Report

Rocco Caruso

December 9, 2015

Chapter 1

Problem definition

Dato uno stream di tweet (ordinato temporalmente) l'obiettivo è quello di individuare dei "topic" o "event". In letteratura esistono due principali metodologie:

- document-pivot
- feature-pivot

In questo lavoro è stata adottata la prima metodologia, l'obiettivo è quindi suddividere lo stream in cluster, tali che ciascun cluster corrisponda a tutti i tweet relativi ad un "evento ". E' altresì necessario, una volta suddiviso lo stream in vari cluster, distinguere quali sono realmente eventi (flashmob) da quelli che non lo sono, per mezzo di un classificatore.
La definizione di evento utilizzata è quella utilizzata per il Topic Detection and Tracking (TDT) [All02] :

Definizione 1. *un evento è qualcosa che accade in un luogo specifico in un tempo specifico*

L'intero processo può essere suddiviso in 5 attività come descritto in figura 1.1

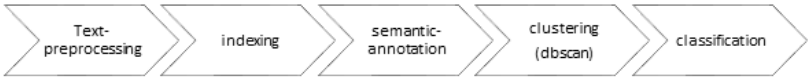


Figure 1.1: workflow

1.1 Preprocessing

Il preprocessing è uno step essenziale per qualunque task di text-mining, quando si ha a che fare con testo derivante dai social media come twitter, tale step diviene di vitale importanza a causa delle caratteristiche dei tweet. È dunque necessaria una fase accurata di preprocessing per il testo dei tweet prima di poter eseguire qualsiasi modellazione su di essi. Il primo passo in questa fase è la rimozione di quelle feature testuali legate ai tweet come:

- *url:* dal testo sono eliminati tutti i riferimenti a url o media
- *@-mentions:* vengono eliminate tutte le mentions ad altri utenti
- *#hashtag:* Per quanto riguarda gli hashtag viene solo eliminato il caratattere # poichè se fossero eliminati si potrebbe perdere della semantica dal testo. Inoltre spesso nei tweet gli hashtag sono composizioni di più parole dove ciascuna parola inizia con una lettera maiuscola (camel Case), come ad esempio #StopBombingGaza. Gli hashtag che si presentano nella forma su descritta verranno scomposti in nelle parole che di cui sono composti (#StopBombingGaza → Stop Bombing Gaza).
- *RT:* per i retweet viene considerato il testo del tweet originale.

Per il primo passo non è necessaria alcuna fase di parsing del testo poichè tutte queste informazioni sono fornite dalle api di twitter sotto forma di dati strutturati[1]. Molto spesso i tweet sono composti da keywords appartenenti a lingue diverse o contengono caratteri speciali, per tale motivo saranno eliminati tutti i non latin characters. Tramite apposita espressione regolare vengono identificati ed eliminate le emoticons presenti nel testo. Una volta effettuate queste pulizie, si passa all'identificazione della lingua del tweet per mezzo di una libreria java[2], e saranno considerati solo i tweet in lingua inglese.

[1] nell attributo entities del tweet

[2] https://code.google.com/p/language-detection/

1.2 Indexing

Il testo del tweet verrà rappresentato con un boosted tf-idf vector utilizzando Apache-lucene[3]. A partire dal testo dei tweet ripulito, sono state effettuati ulteriori step di nlp come :

- stop word removal
- pos tagging (tramite lo stanford pos tagger addestrato su un modello creato a partire da tweet
- stemming (o lemmtization)

Poichè si analizza uno stream di tweet in maniera incrementale è necessario che anche lo schema di pesatura tf-idf sia incrementale ovvero le document-frequencies per una word w variano nel tempo.

Il peso di una word w di un tweet avente tempo t è dato da:

$$tf - idf_w = tf(w)\ log\frac{N_t}{df_t(w)}boost(w) \tag{1.1}$$

Dove N_t è il numero di tweet al tempo t. Risulta fondamentale assegnare un boost a determinate word poiché, a causa della natura dei tweet (max 140 caratteri), il tf di ciascuna keyword è solitamente pari a 1.

$$boost(w) := \begin{cases} 2.0 & \text{se } w \text{ è un hashtag,} \\ 1.5 & \text{se } w \text{ è un Proper-Noun,} \\ 1 & \text{altrimenti.} \end{cases}$$

Solitamente uh hashtag ha un alto potere informativo all'interno di un tweet, nel lavoro di Phuvipadawat [PM10] il boost pari a 1.5 per i poper-noun ha prodotto risultati migliori.

[3]https://lucene.apache.org/core/

1.3 Semantic Annotation

Tramite Dbpedia-Spootlight [Dai+13] è possibile annotare automaticamente il testo dei tweet con DBpedia URIs. Ciò permette di arricchire la rappresentazione testuale andando a lenire i classici problemi di ambiguità del linguaggio naturale come polisemia e sinonimia. Annotare il testo con DBpedia URis permette di sfruttare la base di conoscenza di DBpedia per poter determinare la correlazione semantica fra due termini. Il progetto DBpedia [Aue+07] oltre a organizzare in maniera strutturata le informazioni di Wikipedia, le collega ad ulteriori open-datasets come: US Census, Geonames, MusicBrainz, the DBLP bibliography, WordNet, Cyc, tramite RDF links come mostrato in figura 1.2

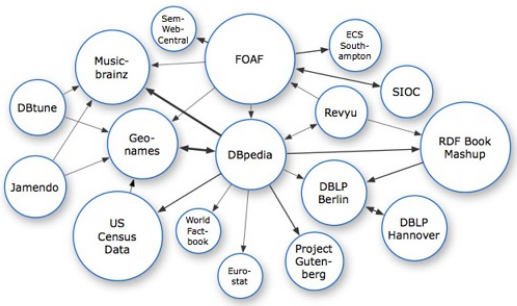


Figure 1.2: Datsets interconnessi a DBpedia.

Avendo due risorse DBpedia è possibile definire una funzione di distanza, che sfrutti la base di conoscenza di DBpedia. È stata definita una funzione di distanza *Dbpedia Semantic Distance DSD*, che impiega sia DBpedia che il dataset geografico Geonames per risorse di tipo geografico, come segue:

$$DSD(a,b) := \begin{cases} GeoDist(a,b) & \text{se } type(a) = location \wedge type(b) = location \ , \\ NDD(a,b) & \text{altrimenti} \end{cases} \tag{1.2}$$

Dove $DSD(a,b) \in [0,1]$, Milne e Witten [MW08] hanno utilizzato gli hyperlink delle pagine Wikipedia per poter definire la correlazione fra due articoli wikipedia (e quindi due risorse dbpedia): date due risorse a, b, possiamo definire una *Normalized DBpedia Distance (NDD)* come segue:

$$NDD(a,b) := \begin{cases} \frac{\log(max\{|A|,|B|\}) - log(|A \cap B|\})}{N - \log(min\{|A|,|B|\})} & \text{se } A \cap B \neq \emptyset \ , \\ 1 & \text{altrimenti} \end{cases} \tag{1.3}$$

A e B sono gli insiemi delle risorse DBpedia che hanno un link rispettivamente verso a e b, mentre N è il numero totale di risorse in DBpedia. Questa distanza varia nell'intervallo [0,1] dove 1 sta ad indicare che non vi è nessuna correlazione fra i due concetti, mentre 0 indica che i due concetti hanno lo stesso significato. L'idea alla base, è che due risorse saranno simili se esiste una terza che ha un link verso entrambe.

Per le risorse DBpedia di tipo "location"[4] è stata definita la funzione di distanza 1.6 che impiega il dataset "GeoNames". Poiché il task, è quello di identificare eventi1 che accadono in un luogo specifico, è utile valutare la correlazione sulla base di informazioni geografiche. Geonames [5] è un database geografico contenente più di 10 milioni di nomi geografici, ogni risorsa è classificata da una "feature class" (administrative divisions, populated places, structures, mountains, water bodies, etc) e ulteriormente sotto classificata da 645 "feature codes". Tra le varie informazione e relazioni che interconnettono le risorse nell'ontologia sono state utilizzate le seguenti per poter valutare la similarità fra le risorse:

- *Country-code:* codice identificativo della nazione cui appartiene la risorsa
- *FeatureClass*
- *FeatureCode*
- *parentADM1:* risorsa di tipo ADM1(regione) che contiene la risorsa
- *parentADM2:* risorsa di tipo ADM2(provincia) che contiene la risorsa
- *Coordinates:* (latitude,longitude)

La funzione di similarità deve quindi tener conto dei diversi livelli di granularità delle risorse Geonames e sfruttare le relazioni fra di esse. Verrà definita una distanza *Geonames Distance (GD)* compresa fra [0,1] cosi definita:

$$GD(a,b) := \begin{cases} 0.8 & \text{se } \exists c \mid (c\ parentADM1\ a) \ \wedge \ (c\ parentADM1\ b) \\ 0.8 & \text{se } \ (a\ parentADM2\ b) \ \vee \ (b\ parentADM2\ a) \\ 0.3 & \text{se } \exists c \mid (c\ parentADM2\ a) \ \wedge \ (c\ parentADM1\ b) \\ 0.3 & \text{se } \ (a\ parentADM2\ b) \ \vee \ (b\ parentADM1\ a) \\ 1 & \text{altrimenti.} \end{cases} \tag{1.4}$$

[4]per risorse di tipo location si intendono quelle risorse classficate come http://dbpedia.org/ontology/Location

[5]http://www.geonames.org/

Per alcune risorse di granularità più fine come città, parchi, edifici ha senso invece, considerare le coordinate. Avendo le coordinate è possibile calcolare la distanza espressa in km fra di esse usando l'Harvesine Forula[6].Sia d la distanza espressa in km tra a e b, verrà definita una *Coordinate Distance (CoordDist)*

$$coordDist(a,b) := \begin{cases} 0.0 & \text{se } 0 < d \leq 5, \\ 0.2 & \text{se } 5 < d \leq 10, \\ 0.25 & \text{se } 10 < d \leq 15, \\ 0.3 & \text{se } 15 < d \leq 25, \\ 0.4 & \text{se } 25 < d \leq 40, \\ 0.7 & \text{se } 40 < d \leq 50, \\ 0.8 & \text{se } 70 < d \leq 80, \\ 1 & \text{altrimenti.} \end{cases} \tag{1.5}$$

Si può cosi definire la distanza geografica *GeoDist* come segue:

$$GeoDist(a,b) := \begin{cases} coordDist(a,b) & \text{se sia a e b hanno coordinate} \\ GD(a,b) & \text{altrimenti.} \end{cases} \tag{1.6}$$

[6]Haversine formula

1.4 Clustering

L'algoritmo di clustering utilizzato è Incremental Dbscan poiché può gestire bene il rumore e non necessita di parametri come il numero di cluster a priori. La distanza utilizzata si basa sia sulla rappresentazione tf-idf del tweet sia sui DBpedia URIs estratti :

$$dist(a,b) = 1 - timeSim(a,b)\,\frac{textSim(a,b) + semanticSim(a,b)}{2} \tag{1.7}$$

La similarità testuale è data dalla similarità del coseno fra i vettori tf-idf dei due tweet:

$$textSim(a,b) = \frac{v_a \cdot v_b}{||v_a||\,||v_b||}$$

Anche il tempo di creazione dei tweet verrà preso in considerazione per valutarne la distanza, poiché anche se due tweet avessero un testo molto simile es *"tonight flashmob in central park"*, ma fossero pubblicati ad un mese di distanza, è molto inverosimile che si riferiscano al medesimo evento. Per tale ragione, è stata definita una similarità temporale [7]

$$timeSim(a,b) := \begin{cases} 1 - \frac{|d_a - d_b|}{31} & \text{se } |d_a - d_b| < 31\ , \\ 0 & \text{altrimenti} \end{cases}$$

Ad un tweet, tramite il processo di annotazione semantica, possono essere associate una nessuna o più risorse DBpedia,la distanza semantica sarà data dalla distanza degli insiemi di risorse associati ai due tweet valutata secondo la distanza definita in precedenza per le risorse DBpedia1.2. La similarità fra un elemento x e un insieme Y è data da:

$$sim(x,Y) = sup\{1 - DSD(x,y)\ |y \in Y\}$$

Dati due insiemi di risorse DBpedia D_a, D_b la similarità fra i due insiemi sarà definita come:

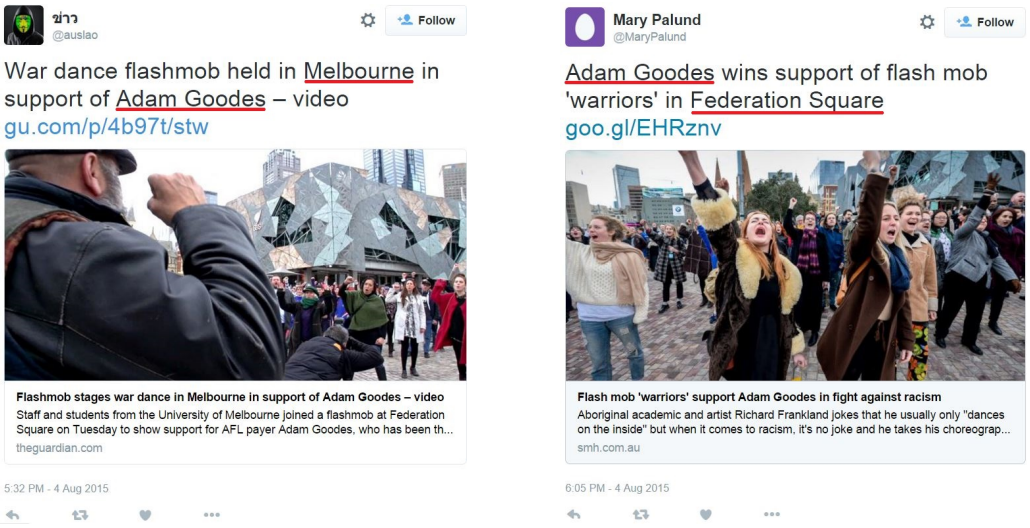
$$sim(D_a,D_b) = \frac{\sum\limits_{x \in D_a} sim(x,D_b)}{|D_a|} \tag{1.8}$$

Questa funzione di similarità non è simmetrica,[8] Per ottenere una funzione simmetrica è sufficiente definirla come:

$$semanticSim(a,b) := \begin{cases} \frac{sim(D_a,D_b)+sim(D_b,D_a)}{2} & \text{se } D_a, D_b \neq \emptyset \\ 1 & \text{altrimenti.} \end{cases}$$

[7] d_a=#days from the epoch of tweet a

[8] se $D_a \subseteq D_b \Rightarrow sim(D_a,D_b) \neq sim(D_b,D_a)$



(a) (b)

Figure 1.3: Due tweet appartenti allo stesso cluster

Se ad uno dei due tweet non è associata nessuna risorsa, la similarità semantica sarà pari ad uno, quindi la loro similarità sarà valutata solo in base alla loro rappresentazione testuale. Utilizzare una similarità semantica serve ad attenuare il problema della "fragmentation" di cui sono affetti i metodi document-pivot, ovvero utilizzando solo la similarità testuale molti eventi possono essere erroneamente suddivisi in più cluster. Inoltre Petkos e Papadopoulus [PPK14] hanno constato che se due tweet condividono uno stesso URL, o un tweet e in reply all'altro, allora si riferiscono allo stesso topic/evento.

Si considerino i tweet a, b in figura 1.3:

- $textSim(a, b) = 0.46$
- $timeSim(a, b) = 1.0$ i tweet sono stati pubblicati entrambi il 4/8/2015
- $semanticSim(a, b) := \frac{sim(D_a, D_b) + sim(D_b, D_a)}{2}$
  $D_a =$ { <Melbourne>, <Adam Goodes>}
  $D_b =$ { <Adam Goodes>, <Federation Square>}

$$
\begin{aligned}
NSD(Melbourne, AdamGoodes) &= NDD(Melbourne, AdamGoodes) \\
&= \frac{\log(max\{|A|, |B|\}) - log(|A \cap B|)}{N - \log(min\{|A|, |B|\})} \\
&= \frac{\log(643) - log(10)}{N - \log(138)} = 0.367
\end{aligned}
$$

$$
\begin{aligned}
&NSD(Melbourne, FederationSquare) = \\
&= GeoDist(Melbourne, FederationSquare) = \\
&= CoordDist(Melbourne, FederationSquare) = 0 \text{ (poichè la distanza in km è 0.8)}
\end{aligned}
$$

$$
\begin{aligned}
&NSD(AdamGoodes, FederationSquare) = \\
&\quad NDD(AdamGoodes, FederationSquare) = 1 \text{ poichè } A \cap B = \emptyset
\end{aligned}
$$

$$
\begin{aligned}
sim(Melbourne, D_b) &= sup\{1 - DSD(Melbourne, y) \,|y \in D_b\} \\
&= sup\{(1 - DSD(Melbourne, AdamGoodes)), \\
&\quad (1 - DSD(Melbourne, FederationSquare))\} \\
&= sup\{(1 - 0.367), (1 - 0)\} = 1 \\
sim(AdamGoodes, D_b) &= sup\{1 - DSD(AdamGoodes, y) \,|y \in D_b\} \\
&= sup\{(1 - DSD(AdamGoodes, AdamGoodes)), \\
&\quad (1 - DSD(AdamGoodes, FederationSquare))\} \\
&= sup\{(1 - 0), (1 - 1)\} = 1 \\
sim(D_a, D_b) &= \frac{1+1}{2} = 1,\ sim(D_b, D_a) := \frac{1+1}{2} = 1 \\
&\implies semanticSim(a, b) = 1
\end{aligned}
$$

$$
\begin{aligned}
dist(a, b) &= 1 - timeSim(a, b)\,\frac{textSim(a, b) + semanticSim(a, b)}{2} \\
&= 1 - \frac{0.46 + 1}{2} = 0.269
\end{aligned}
$$

Se invece, *SemanticSim* fosse definita come la media delle distanze fra tutte le possibili coppie fra i due insiemi avremmo:

$$semanticSim(a,b) = \frac{\sum_{x \in D_a} \sum_{x \in D_b} (1 - NSD(x,y))}{|D_a||D_b|} =$$
$$= \frac{(1-0.36) + (1-0) + (1-0) + (1-1)}{4} = 0.73$$

1.5 Classification

Per poter filtrare gli eventi reali da quelli non reali è stato addestrato un classificatore SVM, sulla base di statistiche derivanti da cluster individuati. Tali statistiche derivano da feature dei tweet che compongono i cluster e dalle annotazioni estratte da tali tweets.

- TWEET FEATURES
  - %retweets: percentuale di retweets presenti nel cluster.
  - %replies: percentuale di tweet nel cluster che sono replies
  - %mentions: percentuale di tweet nel cluster che contengono mentions
  - %hashtags: percentuale di tweet nel cluster che contengono hashtag
  - %urls: percentuale di tweet nel cluster che contengono urls
  - %media: percentuale di tweet nel cluster che contengono media
  - %authors: percentuale di autori distinti dei tweet nel cluster (se i tweet sono stati creati
- TOPIC FEATURES
  - avg token number: numero medio di token dei tweet che compongono il cluster
  - %WhereAnnotations: percentuale di tweet nel cluster che contengono annotazioni semantiche di tipo Location
  - HasSpecificLocation: booleano che indica se fra le annotazioni di tipo location vi sia almeno una con delle coordinate specifiche.

Chapter 2

dbpedia

Wikipedia [1] è divenuta una delle maggiori risorse di conoscenza disponibili nel web, ed è manutenuta da migliaia di utenti (collaboratori). Gli articoli Wikipedia sebbene composti prevalentemente da testo, contengono informazioni semi-strutturate come: template infobox , informazioni sulla categorizzazione dell'articolo, immagini, geo-coordinate e link sia verso altre pagine web sia verso altre pagine wikipedia. Gli infobox sono tabelle di coppie attributo valore, che mostrano i dati più rilevanti di ciascuna pagina wikipedia. Il progetto DBpedia [Biz+09] estrae dati strutturati da wikipedia tramite un extraction framweork open source e li unisce in una base di conoscenza multi dominio e multi lingua. Per ogni pagina presente in wikipedia, viene associato un *Uniform Resource Identifier (URI)* in DBpedia per identificare un'entità o un concetto descritto dalla corrispondente pagina Wikipedia della versione inglese. Durante il processo di trasformazione, i dati semi-strutturati come i campi infobox,categorie, pagelinks sono convertiti in triple RDF e aggiunte alla base di conoscenza come proprietà dell'entità identificata dall URI. Per rendere omogenea la descrizione delle informazioni, è stata sviluppata un ontologia e sono state definite le corrispondenze fra le proprietà presenti negli infobox e l'ontologia. L'ontologia DBpedia consisite di 320 classi e descritte da 1650 proprietà. Le classi organizzate mediante una gerarchia sussuntiva dove *owl:Thing* è la classe più generale. Poiché Il sistema di Wikipedia infobox si è evoluto in maniera decentrata, talvolta accade ad esempio che si usino diversi template per la stessa tipologia di entità (class) o si usino nomi diversi per descrive lo stesso attributo (es placeOfBirth o birthPlace). L'allineamento tra i template infobox e l'ontologia a causa di queste eterogeneità presenti nella nomenclatura, non è quindi completamente automatico, ma si basa anche su mapping definiti manualmente, forniti dalla comunità di

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

DBpedia. Ad esempio 'date of birth' and 'birth date' sono entrambi mappati con la proprietà birthDate.

2.0.1 Accedere a DBpedia

La base di conoscenza DBpedia è disponibile sul web sotto GNU Free Documentation, e può essere consultata mediante varie modalità :

- *Linked Data: linked data è la metodologia di pubblicazione dei dati RDF nel web, che utilizza gli URI http come identificativo delle risorse e il protocollo HTTP per ritrovare al descrizione rdf delle risorse. Quando si accedere ad un URI di una risorsa DBPedia mediante un semantic web agent si ottiene la descrizione rdf della risorsa mentre se si utilizza un semplice web-browser si otterrà una vista html descrizione.*
- *Sparql Endpoint* É fornito un endopoint mediante il quale si può interrogare la base di conoscenza tramite il protocollo SPARQL.
- *RDF dumps* la base di conoscenza è stata suddivisa in varie parti in base agli rdf-predicate

Chapter 3

Stato dell'Arte

3.1 Event detection nei media tradizionali

L'attività di event-detection è stata a lungo utilizzata per individuare eventi da stream testuali derivanti dai media più tradizionali come giornali o radio, infatti l'event-detection è stata per molto tempo oggetto di ricerca del programma di *Topic Detection and Tracking TDT* [**All02**], un' iniziativa promossa dalla DARPA [1], con lo scopo di organizzare stream di notizie testuali sulla base degli eventi di cui discutono. Secondo il TDT, l'obiettivo dell'attività di *event detection*, è scoprire nuovi eventi o eventi precedentemente non noti, a partire da stream di notizie testuali derivanti dai media tradizionali come notiziari o newswire, dove ciascun evento si riferisce ad un *qualcosa, non banale, che accade in un luogo e tempo specifico.* Le tecniche di event-detection possono essere classificate in due macro categorie: *document-piovot* e *document piovot* a seconda che utilizzino feature dei documenti o feature temporali delle singole keywords presenti nei documenti. La prima scopre eventi effettuando un clustering dei documenti sulla base di una qualche funzione di distanza fra i documenti stessi [YPC98], mentre nella seconda si studia la distribuzione delle singole parole e scoprono nuovi eventi raggruppando le parole [Kle02] Come evidenziato da [YPC98] infatti, l'event detection può essere ricondotto al problema della scoperta di pattern in uno stream testuale, quindi il modo più naturale per scoprire nuovi eventi, è quello di usare un algoritmo di clustering. Il task di event-detection si può suddividere in tre fasi principali: data preprocessing, data rappresentation, data organization o clustering. Nella fase di preprocessing in questa fase vengono applicate al testo delle classiche tecniche di NLP come la rimozione di stopwords, tokenizzazione e stemming. I modelli di rappresentazione di

[1] Agenzia di ricerca agenzia per i progetti di ricerca avanzata per la difesa

dati più utilizzati per l'event detection sono *il modello vettoriale* e *i modello bag of words*, i cui elementi saranno diversi da zero, se il termine corrispondente è presente nel documento. A ciascun termine nel vettore, è assegnato un peso secondo lo schema *tf-idf* [Sal89] che valuta quanto è importante una parola per un documento all'interno di un corpus. Questo modello di rappresentazione non prende in considerazione l'ordine temporale delle parole ne caratteristiche sintattiche o semantiche del testo come il part of speech tag o named entities. Per questa ragione utilizzando questo modello, ad esempio, sarebbe difficile distinguere due eventi simili ma accaduti ad un mese di distanza fra loro. Nel lavoro di [YPC98] il task di scoperta di nuovi eventi da uno stream testuale di news è suddiviso in due fasi principali: Retrospective Event Detection (RED), New Event Detection (NED). La prima fase (RED) comporta la scoperta di eventi da una collezione già nota di documenti, mentre nella seconda si cerca di identificare gli eventi dallo stream di notizie in tempo reale. Per il RED è stato utilizzato un algoritmo di clustering gerarchico: *Group Average Clustering GAC*, che consente anche di descrivere gli eventi identificati con diversi livelli di granularità. Per la fase di New Event Detection, invece, solitamente viene adottato un agoritmo di clustering incrementale single-pass [All02; YPC98] che consente di suddividere i documenti nei vari cluster non appena arrivano dallo stream. In particolare ciascun documento viene elaborato sequenzialmente e viene assorbito dal cluster più simile, o verrà creato un nuovo cluster se la similarità è al di sotto di una soglia prestabilita. In un ambiente di detection on-line (NED) un forte vincolo è costituito dal fatto che non si può hanno informazioni di eventi futuri,ovvero non è possibile utilizzare dati provenienti da documenti successivi, cronologicamente, a quello corrente. Utilizzando un modello di rappresentazione vettoriale, questo vincolo pone delle problematiche su come gestire la crescita del vocabolario dei termini quando vengono aggiunti nuovi documenti al corpus e come modificare delle statistiche inerenti l'intero corpus come l'IDF. La soluzione suggerita da [YPC98] è quella di modificare il vocabolario dei termini in maniera incrementale e modificare l'IDF ogni qual volta viene aggiunto un nuovo documento.

$$idf_t(w) = log_2 \left(\frac{N_t}{df_t(w)} \right) \tag{3.1}$$

dove N_t è il numero di documenti fino al tempo t e $df_t(w)$ è la document frequency della keyword w fino al tempo t. In pratica questi approcci NED, tendono a divenire molto costosi sia in termini di risorse computazionali che di tempo richiesto, e in taluni casi addirittura irrealizzabili se non utilizzando delle tecniche che ne migliorino l'efficienza. Una possibile tecnica per ridurre i costi è quella di utilizzare una *slidining time window* [LTY07; Pap99] per

limitare il numero vecchi documenti da analizzare quando si prende in considerazione un nuovo documento. Utilizzare una finestra temporale non solo riduce i costi, ma rende possibile di limitare lo scope degli eventi scoperti, permettendo di identificare eventi simili ma che accadono in uno slot temporale diverso [YPC98]. Tutte queste tecniche per il TDT si basano sull'assunzione che tutti i documenti siano rilevanti e contengono informazioni di eventi, poichè lavorano su stream di informazioni affidabili, assunzione che è chiaramente violata per l violata per quanto riguarda lo stream di Twitter.

Nelle tecniche feature-pivot un evento, all'interno di uno stream testuale, è modellato come una attività che presenta picchi di frequenza (burst), ovvero un evento è rappresentato dall'insieme di keywords che presentano un burst[Kle02]. L'assunzione fatta da queste tecniche è che alcune parole avranno un incremento di utlizzo repentino quando accade un evento. Nel lavoro di [Kle02] viene utilizzato un'automa a stati infiniti per poter identificare i burst delle keyword all'interno dello stream testuale. Gli stati dell'automa corrispondono alla frequenze delle singole parole, mentre le transizioni fra a gli stati identificano i burst che corrisopondo a un cambiamento significativo nella frequenza. A differenza delle tecniche document-pivot, in questo caso si cercano di identificare eventi raggruppando (ovvero effettuando il clustering) quelle keyword che presentano un burst, piuttosto che i documenti. Nel lavoro di [All02] la frequenza delle parole viene modellata tramite una distribuzione bionomiale, dpoi vengono idnvidiuate le bursty-keywords sulla base di una soglia euristica, per poi raggrupparle al fine di identificare gli eventi.

3.1.1 Twitter Event Detection

L'attività di Event Detection nei microblogs come Twitter, è concettualmente molto simile all' event detection nei media tradizionali. In entrambi viene dato in input al sistema uno stream di documenti testuali e devono scoprire degli eventi raggruppando i documenti o le singole parole contenute nei documenti stessi. L'unica differenza che hanno è il tipo e il volume di documenti dello stream che devono analizzare, in pratica tuttavia, questa unica differenza si riflette in una serie di nuove sfide per il task dell'event detection. Innanzitutto il volume di documenti nel caso dei microblogs come twitter è di diversi ordini di grandezza più grande rispetto ai media tradizionali, ma soprattutto nel caso di stream derivanti dai media tradizionali, tutti i documenti hanno una qualche rilevanza rispetto ad un avvenimento, una notizia. Nel caso dei tweet, vi possono essere grandi quantità di messaggi privi di significato (pointless babbles) [HW11] e rumors [CMP11]. Inoltre le caratteristiche di Twitter e la sua poplarità sono molto allettanti per spammers

e altri e altri content polluters [LEC11] per disseminare pubblicità, virus, pornografia pishing o anche per compromettere la reputazione del sistema. La sfida più grande che bisogna affrontare nell'attività di event detection per i tweet, è quindi quella di poter separare informazioni mondane e inquinate da informazioni su eventi reali. Altre difficoltà sono causate principalmente dalla brevità dei messaggi (max 140 caratteri), dall'uso di abbreviazioni, errori di spelling e grammaticali, e l'uso improprio della struttura delle frasi e di più lingue. Per queste ragioni anche le tecniche tradizionali di natural language processing meno appropriate per i tweet.

Bibliography

[Sal89] Gerard Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989. ISBN: 0-201-12227-8.

[YPC98] Yiming Yang, Tom Pierce, and Jaime Carbonell. "A Study of Retrospective and On-line Event Detection". In: *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '98. Melbourne, Australia: ACM, 1998, pp. 28–36. ISBN: 1-58113-015-5. DOI: 10.1145/290941.290953. URL: http://doi.acm.org/10.1145/290941.290953.

[Pap99] R. Papka. *On-line New Event Detection, Clustering, and Tracking TITLE2:* tech. rep. Amherst, MA, USA, 1999.

[All02] James Allan, ed. *Topic Detection and Tracking: Event-based Information Organization*. Norwell, MA, USA: Kluwer Academic Publishers, 2002. ISBN: 0-7923-7664-1.

[Kle02] Jon Kleinberg. "Bursty and Hierarchical Structure in Streams". In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. Edmonton, Alberta, Canada: ACM, 2002, pp. 91–101. ISBN: 1-58113-567-X. DOI: 10.1145/775047.775061. URL: http://doi.acm.org/10.1145/775047.775061.

[Aue+07] Sören Auer et al. "DBpedia: A Nucleus for a Web of Open Data". In: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*. 2007, pp. 722–735. DOI: 10.1007/978-3-540-76298-0_52. URL: http://dx.doi.org/10.1007/978-3-540-76298-0_52.

[LTY07] Gang Luo, Chunqiang Tang, and Philip S. Yu. "Resource-adaptive Real-time New Event Detection". In: *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*. SIGMOD '07. Beijing, China: ACM, 2007, pp. 497–508. ISBN: 978-1-59593-686-8. DOI: 10.1145/1247480.1247536. URL: http://doi.acm.org/10.1145/1247480.1247536.

[MW08] David Milne and Ian H. Witten. "An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links". In: *In Proceedings of AAAI 2008*. 2008.

[Biz+09] Christian Bizer et al. "DBpedia - A Crystallization Point for the Web of Data". In: *Web Semant.* 7.3 (Sept. 2009), pp. 154–165. ISSN: 1570-8268. DOI: 10.1016/j.websem.2009.07.002. URL: http://dx.doi.org/10.1016/j.websem.2009.07.002.

[PM10] Swit Phuvipadawat and Tsuyoshi Murata. "Breaking News Detection and Tracking in Twitter". In: *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*. WI-IAT '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 120–123. ISBN: 978-0-7695-4191-4. DOI: 10.1109/WI-IAT.2010.205. URL: http://dx.doi.org/10.1109/WI-IAT.2010.205.

[CMP11] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. "Information Credibility on Twitter". In: *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. Hyderabad, India: ACM, 2011, pp. 675–684. ISBN: 978-1-4503-0632-4. DOI: 10.1145/1963405.1963500. URL: http://doi.acm.org/10.1145/1963405.1963500.

[HW11] Jonathan Hurlock and Max L. Wilson. "Searching Twitter: Separating the Tweet from the Chaff". In: *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*. 2011. URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/view/2819.

[LEC11] Kyumin Lee, Brian David Eoff, and James Caverlee. "Seven Months with the Devils: A Long-Term Study of Content Polluters on Twitter". In: *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*. 2011. URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/view/2780.

[Dai+13] Joachim Daiber et al. “Improving Efficiency and Accuracy in Multilingual Entity Extraction”. In: *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*. 2013.

[PPK14] Georgios Petkos, Symeon Papadopoulos, and Yiannis Kompatsiaris. “Two-level Message Clustering for Topic Detection in Twitter”. In: *Proceedings of the SNOW 2014 Data Challenge co-located with 23rd International World Wide Web Conference (WWW 2014), Seoul, Korea, April 8, 2014*. 2014, pp. 49–56. URL: `http://ceur-ws.org/Vol-1150/petkos.pdf`.