Università degli Studi di Milano - Bicocca
Scuola di Scienze
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di laurea in Informatica

Enhancing irony detection with affective information

Relatore: Prof. Elisabetta Fersini

Relazione della prova finale di:
Gianluca Giudice
Matricola 830694

Anno Accademico 2019-2020

Contents

Introduzione

Descrizione del problema

La sentiment analysis è un campo dell'elaborazione del linguaggio naturale (NLP) che si occupa di costruire sistemi per l'analisi di un testo, ha il fine di identificare e classificare l'informazione come il sentimento e l'opinione espressa nello stesso. Si basa sui principali metodi di linguistica computazionale e di analisi testuale. L'analisi del sentiment è utilizzata in molteplici settori: dalla politica ai mercati azionari, dal marketing alla comunicazione, dall'analisi dei social media alla valutazione delle preferenze del consumatore.

Il riconoscimento automatico dell'ironia nei contenuti generati da utenti, è uno dei compiti più complessi per quanto riguarda l'elaborazione del linguaggio naturale. Tuttavia è di fondamentale importanza per tutti i sistemi di sentiment analysis, in quanto facendo uso dell'ironia è possibile invertire completamente la polarità di una propria opinione, facendola passare da positiva a negativa e viceversa. Diventa pertanto cruciale sviluppare dei sistemi di sentiment analysis che sono consapevoli del fenomeno e in grado di riconoscerlo.

L'ironia è un tema studiato in diverse discipline, come la linguistica, filosofia e psicologia, ma è difficile da definire formalmente, soprattutto per questo motivo ne è difficile il riconoscimento. Nonostante ciò ci sono basi teoriche che suggeriscono il ruolo importante della sfera emozionale nell'uso dell'ironia, quindi un fattore chiave per riconoscerlo. Con questo si intende anche un uso indiretto e non esplicito del carico emotivo in ciò che si vuole comunicare.

I social netowork in generale, e twitter nello specifico, sono ampiamente utilizzati come fonte di informazione per comprendere la web reputation di un brand, perciò si prestano bene per la sperimentazione di modelli computazionali per il riconoscimento dell'ironia, essendo di fatti una grande risorsa per quanto riguarda i dati testuali generati da utenti.

Approccio al problema

Si può considerare il riconoscimento dell'ironia come un problema di classificazione. Una frase potrà quindi essere classificata come appartenente alla classe ironica o non ironica. Si noti che per come viene approcciato il problema, l'ironia è considerata al pari del sarcasmo.

Come già detto, twitter è una risorsa che fornisce moltissimi contenuti generati da utenti. Viene sfruttato questo aspetto per creare dei modelli supervisionati di machine learning in grado di apprendere dai dati con lo scopo di riconoscere l'ironia nel testo.

Sintesi dei risultati

Devo mettere le tabelle i grafici? Quanto deve essere sintetico?

Chapter 1

Stato dell'arte

1.1 Approccio supervisionato

1.2 Approccio semi supervisionato

1.3 Approccio non supervisionato

Chapter 2

Sistema realizzato

2.1 Descrizione del sistema proposto

Partendo dai dati a disposizione si utilizzano delle tecniche di preprocessing che consistono nel pulire il testo di partenza e prepararlo per essere dato in input ai vari classificatori. Inoltre si estraggono delle features da ogni messaggio, le quali più sono discriminanti nel riconoscimento dell'ironia, meglio distinguono la classe di appartenenza.

Il dataset viene diviso in due parti: training set e test set. La prima ha dimensione decisamente maggiore e viene utilizzata per far apprendere uno specifico classificate, l'altra per testarlo. Questo è importante in quanto un determinato modello dovrà riconoscere l'ironia in un generico documento e non solo tra quelli a disposizione. Si noti che entrambe le porzioni sono composte sia dal testo già codificato e processato che le features estratte, hanno solo scopi diversi quando si tratta di creare il classificatore.

Vengono quindi creati vari modelli supervisionati di machine learning fornendo in input il dataset di training. Avendo costruito più classificatori che utilizzano diversi algoritmi per apprendere dai dati, è possibile confrontarne le performance per valutare il migliore tra tutti, ovvero quello che meglio distingue tra documenti ironici e non ironici.

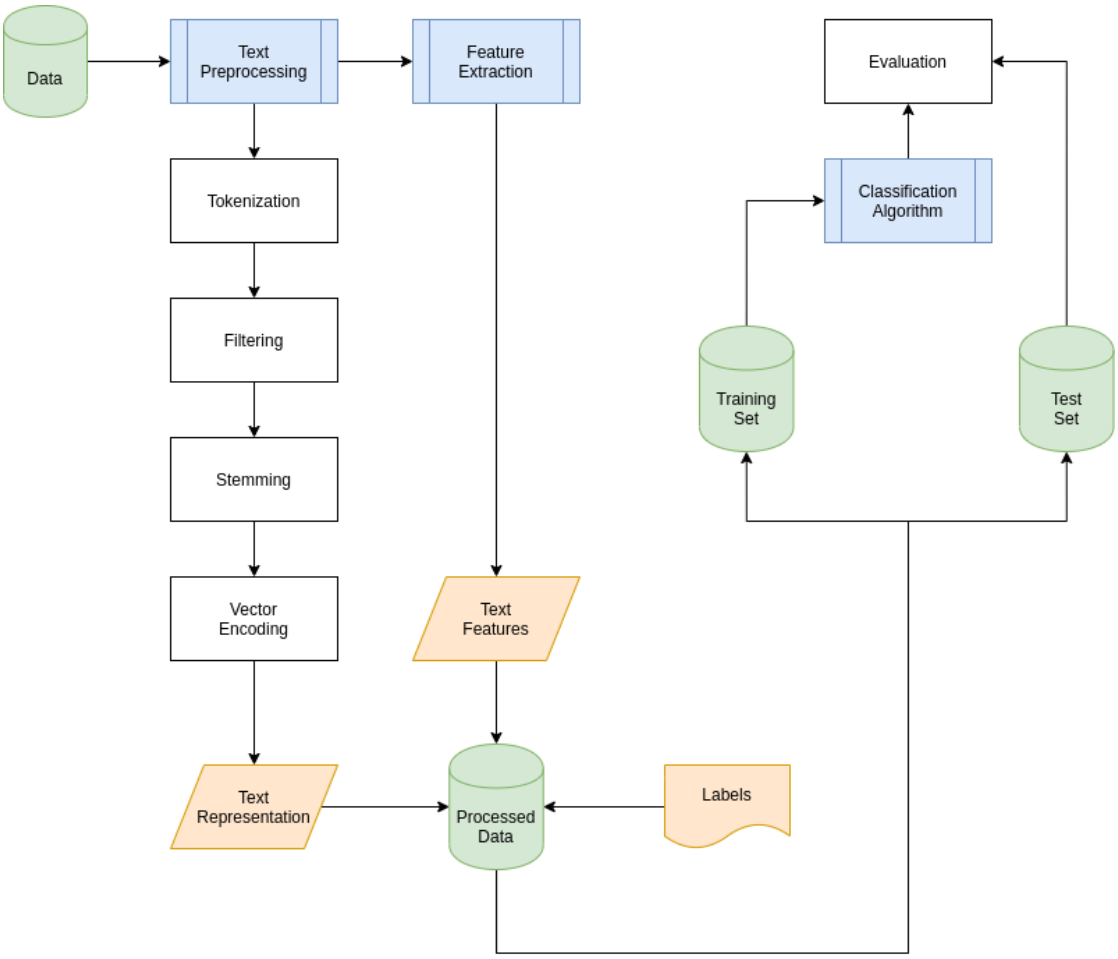

Figure 2.1: Workflow.

2.2 Dati in input

Per procedere con la costruzione di modelli supervisionati per il riconoscimento dell'ironia, è necessaria una collezione di testi (in questo caso tweet) che insieme alle relative etichette associate ad ogni documento, costituisce il dataset. Questo aspetto è cruciale in quanto, una volta essere stati opportunamente processati, saranno proprio questi i dati forniti in input al classificatore, e hanno lo scopo di farlo apprendere e quindi successivamente riconoscere l'ironia in testi che non ha mai visto prima.

Tweet	**Label assocaita**
Lorem ipsum dolor sit amet, consectetur adipiscing elit	ironico
Cras ornare turpis ut odio finibus, viverra porta felis lobortis	ironico
Suspendisse in ex a felis porta convallis	non ironico
Nam laoreet, lacus at ullamcorper iaculis, id interdum nisl elit nec elit.	non ironico

Table 2.1: Esempio di dataset a disposizione.

2.3 Rappresentazione del testo

Il corpora a disposizione è definito "crudo" e non può essere direttamente utilizzato la fase di training. Prima è necessario codificare il testo per ottenere una rappresentazione numerica da dare in input ad un algoritmo di classificazione per la fase di training e test. A questo scopo sono state utilizzate due diverse tecniche

2.3.1 Rappresentazione Bag-of-word

La prima tecnica di rappresentazione utilizzata è boolean bag-of-words. Il testo viene codificato come una matrice booleana costituita dai documenti sulle righe e i tokens sulle colonne. I tokens sono tutte quelle parole appartenenti ad un dizionario costruito a partire dal corpora, tutte le operazioni eseguite sono spiegate di seguito.

2.3.1.1 Tokenization

La tokenizzazione è un passo preliminare per l'elaborazione computazionale del testo. Tokenizzare vuol dire dividere le sequenze di caratteri in unità minime di analisi dette "token". Un approccio potrebbe essere considerare un token come una sequenza di caratteri delimitata da spazi, tuttavia una tale definizione lascia spazio a numerose eccezzione, come ad esempio la presenza di punteggiatura. Per questa operaziona viene pertanto utilizzata la libreria nltkl[1] in grado di gestire correttamente tutti questi casi limite, oltre a poter sfruttare una Twitter-aware tokenization adattandosi quindi bene al dominio in questione.

[1]Natural Language Toolkit: `www.nltk.org`

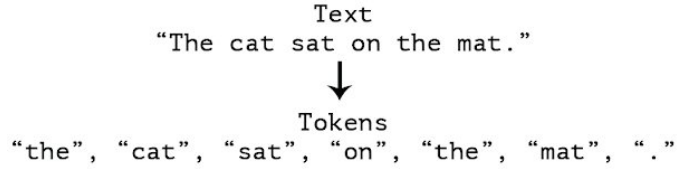

Figure 2.2: Text tokenization.

I vari tokens estratti vengono convertiti in lowercase e sono considerati validi solo se in match con il pattern regex `[a-z]+`. Così facendo non si considerano gli hashtag, la punteggiatura, le emoji e tutto ciò che non è considerato una parola. Si noti che potrebbero verificarsi casi in cui lo stesso token è presente in documenti diversi, questo è ragionevole in quanto le stesse parole possono essere presenti in tweet diversi.

Il risultato ottenuto è una lista di tokens validi contenuti in ogni tweet:

Tweet	**Tokens lowercase**
Lorem ipsum dolor sit amet, consectetur adipiscing elit	$[t_1, t_2, t_3, t_4, t_5]$
Cras ornare turpis ut odio finibus, viverra porta felis lobortis	$[t_6, t_7, t_1, t_8, t_9, t_4]$
Suspendisse in ex a felis porta convallis	$[t_{10}, t_6, t_{11}, t_{12}]$
Nam laoreet, lacus at ullamcorper iaculis.	$[t_{13}, t_{14}, t_8, t_{15}]$

Table 2.2: Esempio di tokens associati ad ogni tweet.

Da qui si crea l'insieme di tokens univoci.

2.3.1.2 Filtering

L'insieme di termini univoci ottenuto è filtrato in base ai seguenti criteri:

- **Occorrenza minimia** fissato valore di threshold

  Viene fissato un valore di soglia (Es. 10) e si scartano tutte quelle parole che compaiono tra tutti i testi meno del valore scelto.

- **Stopwords + RT**

  Vengono scartate tutte le parole più comuni che sono generalmente presenti in una frase. Queste sono un insieme predefinito contenente parole come gli articoli, proposizioni, pronomi e verbi ausiliari.

  Es: $\{a,\ the,\ of,\ is,\ into,\ it,\ ...\}$

  Dato il dominio preso in considerazione, è possibile che un utente compia l'operazione di ReTweet. Questo è codificato nei messaggi dall'abbreviazione *"rt"*. Non ritenendolo un aspetto rilevante per il riconoscimento dell'ironia, viene anch'essa considerata una stopwords, e di conseguenza non trattato come token valido.

- **Irony** e **Ironic**

  Essendo il dataset a disposizione etichettato mediante la tecnica self-tagging (section 3.1), è presente l'hashtag *#irony* in tutti i tweet ironici. Pertanto nell'insieme dei tokens validi verranno escluse le parole *irony* e *ironic*, così da non avere un bias nei dati sotto questo aspetto. Nel caso contrario i dati usati per creare i modelli avrebbero a disposizione una

componente che ben distingue i tweet ironici, ma che nella realtà non sarebbe presente, in quanto non è decisamente corretto assumere che ogni testo ironico contenga una delle due parole.

2.3.1.3 Stemming

Lo stemming è il processo di riduzione della forma flessa di una parola alla sua forma radice.

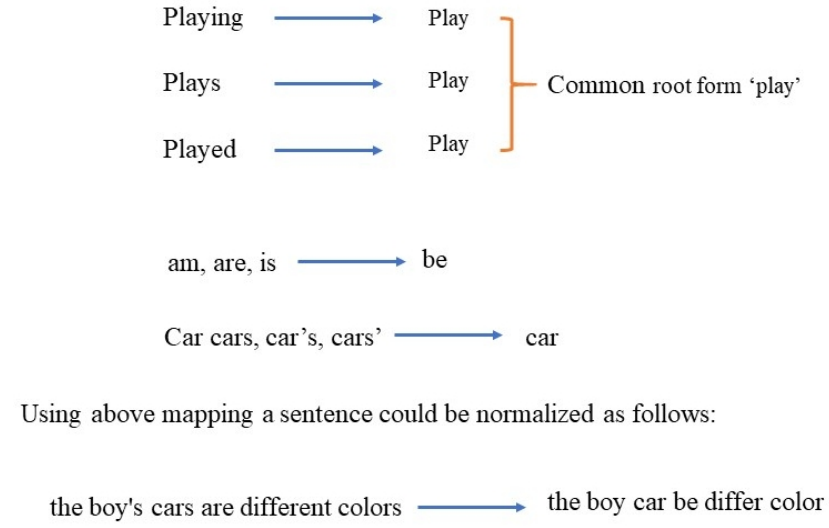

Figure 2.3: Words stemming.
`www.datacamp.com/community/tutorials/stemming-lemmatization-python`.

Con questa tecnica è possibile ridurre il numero di tokens, dal momento che le stesse parole con suffisso diverso verranno mappate nella medesima radice. Inoltre non è necessario che una parola dopo il processo di stemming venga trasformata in un'altra di senso compiuto, quello che conta è la semantica del token.

2.3.1.4 Vector encoding

L'insieme delle parole ottenute costituisce il dizionario del corpora. Si procede costruendo una matrice booleana composta dai tweets sulle righe e le parole del dizionario sulle colonne.

Sia $D = \{w_1,\ w_2, ...,\ w_m\}$ il dizionario individuato dall'insieme delle n parole univoche.
Sia $T_i = \{t_{i0}\ , t_{i1},\ ...,\ t_{ik}\},\ 1 \leq i \leq n$ l'i-esimo tweet composto da k tokens opportunamente stemmatizzati.

Viene definita la matrice $M_{n,m}$

$$M[i,j] = \begin{cases} 1 & \text{se } w_j \in T_i \\ 0 & \text{altrimenti} \end{cases} \quad t.c\ 1 \leq i \leq n,\ 1 \leq j \leq m$$

2.3.2 Rappresentazione mediante transformer

2.3.2.1 BERT

2.3.2.2 Sentence-BERT

2.4 Caratteristiche linguistiche

La scelta di features ad alto contenuto informativo ed indipendenti tra loro, è un passo fondamentale per un corretto riconoscimento di pattern nei dati. A questo scopo vengono estratte delle caratteristiche linguistiche dal testo che sono rilevanti per il riconoscimento dell'ironia. Più le caratteristiche sono discriminanti quando si esprime un messaggio ironico, meglio si potranno distinguere le due classi. Queste features, insieme alla codifica testuale, verranno usate per creare il modello.

2.4.1 Pragmatic particles

Per catturare il senso non letterale racchiuso in ogni messaggio, vengono considerati alcuni elementi linguistici di seguito elencati.

2.4.1.1 Emoticons

Le emoticons sono riproduzioni stilizzate di quelle principali espressioni facciali umane che esprimono un'emozione. In accordo con l'assunzione che esista una relazione tra il sentimento espresso e l'ironia, queste vengono distinte in due categorie:

- Positive

  E.g.: ':-)', ':-D'

- Negative

  E.g.: ':-(', ':=('

2.4.1.2 Acronimi

Gli acronimi sono un ulteriore strumento della comunicazione non verbale. Come le emoticons vengono presi in considerazione dal momento che possono esprire sentimenti positivi o negativi, infatti possiamo anch'essi dividerli in due categoire:

- Positivi

  E.g.: 'ROLF' (Rolling On Floor Laughing), 'LHO' (Laughing Head Off)

- Negativi

  E.g: 'BM' (Bad Manner)

2.4.1.3 Espressioni onomatopeiche

L'uso di espressioni onomatopeiche come 'boom' o 'clap' possono essere utilizzate nell'esprimere un messaggio ironico. Tuttavia in questo caso viene presa in considerazione solo la frequenza di queste espressioni, al contrario di come visto in precedenza per le emoticons e gli acronimi, dove si valutava anche la polarità.

2.4.1.4 Punteggiatura

La punteggiatura nei social media non segue le convenzioni ortografiche:
E.g.: 'Do you hear us now ?????'
Per questa ragione viene contata la frequenza di virgole, punti di domanda e punti esclamativi.

2.4.1.5 Estrazione particelle pragmatiche

Di seguite viene mostrato un esempio di come le particelle pragmatiche vengono estratte da un messaggio:

```
Tweet: ':D lmao! RT @KarlDetkenProDJ iPad Humor Steve, I'ma let you finish,
       but Moses had the greatest  tablet of all time ;) clap clap'
Emot (-, +)  >>> [0, 1]
Init (-, +)  >>> [0, 1]
Onom (#)     >>> [2]
Punct        >>> {',': 2, '!': 1, '?': 0}
```

Listing 2.1: Esempio di tweet processato per estrarre le particelle pragmatiche.

2.4.2 POS

La struttura delle frasi che costituiscono il tweet, può essere un indicatore per il riconoscimento dell'ironia. A questo scopo viene tenuta in considerazione la frequenza dei POS (part-of-speech) tag relativi ai tokens parole presenti in un messaggi.

Per associare i POS tag ad ogni tweet viene utilizzato un POS tagger[2] supervisionato specifico per il dominio considerato. Si procede con una tokenizzazione del tweet per poi associare ad ognuno di questi il corretto tag. Di seguito la lista di tags considerati:

Nominale

- **N**: Nome comune
- **O**: Pronome personale
- **^**: Nome proprio
- **S**: Nominale + Possessiva
- **Z**: Nome proprio + Possessiva

Open-class words

- **V**: Verbo
- **A**: Aggettivo
- **R**: Avverbio
- **!**: Interjection

Closed-class words

- **D**: Determiner
- **P**: Pre/Post-posizione; Congiunzione subordinante
- **&**: Congiunzione coordinante
- **T**: Verb particles
- **X**: Existential there

Composti

- **L**: Nominale + Verbo (e.g.: I'm)
- **M**: Nome proprio + Verbo
- **Y**: X + Verbo

[2] ARK Twitter Part-of-Speech Tagger: `www.ark.cs.cmu.edu/TweetNLP/`

Di seguito è mostrato un esempio di POS tags associati al tweet passando per i tokens:

```
Tweet: 'Need a second opinion? Who gave you the first one?'
Tokens  >>> [Need], [a], [second], [opinion], [?], [Who],
            [gave], [you], [the], [first], [one], [?]
Tags    >>> ['V', 'D', 'A', 'N', ',', 'O', 'V', 'O', 'D', 'A', '$', ',']
Freq.   >>> {'N': 1, 'O': 2, '^': 0, 'S': 0, 'Z': 0, 'V': 2,
             'A': 2, 'R': 0, '!': 0, 'D': 2, 'P': 0, '&': 0,
             'T': 0, 'X': 0, 'L': 0, 'M': 0, 'Y': 0}
```

Listing 2.2: Esempio di tweet con POS tag associte.

Come si può notare, il POS tagger associa dei tag che non sono stati elencati sopra, i verranno semplicemente ignorati e non utilizzati come features per il modello.

2.4.3 EMOT

2.5 Modelli supervisionati

2.5.1 Alberi di decisione

2.5.2 Support Vector Machine

2.5.3 Naive Bayes

2.5.4 Bayesian Network

2.6 Strumenti utilizzati

2.6.1 Scikit-learn

2.6.2 Weka

Chapter 3

Campagna sperimentale

3.1 Dataset

Fare notare che è in inglese

Trattandosi di un modello supervisionato, è richiesto che il dataset sia etichettato. Per associare una specifica label ai messaggi generati dagli utenti si possono seguire due strade:

- **Self-Tagging**

  Twitter mette a disposizione l'utilizzo degli hashtag nei messaggi. Assumendo che un utente utilizzi l'hashtag *#irony* con la volontà di esprimere ironia, è facile collezionare una serie di tweet etichettati come ironici.

- **Crowdsourcing**

  I vari tweet vengono etichettati manualmente da alcune persone

Nel caso specifico, viene utilizzato il dataset *TwReyes2013*, composto da 40,000 tweet accumulati usando la tecnica self-tagging. Vengono quindi considerati 4 hashtag diversi:

Numero	**Hashtag**	**Label assocaita**
10,000	*#irony*	ironico
10,000	*#education*	non ironico
10,000	*#humor*	non ironico
10,000	*#politics*	non ironico

Table 3.1: Hashtag e label associate

3.2 Misure di performance

3.3 BOW + Caratteristiche linguistiche

3.4 BERT + Caratteristiche linguistiche

3.5 SBERT + Caratteristiche linguistiche

3.6 Analisi lessico con PCA

Chapter 4

Conclusioni e sviluppi futuri