Report

Rocco Caruso

February 24, 2016

Chapter 1

Stato dell'Arte

1.1 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* [**Allan:2002:TDT:772260**], 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 [**Yang:1998:SRO:290941.290953**], mentre nella seconda si studia la distribuzione delle singole parole e scoprono nuovi eventi raggruppando le parole [**Kleinberg:2002:BHS:775047.775061**] Come evidenziato da [**Yang:1998:SRO:290941.2909**] 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

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

come la rimozione di stopwords, tokenizzazione e stemming. I modelli di rappresentazione di 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* [**Salton:1989:ATP:77013**] 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 [**Yang:1998:SRO:290941.290953**] 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 [**Allan:2002:TDT:772260**, **Yang:1998:SRO:290941.290953**] 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 [**Yang:1998:SRO:290941.290953**] è quella di modificare il vocabolario dei termini in maniera incrementale e modificare l'IDF ogni qual volta viene aggiunto un nuovo documento.

$$idf_t(w) = log_2 \left(\frac{N_t}{df_t(w)} \right) \tag{1.1}$$

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* [**Luo:2007:RRN:1247480.1247536**, **Papka:1999:ONE:897559**] 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 [**Yang:1998:SRO:290941.290953**]. 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 quanto riguarda lo stream di Twitter. Nelle tecniche feature-pivot un evento viene invece modellato come una attività che presenta picchi di frequenza (burst), ovvero un evento è rappresentato dall'insieme di keywords che presentano un burst [**Kleinberg:2002:BHS:775047.775061**]. L'assunzione fatta da queste tecniche è che alcune parole avranno un incremento di utilizzo repentino quando accade un evento. Nel lavoro di [**Kleinberg:2002:BHS:775047.775061**] 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 [**Allan:2002:TDT:772260**] la frequenza delle parole viene modellata tramite una distribuzione bionomiale, dpoi vengono idnvidiuate le burstykeywords sulla base di una soglia euristica, per poi raggrupparle al fine di identificare gli eventi.

1.1.1 Twitter Event Detection

L'attività di Event Detection nei microblogs come Twitter, è concettualmente molto simile all' Event Detection nei media tradizionali. In entrambi i casi, viene dato in input al sistema uno stream di documenti testuali e l'obiettivo è quello di 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, invece, vi possono essere grandi quantità di messaggi privi di significato (pointless babbles) [**DBLP:conf/icwsm/HurlockW11**] e rumors [**Castillo:2011:ICT:1963405.1963500**]. Inoltre le caratteristiche di Twitter e la sua poplarità sono molto allettanti per spammers e altri e altri content polluters [**DBLP:conf/icwsm/LeeEC11**] 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 l'utlizzo di più lingue nel medesimo tweet. Per queste ragioni anche le tecniche tradizionali di natural language processing meno appropriate per i tweet. [**Sankaranarayanan:2009:TNT:1653771.1653781**] hanno realizzato TwitterStand, un sistema per scoprire le ultime notizie da twitter. Per poter distinguere il rumore dalle news, hanno selezionato manualmente 2000 utenti come "Seeders" ovvero utenti che pubblicano su Twitter news come stazioni televisive, giornali, bloggers etc. I tweets non appartenenti a questi seeders, invece, vengono filtrati per mezzo di un classificatore Naive Bayes. Dopo aver filtrato i tweet, viene applicato un algoritmo di clustering incrementale al fine di creare cluster tali che ognuno corrisponda ad una "news". Il modello di rappresentazione utilizzato è quello vettoriale con pesatura tf-idf e la funzione di similarità adottata è quella del coseno. Inoltre viene mantenuta una lista di cluster "attivi" per ridurre il numero di confronti da effettuare. In particolare un cluster viene definito inattivo se la media delle date di pubblicazione dei tweet, non supera i tre giorni. Una volta identificati i topic (news), il sistema cerca di localizzare ciascun cluster, ovvero cerca di assegnare una posizione geografica sia sulla base del contenuto testuale che sui geotag presenti nei tweet. Un altro sistema per scoprire news da twitter è stato proposto [**Phuvipadawat:2010:BND:1913791.1913911**]. In questo lavoro per ridurre il rumore, i tweet vengono innanzitutto campionati utilizzando le streaming API di twitter e fornendo delle specifiche keyword da monitorare (#breakingNews, #breaking news). I tweet raccolti vengono sucessivamente indicizzati tramite Apache Lucene[2]. I tweet simili fra loro vengono raggruppati per poter identificare news. Anche in questo caso la similarità adottata si basa sulla similarità del coseno fra le rappresentazioni tf-idf dei tweet, ma viene assegnato un *boost* per quei termini che corrispondono a

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

nomi propri e per hashtag e username. I nomi propri sono identificati utilizzando lo Stanford Name Entity Recognizer [3] addestrato su un corpora di news tradizionali. I nuovi messaggi saranno inclusi in un cluster se sono simili al primo tweet e ai top-k termini presenti nel cluster. I cluster prodotti vengono poi ordinati sulla base dell'affidabilità (numero di followers) e popolarità (numero di retweet). Nel lavoro di [**DBLP:conf/icwsm/BeckerNG11**] viene posta maggiore attenzione sull'identificazione di eventi reali da Twitter. Il metodo da loro proposto utilizza un algoritmo di clustering incrementale, che raggruppa i tweet simili fra loro e poi classifica i risultanti cluster in eventi real-world o non events. L'algoritmo di clustering utilizzato è il classico algoritmo di incrementale basato su soglia, ogni tweet è rappresentato mediante un boosted tf-idf vector, e viene utilizzata la similarità del coseno per valutare la distanza fra un tweet e il centroide di ogni cluster. Oltre ai classici step di pre-processing come tokenization, stop-word removal e stemming, viene raddoppiato il peso per gli hashtag, poiché sono considerati fortemente indicativi del contenuto del messaggio. Gli autori hanno definito quattro tipologie di feature per i cluster individuati, per poter distinguere fra eventi reali ed da non-event o "twitter center topic" :

1. temporal-features: sono state definite un insieme di caratteristiche temporali per poter caratterizzare il volume dei termini più frequenti all'interno di un cluster.

2. social-features: insieme di feature che caratterizzano il grado di interazione degli utenti nei tweet del cluster come la percentuale di retweet, di replies. L'assunzione fatta è che i cluster contenenti un alta percentuale di retweet potrebbero non contenere informazioni di eventi reali.

3. topical-features: descrivono la coerenza del cluster rispetto ad un topic. L'idea sottostante è che i cluster relativi ad eventi tendono a svilupparsi attorno ad un tema comune al contrario di non-event cluster. Per stimare questa coerenza, gli autori calcolano la media della similarità dei tweet rispetto al centroide.

4. twitter centric-features: queste caratteristiche hanno lo scopo di identificare le attività twitter-centric. Esempi di queste feature sono la percentuale di tweet contenenti hashtag e la percentuale di tweet contenenti l'hashtag più utilizzato. Gli autori presumono che un'alta percentuale della prima stia ad indicare un topic conversazionale.

[3] http://nlp.stanford.edu/software/CRF-NER.shtml

Poichè i cluster evolvono nel tempo, queste feature sono periodicamente aggiornate. Sulla base di queste feature, è stato addestrato una support vector machine (SVM) a partire da un insieme di cluster etichettati, che verrà usata per decidere se un nuovo cluster contiene o meno informazioni relative a eventi reali.