Trabalho de Conclusão de Curso

Detecção de eventos em redes sociais com auxílio da transição de fase da entropia do bigrama

Pedro Henrique Silva Souza Barros
pedro_h_nr@laccan.ufal.br

Orientador:

Prof. Dr. Heitor Soares Ramos Filho

Maceió, 16 de Outubro de 2018

Pedro Henrique Silva Souza Barros

Detecção de eventos em redes sociais com auxílio da transição de fase da entropia do bigrama

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Engenharia de Computação do Instituto de Computação da Universidade Federal de Alagoas.

Orientador:

Prof. Dr. Heitor Soares Ramos Filho

Maceió, 16 de Outubro de 2018

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Engenharia de Computação do Instituto de Computação da Universidade Federal de Alagoas, aprovada pela comissão examinadora que abaixo assina.

Prof. Dr. Heitor Soares Ramos Filho - Orientador
Instituto de Computação
Universidade Federal de Alagoas

Prof. Dr. André Luiz Lins de Aquino - Examinador
Instituto de Computação
Universidade Federal de Alagoas

Prof. Dr. Aydano Pamponet Machado - Examinador
Instituto de Computação
Universidade Federal de Alagoas

Maceió, 16 de Outubro de 2018

Agradecimentos

A Deus, por todas as bençãos diárias;

A minha família, em especial aos meus pais, Rafael e Vera, e meu irmão, João, por todo o esforço para que eu possa ter uma boa educação, além da dedicação e paciência comigo ;

A todos os meus amigos que, apesar das batalhas da vida adulta, estão sempre comigo, me apoiando e me ajudando a crescer, de uma forma ou de outra;

A todos os membros do LaCCAN, por terem me aberto o mundo científico e me ensinarem o quão interessante pode ser a pesquisa.

> *"... my hand was made strong by the hand of the Almighty. We forward in this generation, Triumphantly."*
>
> – Marley, *Bob*

Resumo

Este trabalho propõem um novo método para detecção de eventos no *Twitter* baseado no calculo da entropia do conteúdo dos *tweets* a fim de classificar se um determinado tópico e um evento ou não. Nós observamos que a entropia dos bigram extraídos sugere que ocorre uma transição de fase continua quando os usuários da rede social começam a reagir e interagir com um evento. Logo propomos um método para detectar transições de fase e ,consequentemente, detectar um evento. Nós comparamos o desempenho do nosso método com outras abordagem da literatura e observamos que nossa proposta apresenta resultados promissores.

Palavras-chave: Detecção de eventos, cidades inteligentes, analise em redes sociais, teoria da Informação, transições de fase

Abstract

This work proposes a novel method to detect events in Twitter based on the calculation of entropy of the content of tweets in order to classify the most shared topic as an event or not. We observed that the entropy of the n-grams extracted from tweets are subject to a continuous phase transition when social media users start to react and interact with an event that is taking place. Hence, we propose a method to detect this phase transition, and consequently detect an event, and extract the keywords related to the corresponding event. We compared the performance of our method to other approaches of the literature and we observed that our method is the more regular among three metrics and reached the best overall performance. Furthermore, we present evidence that our method is very sensitive to correctly detect events that occur almost at same time.

Keywords: Event detection, smart cities, social media analysis, information-theoretic metrics, phase transition

Sumário

Lista de Figuras

Lista de Tabelas

1

Introdução

O crescimento das mídias sociais (como fóruns, blogs e microblogs) mudou a forma como usamos a Internet, possibilitando que as pessoas estejam conectadas, mesmo que não estejam geograficamente próximas. Além disso, agora temos um enorme volume de dados gerados por usuários comuns, convertendo a Internet em uma valiosa fonte de dados que é útil para monitorar aspectos sociais, econômicos, políticos, entre outras atividades.

1.1 Motivação

Dou et al. (2012) define um evento como " uma ocorrência que causa mudanças no volume de dados de texto que discutem o tópico associado em um momento específico ". Essa ocorrência é caracterizada por tópicos associados a tempos que são frequentemente associadas a entidades como pessoas e local. Essa associação pode ser explícita por palavras-chave, também chamadas de tópicos emergentes.

Portanto, as redes sociais podem ser vistas como uma valiosa fonte de dados útil para entender, através do uso de palavras-chave em um intervalo de tempo, uma ampla gama de eventos acontecendo ao redor do mundo, com cada usuário sendo um colaborador em potencial, como podemos ver alguns exemplos na figura 1.1.

O Twitter é uma das redes sociais mais populares atualmente, na qual os usuários interagem através de micro-mensagens, de até 280 caracteres. Com 100 milhões de usuários ativos diários, o Twitter têm cerca de 6.000 *tweets* por segundo, o que corresponde a mais de 500 milhões de *tweets* por dia [1].

Porém, apesar de seu valor como fonte de informação quando comparado a notícias da mídia tradicional, como artigos de notícias, a análise do Twitter apresenta alguns desafios, dos quais podemos mencionar:

[1] https://www.omnicoreagency.com/twitter-statistics/

Figura 1.1: Alguns tipos de eventos

(i) os *tweets* geram uma grande quantidade de dados que exigem uma abordagem escalonável para analisar seu conteúdo;

(ii) os *tweets* são sensíveis ao tempo, ou seja, são compartilhados em tempo real, tendo grande relação com o tempo em que são publicados;

(iii) devido a sua restrição de comprimento, um *tweet* apresenta tipicamente um conteúdo que é breve e informal, contendo frases não estruturadas, erros de digitação, abreviaturas, etc;

(iv) nem todos os *tweets* apresentam informações úteis. De fato, de acordo com Parikh and Karlapalem (2013), " metade dos *tweets* são inúteis e não transmitem nenhuma informação valiosa ". Esses *tweets* estão relacionados principalmente a atualizações pessoais de usuários, spam e auto-promoção. O processamento desses *tweets* não apenas aumenta o tempo de processamento, mas também degrada a qualidade dos resultados.

Logo, dado essa problemática, este trabalho propõe um novo método para detecção de eventos no Twitter. A técnica proposta neste trabalho, utiliza aprendizagem adaptativa, que, a partir da mudança das frequências dos Twitter, consiga inferir se naquele momento está correndo um evento ou não.

1.2 Trabalhos Correlatos

Existem vários estudos com o objetivo de detectar e/ou sumarizar eventos nas mídias sociais, especialmente no Twitter. Vários métodos têm sido extensivamente empregados para extrair informações e, portanto, identificar eventos.

Sakaki et al. (2010) analisaram *tweets* com base em recursos como palavras-chave, número de palavras e o contexto dos *tweets*. Eles construíram um modelo probabilístico, considerando cada usuário do Twitter como um sensor e aplicando um filtro de Kalman (Evensen, 2003) juntamente com um filtragem de partículas (Nummiaro et al., 2003) para encontrar o centro e a trajetória do local do evento. Eles detectaram terremotos com alta probabilidade, alcançando

uma precisão de 96% para terremotos detectados pela *Japan Meteorological Agency* (JMA) com escala de intensidade sísmica de 3 ou mais.

Mathioudakis and Koudas (2010) propuseram o *TwitterMonitor*. Além de detectar eventos no Twitter em tempo real, eles forneceram uma análise significativa que sintetiza uma descrição precisa de cada tópico. Eles identificaram a tendência das palavras-chave e agruparam-nas de acordo com suas co-ocorrências, usando um algoritmo de extração de contexto baseado em Deerwester et al. (1990).

Cataldi et al. (2010) extraíram o conteúdo dos *tweets*, analisando sua energia, ou seja quanto é emergente, e definiram uma janela de tempo para determinar seu ciclo de vida. Para cada conteúdo coletado, eles mediram o grau de influência do usuário usando o conhecido algoritmo *PageRank* (Langville and Meyer, 2011) para analisar sua importância como uma fonte confiável.

Weng and Lee (2011) propuseram EDCoW (Detecção de Eventos com Agrupamento de Sinais Baseados em *Wavelet*), que constrói sinais para palavras individuais aplicando análise de sinais (Rosso et al., 2009) e *wavelets* nas palavras. As palavras triviais foram descartadas e os eventos foram detectados pelo agrupamento de sinais das palavras juntos com o particionamento do gráfico baseado em modularidade.

Li et al. (2012) apresentou o *Tweetvent*, uma abordagem que detecta segmentos de *tweets* em uma janela de tempo fixa e os agrupa em eventos candidatos usando uma variante do algoritmo Jarvis-Patrick (Jarvis and Patrick, 1973). Cada agrupamento é comparado aos artigos da Wikipédia para identificar o evento realista e os segmentos mais importantes para descrever os eventos identificados.

Parikh and Karlapalem (2013) detectaram eventos extraindo bigramas em um período de tempo, com o objetivo de identificar um aumento significativo em suas frequências. Eles agruparam as palavras-chave relacionadas ao mesmo evento com base nas métricas de similaridade.

Dang et al. (2016) propôs um método baseado em Redes Bayesianas Dinâmicas (Murphy and Russell, 2002). Eles criaram um modelo que usa as informações sobre o compartilhamento de *tweets* e o relacionamento de seguidores para detectar palavras-chave emergentes e agrupá-las com base em sua co-ocorrência usando DBSCAN (Ester et al., 1996), detectando tópicos emergentes.

Aiello et al. (2013) propõe uma técnica que utiliza uma métrica baseada na frequência inversa de frequência do documento (TF-IDF) do bigrama em um determinado tempo e cria uma classificação dos eventos mais prováveis. Eles relataram um bom desempenho para detectar eventos (tópicos) e palavras-chave, embora eles considerem como informações prévias que um determinado número de eventos está ocorrendo em um intervalo de tempo e fornecem uma classificação dos eventos mais prováveis com suas palavras-chave correspondentes.

Os estudos acima mencionados indicam que a detecção de eventos no Twitter usando tópicos emergentes é viável. O presente trabalho é inspirado no uso de vários métodos mencionados acima, como bigramas e suas probabilidades, mas aproveitamos o fato de que a entropia é

uma medida de quantidade de informação para modelar a ocorrência de um evento, e com isso, detectar a mudança da dinâmica do sistema.

A mudança na dinâmica da distribuição dos dados com o tempo pode ser observado de diferentes formas. Podendo ocorrer de modo abrupto (por exemplo quando trocamos um sensor por outro que possui uma calibração diferente) ou incremental, um sensor naturalmente tem sua precisão diminuída lentamente com o tempo.

Um dos desafios nesse tipo de abordagem é o fato de tentar diferenciar uma mudança de dinâmica do sistema de um *outlier* ou ruído. Este último caso pode ser advindo de um desvio aleatório ou anomalia (Gama et al., 2014), de modo que pode ser detectado por técnicas que consideram o sistema estacionário. No caso de mudança de dinâmica, o sistema não pode ser considerado estacionário e devemos tratar este caso utilizando técnicas adaptativas (Gama et al., 2014).

Logo, nossa abordagem consiste em identificar a mudança da dinâmica do sistema, considerando que o Twitter possui uma característica de mudança incremental, ou seja, quando ocorre um evento alguns usuários postam imediatamente nas redes sociais, enquanto outros demoram um pouco mais para realizar o *tweet*, ocasionando assim uma mudança lenta da dinâmica do sistema.

1.3 Objetivo

Este trabalho tem como objetivo avançar o estado da arte para os serviços baseados em dados de sensoriamento social na área de computação ubíqua aplicada ao contexto de cidades inteligentes. Sendo assim, objetiva-se a criação de técnicas de análise de dados coletados através de sensoriamento social. Propõe-se investigar a utilização de técnicas computacionais para análise de grande volume de dados coletados utilizando-se os serviços de cidades inteligentes.

Com base na observação de que a entropia dos bigramas extraídos do conteúdo de *tweets* está sujeita a uma transição de fase contínua, ou seja, uma mudança no modelo subjacente, este trabalho propõe um novo método para detectar eventos no *Twitter* baseado no cálculo da entropia das palavras-chave extraídas do conteúdo de *tweets* para classificar o tópico mais compartilhado como um evento ou não.

1.4 Contribuições

As contribuições deste trabalho são:

- Apresentação de uma nova proposta para detecção de eventos no *Twitter* considerando transições de fase, i. e., a dinâmica do sistema muda com o tempo;
- Uma comparação do desempenho de técnicas de detecção de eventos no *Twitter*;

- O avanço do estado-da-arte na detecção de eventos no *Twitter*.

1.5 Estrutura do texto

Este trabalho se divide como segue: Capítulo 2 apresenta o funcionamento da nova abordagem proposta por este trabalho; Capítulo 3 apresenta os materiais utilizados neste trabalho, bem como os resultados obtidos; e, finalmente, o Capítulo 4 apresenta as considerações finais, concluindo este trabalho.

2

Nossa proposta

Propusemos um método para identificar eventos em tempo real no Twitter, detectando os tópicos emergentes apresentados em um determinado momento. Nosso método é baseado no cálculo da entropia da série temporal que agrega a probabilidade das palavras-chave extraídas dos *tweets*. Esta proposta pode ser usada para detectar qualquer evento, porque não consideramos nenhuma informação prévia sobre o evento.

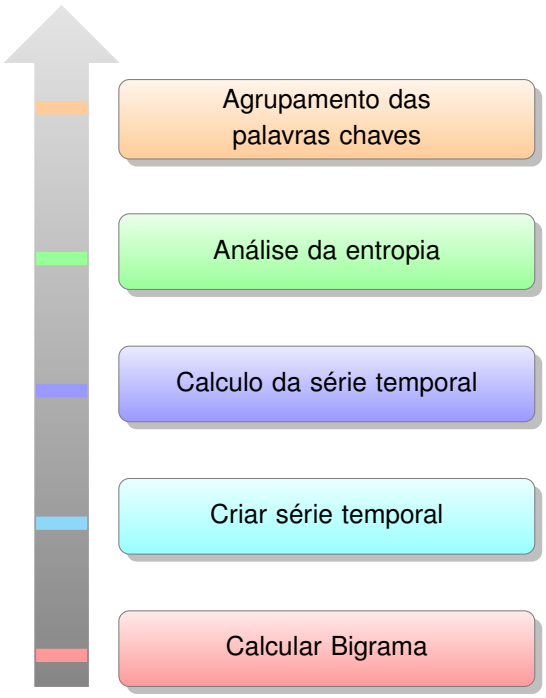


Figura 2.1: Passo a passo da proposta

Nossa abordagem consiste em cinco etapas, como podemos ver na figura 2.1:

(i) calcula o bigrama[1] relacionado ao conteúdo dos *tweets*, capturando as palavras-chave sobre os tópicos mais comentados entre os usuários;

[1] https://en.wikipedia.org/wiki/Bigram

Bigrama	Valor
game-soccer	4
sunday-game	4
fair-play	4
twitter-now	3
match-today	3
half-time	3
now-win	3
competing-teams	2
soccer-great	2
final-match	2
spirit-team	2
start-match	1

Tabela 2.1: Exemplo de alguns bigramas por unidade de tempo.

Bigrama	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7
start-match	1	3	4	2	40	47	62	79

Tabela 2.2: Série temporal para o bigrama **start-match**.

(ii) cria uma série temporal que possui a probabilidade de cada bigrama calculado;

(iii) calcula a entropia da série temporal de probabilidade;

(iv) analisa o valor da entropia, a fim de classificar o tópico como evento ou não; e

(v) agrupamento das palavras chaves a fim de caracterizar um evento;

2.1 Cálculo dos Bigramas

Seja $f\colon \boldsymbol{B} \rightarrow \boldsymbol{X}$ a função que mapeia o bigrama $b \in \boldsymbol{B}$ no conjunto de contagem $\boldsymbol{X}$ para todos os intervalos de tempo no conjunto de dados. $\boldsymbol{B}$ é o conjunto de todos os bigramas. A função f é dada por $f(b) = \{x_k \in \boldsymbol{X} \mid x_k = \#b_k\}$, $\forall k \in \{t_0, t_1, \ldots, t_N\}$, onde k é o conjunto de todos os intervalos de tempo em um conjunto de observações de tamanho N, e $\#b_k$ é o número de ocorrências do bigrama b a cada vez k no conjunto de dados. Podemos ver um exemplo na tabela 2.1 para um valor de tempo t_0.

2.2 Criação das séries temporais

À medida que o método é desenvolvido para processamento em tempo real, primeiro coletamos um conjunto de observações dentro de uma janela e depois avançamos pela janela para analisar mais dados. Portanto, primeiro determinamos $\boldsymbol{B}^W$, $\boldsymbol{X}^W$ e $f^W(b)$ para a primeira janela e continuamos a determinar esses conjuntos para as janelas subsequentes.

A janela é denotada por $W^{i:n} = \{\boldsymbol{X}_p\}$, onde $p \in \{t_i, t_{i+1}, \ldots, t_{i+n-1}\}$ é um intervalo de tempo dentro de W, t_i é o tempo inicial e n é o número de elementos de W, como podemos ver na

Tabela 2.3: Abordagem da janela deslizante.

Bigrama - (start, match)								
Tempo	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7
Contagem	1	3	4	2	40	47	62	79
Contagem ($W^{0:3}$)	1	3	4	2	-	-	-	-
Contagem ($W^{1:4}$)	-	3	4	2	40	-	-	-
Contagem ($W^{2:5}$)	-	-	4	2	40	47	-	-
Contagem ($W^{3:6}$)	-	-	-	2	40	47	62	-
Contagem ($W^{4:7}$)	-	-	-	-	40	47	62	79

tabela 2.2, além do esquemático da figura 2.2, um exemplo para $f(b)$ com o bigrama $b =$ start-match. $\boldsymbol{X}_p$ denota o conjunto de contagens para todos os bigramas no momento p. A janela, com tamanho n, desliza uma unidade de tempo, de forma que a diferença entre janelas consecutivas é exatamente uma observação, como podemos ver na tabela 2.3, para uma janela de tamanho $n = 4$. Observe que essa janela consideramos apenas a contagem de um bigrama específico.

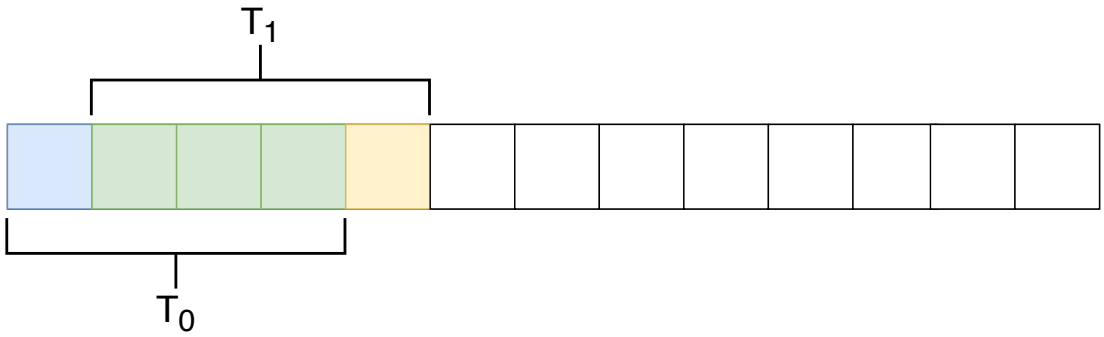


Figura 2.2: Esquemático da janela deslizante com 4 elementos.

Além disso, para cada bigrama obtido, calculamos a probabilidade de sua ocorrência em uma dada janela deslizante. Em seguida, construímos a série temporal de frequências $\boldsymbol{S}^W(b) = \{s_p(b)\}$ restrito a uma janela W como

$$s_p(b) = \frac{x_p(b)}{\sum_{j \in p} x_j(b)}, \tag{2.1}$$

onde $s_p(b)$ representa uma observação das séries temporais de frequências associadas a cada bigrama na janela correspondente no tempo p. A equação (2.1) converte o vetor de contagens $\boldsymbol{X}^W \subset \boldsymbol{X}$, o conjunto de contagens restritas a W, em um vetor de frequências $\boldsymbol{S}^W$, como podemos ver na tabela 2.4.

Tabela 2.4: Abordagem deslizante da frequência

Bigrama - (start, match)								
Tempo	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7
Contagem	1	3	4	2	40	47	62	79
Frequência ($W^{0:3}$)	1/10	3/10	4/10	2/10	-	-	-	-
Frequência ($W^{1:4}$)	-	3/49	4/49	2/49	40/49	-	-	-
Frequência ($W^{2:5}$)	-	-	4/93	2/93	40/93	47/93	-	-
Frequência ($W^{3:6}$)	-	-	-	2/151	40/151	47/151	62/151	-
Frequência ($W^{4:7}$)	-	-	-	-	40/228	47/228	62/228	79/228

Tabela 2.5: Abordagem deslizante da entropia

Bigrama - (start, match)								
Time	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7
Contagem	1	3	4	2	40	47	62	79
Entropia ($W^{0:3}$)	-	-	-	0.28	-	-	-	-
Entropia ($W^{1:4}$)	-	-	-	-	0.67	-	-	-
Entropia ($W^{2:5}$)	-	-	-	-	-	0.93	-	-
Entropia ($W^{3:6}$)	-	-	-	-	-	-	1.14	-
Entropia ($W^{4:7}$)	-	-	-	-	-	-	-	1.35

2.3 Cálculo da entropia

Para a série temporal $\boldsymbol{S}^W(b)$, consideramos $s_p(b)$ como a estimativa da probabilidade da ocorrência do bigrama b em cada intervalo de tempo p em W, e calcular a entropia de Shannon para cada W como

$$H^W\left(\boldsymbol{S}^W(b)\right) = \sum_{j \in p} -s_j(b) \log\left(s_j(b)\right). \tag{2.2}$$

Podemos ver na tabela 2.5 essa abordagem em funcionamento.

2.4 Análise da Entropia

A entropia $H^W(\boldsymbol{S}^W(b))$ fornece uma medida da quantidade de informações em cada janela W. Portanto, janelas que apresentam altas entropias provavelmente representam uma ocorrência de um evento. Com base no valor de entropia, podemos inferir se uma janela apresenta um comportamento de anomalia, ou seja, um evento, para um determinado bigrama b. Em tal situação, estamos interessados em detectar uma transição de fase entre uma janela que não apresenta um evento para uma janela que apresenta um evento. Na seção 3, discutimos como detectamos a transição de fase no contexto deste trabalho.

Se um evento for detectado na janela deslizante W, o evento é atribuído ao tempo t_{i+n-1}, a última observação em W.

2.5 Pseudo-código da proposta

Algoritmo 2.1 Nossa proposta para detecção de eventos nas redes sociais baseada na transição de fase

1: **for** $i \leftarrow i_0$ **to** $i_0 + n - 1$ **do**
2: inicializa B^W
3: **for** $b \in \boldsymbol{B}^W$ **do**
4: **for** $p \in \{t_i, t_{i+1}, \ldots, t_{i+n-1}\}$ **do** ▷ Construindo $\boldsymbol{X}^W$
5: $x_p = f^W(b_P)$
6: **end for**
7: **end for**
8: SORT($\boldsymbol{B}^W$, $type$ = 'decrescente', by='x_p')
9: **for** $b \in \boldsymbol{B}^W_{0:49}$ **do** ▷ Para os primeiros 50 bigramas em $\mathbf{B}^W$
10: **for** $p \in \{t_i, t_{i+1}, \ldots, t_{i+n-1}\}$ **do**
11: $s_p = \frac{x_p}{\sum_{j \in p} x_j}$ ▷ Calculando o vetor de probabilidade $S^W(b)$
12: **end for**
13: $H^W = 0$
14: **for** $p \in \{t_i, t_{i+1}, \ldots, t_{i+n-1}\}$ **do**
15: $H^W = H^W + (-s_p \log(s_p))$ ▷ Calculando a entropia da janela W
16: **end for**
17: **if** $(0.1 \leq H^W \leq 0.7) \wedge (\boldsymbol{x}_i \geq 10) \wedge (\text{MAX}(\boldsymbol{S}^W) == s_{i+n-1})$ **then**
18: SAVEEVENT(b, H^W)
19: **end if**
20: **end for**
21: **end for**

Algoritmo em 2.1 apresenta um pseudocódigo da nossa proposta. O algoritmo calcula a entropia de uma janela e decide se essa janela pertence a uma transição de fase entre a ausência e a presença de um evento. Para fazer isso, o algoritmo recebe como entrada o índice do tempo inicial (i_0) de W e tem acesso aos *tweets* correspondentes a cada intervalo de tempo e retorna uma lista de palavras-chave prováveis para cada evento detectado em W. Primeiramente, ele inicializa o conjunto $\boldsymbol{B}^W$ (Linha 2) extraindo todos os bigramas em W.

Para cada unidade de tempo, ele calcula as contagens de bigramas em W e classifica em ordem decrescente (Linhas 3 - 7 e 8). Nas Linhas 9 até 20, ele percorre $\boldsymbol{B}$ para calcular o vetor de probabilidade e a entropia de W. Limitamos os bigramas (mais frequentes) mais representativos de 50 unidade para descartar os dados irrelevantes. Para cada bigrama em $\boldsymbol{B}^W_{0:49}$, ele constrói o conjunto da probabilidade (frequência) do bigrama em cada intervalo de tempo em W (Linhas 10 até 12) e calcula a entropia subsequente da janela nas linhas 14 até 16.

Nas linhas 17 até 19, ele detecta se uma janela apresenta uma transição para um evento ou não e, em caso de detecção, salva o evento. Escolhemos o intervalo de detecção de entropia de $0,1 \leq H^W \leq 0,7$ para detectar o evento analisando a curva ROC do nosso sistema de detecção, conforme explicado em Seção 3. A regra $\max(\boldsymbol{S}^W) == s_{i+n-1}$ garante que a entropia do último intervalo de tempo seja o maior valor, indicando a ocorrência de uma transição de fase.

Depois de detectar esses eventos, agrupamos todos os bigramas detectados da série temporal $\boldsymbol{S}^W(b)$ com base em seus termos em comum, criando um conjunto de palavras-chave relacionadas a cada evento.

O algoritmo é $O(n^2)$, já que o *loop* do for interno na linha 9 executa um número fixo de vezes (50), e o tamanho da janela é tipicamente pequeno, por exemplo, 15. A carga do algoritmo é principalmente para processar os *tweets* (palavras inúteis removidos (como preposições e pronomes), sinais de pontuação e URLs.

2.6 Clusterização das palavras-chaves

2.6.1 Notação e Definição

Esta seção introduz a terminologia necessária para grafos. Vemos um exemplo de grafo na figura 2.3.

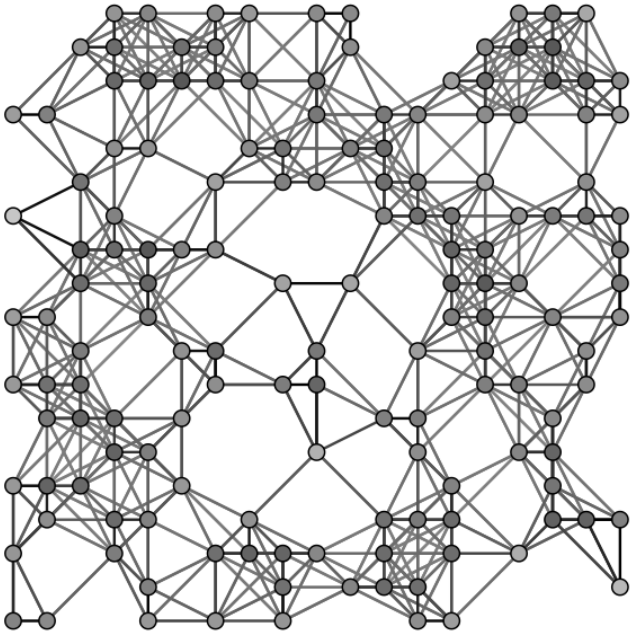


Figura 2.3: Exemplo de um grafo G_0.

Proposição 2.6.1 *Seja V um conjunto finito de coleção de elementos, enumerados da forma $v_1, v_2, v_3, \cdots, v_t$.*

1. Um grafo valorado é um grafo em que cada aresta tem um valor associado. Formalmente, um grafo valorado G $= (V,E)$ consiste de um conjunto V de vértices, um conjunto E de arestas, e uma função w de E para P, onde P representa o conjunto de valores (pesos) associados às arestas. Com isso temos que $w : E \to P$, onde $P \subseteq \mathbb{R}^+$. Seja G $= (V,E)$,

definimos a matriz de peso M_G com elementos m_{pq} indicando o peso do vértice que associa os elementos v_p e v_q.

(a) G é chamado de **não-orientado** se M é simétrico, ou seja, se $m_{ij} = m_{ji}$, $\forall i, j \in V$. Chamamos de **orientado** caso contrário.

(b) G é dito ser **inflexível** se não existem *loops*, isto é, se $m_{ii} = 0$, $\forall i \in V$.

2.6.2 Algoritmo de Clusterização de Markov

O *Markov Cluster Process* (MCL), proposto por Van Dongen (2000) define uma sequência de processos estocástico matriciais, chamados de operadores, que consiste basicamente na alternância de duas operações (inflação e expansão).

Dado um vértice com várias arestas conectadas ao mesmo, ou seja, um nó que possuí o número de arestas bem maior que a média do grafo, um aglomerado denso consiste na região do grafo que contêm um (ou mais) desses vértices.

O paradigma de agrupamento de grafos postula que grupos em grafos têm a seguinte propriedade:

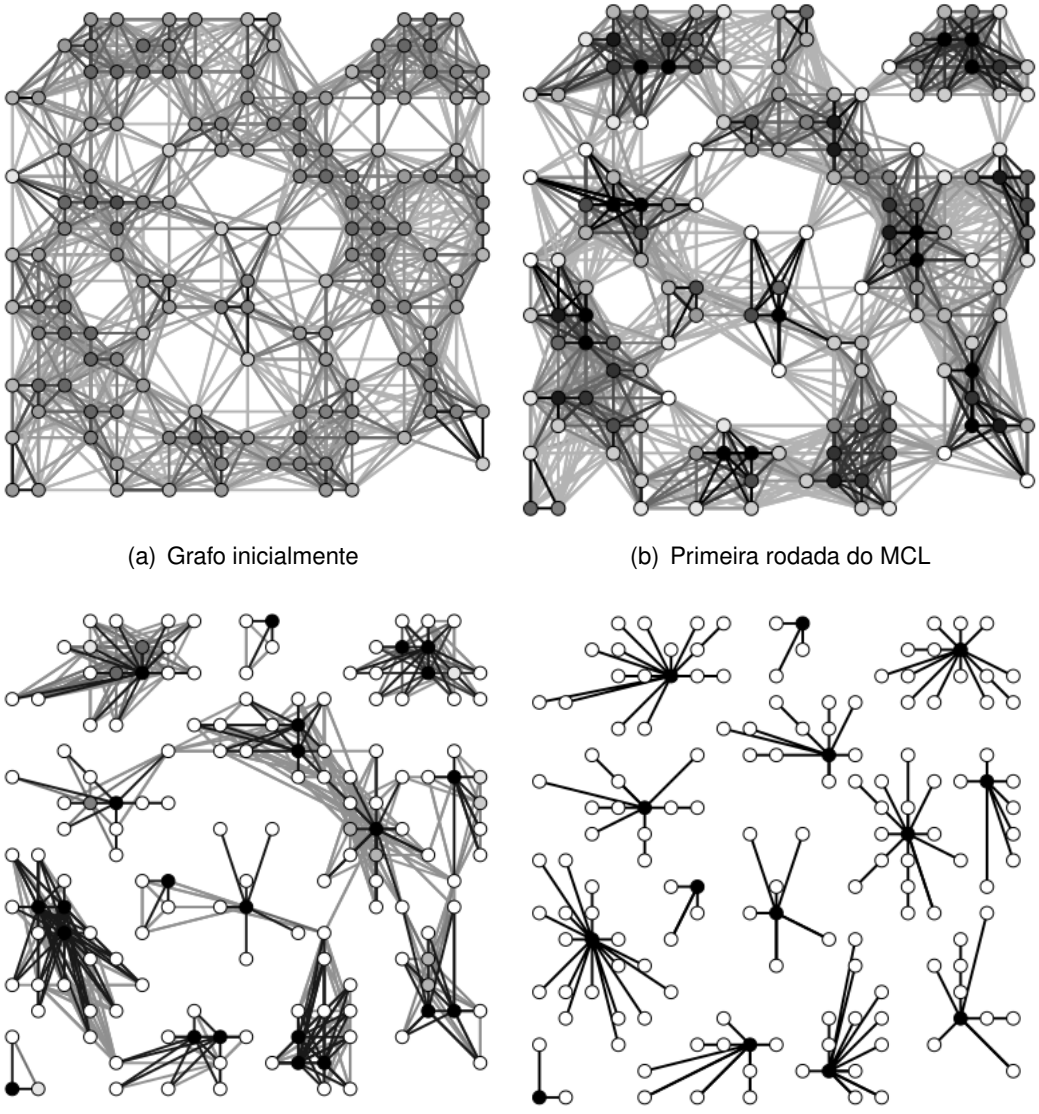
Conjectura 2.6.2 *Um passeio aleatório em um grafo G que visita um aglomerado denso provavelmente não sairá dos seus vértices.*

No coração do algoritmo MCL está a ideia de simular o fluxo dentro de um grafo normalizado, aumentando o fluxo onde a corrente é forte (muitas visitas do caminhante aleatório a um determinado vértice) e baixando o fluxo onde a corrente é fraca (poucas visitas do caminhante aleatório a um determinado vértice). Se grupos estão presentes no grafo, então de acordo com a conjectura 2.6.2, as fronteiras entre os diferentes grupos irão desaparecer, revelando assim a estrutura do gráfico.

2.6.3 Operador de expansão

O operador de expansão privilegia os caminhos de menor comprimento, ou seja caminhadas aleatórias com poucos passos, favorecendo a visita a novos grupos. Este operador associa novas probabilidades a todos os pares de nós, diminuindo a probabilidade para caminhos longos e aumentando para caminhos curtos.

Como os caminhos de maior comprimento são mais comuns intra-grupos do que entre grupos diferentes, devido ao fato do caminhante aleatório ficar “preso” nos aglomerados densos, as probabilidades associadas a pares de nós localizados no mesmo grupo serão, em geral, relativamente grandes, pois há muitas maneiras de ir de um para outro. Esta é a razão do operador de expansão diminuir a probabilidade intra-grupo, favorecendo a visita a novos grupos. Com



(a) Grafo inicialmente

(b) Primeira rodada do MCL

(c) Segunda rodada do MCL

(d) Cluster finais do MCL

Figura 2.4: Sucessivos estágios da simulação do fluxo para o algoritmo do MCL para o grafo G_0.

isso, o operador de expansão é responsável por permitir que o fluxo conecte diferentes regiões do grafo. Para isso, o operador de expansão consiste na multiplicação de matrizes, definida por

$$E_a = \overbrace{M_G \times M_G \times \ldots \times M_G}^{a} = [M_G]^a,$$

onde a corresponde ao parâmetro da operação de expansão (número de vezes que a matriz M_G será multiplicada), E_a corresponde o operador de expansão e a Matriz M_G corresponde a matriz de peso associada ao grafo G. O resultado da potência da matriz M_G com expoente a retorna para cada elemento m_{pq} da matriz resultante, a quantidade de caminhos de tamanho a entre os nós v_p e v_q. Neste caso, com matrizes normalizadas, esta operação calcula a probabilidade de se alcançar o vértice v_q a partir do vértice v_p através de caminhos de tamanho a.

2.6.4 Operador de Inflação

O paradigma é enriquecido com a inserção de um novo operador no processo de Markov, chamado inflação. Enquanto a expansão de fluxo é representada pelo produto matricial, a inflação é representada pelo produto de Hadamard–Schur [2] combinado com um dimensionamento diagonal (normalização da matriz).

O operador de inflação é responsável tanto pelo fortalecimento quanto pelo enfraquecimento do fluxo atual. A inflação terá então o efeito de aumentar as probabilidades de passeios intragrupos e rebaixará as caminhadas entre os grupos. Isto é conseguido sem qualquer conhecimento prévio da estrutura do agrupamento.

Definição 2.6.3 *Dada uma matriz $M \in \mathbb{R}^{VxV}$, temos que a matriz resultante do reescalonamento de cada uma das colunas de M com o coeficiente de potência r é chamado $\Gamma_r(M)$. O operador de inflação com coeficiente de potência r é chamado Γ_r. Formalmente, a ação de $\Gamma_r : \boldsymbol{R}^{VxV} \to \boldsymbol{R}^{VxV}$ é definida por*

$$[\Gamma_r(M)]_{pq} = \frac{(m_{pq})^r}{\sum_{i=1}^{k}(m_{iq})^r}$$

Podemos ver abaixo, um exemplo para o operador de inflação com $r = 2$ para matriz (linha 1) e o respectivo resultado (linha 2).

$$\text{Vetor } v: \begin{pmatrix} 0 & 0 & 1/4 & 0.151 & 0.086 \\ 3 & 1/2 & 1/4 & 0.159 & 0.000 \\ 0 & 0 & 1/4 & 0.218 & 0.113 \\ 1 & 1/6 & 1/4 & 0.225 & 0.801 \\ 2 & 1/3 & 0 & 0.247 & 0.000 \end{pmatrix}$$

[2] https://en.wikipedia.org/wiki/Hadamard_product_(matrices)

$$\Gamma_r(v): \begin{pmatrix} 0 & 0 & 1/4 & 0.110 & 0.011 \\ 9/14 & 9/14 & 1/4 & 0.122 & 0.000 \\ 0 & 0 & 1/4 & 0.229 & 0.019 \\ 1/14 & 1/14 & 1/4 & 0.245 & 0.970 \\ 4/14 & 4/14 & 0 & 0.295 & 0.000 \end{pmatrix}$$

2.6.5 O efeito da inflação na granularidade do agrupamento

Existe uma correlação entre o parâmetro de inflação e a granularidade dos grupos obtidos pelo algoritmo. Quanto maior o parâmetro r, mais o operador de inflação diminui o fluxo ao decorrer de longas distâncias no grafo.

A figura 2.5 apresenta o grafo G_1, onde vemos na figura 2.6 o resultado de seis rodadas diferentes do MCL para G_1. O parâmetro r da inflação, para esta figura, tem distintos valores entre 1.4 a 2.5, enquanto todos os outros parâmetros são mantidos iguais (ou seja,o operador de expansão foi mantido constante e $a = 2$).

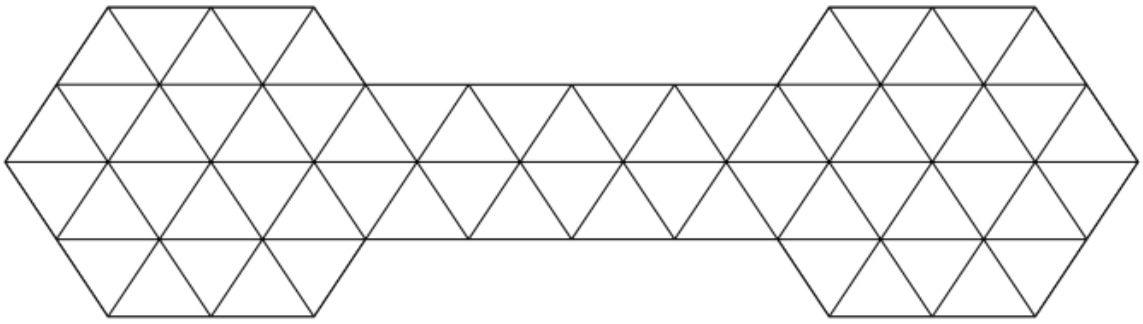
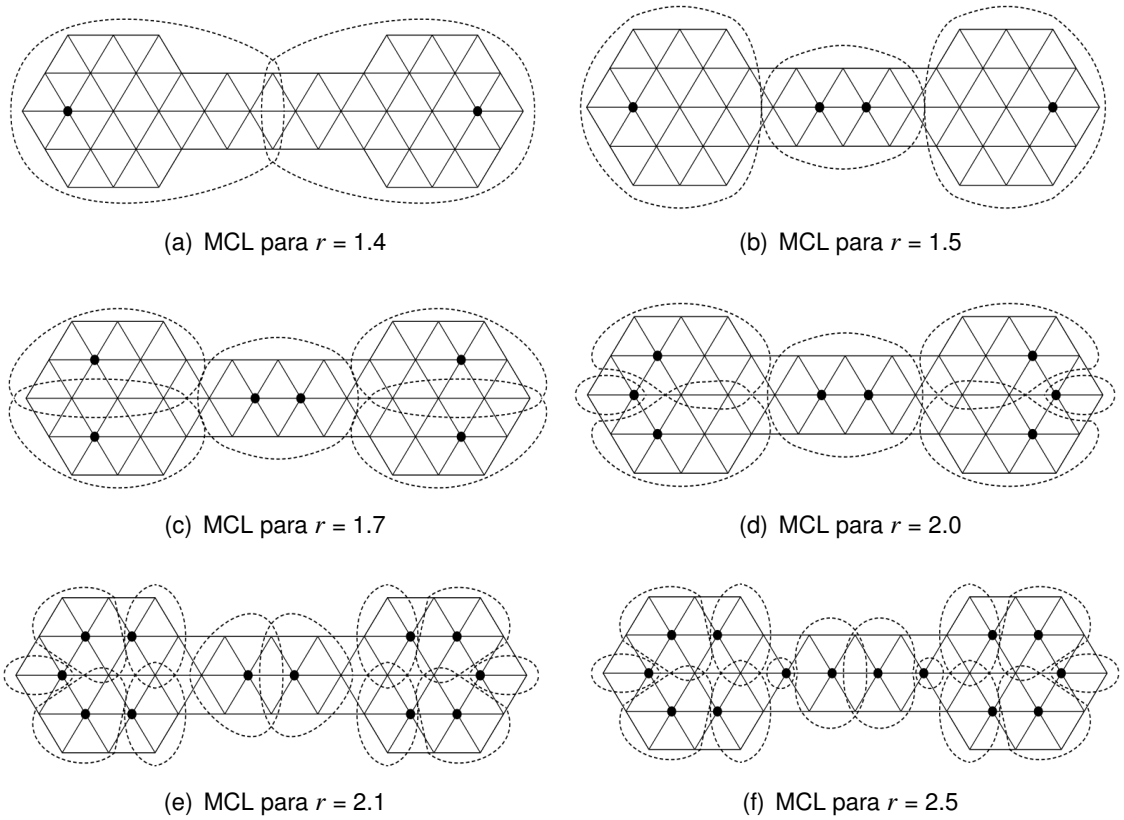


Figura 2.5: Exemplos típico de grafo G_1.

Observe que os agrupamentos possuem sobreposição, e estas estão fortemente relacionadas umas às outras. Pode-se perceber que o agrupamento para o valor de $r = 1.4$ possui sobreposição de todos os outros, ou seja, todos tem uma tendência de formação de subconjuntos desse agrupamento. Isso é muito satisfatório, pois espera-se que os agrupamentos em diferentes níveis de granularidade sejam relacionados uns aos outros. O agrupamentos nos três primeiros níveis $r = x$, onde $x \in \{1.4, 1.5, 1.7\}$ possuem tamanhos distribuídos uniformemente.

O algoritmo 2.2 apresenta o pseudo-código do MCL. O algoritmo executa operações de normalização, expansão e inflação, em sequência (linhas 5, 6, 7) e repete as operações de expansão (linha 10) e de inflação (linha 11) até que não haja mudanças na matriz (convergência).



(a) MCL para r = 1.4 (b) MCL para r = 1.5

(c) MCL para r = 1.7 (d) MCL para r = 2.0

(e) MCL para r = 2.1 (f) MCL para r = 2.5

Figura 2.6: Exemplos da variação de r para o operação de inflação para o grafo G_1.

Algoritmo 2.2 Pseudo-código para o MCL.

```
p ← Expoente da operação de Expansão
r ← Expoente da operação de Inflação
m ← Matriz de adjacência m
function MCL(m, p, r)
    m = Normalizar_Matriz(m)
    m = Expansão(m, p)
    m = Inflação(m, r)
    while m ≠ m_ant do
        m_ant = m
        m = Expansão(m, p)
        m = Inflação(m, r)
    end while
    return m
end function
```

3

Resultados e Discussões

Nesta sessão apresentamos os resultados do trabalho, bem como as discussões e comentários sobre o mesmo. A distribuição de Poisson é o modelo probabilístico mais comum para detectar eventos anômalos e independentes em uma unidade especificada de espaço ou tempo. Devido a esta característica, utilizamos o mesmo como *baseline* para comparação com o método proposto neste trabalho.

Em nossas análises, encontramos fortes indícios de que o sistema sofre transições de fase contínua, além de, caracterizar essas transições através dos expoentes críticos. Por isto, propusemos a técnica de detecção de eventos baseada em transições de fases. No que segue, apresentaremos um estudo compativo entre essas duas abordagens.

3.1 Poisson

A distribuição de Poisson (Poisson, 1837) é uma distribuição de probabilidade discreta que expressa a probabilidade de um número de eventos que ocorrem em um período de tempo fixo, dado que esses eventos ocorrem com uma taxa média conhecida e independentemente do tempo desde o último evento (Katti and Rao, 1968).

3.1.1 Descrição do algoritmo

Consideramos a taxa de Poisson constante dentro de uma janela de tempo (Processo de Poisson homogêneo), e estimamos a taxa de Poisson $\hat{\lambda}$ de um determinado tempo como a média do número de *tweets* coletados dentro de uma janela deslizante $P^{i:n} = \{p_i, p_{i+1}, \ldots, p_{i+n-1}\}$, que corresponde à frequência de todos os *tweets* em um horário i. Nós usamos 15 unidades de um minuto para calcular a janela deslizante.

A taxa de Poisson $\hat{\lambda}$ é estimada como através do método de máxima verosimilhança por

$$\hat{\lambda} = \frac{1}{n} \sum_{j=i}^{i+n-1} p_j.$$

Um evento é detectado quando observamos que a probabilidade de $k = p_{i+n}$ *tweets* em $P^{i:n}$ é suficientemente menor que um limite ε dentro de uma determinada janela deslizante. Para isso, avaliamos a probabilidade de k em $P^{i:n}$ como $\Pr(k;\hat{\lambda}) = \hat{\lambda}^k e^{-\hat{\lambda}}/k!$, com $\hat{\lambda} \leq k$.

Nós detectamos um evento sempre que a distribuição de probabilidade em k for menor que um determinado limite ε. Isso significa que k provavelmente será uma observação rara, dada distribuição estimada de Poisson com média $\hat{\lambda}$.

Devido à alta sensibilidade apresentada pela distribuição de Poisson, ou seja, é exponencial em λ, consideramos $\varepsilon = 10^{-20}$ como um comportamento anômalo.

Se um evento for detectado na janela deslizante $P^{i:n}$, presumimos que o tempo $i+n$ é responsável pela anomalia, ou seja, a hora em que o evento ocorreu.

Depois de detectar os eventos, obtemos os bigramas mais frequentes, agrupando-os com base nos termos em comum, formando assim as palavras-chave relacionadas ao evento detectado.

3.2 Conjunto de dados

Para validar nossa proposta, usamos o conjunto de dados final da FA Cup[1], coletado por Aiello et al. (2013), onde os autores reuniram *tweets* relacionados a alguns importantes eventos mundiais que aconteceram em 2012.

Este conjunto de dados contém *tweets* sobre a Copa do campeonato Inglês de Futebol (FA Cup), o auge da temporada de futebol inglês. A FA Cup é a principal competição do futebol inglês e pertence à mais antiga associação de futebol do mundo.

Em 2012, o Chelsea e o Liverpool jogaram a partida final do FA Cup, com gols de Ramirez (11') e Drogba (52') para o Chelsea e Carrol (62') para o Liverpool. Assim, o Chelsea venceu a partida por $2-1$, onde a mesma durou 90 minutos, mais 15 minutos de intervalo.

Aiello et al. (2013) criou este conjunto de dados usando as *hashtags* oficiais do evento, os nomes das equipes e dos principais participantes no Twitter, coletando um total de 144.294 *tweets*.

Eles construíram os rótulos do conjunto de dados verificando os principais relatórios de imprensa especializada para identificar tópicos significativos, levando a 13 tópicos, que incluem cada um dos três gols, alguns momentos importantes, o início, meio e fim da partida.

A figura 3.1 mostra a série temporal correspondente aos *tweets* coletados sobre a FA Cup. Cada amostra contém o número de *tweets* coletados em um minuto.

[1] https://en.wikipedia.org/wiki/FA_Cup

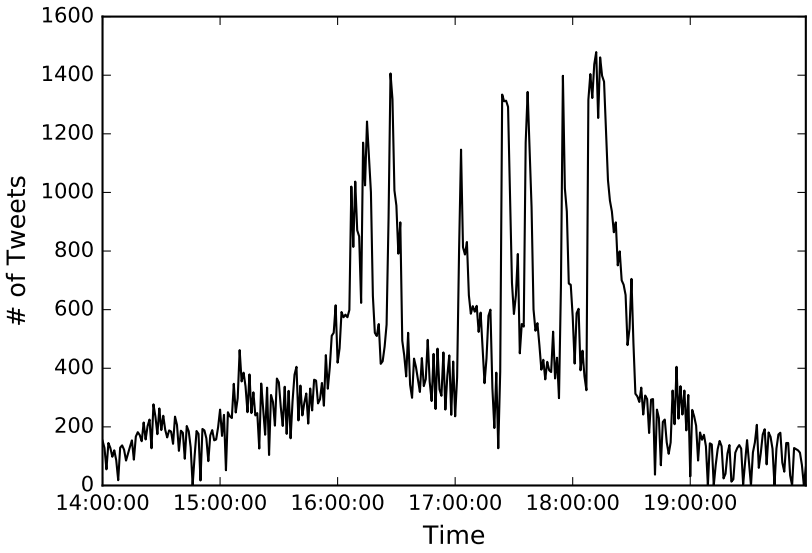


Figura 3.1: *Tweets* sobre o FA Cup.

Realizamos uma limpeza no conjunto de dados, removendo palavras irrelevantes (como preposições e pronomes), sinais de pontuação e URLs. Também deixamos todas as letras minúsculas, a fim de normalizar a escrita.

Observe que, embora a distribuição de Poisson detecte eventos sem conhecer o conteúdo dos *tweets*, nosso método difere analisando o assunto presente em cada *tweet* por meio de bigramas.

3.3 Avaliação

Para analisar nossos resultados, usamos as seguintes métricas apresentadas em (Aiello et al., 2013). Usamos exatamente as mesmas métricas e o mesmo conjunto de dados, portanto, nossos resultados são diretamente comparáveis aos deles.

- *Topic recall* (T-Rec): percentagem de eventos de verdadeiros detectados com sucesso, isto é, a taxa de verdadeiro positivo para a detecção de eventos.

$$\text{T-Rec} = \frac{\text{Eventos detectados} \cap \text{Eventos verdadeiros}}{\text{Eventos verdadeiros}};$$

- *Keyword Precision* (K-Prec): porcentagem de palavras-chave detectadas corretamente sobre o total de palavras-chave para um determinado evento rotulado no conjunto de dados, ou seja, a taxa de negativo negativo para a detecção de palavras-chave.

$$\text{K-Prec} = \frac{\text{Palavras-chaves do evento} \cap \text{Palavras-chaves detectadas}}{\text{Palavras-chaves do evento}};$$

- *Keyword Recall* (K-Rec): Porcentagem de palavras-chaves identificadas corretamente no total de palavras-chaves do evento, ou seja, a taxa de verdadeiro positivo para detecção de palavras-chaves.

$$\text{K-Rec} = \frac{\text{Palavras-chaves do evento} \cap \text{Palavras-chaves detectadas}}{\text{Palavras-chaves detectada}}.$$

3.4 Resultados

Um total dos 50 bigramas mais frequentes foi calculado para construir a série temporal de probabilidade, bem como para selecionar as possíveis palavras-chave descobertas pela distribuição de Poisson, como discutido na seção 3.1. Utilizamos a abordagem da distribuição de Poisson, como *baseline* para nossa proposta.

Usamos uma janela deslizante de 15 unidades com 1 minuto para as duas abordagens descritas neste trabalho. Podemos alterar o intervalo n da janela deslizante, que altera a sensibilidade do método para eventos.

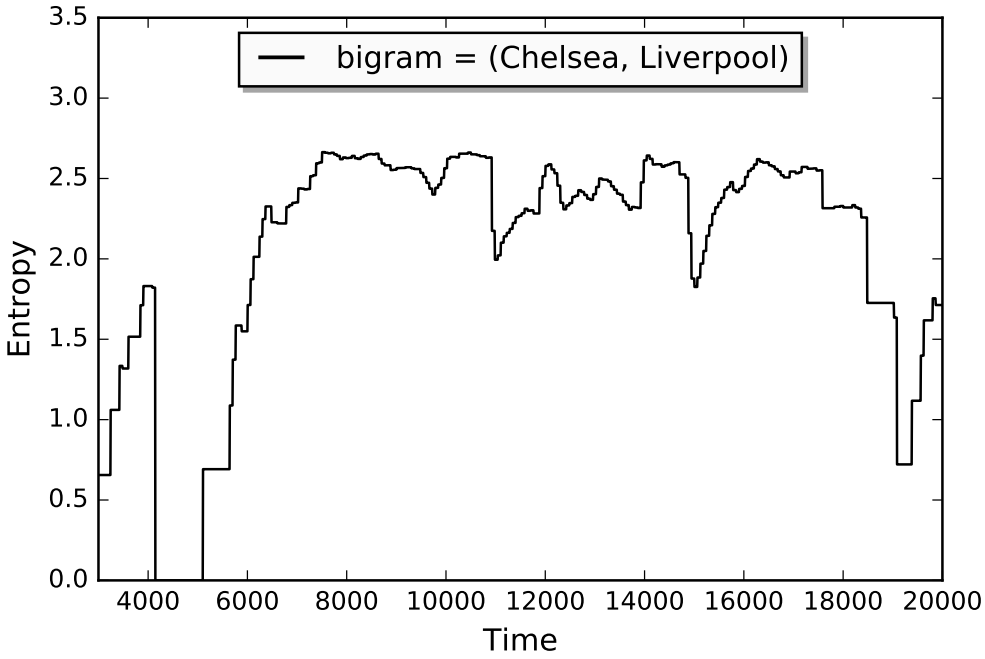


Figura 3.2: Dinâmica da entropia do bigrama mais representativo para o conjunto de dados avaliado.

Conforme declarado na seção 2, o objetivo é detectar as transições de fase da entropia entre janelas consecutivas. A figura 3.2 mostra a dinâmica da entropia do bigrama mais representativo (aquele com mais ocorrências) do conjunto de dados avaliados.

Como calculamos a entropia de uma janela, consideramos que a entropia de todos os intervalos de tempo dentro de uma janela é constante. Observe que após o tempo 5000, o sistema

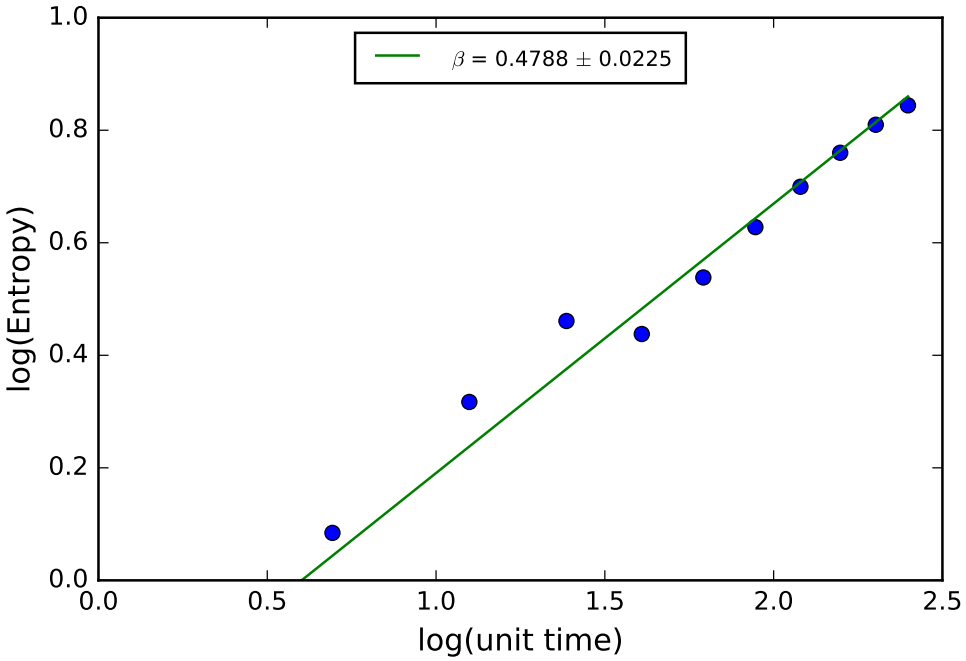


Figura 3.3: Log-log plot da $H^W(t)$ versus $|t - t_c|$ acima do ponto crítico. O expoente crítico $\beta = 0.4788$ sugere que esta é uma transição de fase contínua.

passa a apresentar outra dinâmica onde a entropia é maior que 0.

A figura 3.3 estima o expoente da escala β na vizinhança do ponto crítico $t_c = 5000$ como a inclinação da linha ajustada $H^W(t) \propto |t - t_c|^\beta$ em uma plotagem de log-log de $H^W(t)$ versus $|t - t_c|$. O expoente crítico $\beta = 0.4788$ sugere que esta é uma transição de fase contínua (Argolo et al., 2015).

Esse expoente crítico é caracterizado por um transiente lento que provavelmente se deve à dinâmica intrínseca da postagem na mídia social para circunstâncias como esportes ao vivo. Em tais circunstâncias, alguns usuários postam imediatamente após a ocorrência de um evento, enquanto que, alguns outros levam algum tempo antes de postar, portanto, o sistema muda continua e lentamente.

Para detectar transientes lentos, assumimos que sempre que a entropia se move de valores baixos para valores altos entre os valores mínimo e máximo, somos capazes de detectar a transição. Observe que não queremos detectar o transitório somente quando a entropia atinge seu valor máximo, portanto, o intervalo de detecção de entropia não deve conter o valor máximo.

Para escolher o intervalo mais adequado para detectar o transiente, usamos a curva ROC mostrada na figura 3.4. Este ROC foi construído como a Sensibilidade (taxa verdadeiro positivo) versus 1-Especificidade (taxa falso positivo) para diferentes valores de $n \in \{7, 10, 15\}$.

Nosso método consiste em capturar a transição de fase da entropia, quando ela muda de aproximadamente 0 para um valor intermediário, como evidenciado pela figura 3.5. Esta figura mostra alguns bigramas detectados usando o método proposto no conjunto de dados analisado.

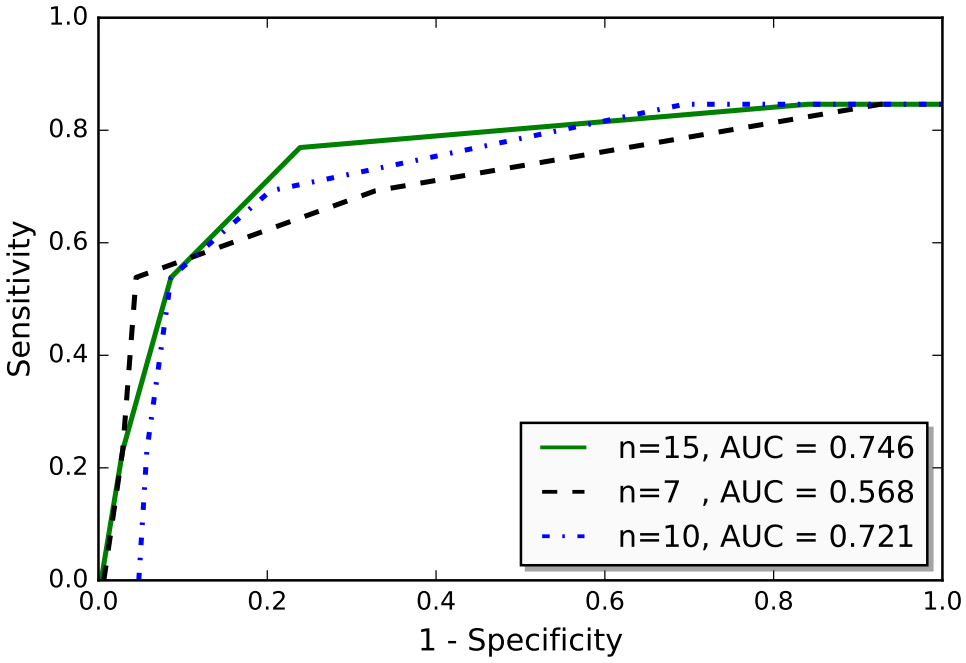


Figura 3.4: Curva ROC para diferentes valores de n para nossa proposta

Os bigramas ({chelsea, goal}; {yellow, card}; {liverpool, goal}) representam os três gols e dois cartões amarelos, mostrando também o tempo aproximado e a importância de cada evento. Esses resultados também fornecem evidências de que nossa proposta tem uma boa sensibilidade ao tempo: os cartões amarelos ocorreram em um período próximo, mas a proposta conseguiu diferenciá-los.

Observe que o horário identificado não é o horário exato em que os eventos ocorreram ou duraram, pois os usuários precisam de um tempo (de duração desconhecida) para responder ao evento. Comparando com os rótulos verdadeiros, podemos ver que os eventos detectados estão realmente próximos do evento real, com cerca de 4 segundos de diferença entre eles.

Por uma questão de ilustração, a tabela 3.1 mostra alguns tópicos detectados junto com os rótulos correspondente.

Para avaliar nossos resultados, usamos as métricas descritas no conjunto de dados da FA Cup que estão apresentados na tabela 3.2.

Comparamos os resultados obtidos com algumas técnicas da literatura, apresentadas por (Aiello et al., 2013) e (Nguyen and Jung, 2017). Todos os resultados, exceto da nossa proposta e Poisson, foram coletados dos artigos originais e copiados para a tabela. Os melhores resultados são apresentados em negrito.

Os resultados apresentados pela distribuição de Poisson são esperados, pois analisa apenas a dinâmica apresentada pelos *tweets*, não acessando seus conteúdos, e é utilizada aqui como *baseline*. Para melhor comparar as técnicas, criamos uma coluna adicional com a média harmônica de T-Rec, K-Prec e K-Rec. Podemos observar que nossa abordagem possui melhor

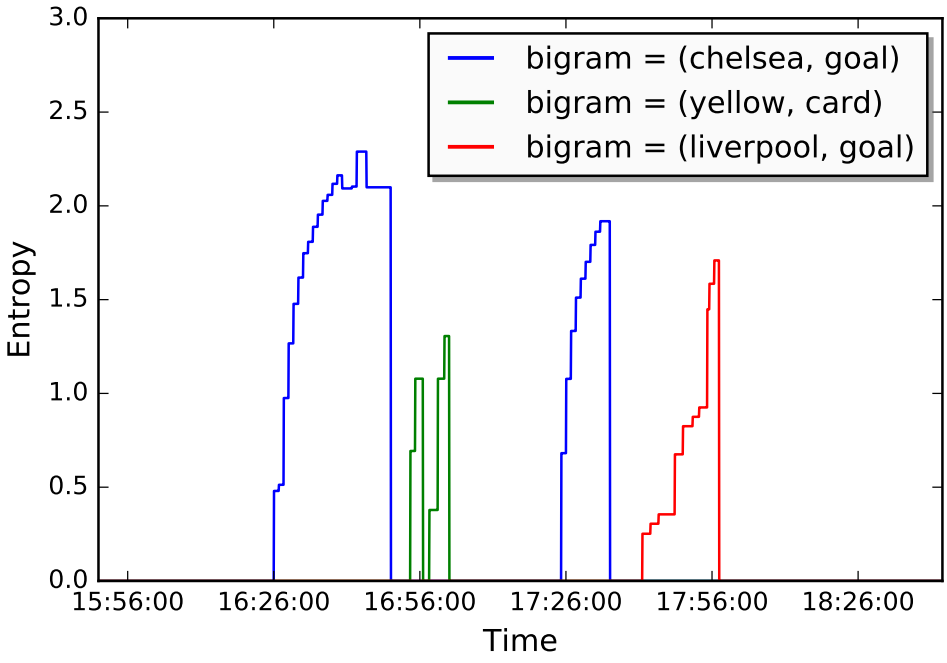


Figura 3.5: Entropia relacionada com 3 bigramas característicos detectadas para analise do conjunto de dados.

Tabela 3.1: Detecção de eventos para o conjunto de dados do FA Cup

#	Tópico detectado	História correspondente	Exemplo de tweet
1	goal chelsea ramires scores yes first liverpool	Gol do jogador Ramires	#Ramires scores in the 11th minute #Chelsea lead at #Wembley
2	mikel gets yellow card agger	O Jogador do Liverpoll Mikel recebeu um cartão amarelo	@:Yellow Card to Mikel for tackling Gerrard... 37"#FA-Cup
3	super cech saved line carroll goal andy liverpool claiming ball	Liverpool ficou muito próximo de fazer um gol, por conta do jogador Andy Carroll. Petr Cech fez uma defesa fantástica.	@: Great Save by Petr Cech After All! #FACupFinal

Tabela 3.2: Comparação entre os métodos usando o conjunto de dados FA Cup

Método	T-Rec	K-Prec	K-Rec	H. Mean
Poisson (baseline)	0.308	0.124	0.202	0.184
Petrović et al. (2010) (Doc-p)	0.692	0.337	0.583	0.490
Aiello et al. (2013) (FPM)	0.308	**0.750**	0.429	0.434
Aiello et al. (2013) (SFPM)	0.615	0.234	0.658	0.404
Aiello et al. (2013) (BNGran)	0.769	0.299	0.578	0.470
Nguyen and Jung (2017)	0.769	0.453	0.548	0.562
Nossa proposta	**0.846**	0.640	0.594	**0.678**
Blei et al. (2003) (LDA)	0.692	0.164	**0.683**	0.333

desempenho quando usada a métrica T-Rec, é a segunda melhor quando usada K-Prec e é a terceira melhor quando usada a métrica K-Rec. Com base nessas métricas, nossa proposta é a melhor no geral.

Vale ressaltar que, como o *Twitter* atualizou sua política de uso, alguns *tweets* antigos foram excluídos. Assim, não conseguimos baixar completamente todos os dados do conjunto de dados, o que possivelmente afetou algumas métricas de avaliação.

4

Considerações Finais

Neste trabalho usamos a entropia, uma medida de informação, para modelar uma ocorrência de um evento nas mídias sociais. Nossa hipótese é que, durante a ocorrência de um evento, a entropia dos bigramas extraída da mídia social muda sua dinâmica e observamos uma transição de fase contínua da dinâmica de entropia. Portanto, propusemos um novo método para detectar eventos no Twitter com base em séries temporais formadas pelas probabilidades das palavras-chave extraídas do conteúdo de *tweets*.

O método proposto neste trabalho apresenta resultados satisfatórios quando comparado ao estado da arte e apresentou melhores resultados gerais comparados a alguns modelos da literatura. Além disso, fornecemos algumas evidências de que nosso método é sensível para detectar eventos que ocorrem próximos ao tempo.

Como trabalhos futuros, nós estamos no processo de criação e submissão deste trabalho para o periódico *IEEE Transactions on Knowledge and Data Engineering* (Qualis A1 - Fator de impacto 2.775) a fim de realizar a finalização do projeto. Além disso, alguns desdobramento deste projeto foram aceitos nas seguintes conferências:

- BARROS, Pedro H. ; CARDOSO, I. ; BARBOSA, K. ; FRERY, A. C. ; ALLENDE-CID, H. ; MARTINS, I. ; RAMOS, H. S. . “Identifying Communities in Social Media with Deep Learnin”. In: 21st International Conference on. Human-Computer Interaction., 2018, Las Vegas.

- P. Barros , I. Cardoso, A. A.F. Loureiro, and H. S. Ramos, “Event detection in social media through phase transition of bigram entropy”, in 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, Jun. 2018.

- Machado, W. S. ; Almeida E. S. ; Aquino A. L. L. ; Barros P. H. ,“Aplicação de técnicas de inteligência computacional para identificação ADLs e quedas”. in SBCUP 2018, Natal, Brasil.

- Minicurso[1] sobre “Introdução a aprendizagem profunda” ministrado no 70º SBPC, realizado na cidade de Maceió.

Por se tratar de um Trabalho de Conclusão de Curso, é importante ressaltar o aprendizado obtido, através do estudo dos assuntos expostos e suas aplicações, ajudando assim na construção e avanço da fronteira do conhecimento.

Este período foi muito importante para o meu crescimento intelectual, profissional e pessoal. No âmbito profissional foi fundamental na minha escolha para o curso de pós-graduação, pois tentarei fazer mestrado na área da bolsa de pesquisa. A importância deste projeto é tal que foi capaz de me familiarizar com o ambiente de pesquisa e a produção científica. Outro benefício é que trabalhei com grandes pesquisadores pertencentes ao LaCCAN, em especial com meu orientador Prof Dr. Heitor Soares Ramos Filho, onde já tive a oportunidade de acompanhar a publicação de um artigo.

[1] https://sites.google.com/a/ic.ufal.br/heitor/short-courses/deep-learning?authuser=0

Referências bibliográficas

Luca Maria Aiello, Georgios Petkos, Carlos Martin, David Corney, Symeon Papadopoulos, Ryan Skraba, Ayse Göker, Ioannis Kompatsiaris, and Alejandro Jaimes. Sensing trending topics in twitter. *IEEE Transactions on Multimedia*, 15(6):1268–1282, 2013.

C Argolo, P Barros, T Tomé, Iram Gleria, and ML Lyra. Stationary and dynamic critical behavior of the contact process on the sierpinski carpet. *Physical Review E*, 91(5):052137, 2015.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, MDMKDD '10, pages 4:1–4:10, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0220-3.

Qi Dang, Feng Gao, and Yadong Zhou. Early detection method for emerging topics based on dynamic bayesian networks in micro-blogging networks. *Expert Syst. Appl.*, 57(C):285–295, September 2016. ISSN 0957-4174.

S Deerwester, ST Duais, GW Furnas, TK Landauer, and R Harshman. Indexing by latent semantics analysis. *Journal of the American Society for Information Science*, 41(6):391–407, Sep 1990.

Wenwen Dou, Xiaoyu Wang, William Ribarsky, and Michelle Zhou. Event detection in social media data. In *IEEE VisWeek Workshop on Interactive Visual Text Analytics-Task Driven Analytics of Social Media Content*, pages 971–980, 2012.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.

Geir Evensen. The ensemble kalman filter: Theoretical formulation and practical implementation. *Ocean dynamics*, 53(4):343–367, 2003.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):44, 2014.

R. A. Jarvis and E. A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Trans. Comput.*, 22(11):1025–1034, November 1973. ISSN 0018-9340.

SK Katti and A Vijaya Rao. Handbook of the poisson distribution, 1968.

Amy N Langville and Carl D Meyer. *Google's PageRank and beyond: The science of search engine rankings*. Princeton University Press, 2011.

Chenliang Li, Aixin Sun, and Anwitaman Datta. Twevent: Segment-based event detection from tweets. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 155–164. ACM, 2012. ISBN 978-1-4503-1156-4.

Michael Mathioudakis and Nick Koudas. Twittermonitor: Trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, pages 1155–1158, New York, NY, USA, 2010. ACM.

Kevin Patrick Murphy and Stuart Russell. Dynamic bayesian networks: representation, inference and learning. 2002.

Duc T. Nguyen and Jai E. Jung. Real-time event detection for online behavioral analysis of big social data. *Future Generation Computer Systems*, 66:137 – 145, 2017. ISSN 0167-739X.

Katja Nummiaro, Esther Koller-Meier, and Luc Van Gool. An adaptive color-based particle filter. *Image and vision computing*, 21(1):99–110, 2003.

Ruchi Parikh and Kamalakar Karlapalem. Et: Events from tweets. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13 Companion, pages 613–620, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2038-2.

Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. pages 181–189, 2010.

Siméon Denis Poisson. Probabilité des jugements en matière criminelle et en matière civile, précédées des règles générales du calcul des probabilitiés. *Paris, France: Bachelier*, 1:1837, 1837.

Osvaldo A. Rosso, Hugh Craig, and Pablo Moscato. Shakespeare and other english renaissance authors as characterized by information theory complexity quantifiers. *Physica A: Statistical Mechanics and its Applications*, 388(6):916 – 926, 2009.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 851–860, New York, NY, USA, 2010. ACM.

Stijn Marinus Van Dongen. *Graph clustering by flow simulation*. PhD thesis, 2000.

Jianshu Weng and Bu-Sung Lee. Event detection in twitter. *ICWSM*, 11:401–408, 2011.