

**Misure di similarità tra hashtags mediante analisi dei contenuti testo o video associati al tag e analisi semantica (Word2Vec).**

Martina Della Corte

Matricola: 451656

Anno Accademico: 2017/2018

# **Introduzione**

La seguente relazione ha lo scopo di descrivere brevemente il progetto svolto per i corsi di ‘Machine Learning’ e di ‘Sistemi Intelligenti per Internet’. In particolare, descriveremo l’obiettivo del progetto, le tecnologie scelte per la sua implementazione, le funzioni sviluppate e analizzeremo alcuni dei risultati ottenuti dalla valutazione del modello progettato.

**Word Embedding e Word2Vec**

Il Word Embedding è un approccio che consente di mappare le parole all'interno di un testo in un valore numerico('embed'). Ogni parola è rappresentata da un punto nello spazio e questi punti vengono appresi e spostati in base alle parole che circondano la parola target. Uno degli algoritmi utilizzati per l'apprendimento del word embedding è Word2vec. Word2vec è implementato come una rete neurale artificiale a due strati. Tale algoritmo riceve in input un Vocabolario di parole e restituisce un insieme di vettori che rappresentano la distribuzione semantica delle parole nel testo. Ogni parola quindi viene rappresentata da un vettore, cosa che ci consente di rappresentare tale vettore nello spazio. In questo spazio le parole saranno più vicine se riconosciute come semanticamente più simili.

**Obiettivo del progetto**

Gli utenti dei social media sono soliti associare ai propri contenuti multimediali, diversi hashtags. Tale pratica di etichettatura risulta estremamente utile alla comprensione semantica di tali contenuti. L’obiettivo del progetto è quindi quello della misurazione della similarità tra hashtags mediante l’analisi semantica e l’analisi dei contenuti multimediali, quali video e foto, associati a tali tag. A tal proposito, è stato richiesto istanziare uno dei tanti tool a disposizione per tradurre il linguaggio naturale in vettori in spazi semantici, utilizzando come dati di input un opportuno dataset <risorsa, {hashtag}>.

**Tecnologie utilizzate**

**Instaloader**

Instaloader è un ‘command-line’ tool che consente di:

* scaricare da profili pubblici e privati di Instagram, storie, foto e video postati e i media salvati.
* Scaricare hashtags, geotags e le captions associate ad ogni media.
* Permette di definire filtri e di specificare dove memorizzare i media scaricati.

Instaloader memorizza le risorse scaricate secondo il seguente protocollo:

* All’interno della directory di instaloader crea una nuova cartella che rinomina con il nome del profilo Instagram scelto;
* All’interno di tale cartella vengono memorizzati il file della risorsa e un file di testo, contenente la caption associata alla risorsa, omonimo.
* Inoltre, memorizza con lo stesso nome foto, caption e metadati della risorsa.

Il tool è un progetto open source ed è possibile visionare il codice e consultare la documentazione al link: <https://github.com/instaloader/instaloader>.

**Gensim**

Gensim è una libreria Python open source progettata per estrarre automaticamente categorie (o aree) semantiche dai documenti, in modo efficiente. Mette a disposizione implementazioni di diversi algoritmi non supervisionati per la generazione di spazi vettoriali. Per ulteriori informazioni consultare il link: <https://radimrehurek.com/gensim/intro.html>

**Funzioni Implementate**

**getHashtags(dataset\_path,regex)**

la funzione ‘getHashtags’ accetta due parametri in input:

* dataset \_path: il path dove è memorizzato il nostro dataset
* regex: la regular expression che ci consente di estrarre da un testo solo gli hashtag(es. #lake)

La funzione ha l’obiettivo di recuperare da ogni caption associata ad ogni risorsa risorsa del dataset, la lista degli hashtags. Data funzione ritorna una lista di liste: ogni lista contiene gli hashtag associati ad una data risorsa. All’interno della funzione viene inoltre popolata una mappa(è possibile salvare il suo contenuto all’interno di un file di testo) che ha:

* per chiave la posizione in cui si trova la lista di hashtag
* per valore il path per accedere alla risorsa associata a tale lista

**trainModel(list\_of\_lists,model\_path,mySize,myWindow,myMin\_count,myWorkers,myEpochs)**

La funzione ha lo scopo di utilizzare Gensim e in particolare l’algoritmo Word2Vec al fine di costruire il vocabolario a partire dalla nostra lista di liste di hashtags e addestrare il modello progettato secondo i parametri che la funzione riceve in input. Di seguito uno snippet del codice che implementa quanto detto:

**model = gensim.models.Word2Vec(list\_of\_lists, size= mySize, window=myWindow,min\_count=myMin\_count,workers=myWorkers)**

**model.train(list\_of\_lists, total\_examples=len(list\_of\_lists),epochs=myEpochs)**

Andiamo a descrivere il significato di ogni parametro più nel dettaglio:

* size: la dimensione del vettore denso per rappresentare ogni hashtag (cioè il contesto o le parole vicine). Se si dispone di dati limitati, la dimensione dovrebbe essere un valore molto più piccolo poiché si avranno solo tanti vicini unici per un determinato hashtag. Se si dispone di molti hashtag, è bene sperimentare con varie dimensioni. Il valore 150 sembra funzionare piuttosto bene;
* window: la distanza massima tra l’hashtag target e l’hashtag adiacente. Se la posizione del vicino è maggiore della larghezza massima della finestra a sinistra o a destra, quindi, alcuni vicini non saranno considerati correlati all’hashtag target. In teoria, una finestra più piccola dovrebbe darti termini più correlati. Ancora una volta, se i tuoi dati non sono sparsi, la dimensione della finestra non dovrebbe essere eccessiva, purché non eccessivamente stretta o eccessivamente ampia;
* min\_count: minimo numero di occorrenze di un hashtag nelle liste. Il modello ignora tutti gli hashtag che non soddisfano il min\_count. Gli Hashtag che non sono frequenti di solito non sono importanti, quindi possono essere scartati. A meno che il set di dati non sia molto piccolo, ciò non influisce sul modello in termini di risultati finali;
* workers: il numero di thread al lavoro.

**loadModel(model\_path)**

Semplice funzione che recupera il modello addestrato, dato il path in input.

**main()**

Nel main andiamo a chiamare le funzioni implementate ed effettuare diverse valutazioni sul modello addestrato.

**Dataset 1: 59K Hashtags( Diversi Target)**

In una prima fase è stato addestrato il modello con un dataset contenente risorse di circa 120 venti profili Instagram tra loro eterogenei. Escludendo risorse le cui caption risultano essere prive di hashtags, abbiamo 59.000 hashtag univoci.

I valori dei parametri utilizzati in questa prima fase sono:

* size: 150
* window: 10
* min\_count: 5
* workers: 10

Di seguito alcuni dei risultati ottenuti:



