Estrazione periodica di informazioni con Twitter Streaming API, in modo da ottenere un subset di informazioni pubblicate sufficientemente rappresentativo sul topic “S.S. Lazio” nelle ultime partite del campionato 2017-2018, con lo scopo di effettuare su questo la sentiment analysis basata su dizionario

Andrea D’Antonio
Sistemi Intelligenti per Internet
2017 - 2018

Riassunto

Nello svolgere il progetto del corso di “Sistemi Intelligenti per Internet” della laurea magistrale in ingegneria informatica dell’Università degli Studi Roma Tre, è stato sviluppato un sistema capace di estrarre i tweets relativi ad un determinato topic, memorizzarli sotto forma di documenti strutturati in un database NoSQL ed effettuare su questi una sentiment analisys basata su dizionario.

Introduzione

Per estrarre i tweets da Twitter è stata utilizzata la libreria Java *twitter4j*, nello specifico la versione 4.0.6 raggiungibile al sito:

https://mvnrepository.com/artifact/org.twitter4j/twitter4j-core/4.0.6

Inizialmente, si era pensato di utilizzare la versione 4.0.4 della libreria, disponibile sul sito ufficiale di Twitter4j, ma si è notato che questa non trattava correttamente i tweet con più di 140 caratteri. Infatti, nel Novembre del 2017, Twitter ha aumentato il numero di caratteri utilizzabili in un singolo tweet, passando da 140 a 280 caratteri. Questo cambiamento ha portato la versione 4.0.4 di twitter4j a troncare il campo *text* dei tweet con conseguente perdita di molta informazione.

Per memorizzare i tweets estratti si è deciso di utilizzare *MongoDB*, essendo quest'ultimo un database orientato ai documenti e quindi molto indicato per l'applicazione che si intendeva realizzare.

Come linguaggio di programmazione è stato utilizzato *Java* e come IDE *Ecplise*.

Infine, si è realizzata un'interfaccia grafica con l'obiettivo di rendere più semplice l'utilizzo dell'applicativo e permettere a chiunque di utilizzarlo per analizzare topic differenti.

API di Twitter

Per utilizzare le API twitter4j, e di conseguenza l'applicazione che in questa relazione viene presentata, è stato necessario ottenere delle chiavi di accesso da Twitter, raggiungendo la sezione *"Developer"* dal sito del noto social network.

Come prima cosa ci si è dovuti registrare a Twitter, creando un account personale. Successivamente, è stato necessario creare la propria applicazione utilizzando il bottone "*Create New App"*. Dopo di ciò, è stato sufficiente generare l'insieme di chiavi di accesso che permetteranno alle nostre applicazioni di interagire con Twitter. Nello specifico, verranno fornite le seguenti quattro stringhe: *Consumer Key*, *Consumer Secret*, *Access Token e Access Token Secret*.

Modalità di estrazione dei tweet

Per la composizione del subset di tweets si è deciso di utilizzare le API Streaming di Twitter. Il programma sviluppato è stato lanciato ogni giorno, per circa due settimane, restando in ascolto per una durata compresa tra i 30 ed i 40 minuti.

La ricerca dei tweets è stata effettuata mediante l'utilizzo di alcune parole chiave, ottenute dopo un'analisi del topic in questione.

Inoltre, la ricerca è stata filtrata in basa alla lingua dei tweets stessi e al fatto che questi fossero retweets o meno.

Per risolvere il problema dell'ambiguità legata al fatto che alcuni stessi termini rappresentano realtà differenti, come ad esempio la parola *Lazio* utilizzata sia per parlare della squadra di calcio di Serie A che della regione del centro Italia, si è pensato di utilizzare un secondo insieme di keywords con la funzione di escludere i tweets che risultano essere dei falsi positivi. Nel nostro caso specifico, sono risultati essere ottimi termini di filtraggio: *regione*, *m5s*, *pd* e altri termini rappresentanti partiti politici.

Per scegliere l'insieme di keywords utilizzate per l'estrazione si è proceduto nel seguente modo: si è realizzata una funzione specifica per memorizzare in un file testuale il testo di circa 1000 tweets contenenti il termine non ambiguo "S.S. Lazio". Successivamente, sono stati calcolati su questo file i termini più frequenti, escludendo quelli non significativi quali articoli, pronomi, punteggiatura ecc.

Struttura dei documenti memorizzati in MongoDB

Utilizzando le API di Twitter è possibile estrarre decine di informazioni relative ad un certo tweet, molte delle quali sono risultate essere nella quasi totalità dei casi nulle e quindi poco utili.

Si è così deciso di pulire le informazioni estratte con lo scopo di formare documenti ben strutturati da memorizza in MongoDB. Nello specifico, si è deciso di memorizzare dei documenti con la seguente struttura:

{

ID,

USER *(userID, name, screenName, email, description, followersCount, favouritesCount, friendsCount, statusesCount, location, createdAt),*

TWEET *(tweetID, text, lang, isRetweet, retweetCount, isRetweeted, isTruncated, isRetweeted, createdAt, hashtags)*

SENTIMENT *(polarity, label)*

}

dove *hastags* è un array di hashtags presenti nel tweet stesso.

Di seguito è mostrato un esempio di un documento rappresentante un singolo tweet:

Key	Value	Type
(1) ObjectId("5ae8a2070d9a8a2db8ae98b7")	{ 4 fields }	Document
_id	ObjectId("5ae8a2070d9a8a2db8ae98b7")	ObjectId
user	{ 11 fields }	Object
tweet	{ 9 fields }	Object
sentiment	{ 2 fields }	Object

Nel dettaglio, lo *user*:

user	{ 11 fields }	Object
userID	84122838	String
name	Salvatore De Rosa	String
screenName	SalvatoreDR1967	String
email	null	Null
description	null	Null
followersCount	35	Int32
favouritesCount	257	Int32
friendsCount	200	Int32
statusesCount	1.059 (1.1 K)	Int32
location	Napoli	String
createdAt	21/10/2009, 19:45:51	Date

Il *tweet*:

tweet	{ 9 fields }	Object
tweetID	991366948276658176	String
text	@giucruciani @raguzenri Poi se tu parli per partito preso e vuoi che tu e la tua lazio come il napoli,roma,etc.continuino a no	String
lang	it	String
isRetweet	false	Bool
retweetCount	0	Int32
isRetweeted	false	Bool
isTruncated	true	Bool
createdAt	1/5/2018, 19:21:13	Date
hashtags	{ 0 fields }	Object

Il *sentiment*:

sentiment	{ 2 fields }	Object
polarity	-2,67	Double
label	negative	String

Per quanto riguarda il *Sentiment*, si è deciso di utilizzare sia un valore numerico che va da -5 (estremamente negativo) a +5 (estremamente positivo), sia una label, utile in fase di analisi, così definita:

[-5, -1) -> NEGATIVE

[-1, 1] -> NEUTRAL

(1,5] - > POSITIVE

Inoltre, durante la realizzazione del progetto è stato individuato in seguente problema: il testo dei tweets che sono dei retweet veniva troncato a 140 caratteri. Questo è un limite delle API di twitter4j. Per questo si è deciso di scartare questa tipologia di tweet direttamente in fase di estrazione, anche per evitare un'eccessiva ridondanza delle informazioni analizzate.

Sentiment analysis basata su dizionario

Nella realizzazione del progetto si è deciso di utilizzare la sentiment analysis basata su dizionario. Questa opzione è settabile da interfaccia grafica e permette di effettuare il calcolo direttamente durante l'estrazione dei tweets, così da memorizzarli su MongoDB con la polarity già calcolata.

Ogni tweet ricevuto durante lo streaming viene pulito, eliminando ad esempio la punteggiatura, e diviso in stringhe. Per ogni stringa viene cercato nel dizionario (già mappato e inizializzato una sola volta durante l'esecuzione) la rispettiva parola (tramite ricerca sulla chiave di una mappa che è molto efficiente) e calcolata così la polarità del tweet sommando le singole polarità e ignorando per il calcolo le parole non conosciute (ancora non presenti nel dizionario).

Il dizionario dei termini italiani è organizzato nel seguente modo: *termine polarity* (termine *x* seguito da uno spazio bianco e dalla polarità di *x*). Ad esempio, questo è un piccolo estratto del dizionario utilizzato:

adorabile 3
adorato 3
adora 3

Così facendo è stato possibile calcolare un valore complessivo di polarità ed associare a quest'ultimo una label, come illustrato precedentemente.

Costruzione del dizionario

Prima di scegliere una soluzione per fare la sentiment analysis sui tweets, si è effettuata una ricerca approfondita sul web con lo scopo di capire lo stato dell'arte. Da questa è emerso che il limite principale di quasi tutte le soluzioni è la lingua utilizzata: quasi tutte sono pensate esclusivamente per la lingua inglese, alcune per lo spagnolo ed il tedesco, ma quasi nessuna per l'italiano. L'italiano, al momento della ricerca, era supportato solamente da un paio di servizi, non ritenuti molto entusiasmanti.

Inoltre, molti dei servizi analizzati pongono dei limiti sull'utilizzo (per la versione free) ed altri funzionano solamente attraverso http, soluzione esclusa poiché non si voleva limitare la velocità dell'applicazione a quella della rete.

In conclusione, si è preferito realizzare in modo autonomo il dizionario: sono state così definite alcune classi Java (estendendo il codice precedente) per supportare la sentiment analysis offline (direttamente durante lo streaming dei tweets).

Per costruire il dizionario si è partiti da diversi dizionari inglesi, provvisti di polarità, reperiti sul web. Alcuni sono stati trovati su GitHub, altri invece erano rilasciati da università per scopi di ricerca. Questi dizionari sono stati tradotti con un programma di traduzione, mantenendone la struttura e aggiustandoli dove necessario. Successivamente, il dizionario in questione è stato arricchito con altre liste di vocaboli italiani trovati sul web, anche con alcune molto particolari come quelle delle parolacce e simili (dato il topic ne verranno trovate molte), delle emoticon e di termini specifici del dominio studiato. In questo caso le polarità sono state scritte manualmente, con coerenza rispetto le altre già presenti nel dizionario.

Attualmente il dizionario costruito conta circa 2800 vocaboli significativi.

Limiti principali

Nell'utilizzo della sentiment analysis sui tweets basata su dizionario sono stati individuati i seguenti limiti principali:

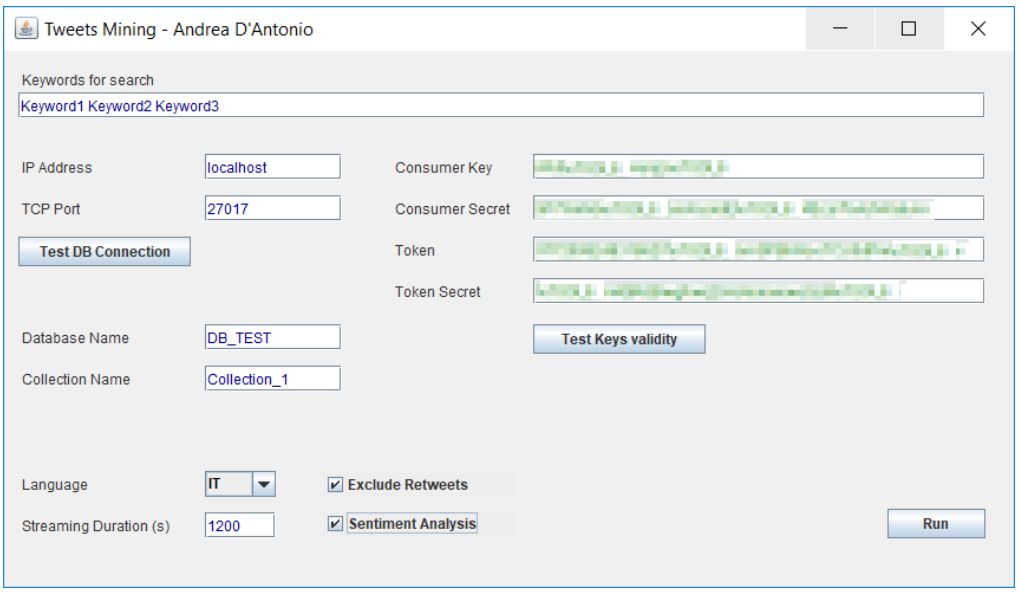
- Sarcasmo e ironia "ingannano" notevolmente il sistema;
- Il dizionario potrebbe essere arricchito per riconoscere più termini significativi;
- Il sistema riconosce solamente singoli vocaboli e non espressioni composte (es. nell'espressione "meglio di niente" vengono considerati "meglio", "di" e "niente" come vocaboli distinti).

Per questo ultimo punto si è anche pensato ad una soluzione, attualmente non implementata. L'idea è quella di sfruttare bi-grammi e 3-grammi per riconoscere tali espressioni, processando il testo del tweet partendo dalle espressioni composte da più termini ed eventualmente escludendole nella fase successiva nel caso di matching.

Questa soluzione al momento non è stata attuata a causa della mancanza di un dizionario contenente questo tipo di espressioni.

Interfaccia Grafica

Di seguito è mostrata l'interfaccia grafica realizzata per il programma sviluppato. Per realizzare la GUI è stato utilizzato *WindowBuilder* di Eclipse.



Nel campo *Keyword for search* è possibile inserire un numero arbitrario di stringhe da utilizzare per l'estrazione dei tweets, queste devono semplicemente essere separate tra loro da uno o più spazi bianchi.

Successivamente, è necessario inserire l'*indirizzo IP* e la *porta TCP* sul quale MongoDB è in ascolto, nell'esempio rispettivamente *localhost* e *27017* (la porta di default di MongoDB).

Fatto questo, si dovrà inserire il nome del database da utilizzare (*Database Name*) e il nome della collezione nel quale memorizzare i tweets estratti (*Collection Name*). Nel caso in cui il database e/o la collezione non fossero già presenti, questi verranno creati prima di iniziare il processo di mining.

In basso a destra è possibile scegliere la lingua dei tweets da ricercare, la durata dello streaming (espressa in secondi), il fatto se si vogliano o meno escludere i retweet ed infine se si vuole attivare o meno la sentiment analysis.

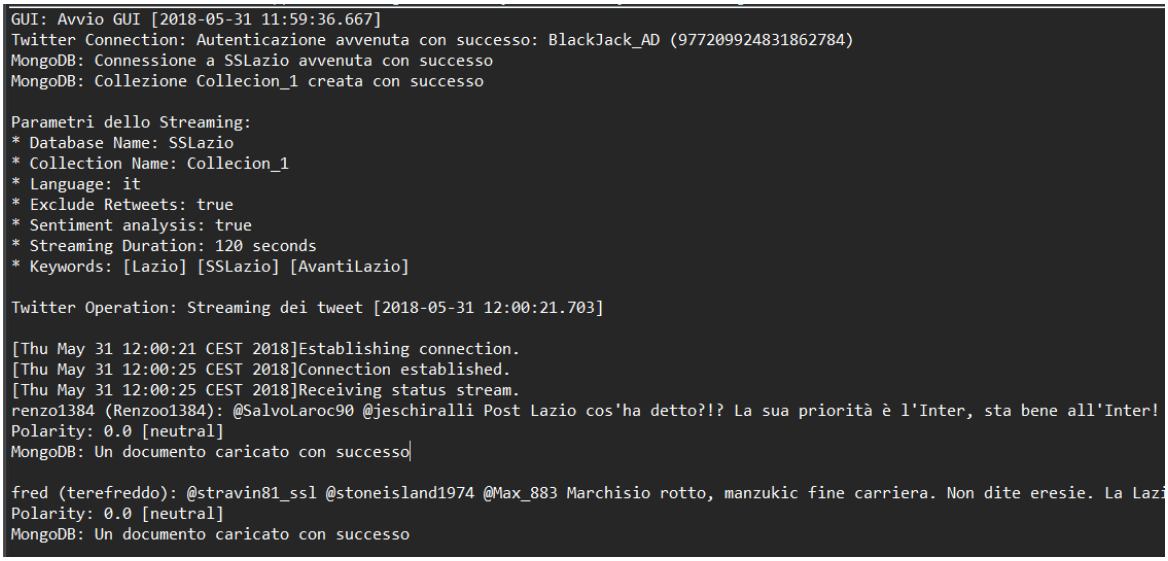
Ovviamente, per lanciare l'applicativo, dovranno essere inserite delle chiavi di accesso valide, ottenibile dal sito di Twitter come precedentemente illustrato. Nell'esempio sono state oscurate per motivi di privacy.

Una volta avviato il programma (bottone *Run*) verrà mostrata una barra di avanzamento per poter osservare il tempo restante al termine del processo.

Nella GUI sono stati anche inseriti dei bottoni appositi per testare lo stato della connessione al database e la validità delle chiavi di accesso utilizzate.

Infine, è possibile osservare dalla console l'esecuzione del programma in ogni suo passo, dove ogni componente dell'applicativo scrive i propri messaggi più importanti, come l'accesso al database, l'estrazione di un nuovo tweet ecc.

Un esempio è mostrato qui sotto.



Risultati

Nelle due settimane di analisi dei tweets è stata effettuata, tutti i giorni, circa mezzora al giorno, la sentiment analysis sui tweets che parlavano di S.S *Lazio*. I tweets sono stati memorizzati in collezioni avanti come nome la data del giorno corrente. Successivamente, sono stati calcolati, attraverso delle query sul database, il valore medio di polarità e la percentuale di tweets positivi, neutrali e negativi per ogni collezione.

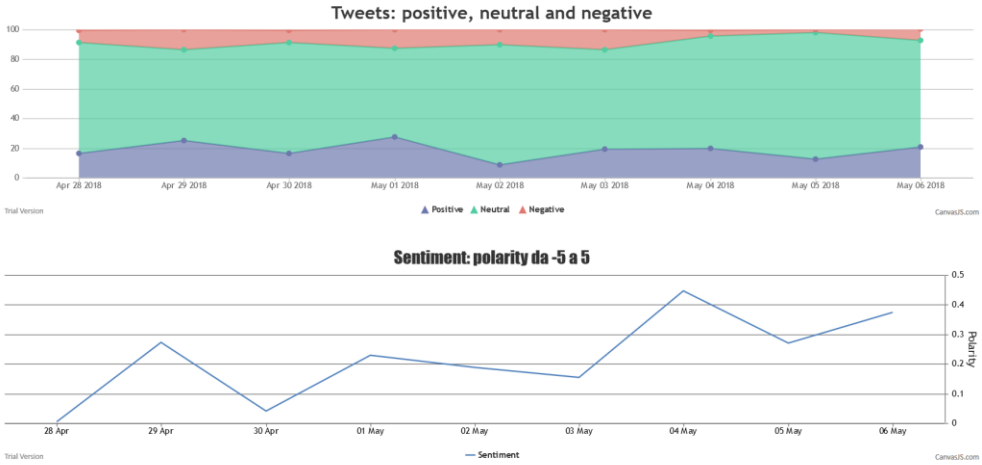
I dati ottenuti sono stati riportati in una pagina html per visualizzare dei grafici riassuntivi utilizzando *javascript*.

Prima di tutto ciò, erano stati presi un certo numero di tweets con l'obiettivo di analizzarli a mano e capire cosa ci si sarebbe dovuto aspettare dall'analisi, per poter fare successivamente un confronto. Quello che è emerso, e che effettivamente è risultato dall'analisi, è che la maggior parte dei tweets sono di stampo giornalistico e risultano essere neutrali. Questo ne è un classico esempio: *"Oggi si gioca Lazio-Atalanta, entra nel vivo la corsa all'Europa"*. Questo fatto spinge notevolmente la polarità delle singole collezioni verso lo 0 (ricordiamo che è stata utilizzata una scala di polarità che va da -5 a 5).

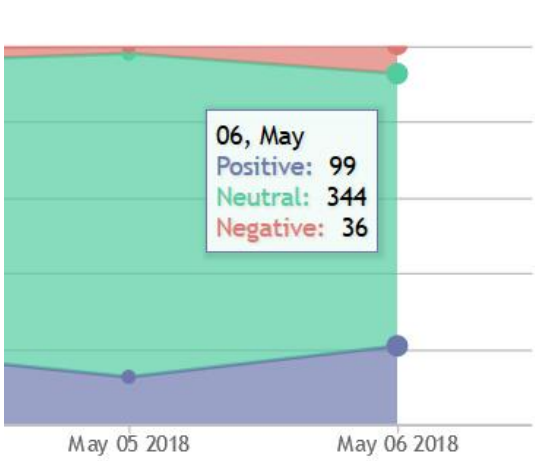
Inoltre, non è stato facile ottenere un numero considerevole di tweets, essendo il topic in questione poco attivo. I tifosi della Lazio non sono poi così tanti (circa 1-2 milioni) ed evidentemente non sono molto "social".

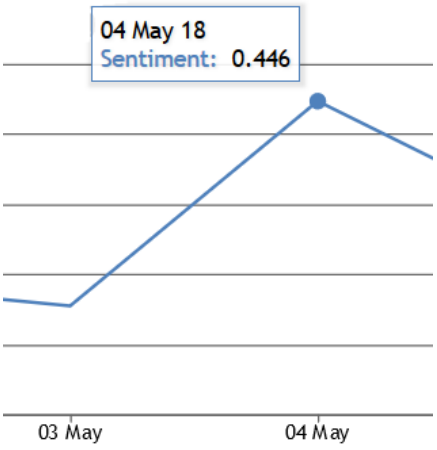
Negli streaming immediatamente successivi alle partite si sono raccolti anche qualche centinaio di tweets con un tempo di ascolto di circa 30-40 minuti, ma in altri giorni non si è riusciti ad andare oltre i 25-30 tweets, davvero pochi per trarre delle conclusioni.

Questi sono alcuni dei grafici ottenuti dall'analisi. Il primo è un *Area Chart Percentage Area 100* il quale rappresenta la percentuale di tweets positivi, neutri e negativi per ogni giorno, il secondo è un classico *Line Chart* con il quale viene mostrato l'andamento della polarity nel tempo.



Per ogni giorno (collezione) è possibile visualizzare i relativi dettagli quantitativi utilizzando in puntatore del mouse, come mostrato nell'immagine qui sotto.





Conclusione

Questo progetto mi ha permesso di studiare ed approfondire diversi argomenti che non avevo mai trattato, e soprattutto di realizzare un sistema utilizzabile per altre analisi di questo tipo. Inoltre, avendo scritto un codice facilmente estendibile e migliorabile, risulta possibile, ad esempio: cambiare il dizionario utilizzato per la sentiment analysis (supportando più lingue), cambiare la lingua dei tweets, aggiungere nuove classi e metodi per effettuare un diverso tipo di sentiment analysis, migliorare il metodo attuale basato sul dizionario, utilizzare le proprie chiavi Twitter per utilizzare e/o testare il programma realizzato ecc.

In conclusione, realizzare questo progetto mi ha permesso di applicare in pratica diverse questioni viste solamente in via teorica e capirne meglio problematiche e punti di forza.

Il codice completo del progetto è raggiungibile su GitHub all'indirizzo: https://github.com/blackjack-ad/TweetMining