Università degli Studi di Bari - "Aldo Moro"

DI

Corso di Laurea Magistrale in Informatica e Tecnologie per la Produzione del Software

Tesi di laurea
in
Programmazione II

Url2Vec:
Clustering di pagine in un grafo Web

Relatori:
Chiar.mo Prof. Michelangelo Ceci
Chiar.mo Prof. Donato Malerba

Correlatore:
Dott.ssa Pasqua Fabiana Lanotte

Laureando:
Christopher Piemonte
Matricola 587662

Anno Accademico 2014-2015

Ai miei genitori ...

Indice

Elenco delle tabelle

Elenco delle figure

Introduzione

La principale caratteristica dell'era dell'informazione è rappresentata dalla possibilità di generare, memorizzare, trasmettere e processare enormi quantità di dati in modo rapido ed economico. La disponibilità di una simile quantità di dati, elaborabili automaticamente, ha consentito un forte incremento del processo di generazione e diffusione di conoscenza, utilizzabile per migliorare processi decisionali. Ad oggi, tuttavia, tali risorse non sono appieno sfruttate in tutti i campi e il loro valore potenziale riserva ancora numerose sorprese. La ricerca di pattern nella generazione e nell'utilizzo di nuovi contenuti web e la conoscenza che portano è un'area ancora giovane nell'informatica e in rapida crescita.
Con l'aumentare dei dati disponibili sul web e le potenzialmente infinite pagine generate dinamicamente, il bisogno di preprocessare questa informazione sembra scontrarsi con problemi computazionali. Indicizzare o cercare milioni di documenti non-omogenee sul web è diventata una sfida.
A dispetto della diversità delle pagine web nella rete, quelle che risiedono all'interno di una particolare organizzazione, spesso, condividono una certa struttura.
Il clustering di pagine web è un argomento trattato estensivamente in letteratura come un modo di raggruppare pagine all'interno di cluster omogenei, anche se gran parte del lavoro svolto si basa su un insieme arbitrario di pagine derivanti da molteplici siti differenti. Relativamente poco è stato il lavoro svolto sul clustering di un specifico sito di una determinata organizzazione.
In quest'ottica nasce Url2vec, che combinando tecniche di Data Mining e di Natural Language Processing, si propone come valida opzione per il clustering

di pagine web estraendo informazioni latenti nella struttura degli hyperlink, denotando una correlazione nascosta nei cammini percorribili nel grafo del web.
Le motivazioni alla base dell'implementazione di Url2vec sono state guidate dal voler sfruttare conoscenza già immagazzinata nella risoluzione di problemi specifici in contesti differenti per il quale erano stati ideati, ricavando un trasferimento della conoscenza.
L'obiettivo in particolare di questa tesi è estendere lo stato attuale esistente, implementando componenti per realizzare l'estrazione di pagine web correlate e raggrupparle sulla base delle sequenze attraversabili per vistarle, offrendo un diverso punto di vista considerando maggiormente le relazioni invece che il solo contenuto.

Nel capitolo 1 ci si occuperà di descrivere lo stato attuale, elencando le metodologie utilizzate, e di analizzare nel dettaglio le diverse problematiche da affrontare durante l'analisi dei dati.

Nel capitolo 2 saranno presentati gli obiettivi principali che la metodologia presentata ed il sistema realizzato hanno seguito, descrivendo nel dettaglio le diverse tecniche utilizzate per la realizzazione delle fasi necessarie all'individuazione dei pattern latenti nella struttura del web.

Nel capitolo 3 si descriverà la sperimentazione effettuata,completa di tabelle, grafici e commenti che evidenziano punti di forza e di debolezza individuati per ciascuna delle tecniche utilizzate per le diverse fasi eseguite dal sistema. Soffermandosi sulle novità introdotte con le metodologie presentate e cercando di confrontarle con quelle consolidate.

Infine nel capitolo 4 si parlerà della frontiera attuale dell'Informatica in tali campi confrontando similitudini e spunti di riflessione.

Capitolo 1

Informazioni latenti nel Web

Il progressivo aumento della dimensione del web e le informazioni in esso contenute fanno di esso il più grande insieme di dati da cui poter estrarre informazione velocemente e liberamente. Il Web è la sorgente informativa più eterogenea tra quelle esistenti, data la sua natura, e contenente dati prevalentemente non strutturati o semi-strutturati. Il problema non è sapere se le informazioni ci sono, ma riuscire a trovarle. Infatti anche se a prima vista il web sembra un insieme disordinato di pagine senza nessuna correlazione logica nella struttura, sia interna che nelle relazioni tra queste, in realtà possono essere trovate numerose correlazioni nascoste. Riuscire ad estrarre informazione da questa enorme mole di dati può rivelare pattern molto specifici ed utilizzabili in numerose attività per produrre vantaggi competitivi. Recenti sviluppi nell'ambito di svariate aree dell'informatica hanno portato negli ultimi anni alla creazione di sempre più nuove ed efficaci tecniche.

Di seguito sarà introdotto il contesto del sistema, le varie aree in cui si colloca e da cui attinge le metodologie ed il bagaglio di conoscenza.

1.1 Data Mining nel Web

Il Data Mining è l'insieme di tecniche e metodologie che hanno per oggetto estrazione di informazione utile, di un sapere o di una conoscenza a partire da grandi quantità di dati.
Tra le fasi più soggette a cambiamento al variare del dominio applicativo:

- estrazione di informazione implicita e nascosta da dati strutturati per renderla disponibile e direttamente utilizzabile;
- esplorazione ed analisi, eseguita in modo automatico o semiautomatico, su grandi quantità di dati allo scopo di scoprire pattern significativi.

In entrambi i casi i concetti di informazione e di significato sono legati strettamente al contesto in cui si esegue data mining, in altre parole un dato può essere interessante o trascurabile a seconda del tipo di applicazione in cui si vuole operare.

Il **Web Mining** è l'applicazione delle tecniche di Data Mining per la scoperta e l'estrazione di conoscenza o di pattern dal World Wide Web. Il web mining può essere diviso in tre principali categorie:

- *Web Usage Mining* è l'applicazione di tecniche di Data Mining per la scoperta di pattern e informazioni utili attraverso l'analisi di log immagazzinati dai web server e contenenti click stream. Obiettivo di questo campo è l'apprendimento dell'identità, origine e comportamenti degli utenti che navigano i siti web al fine di comprendere i loro bisogni e ad offrire loro servizi migliori attraverso una personalizzazione dell'esperienza web.
- *Web Structure Mining* consiste nell'estrazione di relazioni sconosciute o nascoste tra pagine web attraverso l'analisi della struttura ad hyperlink di un sito web (anche chiamato "grafo web"). Questo task verrà analizzato in dettaglio nella sezione 1.2.

- *Web Content Mining* consiste nell'estrazione ed integrazione di informazione utile e precedentemente sconosciuta dal contenuto delle pagine web. Ricadono in questo campo due principali tipologie di algoritmi: *i)* algoritmi capaci di raggruppare e classificare pagine web in funzione del loro contenuto testuale o del topic descritto; *ii)* algoritmi per estrarre pattern significativi all'interno del contenuto delle pagine web (per esempio liste di prodotti commerciali, professori, etc.). Sebbene questi algoritmi possano sembrare molto simili ai più famosi algoritmi di Data Mining e Text Mining, le pagine web hanno delle proprietà e peculiarità che rendono tali algoritmi non direttamente applicabili. Un'importante branca del Web Content Mining è rappresentato dal Web Information Extraction, il cui obiettivo è quello di estrarre dati strutturati da pagine web e integrarli in tabelle relazionali. In questo contesto il Web Content Mining può essere visto quindi come reperimento e immagazzinamento di informazioni.

Quando si confronta il web mining con il data mining tradizionale, ci sono tre principali differenze da considerare:

- **Dimensione**: la mole di dati, in task di Web Mining, può aumentare considerevolmente. Diventa necessario l'utilizzo di tecniche scalabili, ossia la capacità di un sistema di "crescere" ed essere utilizzabile in funzione alla crescita dei dati. Nel data mining tradizionale, processare un milione di record può essere considerato sopra la media, mentre nel web mining dieci milioni di pagine possono non essere abbastanza. Il Web è il primo mezzo di informazione dove il numero di produttori di informazioni è uguale al numero dei consumatori.

- **Dinamicità**: Ogni secondo migliaia sono create, distrutte e modificate migliaia di pagine Web. Questo rende il Web una rete di informazioni dinamica, dove la struttura e il contenuto dell'informazione cambiano frequentemente. Monitorare questi cambiamenti rimane un problema importante per molte applicazioni.

- **Eterogeneità**: L'eterogeneità del Web dipende sia dal formato delle

pagine che dalla scrittura. Nel primo caso, l'eterogeneità è dovuta al fatto che nel Web non esiste uno standard di formato per le pagine Web che possono ricadere in tre categorie: *i)*pagine non strutturate *ii)* pagine strutturate *iii)* pagine semi-strutturate.
Le pagine *non strutturate*, anche chiamate *free-text pages*, sono scritte in linguaggio naturale e possono essere applicate tecniche con un certo grado di affidabilità.
Le pagine *strutturate* sono normalmente ottenute da sorgenti di dati strutturate (e.g. database). Le tecninche di estrazione sono applicate usando l'individuazione di regole sintattiche. Le pagine *semi-strutturate* si posizionano al centro delle precedenti. Possiedono infatti un certo livello di struttura, nascosto nel testo. L'estrazione può avvenire cercando pattern nei tag HTML.
In questo caso l'eterogeneità è dovuta al fatto che le pagine Web son create da milioni di persone aventi differente cultura, abilità, linguaggio, ecc. Questo significa che le pagine potrebbero contenere la stessa informazione, ma presentata in maniera completamente diversa. Questo rende il processo di estrazione dell'informazione una sfida.

- **Connessione**: Il web è generalmente rappresentato come una rete di informazioni dove i nodi sono le pagine e gli archi sono gli hyperlink. Gli hyperlink fra le pagine di uno stesso sito e quelli fra le pagine di siti diversi, hanno diverse caratteristiche e funzionalità. All'interno di un sito servono ad organizzare i contenuti, mentre fra siti diversi servono a trasportare autorità alla pagina di destinazione. In questo caso significa che come persone ci fidiamo del contenuto di queste pagine.

- **Rumore**: Differentemente da altri mezzi di informazione, la pubblicazione di contenuti è libera e non richiede approvazione. Questo contribuisce all'aumentare del volume e della diversità dell'informazione, ma anche

- **Società virtuale**: Il web è generalmente rappresentato come una rete di informazioni dove i nodi sono le pagine e gli archi sono gli hyperlink. Gli hyperlink fra le pagine di uno stesso sito e quelli fra le pagine di siti

diversi, hanno diverse caratteristiche e funzionalità. All'interno di un sito servono ad organizzare i contenuti, mentre fra siti diversi servono a trasportare autorità alla pagina di destinazione. In questo caso significa che come persone ci fidiamo del contenuto di queste pagine.

Enormi sforzi son stati fatti per combinare informazione strutturata e non strutturata presente all'interno di pagine web, al fine di estrarre, indicizzare e reperire nuova conoscenza. La maggior parte delle soluzioni attuali trasformano il contenuto testuale delle pagine web in uno spazio vettoriale []https://www.jair.org/media/2934/live-2934-4846-jair.pdf . Modelli basati su spazi vettoriali sono fondamentali per task che coinvolgono il calcolo della similarità tra oggetti in cui oggetti simili sono caratterizzati da rappresentazioni vettoriali simili. Il modello vettoriale *termini-documenti* rappresenta il più semplice e utilizzato modello di rappresentazione di documenti testuali nel contesto del Text Mining. In tale modello il valore dell'elemento i-esimo in un vettore documento rappresenta il numero di volte che il termine i compare nel documento stesso. Gli algoritmi basati su modelli di rappresentazione vettoriali si basano sull'assunzione di indipendenza tra i termini all'interno di un documento e tra i documenti stessi. Le pagine web violano come tutti i documenti testuali la prima assunzione, inoltre i collegamenti ipertestuali definendo relazioni di interdipendenza tra le pagine stesse comportano la violazione della seconda assunzione.

Di conseguenza gli algoritmi di Web Mining devono tener conto delle informazioni relative alla struttura del sito web a cui appartengono al fine di estrarre informazioni più accurate.

1.2 Web Structure mining

Un enorme quantità di informazioni dal web sono rappresentati come dati semi-strutturati, ossia come combinazione di testo non strutturato e dati strutturati. I dati strutturati contenuti nelle pagine web sono tipicamente dati generati dinamicamente dalla struttura sottostante, come un database

relazionale, o da un template statico.

Estrarre questi dati è utile in molti domini applicativi perchè permette di ottenere ed integrare dati da diverse fonti, producendo così servizi migliori, informazioni personalizzate, meta-ricerche. Con sempre più aziende ed organizzazioni che disseminano informazioni in rete, l'abilità di estrarre questi dati dalle pagine web sta diventando estremamente importante.

Differentemente dai documenti testuali tradizionali, i dati strutturati nelle pagine web sono arricchiti da hyperlink che dividono l'informazione in molteplici ed interdipendenti pagine web. Questi hyperlink possono essere usati per identificare le entità provenienti dal mondo reale (e.g. pagine di professori, corsi, prodotti) e le relazioni che intercorrono fra di esse. Dato che i documenti web non sono nè strutturati come un database nè completamente non-strutturati come documenti testuali, le tecniche tradizionali di data mining o text mining non possono essere applicate direttamente.

Nel primo caso, le tecniche di data mining si basano sul presupposto che i dati usati per apprendere un modello condividono uno schema comune, avente delle tabelle ben definite con attributi, colonne, tuple e vincoli. Le pagine web non hanno questo presupposto perchè contengono dati eterogenei e gli hyperlink definiscono relazioni il cui significato può variare profondamente. Inoltre le pagine web sono codificate in HTML che, differentemente da altri linguaggi di markup, è stato progettato solo per la renderizzazione dei dati. Per questa ragione, il web può essere considerato un moderno legacy system, in quanto una grande quantità di dati non è facilmente accessibile e manipolabile direttamente.

Nel secondo caso, le tecniche di text mining falliscono nell'apprendere modelli accurati perchè richiedono collezioni di documenti scritti in modo consistente e non sono in grado di gestire informazioni complesse con elementi che possiedono diversi ruoli semantici e che forniscono diverse funzionalità. Infatti differentemente dai documenti testuali, le pagine web hanno molteplici rap-

presentazioni che forniscono differenti informazioni. Una è la rapresentazione testuale del testo HTML, l'altra è la rappresentazione visuale renderizzata da un web browser. Algoritmi di text mining si concentrano sulla rapresentazione testuale ed ignorano la rappresentazione visuale. Di conseguenza, esiste un forte bisogno nel campo dell'informatica di creare approcci e tecniche che usando informazioni testuali, strutturali e visuali sono capaci di estrarre uno schema da dati strutturati ed allineare i dati seguendo tale schema.

Il Web Structure Mining può essere diviso in due tipi:

- Estrazione di dati strutturati tra pagine web attraverso l'analisi del sito web. In questo caso un sito web è rappresentato come un grafo $G = (V, E)$ dove V è l'insieme delle pagine web e E è l'insieme degli hyperlink.
- Estrazione di dati strutturati contenuti in una pagina web, analizzandone la struttura ad albero basata su tag HTML ed XML.

Tra i più importanti algoritmi di web structure mining troviamo Page Rank[] e HITS[], i quali sfruttano la struttura ad hyperlink del Web per estrarre rank di pagine web.

1.3 Rappresentazioni vettoriali di pagine Web

Poichè La carenza di struttura delle pagine web obbliga all'effettuare dei passaggi preliminari per rendere processabili i dati disponibili. Metodi efficaci consistono nel rappresentare le pagine web come vettori, motivato dal fatto che task di machine learning richiedono input che sono matematicamente e computazionalmente convenienti da elaborare, considerando solo un sottoinsieme significativo dei dati in modo da astrarre ed eliminare informazioni ritenute non pertinenti al risultato finale. Questa fase è probabilmente la più importante in quanto algoritmi di apprendimento tanto bene quanto meglio i vettori ricavati rappresentano bene i dati di partenza. È necessario quindi estrarre rappresentazioni utili dai dati grezzi. Esistono molti modi

per ricavare questa correlazione, come reti neurali, matrici di co-occorrenza, dimensionality reduction o rappresentazioni del contesto in cui appare la parola. L'alternativa proposta in questa tesi consiste nell'utilizzare tecniche di word embedding per apprendere rappresentazioni utili. Il word embedding è il nome di un insieme di tecniche per il language modeling e per il feature learning nel campo del Natural language processing (NLP)[2]. Utilizzate in collezioni di documenti dove ad ogni parola viene associato un vettore, detto feature vector.

come sei arrivato da rappresentazione vettoriale a word embedding?Il word embedding può essere visto come una funzione parametrizzata

$$W : words \rightarrow \mathbb{R}^{\kappa} \tag{1.1}$$

che associa una parola in un dato linguaggio ad un vettore multidimensionale. Un esempio potrebbe essere:

$$W\left(''cat''\right) = (0.2, -0.4, 0.7, \ldots) \tag{1.2}$$

A parole simili corrisponde un vettore simile. Se si cambia una parola con un sinonimo, la validità della frase in esame non cambia (e.g. molti cantano bene $\rightarrow$ tanti cantano bene). Questo permette di generalizzare da una frase ad una classe di frasi simili o di capire se una frase è valida, ovvero se è formulata correttamente.

Questo non significa solo poter scambiare una parola con un sinonimo, ma anche di cambiare una parola con una altra in una classe simile (eg. il muro è rosso $\rightarrow$ il muro è blu) [4]. Questo può essere appreso analizzando il contesto della parola da analizzare. Ad esempio ci saranno molti casi in cui sono state osservate frasi valide di questo tipo, quindi cambiando la parola "rosso" con la parola "blu" porterebbe alla creazione una frase ugualmente valida.

Da questo potrebbe sembrare necessario osservare esempi relativi ad ogni parola per permetterci di generalizzarla. Comprendi tutte le parole che hai

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

Figura 1.1: Parole con vettori simili

già visto, ma non hai già visto tutte le frasi che riesci a capire. Questo è l'approccio delle reti neurali.

Analogie Il word embedding mostra un altra proprietà interessante anche se molto controversa: le analogie. Le analogie tra parole sembrano essere nascoste nella differenza dei loro rispettivi vettori [14].

$$W(''woman'') - W(''man'') \simeq W(''aunt'') - W(''uncle'') \tag{1.3}$$

Da questo si evince che c'è una correlazione tra delle parole e le rispettive forme del genere opposto in quanto appariranno in contesti simile, differenti solo per alcuni dettagli come pronomi o articoli. Stessa cosa per tra singolare e plurale [14]. Queste proprietà possono essere considerate effetti collaterali.

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Figura 1.2: Alcuni esempi di analogie

Non si è cercato di far apprendere il modello in modo da avere parole simili vicine fra loro. Questo sembra essere un punto di forza delle reti neurali nell'apprendere features che rappresentano bene i dati in modo automati-

co. invece di apprendere un rappresentazione dei dati specifica ed usarla per diversi task, è possibile apprendere un metodo per associare diversi tipi di dati in una singola rappresentazione. Queste tecniche sono note come transfer learning, metodi per applicare la conoscenza già appresa in contesti simili.

Un esempio può essere il word embedding di parole linguaggi diversi. Dato che parole simili saranno associate a vettori simili, parole con significato simile in una lingua e nell'altra finiranno vicine tra loro, così come i loro sinonimi. È possibile notare che anche parole di cui non si conosceva la traduzione o che avessero significati simili, sono finite vicine tra loro [21].

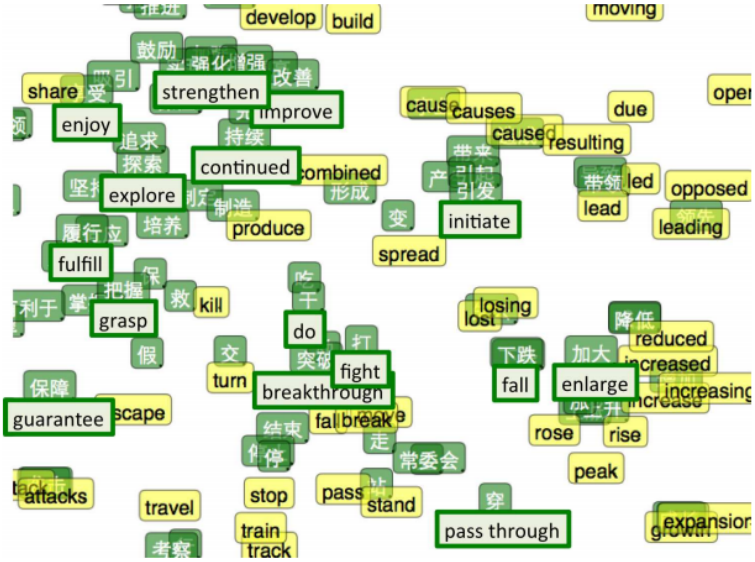

Figura 1.3: Visualizzazione con t-SNE di un word embedding bilingua.

1.3.1 Word2vec

Un algoritmo molto famoso di word embedding è il recente word2vec. L'algoritmo usa i documenti per far apprendere una rete neurale, massimizzando la probabilità condizionata del contesto data una parola, applicando il modello appreso ad ogni parola per ricavare il vettore corrispondente e calcolando il vettore della frase facendo la media dei vettori delle parole, costruisce la

matrice di similarità delle frasi ed usa PageRank per classificare le frasi nel grafo.

$$arg\max_{\theta} \prod_{(w,c)\in D} p\left(c|w;\theta\right) \tag{1.4}$$

L'obiettivo è di ottimizzare il parametro (θ) massimizzando la probabilità condizionata del contesto (c) data la parola (w). D è l'insieme di tutte le coppie (w, c). Per esempio: "ho mangiato un _____ al McDonald ieri sera", molto probabilmente restituirà "Big Mac".

Applicare il modello di ogni parola per ottenere il suo vettore corrispondente (Figura 1.4)

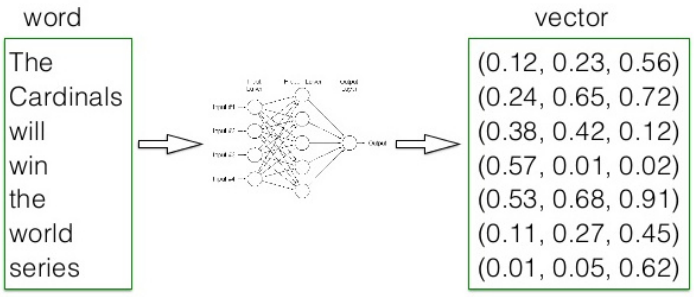

Figura 1.4: Vettori corrispondenti alle parole

Calcolare il vettore delle frasi facendo la media del vettore delle loro parole (Figura 1.5)

Costruire la matrice di similarità delle frasi (Figura 1.6)

Infine usare PageRank par classificare le frasi nel grafo. (Figura 1.7)

Word2vec è una rete neurale a due layer, sebbene non sia profonda (deep neural network) come spesso definita, trasforma il testo in modo che altre reti neurali possano comprenderlo. Prende in input un corpus di documenti

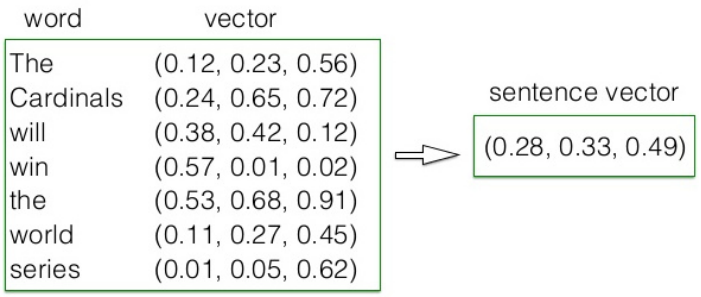

Figura 1.5: Vettori corrispondenti alle frasi

Figura 1.6: Matrice di similarità

e genera un insieme di vettori: feature vectors per ogni parola del corpus. I vettori restituiti sono rappresentazioni numeriche del contesto della singola parola.

Dati abbastanza dati, utilizzo e contesti, word2vec può apprendere rappresentazioni delle parole altamente accurate, basate sulle apparizioni della parola nei diversi contesti. Queste rappresentazioni possono essere usate per trovare associazioni fra parole o per raggruppare documenti e classificarli per argomento. La similarità fra i vettori può essere misurata attraverso la coseno similarità, dove nessuna similarità è espressa come un angolo di 90 gradi, mentre una similarità totale è data da un angolo di 0 gradi tra i vettori. Ad

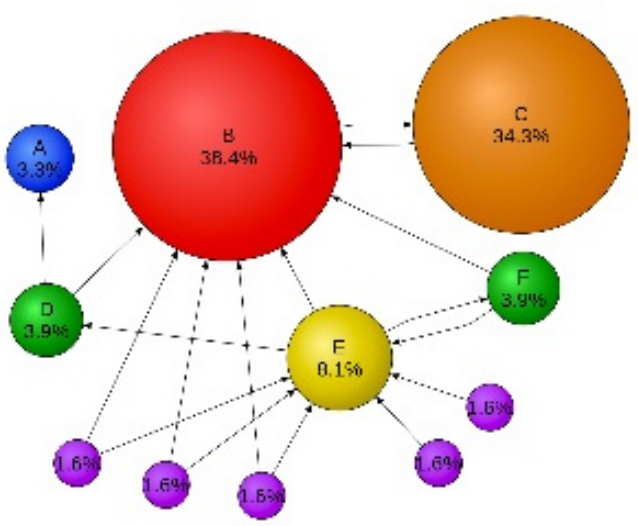

Figura 1.7: Assegna uno score utilizzando Pagerank

esempio il vettore relativo a "Sweden" è uguale al vettore "Sweden" mentre il vettore "Norway" ha una distanza di similarità di 0.760124.

Word2vec può apprendere rappresentazioni principalemente in due modi, o usando il contesto per predire la parola data (metodo conosciuto come "continous bag of word", o **CBOW**), o usando una parola per predire il contesto (**skip-gram**).

1.4 Data Visualization

Un nota sulla visualizzazione dei dati, campo in crescita data la corrispondente crescita su economie basate sull'informazione e sulla crescita dei dati generati (big data) portata avanti anche da campi relativamente nuovi nel campo dell'analisi dei dati, come Business Analytics, Business Intelligence, Data Science etc.

Tale disciplina è indirizzata a comunicare informazioni in modo chiaro e comprensibile, attraverso grafici, tabelle, diagrammi ecc. La visualizzazione

può spesso aiutare ad analizzare e ragionare sui dati, rendendo dati complessi molto più accessibili ed usabili.

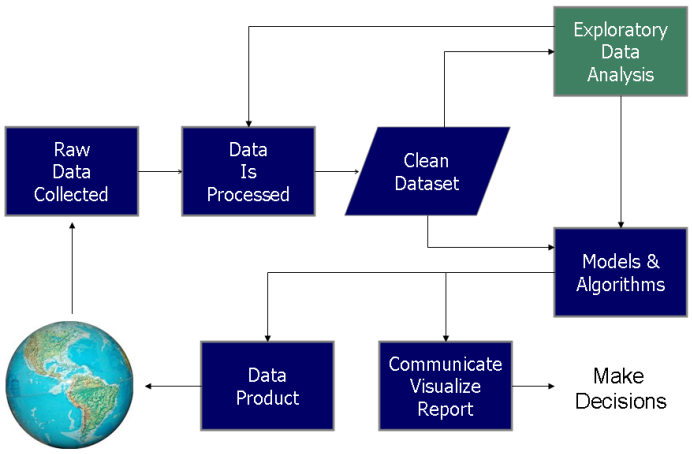

Figura 1.8: La visualizzazione dei dati è un passo fondamentale nell'analisi dei dati.

Capitolo 2

Il Sistema Url2vec

Il sistema Url2vec si propone come valida alternativa alle metodologie di clustering di siti web basate sul contenuto testuale delle pagine [16]. A dispetto della diversità delle pagine web nella rete, quelle che risiedono all'interno di una particolare organizzazione, spesso, condividono una certa struttura. Ad esempio, il sito web di un dipartimento di informatica conterrà pagine riguardante il personale, gli studenti, i corsi, la ricerca che saranno catalogate secondo determinati criteri. Saper sfruttare tale struttura può agevolare notevolmente i task di web mining. Esso infatti aggiunge ulteriore informazione, in modo da estrarre conoscenza strutturata e facilmente processabile. In questa tesi sono state realizzate tre macro componenti.

- Crawling delle pagine web. Questo è regolato da alcuni parametri che rendono il processo flessibile e altamente modificabile.
- Costruzione del dataset, ovvero la strutturazione delle informazioni ottenute nel processo precedente, secondo alcuni criteri necessari per le successive elaborazioni.
- Estrazione di informazione e clustering delle pagine web attraverso la combinazione dell'analisi del contenuto testuale e tecniche di word embedding.

Nelle sottosezioni che seguono si descrive il sistema esistente e le tecniche e gli algoritmi utilizzati per la realizzazione degli obiettivi preposti.

2.1 Web Crawling per l'estrazione dei dati

Le proprietà che caratterizzano le pagine Web rendono complicato il processo di estrazione di informazioni, soprattutto nel caso in cui i contenuti vengono generati dinamicamente o quando le pagine vengono create o eliminate spesso.

Un Web crawler, chiamato anche web spider o web robot, è un componente software. I suoi obiettivi principali sono:

- raccogliere il più velocemente ed efficientemente possibile pagine utili, insieme alla struttura ad hyperlink che le collega;
- aggiornare i contenuti o gli indici dei contenuti di siti Web;
- copiare tutte le pagine che visita per elaborazioni future, per poi indicizzarle cosi che gli utenti possano trovarle più velocemente;
- validare gli hyperlink e il codice HTML;
- eseguire il Web scraping.

Esso esplora una pagina alla volta, analizandone la struttura e gli hyperlink contenuti. Questi sono immagazzinati nella "frontiera" che inizialmente vuota, conserva tutte le pagine ancora da esplorare. L'ordine di esplorazione e le politiche di filtraggio degli hyperlink possono variare in base al risultato desiderato.

2.1.1 Proprietà del web crawler

Nel processo di crawling, data la natura non strutturata del web, è necessaria l'applicazione di numerose tecniche e metodologie per un corretto funzionamento.

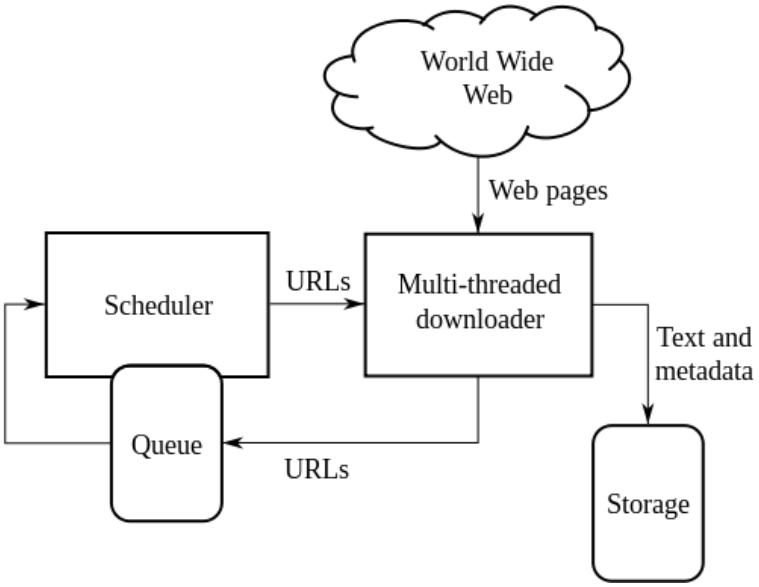

Figura 2.1: Architettura di un web crawler

Normalizzazione degli URL

Il termine di normalizzazione, chiamato anche canonicalizzazione di URL, si riferisce al processo di modifica e standardizzazione di un URL in una maniera consistente, ad esempio alcuni siti Web mettono a disposizione gli stessi file o i medesimi contenuti attraverso URL differenti.

`http://domain.com/products/page.php?product=smartphone`
`http://domain.com/products/smartphone.php`

`http://www.domain.edu/courses`
`http://www.domain.edu/courses/index.html`

Le due coppie di URL nell'esempio puntano agli stessi contenuti. Altri casi sono URL che differscono solo per il protocollo ("`http://`" o "`https://`") o l'omissione della stringa "`www`". Effettuando la normalizzazione, si sceglie un URL come formato di riferimento per accedere ad un determinato contenuto. Ci sono diversi tipi di normalizzazione che possono essere usati, come la conversione in minuscolo, rimuovere i "." e ".." portando gli URL relativi

ad URL assoluti, aggiungere slash finali al componente di percorso non vuoto.

Una soluzione consiste nella creazione di un dizionario che ha come chiavi gli hashcode del contenuto testuale delle pagine e come valori una lista di tutti i diversi URL che hanno quel contenuto. In questo modo si riduce il problema ad una sola operazione così da annullare le relazioni e i vari di pattern da scovare nall'analisi degli URL per capire se sono la stessa pagina o meno.

Ricerca all'interno dello stesso dominio

Per estrarre informazione e per il successivo processo di clustering delle pagine web, è necessario che le pagine estratte si riferiscano allo stesso dominio, o in altri termini, che appartengano allo stesso sito web.
Questo viene effettuato per sfruttare la struttura gerarchica del dito web rivolto principalmente riuscire a raccogliere le informazioni nascoste nel grafo e sopratutto negli hyperlink.

Dimensione della ricerca

Anche limitando la ricerca ad un solo dominio, la dimensione delle pagine da esplorare, e quindi la grandezza della frontiera, può aumentare considerevolmente o addirittura portare il processo di crawling a divergere.

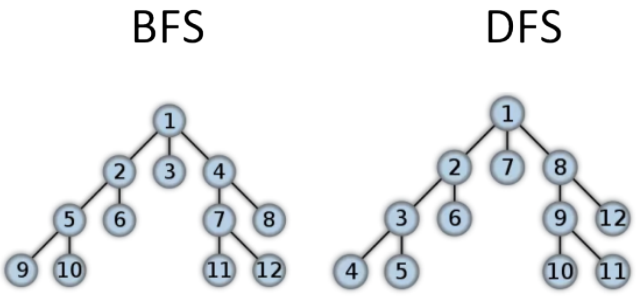

Figura 2.2: Differenze tra ricerca in ampiezza e ricerca in profondità

È stata utilizzata una ricerca in ampiezza con un limite variabile di profondità. Ponendo un limite all'esplorazione del grafo delle pagine si garantisce la la terminazione del processo di crawling e utilizzando una ricerca in ampiezza si dà priorità alle pagine vicine al nodo radice (comunemente l'homepage). Questa scelta è stata dettata anche dal cercare di evitare le cosiddette "spider traps". Queste sono dei meccanismi utilizzati, intenzionalmente o involontariamente, dai server oggetto di crawling, che possono portare ad una generazione dinamica e potenzialmente infinita in URL univoci, e quindi considerati come pagine diverse. Questo può essere evitato non aggiungendo alla frontiera URL che contengono il carattere "?", ma questo non è sempre efficace.

Inoltre è stato osservato che nei siti web entity-oriented, le informazioni ricercate si trovano quasi sempre nei primi livelli di gerarchia [7].

Restrizione dell'esplorazione

La restrizione può essere effettuata sulle pagine da esplorare o dalla frequenza di richieste che è possibile effettuare.

Il robots exclusion protocol è uno standard che consiste in un file (robots.txt), posto alla radice della gerarchia di un sito Web. In pratica, il file indica le regole utilizzate dai crawler per applicare restrizioni di analisi sulle pagine di un sito web.

Molte volte i crawler non sono ben accetti, in quanto possono rallentare pesantemente la navigazione. Se server effettua dei controlli sulle richieste ricevute e queste hanno una frequenza troppo elevata o seguono uno schema riconducibile ad una macchina queste possono essere ignorate. Può capitare che richieste continue e non curanti delle restrizioni imposte portino a bandire l'indirizzo IP del servizio trasgressore.
Per evitare tali conseguenze è opportuno seguire una certa etica nell'operazione di crawling.

2.1.2 Estrazione delle liste

Riuscire a strutturare dati non strutturati può rivelarsi ostico. Molti tentativi, più o meno efficaci, sono stati effettuati a tale proposito.
In questa tesi uno degli obiettivi è stato quello di prendere in considerazione i collegamenti all'interno delle pagine web, seguendo i percorsi generati dalla concatenazione di più hyperlink. Questa scelta è stata effettuata sulla base dei recenti progressi nel campo del natural language processing (NLP) [20]. Esplorando la struttura del grafo del web, data la forte connessione che esiste fra i suoi nodi, estrarre informazione può rivelarsi un operazione tutt'altro che banale.

È stato introdotto il concetto di **lista**. Per permettere una migliore visualizzazione dell'informazione descritta, quasi tutte le pagine web vengono formattate utilizzando regole CSS. Di conseguenza, per poter effettuare correttamente questo task, bisogna prima elaborare la pagina Web con tutte le informazioni grafiche sui nodi HTML e solo successivamente si può procedere alla loro estrazione. Grazie alle informazioni ricavate dall'HTML della pagina Web e dalla posizione e dimensione dei singoli nodi, è possibile stabilire una struttura gerarchica ad albero dei nodi che la compongono. Questa struttura gerarchica ad albero permette di scoprire i record, ovvero dati strutturati (ad esempio provenienti da database) che sono allineati orizzontalmente, verticalmente e anche strutturalmente (elementi HTML ul, li, . . .); permette quindi di individuare dei gruppi di record, ovvero delle liste di record.
L'individuazione delle liste di raggiungere pagine che contengono elementi simili a quelli contenuti nella pagina che si sta visualizzando. In questo modo le pagine dovrebbero essere accessibili solo attraverso percorsi predeterminati e non in maniera fortemente connessa. I nodi e gli archi risultanti risultano così un sottoinsieme di quelli originali.

2.2 Costruzione del dataset

I dati estratti nel processo vanno organizzati e ampliati in modo da garantire l'accesso e l'elaborazione in maniera agevole.
Il processo di crawling restituisce il grafo delle pagine web e il contenuto testuale di ogni pagina esplorata a queste va aggiunta la generazione delle sequenze. Inoltre per un minor spreco di risorse si è optato per la conversione degli URL in codici, ovvero una associazione univoca fra un URL e un codice (e.g. un numero) molto più corto, così da risparmiare tempo di elaborazione e spazio di archiviazione. Segue una analisi sul processo di generazione delle sequenze.

2.2.1 Generazione delle sequenze

Le sequenze rappresentano un percorso che un attraversatore casuale della rete seguirebbe cliccando su un hyperlink a caso fra tutti quelli disponibili nella pagina corrente (o eventualmente nelle liste). Questi percorsi, chiamati **random walk** (o passeggiate aleatorie), sono stati largamente utilizzati in molti algoritmi sui grafi e sul web [1] in quanto buone approssimazioni di comportamenti casuali. Il problema di questa tecnica applicata al web si presenta quando l'attraversatore casuale arriva ad una pagina priva di hyperlink. La soluzione più diffusa consiste nell'effettuare un "salto" verso una qualsiasi altra pagina quando non ci sono outlink da seguire.

Qui il problema non si pone, in quanto le sequenze generate hanno una lunghezza fissata prima dell'esecuzione e se la generazione dovesse bloccarsi, la sequenza risultante sarà solo più piccola. Questa scelta è dovuta dal fatto che l'informazione cercata scaturisce da percorsi reali di navigazione e non necessita una lunghezza obbligatoria da rispettare, in quanto le sequenze possono essere viste come frasi di un testo, dove le parole sono gli URL.

Per motivi di sperimentazione sono stati implementati tre tipi diversi di

random walk, utilizzabili modificando i parametri di esecuzione dell'algoritmo.

2.2.2 Random Walk

In matematica, un Random Walk è la formalizzazione dell'idea di prendere passi successivi in direzioni casuali. Matematicamente parlando, è il processo stocastico più semplice, il processo markoviano. Qui sono utilizzati per ricavare percorsi pseudo casuali da attraversamento del grafo del web per ottenere sequenze di URL collegati semanticamente.

In una passeggiata aleatoria monodimensionale si studia il moto di una particella puntiforme vincolata a muoversi lungo una retta nelle due direzioni consentite. Ad ogni movimento essa si sposta (a caso) di un passo a destra (con una probabilità fissata p) o a sinistra con una probabilità $1 - p$, ed ogni passo è di lunghezza uguale e indipendente dagli altri.

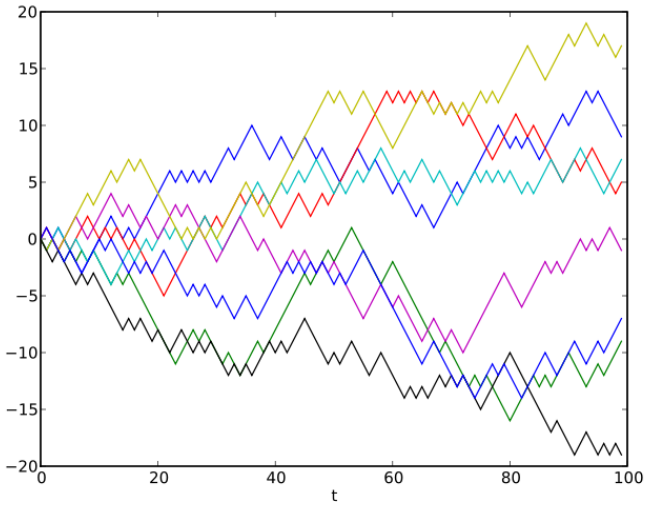

Figura 2.3: Rappresentazione visuale di 8 random walk monodimensionali.

In una passeggiata aleatoria bidimensionale si studia il moto di una par-

ticella vincolata a muoversi sul piano spostandosi casualmente ad ogni passo a destra o a sinistra con probabilità $\frac{1}{2}$, verso l'alto o verso il basso con probabilità $p = \frac{1}{2}$. In pratica ad ogni passo può compiere un movimento lungo una delle quattro diagonali con probabilità $\frac{1}{4}$. Ci si chiede con che probabilità la particella tornerà al punto di partenza.

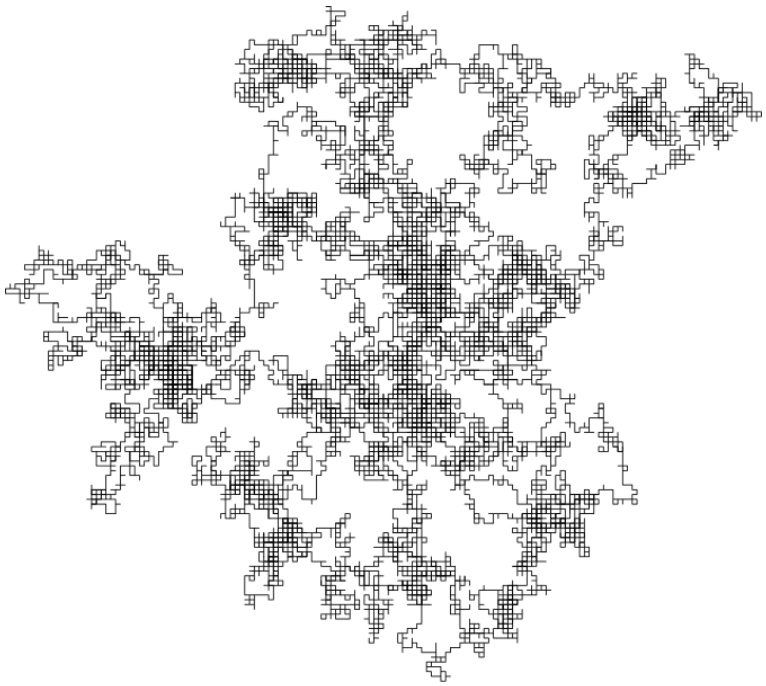

Figura 2.4: Rappresentazione visuale di un random walk di 25.000 passi su due dimensioni.

Queste passeggiate aleatorie possono trovare effettivi riscontri in natura come il traiettoria percorso da una particella in un liquido o in un gas, il tragitto di un animale affamato o anche il prezzo di un titolo fluttuante o la situazione finanziaria di un giocatore d'azzardo. Tutti questi esempi possono essere espressi come random walk, anche se in natura potrebbero non essere veramente casuali.

Un popolare modello di random walk è quello su un reticolo regolare, dove ad ogni passo si segue un determinato percorso basandosi su una qualche distribuzione di probabilità. Nel caso più semplice si può solo "saltare" sul

sito vicino. In un semplice random walk simmetrico in un reticolo localmente finito, le probabilità di saltare su ognuno dei siti vicini è la stessa.

Per definire un random walk formalmente, prese indipendenti variabili casuali $Z_1, Z_2, Z_3, \ldots$ dove ogni variabile è o 1 o -1, con una probabilita del 50% per ognuno dei due casi, e dato

$$S_0 = 0 \tag{2.1}$$

$$S_n = \sum_{j=1}^{n} Z_j \tag{2.2}$$

la serie $\{S_n\}$ è chiamata random walk semplice in $\mathbb{Z}$.

Si consideri un attraversatore casuale della rete (random surfer) che, partendo da un pagina web, esegue un passo alla volta in questo modo: ad ogni iterazione, dalla pagina corrente, procede verso una pagina a caso tale che esiste un link che dalla pagina corrente punta a questa.

Procedendo da pagina a pagina, visiterò alcuni nodi, più spesso di altri; intuitivamente, saranno nodi con molti link entranti da altri nodi frequentemente visitati.

Ci sono alcuni problemi. Che succede se si arriva ad una pagina che non ha link in uscita? In questo caso è necessario introdurre un'altra operazione: il teletrasporto. In questa operazione l'attraversatore casuale salta dal nodo corrente ad un qualsiasi altro nodo sulla rete. Se N è il numero totale dei nodi nel grafo, l'operazione di teletrasporto porta l'attraversatore verso ogni nodo con probabilità $\frac{1}{N}$. Potrebbe anche trasportarsi sulla posizione corrente con probabilità di $\frac{1}{N}$. Questa operazione è chiamata quando si arriva ad un non senza nodi in uscita o con una probabilità d data, con $0 < d < 1$.

Random Walk

Questo è il caso standard, ovvero si parte da un nodo casuale del grafo e si segue ogni volta un arco a caso fra quelli disponibili, fino al raggiungimento della lunghezza prefissata o all'impossibilità di proseguire.
Da notare che questo processo e il precedente non sono completamente separati, in quanto la scelta di un nodo casuale e la consecutiva traiettoria, possono portare all'esplorazione di pagine non precedentemente visitate. È quindi necessario mantenere aggiornato il grafo immagazzinato.

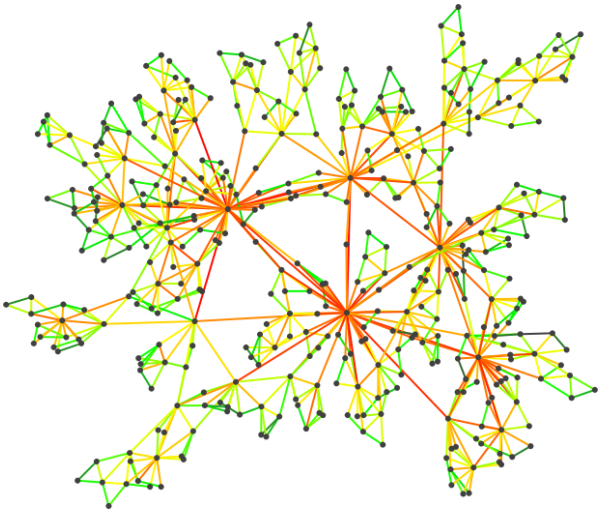

Figura 2.5: Random walk sul grafo.

Random Walk con partenza fissa

Qui l'unica differenza consiste nel punto di partenza del cammino. Infatti si può partire in un nodo prefissato del grafo (generalemnte l'homepage di un sto web), in modo da esplorare più percorsi possibili avente quel nodo come origine.

Random Walk attraverso le Liste

Qui invece si può seguire uno dei due approcci precedenti, ma con il vincolo delle liste, quindi limitando la camminata ad un sottoinsieme di quella precedente.

Algorithm 1 Crawling BFS

```
url ← homepage;  //Pagina da cui iniziare
queue ← empty  //Coda per la BFS
analyzedVertex.add(url);  //Insieme di url già visitati
maxDepthurl;  //Massima profondità di esplorazione

Begin
while queue ≠ empty do
    urlToAnalyze ← queue.dequeue()
    if urlToAnalyze.depth ≤ maxDepth then
        outlinks ← urlToAnalyze.getOutlinks()
    else
        urlToAnalyze
    end if
    if outlinks ≠ null then
        outlinks ← urlToAnalyze.getOutlinks()
        analyzedVertex.add(outlinks)
        serialize(urlToAnalyze)
        for each link in outlinks do
            serialize(link)
            queue.enqueue(link, urlToAnalyze + 1)
        end for
    end if
end while
End
```

Algorithm 2 Generazione delle sequenze

```
numRandomWalks; //Numero di Random Walk da generare
lengthRandomWalks //Lunghezza dei Random Walk

Begin
node ← RANDOMNODE
for i ← 0 → numRandomWalks do
    sequence.add(node)
    while sequence.length ≤ lengthRandomWalks do
        if node.hasOutlinks() then
            node ← node.getOutlinks(RandomIndex)
            sequence.add(node)
        else
            break
        end if
    end while
    serialize(sequence)
end for
End
```

2.2.3 esempio di dataset

Di seguito sono elencati file generati nel processo di crawling e di generazione delle sequenze. Gli esempi riportati di seguito sono stati ottenuti analizzando il sito del dipartimento di informatica di Urbana, IL: `www.cs.illinois.edu`

urlsMap.txt

Contiene la associazioni fra gli URL e il relativo codice identificativo. Questo è dovuto dalle ragioni spiegate in precedenza, ovvero ridurre i tempi di elaborazione e spazio di archiviazione.

```
http://cs.illinois.edu,3
http://cs.illinois.edu/prospective-students,4
http://cs.illinois.edu/current-students,5
```

```
http://cs.illinois.edu/courses,6
http://cs.illinois.edu/alumni,7
http://cs.illinois.edu/research,8
http://cs.illinois.edu/news,9
http://cs.illinois.edu/partners,10
http://cs.illinois.edu/about-us,11
.  .  .
```

vertex.txt

Contiene il contenuto testuale di ogni pagina esplorata. Ogni riga è quindi formata da il codice identificativo di un URL e il relativo contenuto.

```
1 department of computer science at illinois engineering at ...
2 prospective students department of computer science at ...
3 current students department of computer science at ...
4 courses department of computer science at illinois ...
5 alumni department of computer science at illinois ...
6 research department of computer science at illinois ...
7 news department of computer science illinois engineering ...
8 partners department of computer science at illinois ...
9 about us department of computer science at illinois ...
.  .  .
```

edges.txt

Questo è il file principale per la generazione delle sequenze, qui sono immagazzinate tutte le relazioni fra le pagine, ovvero gli archi che le collegano.

```
1 1
```

```
1 2
1 3
1 4
1 5
1 6
1 7
1 8
1 9
.  .  .
```

sequencesIDs.txt

Contiene le sequenze generate. I codici relativi alle pagine web sono separati da " -1 " e la linea finisce con un " -2 ". Da notare che sono riportate le sequenze che partono da un nodo casuale del grafo.

```
137 -1 2 -1 27 -1 8 -1 52 -1 53 -1 8 -1 8 -1 10 -1 13 -1 -2
506 -1 5 -1 14 -1 11 -1 6 -1 2 -1 27 -1 114 -1 111 -1 11 -1 -2
424 -1 4 -1 12 -1 6 -1 8 -1 53 -1 4 -1 7 -1 12 -1 8 -1 -2
616 -1 5 -1 6 -1 8 -1 8 -1 9 -1 1 -1 21 -1 6 -1 3 -1 -2
51 -1 7 -1 7 -1 38 -1 38 -1 25 -1 103 -1 27 -1 113 -1 12 -1 -2
429 -1 10 -1 3 -1 6 -1 4 -1 11 -1 8 -1 9 -1 9 -1 3 -1 -2
783 -1 421 -1 5 -1 9 -1 7 -1 5 -1 2 -1 8 -1 2 -1 24 -1 -2
506 -1 5 -1 8 -1 52 -1 53 -1 25 -1 40 -1 13 -1 11 -1 13 -1 -2
638 -1 63 -1 153 -1 62 -1 63 -1 152 -1 63 -1 155 -1 13 -1 7 -1 -2
.  .  .
```

sequencesIDsFromHomepage.txt

Contiene le sequenze generate che partono da uno stesso nodo. La generazione di questo file avviene esplicitando il nodo di origine di ogni sequenza

nella fase di generazione.

```
1 -1 8 -1 2 -1 14 -1 2 -1 2 -1 10 -1 14 -1 66 -1 3 -1 -2
1 -1 18 -1 39 -1 8 -1 8 -1 24 -1 1 -1 4 -1 7 -1 25 -1 -2
1 -1 23 -1 -2
1 -1 16 -1 10 -1 3 -1 29 -1 29 -1 4 -1 25 -1 97 -1 108 -1 -2
1 -1 20 -1 20 -1 1 -1 11 -1 3 -1 2 -1 9 -1 10 -1 13 -1 -2
1 -1 25 -1 48 -1 48 -1 44 -1 42 -1 4 -1 7 -1 38 -1 38 -1 -2
1 -1 25 -1 115 -1 4 -1 11 -1 11 -1 1 -1 2 -1 26 -1 13 -1 -2
1 -1 24 -1 4 -1 9 -1 60 -1 56 -1 6 -1 25 -1 113 -1 116 -1 -2
1 -1 21 -1 25 -1 135 -1 32 -1 13 -1 4 -1 25 -1 149 -1 59 -1 -2
.  .  .
```

2.3 Clustering

Sono un insieme di tecniche di analisi multivariata dei dati volte alla selezione e raggruppamento di elementi omogenei in un insieme di dati [19]. Le tecniche di clustering si basano su misure relative alla somiglianza tra gli elementi. In molti approcci questa similarità (o dissimilarità) è concepita in termini di distanza in uno spazio multidimensionale. La bontà delle analisi ottenute dagli algoritmi di clustering dipende molto dalla scelta della metrica, e quindi da come è calcolata la distanza. Gli algoritmi di clustering raggruppano gli elementi sulla base della loro distanza reciproca, e quindi l'appartenenza o meno ad un insieme dipende da quanto l'elemento preso in esame è distante dall'insieme stesso dividendo gli elementi in più cluster(soft/fuzzy clustering) o in un solo cluster(hard cluster). Le tecniche di clustering si possono basare principalmente su due filosofie: partizionale e gerarchico.

Partizionale

Gli algoritmi di clustering di questa famiglia creano una partizione delle osservazioni minimizzando una certa funzione di costo:

$$\sum_{j=1}^{k} E\left(C_j\right) \tag{2.3}$$

dove k è il numero desiderato di cluster, C_j è il j-esimo cluster e $E : C \to \mathbb{R}^+$ è la funzione di costo associata al singolo cluster. L'algoritmo più famoso appartenente a questa famiglia è il k-means, chiamato così da MacQueen nel 1967 [13].

Gerarchico

Nel clustering gerarchico, invece, è necessario individuare il cluster da suddividere in due sottogruppi. Per questa ragione sono necessarie funzioni che misurino la compattezza del cluster, la densità o la distanza dei punti assegnati ad un cluster. Le funzioni normalmente utilizzate nel caso divisivo sono:

Single-link proximity Calcola la distanza tra i due cluster come la distanza minima tra elementi appartenenti a cluster diversi:

$$D\left(C_i, C_j\right) = \min_{x \in C_i, y \in C_j} d\left(x, y\right) \tag{2.4}$$

Average-link proximity Questa funzione calcola la distanza tra i due cluster come la media delle distanze elementi:

$$D\left(C_i, C_j\right) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i, y \in C_j} d\left(x, y\right) \tag{2.5}$$

Complete-link proximity Questa funzione valuta la distanza massima tra due punti interni ad un cluster. Tale valore è noto anche come 'diametro del cluster': più tale valore è basso, più il cluster è compatto:

$$D\left(C_i, C_j\right) = \max_{x \in C_i, y \in C_j} d\left(x, y\right) \tag{2.6}$$

Distanza tra centroidi Questa funzione valuta la distanza massima tra due punti interni ad un cluster. Tale valore è noto anche come 'diametro del cluster': più tale valore è basso, più il cluster è compatto:

$$D\left(C_i, C_j\right) = d\left(\hat{c}_i, \hat{c}_j\right) \tag{2.7}$$

Nei casi precedenti, $d\left(x, y\right)$ indica una qualsiasi funzione distanza su uno spazio metrico.

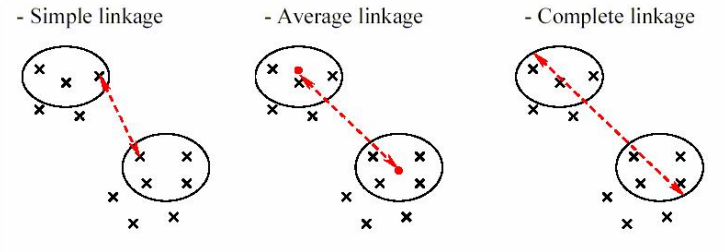

Figura 2.6: Differenze fra i vari tipi di funzioni distanza

2.3.1 Graph Clustering

Un grafo è una coppia ordinata $G = (V, E)$ di insiemi, con V insieme dei nodi ed E insieme degli archi, tali che gli elementi di E siano coppie di elementi da V da $E \subseteq V \times V$ segue in particolare che $|E| \leq |V|^2$.

I grafi sono oggetti discreti che permettono di schematizzare una grande varietà di situazioni e di processi e spesso di consentirne delle analisi in termini quantitativi e algoritmici.

Nello studio di reti complesse, è possibile trovare gruppi di nodi fortemente connessi, che possono essere raggruppati in comunità (potenzialmente sovrapposte). Questa disomogeneità di connessioni suggerisce che esiste una certa divisione naturale all'interno della rete. Nel caso particolare di strutture non sovrapposte, la ricerca di comunità implica la divisione della rete in gruppi di nodi con fortemente connessi internamente e connessioni sparse fra i gruppi. Una definizione più generale è basata sul principio che coppie di nodi sono più probabilmente connessi se fanno parte della stessa comunità, e meno probabilmente connessi se non condividono la stessa comunità.

Le comunità sono molto comuni all'interno delle reti. Le reti sociali includono gruppi di comunità che condividono la posizione, gli interessi, l'occupazione ecc. Essere in grado di individuare queste sotto-strutture all'interno di una rete può fornire indizi su come come funziona la rete in considerazione o la topologia che influenza i nodi. Questi indizi possono essere utili per implementare algoritmi sui grafi. Molti metodi di community detection sono stati sviluppati con diversi livelli di successo.

Minimum-cut method

In questo metodo, la rete è divisa in un numero predeterminato di parti, generalmente della stessa grandezza, scelte in modo che il numero degli archi tra i gruppi è minimizzato. Questo metodo funziona bene in molte applicazioni per le quali è stato ideato, ma non è la scelta migliore per scovare comunità in reti generali, dato che troverà comunità indistintamente dal fatto che queste ci siano o meno e troverà solo un numero fissato di comunità.

Hierachical-clustering

Un altro metodo per scovare sotto-strutture conesse nelle reti viene effettuato tramite algoritmi di clustering gerarchico. Con questo approccio si definisce

una misura di similarità fra coppie di nodi. Misure comunemente usate sono la coseno similarità, l'indice di Jaccard e la distanza di Hamming fra le righe della matrice di adiacenza. Poi i gruppi di nodi simili vengono raggruppati in comunità.

Girvan-newman algorithm

Un altro algoritmo molto utilizzato è quello di Girvan–Newman. Questo algoritmo identifica all'interno della rete, gli archi che uniscono community diverse e li rimuove, isolandole. L'identificazione di tali archi è effettuata applicando una misura nota della teoria dei grafi: la **betweenness centrality**. Questa assegna un valore ad ogni arco, che è alto tanto più l'arco è attraversato nel cammino più breve (geodesico) che collega due qualsiasi nodi della rete.

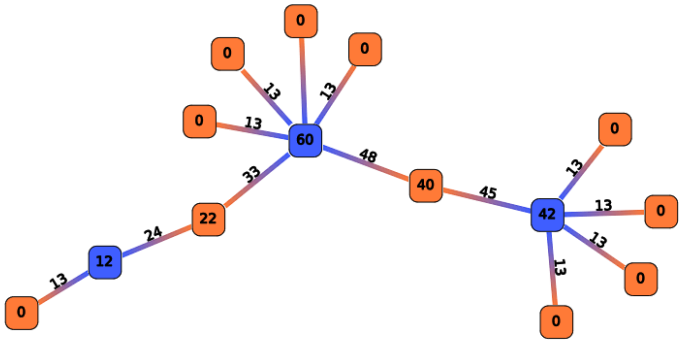

Figura 2.7: Betweenneess centrality score

Modularity maximization

La modularità è una misura che viene attribuita al grafo. Questa compara la densità all'interno dei cluster con la densità fra di essi. Indica una certa divisione intrinseca e viene utilizzata per conoscere "quanto" un grafo è separato.

Nonostante i suoi svantaggi uno dei metodi più utilizzati per il community detection è la massimizzazione della modularità. Questo approccio scova strutture connesse tramite la ricerca della miglior divisione di una rete in modo che la modularità risulti massimizzata.

Dato che effettuare un confronto su tutte le possibili combinazioni è solitamente impraticabile, gli algoritmi di questa famiglia si basano su metodi di ottimizzazione approssimati quali algoritmi greedy, cioè che cercano di ottenere una soluzione ottima da un punto di vista globale attraverso la scelta della soluzione considerata migliore ad ogni passo locale. Un famoso approccio di questo tipo è il metodo Louvain, che ottimizza le community locali iterativamente, fin quando la modularità globale non può più essere migliorata.

L'accuratezza di questi algoritmi, comunque, è dibattuta, in quanto è stato dimostrato che molte volte fallisce nell'individuare cluster più piccola di una certa soglia, dipendente dalla grandezza della rete.

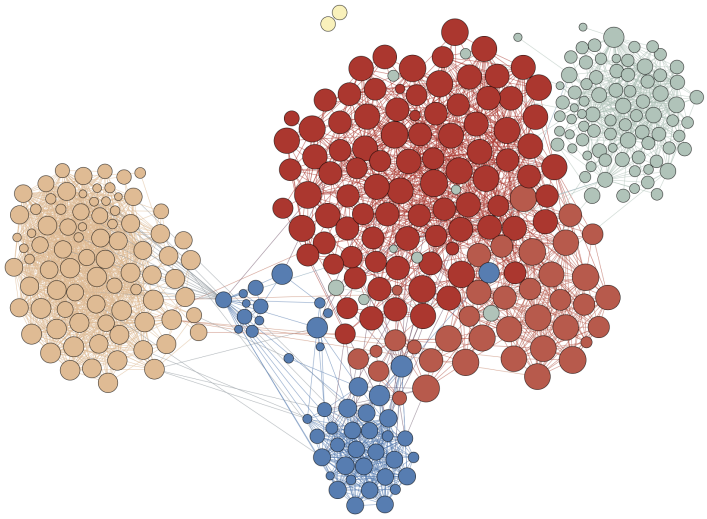

Figura 2.8: Modularity score

Clique-based methods

Le cricche (cliques) sono sottografi dove ogni nodo è collegato con ogni altro nodo nella cricca. Dato che i nodi non possono essere più connessi di così, non è sorprendente che ci siano molti metodi in community detection che si basano su questo approccio. È da notare che dato che un nodo può far parte di più di una cricca, quindi può far parte di più community contemporaneamente, questi metodi restituiscono strutture sovrapposte. Un approccio consiste nel trovare cricche tali che non siano sottografi di altre cricche. Un classico algoritmo per scovare tali strutture è quello di Bron-Kerbosch.

2.3.2 Vector Clustering

Gli algoritmi di clustering in uno spazio vettoriale seguono un altro approccio. Qui viene preso in considerazione la vicinanza (o la distanza) degli elementi rappresentati come punti su un iperpiano. Il feature vector associato può avere grandi dimensioni e può essere ottenuto utilizzando diverse metodologie, dipendentemente dal tipo di dato eleborato.

Hierarchical Clustering

Anche qui il clustering gerarchico è molto utilizzato, seguendo un approccio divisivo (top-down) o agglomerativo (bottom-up), l'idea è quella di unire (o separare) elementi in base alla loro vicinanza, seguendo uno degli approcci già descritti, costruendo così una struttura chiamata dendrogramma che raggruppa gli elementi ad ogni livello. La differenza risiede nel come viene calcolata la distanza.

Centroid-based clustering

Nell'approccio basato sui centroidi, i cluster sonorappresentati da un vettore centrale, che non è neccessariamente un membro del dataset. Quando il nu-

mero dei cluster è prefissato ad un numero k, la seguente definizione formale può essere applicata: vengono definiti k centroidi e si prosegue assegnando ogni elemento al centroide più vicino, tale che il quadrato delle distanze dal cluster è minimizzato. Molti algoritmi di questa famiglia richiedono che il parametro k sia stabilito in precedenza, che è il loro più grande svantaggio. Inoltre solitamente vengono trovati cluster di grandezza simile, dato che verrà assegnato un elemento al centroide più vicino. Questi metodi partizionano lo spazio dei dati in una struttura conosciuta come diagramma di Voronoi. Nonostante questo, rimangono tra degli approcci più utilizzati ed efficaci.

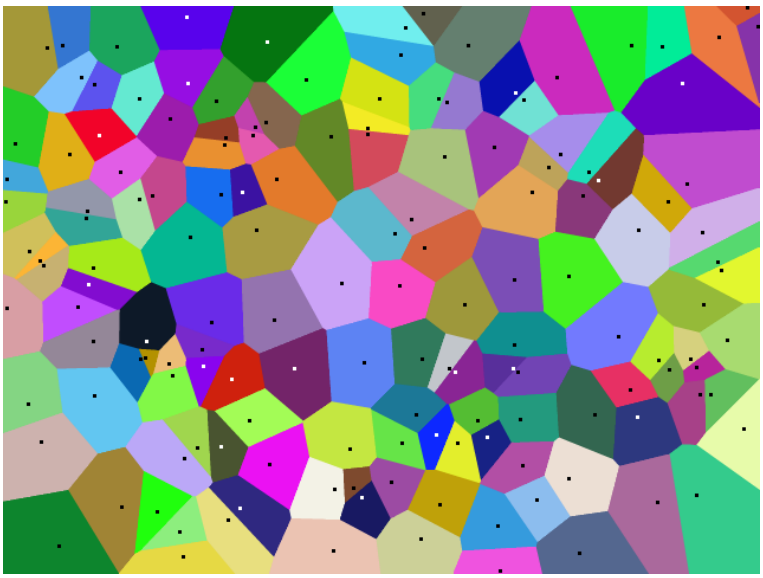

Figura 2.9: Diagramma di Voronoi

Da notare anche la somiglianza concettuale con l'algoritmo di classificazione KNN (k nearest neighbor).

Distribution-based clustering

Il raggruppamento avviene analizzando l'intorno di ogni punto dello spazio. In particolare, viene considerata la densità di punti in un intorno di raggio fissato. Si basano sul considerare collegati due punti che si trovano all'interno di una certa distanza limite. I cluster sono definiti come aree con più

alta densità rispetto al resto del dataset. Elementi in un area meno denso sono spesso considerati rumore, quindi come non facenti parte di nessun cluster.

Uno degli svantaggi di questi algoritmi è che si aspettano un certo tipo di densità comune a tutti i cluster. Inoltre non eccellono nel scovare cluster presenti in molti dati del mondo reale.

Document Clustering

Agisce sempre nello spazio vettoriale, si differenzia principalmente nelle operazioni di pre-processing finalizzate ad ottenere un feature vector utilizzabile. Un approccio efficace consiste nel rappresentare i documenti come vettori dove ogni dimensione rappresenta la frequenza di occorrenza di una parola del vocabolario, un insieme di parole precedentemente costruito utilizzando tutte le parole del corpus.

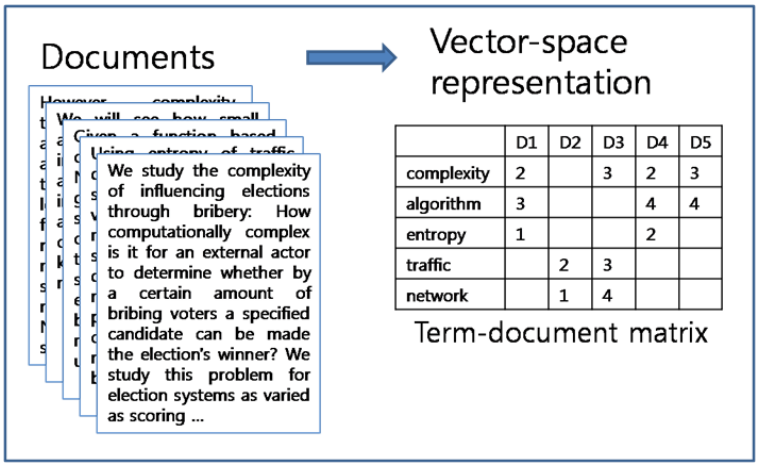

Figura 2.10: Matrice termini-documenti. Ogni riga rappresenta un singolo termine ed ogni colonna rappresenta un singolo documento

Term-Frequency (TF) misura quante volte un termine appare in un documento. Dato che ogni documento ha lunghezza differente, è possibile che un termina possa apparire molte più volte nei documenti più lunghi che in

quelli più corti. Quindi può essere necessario dividere la frequenza dei termini per documenti aventi la stessa lunghezza.

Inverse-Document-Frequency (IDF) un altro aspetto da considerare è a frequenza di un termine in un documento relativamente alla sua presenza globale in tutto il corpus. Tenendo in considerazione solo la frequenza di occorrenza tutti i termini sono considerati ugulamente importanti. Termini che appaiono molte volte in un documento ma meno volte in tutto il corpus potrebbero essere molto più significativi per quel specifico documento e portare molta più informazione, quindi tendono ad essere più importanti. Così come i termini che appaiono molte volte in tutti i dcoumenti del corpus sono spesso poco rilevanti e considerati inutili (stopwords).

$$idf_i = \log \frac{|D|}{|\{d : t_i \in d\}|} \tag{2.8}$$

dove $|D|$ è il numero di documenti nella collezione, mentre il denominatore è il numero di documenti che contengono il termine t_i. Tale funzione aumenta proporzionalmente al numero di volte che il termine è contenuto nel documento, ma cresce in maniera inversamente proporzionale con la frequenza del termine nella collezione. L'idea alla base di questo comportamento è di dare più importanza ai termini che compaiono nel documento, ma che in generale sono poco frequenti.

Altre tecniche utilizzate nel clustering sui documenti sono

LSI il **Latent Semantic Indexing** è un metodo di indicizzazione e reperimento che usa una tecnica matematica chiamata decomposizione a valori singolari (SVD) per identificare pattern nelle relazioni tra i termini e i concetti contenuti in una colezione non strutturata di testo. La LSI è basata sul principio che parole che sono usate nello stesso contesto tendono ad avere significato simile. Chiamata così per la sua abilità di correlare semanticamente termini correlati che sono nascosti (latenti) in grandi collezioni

testuali. La SVD può venire troncata per task di dimensionality reduction, in modo da diminuire la dimensione del vettore mantenendo comunque il significato.

Coseno similarità una euristica per la misurazione della similitudine tra due vettori effettuata calcolando il coseno tra di loro. Dati due vettori di attributi numerici, A e B, il livello di similarità tra di loro è espresso utilizzando la formula

$$similarity = \cos\theta = \frac{A \cdot B}{||A||||B||} \tag{2.9}$$

In base alla definizione del coseno, dati due vettori si otterrà sempre un valore di similitudine compreso tra -1 e $+1$, dove -1 indica una corrispondenza esatta ma opposta (ossia un vettore contiene l'opposto dei valori presenti nell'altro) e $+1$ indica due vettori uguali. Nel caso dell'analisi dei testi, poiché le frequenze dei termini sono sempre valori positivi, si otterranno valori che vanno da 0 a $+1$, dove $+1$ indica che le parole contenute nei due testi sono le stesse (ma non necessariamente nello stesso ordine) e 0 che non c'è nessuna parola che appare in entrambi.

2.3.3 Clustering delle pagine web

In questa sezione si parlerà della soluzione proposta come alternativa alle normali tecniche di clustering delle pagine web basate esclusivamente sul contenuto testuale. Infatti il fulcro del sistema è basato su metodologie nate nel campo del **Natural Language Processing** ma applicate nel contesto web, in modo da aggiungere ulteriore informazione utile. Qui sarà approfondita l'idea alla base del lavoro svolto e le diverse opzioni che l'algoritmo mette a disposizione. L'algoritmo prende in input il grafo di un sito web ele sequenze di random walk generate restituisce rappresentazioni vettoriali per ogni pagina. La fase di sperimentazione vera e propria sarà approfondita nel capitolo successivo.

2.3.4 Algoritmi utilizzati

Vengono riportati di seguito gli algoritmi di clustering testati sul dataset generato. L'obiettivo della tesi comunque non è verificare la validità di questi ma verificare se la soluzione proposta rappresenti un miglioramento ed un possibile alternativa alle soluzioni più usate e consolidate nell'ambito del clustering di pagine web.

- **WalkTrap** È un approccio basato su Random Walk. L'idea generale è che se vengono generati dei Random Walk sul grafo, i percorsi rimarrano probabilmente all'interno della stessa comunità perchè ci sono meno archi che congiungono comunità diverse.
  L'algoritmo esegue piccoli Random Walk (dipendente da un parametro in input) è usa i risultati per fondere comunità diverse in maniera bottom-up. Tagliando il dendrogramma risultante ad una certa altezza è possibile ricevere il numero di cluster desiderati.

- **Fastgreedy** È un approccio gerarchico bottom-up. Cerca di ottimizzare una funzione di modularità in maniera greedy, euristica attuata effettuando la scelta migliore con le informazioni in possesso ad ogni iterazione. Inizialmente ogni nodo è una comunità separata e ricorsivamente si procede ad unire i nodi in modo che la fusione porti al massimo aumento di modularità rispetto al valore corrente.
  L'algoritmo finisce quando non è più possibile aumentare la modularità. È un metodo veloce, solitamente usato come primo approccio perchè non ha parametri da modificare.

- **K-Means** Divide il dataset in un numero prefissato k di cluster. Inizialmente vengono scelti casualmente k punti, non necessariamente facenti parte del dataset, chiamati centroidi. Si procede assegnando ogni data point al centroide più vicino e ricalcolando il centroide sulla media aritmetica dei punti contiene.
  Questo processo di assegnazione dei data point e ricalcolo dei centroidi continua fino a quando non avvengono più assegnazioni. I k cluster risultanti saranno quelli restituiti. Anche questo algoritmo rappresenta

spesso il punto di partenza nell'analisi di un dataset, in quanto molto spesso porta a buoni risultati ma ha come svantaggio il dover sapere a priori il numero di cluster desiderati.

- **DBSCAN** Deriva da *Density-Based Spatial Clustering of Applications with Noise* è un algortimo basato sulla densità, connettendo regioni di punti con densità sufficientemente alta. Fondamentalmente, un punto q è direttamente raggiungibile da un punto p se non viene superata una data distanza ϵ e se p è circondato da un numero sufficiente di punti, allora p e q possono essere considerati parti di un cluster.
  Si può affermare che q è density-reachable da p se c'è una sequenza $p_1, p_2, \ldots, p_n$ di punti con $p_1 = p$ e $p_n = q$ dove ogni p_{i+1} è density-reachable direttamente da p_i. Da notare che la relazione density-reachable non è simmetrica (dato che q potrebbe situarsi su una periferia del cluster, avendo un numero insufficiente di vicini per considerarlo un elemento genuino del cluster). Di conseguenza la nozione density-connected diventa: due punti p e q sono density-connected se c'è un punto o tale che sia o e p che o e q sono density-reachable.
  Un cluster, che è un sotto-insieme dei punti del database, soddisfa due proprietà:

  *i)*Tutti i punti all'interno del cluster sono mutualmente density-connected.
  *ii)*Se un punto è density-connected a un altro punto del cluster, anch'esso è parte del cluster.

- **HDBSCAN** Deriva da *Hierarchical Density-Based Spatial Clustering of Applications with Noise* [3]. Applica DBSCAN variando il valore dell ϵ ed integra i risultati restituendo cluster che stabilizzano meglio tale valore.
  Questo permette ad HDBSCAN di trovare cluster con densità diversa, principale svantaggio di DBSCAN.

2.3.5 NLP nel web: URL embedding

I recenti algoritmi di word embedding, riescono a tenere in considerazione sempre più fattori e quindi restituire vettori sempre più accurati. In particolare vengono incluse informazioni riguardanti il contesto di una parola, ovvero le altre parole che sono contenute all'interno delle frasi in cui compare quella sotto esame. Questa informazione riesce ad estrarre correlazioni nascoste e raggruppare i termini in classi. Può sembrare banale ma diversamente dal text clustering tradizionale, dove le parole vengono considerate al livello di documento, qui vengono considerate a livello di frase. Un esempio può riguardare il controverso caso delle analogie.

È stato osservato che correlazioni nascoste possono trovarsi nella differenza tra coppie di vettori [14], come nel caso di parole simili ma con leggere differenze come il genere o il numero, infatti esse appaiono in frasi tendenzialmente identiche, ma con delle piccole differenze. In questo caso l'informazione può essere interpretata come il genere o il numero.

Informazioni simili possono essere trovate anche nel campo del web, analizzando i percorsi di URL come se fossero frasi. È ormai consolidata la questione dell'autorità di alcune pagine [8] e di come queste abbiano un maggior numero di link in entrata. Alcune pagine tuttavia saranno accessibili prevalentemente attraverso alcuni percorsi e si troveranno quindi in un contesto con elementi simili. Il problema adesso è riuscire a dare un senso a queste correlazioni e dargli un significato utilizzabile. Nello sviluppo di Url2vec tuttavia sono emerse corrispondenze abbastanza naturali, come le pagine dei professori con i relativi corsi insegnati o i laboratori di afferenza. Queste relazioni sono dovute al fatto che il grafo di un sito web è fortemente connesso, ma la maggior parte delle volte in cui appare un determinato corso di studio appare anche il relativo docente e viceversa.

2.3.6 Differenza tra parole ed URL

La sola analisi delle sequenze comunque può risultare limitata. Gli URL non sono parole, in quanto ad essi è associata ulteriore informazione, ovvero il contenuto testuale. Questa proprietà si è rilevata molto utile, infatti combinando le informazioni ricavate esaminando tutti e due gli aspetti, il grado di accuratezza del processo di clustering è salito in modo significativo.

Capitolo 3

Related works

Il clustering di pagine web è un argomento trattato estensivamente in letteratura che ha come obiettivo raggruppare pagine all'interno di cluster omogenei, anche se gran parte del lavoro svolto si basa su un insieme arbitrario di pagine derivanti da molteplici siti differenti. Relativamente poco è stato il lavoro svolto sul clustering di un specifico sito di una determinata organizzazione.

Principalmente si dividono sulla base dei criteri che utilizzano per raggruppare le pagine, ovvero se considerano la semantica, la struttura o l'utilizzo.

Semantica delle pagine web

Come proposto in [5], questo approccio suggerisce l'utilizzo di informazioni semantiche per migliorare il processo di estrazione, necessitano quindi meta-informazioni aggiuntive sulla struttura e sulla gerarchia. Le foglie nel livello più basso sono le pagine web, che sono poi raggruppate in cluster sulla base di una qualche affinità semantica.
Gerarchie semantiche possono essere definite seguendo diversi criteri, che dipendono dagli obiettivi e dalle analisi. Le informazioni aggiuntive necessarie possono essere basate sul contenuto delle pagine web, come ad esempio i me-

tadati, o sapere se è stato un particolare tool o processo nella creazione delle pagine.

Struttura interna delle pagine web

Questi prendono in considerazione la struttura interna di una pagina analizzandone il DOM. La struttura ad albero di una pagine HTML è stata usata per la segmentazione di una pagina ed utilizzare i collegamenti tra queste per raggrupparle [10] o per scovare pattern [9] ricorrenti, utili nel misurare la similarità.

Struttura del grafo del web

Utilizza il grafo del web, dove le pagine sono i nodi e gli archi sono i collegamenti fra queste.
In questo caso il clustering delle pagine web diventa il partizionamento del grafo. Soluzioni proposte [12] si basano sul dividere tale struttura in sottografi, tramite una funzione che minimizza il numero di archi tra cluster e massimizza il numero degli archi tra i nodi di un cluster.

Web usage clustering

Possono essere estratti pattern di utilizzo dai web log ed essere utilizzati per predire il comportamento futuro degli utenti, per raggruppare le pagine in cluster in base agli interessi in comune o per pesare gli archi del grafo, in modo da combinare diversi approcci [18]. Lavori recenti si stanno dirigendo sempre più sullo web usage mining, quindi raggruppando pagine web prevalentemente in base all'utilizzo da parte degli utenti, in modo da personalizzare le risposte alle query immesse nei motori di ricerca [6].

3.1 In-domain clustering

Molto del lavoro svolto sull'argomento si focalizza sul clustering di pagine web provenienti da siti diversi basandosi su di un approccio specifico piuttosto che un altro. Un caso interessante è il sistema SiteMap Generator (SMG) [11] che mette in pratica un approccio ibrido, analizzando sia la struttura interna che quella esterna. Infatti esso divide la pagina in blocchi, li classifica sulla base della loro rilevanza ed infine analizza i collegamenti che ci sono fra blocchi, eventualmente anche di pagine diverse. In seguito i blocchi con alta frequenza di occorrenza e un alto valore hub, ottenuto nell'ultima fase tramite l'algoritmo HITS, vengono utilizzati per generare la sitemap.
In SMG l'obiettivo è dunque la costruzione della sitemap, estraendo dalla struttura delle pagine di uno stesso dominio gli hyperlink necessari.

3.2 Random walk come frasi

Nell'ambito della social networks analysis, DeepWalk [15] propone una metodologia interessante attraverso l'analisi dei grafi. Dato un grafo, vengono generati random walk di piccola lunghezza. Questi vengono poi trattati come frasi, e applicando tecniche di Natural Language Processing viene stimata la verosimiglianza che specifiche sequenze di parole (in questo caso i nodi del grafo) appaiano nel corpus, ovvero l'insieme dei random walk generati. Queste vengono poi mappate in uno spazio vettoriale.
Questo approccio viene applicato nell'ambito dei social network per apprendere rappresentazioni sociali dei vertici. La parte interessante è l'applicazione di tecniche consolidate per la risoluzione di problemi differenti.

3.3 Sviluppi recenti

Articoli recenti si allontanano dalla classica rappresentazione vettoriale dei documenti, basata sula frequenza di occorrenza delle parole all'interno dei

documenti. Questa può evidenziare una similarità tra i documenti non esaustiva. Può rivelarsi utile disambiguare le query immesse nei motori di ricerca o assegnare una misura di rilevanza di un documento ad un dato argomento.

Capitolo 4

Sperimentazione

In questo capitolo si descriverà l'esecuzione della sperimentazione. QUesta si è svolta confrontando i risultati ottenuti attraverso l'applicazione di diversi algoritmi di clustering su dataset ottenuti da differenti rappresentazioni. L'algoritmo è stato testato effettuando il crawling di 3 siti web, che sono stati poi etichettati manualmente per ricavare le metriche necessarie alla valutazione della tesi proposta.

- Il sito web del dipartimento di Computer Science dell'università di Urbana, IL: `cs.illinois.edu`
- Il sito web del dipartimento di Computer Science dell'università di Stanford, CA: `cs.stanford.edu`
- Il sito web del dipartimento di Computer Science del Massachusetts Institute of Technology a Cambridge, MA: `eecs.mit.edu`

Si partirà pertanto dai dati su cui quest'ultima è stata effettuata, proseguendo con la scelta delle modalità di esecuzione più interessanti e concludendo con una serie di tabelle e grafici contenenti i risultati ottenuti.

4.1 Metriche

Valutare le performance di un algoritmo di clustering non è banale come contare il numero di errori o calcolare metriche quali la precision e la recall di un algoritmo di apprendimento supervisionato. In particolare, le metriche di valutazione non dovrebbero prendere in considerazione gli specifici valori delle label. Piuttosto dovrebbero considerare se il raggruppamento generato dall'algortimo definisce una separazione dei dati similmente a quanto fornito nella *ground truth*, ovvero il vero valore delle label, o soddisfare qualche assunzione, come ad esempio che membri dello stesso cluster siano più simili rispetto a quelli di cluster differenti, utilizzando una data funzione di similarità.

- **Homogeneity** Nota la ground truth, questo valore rappresenta quanto ogni cluster restituito dall'apprendimento sia omogeneo, ovvero che contiene solo membri di una classe.

$$h = 1 - \frac{H(C|K)}{H(C)} \tag{4.1}$$

Dove $H(C|K)$ è l'entropia condizionale delle classi date le assegnazioni dei cluster:

$$H(C|K) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{n_{c,k}n}{\cdot}\log\left(\frac{n_{c,k}}{n_k}\right) \tag{4.2}$$

e $H(C)$ è l'entropia delle classi:

$$H(C) = \sum_{c=1}^{|C|}\frac{n_c}{n}\cdot\log\left(\frac{n_c}{n}\right) \tag{4.3}$$

con n il numero totale delle pagine, e n_c ed n_k il numero delle pagine appartenenti alla classe c o al cluster k, ed infine n_c, k le pagine della classe c assegnate al cluster k.

- **Completeness** Note la ground truth, indica se tutti i membri di una sono stati assegnati allo stesso cluster. Da notare che se ad esem-

pio tutte le pagine fossero asegnate ad un unico grande cluster, la *Completeness* sarebbe massima.

$$c = 1 - \frac{H(C|K)}{H(K)} \tag{4.4}$$

Dove $H(C|K)$ è la 4.1 e $H(K)$ è l'entropia dei cluster.

- **V-Measure** rappresenta la media armonica fra l'*homogeneity score* e il *completeness score*.

$$v = 2 \cdot \frac{h \cdot c}{h + c} \tag{4.5}$$

- **Adjusted Rand Index (ARI)** Nota la ground truth, ovvero le classi reali, e le assegnazioni di un algritmo di apprendimento, viene calcolata una funzione che misura la similarità delle due informazioni, ignorandole permutazioni. I valori che può assumere vanno da -1 a 1. Vicino allo 0 rappresentano una assegnazione casuale delle etichette.
  Sia C è gli assegnamenti della ground truth e K le label predette, allora:
  a è il numero di coppie di elementi che si trovano sia in c che in K
  b è il numero di coppie di elementi che si trovano in insiemi diversi in C e in insiemi diversi in K. Il **Random Index** è dato da:

$$RI = \frac{a + b}{C_2^n} \tag{4.6}$$

  Dove C_2^n è il numero totale di tutte le possibili coppie nel dataset.
  Il RI non garantisce comunque che le assegnazioni casuali avranno valori prossimi allo 0.
  Viene definito L'Adjusted Random Index:

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]} \tag{4.7}$$

- **Mutual Information (MI)** Nota la ground truth e le asseganzioni dell'algoritmo di clustering, la *Mutual Information* è una funzione che

misura la corrispondenza delle due informazioni, ignorando le permutazioni.
Anche qui, una assegnazione uniforme (o casuale) dei cluster avrà valori prossimi allo 0.0, evidenziando l'indipendenza delle due informazioni, mentre valori prossimi a 1.0 indicano una corrispondenza significativa.
Dati due assegnamenti di pari lunghezza, U e V, la loro entropia è la quantità di incertezza per una partizione , definita da:

$$H(U) = \sum_{i=1}^{|U|} P(i) \log(P(i)) \tag{4.8}$$

Dove $P(i) = \frac{|U_i|}{N}$ è la probabilità che un oggetto preso a caso da U appartenga alla classe U_i. Similmente per V:

$$H(V) = \sum_{j=1}^{|V|} P'(j) \log(P'(j)) \tag{4.9}$$

Con $P'(j) = \frac{|V_j|}{N}$. La Mutual Information p definita come:

$$MI(U,V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} P(i,j) \log\left(\frac{P(i,j)}{P(i)P'(j)}\right) \tag{4.10}$$

Nelle metriche presentate, fatta eccezione per l'Adjusted Rand Index, 0.0 rappresenta il valore peggiore, mentre 1.0 il perfect score. Questi valori offrono una interpretazione intuitiva e può aiutare alla scoperta degli errori commessi nella assegnazione. Fra i vantaggi è degno di nota che nessuna assunzione viene fatta sulla struttura dei cluster, quindi possono essere utilizzate con algoritmi che identificano cluster di forma diversa.

Silhouette Se la Ground Truth non è nota, può essere usata la Silhouette usando il modello stesso. Ad un alto valore corrispondono modelli con cluster ben definiti. Si compone di due valore

- a : la distanza media tra un vettore e tutti gli altri nella stessa classe

- b : la distanza media tra un vettore e tutti gli altri nella classe più vicina

Il coefficiente di *Silhouette* è dato quindi da:

$$s = \frac{b - a}{\max(a, b)} \tag{4.11}$$

Il valore può variare da -1, per cluster non definiti, a 1 per cluster densamente connessi. Intorno allo 0 indica cluster sovrapposti. Tende comunque ad essere maggiore per cluster convessi, o con alta densità, come quelli ottenuti con DBSCAN.

4.2 Decrizione del dataset

I dataset utilizzati per la sperimentazione sono stati creati eseguendo il processo di crawling e generazione delle sequenze sui seguenti siti:

`http://cs.illinois.edu/`: Il crawling è stato lanciato con profondità massima 10, e immagazzinando un massimo di 10.000 termini per ogni pagina esplorata. Dal grafo sono state poi generate 10.000 sequenze di lunghezza massima 10. Sono state collezionate 728 pagine web.

`http://cs.stanford.edu/`: Il crawling è stato effettuato con profondità massima 10, e immagazzinando un massimo di 10.000 termini per ogni pagina esplorata. Dal grafo sono state poi generate 10.000 sequenze di lunghezza massima 10. Il processo ha collezionato 1458 pagine web.

`http://eecs.mit.edu/`: Il crawling è stato lanciato con profondità massima 10, e immagazzinando un massimo di 10.000 termini per ogni pagina esplorata. Dal grafo sono state poi generate 10.000 sequenze di lunghezza massima 10. Sono state collezionate 1745 pagine web.

4.3 Configurazioni

Dalla scelta di apprendere le rappresentazioni vettoriali delle relazioni invece di utilizzare algoritmi di Community Detection del grafo, sono derivati dei vantaggi. Innanzitutto gli algoritmi di partizionamento dei grafi hanno complessità NP-completa, ovvero necessitano di un tempo superpolinomiale nella dimensione dell'input. Nel contesto del Web Mining la dimensione del dataset può crescere enormemente ed avere soluzioni più efficienti costituisce senz'altro una priorità.
In tutti i casi di seguito riportati sono stati generati grafi sia in modalità classica, che attraverso l'estrazione delle liste. Nel primo caso i dataset saranno chiamati **"nc"** (no-costraint, senza vincoli), mentre nel secondo caso **"lc"** (list-costraint, con il vincolo delle liste. Dove omessa, la configurazione è rimasta invariata.

4.3.1 Community Detection

Sono state applicate metodologie derivanti dalla teoria dei grafi per l'estrazione di strutture connesse all'interno del grafo web. Questo può essere utile nella individuazione di community all'interno di grafi come ad esempio social network. La divisione del grafo in community può essere effettuata seguendo diversi approcci, inoltre la rete costruita su di un tipico sito web è caratterizzata solitamente da un numero elevato di collegamenti tra pagine che suggeriscono l'utilizzo di alcune tipologie piuttosto che altre. Infatti misure come la betweenness non hanno restituito risultati significativi. Sono stati comunque effettuati test utilizzando la rappresentazione a grafo per confrontare al meglio i risultati globali ottenuti.

I grafi utilizzati rappresentano le due diverse operazioni di estrazione di collegamenti effettuate. La prima contiene tutti i collegamenti presenti all'interno di una pagina e mostrerà quindi più archi. La seconda estrae solamente gli hyperlink dalle liste.

L'analisi del grafo considera unicamente le relazioni che intercorrono fra le pagine web e tralascia informazioni riguardanti il contenuto. Dalle metriche rilevate risulta in tabella 4.1, risulta che il partizionamento del grafo web non riesce a dividere al meglio i cluster.

`cs.illinois.edu` Il grafo del sito presenta 728 nodi e 16993 archi. L'algoritmo *Fastgreedy* non richiede parametri specifici ma è stato tagliato il dendrogramma ad altezza 15, il numero di cluster nella operazione di raggruppamento manuale. Ugualmente per *WalkTrap*, che però necessitava della lunghezza dei Random Walk da effettuare. I risultati migliori sono stati osservati con percorsi di lunghezza 3.

G - Illinois	**Homog**	**Compl**	**V-Measure**	**ARI**	**MI**
nc WalkTrap	0.6471	0.6585	0.6527	0.4363	0.6281
nc Fastgreedy	0.5518	0.8563	0.6711	0.5764	0.5354
lc WalkTrap	0.5093	0.4892	0.4991	0.2762	0.4722
lc Fastgreedy	0.5522	0.6035	0.5767	0.3656	0.5382

Tabella 4.1: Risultati sperimentazione di partizionamento del grafo del sito `cs.illinois.edu`

`cs.stanford.edu` Il grafo del sito presenta 1458 nodi e 99686 archi. Il dendrogramma restituito dall'algoritmo *Fastgreedy* è stato troncato ad altezza 10. Anche qui la scelta è ricaduta su Random Wlak di lunghezza 3 per l'algoritmo *WalkTrap*.

G - Stanford	Homog	Compl	V-Measure	ARI	MI
nc WalkTrap	0.1628	0.3967	0.2308	0.1127	0.1340
nc Fastgreedy	0.3568	0.3894	0.3724	0.2439	0.3357
lc WalkTrap	0.3087	0.6314	0.4146	0.0502	0.1911
lc Fastgreedy	0.4854	0.6740	0.5644	0.2999	0.3892

Tabella 4.2: Risultati sperimentazione di partizionamento del grafo del sito `cs.stanford.edu`

`eecs.mit.edu` Il grafo del sito presenta 1745 nodi e 63937 archi. L'algoritmo *Fastgreedy* ha costruito un dendrogramma che è stato tagliato ad altezza 15, il numero di cluster nella operazione di raggruppamento manuale. Ugualmente per *WalkTrap*, che però necessitava della lunghezza dei Random Walk da effettuare. I risultati migliori sono stati osservati con percorsi di lunghezza 3.

G - MIT	Homog	Compl	V-Measure	ARI	MI
nc WalkTrap	0.1792	0.6729	0.2830	0.1762	0.1648
nc Fastgreedy	0.5769	0.5215	0.5478	0.5514	0.5119
lc WalkTrap	0.0856	0.5013	0.1462	0.0450	0.0664
lc Fastgreedy	0.5617	0.7127	0.6282	0.6851	0.5497

Tabella 4.3: Risultati sperimentazione di partizionamento del grafo del sito `eecs.mit.edu`

4.3.2 URL Embedding

Considerando i Random Walk generati sul grafo come frasi, è possibile applicare algoritmi di Word Embedding per raggruppare le pagine sulla base

del contesto in cui appaiono, ovvero le pagine che più verosimilmente appariranno insieme nelle sequenze.
Le squenze di Random Walk sono state usate per apprendere rappresentazioni vettoriali delle pagine Web. La fase di URL embedding è stata effetuata utilizzando l'algoritmo Word2vec, [17] (esaminato in 1.3.1) modificando alcuni parametri e lasciando invariato altri.
I parametri personalizzati sono:

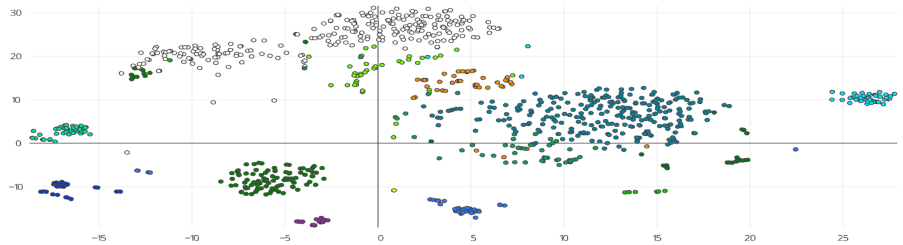

Figura 4.1: Rappresentazione del sito `cs.illinois.edu`, clusterizzato con K-Means.

- **min-count**: tutte le parole (o URL) con frequenza di occorrenza minore i questo valore vengono ignorate.
- **window** rappresenta la distanza massima tra l'URL corrente e quello predetto all'interno di una frase.
- **negative** Nella fase di embedding di una URL, viene calcolato il rapporto tra la similarità del contesto con la parola e la sommatoria di tutte le similarità tra la parola e gli altri contesti. Più precisamente:

$$\frac{v_c \cdot v_w}{\sum_{c \in C} v_c \cdot v_w} \tag{4.12}$$

  Questa operazione può essere molto lenta. Per accelerare il processo possono venir scelti n contesti casuali da confrontare. Questo parametro, se maggiore di 0, rappresenta il numero di vettori da confrontare.

- **sg** definisce l'algoritmo di apprendimento, di default viene usato *CBOW*, mentre se impostato a 1 utilizza *skip-gram* [14]

I migliori risultati sono stati osservati con:

`cs.illinois.edu` è stata impostato un *min-count* pari a 1, *window* con valore 5 ed è stato utilizzato *skip-gram* con 5 *negative* sampling. Gli algoritmi testati sono stati: DBSCAN con $\epsilon = 0.9$ ed *min-samples*= 4; HDBSCAN con *min-cluster-size*= 6; K-Means con numero di cluster pari a 15.

E - Illinois	**Homog**	**Compl**	**V-Meas**	**ARI**	**MI**	**Silh**
nc dbscan	0.5553	0.6579	0.6023	0.4487	0.5234	0.2588
nc hdbscan	0.5759	0.6720	0.6203	0.5282	0.5525	0.2573
nc Kmeans	0.8238	0.7575	0.7892	0.7883	0.7423	0.3131
lc dbscan	0.4163	0.5922	0.4889	0.2250	0.3935	0.1320
lc hdbscan	0.4760	0.5067	0.4908	0.2275	0.4515	0.1054
lc Kmeans	0.8095	0.6593	0.7267	0.6189	0.6473	0.2281

Tabella 4.4: Risultati sperimentazione di URL embedding del sito `cs.illinois.edu`

`cs.stanford.edu` è stata impostato un *min-count* pari a 1, *window* con valore 5 ed è stato utilizzato *skip-gram* con 5 *negative* sampling. Gli algoritmi testati sono stati: DBSCAN con $\epsilon = 0.9$ ed *min-samples*= 7; HDBSCAN con *min-cluster-size*= 10; K-Means con numero di cluster pari a 15. Per il dataset delle liste: DBSCAN con $\epsilon = 1.6$ ed *min-samples*= 3; HDBSCAN con *min-cluster-size*= 5; K-Means con numero di cluster pari a 15.

E - Stanford	Homog	Compl	V-Meas	ARI	MI	Silh
nc dbscan	0.3453	0.4031	0.3720	0.2577	0.3252	0.4058
nc hdbscan	0.4720	0.4418	0.4564	0.3094	0.4210	0.4030
nc Kmeans	0.3810	0.3844	0.3827	0.2276	0.3574	0.5659
lc dbscan	0.4830	0.5798	0.5270	0.0976	0.3363	0.0938
lc hdbscan	0.6506	0.6299	0.6353	0.2466	0.5014	0.1762
lc Kmeans	0.5853	0.6838	0.6308	0.2423	0.4786	0.2513

Tabella 4.5: Risultati sperimentazione di URL embedding del sito `cs.stanford.edu`

`eecs.mit.edu` è stata impostato un *min-count* pari a 1, *window* con valore 5 ed è stato utilizzato *skip-gram* con 5 *negative* sampling. Gli algoritmi testati sono stati: DBSCAN con $\epsilon = 1.0$ ed *min-samples*= 10; HDBSCAN con *min-cluster-size*= 10; K-Means con numero di cluster pari a 10. Per il dataset delle liste: DBSCAN con $\epsilon = 1.5$ ed *min-samples*= 5; HDBSCAN con *min-cluster-size*= 13; K-Means con numero di cluster pari a 10.

E - MIT	Homog	Compl	V-Meas	ARI	MI	Silh
nc dbscan	0.4097	0.4236	0.4164	0.3779	0.3869	0.3264
nc hdbscan	0.5834	0.4413	0.5025	0.5811	0.4231	0.3141
nc Kmeans	0.6422	0.5129	0.5703	0.5141	0.5011	0.3914
lc dbscan	0.1727	0.5736	0.2655	0.1455	0.1550	0.1750
lc hdbscan	0.7882	0.4456	0.5693	0.5217	0.4061	0.2288
lc Kmeans	0.7165	0.5283	0.6081	0.3625	0.5146	0.2646

Tabella 4.6: Risultati sperimentazione di URL embedding del sito `eecs.mit.edu`

4.3.3 Text Mining

Sono state utilizzate tecniche di Text Mining per il clustering basato sul contenuto testuale. I contenuti all'interno di uno stesso sito web avranno una struttura e termini comuni, differenziandosi al variare dell'argomento trattato. La struttura gerarchica di un sito web organizza solitamente le pagine in sezioni simili. Questa metodologia tuttavia, considera solo l'informazione testuale, assumendo che i termini all'interno del sito web siano indipendenti l'uno dall'altro così come i documenti, ignorando le relazioni interdipendenti tra questi. Il web si discosta dall'analisi classica dei documenti proprio per le relazioni che intercorrono tra le pagine, tuttavia l'analisi testuale rimane molto importante.
Nella fase di sperimentazione è stata utilizzata una rappresentazione vettoriale della frequenza dei termini all'interno dell'insieme delle pagine web, calcolata con la tecnica della *frequency–inverse document frequency* (tf-idf).

I parametri personalizzati per la costruzione dell matrice documenti-termini con funzione di peso *idf* sono stati:

- **max-df**: questo valore rappresenta la massima frequenza, all'interno dei documenti, che un termine può avere per essere utilizzato nella matrice tf-idf. Se un termine appare molte volte nel corpus, molto probabilmente avrà poco significato.
- **min-df**: indica il numero minimo di documenti in cui un termine dovrà apparire per essere considerato.
- **ngram-range**: vengono presi in considerazioni gli n-grammi di lunghezza compresa nell'intervallo specificato in questo parametro. Un n-gramma è una sottosequenza di n elementi di un'altra.

I risultati mostrati sono stati ottenuti nel seguente modo:

`cs.illinois.edu` Sul dataset del sito, costituito da 728 pagine e 433 termini, il corpus è stato ripulito delle stopword, stemmatizzato ed è stato impostato il *max-df* all'80%, il *min-df* a 0.1 e sono stati considerati solo uni-grammi, bi-grammi e tri-grammi. Se i termini appaiono in più dell'80% dei documenti, probabilmente avrà poco significato, lo stesso se appare troppe poche volte. Gli algoritmi testati sono stati: DBSCAN con $\epsilon = 0.9$ ed *min-samples*= 4; HDBSCAN con *min-cluster-size*= 4; K-Means con numero di cluster pari a 15.
Sul Dataset costruito estraendo le liste: DBSCAN con $\epsilon = 0.7$ ed *min-samples*= 4; HDBSCAN con *min-cluster-size*= 7; K-Means con numero di cluster pari a 15.

T - Illinois	Homog	Compl	V-Meas	ARI	MI	Silh
nc dbscan	0.5601	0.5962	0.5776	0.4078	0.5346	0.1242
nc hdbscan	0.5152	0.6029	0.5556	0.3862	0.4858	0.0881
nc Kmeans	0.7619	0.5814	0.6596	0.3184	0.5586	0.1767
lc dbscan	0.5710	0.7042	0.6306	0.5197	0.5566	0.1406
lc hdbscan	0.4501	0.5104	0.4783	0.1938	0.4297	0.1018
lc Kmeans	0.8061	0.5892	0.6808	0.4296	0.5748	0.2002

Tabella 4.7: Risultati sperimentazione di Text Mining del sito `cs.illinois.edu`

`cs.stanford.edu` Sul dataset del sito, costituito da 1458 pagine e 843 termini, il corpus è stato ripulito delle stopword, stemmatizzato ed è stato impostato il *max-df* all'80%, il *min-df* a 0.1 e sono stati considerati solo unigrammi, bi-grammi e tri-grammi. Se i termini appaiono in più dell'80% dei documenti, probabilmente avrà poco significato, lo stesso se appare troppe poche volte. Gli algoritmi testati sono stati: DBSCAN con $\epsilon = 0.3$ ed *min-samples*= 5; HDBSCAN con *min-cluster-size*= 15; K-Means con numero di cluster pari a 15.
Sul Dataset costruito estraendo le liste: DBSCAN con $\epsilon = 0.5$ ed *min-samples*= 3; HDBSCAN con *min-cluster-size*= 5; K-Means con numero di cluster pari a 15.

T - Stanford	Homog	Compl	V-Meas	ARI	MI	Silh
nc dbscan	0.2584	0.3435	0.2949	0.0208	0.2403	0.0981
nc hdbscan	0.3015	0.2784	0.2895	0.0089	0.2521	0.0916
nc Kmeans	0.6014	0.4652	0.5246	0.2639	0.4489	0.2704
lc dbscan	0.0876	0.3924	0.1432	0.0102	0.0327	-0.1188
lc hdbscan	0.1827	0.3802	0.2468	0.0687	0.1326	0.0841
lc Kmeans	0.5436	0.5726	0.5577	0.2167	0.4157	0.1732

Tabella 4.8: Risultati sperimentazione di Text Mining del sito `cs.stanford.edu`

`eecs.mit.edu` Sul dataset del sito, costituito da 1745 pagine e 354 termini, il corpus è stato ripulito delle stopword, stemmatizzato ed è stato impostato il *max-df* all'80%, il *min-df* a 0.1 e sono stati considerati solo uni-grammi, bi-grammi e tri-grammi. Se i termini appaiono in più dell'80% dei documenti, probabilmente avrà poco significato, lo stesso se appare troppe poche volte. Gli algoritmi testati sono stati: DBSCAN con $\epsilon = 0.9$ ed *min-samples*= 9; HDBSCAN con *min-cluster-size*= 15; K-Means con numero di cluster pari a 10.
Sul Dataset costruito estraendo le liste: DBSCAN con $\epsilon = 0.9$ ed *min-samples*= 6; HDBSCAN con *min-cluster-size*= 9; K-Means con numero di cluster pari a 10.

T - MIT	Homog	Compl	V-Meas	ARI	MI	Silh
nc dbscan	0.2071	0.1908	0.1986	0.1011	0.1759	0.1133
nc hdbscan	0.2363	0.2679	0.2611	0.0918	0.2207	0.0845
nc Kmeans	0.4175	0.2381	0.3032	0.0917	0.2290	0.1958
lc dbscan	0.3470	0.3466	0.3467	0.2071	0.3326	0.1180
lc hdbscan	0.4402	0.4147	0.4271	0.2806	0.3995	0.1423
lc Kmeans	0.5857	0.4198	0.4890	0.2597	0.4085	0.2296

Tabella 4.9: Risultati sperimentazione di Text Mining del sito `eecs.mit.edu`

4.3.4 Embedding e Text Mining

I risultati hanno evidenziato che l'analisi singola, sia della correlazione tra le pagine sia del contenuto testuale, può non bastare a codificare esaustivamente la conoscenza che una pagina web può offrire. Entrambe le informazioni sono rilevanti ed andrebbero processate combinatamente. Effettuando i test precedenti è stato osservato come le informazioni codificate nelle due tipologie di vettori fossero complementari. Sono stati quindi considerati come un unico vettore. Il vantaggio di associare le relazioni in uno spazio vettoriale offre il vantaggio usare la stessa rappresentazione e quindi di unire i vettori derivanti dagli algoritmi di word embedding con quelli derivanti dall'analisi di contenuto testuale.
Così facendo è possibile dare più importanza ad una tipologia di informazione piuttosto che ad un altra, andando a modificare il rapporto tra le dimensioni dei vettori.

`cs.illinois.edu` I vettori di word2vec sono stati generati di dimensione 48, mentre i vettori-riga documenti sono stati ridotti con *TruncateSVD* a

dimensione 50. Gli algoritmi testati sono stati: HDBSCAN con *min-cluster-size*= 7; K-Means con numero di cluster pari a 15.

ET - Illinois	**Homog**	**Compl**	**V-Meas**	**ARI**	**MI**	**Silh**
nc hdbscan	0.7327	0.7534	0.7429	0.7204	0.7186	0.2070
nc Kmeans	0.8812	0.8069	0.8424	0.8299	0.7949	0.3198
lc hdbscan	0.6541	0.6129	0.6328	0.3249	0.5992	0.1203
lc Kmeans	0.8548	0.6885	0.7627	0.6488	0.6773	0.2573

Tabella 4.10: Risultati sperimentazione di Text-Embedding del sito `cs.illinois.edu`

`cs.stanford.edu` I vettori di word2vec sono stati generati di dimensione 48, mentre i vettori-riga documenti sono stati ridotti con *TruncateSVD* a dimensione 50. Gli algoritmi testati sono stati: HDBSCAN con *min-cluster-size*= 10; K-Means con numero di cluster pari a 15.

ET - Stanford	**Homog**	**Compl**	**V-Meas**	**ARI**	**MI**	**Silh**
nc hdbscan	0.3495	0.5272	0.4203	0.3335	0.3307	0.3469
nc Kmeans	0.3490	0.3629	0.3558	0.2061	0.3238	0.4336
lc hdbscan	0.2343	0.7045	0.3517	0.1012	0.2033	0.2342
lc Kmeans	0.7508	0.7422	0.7465	0.5123	0.6656	0.2375

Tabella 4.11: Risultati sperimentazione di Text-Embedding del sito `cs.stanford.edu`

`eecs.mit.edu` I vettori di word2vec sono stati generati di dimensione 48, mentre i vettori-riga documenti sono stati ridotti con *TruncateSVD* a dimensione 50. Gli algoritmi testati sono stati: HDBSCAN con *min-cluster-size*= 14; K-Means con numero di cluster pari a 10.

ET - MIT	Homog	Compl	V-Meas	ARI	MI	Silh
nc hdbscan	0.5573	0.3748	0.4482	0.2845	0.3600	0.1607
nc Kmeans	0.5505	0.4344	0.4856	0.2687	0.4252	0.1699
lc hdbscan	0.6076	0.4329	0.5056	0.1910	0.4181	0.1464
lc Kmeans	0.6679	0.5332	0.5930	0.3499	0.5235	0.2311

Tabella 4.12: Risultati sperimentazione di Text-Embedding del sito `eecs.mit.edu`

4.3.5 Analisi dei risultati

Il processo di crawling è stato svolto su 3 siti, sui quali sono state generate sequenze di 2 tipi diversi, ossia Random Walk senza vincoli e Random Walk con i vincoli delle liste. Su di esse sono state poi effettuate 4 metodologie diverse: grafo, Word Embedding, Text Mining e Word Embedding & Text Mining. Per ogni metodologia, inoltre, sono stati usati 3 algoritmi di apprendimento diversi, ovvero K-Means, HDBSCAN e DBSCAN. Infine, su ogni algoritmo sono state ricavate 6 metriche. In tutto sono 432 valori.

Di seguito sono riportati i risultati più interessanti per una migliore comprensione.

K-Means nc - cs.illinois.edu

Type	Homog	Compl	V-Meas	ARI	MI	Silh
Embedding	0.5553	0.6579	0.6023	0.4487	0.5234	0.2588
Text Mining	0.5759	0.6720	0.6203	0.5282	0.5525	0.2573
Emb + Text	0.8238	0.7575	0.7892	0.7883	0.7423	0.3131

Tabella 4.13: Risultati sperimentazione raggruppati per algoritmo K-Means sul sito `cs.illinois.edu`

In questo caso la combinazione delle due informazioni ha portato un miglioramento complessivo del clustering. Nel terzo caso si è verificato un miglioramento in media di 22.0% rispetto al primo, e di 18.8% rispetto al secondo.

K-Means lc - cs.stanford.edu

Type	Homog	Compl	V-Meas	ARI	MI	Silh
Embedding	0.5853	0.6838	0.6308	0.2423	0.4786	0.2513
Text Mining	0.5857	0.4198	0.4890	0.2597	0.4085	0.2296
Emb + Text	0.7508	0.7422	0.7465	0.5123	0.6656	0.2375

Tabella 4.14: Risultati sperimentazione raggruppati per algoritmo K-Means sul sito `ecs.mit.edu`

Anche in questo caso l'utilizzo combinato di Text Mining e Word Embedding ha portato miglioramento. In media di 16% rispetto all'Embedding e di 21.2% rispetto al Text Mining.

K-Means lc - eecs.mit.edu

Type	Homog	Compl	V-Meas	ARI	MI	Silh
Embedding	0.7165	0.5283	0.6081	0.3625	0.5146	0.2646
Text Mining	0.5436	0.5726	0.5577	0.2167	0.4157	0.1732
Emb + Text	0.6679	0.5332	0.5930	0.3499	0.5235	0.2311

Tabella 4.15: Risultati sperimentazione raggruppati per algoritmo K-Means sul sito `ecs.mit.edu`

Qui, invece, è risultata migliore l'applicazione dell'embedding, con un miglioramento di 1.2% rispetto all Word Embedding & Text Mining, mentre di 8.4% rispetto al solo Text Mining.

Lo scopo della tesi non era valutare l'efficacia dei vari algoritmi riportati, ma verificare un eventuale miglioramento nei risultati ottenuti.
Dalle metriche rilevate è emerso che l'uso delle liste non ha influenzato particolarmente i risultati ottenuti. Valutare i risultati di un algoritmo di clustering non è un operazione semplice. Infatti l'assegnazione manuale delle etichette denota una certa arbitrarietà.
Inoltre, l'analisi dei percorsi ha fatto notare come certe classi, idealmente raggruppate insieme in quanto stessa entità (e.g. docenti), possano invece essere divise in fase di apprendimento per motivi ragionevoli. Ad esempio, nel sito *cs.illinois.edu*, erano presenti molteplici pagine relative agli stessi professori. Questo era dovuto al fatto che durante gli anni erano state pubblicate diverse edizioni del sito. Questo ha portato le diverse versioni della stessa pagina (concettuale) ad essere presenti con diversa frequenza in percorsi diversi. O ancora ad avere anche testo differente. Infatti anche l'analisi testuale fra le diverse versioni era differente.

Un altro esempio era il raggruppamento di pagine con poco testo oppure, diverse pagine relative agli studenti "undergraduates" etichettate in classi diverse, sono state raggruppate nello stesso cluster, probabilmente dovuto al fatto che apparivano in percorsi simili ed avevano testi simili.

In conclusione il problema del clustering di pagine web può rivelarsi ostico e dare risultati diversi da quelli desiderati ma comunque sensati. Considerare più aspetti può essere rivelarsi utile in molti contesti applicativi.

Capitolo 5

Conclusioni e sviluppi futuri

In questo lavoro di tesi è stata trattata la problematica della sicurezza, evidenziando l'importanza che essa riveste dal punto di vista sociale e le difficoltà sociali e tecniche incontrate nella realizzazione di alcuni progetti. La trattazione effettuata, incentrata sull'estrazione di entità nominali da documenti testuali e sulla predizione ed evoluzione delle categorie criminali nel tempo, non pretende di essere esaustiva, ma piuttosto un punto di partenza per ulteriori sviluppi, sia teorici che sperimentali.

Il sistema TB-CREDIS è stato esteso per permettere l'estrazione dei perpetratori di un crimine all'interno di documenti più o meno segretati, che possono spaziare da articoli giornalistici a report investigativi, e predire le evoluzioni dei criminali, relativamente alle loro attività, in quegli intervalli temporali per cui non si possiede alcun documento associato al soggetto in esame relativo a tal periodo. L'alta modularità del sistema ha permesso inoltre la facile integrazione della componente di interfaccia utente: l' utente può caricare un nuovo criminale nella collezione di dati, effettuare le operazioni di predizione e visualizzare graficamente i risultati prodotti e le evoluzioni del criminale sotto analisi nel tempo.

I risultati sperimentali prodotti si sono rivelati all'altezza delle aspettative e incentivano a proseguire gli studi in questa direzione in modo da individuare

nuove tecniche che permettano di migliorare i risultati raggiunti in termini di tempi di elaborazione e di qualità. In particolare la componente di estrazione delle entità perpetratrici può essere estesa con nuove euristiche per permettere di estrarre le entità nominali da quei pattern ad oggi non considerati, o estrarre nuove entità quali nomi di possibili vittime, luoghi, proprietà private e tutto ciò che possa rivelarsi utile alle attività investigative. Per la componente di predizione possibili sviluppi futuri potrebbero riguardate l'analisi e individuazione dei documenti relativi allo stesso reato, commesso da un criminale, in modo da evitare che reati più "documentati" abbiano più importanza di altri reati commessi nel calcolo della posizione semantica del criminale stesso.

Bibliografia

[1] David Aldous and James Allen Fill. Reversible markov chains and random walks on graphs, 2002. Unfinished monograph, recompiled 2014, available at `http://www.stat.berkeley.edu/$\sim$aldous/RWG/book.html`.

[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003.

[3] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Trans. Knowl. Discov. Data*, 10(1):5:1–5:51, July 2015.

[4] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November 2011.

[5] Robert Cooley. The use of web structure and content to identify subjectively interesting web usage patterns. *ACM Trans. Internet Technol.*, 3(2):93–116, May 2003.

[6] Daniel Crabtree, Peter Andreae, and Xiaoying Gao. Query directed web page clustering. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '06, pages 202–210, Washington, DC, USA, 2006. IEEE Computer Society.

[7] Yeye He, Dong Xin, Venkatesh Ganti, Sriram Rajaraman, and Nirav Shah. Crawling deep web entity pages. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 355–364, New York, NY, USA, 2013. ACM.

[8] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999.

[9] Miloš Kudělka, Václav Snášel, Ondřej Lehečka, Eyas El-Qawasmeh, and Jaroslav Pokorný. Web pages reordering and clustering based on web patterns. In *Proceedings of the 34th Conference on Current Trends in Theory and Practice of Computer Science*, SOFSEM'08, pages 731–742, Berlin, Heidelberg, 2008. Springer-Verlag.

[10] Cindy Xide Lin, Yintao Yu, Jiawei Han, and Bing Liu. Hierarchical web-page clustering via in-page and cross-page link structures. In *Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part II*, PAKDD'10, pages 222–229, Berlin, Heidelberg, 2010. Springer-Verlag.

[11] Shian-Hua Lin, Kuan-Pak Chu, and Chun-Ming Chiu. Automatic sitemaps generation: Exploring website structures using block extraction and hyperlink analysis. *Expert Syst. Appl.*, 38(4):3944–3958, April 2011.

[12] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, December 2007.

[13] MacQueen. Some methods for classification and analysis of multivariate observation. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*. University of California Press, 1967.

[14] Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*

(NAACL-HLT-2013). Association for Computational Linguistics, May 2013.

[15] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA, 2014. ACM.

[16] Anand Rajaraman and Jeffrey David Ullman. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA, 2011.

[17] Radim Řehůřek and Petr Sojka. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[18] C. Shahabi, A. M. Zarkesh, J. Adibi, and V. Shah. Knowledge discovery from users web-page navigation. In *Proceedings of the 7th International Workshop on Research Issues in Data Engineering (RIDE '97) High Performance Database Management for Large-Scale Applications*, RIDE '97, pages 20–, Washington, DC, USA, 1997. IEEE Computer Society.

[19] R.C. Tryon and D.E. Bailey. *Cluster Analysis;corelation Profile an Orthometric (factor) Analysis for the Isolation of Unities in Mind and Personality, by Robert Choate Tryon ... Ann Arbor, Mich., Edwards Brothers, Inc., Lithoprinters and Publishers, 1939*. McGraw-Hill, 1970.

[20] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[21] Will Y. Zou, Richard Socher, Daniel M. Cer, and Christopher D. Manning. Bilingual word embeddings for phrase-based machine translation. In *EMNLP*, pages 1393–1398. ACL, 2013.