DELIVERABLE SUBMISSION SHEET

To: Susan Fraser *(Project Officer)*

EUROPEAN COMMISSION
Directorate-General Information Society and Media
EUFO 1165A
L-2920 Luxembourg

From:

Project acronym: PHEME Project number: 611233

Project manager: Kalina Bontcheva

Project coordinator The University of Sheffield (USFD)

The following deliverable:

Deliverable title: Longitudinal models of users, networks, and trust: Final Public Release

Deliverable number: D3.3.2

Deliverable date: 31 March 2016

Partners responsible: The University of Sheffield (USFD)

Status: ☒ Public ☐ Restricted ☐ Confidential

is now complete. ☒ It is available for your inspection.
☒ Relevant descriptive documents are attached.

The deliverable is:

☒ a document
☐ a Website (URL: .....)
☐ software (.....)
☐ an event
☒ other (Prototype.....)

Sent to Project Officer:	Sent to functional mail box:	On date:
Susan.Fraser@ec.europa.eu	*CNECT-ICT*-611233@ec.europa.eu	12 April 2016

D3.3.2 Longitudinal models of users, networks, and trust

P.K. Srijith, University of Sheffield
Michal Lukasik, University of Sheffield
Trevor Cohn, University of Melbourne
Kalina Bontcheva, University of Sheffield

Abstract.
FP7-ICT Collaborative Project ICT-2013-611233 Pheme
Deliverable D3.3.2 (WP3)

The deliverable describes the results of Task 3.3 in WP3 on longitudinal modelling of users, pheme virality, and user trustworthiness. The methods described within address the challenge of modelling the evolution of phemes and user behaviour over time. Such longitudinal models aim to predict which phemes will become popular in the near future, as well as helping to identify the most influential users in spreading those phemes.

Keyword list: Hawkes Process, temporal social data, Twitter

Project	Pheme No. 611233
Delivery Date	April 12, 2016
Contractual Date	31 Mar 2016
Nature	Prototype/Report
Reviewed By	Thierry Decklerk
Web links	http://www.pheme.eu
Dissemination	PU

PHEME Consortium

This document is part of the PHEME research project (No. 611233), partially funded by the FP7-ICT Programme.

University of Sheffield
Department of Computer Science
Regent Court, 211 Portobello St.
Sheffield S1 4DP
UK
Contact person: Kalina Bontcheva
E-mail: K.Bontcheva@dcs.shef.ac.uk

MODUL University Vienna GMBH
Am Kahlenberg 1
1190 Wien
Austria
Contact person: Arno Scharl
E-mail: scharl@modul.ac.at

ATOS Spain SA
Calle de Albarracin 25
28037 Madrid
Spain
Contact person: Tomás Pariente Lobo
E-mail: tomas.parientelobo@atos.net

iHub Ltd.
NGONG, Road Bishop Magua Building
4th floor
00200 Nairobi
Kenya
Contact person: Rob Baker
E-mail: robbaker@ushahidi.com

The University of Warwick
Kirby Corner Road
University House
CV4 8UW Coventry
United Kingdom
Contact person: Rob Procter
E-mail: Rob.Procter@warwick.ac.uk

Universitaet des Saarlandes
Campus
D-66041 Saarbrücken
Germany
Contact person: Thierry Declerck
E-mail: declerck@dfki.de

Ontotext AD
Polygraphia Office Center fl.4,
47A Tsarigradsko Shosse,
Sofia 1504, Bulgaria
Contact person: Georgi Georgiev
E-mail: georgiev@ontotext.com

King's College London
Strand
WC2R 2LS London
United Kingdom
Contact person: Robert Stewart
E-mail: robert.stewart@kcl.ac.uk

SwissInfo.ch
Giacomettistrasse 3
3000 Bern
Switzerland
Contact person: Peter Schibli
E-mail: Peter.Schibli@swissinfo.ch

Executive Summary

Online social networks play a major role in generating and disseminating information, as well as being a platform where users tend to voice their opinion. Moreover, social networks provide mainstream media and journalists an opportunity to obtain real time information about events happening around the world. Understanding the evolution of events in social networks is important for government organisations, responders in crises, companies, and ordinary citizens.

This deliverable, in particular, addresses the challenge of modelling the evolution of rumours/phemes and user behaviour over time. Such longitudinal models aim to predict which phemes will become popular in the near future, as well as helping to identify the most influential users in spreading those phemes. This is a very challenging problem, since phemes tend to exhibit highly complex temporal behaviour.

The challenge of longitudinal modelling is addressed through the development of several Hawkes process models. These consider the conversational structure, social network information and, user profile metadata available in phemes. We introduce embeddings of user features and use those to determine the influential and susceptible users.

The different HP models are evaluated on synthetic and real pheme data sets. We found that network information is crucial in learning user influence and that, in turn, it affects the predictive performance. Regularizing the influence matrix irrespective of the connection information can be detrimental.

The second part of the deliverable focuses on identifying spam users in phemes, where we found the logistic regression classifier to perform best. The balance of spam vs non-spam users in the training data was found to impact performance, with the widely used MPI spam user dataset being the best for training. Manual inspection of the top 100 suspected spam users in two pheme datasets found that, in general, the amount of spam tweets and spam users within was very low.

Contents

Chapter 1

Introduction

Online social networks play a major role in generating and disseminating information, as well as being a platform where users tend to voice their opinion. Moreover, social networks provide mainstream media and journalists an opportunity to obtain real time information about events happening around the world. Understanding the evolution of events in social networks is important for government organisations, responders in crises, companies, and ordinary citizens.

This deliverable, in particular, addresses the challenge of modelling the evolution of rumours/phemes and user behaviour over time. Such longitudinal models aim to predict which phemes will become popular in the near future, as well as helping to identify the most influential users in spreading those phemes.

This is a very challenging problem, since phemes (and memes) tend to exhibit highly complex temporal behaviour. For instance, Figure 1.1 represents the evolution of different rumours associated with the Ferguson unrest in 2014.

Statistical models based on point processes provide a suitable framework for modelling the temporal dynamics of Twitter. In particular, since a given tweet may easily lead to a fast-spreading pheme this can be modelled using a self-exciting point process, called Hawkes Process (HP) (Yang and Zha, 2013). HPs consider the influence from previous tweets in the generation of future tweets. In particular, we consider multivariate HPs, which consider the influence of other users in the network when modelling the appearance of new tweets. The model has been found useful for modelling the popularity of tweets (Zhao et al., 2015). This prior work, however, does not consider features specific to Twitter, which we believe are useful for modelling tweet popularity. Other related work on popularity prediction (Cheng et al., 2014; Hong et al., 2011; Kupavskii et al., 2012; Suh et al., 2010) has indeed used content, network, and temporal features, coupled with simple logistic regression models. Our Hawkes Process approach combined elegantly those features with the point process framework, resulting in models with better predictive performance.

The deliverable investigates different HP variants, which consider different

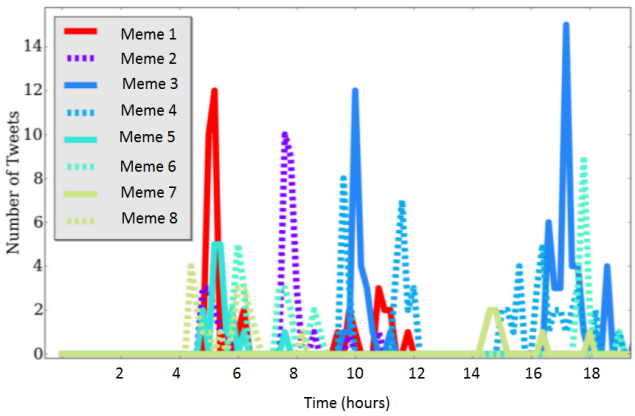

Figure 1.1: Temporal profile of rumours in the Ferguson data set

Twitter-specific information. The first HP model considers the Twitter conversational threads, which convey information about infectivity, which can be useful in predicting tweet popularity. The second HP model considers the user social networks. In Twitter, for each user we can obtain information about which users are being followed by the given user. This can be used as a prior information, when modelling influence among users. Twitter also makes available user statistics, such as the user's number of tweets, retweets, friends and followers. This information is useful in modelling the influences with which the given user spreads information.

We evaluate the proposed models on the PHEME rumour datasets, collected around key events such as the Ferguson unrest and the Ottawa shooting. We evaluate the ability of the different HP models to predict the evolution of rumours and users behaviour. We found that incorporating Twitter features resulted in improved performance.

The second part of the deliverable describes our work on detecting spam users in rumorous threads. The ability to filter out spam users is key, as these users have very low trustworthiness and their tweets should be disregarded during subsequent rumour analysis.

1.1 Relevance to PHEME

The PHEME project aims to detect and study the emergence and propagation of phemes in social media, i.e. memes with attached veracity information. An important aspect of pheme analysis is predicting whether an emerging pheme is likely to become viral, and thus needs to be acted upon urgently. This is precisely the task addressed by the longitudal Hawkes Process models, described in this deliverable. Likewise, as a given pheme is starting to gain momentum, our models aim to identify

automatically the most influential users, responsible for the pheme propagation, as well as identifying all spam users, who need to be filtered out.

1.1.1 Relevance to project objectives

Work in this deliverable is directly relevant to the third project objective, i.e. predicting pheme virality over time, and within users' social networks. Identification of influential users and untrustworthy users and their content is another key aspect of this objective.

1.1.2 Relation to other workpackages

The methods reported in this deliverables make use of the linguistic pre-processing components developed in WP2. They are now being integrated within the PHEME veracity framework (WP6) and thus, the end users of the WP7 and WP8 prototypes will soon be able to experiment with the results. The PHEME visual dashboard (WP5) will be able to show influential users and viral phemes.

1.2 Related Work

Information cascades in social networks are predominantly modelled by extracting relevant features and learning machine learning models. These approaches use content, network, and temporal features and learn simple regression models or regression trees (Agarwal et al., 2009; Cheng et al., 2014; Bakshy et al., 2011). The problem of predicting the popularity of tweets is solved as a classification problem considering content, network and user information in (Hong et al., 2011). In addition to user features, (Kupavskii et al., 2012) also uses features determining flow of cascade and PageRank to train a machine learning algorithm, which can predict number of retweets. It is found in (Suh et al., 2010) that content features (e.g. hashtags and URLs) and contextual features (e.g. number of followers) affect retweetability.

There are also approaches based on point process models, which do not require extensive feature engineering to model Twitter dynamics. However, most of these models are developed to infer the underlying influence network through information cascades (Du et al., 2012; Gomez-Rodriguez et al., 2011; Gomez Rodriguez et al., 2013b,a; Yang and Zha, 2013). These models were successfully applied to study the spread of memes in social networks. Recently, (Zhao et al., 2015) developed a model based on self exciting point processes to predict the popularity of tweets, while (Lukasik et al., 2015a) used a log-Gaussian Cox processes to predict the evolution of rumours. Our approach differs from this prior work in that it uses user, content and network features in a point process framework to model the evolution of phemes and

behaviour of users. Thus, it conjoints two related approaches (point process based and feature based) into a unified framework for modelling Twitter dynamics.

Chapter 2

Modelling Twitter Dynamics

We consider a set of N tweets, $D = \{d_1, \cdots, d_N\}$, belonging to M phemes. Let R denote the total number of users in the Twitter data set. Each tweet contains information about their time of occurrence, text, pheme topic and the user who generated the tweet. Thus, a tweet can be represented as a tuple $d_n = (t_n, m_n, i_n)$, where: t_n is the time of tweet occurrence, m_n is the pheme topic and i_n is the user generating the tweet. We assume the dataset is ordered by the time of occurrence of tweets. Given the history of past occurrences of tweets, our goal is to predict future occurrences of tweets (i.e. pheme virality), as well as identify influential users. We use a point process framework to model the occurrence of tweets, namely a self-exciting point process called Hawkes Process (HP). We propose various Hawkes process models for predicting the temporal dynamics of phemes.

2.1 Point Processes

The problem of modelling longitudinal pheme dynamics can be solved using a point process such as Poisson processes (Lukasik et al., 2015b; Perera et al., 2010) over a continuous time period. Poisson processes indexed by time are also known as counting processes as they count the number of occurrences of events (i.e. tweet appearance) over time. A homogeneous Poisson process (HPP) assumes the intensity to be constant (with respect to time), which makes them ill suited to modelling pheme dynamics, which is much more irregular.

Inhomogeneous Poisson process (IPP) (Lee et al., 1991) can model tweets occurring at a variable rate by considering the intensity to be a function of time, *i.e.* $\lambda(t)$. For example, in Figure 2.1 we show the points generated by an IPP with two different intensity functions. The occurrence of points are different for the two intensity functions. We can also observe that the number of point occurrences are higher in the regions of larger intensity.

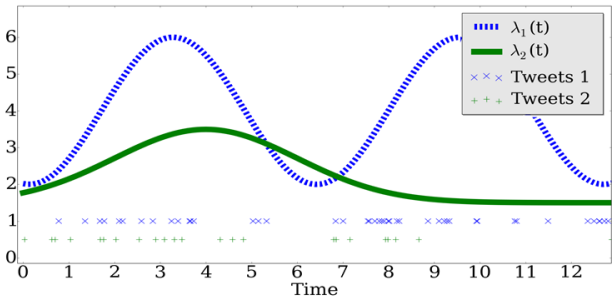

Figure 2.1: Points generated from an inhomogeneous Poisson process with different intensities

In an IPP, the number of tweets (y) occurring in an interval $(0, s)$ is Poisson distributed with rate $\int_0^s \lambda(t)dt$.

$$p(y|\lambda(t),(0,s)) = \text{Poisson}(y|\int_0^s \lambda(t)dt) = \frac{(\int_0^s \lambda(t)dt)^y \exp(-\int_0^s \lambda(t)dt)}{y!} \quad (2.1$$

The rate $\int_0^s \lambda(t)dt$ provides the expected number of occurrences in the interval $[0, s]$. Assuming an event occurred at time 0, the intensity function also allows us to find the probability that no event occurs by time s (survival probability) which is given by $\exp(-\int_0^s \lambda(t)dt)$. This is obtained by substituting $y = 0$ in Equation (2.1). The probability density that an event occurs at time s in the interval $(0, s)$ is obtained from the survival probability as:

$$\lambda(s)\exp(-\int_0^s \lambda(t)dt) \quad (2.2$$

2.2 Multivariate Hawkes Process

Due to the social nature of Twitter, users tend to influence one another – a process which occurs alongside exposure to past tweets. Therefore, we propose to model such mutually exciting phenomena between users and tweets through multi-variate Hawkes processes. A multi-variate Hawkes process is modelled through an underlying non-negative intensity function which considers the influence of tweets from other users. The intensity function models the self-exciting nature by adding up influence from past tweets. We assume that the temporal dynamics of different rumour categories are independent of each other, given their corresponding intensity function.

The intensity function used in a Hawkes process depends on the history of previous occurrences of tweets. This is a conditional intensity function, depending on

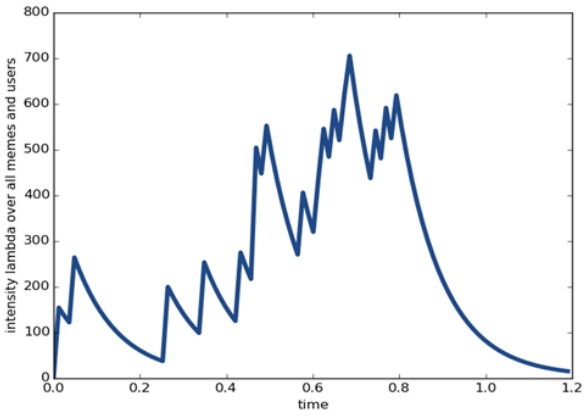

Figure 2.2: Intensity function associated with a Hawkes process

tweet history $(\mathcal{H}_t)$[1]. The intensity function associated with a multivariate Hawkes process takes the following form:

$$\lambda_{i_n,m_n}(t) = \mu_{i_n}\gamma_{m_n} + \sum_{\ell=1}^{n-1} \mathbb{I}(m_\ell = m)\alpha_{i_\ell,i_n}\kappa(t - t_\ell) \qquad (2.3$$

where the first term represents the constant base intensity in which the tweets are generated by user i_n for pheme m_n. The second term represents the influence from the tweets that happen prior to time of interest. The influence from each tweet decays over time and is modelled using an exponential decay term $\kappa(t - t_\ell) = \omega \exp(-\omega(t - t_\ell))$. The matrix α of size $R \times R$ encodes the degrees of influence between pairs of users generating the tweets. The influence the users have on each other is asymmetric and so is the α matrix. The α matrix provides the underlying influence network among the users. Since the intensity function has to be non-negative, all the parameters associated with the intensity function are also non-negative. Figure 2.2 shows example behaviour of an intensity function associated with a Hawkes process.

The intensity function associated with the Hawkes process can be seen to be obtained from the superposition of

1. Poisson process with constant intensity $\mu_i\gamma_m$ and
2. Poisson processes with intensities $\alpha_{i_\ell,i}\kappa(t - t_\ell)$

Hawkes process intensity function given in Equation (2.3) can be obtained by applying the Poisson superposition theorem (Kingman, 1993).

[1]For notational convenience, we ignore the conditioning of intensity with respect to $\mathcal{H}_t$

2.3 Considering Conversational Structure

The intensity function defined in (2.3) for a Hawkes process assumes all previous tweets influence the current tweet. However, many of the new tweets are generated in response to other unrelated tweets, in different conversational threads.

Therefore, we modify the model to take into account the tweet thread structure, which explicitly encodes which tweets have influenced the generated tweet directly. In addition, the proposed model avoids the computational complexity of summing over all previous tweets for calculating the HP intensity function. We modify the intensity function defined in (2.3) to consider this explicit infectivity information.

Let U represents an $N \times N$ binary matrix with U_{ij} representing whether tweet i has influenced tweet j. If i influenced j, then $U_{ij} = 1$ otherwise 0. In cases where the infecting tweet is unknown, we assume the tweet is spontaneous, *i.e.* generated by the user himself without influence of other tweets. In other words, we assume that a tweet is either spontaneous or caused by another tweet. Thus, only one column in a row of the matrix is 1 and the rest of the elements are 0. Also, tweets happening later in time have no influence on the ones happening before. Hence, U is an upper triangular matrix.

The consideration of conversational information leads to a Hawkes process (HP-conversation) with intensity function defined as follows.

$$\lambda_{i_n,m_n}(t) = \mu_{i_n}\gamma_{m_n}U_{nn} + \sum_{\ell=1}^{n-1}\mathbb{I}(m_\ell = m)U_{ln}\alpha_{i_\ell,i_n}\kappa(t-t_\ell) \qquad (2.4$$

Here, only one of the term in (2.4)is active for a tweet. If the tweet is spontaneous, then intensity is same as base intensity. If the tweet is under the infection of another tweet, then the intensity is given by the influence the infecting tweet has on the current tweet. The intensity function in (2.4) can also be written in a product form as

$$\begin{aligned}\lambda_{i_n,m_n}(t_n) &= \left[\mu_{i_n}\gamma_{m_n}\right]^{U_{n,n}} \\ &\times \prod_{\ell=1}^{n-1}\left[\alpha_{i_\ell,i_n}\kappa(t_n-t_\ell)\right]^{\mathbb{I}(m_\ell=m_n)U_{\ell,n}},\end{aligned} \qquad (2.5$$

where only one term in the product is active for any tweet.

The parameters associated with the intensity function are learnt by maximizing the likelihood over observations. For a Hawkes process, the complete likelihood is given by

$$L(\boldsymbol{t},\boldsymbol{i},\boldsymbol{m}) = \prod_{n=1}^{N}\lambda_{i_n,m_n}(t_n)\times P(E_T) \qquad (2.6$$

where, $P(E_T)$ represents the likelihood that no event happens in the interval $[0,T]$ except at times given in the data set. Here, T represent an upper bound on the time period considered in the data set.

The likelihood $p(E_T)$ is given by the survival function $S(T) = \exp(-\sum_{i=1}^{R}\sum_{m=1}^{M}\int_0^T \lambda_{i,m}(s)ds)$. The parameters in the Hawkes process model are estimated by maximizing the log-likelihood

$$\begin{aligned}&\ell(\boldsymbol{t}, \boldsymbol{i}|\boldsymbol{\mu}, \boldsymbol{\gamma}, \boldsymbol{\alpha}, \omega) = \\ &+\sum_{n=1}^{N}\log\lambda_{i_n,m_n}(t_n) - \sum_{i=1}^{R}\sum_{m=1}^{M}\int_0^T \lambda_{i,m}(s)ds\end{aligned} \quad (2.7$$

Substituting the intensity function defined in (2.5) in (2.7) results in the following optimization problem:

$$\begin{aligned}&\arg\max_{\boldsymbol{\mu},\boldsymbol{\gamma},\boldsymbol{\alpha},\omega}\sum_{n=1}^{N}U_{n,n}\log(\mu_{i_n}\gamma_{m_n}) \\ &+\sum_{n=1}^{N}\sum_{\ell=1}^{n-1}\mathbb{I}(m_\ell = m_n)U_{\ell,n}\log\alpha_{i_\ell,i_n}\kappa(t_n - t_\ell) \\ &-T\sum_{i=1}^{R}\sum_{m=1}^{M}\mu_i\gamma_m - \sum_{i=1}^{R}\sum_{\ell=1}^{N}\alpha_{i_\ell,i}K(T-t_\ell)\end{aligned} \quad (2.8$$

where $K(T-t_\ell) = 1-\exp(-\omega(T-t_\ell))$ arises from the integration of $\kappa(t-t_\ell)$. A coordinate descent approach is followed to solve the optimization problem where each parameter is solved keeping all others fixed. The updated values of the parameters $\boldsymbol{\mu}, \boldsymbol{\gamma}$ and $\boldsymbol{\alpha}$ can be obtained in a closed form.

$$\begin{aligned}\mu_i &= \frac{\sum_{n=1}^{N}U_{n,n}\mathbb{I}(i_n = i)}{T\sum_{m=1}^{M}\gamma_m} \quad \gamma_m = \frac{\sum_{n=1}^{N}U_{n,n}\mathbb{I}(m_n = m)}{T\sum_{i=1}^{R}\mu_i} \\ \alpha_{i,j} &= \frac{\sum_{n=1}^{N}\sum_{\ell=1}^{n-1}\mathbb{I}(m_\ell = m_n)\mathbb{I}(i_l = i)\mathbb{I}(i_n = j)U_{\ell,n}}{\sum_{\ell=1}^{N}\mathbb{I}(i_l = i)K(T-t_\ell)}\end{aligned} \quad (2.9$$

The parameter ω is obtained by maximizing the following objective function using gradient based techniques with the constraint ω to be positive.

$$\begin{aligned}J(\omega) &= \sum_{n=1}^{N}\sum_{\ell=1}^{n-1}\mathbb{I}(m_\ell = m_n)u_{\ell,n}\left(\log\omega - \omega(t_n - t_\ell)\right) \\ &\quad +\sum_{\ell=1}^{N}e^{-\omega(T-t_\ell)}\sum_{i=1}^{R}\alpha_{i_\ell,i}\end{aligned} \quad (2.10$$

2.4 Influence decomposition

The user infectivity and susceptibility information is available from the influence matrix. The rows of the influence matrix represents influential users, while the columns represents susceptible users. The influence matrix can be decomposed to learn latent features generating susceptible users and influential users. We propose a HP model where we decompose the influence matrix to lower rank factors. These lower rank factors provide latent features determining influence and susceptibility.

The advantage of the model is that it can learn the influence matrix with a lower number of parameters. This is especially useful in social networks with large number of users, as is the case with Twitter. Assuming the rank of the latent factors to be K, we now need to estimate only $2 \times R \times K$ parameters instead of $R \times R$ parameters. This helps avoid over-fitting and thus could improve the predictive performance of the model. The experimental results we discuss later support this claim.

We assume the influence matrix $\boldsymbol{\alpha}$ can be decomposed into a product of two lower dimensional matrices $\mathbf{I}$ and $\boldsymbol{S}$ of rank K as shown in Figure 2.3. Thus, $\alpha_{i,j} = \sum_{k=1}^{K} I_{ik} S_{jk}$, where $\mathbf{I}$ is the influence embedding and $\boldsymbol{S}$ is the susceptibility embedding. bI gives a lower dimensional latent representation of user features governing influence, while bS provides a representation of user features governing susceptibility. Since each element of $\boldsymbol{\alpha}$ is positive, we consider a non-negative matrix factorization of $\boldsymbol{\alpha}$ into components $\mathbf{I}$ and $\boldsymbol{S}$. This results in a Hawkes process (HPdecomposition) with intensity function of the following form:

$$\lambda_{i_n,m_n}(t) = \mu_{i_n}\gamma_{m_n}U_{nn} + \sum_{\ell=1}^{n-1} \mathbb{I}(m_\ell = m) U_{ln} [\mathbf{I} \cdot \boldsymbol{S}^\top]_{i_\ell,i_n} \kappa(t - t_\ell). \qquad (2.11$$

The latent factors are learnt by maximizing the log-likelihood obtained by plugging the intensity function (2.11) in (2.7). We use a gradient based optimization to learn the latent factors I and S. The optimization is constrained to maintain non-negativity of the latent factors.

2.5 Network Information

Twitter also provides information about connectivity between users. Unlike other social networks such as Facebook, the relation between users is not symmetric. In Twitter users follow other users, for instance user i may follow user j while user j may not follow user i. We can obtain 'who follows whom' information from Twitter and this can be very useful in modelling the way users influence each other.

In other words, if user i follows user j, there is a higher chance that the tweets of user i are influenced by user j than by those published by an unconnected user k. We

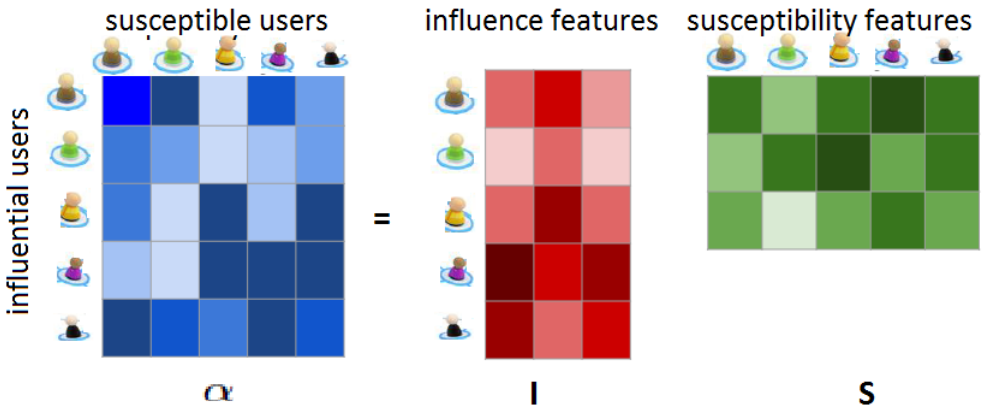

Figure 2.3: Non-negative matrix factorization of Influence matrix

provide an approach to consider this prior social connection information in a Hawkes process learning framework (HPconnection). This is achieved by regularizing the $\boldsymbol{\alpha}$ matrix appropriately while learning it using the Hawkes process framework.

Let $\mathcal{F}$ represent the set containing the followee and follower user pairs. The users which are not connected in the network are less likely to influence each other. For these users we would like to obtain low values in the corresponding entries of the $\boldsymbol{\alpha}$ matrix. This can be achieved by regularizing the values of the $\boldsymbol{\alpha}$ matrix corresponding to the disconnected users. We use $L2$ regularization (squared loss) over such values. In this case, $\boldsymbol{\alpha}$ is learnt by minimizing the following objective function subject to the constraint that $\boldsymbol{\alpha} \geq 0$.

$$\begin{aligned} &\arg\max_{\boldsymbol{\alpha}} \sum_{n=1}^{N} U_{n,n} \log(\mu_{i_n} \gamma_{m_n}) \\ &+ \sum_{n=1}^{N} \sum_{\ell=1}^{n-1} \mathbb{I}(m_\ell = m_n) U_{\ell,n} \log \alpha_{i_\ell, i_n, m_n} \kappa(t_n - t_\ell) \\ &- \sum_{i=1}^{R} \sum_{\ell=1}^{N} \alpha_{i_\ell, i, m_\ell} K(T - t_\ell) - \sum_{i,j \notin \mathcal{F}} \alpha_{i,j}^2 \end{aligned} \quad (2.12)$$

The gradient of (2.12) has a quadratic form and hence alpha can be obtained in a closed form using the formula to find roots of a quadratic form.

2.6 Incorporating User Features

Twitter also provides user metadata, in addition to the conversational structures and network information already discussed above. Therefore, we propose also a corresponding modification of the Hawkes process approach, where user features are

Table 2.1: User features used for learning influences.

Feature id	Feature name
rt_ratio	proportion of retweets
hash_ratio	proportion of hashtags
hash_tok_rat	hashtag-token ratio
ment_ratio	proportion of user-mentions
unq_ment	number of unique mentions
lnk_ratio	links ratio
rpl_tw_ratio	proportion of reply tweets
fav_cnt	number of favourites
hist_tw	number of historical tweets
pro_bgr	binary profile background
pro_img	binary profile image
pro_loc	binary geolocation

taken into account (e.g. location, number of followers and followees). As demonstrated in prior work, user features can play an important role in determining the influence and susceptibility of Twitter users (Lampos et al., 2014). Next we demonstrate how the underlying influence network ($\boldsymbol{\alpha}$) can be modelled to consider such user features.

Let u_i represent features of the i^{th} user. Let the number of user features be D. We find an embedding of the user features in a lower dimensional space of size K. We assume that users have different embeddings representing their infectivity and susceptibility. The infectivity embedding is obtained by linearly transforming the user features using matrix I (of size $D \times K$) while the susceptibility embedding is obtained by linearly transforming user features through matrix S (of size $D \times K$). The influence matrix $\boldsymbol{\alpha}$ is obtained using a non-linear transformation of the infectivity and susceptibility embeddings. We consider a sigmoid function as the non-linear function, since it ensures non-negativity of elements in $\boldsymbol{\alpha}$.

The intensity function associated with a Hawkes process considering user features (HPfeatures) is:

$$\begin{aligned}&\lambda_{i_n,m_n}(t) = \mu_{i_n}\gamma_{m_n}U_{nn}+ \\ &\textstyle\sum_{\ell=1}^{n-1}\mathbb{I}(m_\ell = m)U_{ln}\sigma([\mathbf{u}_{i_l}I]\cdot[\mathbf{u}_iS]^\top)\kappa(t-t_\ell)\end{aligned} \qquad (2.13$$

where σ represents a sigmoid. In this case, we do not restrict the matrices $\mathbf{I}$ and $\boldsymbol{S}$ to be non-negative, since we use the sigmoid to have a non-negative representation of the influence matrix. The matrices $\mathbf{I}$ and $\boldsymbol{S}$ are learnt by maximizing the likelihood (2.7) by plugging in the intensity function (2.13).

We consider the 12 user features listed in Table 2.1, which are contained in the user field of each tweets and are thus readily available in the training data. The

Algorithm 1 Ogata's thinning algorithm for sampling points from Hawkes process

Input: conditional intensity function $\lambda(t)$, past tweet occurrence times t_1, t_2, t_n, time T
Initialize $t = t_n$, $S = \{\}$.
while $t \leq T$ **do**
  Compute $\beta = \lambda(t)$.
  Generate candidate next arrival time from $HPP(\beta)$, $s \sim \exp(\beta)$
  Generate random number $U \sim Unif([0, 1])$
  if $(t + s > T)$ **or** $(U > \frac{\lambda(t+s)}{\beta})$ **then**
    Set $t = t + s$
  else
    Set $t = t + s$, $S = S \cup t$
  end if
end while
Return: S

links ratio mentioned in Table 2.1 depends on the number of followers (ϕ_{in}), the number of followees (ϕ_{out}) and the number of times a user has been listed by others (ϕ_{lis}). The links ratio is calculated as in (Lampos et al., 2014)

$$\log(\frac{(\phi_{lis} + 1)(\phi_{in} + 1)^2}{\phi_{out} + 1}) \qquad (2.14$$

2.7 Prediction Algorithm

As noted above, the parameters associated with the intensity function are learnt using a maximum likelihood approach. We use the learnt intensity function to make predictions in the test interval. The predictions are done by sampling points from the learnt intensity function using Ogata's modified thinning algorithm (Ogata, 1981) (see Algorithm 1). The thinning algorithm is used to sample points from a point process given their conditional intensity. Each step of the algorithm samples a point from a Homogeneous Poisson Process (HPP) with a constant intensity β. This intensity is taken as an upper bound to the conditional intensity $\lambda(t)$ associated with the Hawkes process. Assuming an event occurred at time u,in order to sample a point in the interval $[u, v]$, the algorithm considers an HPP with intensity $\beta \geq \sup_{t\in[u,v]} \lambda(t)$. For a Hawkes process this supremum can be easily obtained as $\beta = \lambda(u)$. This is due to the fact that the intensity associated with a Hawkes process is highest at the point of occurrence of the event and then decreases exponentially over time. The point generated by the HPP with intensity β is accepted as a point from a Hawkes process with probability $\frac{\lambda(t)}{\beta}$.

We predict tweet occurrences separately for each user and each pheme, as well as jointly across phemes and users. The prediction of a future tweet for a user i

is done by considering an intensity function summed over all phemes, *i.e.* $\lambda_i(t) = \sum_{m=1}^{M} \lambda_{i,m}(t)$. Similarly, when we predict the likely virality of a pheme m, we consider an intensity function $\lambda_m(t) = \sum_{i=1}^{R} \lambda_{i,m}(t)$.

When predictions are made jointly across users and phemes, we consider the intensity to be $\lambda(t) = \sum_{i=1}^{R} \sum_{j=1}^{M} \lambda_{i,j}(t)$.

Chapter 3

Experiments in Predicting Pheme Virality and User Influence

We conduct experiments to study the behaviour of various Hawkes process models. The experiments try to evaluate the predictive performance of the models as well as their ability to predict popular rumours. We also study how well it can learn influences between users. Learning influential users will be helpful in preventing the spread of rumours in Twitter.

3.1 Evaluating predictive performance

This section describes experiments that evaluate the predictive ability of the different HP models, introduced in the previous chapter. The models are trained on tweets up until a given point in time and then the learnt models are used to predict likely tweet time in the future. Predictive performance of the models is evaluated alongside three different dimensions:

1. User centric: ability to predict future tweets of a given user, irrespective of the pheme they appear in. This measures how well we model each user.
2. Pheme centric: ability to predict future tweets in each pheme, irrespective of the user who generated it. This measures how well we model each pheme.
3. Joint: ability to predict future tweets, irrespective of the pheme and users. This measures the overall predictive performance of each model.

Let the arrival times predicted by a model be $(\hat{t}_1, \ldots, \hat{t}_M)$ and let the actual arrival times be $(t_1, \ldots, t_N)$. In these experiments, predictive performance is measured using aligned root mean squared error (ARMSE), which aligns the initial $K = \min(M, N)$ arrival times and calculates the Root Mean Squared Error (RMSE)

Table 3.1: Properties of the datasets

Data	# Phemes	# Users	# tweets
Synthetic	5	5	2000
Ferguson	31	1481	2190
Ottawa	39	4096	6134

between the two sub-sequences of predicted and actual tweet arrival times. For the user centric and pheme centric evaluations, ARMSE scores are calculated separately for each user and pheme respectively and the average ARMSE is presented along standard deviation.

Tweets are ordered based on their timestamps. The first 70% are used to learn the parameters and the rest are used to evaluate the predictive performance of the different HP methods.

In the rest of this chapter we report experiments on both synthetic and real world pheme datasets. The purpose of the synthetic dataset is to evaluate the correctness of the model. The real world datasets consists of phemes that emerged during two major events: the Ferguson unrest and the Ottawa shooting. The properties of the datasets are shown in Table 3.1. In all these experiments, tweet timestamps are used at hourly grannularity.

The Hawkes process models discussed in Chapter 2 are compared against two Hawkes process baselines:

1. HPbase: A Hawkes process model, which considers a constant influence matrix, where all the values are set to 1.

2. HPregularization: A Hawkes process model, which learns the influence matrix by regularizing all elements in the $\boldsymbol{\alpha}$ matrix to be close to zero. This is achieved by using $L2$ regularization over elements of the $\boldsymbol{\alpha}$ matrix, irrespective of user connection information.

3.1.1 Experiments on Synthetic Data

The synthetic dataset is generated by fixing the number of users, phemes and the parameters associated with the intensity function. The times of tweets are simulated using Ogata's thinning algorithm.

In particular, the number of users and phemes in the synthetic data are both set to 5. The parameters of the intensity function used to generate the synthetic data are as follows: $\mu = [0.13, 0.13, 0.09, 0.14, 0.003]$ and $\gamma = [0.97, 0.68, 0.92, 0.96, 0.64]$. These values are generated randomly from a uniform distribution between 0 and 1.

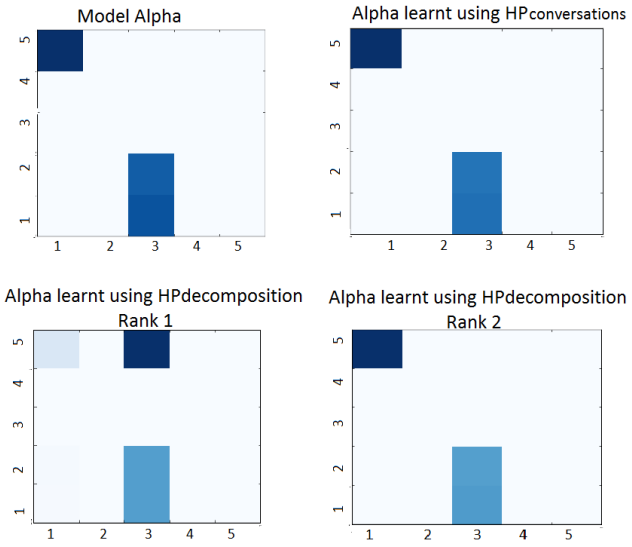

Figure 3.1: Ground truth and learnt α matrices for the synthetic experiment.

The parameter ω in the delay function is fixed to a constant 1. We assume users 1 and 2 influence user 3 and user 5 influences user 1. Therefore, the $\boldsymbol{\alpha}$ values are zero except for $\alpha_{1,3}$, $\alpha_{2,3}$ and $\alpha_{5,1}$ which are set to 0.45, 0.42 and 0.52 respectively. We conduct experiments to verify the ability of each model to learn the influence network, as well as evaluating the predictive performance.

Figure 3.1 provides the heat map associated with the influence matrix learnt by the different HP models. We can observe that the $\boldsymbol{\alpha}$ matrix learnt using *HPconversation* is close to the model $\boldsymbol{\alpha}$ which generated the observations. We also show the $\boldsymbol{\alpha}$ matrix learnt using *HPdecomposition* with two different ranks. The *HPdecomposition* with rank 1 was not able to learn the influences correctly, whereas *HPdecomposition* with rank 2 performed better at this task.

We also conduct experiments to study the dependence of the influence matrix on the number of training examples. Figure 3.2 depicts how the average relative difference of $\boldsymbol{\alpha}$ values learnt using *HPconversation* differs from the model $\boldsymbol{\alpha}$ as more and more points are generated using the model. We observe that with more points available to learn $\boldsymbol{\alpha}$ the difference becomes smaller and converges to zero.

The synthetic dataset is also useful for evaluating the predictive performance of *HPconversation* and *HPdecomposition*. Since *HPconnection* requires a social network and *HPfeatures* needs user features, we did not evaluate these methods on the synthethic data, since it lacks these two types of information.

As can be seen in Table 3.2, in all three evaluation scenarios the inclusion of the conversational structure and learning user-specific influences leads to a significantly improved predictive performance. The *HPdecomposition* approach with rank 5 has performed better than *HPbase*, but not as well as *HPconversation*. *HPdecomposition*

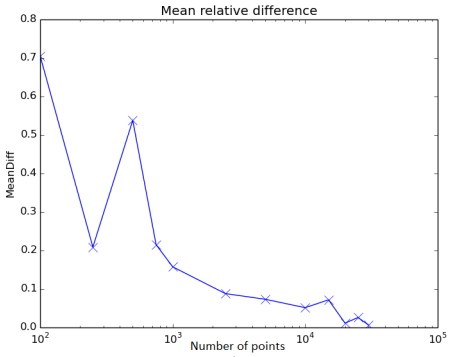

Figure 3.2: Relative difference in learnt α values against number of points

Table 3.2: Results on the synthetic dataset, reporting using ARMSE

Approach	User	Pheme	Joint
HPbase	4.39±2.88	3.42±3.03	3.31
HPconversation	0.74±1.19	0.17±0.20	0.22
HPdecomposition(5)	0.81±0.91	0.57±0.24	0.41

did not improve over *HPconversation* in the synthetic dataset, due to the smaller number of users.

3.1.2 Experiments on the Ferguson phemes

The Ferguson dataset consists of tweets collected during the Ferguson unrest in 2014, collected and annotated by journalists from SwissInfo (WP8). In particular, 2190 tweets were labelled manually by journalists, as belonging to 31 different phemes. It contains tweets from 1481 users. For a detailed description, see (Zubiaga et al., 2016).

First, we conduct experiments on the predictive performance of *HPdecomposition* for a range of factors. Figure 3.3 plots how predictive performance in the rumour centric evaluation varies with respect to the ranks of the factors used in decomposition. We can observe that lower rank factors tend to give better performance than higher rank factors.

Next, Table 3.3 compares the various HP models on their ability to predict tweet occurrence. We observe that *HPconversation* (learns influences among users based on the conversation structure) was able to give a better predictive performance. The performance of *HPdecomposition* approach with rank 100 was found to be close to that of *HPconversation* but not as good. However, *HPdecomposition* with rank 1 outperformed *HPconversation*s in all three evaluation settings. This is due to

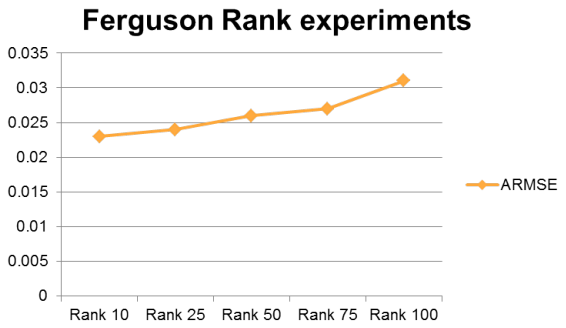

Figure 3.3: Variation in predictive performance of *HPdecomposition* with respect to rank.

Table 3.3: Results on Ferguson data using ARMSE measure

Approach	User	Pheme	Joint
HPbase	0.018±0.06	0.052±0.10	0.29
HPregularization	0.018±0.06	0.06±0.18	0.17
HPconversation	0.012±0.05	0.032±0.08	0.08
HPdecomposition(100)	0.013±0.04	0.038±0.06	0.12
HPdecomposition(1)	0.003±0.03	0.02±0.04	0.03
HPconnection	0.009±0.02	0.02±0.05	0.02
HPfeatures (1)	0.003±0.035	0.049±0.137	0.13

it needing a lower number of parameters to learn infectivity and, thus, avoiding over-fitting. It is particularly useful with sparse data and a large number of users.

The results also demonstrate that *HPconnection* (which considers the network information) also performs very well.

Table 3.3 also compares the predictive performance of the HP model which consider user features(*HPfeatures*). Although it doesn't offer significant performance improvement, it does have the ability to learn the features affecting infectivity and susceptibility. The main reason for the inferior performance of *HPfeatures* is due to the non-linear transformation (sigmoid) which restricts the alpha values to be between 0 and 1.

We learn the features influencing user infectivity and susceptibility from the one dimensional matrices (weight vectors) that transform user features to a latent dimension. Table 3.4 lists the features ordered by the magnitude of weight vector values. We find that features such as number of historical tweets, reply tweets and favourites have a larger impact on infectivity than other features. The susceptibility of a user is mainly affected by links ratio and number of user mentions.

Table 3.4: Top user features determining infectivity and susceptibility on Ferguson data

Infectivity		Susceptibility	
feature id	Value	feature id	Value
hist_tw	1.52	lnk_ratio	-7.55
rpl_tw_ratio	0.68	ment_ratio	-4.90
fav_cnt	-0.59	unq_ment	4.40
hash_ratio	0.54	hash_ratio	-3.72
unq_ment	-0.49	rpl_tw_ratio	-2.15
ment_ratio	0.39	fav_cnt	1.51
lnk_ratio	-0.21	pro_bgr	-0.99
pro_img	0.20	hist_tw	0.85
hash_tok_rat	0.15	pro_img	-0.43
pro_bgr	-0.03	pro_loc	-0.34
pro_loc	-0.01	hash_tok_rat	0.12
rt_ratio	0	rt_ratio	0

3.1.3 Experiments on the Ottawa phemes

The Ottawa data set consists of tweets related to shootings at the parliament building in Ottawa during October 2014 (Zubiaga et al., 2016). It is structured similarly to the Ferguson dataset, including the complete tweet threads (i.e. source tweets and their replies). The dataset consists of 6134 tweets covering 39 rumours and 4,096 users.

Table 3.5 reports predictive performance on this dataset, alongside the three user, rumour and joint evaluation criteria. For user based evaluation, *HPconnection* gives best performance. This implies that considering the underlying network information is useful for learning influence matrices, which in turn helps predict user behaviour.

Unsurprisingly, for pheme-centric evaluation, connection information was not found to be as useful as for user-centric evaluation. This is due to connection information helping to predict tweeting behaviour of individual users, but not the evolution of phemes over time. It was, nevertheless, found to be helpful in joint evolution, due to the larger number of users.

As in the case of Ferguson, *HPdecomposition* outperforms *HPconversation*. Here, *HPfeatures* is found to outperform *HPconversation* and *HPdecomposition* when predicting the joint evolution of phemes and users.

Next, Table 3.6 presents the list of user features influencing susceptibility and infectivity, ordered by the magnitude of weight vector values. We find that the order of user features affecting susceptibility is the same as in the Ferguson dataset, with links ratio being the most important. In the Ottawa data set, the links ratio also

Table 3.5: Results on the Ottawa dataset using ARMSE

Approach	User	Pheme	Joint
HPbase	6.18±65.78	170.11±280.08	80.79
HPregularization	1.70±40.93	167.88±271.28	60.79
HPconversation	5.02±63.86	85.43±154.86	71.39
HPdecomposition (1)	4.34±67.21	79.83±155.83	73.17
HPconnection	1.60±49.97	134.53±206.94	34.91
HPfeatures (1)	5.04±65.91	82.42±154.87	70.25

Table 3.6: Top user features determining infectivity and susceptibility on the Ottawa dataset

Infectivity		Susceptibility	
feature id	Value	feature id	Value
lnk_ratio	3.61	lnk_ratio	-2.26
rpl_tw_ratio	2.76	unq_ment	0.94
ment_ratio	1.97	ment_ratio	-0.79
pro_bgr	1.01	rpl_tw_ratio	-0.48
hash_ratio	0.91	hash_ratio	-0.31
pro_loc	0.82	pro_bgr	-0.22
unq_ment	-0.67	pro_loc	-0.11
pro_img	0.08	fav_cnt	0.08
hash_tok_rat	0.06	pro_img	-0.06
fav_cnt	0.03	hist_tw	0.02
hist_tw	0.01	hash_tok_rat	0.02
rt_ratio	0	rt_ratio	0

has an impact on infectivity, unlike on the Ferguson dataset. In addition to links ratio, the proportion of reply tweets affects infectivity while user mentions affect susceptibility. This aligns well with the observations on the Ferguson data.

3.2 Predicting pheme virality

This section reports experiments on predicting pheme virality. More specifically, we predict the future occurrences of tweets in each pheme until a time T and rank them in descending order, based on the number of predicted tweets. We also rank phemes according to the actual number of tweets observed until time T.

Both rankings are compared, based on the Spearman rank correlation coefficient (see Table 3.7). A correlation coefficient value of 1 indicates perfect correlation and a correlation value of 0 indicates no correlation.

Table 3.7: Predicting pheme popularity, measured using the Spearman rank correlation coefficient on the three datasets

Approach	Synthetic	Ferguson	Ottawa
HPbase	0.9	0.45	0.65
HPregularization	0.9	0.52	0.66
HPconversation	0.9	0.70	0.68
HPdecomposition(1)	1.0	0.86	0.69
HPconnection	N/A	0.66	0.68
HPfeatures (1)	N/A	0.74	0.63

We observe that on the synthetic dataset, all approaches do well in predicting popular phemes. Here, number of phemes is small and thus the task is not quite as challenging as the two real world data sets.

On both Ferguson and Ottawa datasets *HPdecomposition* is the best at predicting pheme popularity. There is not much difference in the scores of the different approaches on the Ottawa data set.

3.3 Determining influential users

In order to determine which users are the most influential in spreading phemes, we sum up the columns of the $\boldsymbol{\alpha}$ influence matrix and rank users accordingly.

We consider a ranked list of users found influential by *HPconversation* and study their influence on the two rumour data sets (Ferguson and Ottawa). The ranked list of users obtained from the influence matrix is compared against a list of users ranked by number of followers and also against a second list, ordered by the number of replies they receive.

Table 3.8 shows the Spearman rank correlation coefficient between the two sets of ranked lists The values indicate there is no significant correlation between the influential users obtained from the influence matrix and those based number of followers or replies. When we consider the top 100 most influential users according to our matrix against those sorted by follower number, then there are 11 users in common in the Ferguson data, while none overlap in the Ottawa dataset. Likewise, when we consider the overlap between the top 100 users from the influence matrix against the reply list, there are 6 users in common in the Ferguson data and 2 users – in the Ottawa data.

Table 3.8: Spearman rank correlation coefficient on rumour data sets for predicting influential users

	Followers		Replies	
Approach	Ferguson	Ottawa	Ferguson	Ottawa
HPconversation	-0.0001	0.0069	-0.0035	-0.0060

Chapter 4

Detecting Untrustworthy Users

The viral spread of phemes through social media is often being exploited by a minority of untrustworthy users to spread spam. This chapter discusses features and algorithms for detecting such spam users automatically.

Spam users and their tweets need to be filtered out, in order to reduce the overhead in following fast spreading phemes with limited resources (e.g. by journalists in a newsroom). Spammers typically try to disguise spam messages by augmenting them with trending hashtags and by hiding the actual domain names in the URLs through URL shortening.

Many approaches have been developed to identify spam users automatically (Benevenuto et al., 2010; Lee et al., 2010; Yang et al., 2011; Lee et al., 2011; Amleshwaram et al., 2013; Meda et al., 2014). Typically they use a set of user-related features and a machine learning (classification) algorithm to classify users into spam and non-spam. This chapter investigates how automatic spam detection applies on the PHEME rumour data.

4.1 Spam Detection

We follow the standard approach of spam user classification by extracting user features and training a classification model. We consider both user behaviour features and content features (see Table 4.1 for details). These features were found to be the best in discriminating spam users in previous studies (Benevenuto et al., 2010; Lee et al., 2010; Meda et al., 2014).

4.1.1 Content features

Content features are derived from the user's tweets in the dataset. In line with prior work, we consider the following four features: 1. fraction of tweets with spam words

Table 4.1: Features used for spam user classification

Feature id	Kind	Description
1	user	fraction of followers to followees
2	content	fraction of tweets with spam words
3	content	fraction of tweets with URLs
4	content	average number of hashtags per tweet
5	content	average number of URLs per tweet
6	user	number of followees
7	user	number of followers
8	user	number of tweets, for which the user received a reply
9	user	number of tweets user replied
10	user	age of user account

2. fraction of tweets with URLs 3. average number of URLs per tweet 4. average number of hashtags per tweet

The first content feature is the fraction of tweets from the user, which contain blacklisted spam words. For the latter, we use the regularly updated Wordpress list of spam words [1].

URLs are the primary target of spammers, since Twitter shortens URLs and hence it is hard for Twitter users to distinguish a spam URL from a legitimate URL. Most spam tweets thus contain a URL leading an untrustworthy web site. Therefore the URL plays an important role in distinguishing spam users from legitimate users. We consider two features based on URL, the fraction of tweets from a user with URL and the average number of URLs per tweet from a user. Higher value of these two features tend can be indicative of spam users.

Lastly, in order to draw the widest attention possible, spammers typically post tweets with a large number of trending hashtags. This motivates the last feature – average number of hashtags per tweet for a user.

4.1.2 User behaviour features

User behaviour features characterise users based on their social interactions and social network. We consider six such features: 1. fraction of followers to followees 2. number of followees 3. number of followers 4. number of tweet user receives a reply 5. number of tweets user replied 6. age of user account

Spam user accounts get suspended by Twitter frequently and thus spam user accounts tend to be quite recent. Spam users also typically employ a link farming strategy to get more followers so that they spread spam faster (Ghosh et al., 2012).

[1] Word press comment blacklist. https://github.com/splorp/wordpress-comment-blacklist/

Essentially, they follow a large number of users, hoping that users will follow them back by courtesy. Thus the number of followees, number of followers, and fraction of followers to followees are three good features, potentially indicative of spam vs legitimate users.

Lastly, spam users typically reply to a large number of tweets trying to spread their spam message. At the same time, legitimate users generally do not reply to spam tweets, which leads to a low number replies received by spam users.

4.1.3 Learning Algorithms

In line with prior work, we use two supervised machine learning algorithms (logistic regression and Naive Bayes), with the features represented as vectors of values. Both logistic regression and Naive Bayes are probabilistic classifiers, which associate a probability with the predictions. Having a predictive probability is useful in ranking users in the rumour datasets, according to their probability of being spam users. The parameters of the models are learnt from a training data set, where users are classified as either spammers or non-spammers.

4.2 Data

The supervised learning models are trained on two standard Twitter spam data sets, where users are labelled as spam and non-spam: the MPI data and the social honey pot data.

4.2.1 MPI data

The MPI dataset consists of 1065 users, with 355 users labelled as spammers and the rest – as non-spammers (Benevenuto et al., 2010). The dataset was collected in 2009 by considering trending topics such as 'Michael Jacksons death', 'Susan Boyle's emergence' and 'music Monday'. Spammers disguised their tweets by including hashtags describing trending topics to reach a wider audience. Users were selected for annotation, if their tweets contained trending topic hashtags and with at least one URL. These users were annotated manually as spammer or non-spammer by human volunteers.

The features listed in Table 4.1 are seen to have good discriminative power on the MPI dataset. Figure 4.1 shows the distribution of feature values with respect to the spammer and non-spammer classes. For example, Feature 2 (the fraction of spam words in a tweet) has a value close to 0 for all legitimate users and a value close to 1 for most of the spam users.

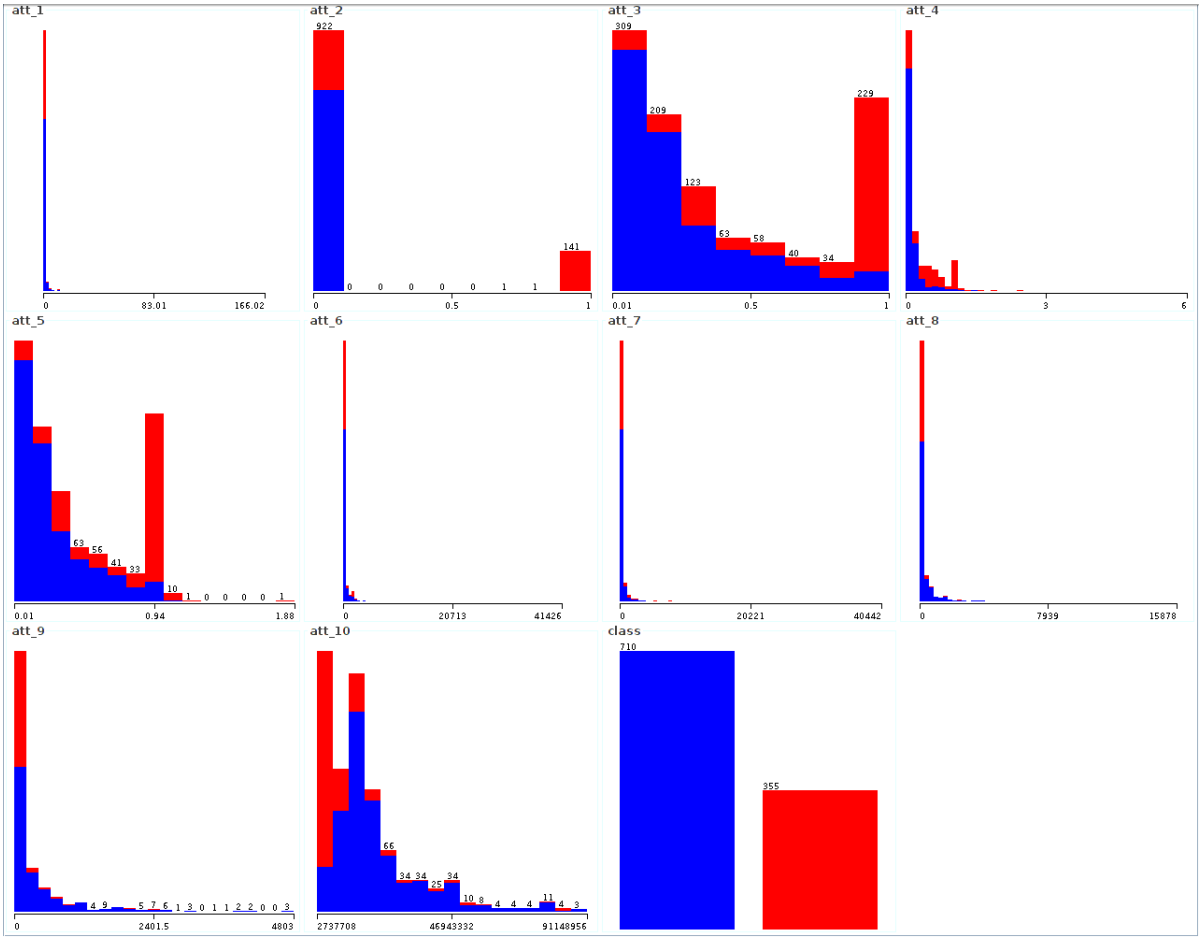

Figure 4.1: Distribution of attribute values for spammer and non-spammer classes. Blue represents non-spammers and Red represents spammers.

4.2.2 Social honey pot data

The social honey pot data was collected by studying the content polluters in Twitter using honey pot accounts (Lee et al., 2011). The data consists of tweets of content polluters (spammers) and legitimate users collected in the period December, 2009 to August, 2010.

The data contains 22,223 content polluters and 19,276 legitimate users. The content polluters are collected by selecting Twitter users who contacted more than one among 60 honey pot accounts deployed in Twitter. They found that Twitter eventually suspended 23% of the identified content polluters, indicating they are spam users. Clustering analysis further confirmed the spamming behaviour of the content polluters. The legitimate users are obtained by monitoring user accounts for a few months and randomly sampling users not suspended by Twitter. The

features listed in Table 4.1 are extracted from the data set, considering the network information of users and and their tweets.

4.3 Experimental Results

We train a logistic regression and naive Bayes classifiers [2] on the features extracted from the annotated datasets and study the predictive performance of the classifiers on the two traditional spam datasets, as well as on the two PHEME datasets (Ferguson and Ottawa) introduced in Chapter 3. We first study the predictive performance of the classifiers on the two traditional spam datasets.

4.3.1 MPI data

The predictive performance of the classifiers on the MPI dataset is evaluated using 5-fold cross-validation. Table 4.2 shows the precision, recall and f-score values of the classifiers on the MPI data.

We observe that logistic regression outperformed slightly Naive Bayes and also that performance of both approaches is better on the non-spam class, than on the spam class. In particular, the classifiers could recall only around 70% of the spam users, while they recalled more than 90% of the non-spam users.

Table 4.3 shows the confusion matrix for each of the two classifiers. Logistic regression was able to correctly classify comparatively more spam and non-spam users. Previous results on this data set (using Random Forest) reported an F-score of 0.92 for non-spammers and 0.82 for spammers (Meda et al., 2014). Our logistic regression approach performs similarly on non-spammers, while still needing a little further improvement on detecting spammers. Nevertheless, we chose logistic regression for its probabilistic output.

Table 4.2: Performance of logistic regression and Naive Bayes on MPI data.

	Logistic Regression			Naive Bayes		
Class	Precision	Recall	F-score	Precision	Recall	F-score
Non-spam	0.858	0.938	0.901	0.85	0.928	0.888
Spam	0.848	0.70	0.761	0.824	0.673	0.741
micro-average	0.855	0.855	0.851	0.842	0.843	0.839

[2] We use the Weka implementation of these classifiers with default parameters.

Table 4.3: Confusion matrix on the MPI data for logistic regression and Naive Bayes.

	Predicted class			
	logistic Regression		Naive Bayes	
Actual class	Non-spam	Spam	Non-spam	Spam
Non-spam	666	44	659	51
Spam	110	245	116	239

4.3.2 Social Honey pot Data

The predictive performance of the classifiers on the Social Honey pot data using 5-fold cross-validation is provided in Table 4.4 and Table 4.5.

Again, the logistic regression model is seen to perform much better than the Naive Bayes classifier, with good recall and accuracy on both spam and non-spam classes. Naive Bayes misclassified 17 – 19 % of users from each category, while logistic regression misclassified only around 10 – 12% of users.

Table 4.4: Performance of logistic regression and Naive Bayes on social honey pot data.

	Logistic Regression			Naive Bayes		
Class	Precision	Recall	F-score	Precision	Recall	F-score
Non-spam	0.89	0.898	0.894	0.837	0.839	0.838
Spam	0.905	0.897	0.901	0.85	0.849	0.85
micro-average	0.898	0.898	0.898	0.844	0.844	0.844

Table 4.5: Confusion matrix on the social honey pot data for logistic regression and Naive Bayes.

	Predicted class			
	logistic Regression		Naive Bayes	
Actual class	Non-spam	Spam	Non-spam	Spam
Non-spam	17268	1959	16128	3099
Spam	2128	18601	3133	17596

4.3.3 Rumour Data

The last experiment is investigating the presence of spam users in the Ferguson and Ottawa rumour data sets, using the classification algorithm learnt from the annotated training data sets. We considered the Ferguson rumour data with 3309

Table 4.6: Spam tweets of suspended users in Ferguson and Ottawa

Ferguson	
User id	Tweet
377440844	@Virtuous_QueenT https://t.co/BdIdMLkVgs @TheDailyEdge https://t.co/8VXgx4qJEl @Ian56789 https://t.co/8VXgx4qJEl
275331952	@ReutersUS http://t.co/Fmb6dDwOEL @ReutersUS http://t.co/KgeCZVWWLp
2727770608	@PrisonPlanet @LeeCF80 How about this http://t.co/cdEHPUe10G
Ottawa	
User id	Tweet
2796029406	@BBCNewsUS @BBCWorld #Islamic_State Message from Mujahid http://t.co/fq0T6xffRr
2598218616	@Reuters هههههه وربي حساب عجيب شوفوآ بالله @Question_arab
2252226043	@AP https://t.co/8jMhL2fC0O

users and Ottawa rumour data with 4168 users. The rumour data sets do not categorise already users into spammers vs non-spammers and hence, we cannot use these for training the models. This also makes a full-fledged evaluation on the rumour data sets not possible, without expensive human re-annotation.

Instead, here we carry out partial evaluation, by considering suspended users as spammers and also by analysing the tweets of the users which the logistic regression classifier marked as spammers with the highest probability.

In more detail, since the main reason for Twitter to suspend users is their spamming behaviour, first we investigated whether we can find such suspended users in the rumour data sets, by trying to obtain their profile information via the Twitter API. If it returns a message that the user is 'suspended', then that user is added to our suspended users list.

In this way we established that 49 users in the Ferguson dataset and 56 users in the Ottawa dataset were suspended by Twitter. Table 4.6 lists some suspended users and their tweets which were found to be spam.

In our experiments we treat those suspended users as suspected spam users and evaluate the performance of the logistic regression classifier on the rumour datasets. We chose logistic regression since it was consistently the better performing classifier.

First, we consider the performance of the logistic regression classifier trained on the MPI data set and tested on the rumour datasets. Table 4.7 provides the precision, recall, F-score measures of logistic regression on both Ferguson and Ottawa. Table 4.8 provides the confusion matrix for the same.

Unsurprisingly, it can classify the non-spam users with high precision and recall, while performance on the spam class (suspended users) is quite low. However, these

figures do not indicate that the model is not able to find spam users, since we ourselves found through manual inspection that only 3 users out the 49 suspended users in Ferguson and 8 users from the 56 in Ottawa showed spamming behaviour. Considering these figures, the logistic regression was able to identify 1 out of 3 spam users (0.33 recall) and 2 out of 8 spam users (0.25 recall). The logistic regression model identified 104 users in Ferguson and 147 users in the Ottawa dataset as spam.

Table 4.7: Performance of logistic regression trained on MPI data and tested on Ferguson and Ottawa rumour data

	Ferguson			Ottawa		
Class	Precision	Recall	F-score	Precision	Recall	F-score
Non-spam	0.985	0.968	0.977	0.987	0.965	0.976
Spam (suspended)	0.01	0.02	0.013	0.014	0.036	0.02
micro-average	0.971	0.954	0.962	0.973	0.952	0.963

Table 4.8: Confusion matrix obtained with logistic regression trained on MPI data and tested on Ferguson and Ottawa rumour data

	Predicted class			
	Ferguson		Ottawa	
Actual class	Non-spam	Spam	Non-spam	Spam
Non-spam	3157	103	3967	145
Spam (suspended)	48	1	54	2

Table 4.9 provides the precision, recall, F-score measures of logistic regression model trained on the social honey pot dataset and tested on the Ferguson and Ottawa tweets. Table 4.10 provides the confusion matrix for the same.

We find that the logistic regression model trained on the social honey pot data classified a large number of users as spam. For instance, it classified 2,59 users in Ferguson and 3,271 users in Ottawa as spam users, which leads to very poor precision. The model achieve higher recall on spam users at the expense of classifying legitimate users as spam. Therefore, the model learnt from the social honey pot data is not well suited to spam detection on rumourous threads, since the misclassification cost of classifying a legitimate user as spam is higher than that of classifying a spam user as a legitimate one. Besides, the social honey pot dataset is artificially balanced in terms of ratio of spam vs non-spam users in the training data, whereas spam users tend to be a minority in the rumorous threads in PHEME.

In our last experiment, we consider the logistic regression model trained on the MPI dataset and analyse the tweets of the top spam-classified users in the rumour datasets. This is possible, since logistic regression is a probabilistic classifier which

Table 4.9: Performance of logistic regression trained on the social honey pot data and tested on the Ferguson and Ottawa rumour data

	Ferguson			Ottawa		
Class	Precision	Recall	F-score	Precision	Recall	F-score
Non-spam	0.996	0.278	0.434	0.99	0.216	0.355
Spam (suspended)	0.019	0.918	0.037	0.014	0.839	0.028
micro-average	0.981	0.287	0.428	0.977	0.225	0.351

Table 4.10: Confusion matrix obtained with logistic regression trained on the social honey pot data and tested on the Ferguson and Ottawa rumour data

	Predicted class			
	Ferguson		Ottawa	
Actual class	Non-spam	Spam	Non-spam	Spam
Non-spam	905	2355	889	3223
Spam (suspended)	4	45	9	47

outputs the probability with which a user is classified as spam. We consider 100 users classified as spam by the logistic regression model, ordered by decreasing probability. Next we analysed their tweets to see if they showed any spamming behaviour.

Based on our manual inspection, in the Ferguson data we found only 2 spam users, while in the Ottawa data we found 5 spam users. Table 4.11 shows some example tweets from those users.

Table 4.11: Spam tweets of users found to be spam by logistic regression in Ferguson and Ottawa

Ferguson	
User id	Tweet
1359421328	#FergusonShooting #KillerCops #MichaelBrown https://t.co/m8ju5jvIiW
275331952	@ReutersUS http://t.co/Fmb6dDwOEL @ReutersUS http://t.co/KgeCZVWWLp
Ottawa	
User id	Tweet
2796029406	@BBCNewsUS @BBCWorld #Islamic_State Message from Mujahid http://t.co/fq0T6xffRr
1656323702	@cnnbrk #Networking #Crowdfunding #exposure + #hit = #millions http://t.co/Rh69mRtZMJ
2153254021	@cnnbrk company https://t.co/liBfXjcv6y #ROC #SmallBusiness #the585

Chapter 5

Conclusion

This deliverable addressed the challenges of modelling phemes, users, and their trustworthiness over time.

For longitudinal modelling of pheme virality and user susceptibility, we proposed an effective approach that combines social network and user features into a point process framework. We considered the Twitter conversational structure in a modified Hawkes process model, which has the computational advantage in that it avoids summation over all previous tweet occurrence times. The model was found to be useful in learning user influences.

An HP model which considered a non-negative matrix factorization of user influences to lower rank factors was found to be effective in improving the predictive performance. It reduces the number of parameters to learn and is especially useful in modelling social networks with a large number of users. We found that network information is crucial in learning user influence and that, in turn, it affects the predictive performance. Regularizing the influence matrix irrespective of the connection information can be detrimental.

Although user features were not found to be useful for improving predictive performance, they are really useful as features for learning the infectivity and susceptibility of users. For example, links ratio was found to be useful for determining user influence.

The second part of the deliverable focused on identifying spam users in phemes, where we found the logistic regression classifier to perform best. The balance of spam vs non-spam users in the training data was found to impact performance, with the widely used MPI spam user dataset being the best for training. Manual inspection of the top 100 suspected spam users in two pheme datasets found that, in general, the amount of spam tweets and spam users within was very low.

Work is ongoing on integrating these methods in Kafka, evaluating them with users, and also on refining them further based on those evaluation outcomes.

Bibliography

Agarwal, D., Chen, B.-C., and Elango, P. (2009). Spatio-temporal models for estimating click-through rate. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 21–30.

Amleshwaram, A. A., Reddy, A. L. N., Yadav, S., Gu, G., and Yang, C. (2013). Cats: Characterizing automation of twitter spammers. In *COMSNETS*, pages 1–10.

Bakshy, E., Hofman, J. M., Mason, W. A., and Watts, D. J. (2011). Everyone's an influencer: Quantifying influence on twitter. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 65–74.

Benevenuto, F., Magno, G., Rodrigues, T., and Almeida, V. (2010). Detecting spammers on twitter. In *Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS)*.

Cheng, J., Adamic, L., Dow, P. A., Kleinberg, J. M., and Leskovec, J. (2014). Can cascades be predicted? In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 925–936.

Du, N., Song, L., Yuan, M., and Smola, A. J. (2012). Learning networks of heterogeneous influence. In Pereira, F., Burges, C., Bottou, L., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 25*, pages 2780–2788.

Ghosh, S., Viswanath, B., Kooti, F., Sharma, N. K., Korlam, G., Benevenuto, F., Ganguly, N., and Gummadi, K. P. (2012). Understanding and combating link farming in the twitter social network. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 61–70.

Gomez-Rodriguez, M., Balduzzi, D., and Schölkopf, B. (2011). Uncovering the temporal dynamics of diffusion networks. In *Proceedings of the 28th International Conference on Machine Learning, ICML*, pages 561–568.

Gomez Rodriguez, M., Leskovec, J., and Schölkopf, B. (2013a). Modeling information propagation with survival theory. In *International Conference on Machine Learning*, pages 666–674.

Gomez Rodriguez, M., Leskovec, J., and Schölkopf, B. (2013b). Structure and dynamics of information pathways in online media. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 23–32.

Hong, L., Dan, O., and Davison, B. D. (2011). Predicting popular messages in twitter. In *Proceedings of the 20th International Conference Companion on World Wide Web*, WWW '11, pages 57–58.

Kingman, J. F. C. (1993). *Poisson Processes*, volume 3. Oxford University Press.

Kupavskii, A., Ostroumova, L., Umnov, A., Usachev, S., Serdyukov, P., Gusev, G., and Kustarev, A. (2012). Prediction of retweet cascade size over time. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 2335–2338.

Lampos, V., Aletras, N., Preotiuc-Pietro, D., and Cohn, T. (2014). Predicting and characterising user impact on twitter. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 405–413.

Lee, K., Caverlee, J., and Webb, S. (2010). Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 435–442.

Lee, K., Eoff, B. D., and Caverlee, J. (2011). Seven months with the devils: a long-term study of content polluters on twitter. In *In AAAI Int'l Conference on Weblogs and Social Media (ICWSM*.

Lee, S. H., Crawford, M. M., and Wilson, J. R. (1991). Modeling and simulation of a nonhomogeneous poisson process having cyclic behavior. *Communications in Statistics Simulation, 20(2):777–809*.

Lukasik, M., Cohn, T., and Bontcheva, K. (2015a). Point process modelling of rumour dynamics in social media. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015*, pages 518–523.

Lukasik, M., Srijith, P. K., Cohn, T., and Bontcheva, K. (2015b). Modeling tweet arrival times using log-gaussian cox processes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 250–255.

Meda, C., Bisio, F., Gastaldo, P., and Zunino, R. (2014). A machine learning approach for twitter spammers detection. In *2014 International Carnahan Conference on Security Technology (ICCST)*, pages 1–6.

Ogata, Y. (1981). On lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–30.

Perera, R. D., Anand, S., Subbalakshmi, K., and Chandramouli, R. (2010). Twitter analytics: Architecture, tools and analysis. In *Military Communications Conference, 2010-MILCOM 2010*, pages 2186–2191.

Suh, B., Hong, L., Pirolli, P., and Chi, E. H. (2010). Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *Proceedings of the 2010 IEEE Second International Conference on Social Computing*, pages 177–184.

Yang, C., Harkreader, R. C., and Gu, G. (2011). Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers. In *Proceedings of the 14th International Conference on Recent Advances in Intrusion Detection*, RAID'11, pages 318–337.

Yang, S.-H. and Zha, H. (2013). Mixture of mutually exciting processes for viral diffusion. In *International Conference on Machine Learning*, volume 28, pages 1–9.

Zhao, Q., Erdogdu, M. A., He, H. Y., Rajaraman, A., and Leskovec, J. (2015). Seismic: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1513–1522.

Zubiaga, A., Liakata, M., Procter, R., Wong Sak Hoi, G., and Tolmie, P. (2016). Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PLoS ONE*, 11(3):1–29.