UNIVERSITÀ DEGLI STUDI DI BARI
"ALDO MORO"

Dipartimento di Informatica
Corso di Laurea Triennale in Informatica

Tesi di Laurea
in
Metodi Avanzati di Programmazione

Analisi di Social Media per la Scoperta di Pattern di Interazione

Relatori:
dr. Corrado Loglisci
prof. Donato Malerba

Correlatore:
dott. Angelo Impedovo

Laureando:
Donato Meoli

Anno accademico 2016-2017

La dignità non consiste nel possedere onori, ma nella coscienza di meritarli.
ARISTOTELE

Sommario

Obiettivo generale del presente progetto è quello di realizzare strumenti computazionali in grado di modellare comunità collaborative attraverso l'analisi di dati, prodotti e memorizzati dalle medesime comunità. In un contesto fortemente caratterizzato dall'interazione tra gli uomini, benchè mediata da tecnologie, una comunità rappresenta un dominio che evolve nel tempo, i cui cambiamenti possono essere molteplici e possono riguardare fattori interni o esterni alla comunità. Lo studio dei cambiamenti diventa cruciale per caratterizzare l'evoluzione della comunità e modellarla, quindi, in maniera completa. A tal proposito si condurrà una attività mirata a sintetizzare metodi di analisi che siano in grado di scoprire le interazioni tra partecipanti, a partire da dati riguardanti relazioni sociali tra questi. Molte delle informazioni di una comunità sono prodotte in forma non strutturata e in linguaggio naturale. Tipicamente, la loro comunicazione avviene tramite strumenti di messaggistica istantanea o su forum di discussione tematici. Questo significa che molte delle informazioni assunte non sono apertamente esplicitate e direttamente accessibili, ma riportate in testo scritto. Nasce quindi la necessità di investigare su approcci di elaborazione del linguaggio naturale per l'identificazione di relazioni tra i partecipanti.

Indice

Elenco delle figure

Elenco dei Listati

Parte I

Contesto & Obiettivi

Capitolo 1

Comunità Online

Le comunità online sono una realtà collaborativa che trovano spazio in molteplici ambiti: dallo sviluppo di progetti software in comunità aperte (es. MySQL developer community) o chiuse, all'interno di organizzazioni aziendali (distribuite sul territorio), alla ricerca di lavoro e pubblicazione di profili professionali (es. LinkedIn), alla condivisione di informazioni di singoli individui (es. Facebook). Le comunità online, in quanto tali, agevolano collaborazioni e scambio di conoscenza, abbattendo spesso barriere organizzative e geografiche. Molte di esse interagiscono tramite strumenti di comunicazione e dispongono anche di soluzioni di data storage che memorizzano informazioni di vario genere, relative alla comunità medesima.

1.1 Obiettivi

Con la finalità di modellare comunità collaborative attraverso strumenti analitici, si considerino gli specifici sotto-obiettivi scientifico-tecnologici di seguito elencati:

1. *Identificazione e selezione di sorgenti dati prodotti da comunità*: il recente paradigma di data science suggerisce di investigare sorgenti dati da cui estrarre informazioni utili per la modellazione computazionale della comunità. In tal senso, saranno considerate sorgenti di dati non strutturate prodotte da piattaforme tecnologiche che supportano comunità digitali. Una strategia data-driven è innovativa rispetto all'usuale approccio basato sull'intervento di esperti con forte background sociologico. L'innovatività diventa più significativa dal momento che l'analisi considererà osservazioni puntuali della comunità, nella forma di comunicazioni e messaggi prodotte/consumate al suo interno;

2. *Modellazione della comunità con strumenti di analisi*: considerando che una comunità è un dominio complesso caratterizzato da partecipanti, relazioni ed

interazioni tra questi e ruoli da loro ricoperti, una prima problematica da investigare è lo studio di modelli computazionali adeguati a rappresentare le diverse componenti di una comunità. In questo senso si investigheranno soluzioni per l'estrazione di informazioni che caratterizzano una comunità dalle sorgenti precedentemente identificate e selezionate. In questo senso, l'innovazione consisterà nell'uso di strumenti di elaborazione del linguaggio naturale per analizzare comunicazioni e messaggi al fine di individuare informazioni che caratterizzano i singoli individui e le loro interazioni. Ci si propone di progettare ed implementare strumenti prototipali in grado di sfruttare le informazioni precedentemente estratte per fornire un modello computazionale della comunità. L'innovazione risiede nel problema e nella soluzione computazionale. Si ritiene innovativo analizzare lo storico di una comunità per trarne una caratterizzazione rispetto alla struttura, così come progettare strumenti di analisi dei dati per caratterizzare una comunità attraverso la scoperta delle sue collaborazioni intrinseche e dei principali pattern di interazione soggetti a cambiamenti;

3. *Sperimentazione del prototipo*: ci si propone di sperimentare il prototipo sulle sorgenti dati di partenza al fine di raccogliere specifiche computazionali quantitative attraverso consolidati protocolli di validazione empirica. Una prima innovazione sta nella definizione di specifiche quantitative che, a differenza di quelle qualitative, non risultano essere investigate per la modellazione di comunità digitali. Una seconda strada prevede l'analisi di aspetti tecnici e tecnologici del prototipo per renderlo generale ed applicabile anche a comunità digitali differenti da quella/e dell'obiettivo specifico.

1.2 Risultati

Si prevedono risultati intermedi e finali nella forma di deliverable digitali, pubblicazioni scientifiche e prototipi software organizzati attraverso i seguenti punti:

1. *Sorgenti dati prodotte da comunità collaborative*: trattasi di un report tecnico strutturato in due parti:

   (a) la prima sarà incentrata sulla descrizione delle sorgenti dati, prodotte da comunità digitali e accessibili pubblicamente, dal punto di vista della tipologia della comunità e dal punto di vista della tipologia delle informazioni contenute;

   (b) la seconda parte verterà su progettazione ed implementazione di un prototipo software in grado di analizzare comunicazioni e messaggi dalle

sorgenti dati, identificando informazioni di interesse per la costruzione di un modello a grafo della comunità;

2. *Scoperta di pattern di interazione*: trattasi di un report tecnico in cui viene dettagliata la progettazione ed implementazione di un prototipo software in grado di analizzare i dati in forma di grafo, al fine di scoprire conoscenza nella forma di pattern di interazione. Il risultato sarà disponibile anche in forma di prototipo software multipiattaforma open-source;

3. *Validazione e applicazione dei prototipi*: trattasi di un report tecnico in cui viene descritta la sessione sperimentale volta a testare i prototipi sui dati prodotti al risultato 1. Il report fornirà:

   (a) specifiche quantitative ed interpretazioni qualitative di pattern di interazione risultanti dalla sperimentazione;

   (b) specifiche tecniche ed eventuali risultati sull'applicazione dei prototipi a comunità digitali differenti da quelle di riferimento usate nel progetto;

4. *Sito web*: trattasi di un contenitore, pubblicamente accessibile, dei vari prodotti del progetto, comprensivi delle relative pubblicazioni scientifiche.

Parte II

Costruzione di Grafi da Social Media

Capitolo 2

Estrazione di Grafi Basati sul Contenuto

2.1 Pipeline di Elaborazione del Linguaggio Naturale sul Contenuto

Il natural language processing è una branca dell'intelligenza artificiale che concerne l'elaborazione automatica del linguaggio naturale. Trattandosi di un processo particolarmente complesso a causa delle caratteristiche intrinseche di ambiguità del linguaggio umano, tra cui sinonimia e polisemia, esso viene suddiviso in fasi diverse. Lo scopo è quello di determinare la funzione delle singole parole, che vengono suddivise in classi sintattiche (nomi, verbi, aggettivi e avverbi), in base alla loro funzione all'interno di una frase.

Lo Stanford NLP[1] Group è uno dei gruppi di ricerca di maggiore importanza nel settore dell'elaborazione del linguaggio naturale ed offre validi strumenti per realizzare tale scopo. Attraverso le API Java del modulo software CoreNLP è possibile analizzare il dataset corpus per poter distinguerne le parole (tokenization), lemmatizzarle, nonchè ridurle alla loro forma canonica, e determinarne la funzione (PoS[2] tagging) all'interno della frase. È così possibile classificarle nelle categorie sintattiche di nostro interesse che includono:

Nomi

(i) *NN*: singolare;

[1]Natural Language Processing

[2]Part of Speech

(ii) *NNP*: proprio singolare;

(iii) *NNS*: plurale;

(iv) *NNPS*: proprio plurale.

Verbi

(i) *VB*: forma base;

(ii) *VBD*: passato;

(iii) *VBG*: gerundio o participio presente;

(iv) *VBN*: participio passato;

(v) *VBP*: presente singolare (esclusa terza persona singolare);

(vi) *VBZ*: presente, terza persona singolare.

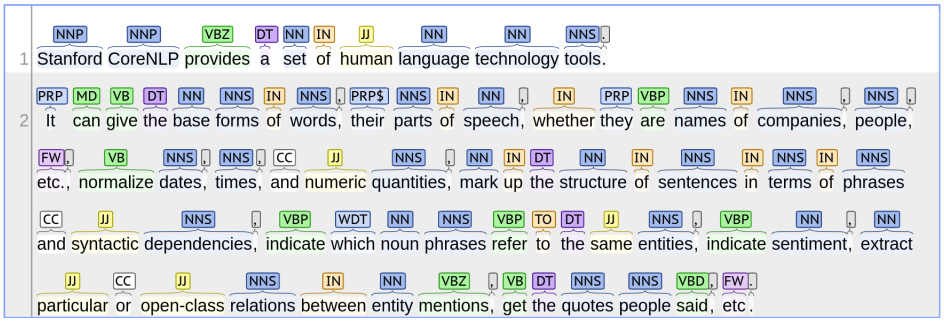

Figura 2.1. Esempio di part-of-speech tagging.

2.2 Identificazione di Archi Basati su Similarità

Il risultato della fase di NLP, costituita dai verbi e dai nomi lemmatizzati, si presta ad essere utilizzato per l'estrazione di caratteristiche linguistiche.

2.2.1 Archi Basati su Similarità Lessicale

La similarità lessicale tra i messaggi è stata calcolata grazie ad una versione modificata del Lexical Match Algorithm (LMA) [1] che considera i messaggi che si presuppone possano essere simili. Il LMA integra il Vector Space Model (VSM), nonchè uno dei più popolari metodi usati per identificare somiglianze lessicali. La formula del LMA per calcolare la similarità lessicale produce un valore compreso tra 0 e 1 e può essere formalizzata come segue:

$$\sum_{i=0}^{LenX} \sum_{j=0}^{LenY} \sum_{if POS(X_i)=POS(Y_j)} \frac{TF_{X_i} + TF_{Y_j}}{DF_{X_i} + DF_{Y_j}} \cdot (LenX \cdot LenY)^{-1} \tag{2.1}$$

dove:

(i) X ed Y sono i due messaggi;

(ii) $LenX$ e $LenY$ sono il numero di verbi e nomi contenuti rispettivamente nei messaggi X ed Y;

(iii) POS è il part-of-speech tag;

(iv) X_i ed Y_j si riferiscono rispettivamente agli i-esimi e j-esimi nomi o verbi lemmatizzati contenuti nei messaggi X ed Y;

(v) TF è il term frequency e DF è il document frequency.

2.2.2 Archi Basati su Similarità Semantica

La similaritè semantica è una metrica definita su un insieme di documenti o termini, in cui l'idea della distanza tra loro si basa sulla similarità del loro significato o contenuto semantico rispetto alla similarità che può essere stimata per quanto riguarda la loro rappresentazione sintattica. Questi sono strumenti matematici usati per stimare la forza della relazione semantica tra unità di linguaggio, concetti o istanze, attraverso una descrizione numerica ottenuta in base al confronto di informazioni che supportano il loro significato o descrivono la loro natura. Il termine similarità semantica è spesso confuso con la relazione semantica. La relazione semantica include qualsiasi relazione tra due termini, mentre la similarità semantica include solo le relazioni "is a".

Computazionalmente, la similarità semantica può essere stimata definendo una similarità topologica, utilizzando le ontologie per definire la distanza tra termini/-concetti. Ad esempio, una metrica per il confronto di concetti ordinati in un insieme parzialmente ordinato e rappresentati come nodi di un grafo aciclico e diretto (ad

esempio una tassonomia), sarebbe il percorso più breve che collega i due nodi concettuali. Sulla base di analisi testuali, la correlazione semantica tra unità linguistiche può anche essere stimata utilizzando mezzi statistici come il VSM [3] per correlare parole e contesti testuali da un corpus testuale adatto.

Il concetto di similarità semantica è più specifico della relazione semantica, in quanto quest'ultimo include concetti come l'antonimia e la meronimia, mentre la similarità no. Tuttavia, gran parte della letteratura usa questi termini in modo intercambiabile, insieme a termini come la distanza semantica. In sostanza, la similarità semantica, la distanza semantica e la relazione semantica hanno in comune il fatto che il loro risultato è solitamente un valore compreso tra -1 e 1 o tra 0 e 1, dove 1 indica una similarità estremamente alta.

Nel modello proposto in questa tesi, la similarità semantica tra i messaggi viene calcolata, utilizzando l'ontologia linguistica WordNet 3.0, attraverso la formula di similarità di Lin (1998) [2], implementata dalle API WS4J (WordNet Similarity for Java). La similarità di Lin è compresa tra 0 e 1 e può essere espressa in maniera formale come segue:

$$lin(x,y) = \frac{2 \cdot \sum_{t \in syn(x) \cap syn(y)} logP(t)}{\sum_{t \in syn(x)} logP(t) + \sum_{t \in syn(y)} logP(t)} \tag{2.2}$$

dove:

(i) x ed y sono due parole;

(ii) syn è il synset;

(iii) $P(t)$ è la probabilità del termine t.

Il risultato della formula di similarità Lin è stato poi pesato con i TFIDF (term frequency - inverse document frequency) delle stesse per far si che i termini più frequenti, e quindi semanticamente meno discriminanti, abbiano un'incidenza minore sul valore finale. Quindi, dati due messaggi, la formula per il calcolo della similarità semantica è la seguente:

$$\sum_{i=0}^{LenX} \sum_{j=0}^{LenY}{}_{if POS(X_i)=POS(Y_j)} \frac{lin(X_i, Y_j) \cdot \frac{TFIDF(X_i)+TFIDF(Y_j)}{2}}{\sum_{t \in X \cup Y} TFIDF(t)} \tag{2.3}$$

(i) X ed Y sono i due messaggi;

[3] Vector Space Model

(ii) $LenX$ e $LenY$ sono il numero di verbi e nomi contenuti rispettivamente nei messaggi X ed Y;

(iii) POS è il part-of-speech tag;

(iv) X_i ed Y_j si riferiscono rispettivamente agli i-esimi e j-esimi nomi o verbi lemmatizzati contenuti nei messaggi X ed Y;

(v) lin è la similarità di Lin definita dalla formula (2.2);

(vi) $TFIDF$ è il term frequency - inverse document frequency.

Capitolo 3

Modellazione dei Grafi

Le comunità analizzate prevedono relazioni tra gli utenti che sono supportate dagli strumenti tecnologici in uso, piuttosto che essere identificate a priori per via delle relazioni sociali che intercorrono tra questi, come nel caso di "follow" o "amicizia".

La struttura dati che si presta meglio alla rappresentazione di un dominio applicativo di questo tipo è sicuramente un grafo in cui i nodi rappresentano gli utenti e gli archi corrispondono alle relazioni che intercorrono tra di loro.

La fase di costruzione del grafo come astrazione della comunità da analizzare nel prossimo step prevede la rappresentazione di ben 5 diverse tipologie di relazioni possibili che possono intercorrere tra gli utenti. Le prime 3 sono basate su delle banali relazioni sociali immediatamente disponibili, in quanto deducibili dalla struttura multi-threading implementata dalla maggior parte delle tecnologie che supportano le comunità sociali. Le altre 2, invece, sono basate sul contenuto dei messaggi che gli utenti hanno scambiato all'interno di uno stesso forum di discussione, ossia sulle similarità lessicali e semantiche calcolate come descritto nel capitolo precedente.

Il risultato sarà quindi un grafo orientato, etichettato e pesato con gli score di similarità per gli archi che rappresentano tale tipologia di relazione.

COMMENT_TO È una delle relazioni più semplici che può intercorrere tra due nodi del grafo: un utente b commenta il post di un utente a.

REPLY_TO Prevede la replica di un utente c al commento dell'utente b sul post dell'utente a.

MENTION_TO Comporta la creazione di un link tra un utente s ed un utente m oggetto del menzionamento, anche se coinvolti in forum o thread di discussione diversi.

LEXICALLY_SIMILAR_TO È dovuta alla similarità lessicale tra post di due utenti coinvolti nella stessa discussione, la cui direzionalità dell'arco è definita in base al timestamp.

SEMANTICALLY_SIMILAR_TO È dovuta alla similarità semantica tra post di due utenti coinvolti nella stessa discussione, la cui direzionalità dell'arco è decisa in base al timestamp.

Parte III

Analisi di Grafi Costruiti da Social Media

Capitolo 4

Analisi di Dati Temporali

La neccessità di trarre conclusioni incrementali per quanto riguarda i dati presenti in uno stream spesso non è soddisfacibile. Sebbene l'idea chiave risulti ancora oggi abbastanza vincente, tuttavia, la ricorsività non è caratteristica comune a tutti i problemi che si vorrebbero affrontare sui dati provenienti da uno stream: molti problemi hanno bisogno di processare necessariamente un insieme di dati per volta, con le dovute complicazioni in termini di consumi e performance.

La letteratura ha fornito diversi esempi di modelli per l'analisi progressiva dei data stream, probabilmente quello più vincente è sicuramente il modello basato sulle finestre temporali (time windows).

Una finestra temporale è, intuitivamente, una partizione dell'asse temporale, di ampiezza fissa o variabile, a partire da un determinato punto temporale.

Definizione 1. Si definisce finestra temporale $W(t, \Delta t)$, che comincia all'istante t con un'ampiezza Δt, come l'intervallo di tempo intercorrente fra l'istante t e l'istante $t + \Delta t$.

La finestra temporale, appena definita in accezione puramente temporale, solitamente sottende una partizione sui dati in ingresso che sono time-stamped. Se, ad esempio, avessimo un data stream caratterizzato alla maniera seguente:

$$DS = \{(d_1, t_1), (d_2, t_2), (d_3, t_3), (d_4, t_4), (d_5, t_5)\} \tag{4.1}$$

ossia un data stream che ha ricevuto soltanto 5 dati (d_i) in altrettanti istanti temporali (t_i) differenti, allora la finestra temporale $W(2,2)$ risulta essere come segue:

$$W(2,2) = \{(d_2, t_2), (d_3, t_3)\} \tag{4.2}$$

È facile capire come $W(t, \Delta t)$ induca sempre l'individuazione di un sottinsieme di DS; occorre pertanto dare una definizione operativa di finestra temporale in funzione di un data stream associato.

Definizione 2. Si definisce finestra temporale associata al data sream DS, a partire dall'istante t con un'ampiezza Δt, come il sottinsieme dei dati presenti nel data stream il cui istante di arrivo è compreso nell'intervallo $[t, \Delta t]$

$$W(DS, t, \Delta t) = \{(d_i, t_i) \in DS \mid t_i \in [t, \Delta t]\} \tag{4.3}$$

In quest'ottica operativa vale sempre la proprietà secondo cui
$W(DS, t, \Delta t) \subseteq DS$.

Questa definizione generale di finestra temporale, in realtà, nasconde tutta una serie di modelli di finestre temporali derivate a partire dal concetto base.

Esistono modelli che consentono ad alcune delle variabili indipendenti $(DS, t, \Delta t)$ di variare nel tempo: una finestra temporale infatti, a seconda del modello considerato, può traslare in avanti nel tempo, variare la propria ampiezza o addirittura fare entrambe le cose.

4.1 Landmark Windows

Il modello temporale di tipo landmark afferisce ad una particolare tipologia di finestre temporali $W(DS, t, \Delta t)$ in cui il parametro di ampiezza Δt può aumentare nel tempo. Questo tipo di finestra temporale ha la particolare inefficienza di crescere di dimensione nel tempo ($\Omega(\Delta t)$ dati dove Δt è l'ampiezza iniziale), se non limitata, rendendo potenzialmente inefficiente (in base alla quantità di dati esibiti) qualsiasi algoritmo che non scala in maniera adeguata.

Figura 4.1. Una landmark window che cresce fino all'instante temporale $i - 1$.

Le finestre temporali di tipo landmark, tuttavia, hanno provato la loro efficacia nel risolvere quei problemi incrementali, specie in quelli dove c'è un uso abbondante di "blocking operations" oltre ad un'assenza di ricorsività.

Ai fini della tesi in esame, questa finestra temporale crescente nel tempo riveste un interesse particolare in quanto ci permetterà di adempiere agli obbiettivi prefissati, ossia quello di investigare lo studio dei cambiamenti della comunità per caratterizzare l'evoluzione attraverso la scopera di interazioni tra i partecipanti.

Capitolo 5

Analisi di Dati in Forma di Grafo

Nell'analisi dei social networks, i concetti di teoria dei grafi vengono utilizzati per capire e spiegare i fenomeni sociali.

Gli algoritmi sui grafi vengono utilizzati per calcolare metriche riguardo nodi e relazioni. Possono fornire informazioni su entità rilevanti nel grafo, esprimibili in termini di indicatori di centralità, o strutture intrinseche come comunità per mezzo di metodi di partizionamento o clustering. Molti di questi approcci attraversano frequentemente il grafo per il calcolo di tali metriche, eseguendo ricerche in ampiezza o in profondità, per cui, a causa della crescita esponenziale dei possibili cammini, molti approcci hanno anche un'elevata complessità algoritmica. Per questo motivo, al fine di eseguire un'analisi efficiente dei grafi temporali cumulativi, sono state utilizzate le API di neo4j, ottimizzate in quanto utilizzano determinate strutture del grafo, memorizzano parti già esplorate e parallelizzano le operazioni.

5.1 Ruolo dell'Utente: Indicatori di Centralità

Uno strumento essenziale ed efficace per l'analisi dei social networks è costituito dagli indicatori di centralità definiti sui nodi (Bavelas, 1948 [3]; Sabidussi, 1966 [4]; Freeman, 1979 [5]) e progettati per classificarli in base alla loro posizione nel grafo.

5.1.1 Centralità "Degree"

La centralità "degree" di un nodo è data dal numero delle relazioni in cui esso è coinvolto ed è una misura di centralità semplice ma efficace che fornisce un grado di importanza nodale. Più alto sarà il suo valore, più esso ricoprirà un ruolo importante nel grafo.

In un grafo orientato come quello sociale in questione, per il calcolo di tale indicatore si fa una distinzione in base al fatto che un nodo sia coinvolto in una

relazione in entrata o in uscita. Tali valori prendono rispettivamente il nome di **in-degree** ed **out-degree** e sono utili per mettere in evidenza il ruolo di alcuni nodi nel grafo come attrattori o mittenti. I nodi hub hanno un degree elevato, al più ugule alla somma di tutti i degree degli altri nodi del grafo, mentre i nodi spoke hanno un degree almeno pari ad 1.

Definizione 3. Sia $G = (V, A)$ un grafo orientato dove V è l'insieme dei vertici ed A l'insieme degli archi, le centralità "in-degree" ed "out-degree" possono essere definite rispettivamente come segue:

$$C_D{}^+(v) = \sum_{\substack{a \in A \\ s \in V}} a_{sv} \tag{5.1}$$

$$C_D{}^-(v) = \sum_{\substack{a \in A \\ s \in V}} a_{vs} \tag{5.2}$$

dove a_{sv} è l'arco che va dal nodo s al nodo v e viceversa.

5.1.2 Centralità "Betweenness"

La centralità "betweenness" (Anthonisse, 1971 [6]; Freeman, 1977 [7]) è una misura di centralità basata sul calcolo dei cammini minimi ed è utile per trovare i nodi che fungono da ponte da una parte all'altra del grafo.

Per ogni coppia di nodi in un grafo connesso, esiste almeno un percorso più breve che può essere basato sul numero di relazioni che il percorso attraversa, se il grafo non è pesato, o sulla somma dei pesi delle relazioni, in caso contrario. Tale indicatore viene calcolato sommando, per ciascun nodo, il numero di percorsi più brevi che lo attraversano. I nodi che si trovano più frequentemente su questi percorsi avranno un punteggio di centralità più elevato.

Definizione 4. Sia $G = (V, A)$ un grafo orientato dove V è l'insieme dei vertici ed A l'insieme degli archi, la centralità "betweenness" di un vertice v è definita come segue:

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{5.3}$$

dove:

(i) $\sigma_{st}(v)$ è il numero totale di percorsi minimi dal nodo s al nodo t che passano per il nodo v;

(ii) σ_{st} è il numero totale di percorsi minimi dal nodo s al nodo t.

In caso di grafi sociali di grandi dimensioni, il calcolo della misura di centralità "betweenness" di un nodo porebbe risultare computazionalmente costoso. Per un grafo non pesato, la prima versione dell'algoritmo che implementava il calcolo di tale indicatore di centralità aveva una complessità temporale pari a $\Theta(|V|^3)$ ed una complessità spaziale pari a $\Theta(|V|^2)$. Per lo studio in questione è stato utilizzato un algoritmo più efficiente messo a punto negli anni a venire [8] (Brandes, 2001) con complessità spaziale pari a $\mathcal{O}(|V| + |A|)$ ed una complessità temporale pari a $\mathcal{O}(|V||A|)$.

5.1.3 Page Rank

PageRank (Larry Page, Sergey Brin; 1996) è il noto algoritmo di Google utilizzato dall'omonimo motore di ricerca per stabilire il ranking delle pagine web tra i risultati della ricerca. L'algoritmo conta il numero e la qualità delle relazioni di un nodo per determinare una stima dell'importanza del nodo stesso all'interno del grafo, nel caso di Google la qualità delle pagine web in relazione ai link tra di esse. L'ipotesi di fondo è data dalla probabilità che pagine web importanti vengano linkate da altri siti web.

L'algoritmo per il calcolo del PageRank può essere formalizzato come segue:

$$PR[n] = \frac{(1-d)}{N} + d \cdot (\sum_{k=1}^{n} \frac{PR[n_k]}{C[n_k]}) \tag{5.4}$$

dove:

(i) $PR[n]$ è il valore di PageRank del nodo n che si vuole calcolare;

(ii) d (damping factor) è un fattore dal quale dipende la percentuale di PageRank che deve transitare da un nodo all'altro (per Google 0,85);

(iii) N è il numero totale di nodi del grafo;

(iv) n è il numero di nodi dai quali parte almeno una relazione verso u, p_k rappresenta ognuno di tali nodi;

(v) $PR[n_k]$ sono i valori di PageRank di ogni nodo n_k;

(vi) $C[n_k]$ è il numero complessivo di relazioni in cui è coinvolto il nodo n_k.

5.2 Scoperta di Interazioni Basate su Densità

Nello studio di reti complesse, si dice che una rete ha una struttura di comunità se i suoi nodi possono essere raggruppati in modo tale che quelli di ogni insieme siano densamente connessi internamente ed abbiano connessioni più sparse tra i gruppi.

Trovare comunità all'interno di una rete può essere un compito computazionalmente difficile. Il numero di comunità, se ve ne sono, all'interno della rete è tipicamente sconosciuto e le comunità sono spesso di dimensioni e/o densità non uguali. Nonostante queste difficoltà, sono stati sviluppati e impiegati diversi metodi per la ricerca della comunità con diversi livelli di successo.

5.2.1 Metodo di Louvain

Uno dei metodi più utilizzati per la scoperta di comunità all'interno di una rete è la massimizzazione della modularità di una partizione. La modularità è una funzione euristica che misura la qualità di una particolare divisione di una rete in comunità.

Il metodo di massimizzazione della modularità rileva le comunità ricercando le possibili divisioni di una rete per una o più che hanno una modularità particolarmente elevata. Poichè la ricerca esaustiva su tutte le possibili divisioni è solitamente intrattabile, gli algoritmi pratici si basano su metodi di ottimizzazione approssimativi come algoritmi greedy, con approcci diversi che offrono diversi equilibri tra velocità e accuratezza. La modularità di una partizione è un valore compreso tra -1 e 1 che misura la densità dei collegamenti all'interno delle comunità rispetto ai collegamenti tra le comunità. Nel caso di reti ponderate, la modularità è definita come:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \tag{5.5}$$

dove:

(i) A_{ij} rappresenta l'arco pesato tra il nodo i ed il nodo j;

(ii) k_i e k_j sono le somme dei pesi degli archi in cui sono coinvolti rispettivamente i nodi i e j;

(iii) $2m$ è la somma di tutti i pesi degli archi del grafo;

(iv) c_i e c_j sono le rispettive communità dei nodi i e j;

(v) δ è una funzione delta che restituisce 1 se $c_i = c_j$, 0 altrimenti.

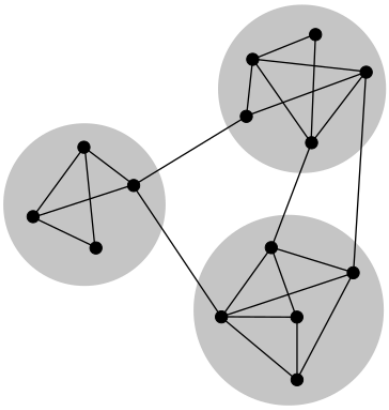

Figura 5.1. Esempio di una piccola rete nella quale è possibile visualizzare la struttura della comunità, con tre gruppi di nodi con connessioni interne dense e connessioni più sparse tra i gruppi.

Un metodo di massimizzazione della modularità è quello di Louvain [9] che consiste di due fasi, ripetute in modo iterativo per massimizzare questo valore.

Innanzitutto, ciascun nodo nella rete è assegnato alla propria comunità, favorendo le ottimizzazioni locali della modularità. Per ogni nodo i, viene calcolato il cambiamento di modularità per rimuovere i dalla propria comunità e spostarlo nella comunità di ciascun suo vicino j. Questo valore viene calcolato attraverso la seguente formula:

$$\Delta Q = \left[\frac{\sum_{in} + 2k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m} \right)^2 \right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m} \right)^2 - \left(\frac{k_i}{2m} \right)^2 \right] \tag{5.6}$$

dove:

(i) $\sum_{in}$ è la somma di tutti i pesi degli archi all'interno della comunità nella quale i si sta muovendo;

(ii) $\sum_{tot}$ è la somma di tutti i pesi degli archi che coinvolgono i nodi della comunità;

(iii) k_i è il grado di i;

(iv) è la somma dei pesi degli archi tra i e gli altri nodi nella comunità;

(v) è la somma dei pesi di tutti gli archi del grafo.

Una volta calcolato questo valore per tutte le comunità a cui il nodo i è collegato, esso viene inserito nella comunità che ha prodotto il maggiore aumento della modularità. Se nessun aumento è possibile, il nodo i resta nella sua comunità di partenza.

Questo processo viene applicato ripetutamente e in sequenza a tutti i nodi fino a quando non si verifica un aumento della modularità. Una volta raggiunto questo massimo locale di modularità, la prima fase à terminata.

Nella seconda fase dell'algoritmo, tutti i nodi nella stessa comunità vengono raggruppati e si crea una nuova rete in cui i nodi sono le comunità della fase precedente. Tutti i collegamenti tra i nodi della stessa comunità sono ora rappresentati da loop automatici sul nuovo nodo della comunità e i collegamenti da più nodi nella stessa comunità a un nodo in una comunità diversa sono rappresentati da bordi ponderati tra le comunità. Una volta creata la nuova rete, la seconda fase è terminata e la prima fase può essere riapplicata sulla nuova rete.

5.3 Scoperta di Interazioni Basate su Frequenza

Uno dei task tipici del data mining è quello di indagare la presenza di specifici aspetti frequenti in un archivio di dati. Questa necessità ha le sue radici, come molti altri problemi, nella questione di dover profilare i comportamenti frequenti assunti da un agente a cui si è interessati.

5.3.1 Scoperta di Itemset Frequenti

Il frequent itemset mining (o frequent pattern mining) è una delle aree di ricerca nell'ambito del data mining che si occupa nello specifico di offrire algoritmi e strutture dati che agevolino questo tipo di analisi.

Come il nome stesso suggerisce, il frequent pattern mining si occupa pertanto di indagare gli insiemi di oggetti che compaiono frequentemente insieme in una, più o meno grande, quantità di dati: questo insieme di oggetti prende il nome di itemsets o di pattern frequenti.

Definizione 5. Dato un insieme $I = \{i_1, ..., i_n\}$ di oggetti, si definisce k-itemset un qualsiasi sottinsieme $I_K \subseteq I$ di lunghezza $K(|I_K| = K)$.

Generalmente c'è interesse soltanto per gli itemsets che esprimono caratteristiche particolari, nel caso del frequent itemset mining c'è interesse solo verso quelli che figurano un elevato numero di volte all'interno di un insieme di transazioni.

Definizione 6. Si definisce transazione i-esima T_i l'insieme $T_i \subseteq I$. Ciascuna transazione possiede un identificativo univoco i.

Definizione 7. Dato un itemset I_k, si dice che è contenuto in una transazione T_i se e sole se $I_k \subseteq T_i$.

Definizione 8. Si definisce database di transazioni l'insieme D delle transazioni T_i.

Intuitivamente è facile comprendere la notazione di frequenza, operativamente però si necessita di una misura di confidenza in grado di suggerire quantitativamente se l'itemset considerato presenti effettivamente la caratteristica di "essere frequente".

È possibile ricorrere a delle semplici statistiche per definire una che quantifichi la frequenza di un itemset in funzione di un livello di confidenza.

In maniera abbastanza elementare è chiaro come, per un particolare itemset I_k ci sia interesse nell'enumerare le transazioni T_i che contengono I_k; nel gergo scientifico si usa dire che un itemset (o un pattern) "copre" un determinato numero di transizioni.

Definizione 9. Si definisce copertura di un itemset I_k come l'insieme delle transazioni che lo includono:

$$coverage(I_k) = \{T_i \in D \mid I_k \subseteq T_i\} \tag{5.7}$$

Chiaramente $coverage(I_k) \subseteq D$.

Altrettanto sistematicamente, a partire dalla copertura, è possibile misurare il numero di transazioni che "supportano" l'itemset.

Definizione 10. Si definisce supporto assoluto di un itemset I_k come la cardinalità della sua copertura:

$$support_{abs}(I_k) = |coverage(I_k)| \tag{5.8}$$

Chiaramente il supporto è una misura discreta: $support_{abs}(I_k) \in \aleph$.

Spesso al posto del supporto assoluto si usa il supporto relativo, una quantità analoga che quantifica l'espressione della frequenza statistica di un itemset.

Definizione 11. Si definisce supporto relativo di un itemset I_k come il rapporto del supporto assoluto ed il numero totale di transazioni nel database D:

$$support_{rel}(I_k) = \frac{support_{abs}(I_k)}{|D|} = \frac{|coverage(I_k)|}{|D|} \tag{5.9}$$

Il supporto relativo è sempre compreso fra 0 e 1.

Il supporto relativo è, a tutti gli effetti, il parametro che stabilisce la frequenza di un itemset ma, naturalmente, non stabilisce in maniera inequivocabile se sia frequente o infrequente.

La capacità di attirbuire un valore di frequenza (o di infrequenza) ad un itemset è appannaggio esclusivo dell'operatore del task di data mining: uno stesso itemset I_k per un utente è da considerarsi frequente se copre il 20% delle transazioni quando per un secondo utente potrebbe non essere così, ad esempio nel caso in cui debba coprire l'80% delle transazioni.

Definizione 12. Un pattern I_k si dice frequente se $support_{rel}(I_K) > \alpha$ ove α è un valore di soglia minima stabilito dall'utente del task di mining.

Avendo dato la definizione di supporto di un itemset occorre fare alcune particolari considerazioni: un database delle transazioni solitamente contiene più e più itemset differenti, che avranno un supporto altrettanto differente. Compito del frequent itemset mining è quello di scoprire tutti gli itemsets frequenti all'interno del database delle transazioni.

Osservando il database delle transazioni può darci informazioni sull'insieme I degli oggetti ivi contenuti ma non sui k-itemsets di qualsiasi lunghezza.

Estrapolare questo tipo di informazione richiede un approccio di logica induttiva capace di trarre conclusioni su fatti specifici partendo da fatti più generali.

Una strategia di ragionamento automatico del genere presuppone dei principi ben saldi di funzionamento che diano luogo ad uno spazio di ricerca facilmente esplorabile. Consideriamo l'esempio seguente:

Dato un database D con 10 transazioni e dato l'insieme $I = \{a, b, c, d, e\}$, per poter valutare tutti gli itemsets frequenti, in questo caso fino alla lunghezza massima di 5, dovremmo indagare in un insieme di 2^5 combinazioni possibili.

Dato $= \{T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}\}$

$T_1 = \{a, d, e\}$, $T_2 = \{b, c, d\}$, $T_3 = \{a, c, e\}$, $T_4 = \{a, c, d, e\}$, $T_5 = \{a, e\}$

$T_6 = \{a, c, d\}$, $T_7 = \{b, c\}$, $T_8 = \{a, c, d, e\}$, $T_9 = \{b, c, e\}$, $T_{10} = \{a, d, e\}$

Itemsets frequenti ($\alpha = 30\%$):

0-itemsets	1-itemsets	2-itemsets	3-itemsets
$\emptyset$: **10**	$\{a\}$: 7	$\{a, c\}$: 4	$\{a, c, d\}$: 3
	$\{b\}$: 3	$\{a, d\}$: 5	$\{a, c, e\}$: 3
	$\{c\}$: 7	$\{a, e\}$: 6	$\{a, d, e\}$: 4
	$\{d\}$: 6	$\{b, c\}$: 3	
	$\{e\}$: 7	$\{c, d\}$: 4	
		$\{c, e\}$: 4	
		$\{d, e\}$: 4	

Figura 5.2. Esempio di costruzione di k-itemsets a partire da un database di transazioni D.

Lo stesso esempio suggerisce che, con la soglia del supporto indicata, il numero degli itemsets frequenti è pari a 16 con una lunghezza al più pari a 3: questo dovrebbe suggerire qualcosa.

Si è detto come il processo enumerativo di scoperta tenda a trarre conclusioni, per situazioni via via più specifiche a partire da situazioni più generali: questo è possibile mediante l'introduzione di una speciale relazione d'ordine sull'insieme degli itemsets.

Definizione 13. Dati due itemsets $I1$ ed $I2$, si dice che $I1$ θ-sussume $I2$ se e solo se esiste una sostituzione θ tale che $I2\ \theta \subseteq I1$.

Definizione 14. Dati due itemsets $I1$ ed $I2$, allora $I1$ è più generale di $I2$ in sussunzione se e sole se $I2$ θ-sussume $I1$; si scrive che $I1 \geq_\theta I2$.

Tale definizione contribuisce a concretizzare lo spazio di ricerca, parzialmente ordinato per generalità, dei 2^I pattern: il reticolo, o insieme parzialmente ordinato, $(2^I, \geq_\theta)$.

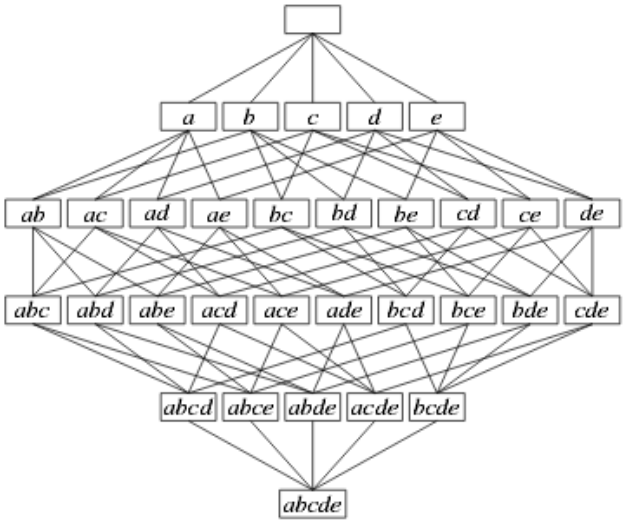

Figura 5.3. Diagramma di Hasse del reticolo $(2^I, \geq_\theta)$ di tutti i possibili itemsets generabili nell'esercizio precedente.

Anche così, se si desiderasse trovare tutti e soli i frequenti bisognerebbe verificare esattamente tutti i possibili candidati: cercare nello spazio esponenziale di tutte le possibili combinazioni rende il problema velocemente intrattabile, è possibile ottimizzare la ricerca nel reticolo grazie ad una serie di proprietà:

(i) proprietà di *antimonotonicità* del supporto: ad itemset più specifici, ossia quelli più in basso del reticolo, corrispondono supporti via via inferiori;

$$I_K \subset I_{K+h} \Rightarrow support_{rel}(I_K) > support_{rel}(I_{K+h}) \tag{5.10}$$

(ii) proprietà *apriori*: nessuna specializzazione (superinsieme) di un itemset infrequente può essere frequente e, analogamente, tutti gli itemset più generali (sottoinsiemi) di un itemset frequente sono anch'essi frequenti.

$$support_{rel}(I_K) < \alpha \Rightarrow support_{rel}(I_{K+h}) < \alpha \tag{5.11}$$

Sfruttando tali proprietà è possibile costruire un approccio altamente enumerativo per la scoperta dei pattern frequenti. Tale tecninca consente di valutare solo gli itemsets direttamente frequenti, il tutto a partire dai più generali fino a terminare la ricerca con i più specifici: ricerca levelwise di tipo top-down.

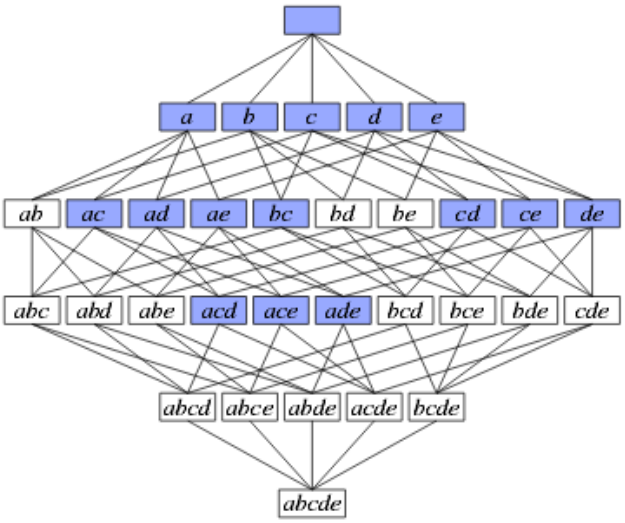

Figura 5.4. Diagramma di Hasse del reticolo $(2^I, \geq_\theta)$ di tutti i possibili itemsets generabili nell'esercizio precedente. Gli itemsets blu sono i soli frequenti: si noti come tengano le proprietà apriori e di antimonotonicità.

In letteratura esistono diversi algoritmi di frequent itemset mining, fra i più celebri vi sono sicuramente apriori [10] ed ECLAT [11] che verranno citatati di seguito in quanto fondamentale per questo lavoro di tesi.

5.3.2 Integrazione di Set Enumeration Tree

In questa sezione si rivolgeranno delle brevi considerazioni ad una delle strutture dati utilizzate in letteratura per rappresentare il reticolo degli itemset $(2^I, \geq_\theta)$. Guardando alla figura 5.3.1 del paragrafo precedente si nota come, effettivamente, una struttura dati puramente reticolare risulterebbe difficile da mantenere in memoria e da gestire; l'insidia preponderante in questo tipo di questione è rappresentata

dall'elevato numero di puntatori (archi) fra gli elementi ivi presenti. L'ideale sarebbe rocondurre una struttura reticolare ad una struttura dati basata sul concetto di albero.

Molti problemi informatici ammettono soluzioni i cui elementi appartengono ad un determinato insieme potenza. Nel caso del frequent itemset mining tale insieme potenza è proprio lo spazio 2^I dei possibili itemsets. L'idea chiave è quella di introdurre una struttura dati capace di operazioni di ricerca non ridondanti, ad esempio basata sulla riduzione degli archi.

Nei problemi dove lo spazio di ricerca è un sottoinsieme dell'insieme potenza (nel caso del frequent itemset mining lo è l'insieme dei pattern frequenti ad esempio) chiuso per inclusione insiemistica è possibile utilizzare un set enumeration tree, meglio conosciuto come SE-Tree [12].

Considerando come insieme base degli oggetti su cui enumerare gli itemset, l'insieme I, si introduca al concetto di indice e di vista di un nodo.

Definizione 15. Si definisce indice posizionale per gli oggetti $i \in I$ una funzione bigettiva del tipo $view : I \mapsto \aleph$.

Definizione 16. Dato l'insieme $S \subseteq I$ si definisce la vista di S come l'insieme:

$$view(S) = \{i \in I \mid index(i) > max_{j \in S} ind(j)\} \tag{5.12}$$

La vista di un sottinsieme S fornisce precise indicazioni di quali oggetti è possibile usare per poter espanderlo: nel reticolo, aumentando la lunghezza degli itemset, non si faceva altro che espandere un k-itemset, crearne cioè un superinsieme, sulla base di alcuni suoi 1-itemsets.

Si definisce un SE-Tree base, l'albero definito alla maniera seguente:

Definizione 17. Sia F una collezione di insiemi su cui vale la chiusura per inclusione insiemistica, ossia $\forall\ S \in F$ se $S' \subseteq S$ allora $S' \in F$, allora T è un SE-Tree per F se e solo se:

(i) la radice di T è etichettata con l'insieme vuoto $\emptyset$;

(ii) i figli di un nodo etichettato con S in T sono dati dall'insieme $\{S \cup \{e\} \in F \mid e \in view(S)\}$.

La definizione dice che un set enumeration tree per un insieme potenza (o insieme delle parti) è radicato nell'insieme vuoto e che i figli di ogni nodo sono risultati dall'unione di quel nodo con gli elementi di indice maggiore.

Il set enumeration tree, possiede anche il carattere di albero dei prefissi: si noti come i nodi in relazione di discendenza condividano un prefisso via via crescente. Questo tipo di albero è chiaramente sbilanciato ma permette di valutare in maniera rapida la presenza di itemsets anche piuttosto grandi.

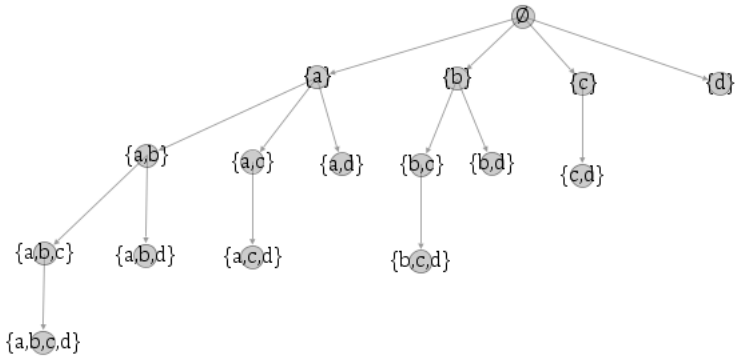

Figura 5.5. SE-Tree dell'insieme $\{a, b, c, d\}$, questo SE-Tree potrebbe essere quello del reticolo degli itemsets costruibili a partire da $I = \{a, b, c, d\}$ secondo il reticolo $(2^I, \geq_\theta)$.

5.3.3 Algoritmo ECLAT

Nel frequent itemsets mining in generale molti algoritmi sfruttano a proprio vantaggio l'antimonotonicità del supporto e la proprietà a priori; il tutto congiuntamente a strutture dati come il SE-Tree.

Ai fini della tesi si studia molto brevemente un algoritmo di computazione degli itemset frequenti che getta le proprie basi su una versione insiemistica della regola a priori.

L'algoritmo ECLAT [11] è un algoritmo efficiente di tipo top-down per la valutazione automatica degli itemset frequenti in un SE-Tree: significa che combina in qualche modo situazioni generali per trarre conclusioni su situazioni più specifiche.

ECLAT viene usato congiuntamente ad un SE-Tree per dedurre i pattern dal supporto elevato facendo un uso considerevole di intersezioni insiemistiche.

L'algoritmo funziona grazie ad un principio molto elementare, è certamente utile ad interessante considerare l'idea chiave alla base, anche chiamata "proprietà ECLAT (Equivalent CLAss of Transactions)".

Definizione 18. Dati due itemset $I1$ e $I2$ di uguale lunghezza k, allora vale la seguente proprietà:

$$coverage(\langle I1, I2\rangle) = coverage(I1) \cap coverage(I2) \tag{5.13}$$

La proprietà di ECLAT è in accordo con la regola apriori, infatti si dimostra abbastanza banalmente che se $I1$ e $I2$ sono itemsets di lunghezza k, allora $I12 = \langle I1, I2\rangle$ è un itemset di lunghezza $k + 1$, chiaramente risulta che:

$$\begin{aligned} coverage(I12) = coverage(I1) \cap coverage(I2) \Rightarrow \\ \Rightarrow support_{abs}(I12) = |coverage(I1) \cap coverage(I2)| \end{aligned} \tag{5.14}$$

poichè:

$$|coverage(I1) \cap coverage(I2)| \leq min(support_{abs}(I1), support_{abs}(I2)) \tag{5.15}$$

allora vale:

$$support_{abs}(I12) \leq min\big(support_{abs}(I1), support_{abs}(I2)\big) \tag{5.16}$$

In totale accordo con la regola di apriori: all'aumentare della lunghezza di un itemset il proprio supporto tende a decrescere.

Nel gergo dell'algoritmo l'insieme della copertura di un itemset prende il nome di tidlist (list of transaction ids) ma il funzionamento è veramente analogo.

Sebbene rsultati sperimentali preferiscano ECLAT ad altri algoritmi di frequent itemset mining, le sue maggiori inefficienze sono quelle di richiedere numerose operazioni di intersezioni insiemistiche, inefficienti spesso sia in spazio che in tempo, oltre a richiedere di memorizzare una tidlist per ogni itemsets [1].

[1]Per grandi database di transazioni mantenere le tidlist (o le coperture) di ogni itemset può richiedere uno spreco enorme in memoria: l'uso di ECLAT è stato accuratamente valutato nel tempo anche in presenza di framework dinamici in grado di cambiare il tipo di rappresentazione delle tidlist per ragioni puramente di efficienza. Come se non bastasse l'aumento della dimensione delle tidlist favorisce una rapida degradazione delle performance al momento dell'intersezione.

Parte IV

Esperimenti

Capitolo 6

Reddit

Reddit è un aggregatore di social news nonché sito internet di discussione e valutazione di contenuti web. Gli utenti registrati, chiamati redditors, possono pubblicare contenuti sotto forma di link o messaggi di testo che possono essere soggetti a commenti e a voti favorevoli o contrari da parte di altri membri, il che determina posizione e visibilità dei vari contenuti sulle pagine del sito. Tali contenuti, chiamati submissions, vengono organizzati per argomenti in aree di interesse chiamate subreddits che coprono una varietà di argomenti tra cui istruzione, intrattenimento, humor, tecnologia, ecc..

Figura 6.1. Logo di Reddit.

6.0.1 Datasets

I datasets mensili di Reddit, resi disonibili online in formato JSON dal redditor u/Stuck_In_the_Matrix e suddivisi in submissions (10 mln ca.) e commenti (80 mln ca.), sono stati importati all'interno di MongoDB, un DBMS[1] NoSQL orientato ai documenti, che ha permesso, per mezzo di indici di ricerca, una gestione più efficiente dei dati nei passi che verranno descritti in seguito.

[1]Database Management System

Figura 6.2. Logo di MongoDB.

6.1 Filtraggio

Per poter ridurre la dimensione dei dati in input ed allo stesso tempo dare un senso allo studio di cui parleremo in seguito, è stato effettuato un filtraggio per mezzo di alcuni criteri descritti nei paragrafi seguenti. A partire da quelle che sono le entità del dominio applicativo in questione ci assicuriamo quindi di lavorare su un dataset formato da utenti particolarmente attivi e da submissions e commenti semanticamente significativi.

6.1.1 Criteri

Submissions Vengono considerati soltanto i submissions per cui:

(i) il numero di commenti è maggiore rispetto alla media, ossia:

$$\#\text{ commenti} > \frac{\#\text{ commenti}}{\#\text{ submissions}} \tag{6.1}$$

(ii) la lunghezza del selftext è maggiore rispetto alla media, ossia:

$$\text{length(selftext)} > \frac{\sum_{s=1}^{\#\text{ submissions}} \text{length(selftext(s))}}{\#\text{ submissions}} \tag{6.2}$$

Commenti Vengono considerati soltanto i commenti per cui la lunghezza del body è maggiore rispetto alla media, ossia:

$$\text{length(body)} > \frac{\sum_{c=1}^{\#\text{ commenti}} \text{length(body(c))}}{\#\text{ commenti}} \tag{6.3}$$

Redditors Vengono considerati soltanto i submissions ed i commenti scritti dai redditors in numero maggiore rispetto alla media, ossia:

$$\#\text{ submissions} + \#\text{ commenti} > \frac{\#\text{ submissions} + \#\text{ commenti}}{\#\text{ distinti utenti}} \tag{6.4}$$

Subreddits Vengono considerati soltanto i subreddits per cui il numero di submissions appartenenti, in seguito al filtraggio definito dai criteri (6.1) e (6.2), risulti maggiore rispetto alla media, ossia:

$$\#\text{ submissions filtrati} > \frac{\#\text{ submissions}}{\#\text{ distinti subreddit}} \tag{6.5}$$

In riferimento al dataset Reddit di Novembre 2017, ad esempio, i criteri appena definiti hanno prodotto un filtraggio dei dati come segue. Vengono considerati:

(i) i *submissions* per cui il numero di commenti è maggiore di 8 e per cui la lunghezza del selftext è maggiore di 168 caratteri;

(ii) i *commenti* per cui la lunghezza del body è maggiore di 170 caratteri;

(iii) i submissions ed i commenti che i *redditors* autori hanno scritto in numero maggiore di 20;

(iv) i *subreddits* per cui il numero di submissions che vi appartengono, in seguito al filtraggio, è maggiore di 77.

I dati sono stati modellati optando per una rappresentazione ad albero dei subreddits che rispecchiasse la struttura del dominio applicativo dovuta alla possibilità di interazioni multi-threading tra i redditors sotto un determinato submission. In dettaglio: un documento della nuova collezione rappresenta un subreddit con relativo id e nome. Ogni subreddit contiene una lista di submissions che, a loro volta, possono contenere una lista di commenti. Ogni submission e/o commento contiene un id, l'autore, la data e l'ora della pubblicazione ed il risultato della fase di NLP sul corpo testuale. Ogni submission contiene una lista di similarità lessicali e semantiche calcolate come definito nelle sezioni precedenti ed ogni commento può a sua volta contenere un'altra lista di commenti con gli stessi campi e così via per oguno di essi.

```
{
  "id":"t5_2s4f7",
  "name":"granturismo",
  "submissions": [
    {
      "id":"7a3w3p",
      "redditor":"noatakzak",
      "UTC":"Nov 1, 2017, 3:25:57 PM",
      "mentions":[ ],
      "selftextPOSTags":[{"word":"Forgot","lemma":"forget","tag":"VBD"},{
         "word":"issues","lemma":"issue","tag":"NNS"}, ecc.],
      "comments": [
        {
          "id":"dp6v33n",
          "redditor":"substrate80",
          "UTC":"Nov 1, 2017, 3:37:43 PM",
          "mentions":[ ],
          "bodyPOSTags":[{"word":"had","lemma":"have","tag":"VBD"},{"word
             ":"messed","lemma":"mess","tag":"VBN"}, ecc.],
          "replies": [
            {
              "id":"7a3w3p",
              "redditor":"noatakzak",
              "UTC":"Nov 1, 2017, 3:42:50 PM",
              "mentions":["substrate80"],
              "bodyPOSTags":[{"word":"making","lemma":"make","tag":"VBG"}
                 ,{"word":"music","lemma":"music","tag":"NN"}, ecc.],
              "replies": [ ecc. ]
            }
          ]
        }
      ],
      "lexicalSimilarities":[{"id":"dp694m4","authorFrom":"bloodhawk713",
         "authorTo":"-Kaneki-","UTC":"Nov 1, 2017 3:31:50 AM","score":
         0.005865229270958179}, ecc.],
      "semanticSimilarities":[{"id":"dp694m4","authorFrom":"bloodhawk713
         ","authorTo":"-Kaneki-","UTC":"Nov 1, 2017 3:31:50 AM","score":
         0.03389350994627763}, ecc.]
    }
  ]
}
```


Listing 6.1. Esempio di JSON Subreddit

A partire da oggetti Java creati per mezzo di classi modellate opportunamente per poter rappresentare le entità del social media Reddit così come sopra descritte, è stato possibile, grazie alle API Gson di Google, convertire tali oggetti in documenti JSON che sono poi stati serializzati all'interno di una nuova collezione di MongoDB.

Bibliografia

[1] Fu, Tianjun, Ahmed Abbasi, and Hsinchun Chen. "A hybrid approach to web forum interactional coherence analysis." Journal of the Association for Information Science and Technology 59.8 (2008): 1195-1209.

[2] Lin, Dekang. "An information-theoretic definition of similarity." Icml. Vol. 98. No. 1998. 1998.

[3] Bavelas, Alex. "A mathematical model for group structures." Human organization 7.3 (1948): 16-30.

[4] Sabidussi, Gert. "The centrality index of a graph." Psychometrika 31.4 (1966): 581-603.

[5] Freeman, Linton C. "Centrality in social networks conceptual clarification." Social networks 1.3 (1978): 215-239.

[6] Anthonisse, Jac M. "The rush in a directed graph." Stichting Mathematisch Centrum. Mathematische Besliskunde BN 9/71 (1971).

[7] Freeman, Linton C. "A set of measures of centrality based on betweenness." Sociometry (1977): 35-41.

[8] Brandes, Ulrik. "A faster algorithm for betweenness centrality." Journal of mathematical sociology 25.2 (2001): 163-177.

[9] Blondel, Vincent D., et al. "Fast unfolding of communities in large networks." Journal of statistical mechanics: theory and experiment 2008.10 (2008): P10008.

[10] Agarwal, Rakesh, and Ramakrishnan Srikant. "Fast algorithms for mining association rules." Proc. of the 20th VLDB Conference. 1994.

[11] Zaki, Mohammed Javeed, et al. "New Algorithms for Fast Discovery of Association Rules." KDD. Vol. 97. 1997.

[12] Rymon, Ron. "Search through systematic set enumeration." (1992).