the set of mined rules were:

1. $\{f_1\} \rightarrow \{\text{'animal'}\} (\text{conf} = 0.67)$
2. $\{f_4\} \rightarrow \{\text{'sunset'}\} (\text{conf} = 0.82)$
3. $\{f_2\} \rightarrow \{\text{'city'}\} (\text{conf} = 0.72)$
4. $\{f_2\} \rightarrow \{\text{'sunset'}\} (\text{conf} = 0.93)$

then the rules 2, 3 and 4 would be selected because their antecedent contains either the feature item f_2 or f_4 . Rule 1, however, would not be selected because feature item f_1 is not present in the itemset representation of the input image.

The relevance score of the annotation ‘sunset’, considering the selected rules, is the average of the confidence of the rules that contains the annotation ‘sunset’ in its consequent, that is, rules 2 and 4 which results in a relevance score of $(0.82 + 0.93)/2 = 0.875$. For annotations that do not appear in the consequent of any selected rules, such as ‘animal’ in the example, the relevance score is zero.

5.4 Experiments

We evaluated MFS-Map for the task of image annotations using two publicly available datasets: *MIR Flickr* [49] and *Image CLEF 2011*. Both datasets were employed by many works in automatic

image annotation [100, 114, 102] and are composed of images downloaded from the social media service Flickr. The MIR Flickr and Image CLEF 2011 datasets contain, respectively, 25,000 and 18,000 images.

Each image in both datasets is manually annotated. The number of different possible annotations for MIR Flickr is 25 and for Image CLEF is 98. The annotations correspond to depicted objects (e.g. “cars”, “dog”, “flower”) and scene description (e.g. “sunset”, “indoor”, “night”). Additionally, each image has also tags assigned by Flickr users. As we have discussed in Section 5.2, when we formalized the problem of image annotation, the tags are a valuable resource, but they contain noise, since not all tags are relevant to the image visual content. Because of the noise in the tags, performance evaluation is based on manual annotations. In our experiments, we discarded tags whose minimum frequency in the dataset was lower than 25. Figure 5.3 shows a sample image from the MIR Flickr dataset, its user tags and manual annotations.

user tags: crane, gru, sunset,
hdr, tramonto, cielo, sky, ray,
raggi, light, luci, chdk

manual annotations: clouds,
sky, structures, sunset

Figure 5.3: Sample image from the MIR Flickr dataset, its user tags and manual annotations. Tags are potentially wrong or unrelated to the image visual content while annotations have a higher quality, since they were validated by the MIR Flickr dataset creators.

5.4.1 Feature extraction

The visual feature extractors employed in the experiments were the following:

RGB and HSV histograms: Histograms computed by re-quantizing each color channel of the RGB and HSV color spaces to 7 bins yielding two $7^3 = 343$ dimensional feature vectors.

HSV Histogram with layout information: Feature vector computed by dividing the input image into 3 horizontal stripes of the same height and computing a local HSV Histogram. The local HSV histogram is computed by re-quantizing each color channel to 5 bins yielding a $3 \times 5^3 = 375$ dimensional feature vector.

SIFT: 100 bin bag-of-visual-words histogram computed by extracting local SIFT features [70] using a dense multi-scale grid for sampling.

Gist: 512 dimensional feature vector computed using the Gist descriptor [77] by resizing the image size to 256×256 pixels and using 8 orientations per scale.

SFTA: 21 dimensional feature vector computed by applying the SFTA texture descriptor [26].

Gist and SFTA feature vectors were normalized to the range 0.0 to 1.0. The remaining feature vectors were L_1 normalized.

5.4.2 Annotation methods

We compared MFS-Map performance to the three following methods: (i) cross-training using late-fusion (**LF-SVM**); (ii) cross-training using early-fusion (**EF-SVM**), and (iii) a nearest neighbor model that annotates images with the annotations of the most similar image from the training set (**1-NN**). These methods were discussed in Section 2.4.

For MFS-Map we set the confidence parameter to 0.5, which we found out to provide the best average performance. For cross-training, we employed SVM classifiers with radial basis function (RBF) kernels. In order to find the best SVM parameters (SVM cost and RBF kernel degree) we employed cross-validation before each binary SVM was trained. An important aspect of cross-training is that the classifiers must provide class membership probabilities for each prediction. Since SVMs do not output class membership probabilities, we mapped SVM scores to probabilities by learning a regression model using 20% of the training data.

For methods that rely on distance computation, we employed the cosine distance for the SIFT and Flickr tags when measuring dissimilarity. For all other features we used the Euclidean distance. All annotation methods were implemented in C++. For cross-training annotation (LF-SVM and EF-SVM) we employed LibSVM [22].

To quantify the performance of the annotation methods, we employed two metrics: the average precision (AP) and the break-even point precision (BEP). Both AP and BEP are computed for each annotation but can be averaged to provide a single measurement. To compute AP and BEP for a given annotation a , we rank all images by the predicted relevance (the returned annotation score) and compute the precision at each position n . The precision at position n corresponds to the fraction of images ranked lower or equal to n that have been correctly annotated. AP averages the precision over all positions of correctly annotated images and BEP is the precision at position n' , where n' corresponds to the number of images manually annotated with annotation a . AP and BEP results were obtained by ten-fold cross-validation with ten repetitions.

5.4.3 Annotation precision

In this section we describe the experiments performed to evaluate MFS-Map precision for the task of annotating images. Table 5.1 shows the mean AP and BEP of each annotation method for different feature configurations: T, V and T+V. Column T shows the results obtained using only Flickr tags. Column V shows the results obtained using only visual features. Column V+T shows the results obtained using both Flickr tags and visual features. Standard deviation is indicated between parentheses.

We compared AP and BEP values for each combination of features configuration and dataset using a two-tailed t-Student test with $p=0.05$ and the best results are indicated in bold. For both MIR Flickr and Image CLEF, MFS-Map obtained the best results for the three configurations (T, V and T+V) with one exception. 1-NN obtained the best BEP for Image CLEF using visual features.

The combination of visual and textual features (V+T) improved MFS-Map precision by a small but statistically significant value with $p=0.03$. However, for the other methods, the V+T precision was equal or inferior to the values obtained with only visual or only textual features.

Table 5.1: Average precision (AP) and break even precision (BEP) obtained by each annotation method for the MIR-Flickr and Image CLEF 2011 datasets under different feature configurations. T corresponds to results obtained using only Flickr tags. V corresponds to results obtained using only visual features. V+T corresponds to results using both Flickr tags and visual features. Standard deviation is indicated between parentheses.

Dataset	Method	T		V		V+T	
		AP (%)	BEP (%)	AP (%)	BEP (%)	AP (%)	BEP (%)
MIR Flickr	MFS-Map	54.0 (0.69)	47.3 (0.66)	41.1 (0.42)	39.3 (0.42)	55.1 (0.66)	52.0 (0.37)
	EF-SVM	29.9 (0.26)	26.9 (0.35)	28.5 (0.89)	25.0 (1.28)	29.9 (0.26)	27.0 (0.44)
	LF-SVM	29.9 (0.27)	26.9 (0.35)	29.9 (0.73)	27.0 (0.86)	29.9 (0.31)	26.9 (0.47)
	1-NN	30.8 (2.92)	28.3 (4.36)	35.1 (0.43)	37.7 (0.53)	30.9 (3.19)	28.5 (4.80)
Img. CLEF	MFS-Map	48.1 (0.27)	45.5 (0.35)	42.9 (0.33)	40.2 (0.34)	49.4 (0.58)	47.6 (0.59)
	EF-SVM	37.7 (0.22)	37.2 (0.22)	36.8 (0.59)	35.7 (0.73)	37.8 (0.27)	37.2 (0.32)
	LF-SVM	37.8 (0.21)	37.2 (0.22)	37.7 (0.18)	37.2 (0.19)	37.8 (0.33)	37.2 (0.31)
	1-NN	38.2 (1.36)	37.9 (2.10)	40.2 (0.29)	41.6 (0.53)	38.1 (0.98)	37.7 (1.59)

Figure 5.4: Average precision (AP) versus training and annotation time for each annotation method. Our proposed method, MFS-Map, is indicated by the square symbol.

This indicates that MFS-Map performs well also when different feature modalities (visual and textual) are combined.

5.4.4 Training and annotation time

Figure 5.4 shows a plot of training plus annotation time (total time) versus average precision for each method. Times were obtained by using a random sample of 20% of the MIR Flickr dataset and 40% of the Image CLEF dataset. The experiments were executed in a computer with an Intel i7 2.66GHz processor, 8GB RAM running Windows 64-bit OS. AP values may differ from Section 5.4.3 because of the sampling process. For the two datasets, MFS-Map was the fastest method. Additionally, as shown in Section 5.4.3, MFS-Map also obtained the highest average precision.

1-NN does not have a training phase and the total time corresponds to its annotation (test)

time only. However, for all other methods, the training phase accounted for at least 95% of total time. MFS-Map training time was at least 48 times faster for the Image CLEF dataset and 3.6 times faster for MIR Flickr dataset when compared to the other cross-training approaches (EF-SVM and LF-SVM). EF-SVM and LF-SVM larger training time can be attributed to the need to train a separated classifier for each annotation. Thus, when the number of different annotations is higher – Image CLEF has 98 annotations while MIR Flickr has 25 – the cross-training time is also higher.

5.5 Conclusions

In this chapter we presented MFS-Map, a method that automatically annotates images from social media services. MFS-Map efficiently combines the images' visual content (represented by feature vectors) with textual tags when providing annotations. Instead of concatenating the feature vectors and tags into a large feature vector, MFS-Map divides the problem of automatic image annotation in subspaces, handling each visual feature separately. This allows MFS-Map's rule generation algorithm to discard rules that present weak relationship between features and annotations. This is important in annotation problems, since a particular feature may be useful to annotate an image but may introduce noise for other images.

Chapter 6

Conclusions

The widespread use of social media services has produced an unprecedented volume of data on human behavior. Knowledge extracted from this data could be used in many potential applications, ranging from advertising to spam detection. However, this data is often complex and diverse. As a result, many traditional machine learning techniques are not able to directly handle the data, often requiring a prior feature extraction step.

In this PhD work we focused on developing methods for modeling and mining social media data. Specifically, we presented three contributions. First, we proposed Act-M, a model for the timing of users' communication in social media that can be also used to detect bots. Our second contribution is VnC, a mathematical model that describes the volume of user interactions received by content submitted to social media services. Finally, as our third contribution, we presented MFS-Map, a method that automatically suggests textual annotation to social media images.

The remainder of this chapter is organized as follows. In Section 6.1 we summarize the main contributions of this thesis. Section 6.2 discusses future lines of research. Finally, in Section 6.3, we list the publications that resulted from this PhD work.

6.1 Contributions of this PhD Work

The main contributions of this PhD dissertation were result of three research problems that we investigated in order to support our thesis. The first research problem consisted in analyzing how to extract useful knowledge from data describing the timing of social media users' actions. We also worked on how to provide textual annotations to social media images in order to support keyword-based queries. Finally, we studied the problem of modeling how the volume of users' interactions with a piece of content changes over time.

The main contributions that we describe in the following sections were evaluated using real data from social media services. Since most of the data was not readily available, it required the development of crawlers to collect and store them into local databases. For example, in Chapter 4 we analyzed time-series of users interactions from Reddit and Imgur. The APIs from these two services only provided a snapshot of the current number of interactions (votes and comments) for a given submission. In order to create time-series of the volume of user interactions, we tracked submissions, collecting a snapshot of the data every 20 minutes. As a result, in addition to

the intellectual contributions, this PhD work also produced the following open-sourced code and datasets:

- **Open-Source Code:**

- We open-sourced a Matlab implementation¹ of the Act-M model described in Chapter 3. The implementation also includes code for detecting bots based only on time-stamp data.
- We open-sourced a Matlab implementation² of the VnC model that we presented in Chapter 4.

- **Datasets:**

- We released a dataset³ of up-votes, down-votes and comments time-series of Reddit and Imgur submissions. This data was used to design and validate the VnC.
- We released a dataset⁴ with sequences of postings' time-stamps from Reddit and Twitter users with over 36 Million time-stamps and 100 thousand users.

6.1.1 The Act-M Model for Communication Dynamics

The Act-M, described in Chapter 3, is a mathematical model that fits the distribution of inter-arrival times (IAT) from social media users' postings. We took advantage of time-stamp data from several social media services to analyze user activity. Our contributions are:

- The IAT between postings is characterized by four activity patterns: (i) the distribution of IAT is heavy-tailed; (ii) the distribution of IAT has two modes; (iii) the circadian rhythm affects the users' postings times, generating periodic peaks in the IAT distribution every 24 hours; (iv) IAT is not an independent and identically distributed (i.i.d.) variable. Although part of the temporal patterns we have observed agrees with previous studies in human communication, to the best of our knowledge, the combination of all four discovered patterns in several social media services is a novel finding.
- The Act-M accurately fits the distribution of IAT from social media users' postings. More specifically, Act-M was able to match all the four activity patterns that we discovered. Additionally, we compared Act-M against state-of-the-art models for human activity. Our experiments show that Act-M provided the most accurate fits;
- We used Act-M to build a bot detection method that relies only on the timing of social media postings. This was possible because the timing of postings made by humans are accurately modeled by Act-M. As a result, we found that social media users whose time-stamps deviate from Act-M are likely to be bots.

6.1.2 The VnC Model for the Coevolution of User Interactions

In Chapter 4 we analyzed data from social voting services in which users can up-vote, down-vote or comment on submitted content. Based on this data, our contributions summarized as follows:

¹https://github.com/alceufc/rsc_model

²https://github.com/alceufc/vnc_model

³<https://osf.io/bc3k6/>

⁴https://github.com/alceufc/rsc_model#datasets

- We discovered the following patterns by analyzing different modalities of user interactions: (i) the number of comments grows as a power-law on the number of votes and (ii) the time between a submission creation and a user’s reaction obeys a log-logistic distribution.
- We presented VnC (Vote-and-Comment), which is a model for user activity on social voting services. Given a submission such as an image or a link to a news article, VnC describes how the number of up-votes and down-votes evolves over time and how the number of comments grows with respect to the number of votes. VnC requires only a small number of interpretable numerical parameters to accurately describe the time-series from social voting services. In order to validate VnC, we have tracked more than 20,000 submissions from three social voting services: Reddit, Imgur and Digg. We compared VnC against representative models for information diffusion from the literature. VnC consistently provided the most accurate fit and forecast to real data for all analyzed datasets. For the comment time-series, in particular, VnC performed significantly better than the existing models.

6.1.3 The MFS-Map Method for Image Annotation

MFS-Map is a method for automatic image annotation (AIA). The goal of AIA is to automatically suggest textual annotation to images based on their visual content. These textual annotations are then used mostly to allow users to search images using keyword-based queries. Social media images often have associated context data such as textual information. MFS-Map takes advantage of this fact by efficiently combining the images’ visual and context features to improve annotation precision. To achieve this goal, MFS-Map starts by dividing the visual and textual features into feature subspaces. Finally, MFS-Map uses an algorithm to mine decision rules that associate regions of these subspaces (represented by clusters’ centroids) with annotations.

6.2 Future Work

This PhD dissertation resulted in the development of two models for the behavior of social media users: Act-M and VnC. Additionally, we proposed MFS-Map, a method that automatically suggests annotations to social media images. By building on these results, the following areas of research could be explored in future work:

1. Extending the Act-M and VnC models for the task of modeling human behavior outside the domain of social media. For this purpose, different sources of data – such as online stores, and crowdsourced review services – could be explored.
2. Designing anomaly detection methods based on the output of user behavior models. The designed methods could be applied to the problem of detecting undesired behavior such as spam, fraud and false news in social media services.
3. Designing scalable data mining techniques to efficiently analyze the models’ parameters when designing anomaly detection methods. For this purpose, metric access methods could be employed to index the models’ parameters.
4. Improving MFS-Map results by incorporating deep neural networks into the process of combining textual visual content when annotating images from social media services.

6.3 List of Produced Publications

The Act-M model was first described in [28], a full paper published and presented at the ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2015, which is one of the leading conferences in data mining. An extended version of the Act-M conference paper is already *accepted* for publication as a regular paper at the ACM Knowledge Discovery from Data (TKDD) journal.

A paper [35] describing the VnC model was published at the IEEE International Conference on Data Mining (ICDM) 2016, which is also a top quality conference in the data mining and knowledge discovery research area. An extended journal version of this paper is currently being written. Finally, the MFS-Map method for image annotation was published in the ACM SAC 2014 conference [27].

In addition to the mentioned main contributions, the PhD candidate also collaborated with his colleagues at the Databases and Images Group⁵. As result of these collaborations, a full paper was accepted for the SIAM International Conference on Data Mining that will be held in April 2017. This paper describes a method named VolTime that spots individuals with suspicious behavior based on the volume and timing of their activity. This PhD also resulted in works on the task of detecting fire [14] and smoke [20] in social media images.

⁵GBdI (<http://gbdi.icmc.usp.br/>)

Bibliography

- [1] Panagiotis Adamopoulos and Vilma Todri. The Effectiveness of Marketing Strategies in Social Media : Evidence from Promotional Events Categories and Subject Descriptors. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1641–1650, Sydney, Australia, 2015.
- [2] Charu C. Aggarwal. *Outlier Analysis*. Springer, New York, NY, USA, 2013.
- [3] R. Agrawal, T. Imielinki, and A. Swami. Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, pages 207–216, 1993.
- [4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *International Conference on Very Large Data Bases (VLDB)*, pages 487–499, Santiago de Chile, Chile, 1994. Morgan Kaufmann.
- [5] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.
- [6] Saleem Alhabash, Juan Mundel, and Syed Ali Hussain. Social media advertising: Unraveling the mystery box. In Shelly Rodgers and Esther Thorson, editors, *Digital Advertising: Theory and Research*, pages 285–299. Routledge, 2017.
- [7] Rodrigo A. S. Alves, Renato Assunção, and Pedro O. S. Vaz de Melo. Burstiness Scale: a highly parsimonious model for characterizing random series of events. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1–10. ACM, 2016.
- [8] M.L. Antonie and O.R. Zaïane. An associative classifier based on positive and negative rules. In *SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 64–69, Paris, France, 2004. ACM.
- [9] Monya Baker. A network boost. *Nature*, 518:263–265, 2015.
- [10] Albert-László Barabási. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039):207–211, 2005.
- [11] E. Baralis and P. Garza. A lazy approach to pruning classification rules. In *International Conference on Data Mining (ICDM)*, pages 35–42, IEEE Computer Society, 2002.
- [12] Frank M. Bass. A new product growth for model consumer durables. *Management Science*, 15(5):215–227, 1969.

- [13] Marco T. Bastos, Dan Mercea, and Arthur Charpentier. Tents, tweets, and events: The interplay between ongoing protests and social media. *Journal of Communication*, 65(2):320–350, 2015.
- [14] Marcos Vinicius Naves Bedo, Gustavo Blanco, Willian Dener De Oliveira, Mirela Teixeira Cazzolato, Alceu Ferraz Costa, José Fernando Rodrigues Jr., Agma Juci Machado Traina, and Caetano Traina Jr. Techniques for Effective and Efficient Fire Detection from Social Media Images. In *International Conference on Enterprise Information Systems (ICEIS)*, pages 34–46, Barcelona, Spain, 2015.
- [15] O. Boiman, E. Shechtman, and M. Irani. In defense of Nearest-Neighbor based image classification. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, jun 2008.
- [16] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.
- [17] Adrian W. Bowman and Adelchi Azzalini. *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations*. Oxford University Press, 2004.
- [18] Marco Bressan, Stefano Leucci, Alessandro Panconesi, Prabhakar Raghavan, and Erisa Terolli. The Limits of Popularity-Based Recommendations, and the Role of Social Ties. In *International Conference on Knowledge Discovery and Data Mining (ACM KDD)*, pages 745–754, San Francisco, CA, USA, 2016.
- [19] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying Density-Based Local Outliers. In *International Conference on Management of Data (SIGMOD)*, pages 93–104, Dallas, TX, USA, 2000.
- [20] Mirela T. Cazzolato, Marcos Vinicius Naves Bedo, Alceu Ferraz Costa, Jessica Andressa de Souza, Caetano Traina Jr., and Agma Juci Machado Traina. Unveiling smoke in social images with the SmokeBlock approach. In *ACM SAC*, pages 49–54, Pisa, Italy, 2016. ACM.
- [21] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: a survey. *ACM Computing Surveys*, 43(3):1–72, 2009.
- [22] C. Chang and C. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1—27:27, 2011.
- [23] Junghoo Cho and Hector Garcia-Molina. Estimating frequency of change. *ACM Transactions on Internet Technology*, 3(3):256–290, 2003.
- [24] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Who is Tweeting on Twitter: Human, Bot, or Cyborg? In *Annual Computer Security Applications Conference (AC-SAC)*, pages 21–30, New York, NY, USA, 2010. ACM.
- [25] S. Clinchant, J. Ah-Pine, and G. Csurka. Semantic combination of textual and visual information in multimedia retrieval. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, pages 44:1–44:8, Trento, Italy, 2011. ACM.

-
- [26] Alceu Ferraz Costa, Gabriel Humpire-Mamani, and Agma Juci Machado Traina. An Efficient Algorithm for Fractal Analysis of Textures. In *Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 39–46, Ouro Preto, MG, Brazil, 2012.
 - [27] Alceu Ferraz Costa, Agma Juci Machado Traina, and Caetano Traina Jr. MFS-Map: efficient context and content combination to annotate images. In *ACM SAC*, pages 945–950, Gyeongju, Korea, 2014. ACM.
 - [28] Alceu Ferraz Costa, Yuto Yamaguchi, Agma Juci Machado Traina, Caetano Traina Jr., and Christos Faloutsos. RSC: Mining and Modeling Temporal Activity in Social Media. In *International Conference on Knowledge Discovery and Data Mining*, pages 269–278, Sydney, Australia, 2015. ACM.
 - [29] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences (PNAS)*, 105(41):15649–15653, 2008.
 - [30] Qi Ding, Natallia Katenka, Paul Barford, Eric Kolaczyk, and Mark Crovella. Intrusion as (Anti)social Communication: Characterization and Detection. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 886–894, Beijing, China, 2012.
 - [31] Chris Drummond and Robert C. Holte. Cost curves: An improved method for visualizing classifier performance. *Machine Learning*, 65:95–130, 2006.
 - [32] Jean-Pierre Eckmann, Elisha Moses, and Danilo Sergi. Entropy of dialogues creates coherent structures in e-mail traffic. *Proceedings of the National Academy of Sciences (PNAS)*, 101(7):14333–14337, 2004.
 - [33] Usama Fayyad, G Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, pages 37–54, 1996.
 - [34] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini. The rise of social bots. *Communications of the ACM*, 59(7), 2016.
 - [35] Alceu Ferraz Costa, Agma Juci Machado Traina, Caetano Traina Jr., and Christos Faloutsos. Vote-and-Comment: Modeling the Coevolution of User Interactions in Social Voting Web Sites. In *IEEE International Conference on Data Mining (IEEE ICDM)*, pages 91–100, Barcelona, Spain, 2016. IEEE.
 - [36] F. Figueiredo, J. M Almeida, Y. Matsubara, B. Ribeiro, and C. Faloutsos. Revisit Behavior in Social Media: The Phoenix-R Model and Discoveries. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, pages 386–401, 2014.
 - [37] Peter R. Fisk. The graduation of income distributions. *Econometrica*, 29(2):171–185, 1961.
 - [38] E. Gilbert. Widespread underprovision on Reddit. In *Computer supported cooperative work (CSCW)*, pages 803–808, 2013.

- [39] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.
- [40] S. Golder, D. Wilkinson, and B. Huberman. Rhythms of Social Interaction: Messaging within a Massive Online Network. In *ICCT*, pages 41–66, 2007.
- [41] Shannon Greenwood, Andrew Perrin, and Maeve Duggan. Social Media Update 2016. *Pew Research Center*, (November), 2016.
- [42] Stephan Günnemann, Nikou Günnemann, and Christos Faloutsos. Detecting Anomalies in Dynamic Rating Data: a Robust Probabilistic Model for Rating Evolution. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 841–850, New York, NY, USA, 2014. ACM Press.
- [43] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *ACM SIGMOD Record*, 29(2):1–12, 2000.
- [44] Y. Han. CPAR: Classification based on predictive association rules. In *SIAM Conference on Data Mining (SDM)*, pages 331–335, 2003.
- [45] H. He and E. A. Garcia. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 21(9):1263–1284, 2009.
- [46] César A. Hidalgo R. Conditions for the emergence of scaling in the inter-event time of uncorrelated and seasonal systems. *Physica A*, 369(2):877–883, sep 2006.
- [47] Paul Gerhard Hoel, Sidney C. Port, and Charles J. Stone. *Introduction to Stochastic Processes*. Waveland Pr. Inc., 1986.
- [48] Bryan Hooi, Neil Shah, Alex Beutel, Stephan Gunneman, Leman Akoglu, Mohit Kumar, Disha Makhija, and Christos Faloutsos. BIRDNEST: Bayesian Inference for Ratings-Fraud Detection. In *SIAM Conference on Data Mining (SDM)*, pages 495–503, Miami, Florida, USA, 2015.
- [49] M. J. Huiskes and M. S. Lew. The MIR Flickr Retrieval Evaluation. In *International Conference on Multimedia Information Retrieval (ICMR)*, pages 39–43, Vancouver, Canada, 2008. ACM.
- [50] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 207–216. ACM Press, 2006.
- [51] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang. CatchSync: Catching Synchronized Behavior in Large Directed Graphs Categories and Subject Descriptors. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.
- [52] Da-Cheng Juan, Lei Li, Huan-Kai Peng, Diana Marculescu, and Christos Faloutsos. Beyond Poisson: Modeling Inter-Arrival Time of Requests in a Datacenter. In *PAKDD*, volume 1, pages 198–209, 2014.

- [53] Andreas Kaltenbrunner, Andreas Kaltenbrunner, Vicenç Gómez, Vicenç Gómez, Vicente López, and Vicente López. Description and Prediction of Slashdot Activity. In *Latin American Web Conference (LA-WEB)*, pages 57–66. IEEE, oct 2007.
- [54] Andreas M. Kaplan and Michael Haenlein. Users of the world, unite! The challenges and opportunities of Social Media. *Business Horizons*, 53(1):59–68, 2010.
- [55] Márton Karsai, Kimmo Kaski, Albert-László Barabási, and János Kertész. Universal features of correlated bursty behaviour. *Scientific Reports*, 2:1–7, 2012.
- [56] Simon Kemp. Digital in 2017: Global Overview, In *We Are Social*, 2017.
- [57] Jon Kleinberg. Bursty and Hierarchical Structure in Streams. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 373–397, Edmonton, Alberta, Canada, 2003. ACM Press.
- [58] H. P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *Transactions on Knowledge Discovery from Data (TKDD)*, 3(1):1–58, 2009.
- [59] Narayanan C. Krishnan and Diane J. Cook. Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, 10:138–154, 2014.
- [60] J. Kunegis, A. Lommatzsch, and C. Bauckhage. The Slashdot Zoo: Mining a Social Network with Negative Edges. In *International Conference on World Wide Web (WWW)*, pages 741–750, 2009.
- [61] H. Lakkaraju, J. McAuley, and J Leskovec. What’s in a name? Understanding the interplay between titles, content, and communities in social media. In *AAAI Conference on Weblogs and Social Media*, pages 1–10, 2013.
- [62] T. Lappas, E. Terzi, D. Gunopulos, and H. Mannila. Finding effectors in social networks. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1059–1068, 2010.
- [63] Theodoros Lappas, Marcos R. Vieira, Dimitrios Gunopulos, and Vassilis J. Tsotras. On the spatiotemporal burstiness of terms. In *Very Large Data Bases (VLDB)*, pages 836–847, Istanbul, Turkey, 2012. VLDB Endowment.
- [64] J. G. Lee, S. Moon, and K. Salamatian. An Approach to Model and Predict the Popularity of Online Contents with Explanatory Factors. In *Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pages 623–630, 2010.
- [65] K. Lerman and R. Ghosh. Information Contagion: an Empirical Study of Spread of News on Digg and Twitter Social Networks. In *ICWSM*, pages 90–97, 2010.
- [66] K. Lerman and T. Hogg. Using a Model of Social Dynamics to Predict Popularity of News. In *International Conference on World Wide Web (WWW)*, pages 621–630, 2010.

- [67] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M Hurst. Cascading Behavior in Large Blog Graphs. In *SIAM Conference on Data Mining (SDM)*, pages 551–556, 2007.
- [68] Hui Lim, Dongyuan Lu, Tao Chen, and Min-Yen Kan. #mytweet via Instagram: Exploring User Behaviour across Multiple Social Networks. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 113–120, 2015.
- [69] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 80–86, New York, NY, USA, 1998. AAAI Press.
- [70] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [71] R. Dean Malmgren, Jake M. Hofman, Luis A. N. Amaral, and Duncan J. Watts. Characterizing Individual Communication Patterns. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 607–616, Paris, France, 2009. ACM Press.
- [72] Emaad Manzoor, Sadegh M. Momeni, and Leman Akoglu. Fast Memory-efficient Anomaly Detection in Streaming Heterogeneous Graphs. In *International Conference on Knowledge Discovery and Data Mining (ACM KDD)*, pages 1035–1044, San Francisco, CA, USA, 2016.
- [73] Donald W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *SIAM*, 11(2):431–441, 1963.
- [74] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: model and implications. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 6–14, 2012.
- [75] Amy Mitchell, Jeffrey Gottfried, Michael Barthel, and Elisa Shearer. The Modern News Consumer. *Pew Research Center*, 2016.
- [76] H. Müller, P. Clough, T. Deselaers, and B. Caputo. *ImageCLEF Experimental Evaluation in Visual Information Retrieval*. Springer Berlin Heidelberg, 2010.
- [77] A Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.
- [78] João Gama Oliveira and Albert-László Barabási. Human dynamics: Darwin and Einstein correspondence patterns. *Nature*, 437:1251, 2005.
- [79] Raphael Ottoni, Diego Las Casas, João Paulo Pesce, Wagner Meira Jr., Christo Wilson, Alan Mislove, and Virgilio Almeida. Of Pins and Tweets: Investigating How Users Behave Across Image-and Text-Based Social Networks, In *AAAI Conference on Weblogs and Social Media*, 2014.
- [80] T. Pang-Ning, M. Steinbach, and K. Vipin. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, first edit edition, 2005.

- [81] Sridhar Ramaswamy, R Rastogi, and K Shim. Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD Record*, pages 427–438, 2000.
- [82] Shebuti Rayana and Leman Akoglu. Collective Opinion Spam Detection: Bridging Review Networks and Metadata. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 985–994, 2015.
- [83] M. X. Ribeiro, Pedro H. Bugatti, C. Traina Jr., P. M. A. Marques, N. A. Rosa, and A. J. M. Traina. Supporting content-based image retrieval and computer-aided diagnosis systems with association rule-based techniques. *Data & Knowledge Engineering*, 68(12):1370–1382, dec 2009.
- [84] Anderson Rocha, Walter J. Scheirery, Christopher W. Forstall, Thiago Cavalcante, Antonio Theophilo, Bingyu Shen, Ariadne R B Carvalho, and Efstathios Stamatatos. Authorship attribution for social media forensics. *IEEE Transactions on Information Forensics and Security*, PP(99):5–33, 2016.
- [85] Marina Rossi. Protestos contra Dilma voltam embalados por escalada da crise política, In *El País*, March 2016.
- [86] A. Savasere, E. Omiecinski, and S. B. Navathe. An Efficient Algorithm for Mining Association Rules in Large Databases. In *International Conference on Very Large Data Bases (VLDB)*, pages 432–444. Morgan Kaufmann Publishers Inc., sep 1995.
- [87] Galit Shmueli. To explain or to predict? *Statistical Science*, 25(3):289–310, 2010.
- [88] Ka Cheung Sia, Junghoo Cho, and Hyun-Kyu Cho. Efficient monitoring algorithm for fast news alerts. *Transactions on Knowledge and Data Engineering (TKDE)*, 19(7):950–961, 2007.
- [89] Xiaogang Su and Chih-Ling Tsai. Outlier detection. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):261–268, 2011.
- [90] Lorne Swersky, Henrique O Marques, Ricardo J G B Campello, and Arthur Zimek. On the Evaluation of Outlier Detection and One-Class Classification Methods. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10, 2016.
- [91] G. Szabo and B. A. Huberman. Predicting the popularity of online content. *Communications of the ACM*, pages 80–88, 2010.
- [92] Sabrina Tavernise. As Fake News Spreads Lies, More Readers Shrug at the Truth, In *The New York Times*, December 2016.
- [93] Ricardo da Silva Torres and Alexandre Xavier Falcão. Content-based image retrieval: Theory and applications. *Revista de Informática Teórica e Aplicada*, 2(13):161–185, 2006.
- [94] William Trouleau, Azin Ashkan, Weicong Ding, and Brian Eriksson. Just One More: Modeling Binge Watching Behavior. In *International Conference on Knowledge Discovery and Data Mining (ACM KDD)*, pages 1215–1224, San Francisco, CA, USA, 2016.

- [95] Mikalai Tsytsarau, Themis Palpanas, and Malu Castellanos. Dynamics of News Events and Social Media Reaction. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 901–910, New York, NY, USA, 2014. ACM.
- [96] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.
- [97] Pedro O. S. Vaz de Melo, Leman Akoglu, Christos Faloutsos, and Antonio A. F. Loureiro. Surprising patterns for the call duration distribution of mobile phone users. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, pages 354–369, Barcelona, Spain, 2010.
- [98] Pedro O. S. Vaz de Melo, Christos Faloutsos, Renato Assunção, Rodrigo Alvez, and Antonio A. F. Loureiro. Universal and Distinct Properties of Communication Dynamics: How to Generate Realistic Inter-event Times. *Transactions on Knowledge Discovery from Data (TKDD)*, 9(3):24:1–24:31, 2015.
- [99] Pedro O. S. Vaz de Melo, Christos Faloutsos, Renato Assunção, and Antonio A. F. Loureiro. The Self-Feeding Process: A Unifying Model for Communication Dynamics in the Web. In *International Conference on World Wide Web (WWW)*, pages 1319–1330, Rio de Janeiro, Brazil, 2013. IW3C2.
- [100] J. Verbeek, M. Guillaumin, T. Mensink, and C. Schmid. Image annotation with tagprop on the MIRFLICKR set. In *International Conference on Multimedia Information Retrieval (MIR)*, pages 537–546. ACM, 2010.
- [101] C. Wang, M. Ye, and B. A. Huberman. From user comments to on-line conversations. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 244–252. ACM Press, 2012.
- [102] G. Wang, D. Hoiem, and D. Forsyth. Building text features for object image classification. In *International Conference on Pattern Recognition (ICPR)*, pages 1367–1374, 2009.
- [103] T. Weninger, X. Zhu, and J. Han. An Exploration of Discussion Threads in Social News Sites: A Case Study of the Reddit Community. In *Advances in Social Networks Analysis and Mining (ASONAM)*, pages 579–583, 2013.
- [104] Tong Xu, Hengshu Zhu, Xiangyu Zhao, Hao Zhong, Qi Liu, Enhong Chen, and Hui Xiong. Taxi Driving Behavior Analysis in Latent Vehicle-to-Vehicle Networks: A Social Influence Perspective. In *International Conference on Knowledge Discovery and Data Mining (ACM KDD)*, pages 1285–1294, San Francisco, CA, USA, 2016.
- [105] Shipeng Yu, Evangelia Christakopoulou, and Abhishek Gupta. Identifying Decision Makers from Professional Social Networks Categories and Subject Descriptors. In *International Conference on Knowledge Discovery and Data Mining (ACM KDD)*, pages 333–342, San Francisco, CA, USA, 2016.
- [106] Reza Zafarani and Huan Liu. 10 Bits of Surprise: Detecting Malicious Users with Minimum Information. In *International Conference on Information and Knowledge Management (CIKM)*, pages 423–431, 2015.

-
- [107] Chao Zhang, Keyang Zhang, Quan Yuan, Luming Zhang, Tim Hanratty, and Jiawei Han. GMove : Group-Level Mobility Modeling Using Geo-Tagged Social Media. In *International Conference on Knowledge Discovery and Data Mining (ACM KDD)*, pages 1305–1314, San Francisco, CA, USA, 2016.
 - [108] Chao Michael Zhang and Vern Paxson. Detecting and Analyzing Automated Activity on Twitter. *Lecture Notes in Computer Science (LNCS)*, 6579:102–111, 2011.
 - [109] D. Zhang, Md. M. Islam, and G. Lu. A review on automatic image annotation techniques. *Pattern Recognition*, 45(1):346–362, 2011.
 - [110] H. Zhang, A.C. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2126–2136. IEEE, 2006.
 - [111] Tianyang Zhang, Peng Cui, Christos Faloutsos, Wenwu Zhu, and Shiqiang Yang. Come-and-Go Patterns of Group Evolution: A Dynamic Model. In *International Conference on Knowledge Discovery and Data Mining (ACM KDD)*, pages 1355–1364, San Francisco, CA, USA, 2016.
 - [112] Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1513–1522, 2015.
 - [113] H. Zhuang, Y. Sun, J. Tang, J. Zhang, and X. Sun. Influence Maximization in Dynamic Social Networks. In *International Conference on Data Mining (ICDM)*, pages 1313–1318, 2013.
 - [114] A. Znaidia, A. Shabou, A. Popescu, H. le Borgne, and C. Hudelot. Multimodal Feature Generation Framework for Semantic Image Classification. In *International Conference on Multimedia Retrieval (ICMR)*, pages 38:1—38:8, 2012.
 - [115] Mohammad Zulkernine and Jiong Zhang. Network intrusion detection using random forests. In *Conference on Privacy, Security and Trust*, pages 53–61, 2005.