POLITECNICO DI MILANO
Corso di Laurea Magistrale in Ingegneria Informatica
Dipartimento di Elettronica e Informazione

Utilizzo di strumenti semantici per l'individuazione di entità contestuali

Relatore: Prof. Paolo Cremonesi
Correlatore: Dott. Roberto Turrin

Tesi di Laurea di:
Stefano Camocini, matricola 735669

Anno Accademico 2012-2013

Sommario

L'utilizzo di informazioni contestuali nei sistemi di raccomandazione è sempre più riconosciuto come un fattore chiave; ad esempio, il fatto che un utente guardi la televisione da solo oppure con degli amici condiziona la scelta del film da acquistare. Il sistema di raccomandazione, oltre a consigliare l'utente in base alle sue preferenze, deve anche essere in grado di identificare il contesto.
Sono state proposte molteplici definizioni di contesto; secondo la nostra visione, *il contesto di un utente rappresenta tutte le informazioni relative ad esso e all'ambiente che lo circonda che sono limitate ad una determinata "sessione"*. Esempi di contesto possono essere: il giorno, l'ora, le condizioni meteorologiche, il luogo, l'attività che l'utente sta svolgendo.
A partire da questa definizione, i principali contributi di questo lavoro di tesi sono stati: (i) l'analisi e la formalizzazione del contesto, prima in generale poi con particolare riferimento al dominio sistemi di video on demand; (ii) lo studio di un particolare contesto definito dalle notizie che l'utente sta leggendo. Immaginando che l'utente stia leggendo delle notizie online, occorre identificare il contesto (ad esempio il soggetto delle notizie) e rappresentarlo. Abbiamo rappresentato il contesto come un insieme di entità; per estrarre le entità sono stati utilizzati strumenti per l'analisi semantica (Stanbol). L'efficacia di questi strumenti semantici è stata valutata con classici test di qualità applicati ad algoritmi content-based.
Dai risultati è emerso che le parole estratte filtrano molte entità non significative, permettendo di identificare in maniera più precisa il contesto.

Ringraziamenti

In primis, non me ne vogliano parenti e amici, voglio ringraziare Roberto, per avermi messo subito a mio agio dal nostro primo incontro e per avermi sopportato durante tutto il periodo di tesi. Grazie di cuore.
Inoltre, ringrazio il Professor Cremonesi che, con i suoi suggerimenti, ha permesso il completamento del lavoro di tesi.
Voglio ringraziare Claudia, che potrei definire il mio faro nella notte, colei che è stata sempre disponibile ad ascoltarmi, ad aiutarmi e ad avere sempre una parola di conforto per farmi forza tra tutte le difficoltà.
Non posso non ringraziare i miei SuperAmici: Mirko, Anto, Cesare, Ale, Davide e Marco e relative compagne, senza di loro le serate dell'ultimo anno sarebbe state davvero tristi.
Per ultimi, ma sempre presenti nei miei pensieri, un sentito ringraziamento ai miei genitori, che, nonostante le mille difficoltà, con il loro incrollabile sostegno morale ed economico, mi hanno permesso di raggiungere questo traguardo.

Indice

Capitolo 1

Introduzione

I *sistemi di raccomandazione* sono degli strumenti informatici in grado di predire le preferenze e i gusti di un utente.
La raccomandazione si applica in molteplici processi decisionali: da quali oggetti comprare a quale musica ascoltare oppure quali notizie leggere. *Item* è il termine generale usato per denotare cosa il sistema raccomandi agli utenti (es., film, libri, vacanze).
Un sistema di raccomandazione normalmente è pensato per un tipo specifico di item, per esempio film, CD o notizie. Il design, l'interfaccia grafica e la tecnica di raccomandazione sono specificatamente tarate in modo tale da fornire suggerimenti utili ed efficaci in base al dominio di lavoro.

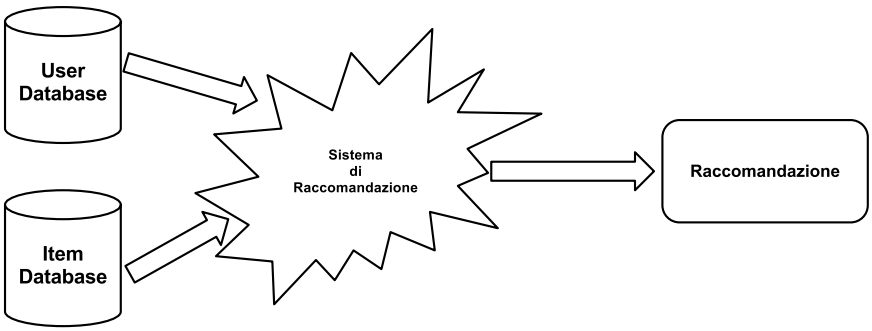

Figura 1.1: Sistema di raccomandazione standard

Come possiamo vedere in Figura 1.1, alla base di ogni sistema di raccomandazione vi è un insieme di informazioni relative ad utenti e item, che elaborate dagli algoritmi di raccomandazione permettono di desumere le preferenze degli utenti verso oggetti specifici. Queste preferenze sono solitamente identificate come *rating*. Tali rating possono essere esplicitamente

espressi (ad esempio il giudizio espresso da un utente per un per un film in una scala da uno a cinque), o dedotti implicitamente dalle interazioni di un utente con uno specifico oggetto (ad esempio la visione di un film).
Le raccomandazioni finora presentate non tengono in considerazione in nessun modo dell'ambiente contestuale in cui vengono effettuate (es., il luogo, il tempo, la presenza di altre persone ecc...). Per esempio, utilizzando il contesto temporale ci si aspetta che, una raccomandazione riguardante un viaggio vacanza produca risultati differenti nel caso si tratti di periodo estivo o di periodo invernale. Quindi, in molti casi, non è sufficiente considerare solamente gli utenti e le loro preferenze, ma è altrettanto importante incorporare informazioni contestuali durante il processo di raccomandazione, in maniera tale da fornire item raccomandabili diversi da circostanza a circostanza.

In Figura 1.2, possiamo vedere come le tre entità, user, item e contesto,

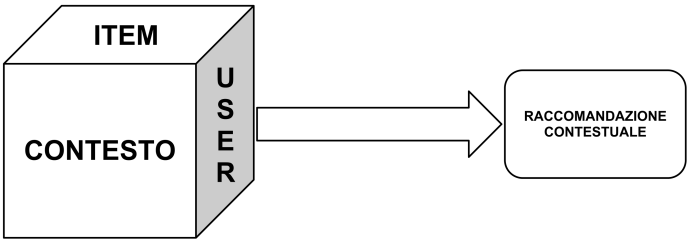

Figura 1.2: Sistema di raccomandazione contestuale

vengano date in input al sistema di raccomandazione, il quale predirrà rating differenti da contesto a contesto; la scelta di un film da vedere con la propria famiglia probabilmente differirà dal film da vedere da solo o con la propria fidanzata.
Da questi esempi, risulta evidente come il contesto abbia un forte impatto nei processi decisionali degli utenti e di conseguenza nelle tecniche di marketing aziendali. Naturalmente, l'efficienza di questi metodi è dovuta allo studio delle variabili contestuali più adatte per lo sviluppo della raccomandazione. Ad esempio, il tempo atmosferico potrebbe non influenzare la scelta di quale film vedere al cinema, al contrario la compagnia (fidanzata, famiglia o amici) avrà un'influenza consistente su tale scelta.
Sono state proposte molteplici definizioni di contesto in letteratura; secondo la nostra interpretazione, il contesto di un utente rappresenta tutte quelle informazioni dell'utente e dell'ambiente attorno a lui che sono limitate ad una determinata sessione, ovvero ad un periodo di tempo abbastanza breve in cui viene effettuata la raccomandazione.
A partire da questa definizione, i principali contributi della tesi sono stati:

(i) la formalizzazione e analisi del contesto, con particolare riferimento al dominio dei sistemi di video on demand, e (ii) lo studio di uno di questi contesti, con l'implementazione di un framework basato su strumenti semantici per rappresentarlo.
Quindi, dopo aver valutato l'osservabilità e l'utilità dei contesti applicati al dominio, è stato scelto di sviluppare un software che, dato un catalogo di item e un set di notizie (che rappresentano le notizie a cui è interessato il nostro utente) estrae un set di film la cui trama è coerente semanticamente con le notizie viste dall'utente. Consideriamo, per esempio, che l'utente stia leggendo notizie astronomiche che parlano di meteoriti, oppure notizie di cronaca che descrivono uno sciame di meteoriti molto vicino alla terra, il software proporrà film come "Armageddon" o "Deep Impact", ovvero film che hanno a che fare con lo spazio e con i meteoriti.
Possiamo notare che nel caso di studio non si è tenuto conto dei gusti e delle preferenze dell'utente, l'obiettivo coincide con l'identificazione di una relazione semantica tra un set di film e un set di notizie.
Per riuscire a sviluppare questo software abbiamo utilizzato strumenti di analisi semantica in maniera tale da riuscire ad estrarre un insieme di entità presenti nei testi delle notizie e nelle trame dei film.
Infine, per verificare l'efficacia delle parole chiave estratte sono stati utilizzati algoritmi di raccomandazione content-based presenti in letteratura e i risultati sono stati confrontati con tecniche di estrazione già esistenti.

In Figura 1.3 possiamo vedere uno schema semplificato del processo di

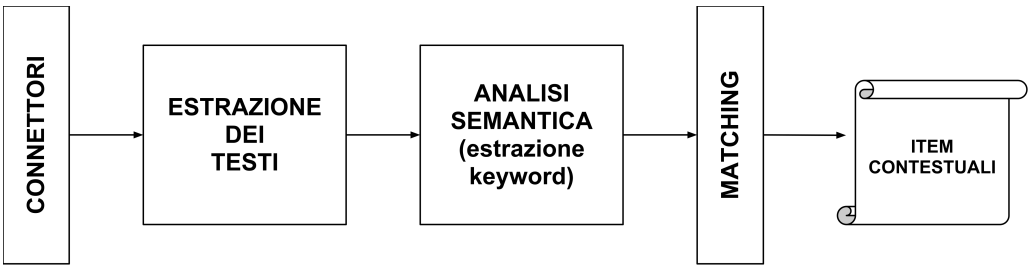

Figura 1.3: Schema del lavoro di tesi

raccomandazione: come primo passo vengono caricati film e news attraverso dei connettori(es., connettore a coming soon o al Wall Street Journal), a questo punto gli oggetti caricati vengono analizzati ed elaborati (arricchiti semanticamente come vedremo nel Capitolo 5), ne viene estratto il testo e in seguito vengono individuate le entità. Infine, viene effettuato un matching tra i film e il set di news selezionato, ovvero viene dato un punteggio ad ogni film in base alle parole chiave individuate sia in esso sia nella news; i film

con il maggior punteggio saranno degli oggetti contestuali candidabili per la raccomandazione.

Struttura della tesi Nel Capitolo 2 verrà offerta una panoramica sullo stato dell'arte dei sistemi di raccomandazione, con particolare attenzione ai sistemi di raccomandazione contestuali. Nel Capitolo 3 verrano illustrati i principali strumenti e algoritmi di analisi semantica utilizzati per l'estrazione di keyword da testi di medio-breve lunghezza. Nel Capitolo 4 verranno presentati i vari tipi di contesto individuati per il dominio di studio preso in considerazione. Nel Capitolo 5 verrà illustrato in maniera dettagliata il software sviluppato mentre nel Capitolo 6 verranno presentati i test fatti e le discussioni dei risultati ottenuti. Infine, nel capitolo conclusivo, verrà analizzato il lavoro fatto e verranno proposti temi di approfondimento per lavori futuri.

Capitolo 2

Stato dell'arte

In questo capitolo verrà illustrata brevemente la nascita dei sistemi di raccomandazione e i principali modelli utilizzati nel processo predittivo. Successivamente, in maniera più dettagliata, verranno presentati e descritti i sistemi di raccomandazione contestuali. Infine, verranno presentati i tre algoritmi principalmente utilizzati in questi particolari sistemi di raccomandazione.

2.1 Storia dei sistemi di raccomandazione

I sistemi di raccomandazione nascono dall'esigenza di selezionare un set di oggetti rilevanti per un determinato utente, in contesti in cui tale selezione e resa difficile dall'elevato numero di oggetti tra cui scegliere (es., un interessante libro da leggere all'interno di una biblioteca).
Essi iniziano a diffondersi come strumento per il filtraggio dell'informazione (*information filtering*) all'inizio degli anni '90. Dopo quasi vent'anni, i sistemi di raccomandazione hanno acquisito un ruolo fondamentale all'interno di molti siti di fama mondiale. Siti di e-commerce, come amazon.com e Ebay, utilizzano i sistemi di raccomandazione integrandoli alle loro query di ricerca per proporre oggetti sempre più mirati alle esigenze degli acquirenti, mentre siti di intrattenimento, come youtube e IMDb, cercano di proporre contenuti diversificati in base alle preferenze di ciascun utente.
Un significativo esempio dell'importanza dei sistemi di raccomandazione si può riscontrare nell'iniziativa di Netflix (servizio di noleggio film online) che nel 2009 sponsorizzò un concorso con in palio un premio da un milione di dollari per il team che fosse riuscito a migliorare con successo le performance del suo sistema di raccomandazione [53].
A livello accademico, negli ultimi anni, sono state dedicate conferenze e workshop al mondo dei sistemi di raccomandazione, come l'ACM Recom-

mender Systems (RecSys), fondata nel 2007 e ora divenuta il principale evento annuale per la ricerca sui sistemi di raccomandazione e sulle loro applicazioni.
Inoltre, sessioni dedicate ai sistemi di raccomandazione sono frequentemente incluse in conferenze più tradizionali di basi di dati, sistemi informativi e sistemi adattivi.
Tra queste si possono citare l'ACM Special Interest Group on Information Retrieval (SIGIR), User Modeling, Adaptation and Personalization (UMAP) e l'ACM's Special Interest Group on Management Of Data (SIGMOD).
Non dimentichiamo, infine, i riferimenti in importanti giornali accademici tra cui: AI Communications (2008), IEEE Intelligent Systems (2007), International Journal of Electronic Commerce (2006), International Journal of Computer Science and Applications (2006), ACM Transactions on Computer-Human Interaction (2005) e ACM Transactions on Information Systems (2004).

Questi studi hanno portato alla diffusione principalmente di due tecniche di raccomandazione: *content-based filtering* (CBF) e *collaborative filtering* (CF).

2.1.1 Content-Based Filtering

L'approccio basato sul filtraggio dei contenuti ha come obiettivo quello di suggerire ad un utente oggetti simili ad altri da lui già apprezzati.
Le prime tecniche content-based derivano dagli studi effettuati su *information retrieval* [14, 73] e *information filtering* [16].
In seguito a recenti sviluppi in questi due ambiti, e alla crescente importanza di applicazioni basate su informazioni testuali, si è sviluppato un filone di sistemi di raccomandazione content-based orientato ad oggetti contenenti informazioni testuali, per esempio url, news e messaggi. Il contenuto di questi sistemi è generalmente descritto con parole chiave.
Il filtraggio dei contenuti viene effettuato tramite diversi metodi, alcuni di questi metodi calcolano il profilo dell'utente come una media dei vettori dei contenuti [15, 47], come ad esempio l'algoritmo Rocchio [72]. In alternativa è possibile utilizzare un Bayesian classifier [66] per stimare la probabilità che un documento sia apprezzato. L'uso dell'algoritmo di Winnow [50] ha prodotto ottimi risultati in questo ambito, specialmente in situazioni in cui ci sono diversi tipi di caratteristiche degli item da tenere in considerazione [67]. Sono state impiegate anche altre tecniche di machine learning quali il clustering, gli alberi decisionali e le reti neurali [66].

L'individuazione dei gusti degli utenti possono essere dedotte in maniera *esplicita*, per esempio attraverso il rating che l'utente attribuisce ad un prodotto, oppure in maniera *implicita*, per esempio mantenendo in memoria la lista acquisti di ogni utente. Attualmente, proprio l'analisi dei log è la fonte principale di informazioni per le raccomandazioni online, tuttavia le inferenze desunte dalle interazioni degli utenti non sono sempre attendibili [42]. L'acquisizione esplicita, invece, è più accurata ma può gravare esageratamente sull'utente [57].
Uno dei principali problemi dell'*user profiling* nei sistemi di raccomandazione è che esso si basa fortemente sui dati estratti dai rating degli utenti, tuttavia la densità dei rating disponibili nei sistemi commerciali è spesso inferiore all'1% [75]: questo tipo di problema viene spesso indicato come *sparsità dei dati* o *cold start problem* [76].
Altra pratica sempre più diffusa nell'ambito dell'e-commerce è la falsificazione dei dati: un attacco molto comune consiste nel creare una piccola comunità di falsi utenti e assegnare rating positivi o negativi per avvantaggiare o svantaggiare determinati prodotti (*malicious rating*).
Inoltre, i sistemi di raccomandazione content-based sono piuttosto limitati in determinate circostanze [15, 78].
In prima istanza, queste tecniche sono limitate dalle caratteristiche esplicitamente associate agli item che si intende raccomandare. Sebbene sia presente una florida letteratura riguardante metodi per estrarre automaticamente dati di tipo testuale, altri domini sono meno adatti per questo tipo di analisi (si pensi a qualsiasi tipo di dati multimediali, immagini, video, audio), quindi spesso è richiesto un contributo manuale per l'individuazione delle caratteristiche e da un punto di vista pratico questa è una grossa limitazione [78].
Si consideri, inoltre, che è impossibile distinguere due item diversi se i rispettivi insiemi di caratteristiche sono uguali. Questo problema è piuttosto evidente in domini di tipo testuale, infatti un sistema di raccomandazione non è in grado di distinguere un articolo ben scritto da uno di bassa qualità, questo accade poichè la sua analisi è strettamente limitata alle occorrenze dei termini nel testo ed ogni articolo è individuato esclusivamente da un set di parole chiave, quindi da informazioni poco rappresentative per poter inferire sulla qualità [78].
Un'altra questione importante è nota in letteratura come *over-specialization*. Si tratta di quel fenomeno per cui l'algoritmo tenderà a raccomandare item sempre più simili a quelli già predetti.
Si pensi ad esempio alla raccomandazione di film: un utente appassionato di film di azione difficilmente riceverà un suggerimento per un film romantico. Una soluzione che generalmente viene adottata è quella di introdurre delle

componenti casuali nella raccomandazione, ad esempio adoperando algoritmi genetici [79]. In generale la varietà di raccomandazioni è una caratteristica desiderabile. Alcuni algoritmi, come il DailyLearner [20], cercano di garantire una certa varietà non solo tenendo in considerazione item corrispondenti ai gusti dell'utente, ma anche escludendo item troppo simili ad altri precedentemente che ha precedentemente preso in esame. Per alleviare questo problema nell'ambito della raccomandazione di documenti sono state proposte alcune misure di ridondanza per stabilire se un documento, oltre ad essere rilevante, contenga anche nuove informazioni [85].
Infine, un'altra grande limitazione dei sistemi content-based riguarda il numero di rating necessari affinché l'algoritmo sia in grado di effettuare una predizione. Un nuovo utente, quindi, avrà pochi rating e conseguentemente la raccomandazione non potrà essere accurata. In letteratura ci si riferisce a questo problema come *new user problem*.

2.1.2 Collaborative Filtering

I sistemi di raccomandazione collaborativi (o *collaborative filtering systems*) hanno come obiettivo quello di suggerire oggetti ad un utente, in base ai rating rilasciati da un gruppo di utenti simili (*neighbourhood*).
Proviamo ad immaginare uno scenario di raccomandazione in cui gli item fanno parte di una collezione di film visionabili online: un sistema di raccomandazione collaborativo cercherà di individuare gli utenti simili all'utente per cui si intende effettuare la raccomandazione, ovvero utenti con gli stessi gusti cinematografici, e raccomanderà quindi al nostro utente film più apprezzati da questo set di utenti.
Dal momento che il collaborative filtering si basa su rating di un set di utenti con un profilo simile a quello dell'utente destinatario della raccomandazione, vi è la necessita di avere a disposizione profili utente contenenti la storia pregressa dei rating. In tal modo non è necessaria nessuna conoscenza sugli item, quindi nessun intervento umano (catalogo di metadati), e questo motivo porta il collaborative filtering ad essere un sistema di raccomandazione usato con successo in quasi tutti i siti di e-commerce.
Nel corso degli anni sono stati sviluppati diversi sistemi di raccomandazione collaborativi, uno dei primi è il Grundy [70], utilizzato per la raccomandazione di libri, il quale proponeva l'utilizzo di stereotipi per costruire modelli di utenti basandosi su una quantità limitata di informazioni.
I primi sistemi ad utilizzare questo tipo di algoritmi per la predizione automatica furono GroupLens [45, 69], Video Recommender [35], e Ringo [78]. Altri noti esempi di collaborative filtering sono il sistema di Amazon per la

raccomandazione di libri e PHOAKS per suggerire informazioni rilevanti sul web [81].
Gli algoritmi di tipo collaborativo possono essere raggruppati in due classi: *memory-based* e *model-based* [13].
Gli algoritmi memory-based [19, 28, 59, 69, 78] sono essenzialmente euristiche basate sull'insieme di preferenze precedentemente espresse dagli utenti, mentre gli algoritmi model-based [18, 19, 32, 33, 52, 65, 83] utilizzano le preferenze espresse dagli utenti per apprendere un modello con cui poi effettuare le predizioni. Questo tipo di algoritmi non soffrono della maggior parte dei problemi descritti per gli algoritmi content-based; essi, ad esempio, non dipendono dalla tipologia degli item, e sono insensibili anche al caso in cui nuovi item siano diversi da item osservati in precedenza.
Non bisogna però dimenticare che anche i sistemi di collaborative filtering soffrono di alcune debolezze. Analogamente a quanto detto per i sistemi content-based, è presente il problema del nuovo utente: un utente nuovo, con una storia di rating povera, potrebbe avere difficoltà a trovare utenti simili a lui.
Oltre al problema del nuovo utente, si presenta anche il problema del nuovo oggetto. Gli oggetti, come visto in precedenza, vengono raccomandati esclusivamente sulle preferenze degli utenti, quindi un oggetto nuovo, ovvero una novità nel catalogo, si ritroverà senza rating e non sarà sponsorizzato nelle raccomandazioni.
Infine, rimane un problema comune a tutti i sistemi di raccomandazione: il problema della scarsità dei dati ovvero quando il numero di rating ottenuti è generalmente molto piccolo rispetto a quello che si vorrebbe predire. Si immagini uno scenario di raccomandazione di film: i film rivolti ad un pubblico di nicchia sono valutati solo da un'esigua porzione di utenti; per questo motivo il sistema li raccomanderà piuttosto raramente, anche nel caso in cui questi pochi utenti abbiano assegnato valutazioni molto alte.
Si può facilmente pensare ad uno scenario duale in cui un utente abbia dei gusti particolari con quasi nessun punto in comune con altri utenti, anche in questo caso la predizione sarà poco efficace [15].
Una soluzione che generalmente viene adottata per risolvere il problema della sparsità è quella di calcolare la similarità tra utenti utilizzando altri dati oltre quelli delle valutazioni, ad esempio i dati demografici: sesso, età, posizione geografica, istruzione e posizione sociale. Questa tecnica è nota come demographic filtering [67]. Un altro approccio importante per ovviare il problema della sparsità consiste nell'utilizzare una tecnica di riduzione della dimensionalità della matrice dei rating: Singular Value Decomposition (SVD) [18, 74].

2.2 Sistemi di raccomandazione contestuali

In questo paragrafo illustreremo la nozione generale di contesto e vedremo come potrà essere modellato in un sistema di raccomandazione. Infine illustreremo tre differenti metodi di raccomandazione contestuali: *contextual prefiltering*, *contextual postfiltering* e *contextual modeling*.
L'importanza delle informazioni contestuali è stata riconosciuta da molti ricercatori e professionisti in numerosi campi dell'informatica, alcuni esempi sono la personalizzazione dell'e-commerce, l'information retrieval, il data mining e l'appoggiarsi ad esse per decisioni di tipo marketing e management. Come già scritto, durante questi anni sono stati effettuati numerosi studi nel campo dei sistemi di raccomandazione il cui scopo principale è stato quello di consigliare l'oggetto più idoneo ai gusti dell'utente, tralasciando completamente qualsiasi informazione contestuale (come per esempio il luogo, il tempo, la compagnia di altre persone ecc...). In poche parole, i sistemi di raccomandazione tradizionali utilizzano solo due tipi di entità, user e item, tralasciando qualsiasi informazione contestuale durante la raccomandazione. Tuttavia, in molti casi, non è sufficiente considerare solamente user e item, è altrettanto importante incorporare informazioni contestuali durante il processo di raccomandazione in maniera tale da fornire risposte del tipo $User \times Item$ diverse da circostanza a circostanza. Per esempio, utilizzando il contesto temporale, una raccomandazione su un viaggio vacanza produrrà risultati differenti nel caso si tratti di periodo estivo o di periodo invernale. Analogamente, per quando riguarda i contenuti di un sito web, è importante determinare quali debbano essere presentati all'utente e quando: l'utente potrebbe preferire leggere news di cronaca la mattina, i report di borsa la sera e durante i weekend leggere recensioni di film e libri.
Da questi esempi, risulta evidente come il contesto abbia un forte impatto nei processi decisionali degli utenti e di conseguenza nelle tecniche di marketing aziendali. Naturalmente, l'efficienza di questi metodi è dovuta allo studio delle variabili contestuali più adatte per lo sviluppo della raccomandazione. Ad esempio, il tempo atmosferico non influenzerà la scelta di quale film vedere al cinema, al contrario la compagnia (fidanzata, famiglia o amici) potrebbe influenzare in maniera consistente la scelta.

Prima di proseguire la discussione del ruolo e delle opportunità offerte della informazioni contestuali, analizziamo brevemente la nozione di contesto.

2.2.1 Il contesto

Durante gli anni, la nozione di contesto è stato studiata da molteplici discipline, scientifiche e non, e ogni disciplina ha tentato di dare la propria definizione. Di seguito cercheremo di descrivere il contesto in quei campi di applicazioni collegati all'informatica.

- *Data mining*: nella comunità che si occupa di data mining, il contesto è definito come l'insieme degli eventi che identificano lo stato di un cliente nel sistema e che possono determinare un cambiamento nelle sue preferenze, nel suo status e nel suo valore per la compagnia [17]. Esempi di contesti possono essere: un nuovo lavoro, la nascita di un figlio, il matrimonio, il divorzio o la pensione.
  La conoscenza di queste informazioni contestuali porta alla costruzione di mining pattern pertinenti ad un contesto particolare focalizzandosi solo su dati e risultati rilevanti.

- *E-commerce*: Palmisano et al. [63] decidono di considerare l'intenzione di un acquisto da parte di un utente come una informazione contestuale. Per esempio, lo stesso utente può effettuare diversi acquisti dallo stesso account per diverse ragioni: un libro per passare del tempo, un libro per il lavoro, un libro da regalare.
  Attraverso queste informazioni possiamo creare per ogni utenti diversi profili che potranno essere utilizzati per costruire diversi modelli predittivi.

- *Mobile context-aware systems*: solo recentemente con la diffusione di massa degli smartphone, informazioni contestuali come l'aggiornamento della posizione dell'utente e l' identità delle persone e dei luoghi vicino a lui, sono diventate importanti per i sistemi di raccomandazione [58, 62, 77]. Per esempio, il teatro di Broadway propone sconti sui biglietti ai visitatori di time square 30 minuti prima dell'inizio dello spettacolo, il tutto informandoli via smartphone o atri dispositivi di comunicazione.

- *Databases*: negli ultimi anni alcuni dbms affiancano le informazioni contestuali ai profili utente, in maniera da restituire risposte differenti in base al contesto in cui vieni formulata la query e al rapporto dell'utente con esso. Inoltre sono stati definiti nuovi linguaggi e versioni estese di sql che permettono di eseguire query contestuali [9, 40, 56].

- *Information retrieval*: è stato dimostrato che le informazioni contestuali possono essere di aiuto nel campo dell'information retrieval [39],

anche se la maggior parte dei sistemi esistenti effettua le sue decisioni tralasciando tutto quello che riguarda il contesto [10].
L'efficacia di un sistema di recupero informazione contestuale, si basa sull'abilità di saper generare delle query contestuali in grado di ottenere risultati rilevanti sul contesto dato [48, 80].

- *Marketing and management*: gli esperti di marketing hanno sostenuto che il processo di acquisto è subordinato al contesto in cui ha luogo l'operazione, dal momento che, lo stesso cliente, può adottare strategie decisionali diverse a seconda dell'ambiente in cui opera [38].
  Secondo Lilien et al.[31], i consumatori variano le loro scelte decisionali a secondo dell'utilizzo del bene o del servizio (per la famiglia, per il regalo, per sé) e della situazione di acquisto (vendita online, vendita in-store). Per cui, un efficace sistema di predizione delle scelte utente dipende dal grado di conoscenza delle informazioni contestuali.

In questa sezione è stato mostrato come il contesto sia utilizzato da varie discipline informatiche e interpretato da ognuna di esse. Per stabilire un certo ordine in questa diversità di interpretazione Dourish [29] cerca di introdurre una classificazione dei vari tipi di contesto individuando due grandi famiglie: intenzionali e rappresentative.
Il contesto rappresentativo è definito a priori come un set di attributi osservabili e non mutabili con il passare del tempo. Al contrario il contesto intenzionale è definito dal comportamento dell'utente e può non essere osservabile dall'esterno.

2.2.2 Modellare informazioni contestuali

Come illustrato precedentemente un sistema di raccomandazione, dopo aver raccolto in maniera implicita o esplicita un set di rating, cerca di stimare il valore di nuovi rating secondo la funzione R per la coppia (user,item) che non è stata ancora valutata dall'utente.

$$R : User \times Item \rightarrow Rating$$

Per Rating si intende un set di valori ordinati (non negativi e compresi in un range) mentre User e Item sono rispettivamente i domini degli utenti e degli oggetti. Dopo che la funzione R elabora risultati su tutto lo spazio User x Item, un sistema di raccomandazione raccomanderà gli oggetti con il rating più alto(o gli n oggetti con i rating più alti).
Questo sistema viene chiamato *Modello tradizionale* o *Modello bi-dimensionale*, dal momento che vengono considerate solo due dimensioni. Il problema è

quindi ridotto alla ricerca di rating per oggetti non ancora visionati dall'utente.
Come visto precedentemente, nessuna di queste raccomandazione tiene conto della situazione contestuale. L'obiettivo successivo è quindi quello di riuscire ad integrare le informazioni contestuali all'interno della raccomandazione. Possiamo modellare questo tipo di raccomandazione nella seguente funzione R:

$$R : User \times Item \times Context \rightarrow Rating$$

dove Context rappresenta l'insieme delle informazioni contestuali associate alla raccomandazione.
Per illustrare al meglio questi concetti, consideriamo il seguente esempio.
Analizziamo un sistema di raccomandazione di film, dove utenti e film sono descritti dai seguenti attributi:

- Film: il set di tutti i film della collezione, definiti come Film(Id, Titolo, Durata, Anno, Regista, Genere);
- Utente: l'insieme di persone a cui raccomandare i film, definiti come Utente(Id, Nome, Indirizzo, Età, Sesso, Professione);

Inoltre, consideriamo i tre diversi tipi di informazioni contestuali:

- Cinema: il cinema che proietta il film, definito come Cinema(Id, Nome, Indirizzo, Capacità, Città, Stato);
- Tempo: il momento in cui il film può essere visto, definito come Tempo(Data, Giorno della settimana, Periodo della settimana, Mese, Anno);
- Compagnia: rappresenta una persona o un insieme di persone con cui poter vedere un film, definito come Compagnia(Tipo di compagnia). L'attributo Tipo di compagnia può avere come valore: da solo, Fidanzato/Fidanzata, Famiglia, Colleghi e Altri.

A questo punto, è facile intuire che il rating assegnato ad un film da un persona varia in base a dove, a quando e con chi è stato visto, quindi il sistema di raccomandazione in base al contesto deve saper produrre risultati coerenti con esso. Per esempio, il tipo di film che lo studente Mario Rossi vorrà vedere sabato sera con la sua fidanzata molto probabilmente differirà dal film che ha in programma di vedere in settimana con la sua famiglia.
Come possiamo vedere da questo esempio e da molti altri casi, le informazioni contestuali possono essere di diverso tipo e ogni tipo definisce un

certo aspetto del contesto. Inoltre, non sempre la loro struttura è di facile interpretazione ed è proprio per questo che sono molti gli studi che si sono occupati della loro rappresentazione. Alcuni esempi di rappresentazione possono essere strutture gerarchiche [6, 63], modelli a memoria breve[12], set di dimensioni contestuali[6, 63] oppure cubi multidimensionali simili a quelli delle analisi OLAP[25, 43].
L'obiettivo di questi studi è quello di individuare in maniera automatica la rilevanza delle informazioni contestuali per ottimizzare la raccomandazione, tutto questo viene fatto attraverso procedure [24], di machine learning [44] e di data mining [51].

2.2.3 Come ottenere le informazioni contestuali

Vi sono principalmente tre metodi per ricavare informazioni contestuali:

- *Esplicite*: vi è un approccio diretto con gli utenti, ponendo domande attraverso questionari. Per esempio, un sito internet può ottenere informazioni contestuali chiedendo ai propri utenti di compilare un form oppure di rispondere a specifiche domande prima di accedere ad alcuni sezioni del sito;

- *Implicite*: in questo caso vengono elaborate informazioni dai dati a disposizione del sistema, come per esempio il rilevamento di posizione attraverso i dispositivi gps oppure il timestamp di una transazione. In questo caso non vi è nessuna interazione con l'utente;

- *Dedotte*: queste informazioni vengono elaborate attraverso analisi statistiche o di data mining. Per esempio, l'identità della persona che sta guardando la televisione in uno specifico momento non può essere identificata dalla compagnia televisa ma potrebbe essere dedotta analizzando i canali che la persona si ferma a vedere.
  Affinchè questo sia possibile è necessario sviluppare un modello predittivo ed effettuare dei test di training con dati appropriati.

Abbiamo visto, quindi, che le informazioni contestuali possono essere nascoste nei dati in qualche forma latente e il sistema può estrarle implicitamente per effettuare raccomandazioni senza conoscere in maniera esplicita il contesto.
Dopo aver raccolto le informazioni contestuali, bisogna effettuare un'attenta analisi per definire quali attributi sono utili e quali meno ai fini della raccomandazione. Adomavicius [6] propone di selezionare inizialmente un vasto set di attributi definiti da esperti del dominio di raccomandazione come

possibili candidati per l'applicazione. Per esempio, in un sistema di raccomadazione di film, come quello visto precedentemente, possiamo considerare attributi come il Tempo, i Cinema, la Compagnia, il Tempo Atmosferico. In seguito, dopo aver raccolto e analizzato i dati (rating e informazioni contestuali), possiamo utlizzare diversi modelli per individuare quali attributi siano effettivamente significanti ai fini della raccomandazione. Per esempio, possiamo vedere se il bello o il brutto tempo oppure se vedere un film da solo o in compagnia influenza la raccomandazione in maniera significante. Questa procedura porta ad un filtraggio degli attributi presi in considerazione e al raggiungimento di un set valido per l'applicazione.

2.3 Modelli per sistemi di raccomandazione contestuali

I primi utilizzi di informazioni contestuali nei sistemi di raccomandazione possono essere fatti risalire al lavoro di Herlocker e Konstan [34]. I due ipotizzarono che, in alcune applicazioni, l'inclusione di informazioni contestuali riguardanti l'utente nell'algoritmo di raccomandazione avrebbero potuto portare a raccomandazioni migliori. Per esempio, se si vuole consigliare un libro come regalo ad un bambino, potremmo osservare quali, tra i libri che il bambino possiede, gli piacciono e passare queste informazioni al sistema di raccomandazione per il calcolo di nuovi libri simili.
Si noti che questo approccio opera all'interno dello spazio 2D tradizionale (utente e item), tuttavia, serve a illustrare come informazioni aggiuntive (come un set specifico di oggetti, oppure obiettivo della raccomandazione) possano essere incorporati all'interno di una raccomandazione standard. Inoltre, l'uso di punteggi assegnati ai temi di interesse possono essere utilizzati per creare dei profili contestuali per ogni utente [86].
I diversi approcci utilizzati per inserire le informazioni contestuali nei sistemi di raccomandazione, possono essere classificati in due gruppi:

- raccomandazioni contestuali attraverso query e ricerca
- raccomandazioni contestuali statistiche

Il primo metodo è un approccio utilizzato da una vasta gamma di sistemi turistici nel campo della telefonia mobile [5, 23].
I sistemi che utilizzano questo approccio utilizzano le informazioni contestuali per eseguire query o cercare un certo insieme di risorse (ad esempio, ristoranti) e presentare le migliori risorse corrispondenti all'utente(ad esempio, ristoranti nelle vicinanze che sono attualmente aperti).

Queste informazioni possono essere ottenute direttamente dall'utente (specificando stato d'animo o interessi particolari) o dall'ambiente (ottenendo l'ora locale, il tempo o la posizione corrente).
Uno dei primi esempi ad utilizzare questo approccio è il progetto Cyberguide [5], che ha sviluppato diversi prototipi di guide turistiche per diverse piattaforme portatili.
Il secondo approccio per utilizzare informazioni contestuali nel processo di raccomandazione, utilizza principalmente analisi statistiche e rappresenta una più recente tendenza nel campo dei modelli contestuali di raccomandazione [6, 61, 64, 84]. A differenza del precedente metodo, che integrava le informazioni contestuali nelle query, le tecniche di questo secondo modello tendono a studiare ed imparare le preferenze degli utenti, ad esempio, osservando le interazioni di più utenti con il sistema oppure studiando i gusti dell'utente su un determinato set di oggetti.
Sebbene entrambi gli approcci offrano casi di studi molto interessanti, nel proseguo del capitolo ci soffermeremo principalmente sull'approccio statistico e sulla possibilità di combinare entrambi gli approcci in un unico sistema. Un esempio di sistema ibrido è UbiquiTO [23], l'applicazione implementa una guida turistica in ambito mobile e fornisce dei risultati non solo sulla base delle informazioni contestuali esplicite, ma anche attraverso diverse tecniche di modelli statistici per adattare l'applicazione in base alle preferenze e i gusti dell'utenza. Un'altro esempio è il News@hand system [22], il quale utilizza tecniche semantiche per fornire raccomandazioni personalizzate delle notizie basate sui gusti dell'utente.

Come già illustrato in precedenza, nella sua forma generale, un sistema tradizionale di raccomandazione a due dimensioni può essere descritto come una funzione il cui input sono delle parziali informazioni sui gusti dell'utenza e il cui output produce una lista di raccomandazioni per ciascun utente.
La Figura 2.1, ci presenta un semplice schema con i tre componenti della raccomandazione: data (input), sistema di raccomandazione 2D (funzione) e lista di raccomandazioni (output). Si può notare che dopo aver definito la funzione di raccomandazione in base ai dati disponibili, per ogni generico utente u viene generata dalla funzione una lista di raccomandazioni, la quale organizza gli oggetti estratti in base al rating predetto.

L'introduzione di una variabile contestuale ci permette quindi di modificare il set $\langle utente, oggetto, rating\rangle$ in $\langle utente, oggetto, contesto, rating\rangle$, dove ogni record non include soltanto quanto un oggetto è piaciuto ad un utente ma anche in quale circostanza quel voto sia stato dato.
La presenza di informazioni contestuali, come si può vedere in Figura 2.2, ci

Figura 2.1: Sistema di raccomandazione 2D tradizionale

porta ad avere una forma $U \times I \times C \times R$, dove C è la dimensione contestuale rappresentata da una lista di raccomandazioni contestuali $i_1, i_2, i_3, \ldots$ per ogni utente.
Come si può notare possiamo decidere di tenere in considerazione le informazioni contestuali in diverse fasi del processo di raccomandazione.
Per essere più precisi, possiamo dire che si possono estrapolare tre diversi paradigmi di raccomandazione [8]:

- *Contextual pre-filtering*: in questo paradigma di raccomandazione (presentato nella Figura 2.2a), le informazioni contestuali vengono usate per selezionare e costruire un set di dati rilevanti.
  A questo punto i rating posso essere predetti usando un qualsiasi tradizionale sistema di raccomandazione 2D.

- *Contextual post-filtering*: in questo paradigma (Figura 2.2b) le informazioni contestuali sono inizialmente ignorate e i rating sono predetti usando un qualsiasi tradizionale modello 2D sulla totalità dei dati.
  A questo punto, il risultante set di raccomandazione, è adattato ai gusti di ogni utente utilizzando le informazioni contestuali a disposizione.

- *Contextual Modeling*: in questo paradigma (Figura 2.2c) le informazioni contestuali sono usate direttamente nella tecnica di modellizzazione per la predizione di rating.

Di seguito, verrano affrontati in maniera più dettagliata i paradigmi appena presentati.

2.3.1 Contextual Pre-Filtering

Come mostrato in Figura 2.2a, l'approccio del pre-filtering utilizza informazioni contestuali per costruire un set di dati, i quali verrano utilizzati come input della raccomandazione.
Uno dei principali vantaggi di questo approccio è dato dalla possibilità di

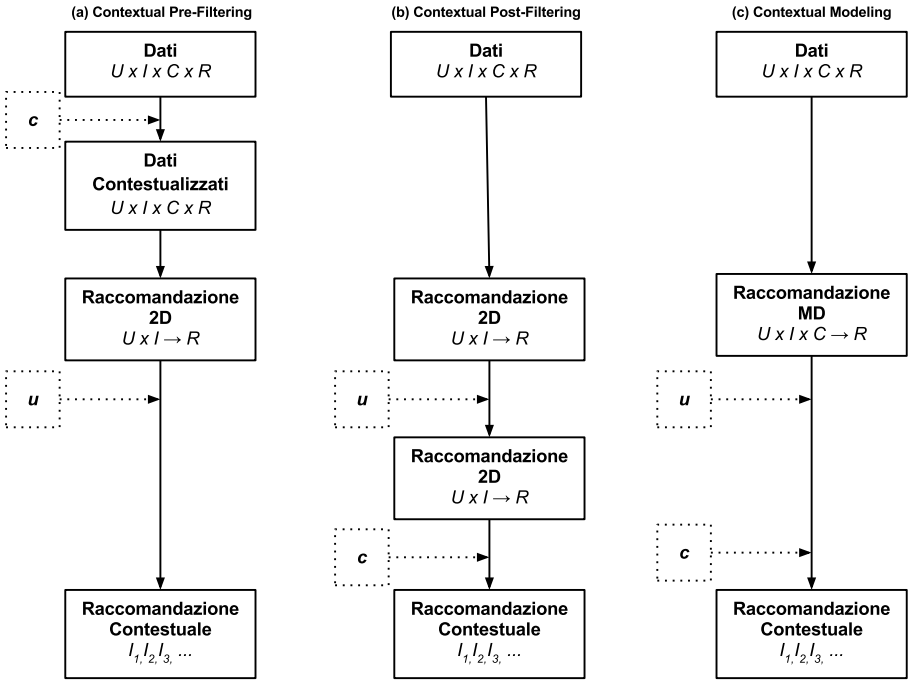

Figura 2.2: Paradigmi per incorporare il contesto nel processo di raccomandazione

utilizzare numerose tecniche tradizionali di raccomandazione in letteratura [7]. In particolare, in un possibile impiego di questo approccio, il contesto c viene utilizzato in una query di selezione (filtraggio) pertinente ai dati.
Prendiamo come esempio un sistema di raccomandazione di film. Se un utente vuole vedere un film sabato sera, solamente i rating rilasciati sui film proiettati di sabato verranno usati per la raccomandazione. In altre parole, la query di filtraggio dei dati è stato costruita usando esattamente il contesto specificato.
Adomavicius [6] propone per il paradigma del pre-filtering un approccio basato sulla riduzione, in maniera tale da ridurre il problema della multidimensionalità introdotta dal contesto sul modello standard 2D ($User \times Item$).In questo modo, dopo aver effettuato la riduzione, è possibile applicare gli algoritmi noti.
In particolare, consideriamo

$$R^{D}_{User \times Item} : U \times I \rightarrow Rating$$

come una qualsiasi funzione di raccomandazione 2D che, dati alcuni esistenti ratings su D (D contiene i record nella forma $\langle user, item, rating \rangle$ per ogni utente di cui è conosciuto il rating), può essere usata per produrre un qual-

siasi rating, come per esempio $R^D_{User \times Item}(John, StarWars)$.
Quindi, è possibile definire una funzione a 3 dimensioni con l'introduzione del contesto nella forma

$$R^D_{User \times Item \times Time} : U \times I \times T \rightarrow Rating$$

dove D conterrà i record $\langle user, item, time, ranking \rangle$.
A questo punto la funzione di predizione a 3 dimensioni può essere espressa attraverso un modello 2D in diversi modi, incluso il seguente:

$$\vee(u, i, t) \in U \times I \times T, R^D_{User \times Item \times Time}(u, i, t) = R^{D[Time=t](User,Item,Rating)}_{User \times Item}(u, i).$$

Qui, $[Time = t]$, rappresenta un semplice filtro contestuale,invece, D $[Tempo = t](User, Item, Rating)$ indica un insieme di dati ottenuto da D selezionando solo i record in cui la dimensione Time ha valore t.
Quindi, se trattiamo un set di dati in 3 dimensioni, D è semplicemente una relazione ottenuta da D eseguendo due operazioni: prima una selezione e, successivamente, una proiezione.
Tuttavia, il contesto considerato a volte può essere troppo ristretto.
Consideriamo ad esempio, la raccomandazione contestuale attraverso la quale vorremmo ci venissero restituiti i film da vedere con la propria fidanzata al cinema il sabato sera $(c = (fidanzata, cinema, sabato))$.
Utilizzando esattamente questo contesto il processo di pre-filtering potrebbe essere problematico.
In primo luogo, certi aspetti del contesto eccessivamente specifici potrebbero non risultare significativi. Per esempio, il film che l'utente vorrebbe vedere al cinema sabato sera con la propria ragazza potrebbe essere lo stesso di domenica ma diverso da quello di mercoledì. Pertanto, può essere più appropriato in alcuni casi generalizzare il contesto, vale a dire per esempio, utilizzare fine settimana invece di sabato.
In secondo luogo, il contesto definito in maniera così specifica potrebbe non avere abbastanza riscontri nella base di dati, dando luogo al problema già accennato precedentemente della sparsità di dati.

Generalizzazione del contesto Sempre Adomavicius [6] introduce la nozione di pre-filtering generalizzato, che permette di generalizzare i dati della query di filtraggio ottenuti sulla base di un contesto specificato.
Più formalmente, definiamo $c' = (c'_1, \ldots, c'_k)$ come una generalizzazione del contesto $c = (c_1, \ldots, c_k)$ se e solo se $c_i \rightarrow c'_i$ per ogni $i = 1, \ldots, k$ nella gerarchia contestuale corrispondente. A questo punto, c'_1 può essere utilizzato nella raccomandazione al posto di c.

Seguendo l'idea di contesto generalizzato, viene proposto quindi di non utilizzare un semplice pre-filtro $[Time = t]$, che rappresenta l'esatto contesto t del rating (u, i, t), ma piuttosto di usare un filtro generalizzato $[Time = S_t]$, dove S_t denota un superset del contesto t. S_t viene chiamato contesto segmentizzato[3].
Per esempio, se vogliamo predire quanto Mario Rossi voglia vedere il film Il Gladiatore di lunedì, per calcolare

$$R^D_{User \times Item \times Time}(MarioRossi, IlGladiatore, Lunedi),$$

possiamo utilizzare oltre alle raccomandazioni filtrate per il singolo giorno di lunedì, le raccomandazione filtrate per Giorno della Settimana. Quindi, per ogni (u, i, t), dove $t \in GiornodellaSettimana$, possiamo predire il rating come

$$R^D_{User \times Item \times Time}(u, i, t) = R^{D[Time \in Weekday](User, Item, AGGR(Rating))}_{User \times Item}(u, i)$$

In maniera più generale, per stimare i rating R(u,i,t) possiamo usare contesti segmentati specifici S_t come:

$$R^D_{User \times Item \times Time}(u, i, t) = R^{D[Time \in S_t](User, Item, AGGR(Rating))}_{User \times Item}(u, i)$$

L'espressione $AGGR(Rating)$ usata nelle funzione precedenti, viene utilizzata perché potrebbero esserci rating assegnati dallo stesso utente sullo stesso oggetto in istanze di tempo differenti ma appartenenti entrambe al contesto segmentizzato. Per esempio, rating diversi il lunedì e il martedì, ma comunque appartenti al segmento Giorni della Settimana. Pertanto, quando si generalizza la dimensione contestuale, bisogna utilizzare qualche funzione, anche una semplice media, per aggregare i dati.
Anche se può sembrare banale, facciamo notare che esistono differenti possibilità di generalizzazione del contesto, basate sulla tassonomia del contesto e sulla granularità del contesto desiderata.
Per esempio, assumiamo di avere la seguente tassonomia contestuale:

- Company: Fidanzata → Amici → Non da solo → Da solo
- Luogo: Cinema → Qualunque luogo
- Tempo: Domenica → Weekend → Qualsiasi Giorno

I seguenti sono solo alcuni esempi di possibili generalizzazioni c' del contesto appena precedentemente menzionato $c = (Fidanzata, Cinema, Sabato)$:

- $c' = (Fidanzata, Qualunqueluogo, Sabato)$

- $c' = (Amici, Cinema, QualsiasiGiorno)$
- $c' = (Nondasolo, Cinema, Weekend)$

Quindi, la scelta del filtro corretto si presenta come uno dei problemi fondamentali nei sistemi di raccomandazione contestuale.

2.3.2 Contextual Post-Filtering

Come mostrato in Figura 2.2b, l'approccio post-filtering non considera il contesto come dato di input per la generazione di raccomandazioni, vale a dire, che il sistema si comporterà come un normale sistema di raccomandazione 2D e produrrà una graduatoria di tutti gli elementi candidati. Solo a questo punto, le variabili contestuali influenzeranno la raccomandazione modificando la lista degli elementi proposti.
La regolazione della lista di raccomandazione può essere fatta da:

- *Filtraggio:* in questo caso vengono eliminate dalla raccomandazione gli oggetti ritenuti non rilevanti nel contesto preso in considerazione;
- *Modifica del ranking:* viene modificato l'ordine della presentazione degli oggetti, gli oggetti inerenti al contesto hanno un peso maggiore nel ranking totale.

Per esempio, in una raccomandazione di film, se una persona vuole vedere un film nel weekend, e durante la settimana guarda solo commedie, il sistema può filtrare tutti i film che non siano commedie dalla lista di raccomandazione proposta.
Quindi, l'idea alla base dell'approccio post-filtering è quello di analizzare le preferenze contestuali dell'utente in un dato contesto per trovare un pattern specifico (per esempio il fatto di guardare sol commedie durante la settimana) e poi usare questo pattern per regolare la lista delle raccomandazioni, proponendo in questo modo raccomandazioni contestuali, come illustrato in Figura 2.3.

Come per molte altre tecniche di raccomandazione, le raccomandazioni effettuate con un approccio di tipo post-filtering possono essere inserite nella famiglie delle tecniche euristiche o nella famiglia delle tecniche di model-based.
Approcci di tipo euristico si concentrano sulla ricerca di elementi comuni caratteristiche (attributi) per un determinato utente in un dato contesto (ad esempio, quale attore piace guardare all'utente quando questo è triste?), con l'obiettivo di utilizzare questi attributi per regolare le raccomandazioni.
Questo viene effettuato con:

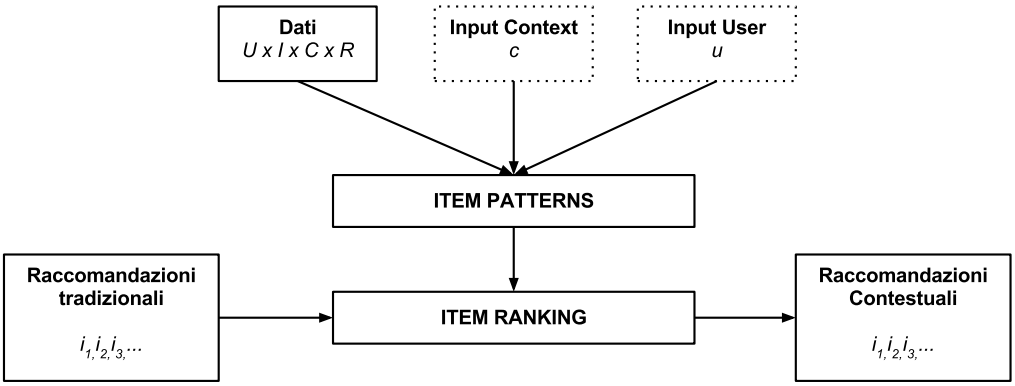

Figura 2.3: Fase finale dell'approccio post-filtering

- *Filtraggio:* vengono eliminati gli oggetti che non hanno un significativo numero di queste caratteristiche;
- *Modifica del ranking:* in base al numero di caratteristiche gli oggetti vengono inseriti in una classifica pesata.

Il mode-based, a differenza, può costruire modelli predittivi che possono calcolare la probabilità che un utente scelga un determinato oggetto in un contesto specifico. Anche in questo caso vengono effettuati:

- *Filtraggio:* vengono eliminati gli oggetti che hanno una bassa probabilità di superare la soglia minima accettabile;
- *Modifica del ranking:* gli oggetti vengono ordinati in base alla probabilità che possano piacere o meno all'utente.

2.3.3 Contextual Modeling

Come mostrato in Figura 2.2c, l'approccio contextual modeling usa le informazioni contestuali direttamente nella funzione di raccomandazione. Esse vengono utilizzate come variabili predittive esplicite per la scelta del rating dato da un user ad un determinato oggetto.
Mentre gli approcci del pre-filtering e del post-filtering possono utilizzare le funzioni di raccomandazione 2D presenti in letteratura, il contextual modeling sviluppa una funzione di raccomandazione realmente multidimensionale, che rappresenta un modello predittivo (sviluppato attraverso alberi decisionali, funzioni di regressione, modelli probabilistici o altre tecniche) oppure un calcolo euristico che incorpora informazioni contestuali in aggiunta ai dati dall'utente e dell'oggetto, $Rating = R(User, Item, Context)$.

E' importante notare comunque che questi algoritmi multidimensionali possono essere implementati partendo dalle classiche funzioni di raccomandazione 2d utilizzate negli ultimi anni 10-15 anni.

2.4 Combinare multipli approcci

Come è stato ben documentato in letteratura una combinazione di sistemi di raccomandazione forniscono significativi miglioramenti di performance rispetto ai singoli approcci [21, 46, 55, 68].
I tre paradigmi presentati offrono diverse opportunità per l'impiego di approcci combinati.

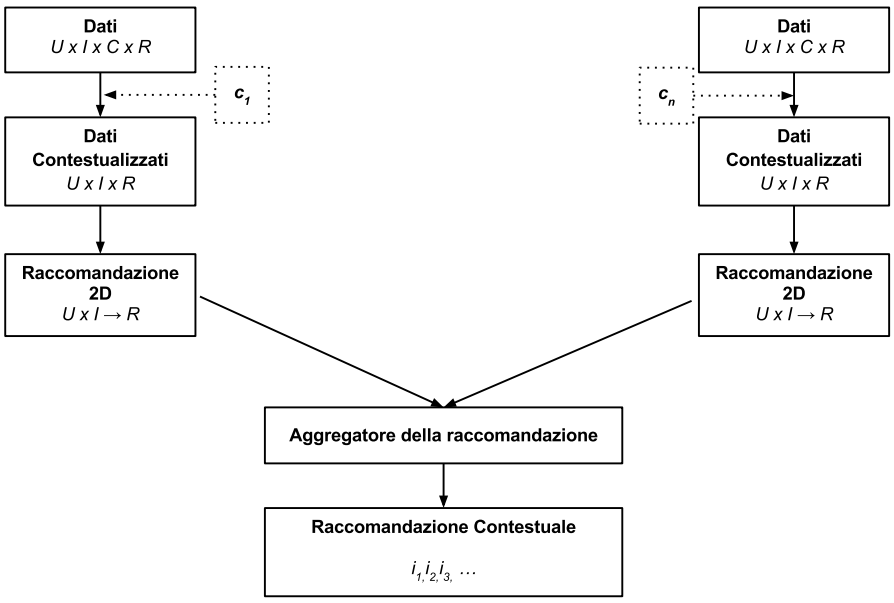

Figura 2.4: Approccio multiplo di sistemi pre-filtering

Una possibilità è quella di sviluppare e combinare diversi modelli dello stesso tipo. Per esempio, Adomavicius [6] segue questo approccio per sviluppare una tecnica che combina informazioni provenienti da diversi pre-filtri contestuali. L'utilizzo di diversi pre-filtri si basa sul fatto che, come detto, in genere ci possono essere più generalizzazioni del medesimo contesto specifico. Ad esempio, il contesto $c = (Fidanzata, Teatro, Sabato)$ può essere generalizzato a $c1 = (Amico, ovunque, ilSabato)$, oppure in un'altra serie di contesti come ad esempio $c2 = (nondasolo, Teatro, inqualsiasimomento)$.
Seguendo questa idea, Adomavicius [6] utilizza pre-filtri in base al numero

di possibili contesti per ogni voto, e poi unisce le raccomandazioni derivanti da ogni pre-filtro contestuale.
La panoramica generale di questo approccio è illustrato nella Figura 2.4.
Si noti che la combinazione di diversi pre-filtri può essere fatta in diversi modi. Ad esempio, per un dato contesto, si può scegliere la migliore performance di pre-filtering, o utilizzare un insieme di contesti attraverso il pre-filtering.
Una possibilità interessante è quella di poter dividere le varie informazioni contestuali e utilizzarle in diversi momenti, in maniera tale da ottenere risultati differenti nel caso di applicazioni di pre-filtering, post-filtering o contextual modeling. Per esempio, l'informazione contestuale Weekend o Giorno della settimana è più utile come dato rilevante in un pre-filtering, mentre l'informazione climatica Sole o Pioggia è più appropriata in un contesto di post-filtering.

In questo capitolo abbiamo osservato come le informazioni contestuali siano rilevanti nei sistemi di raccomandazione e come sia importante non tralasciarle durante il processo predittivo.
Abbiamo anche illustrato come le informazioni contestuali possano essere utilizzate in diverse fasi del processo di raccomandazione, presentando i tre paradigmi principali e la possibilità di multipli approcci.

Capitolo 3

Strumenti di Analisi Semantica

Durante lo svolgimento della tesi è stato necessario utilizzare diversi strumenti di l'analisi semantica.
In questo capitolo verrano presentati gli algoritmi e gli strumenti utilizzati per effettuare analisi semantica in ambito informatico, con lo scopo di estrarre parole chiave da testi di lunghezza medio-breve.

3.1 Estrazione di parole chiave

Le *keyword* (parole chiave) sono termini che contengono le informazioni più importanti all'interno di un documento.
La loro estrazione automatica consiste nell'individuare un gruppo ristretto di parole in un documento, che ne descrivano in linea generale il contenuto. Per esempio, se considerassimo la trama del film "Titanic" l'estrazioni di parole come "iceberg, incidente, nave" sarebbe più che soddisfacente.
Si nota subito come l'estrazione di keyword sia molto importante quando si devono processare oggetti contenenti del testo. Ogni giorno migliaia di libri e articoli vengono pubblicati online, per cui sono necessarie ottime tecniche di estrazione dei contenuti per poter generare e organizzare le keyword di un determinato documento.
Possiamo quindi notare come questi processi siano alla base dell'information retrieval, ovvero abbiano come obiettivo il riuscire a restituire ad un determinato utente in cerca di determinate informazioni un set di documenti che si suppone siano coerenti con le informazioni ricercate.

I metodi di estrazione automatica delle parole chiave possono essere suddivisi in quattro categorie[41]:

- *Simple Statistics Approach:* questi metodi sono i meno complessi e non necessitano di dataset di training. Le informazioni statistiche di una parola(come per esempio il numero di occorrenze) possono essere utilizzate per identificare parole chiave all'interno di un documento o di un set di documenti.
- *Approccio Linguistico:* questo approccio utilizza le caratteristiche linguistiche delle parole all'interno di frasi e documenti. L'approccio linguistico include l'analisi lessicale, metodi sintattici, ecc ....
- *Machine Learning:* attraverso un approccio di tipo machine learning l'obiettivo è quello di estrarre keyword da dataset di training in maniera tale da definire un modello per poter estrarre parole chiave da nuovi documenti.
- *Altri Approcci:* con altri approcci intendiamo identificare tutti quegli approcci che combinano le tecniche precedenti e altre tecniche di analisi dei testi come la posizione delle parole, la lunghezza delle parole e i tipi di stile utilizzati nel testo(una parola in grassetto ha una rilevanza maggiore rispetto ad una non in grassetto).

La maggior parte degli approcci discussi cerca di estrarre i nomi (propri e comuni) all'interno dei documenti perché è proprio attraversi di essi che si può identificare in maniera più efficace il contesto del documento. La scelta dei nomi (o in alcuni casi di altre parole) è effettuata analizzando le caratteristiche linguistiche [36] e le caratteristiche del testo [11] come per esempio una parte di testo sottolineata. Osservare la posizione della parole è un'altra buona tecnica per identificare candidate parole chiave (per esempio parole che compiano nel titolo del documento).
I metodi descritti precedentemente vengono applicati ad un singolo documento ma nulla vieta di applicarli ad interi set di documenti.
Metodi come le occorrenze multiple [54, 60] e di machine learning [37, 82] sono utilizzate solitamente per estrarre keyword da un solo documento, mentre metodi come il keyword clustering vengono utilizzati su interi set di documenti.

3.2 Metodi di estrazione

In questo paragrafo verrano mostrati brevemente alcuni approcci per la selezione e l'elaborazione delle parole chiave.

3.2.1 I nomi come parole chiave

All'interno di un documento il fulcro informativo risiede per la maggior parte nei nomi, propri o comuni, utilizzati.
Le tecniche per l'estrazione dei nomi richiedono un analizzatore morfologico e un vocabolario con le relative regole grammaticali della lingua presa in considerazione.
Dal momento della loro estrazione i nomi divengono dei possibili candidati a diventare parole chiave.

Come primo passo vengono identificate le frasi nominali all'interno del

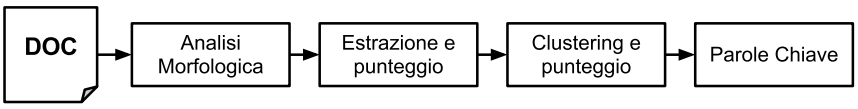

Figura 3.1: Estrazione di nomi come parole chiave

documento ovvero quelle frasi che hanno un nome o un pronome come parola iniziale (anche se aggregata ad aggettivi). A questo punto si associa un punteggio ad ogni frase nominale, le frasi vengono raggruppate in cluster e viene dato un nuovo punteggio. La più corte frasi nominali del cluster con punteggio più alto sono candidate ad avere parole chiave.
Di seguito verranno mostrati brevemente i passaggi di questo procedimento:

1. Analisi Morfologica
    - Word Segmentation: le parole vengono separate l'una dell'altra identificando e filtrando la punteggiatura;
    - Part of Speech Tagging (POS): viene effettuata l'analisi grammaticale e viene associata ad ogni parola uno delle nove parti del discorso (verbo, aggettivo, sostantivo, pronome, articolo, preposizione, avverbio, interiezione e congiunzione);
    - Stemming: lo stemming è il processo di riduzione della forma flessa di una parola alla sua forma radice, detta tema. Il tema non corrisponde necessariamente alla radice morfologica (lemma) della parola: normalmente è sufficiente che le parole correlate siano mappate allo stesso tema (ad esempio, che andare, andai, andò mappino al tema and), anche se quest'ultimo non è una valida radice per la parola.
2. Estrazione frasi nominali e punteggio
    - Estrazione frasi nominali

- Rimozione Stopword: in questo processo vengono eliminate tutte le parole della lingua in esame di poco valore a livello di contenuti (congiunzioni, articoli, ecc. . .)
- Attribuzione di un punteggio: vengono attribuiti dei punteggio alla frasi estratte secondo algoritmi noti [4]

3. Clustering

   - Raccolta: le parole uguali vengono raccolte in cluster
   - Punteggio: viene attribuito un punteggio ad ogni cluster
   - Valutazione: vengono valutati i punteggi precedentemente dati

4. Scelta delle parole chiave

   - Le frasi nominali più brevi facenti parte dei cluster con punteggio più alto vengono selezionate come parole chiave

3.2.2 Term Frequency-Inverse Document Frequency

Attraverso il metodo TF-IDF possiamo valutare il peso, ovvero l'importanza, di una parola all'interno di una collezione di documenti.
L'importanza della parola cresce proporzionalmente con il numero delle occorrenze in un documento ma diminuisce se le occorrenze aumentano all'interno di tutta la collezione [49]. Consideriamo il termine T_i come la parola T-iesima nel documento D_j.
Le frequenza del termine è:

$$TF_{i,j} = \frac{n_{i,j}}{\sum_{k=1}^{n} n_{k,j}}$$

dove $n_{i,j}$ è il numero di occorrenze del termine T_i considerato nel documento D_j e il denominatore è la somma delle occorrenze di tutti i termini all'interno del documento D_j.
L'inverse document frequency [71] è una misura utilizzata per calcolare l'importanza del termine all'interno di un set di documenti.
La formule del termine è:

$$IDF_i = \frac{log\,|D|}{nD_i}$$

dove D è il numero totale dei documenti del set preso in considerazione e il denominatore è il numero dei documenti dove compare il termine T_i. A questo punto possiamo ricavare il peso della parola con:

$$TFIDF_i = TF_{i,j} \times IDF_i$$

Come si può notare la limitazione di questo metodo non permette di fare stime sul singolo documento ed inoltre non considera il tipo grammaticale delle parole prese in considerazione.

3.2.3 Selezione tramite informazioni testuali

Le parole all'interno di un documento possono essere scritte in varie forme che forniscono informazioni addizionali riguardo alla loro importanza [11].
Vi sono vari tipi caratteristiche:

- Enfatizzazione della parola con grassetto, corsivo o sottolineatura
- Parole maiuscole
- Grandezza del font utilizzato
- Normalize Sentence Length, che rappresenta la percentuale del numero di occorrenze di una parola in una frase rispetto al numero di occorrenze di parole nella frase più lunga del documento.
- Parole comprese in frasi (Cue-Phrases) che iniziano con parole significative, come "in particolare", "in conclusione" ...

3.2.4 Algoritmo position weight

Una parola in differenti posizioni può avere pesi diversi all'interno di un documento. Per esempio apparire nell'introduzione potrebbe contare di più che apparire nel corpo del testo, oppure apparire nel titolo potrebbe identificare il topic del documento.
Questo algoritmo utilizza principalmente tre elementi:

- Peso del paragrafo
- Peso della frase
- Peso della parola

Se il paragrafo(p_k) coincide con il titolo, il sottotitolo, l'introduzione o la conclusione ha più valore di un paragrafo del corpo del testo.
La prima e l'ultima frase(s_k) valgono più delle frasi che iniziano con "Per esempio" (che hanno quasi valore zero).
Inoltre le parole che contengono informazioni testuali (viste precedentemente) pesano più delle parole comuni.
Il peso di un termine w (w_r) all'interno di un documento quindi corrisponde

alla somma dei pesi di tutte le posizioni in cui appare.

$$Pw(t_i) = pw(t_i, p_j) \times pw(t_i, s_k) \times pw(t_i, w_r)$$

Di seguito verranno presentati due strumenti utilizzati nel lavoro di tesi che implementano alcuni dei metodi appena elencati.

3.3 Stanford NLP

Lo *Stanford Natural Language Processing* è un lavoro svolto da un gruppo di persone dell'università di Stanford che comprende ricercatori, dottorandi, docenti e studenti [3].
Il lavoro di ricerca è focalizzato sullo sviluppo di algoritmi che permettano ai calcolatori di processare e comprendere il linguaggio umano.

3.3.1 Aree Tematiche

Di seguito verranno presentate le aree tematiche coperte dal progetto.

Studi semantici

Principalmente si tratta dell'estrazione e dell'identificazione delle parole di un testo.
La ricerca include i seguenti argomenti specifici:

- Identificazione di coreferenze: il suo obiettivo è quello di identificare all'interno di un testo tutte quelle espressioni che si riferiscono ad una stessa identità.

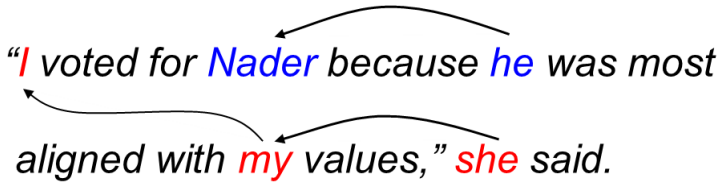

Figura 3.2: Esempio di identificazione di coreferenze

- Progetto Natlog: l'obiettivo di questo progetto è lo sviluppo di un approccio al linguaggio naturale basato su un modello di inferenze di natura logica, il quale identifica le inferenze dalle loro caratteristiche lessicali e sintattiche senza la necessità una completa interpretazione semantica.

- NER e IE: la NER (Named Entity Recognition) e la IE (information Extraction) hanno come obiettivo l'identificazione dei sostantivi e l'associazione delle loro entità (persona, luogo, organizzazione).

Estrazione dello stato d'animo

L'obiettivo è l'identificazione automatica dello stato d'animo dello scrittore. Un esempio applicativo potrebbe essere, data una lettera, identificare se essa è per la famiglia, un amico o una fidanzata.

Deep Learning in NPL

L'obiettivo è lo studio di testi attraverso i quali identificare la natura ricorsiva dei linguaggi naturali.

Probabilistic Parsing

Utilizzando algoritmi dinamici vengono parsate tutte le frasi di un documento e, con l'aiuto di un vocabolario sintattico, vengono identificate tutte le parti del discorso del testo. Al momento stanford offre modelli per l'analisi di testi in inglese, cinese arabo e tedesco.

Machine Translation

L'obiettivo del Machine Translation consiste nel convertire automaticamente un linguaggio naturale in un altro, mantenendo il significato del testo in input e producendo un testo in uscita coerente e fluente. Principalmente si sta cercando di sviluppare un software per la produzione di testi in inglese avendo come input testi orientali.

Dialog and Speech Processing

L'obiettivo è quello di creare utenze robotiche per la realizzazione di dialoghi tra utenti e macchine.

3.3.2 Software

Lo Stanford NLP Group mette gratuitamente a disposizione i suoi tools, disponibili online all'indirizzo http://nlp.stanford.edu/software/index.shtml. Questi pacchetti software offrono tutti gli strumenti statistici necessari per poter effettuare delle analisi semantiche su testi e documenti. Tutti i software sono scritti in java. Di seguito verrà illustrato il pacchetto scaricato mentre il suo utilizzo verrà affrontato nel Capitolo 5.

Lo Stanford CoreNPL è il cuore dell'applicazione attraverso di esso è possibile processare, scaricando il vocabolario associato, tutti i testi scritti in lingua inglese.
Al suo interno troviamo un tagger di part of speech per l'individuazione delle parti del discorso, un Named Entity recognizer per l'identificazione di nomi di compagnie, nomi di persone, nomi di organizzazioni e nomi di ricorrerenze e un sistema di risoluzione delle coreferenze.
Dati questi strumenti l'obiettivo del lavoro di tesi può essere riassunto nel seguente esempio: data una frase del tipo:

"Barack Hussein Obama is the 44th and current
President of the United States, the first African American to hold the office".

per prima cosa verrano identificate le parti del discorso 3.3
e in seguito verrano estratte le entità che decideremo di filtrare 3.4

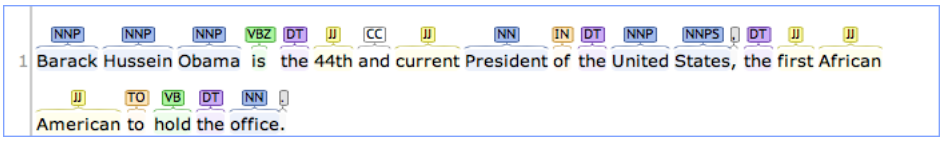

Figura 3.3: Part of Speech

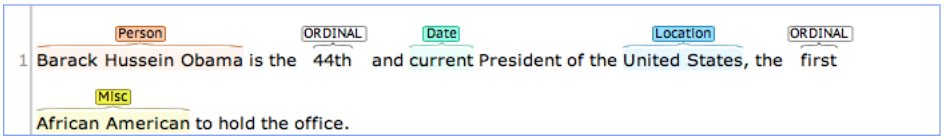

Figura 3.4: Named Entity Recognition

3.4 Stanbol enhancer

La suite di software di Stanbol offre una moltitudine di strumenti per la gestione di contenuti semantici[2]. Tra le sue applicazioni più diffuse troviamo l'estrazione di informazioni da pagine web, il completamento automatico nelle barre di ricerca e l'identificazione di topic all'interno di mail per il loro reindirizzamento.
In questo lavoro di tesi viene utilizzato il servizio API proposto dal pacchetto Stanbol Enhancer il cui obiettivo è quello di estrarre le entità dai testi e tramite dbpedia (una base di dati semantica) arricchirne il contenuto. Per

essere più precisi il testo passato viene processato da un *Enhancement Engine* e in seguito definito da un *Enhanchement Chain*. La Figura 3.5 offre una panoramica su tale processo:

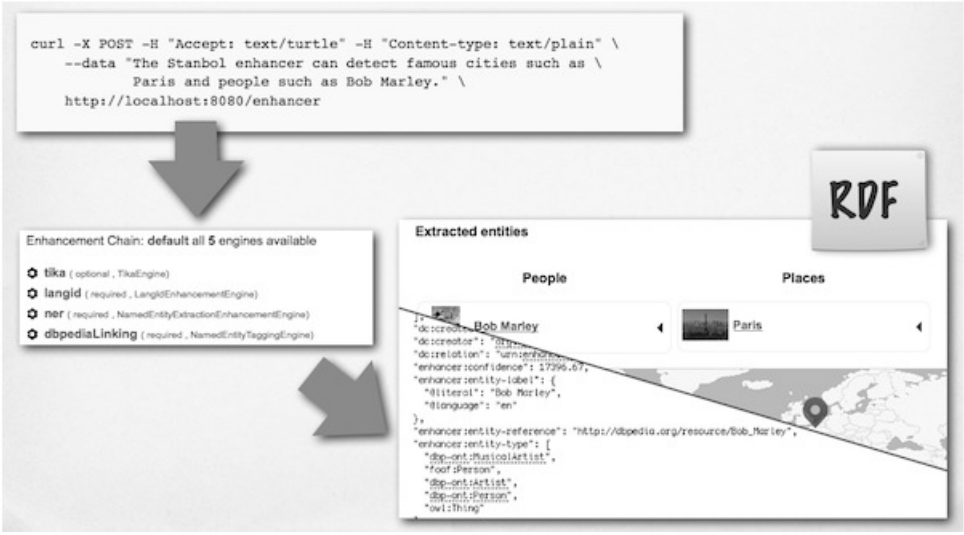

Figura 3.5: Schema di una chiamata a Stanbol

La chiamata, che verrà trattata successivamente, passa il testo al sistema che viene parsato ed laborato tramite diversi engine per poi produrre un documento (per la tesi è stato selezionato un formato JSON) nel quale è possibile visualizzare i risultati.
Durante la chiamata (fig. 3.6) all'API vengono passati diversi parametri: il tipo di richiesta http (Post o Get), il tipo di testo passato e il tipo di formato in output. Dopo che il testo è stato passato all'applicazione viene

```
curl -X POST -H "Accept: text/turtle" -H "Content-type: text/plain" \
--data "The Stanbol enhancer can detect famous cities such as Paris \
and people such as Bob Marley." http://localhost:8080/engines
```

Figura 3.6: Schema di una chiamata a Stanbol

parsato ed elaborato dai diversi engine. Il compito svolto dagli engine include l'estrazione del testo, l'estrazione dei metadati e il rilevamento della lingua come primo passo.
In seguito vengono analizzato i contenuti, in questa categoria vengono compresi tutti quegli engine che hanno a che fare con il Natural Language Processing (NLP).

La terza categoria consiste in engine che hanno il compito di arricchire le entità estratte con nuovo contenuto semantico, ad esempio collegando l'entità estratta a basi di dati semantiche. Infine, i motori post elaborazione possono essere utilizzati per filtrare i dati, regolare classifiche o fare altri tipi di trasformazione sui risultati ottenuti.
Di seguito verranno elencati e descritti brevemente i motori utilizzati:

- *Tika Engine:* è il motore principale si occupa dell'identificazione del testo, della sua estrazione e dell'estrazione dei metadati dai vari tipi di formati di documento;
- *Language Detection Engine:* identifica la lingua del documento e si appoggia ai corrispettivi vocabolari;
- *OpenNLP Sentence Detection Engine:* il motore si appoggia alle librerie OpenNLP. OpenNLP è un insieme di tools utilizzati per processare testi in linguaggio naturale. Questo engine si base sull'identificazione e segmentizzazioni delle frasi all'interno del testo;
- *OpenNLP Tokenizer Detection Engine:* anch'esso si appoggia alle libreria OPENNLP e si occupa del'identificazione delle parole all'interno delle frasi;
- *OpenNLP POS Tagging Engine:* identifica le parti del discorso all'interno del testo;
- *OpenNLP NER Engine:* si occupa dell'identificazione delle entità all'interno del testo;
- *Named Entity Linking Engine:* questo engine si occupa di arricchire semanticamente le entità estratte. Collegamendosi a basi semantiche, come per esempio Dbpedia, aggiunge informazioni come il tipo di entità e il gruppo di appartenenza.

In Figura 3.7 è illustrato un esempio di riconoscimento dell'entità e arricchimento di essa attraverso dbpedia, possiamo vedere come nell'entità "Bob Marley" vengano identificate le ontologie "cantante" e "persona" ed in seguito dopo l'individuazione della sorgente dbpedia ci sia un ulteriore arricchimento semantico.
Dopo aver definito il formato che l'api dovrà restituire, nel nostro caso Json, avremo l'elenco di tutte le entità estratte. In Figura 3.8 abbiamo un pezzo di codice json estratto dall'output ricevuto.
Alla voce "dc:creator" viene identificato l'engine utilizzato per l'identificazione dell'entità, che verrà utilizzato da parte nostra come identificatore di

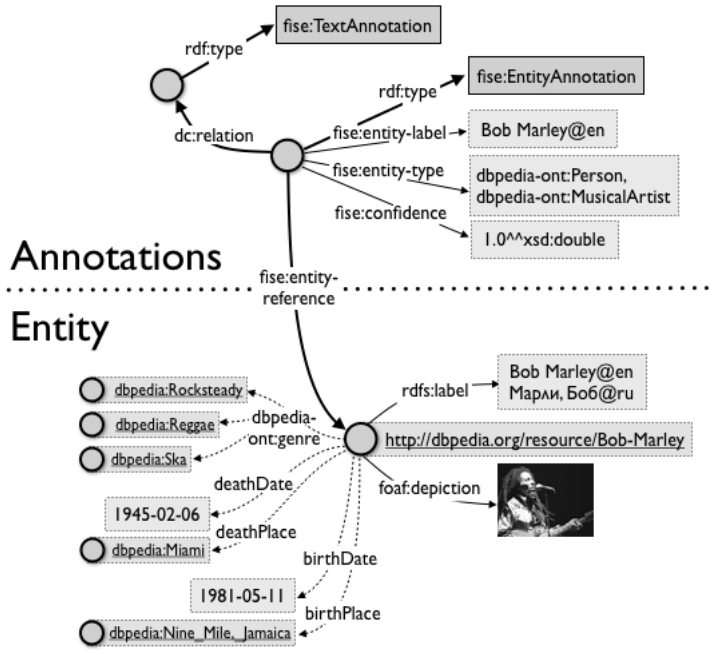

Figura 3.7: Risultato del process Stanbol Enhancement


```
{
  "@id": "urn:enhancement-18f1484a-62c0-6245-892b-03d51da9d5b4",
  "@type": [
    "enhancer:Enhancement",
    "enhancer:EntityAnnotation"
  ],
  "dc:created": "2013-08-17T21:01:49.290Z",
  "dc:creator": "org.apache.stanbol.enhancer.engines.entitylinking.engine.EntityLinkingEngine",
  "dc:relation": "urn:enhancement-1a2862cb-6fcc-7c01-6b16-8d6ef227a18e",
  "enhancer:confidence": 1.0,
  "enhancer:entity-label": {
    "@language": "en",
    "@value": "Barack Obama"
  },
  "enhancer:entity-reference": "http://dbpedia.org/resource/Barack_Obama",
  "enhancer:entity-type": [
    "dbp-ont:Agent",
    "dbp-ont:OfficeHolder",
    "dbp-ont:Person",
    "schema:Person",
    "owl:Thing",
    "foaf:Person"
  ],
  "enhancer:extracted-from": "urn:content-item-sha1-81b69e172aec7ba379e94017d682f7bb25b7d041",
  "entityhub:site": "dbpedia"
},
```


Figura 3.8: Esempio di output Json

identità all'interno del file json. Dopo aver individuato l'engine responsabile dell'estrazione alla voce "enhancer:entity-label" viene identificata l'entità e la sua lingua, nel nostro caso "Barack Obama" rappresenta l'entità e "en" la lingua.
Nella seconda parte del json possiamo vedere l'arricchimento semantico: come prima cosa viene individuata l'indirizzo sorgente dell'entità su dbpedia e in seguito vengono elencate tutte le entità ontologiche dell'entità dalla più specifica alla più globale.

In questo capitolo abbiamo introdotto le nozioni di analisi semantica necessaria a capire il funzionamento di estrazione delle entità da testi in linguaggio naturale e presentato i due strumenti utilizzati per lo sviluppo. Va evidenziato che attraverso Stanbol sono state solo effettuate delle chiamate e ricevuti dei risultati, invece per quando riguarda i tools di stanford sono stati scaricati i pacchetti java e sono stati integrati nello sviluppo di codice svolto durante il lavoro di tesi.

Capitolo 4

Tassonomia del Contesto nel dominio "Film"

Come visto nel Capitolo 2, l'introduzione della dimensione contestuale nel processo di raccomandazione porta ad avere risultati diversi in base ai diversi input contestuali.
Questo porta sicuramente ad affinare la tecnica di raccomandazione rendendola maggiormente performante in base alle circostanze. Non bisogna però dimenticare che la scelta dei filtri contestuali è di fondamentale importanza per la buona riuscita della raccomandazione. La scelta del periodo dell'anno per la raccomandazione di una vacanza è sicuramente un buon filtro contestuale, mentre il tempo atmosferico per la visione di un film non è un buon filtro.
L'obiettivo di questo capitolo è quello di presentare e analizzare un insieme di contesti relativi al nostro dominio di studio, ovvero la raccomandazioni di film.

4.1 Analisi del Contesto

Il primo passo nello svolgimento del lavoro di tesi è stato quello di effettuare un'analisi preliminare per identificare il maggior numero possibile di contesti [8], in seguito si è cercato di adattarli al nostro dominio con dei voti di utilità (ovvero l'utilità del contesto ai fini della raccomandazione) e di osservabilità (ovvero la difficoltà di realizzazione della raccomandazione).
Da questa prima analisi sono state individuate 4 famiglie di contesto (physical, interactive media, social, modal) che adesso verrano affrontate in maniera approfondita.

Di seguito vengono analizzati tutte le categorie di contesto individuate, discutendo i valori di utilità e osservabilità, definendo degli Use Case, elencando i possibili metodi di individuazione del contesto e per concludere evidenziando le varie tecniche di raccomandazione possibili.
Per ogni contesto viene descritto un esempio significativo di Use Case, ovvero una situazione reale a cui si potrebbe applicare una raccomandazione, e una sorgente, ovvero l'insieme di strumenti necessari per effettuare quel determinato tipo di raccomandazione.

4.1.1 Physical Context

Per *Physical Context* si intendono tutte quelle tipologie di contesto che riguardano l'ambiente circostante e l'interazione tra l'utente ed esso. Per questo vengono individuati due sottoinsiemi contestuali l'*environment* (ambiente) e lo *user*.
Nella tabella 4.1 vengono illustrati i tipi di contesto selezionati per queste tipologie, attribuendo a ciascuno una misura di utilità(Util.) e osservabilità(Oss.) rappresentata da un valore numerico su una scala da 1 a 5.

Categoria	**Tipo**	**Contesto**	**Esempi**	**Util.**	**Oss.**
Physical	Environment	Posizione geografica	Milano-Roma	3	5
		Luogo	Casa-Cinema	5	5
		Tempo atmosferico	Sole-Pioggia Caldo-Freddo	0	4
		Periodo temporale	9.00-18.00 notte-giorno estate-inverno	5	5
	User	Attività utente	Fare jogging Guardare la TV Riposare	3	2

Tabella 4.1: Physical Context

Posizione geografica

Per posizione geografica si intende il luogo dove l'utente si trova nel momento in cui viene effettuata la raccomandazione e non il luogo da cui proviene,

in questa maniera cerchiamo di separare il contesto dallo user profile(dove risiedono le informazioni anagrafiche).

- **Use Case**: un esempio di Use Case è la raccomandazione di film che sono attualmente disponibili in una particolare zona geografica. Ad esempio, considerando un catalogo di film su scala internazionale, questo tipo di raccomandazione può tornare utile perché permette di raccomandare item che siano effettivamente disponibili nella zona in cui l'utente si trova in quel momento. Considerando contesti più ristretti (per esempi luoghi della stessa nazione), una raccomandazione basata sulla posizione geografica potrebbe risultare meno significativa. Per questo motivo l'utilità di questo contesto è stata valutata pari a 3.

- **Sorgente:** le informazioni sul contesto necessarie per effettuare questo tipo di raccomandazione possono essere ricavate facilmente tramite l'utilizzo di dispositivi gps o triangolazioni di posizione appoggiandosi a celle wifi. A causa della facilità di realizzazione l'osservabilità è stata valuta 5.

Di seguito sono mostrati due esempi di raccomandazione (si noti che sono possibili più interpretazioni e qui vengono inseriti solo due esempi significativi):

- **pre-filtering:** vengono considerati solo i rating dei film proiettati (al cinema o in tv) nella location a me più vicina.

- **post-filtering:** vengono inseriti in una classifica i film "a me più vicini" suddivisi, per esempio, per genere.

Luogo

Per luogo intendiamo la struttura dove visionare il film per esempio casa o cinema.

- **Use Case:** in base alla posizione dell'utente proponiamo film da poter vedere nei cinema vicini o quelli da poter vedere a casa. Come è facilmente intuibile questa caratterizzazione contestuale è particolarmente significativa, per questo motivo l'utilità di questa categoria è stata valutata 5.

- **Sorgente:** le sorgenti sono analoghe al caso precedente in quanto l'obiettivo è sempre l'individuazione di una posizione.

Gli esempi di raccomandazione sono riconducibili a quelli elencati per la categoria "Luoghi Geografici".

Tempo Atmosferico

Il contesto tempo atmosferico tiene conto della temperatura e di tutti i fenomeni atmosferici del luogo in cui si trova l'utente al momento della raccomandazione.

- **Use Case:** in questo caso non è stato possibile trovare uno Use Case significativo, in quanto il tempo atmosferico non influisce quasi per nulla sulla scelta di vedere un determinato film, proprio per questo è stato assegnato 0 come voto all'utilità.
- **Sorgente:** l'osservabilità ha ricevuto come voto 4, in quanto è sostanzialmente uguale alle precedenti ma va integrata con siti meteorologici per l'identificazione delle condizioni atmosferiche.

Come possiamo vedere una utilità di valore 0 viene scartata subito nel processo di definizione dei possibili contesti per il sistema di raccomandazione.

Periodo Temporale

Con periodo temporale intendiamo qualsiasi periodo di tempo: dal singolo orario, come per esempio le 19.00, a periodi più lunghi come giorni, settimane o perfino stagioni.
L'informazione temporale può anche essere utilizzata come informazione per definire il profilo utente, ovvero può identificare gli utenti che usufruiscono del servizio in base alla fascia di tempo di utilizzo, per esempio la mattina casalinghe, pomeriggio bambini e sera famiglie.

- **Use Case:** gli Use Case in questa categoria sono molti e tutti validi, un esempio potrebbe essere la proposizione di film differenti nel weekend rispetto ai giorni lavorativi della settimana. La validità della categoria è evidenziata dal voto ricevuto pari a 5.
- **Sorgente:** possiamo individuare il momento esatto in cui viene effettuata la raccomandazione attraverso un timestamp. Per la semplicità di realizzazione è stato assegnato voto 5 all'osservabilità.

Di seguito sono mostrati due esempi di raccomandazione (si noti che sono possibili più interpretazioni e qui vengono inseriti solo due esempi significativi):

- **pre-filtering:** durante la raccomandazione pre-filtering consideriamo solo i rating di item nella fascia oraria presa in considerazione.

- **post-filtering:** push up nella classifica di item aventi un certo genere.

Dopo aver analizzato i contesti relativi al sottoinsieme dell'environment, passiamo al successivo sottoinsieme che riguarda l'utente e l'interazione con l'ambiente.

User Activities

Per User Activities intendiamo tutte quelle attività che si possono svolgere durante una giornata, possiamo intendere una seduta di jogging, il rilassarsi guardando la tv, il riposare sul divano o anche il lavorare.

- **Use Case:** non sono molti gli Use case rilevabili in questo contesto (decisamente più utile in un dominio musicale) per questo il voto attribuito è stato 3. Mostriamo comunque un esempio di raccomandazione: immaginando un viaggio di lavoro in aereo possiamo proporre film che abbiano una durata inferiore alla durata del volo, oppure che abbiano un nesso con la località di arrivo.

- **Sorgente:** in questo caso è molto difficile individuare in maniera implicita il contesto per questo è stato attribuito al campo osservabilità il voto 2.

4.1.2 Interaction Media

Questa categoria si occupa di individuare contesti differenti in base al tipo di media presi in considerazione e dei device attraverso i quali avviene la loro distribuzione.
Nel nostro caso ci occuperemo solo della categoria device in quanto il nostro studio si occupa solo dei media del dominio "Film".

Categoria	Tipo	Contesto	Esempi	Util.	Oss.
Interaction Media	Device	Type of Device	Laptop Smartphone Smart TV	5	5

Tabella 4.2: Interaction Media Context

Type of Device

Con Type of Device si cerca di proporre il contenuto più idoneo al tipo di dispositivo in possesso dell'utente, per esempio sarebbe inutile proporre un contenuto ad alta definizione ad un utente che si trova a navigare con il suo telefonino appoggiandosi ad una rete telefonica.

- **Use Case:** come appena evidenziato la maggiore parte degli Use Case di questa categoria si focalizzeranno sulla scelta del miglior contenuto in base alle specifiche dei dispositivi dell'utente attraverso il quale verranno effettuate le raccomandazioni. Un possibile esempio è quello di proporre contenuti in diverse definizioni in base al device. (Una raccomandazione per una tv Full HD proporrà film a 1080p). L'ottimizzazione dei contenuti è di fondamentale importanza per una migliore qualità del servizio per questo è stato dato come voto 5 all'Utilità.

- **Sorgente:** la tipologia del dispositivo viene identificata duranta la comunicazione con il server, per la semplicità di realizzazione il voto assegnato all'osservabilità è 5.

Di seguito sono mostrati due esempi di raccomandazione (si noti che sono possibili più interpretazioni e qui vengono inseriti solo due esempi significativi):

- **pre-filtering:** considero i rating rilasciati solo dagli utenti che utilizzano lo stesso device dello user.

- **post-filtering:** push-Filtraggio di item aventi una certa risoluzione compatibile con il device corrente.

4.1.3 Social

In questa categoria si è cercato di individuare tutti quei contesti che hanno a che fare con il sociale.
Sono state individuate due categorie: della prima fanno parte le interazioni dell'utente con le persone che lo circondano, nella seconda, invece, vengono raccolti i fattori dell'ambiente esterno che possono influenzare le scelte l'utente.

Categoria	Tipo	Contesto	Esempi	Util.	Oss.
Social	Internal Factors	Persone con cui l'utente interagisce	Da solo Con amici Con famiglia	4	1
	External Factors	Ultimi Trend	Evento ricorrente o importante	5	3
		Coming Soon	Film in uscita	5	5

Tabella 4.3: Social Context

Persone con cui l'utente interagisce

In questa categoria viene sottolineata l'importanza del gruppo di utenza che sta usufruendo del servizio.
Un utente in base al tipo di compagnia al momento della raccomandazione potrebbe preferire guardare una certa tipologia di film.

- **Use Case:** definito un gruppo di utenza propongo film di generi differenti, per esempio ad una coppia potrebbero venir proposti film romantici, ad un gruppo di amici film d'azione e ad una famiglia con dei bambini film d'animazione.
  Il voto attribuito all'utilità è 4 perché vengono trovati degli use case significativi, non raggiunge il voto massimo perché non sempre possiamo individuare in maniera univoca i gusti di una certa tipologie di utenti (non tutte le coppie potrebbero gradire di vedere film romantici).

- **Sorgente:** in questo caso l'osservabilità, definita non in maniera esplicita, riceve un voto molto basso in quanto sarebbero necessari strumenti complessi per rilevare in maniera implicita il contesto(hardware e software per il riconoscimento facciale).

Di seguito sono mostrati due esempi di raccomandazione (si noti che sono possibili più interpretazioni e qui vengono inseriti solo due esempi significativi):

- **pre-filtering:** considero i rating rilasciati da gruppi di utenza uguali al gruppo che sta effettuando la raccomandazione.

- **post-filtering:** business Rules predefinite per fare push di item (esempio film romantici) in base alla presenza di altri utenti (esempio partner).

Adesso verranno illustrati le due categorie contestuali relative ai fattori esterni.

Ultimi trend

Gli ultimi trend sono i temi più recenti e più trattati dagli utenti. Nell'ultimo decennio con la dilagante diffusione di internet e dei social network l'individuazione di trend è diventata di fondamentale importanza, al punto che sono nate società che si occupano di monitorare la rete per identificare le mode del momento.

- **Use Case:** in questa categoria vi sono vari e utili use case per questo viene assegnato voto 5 alla Osservabilità. Un esempio potrebbe sere il seguente: considerata una notizia di cronaca che illustra il passaggio di un'asteroide vicino al pianeta terra, potrebbero venir proposti film di fantascienza o catastrofici. Questo tipo di raccomandazione può essere utilizzato anche nel caso di eventi ricorrenti o di particolare importanza.

- **Sorgente:** l'analisi degli ultimi trend non risulta particolarmente facile da implementare, per questo il voto assegnato all'osservabilità è 3. Il web è pieno di fonti da quali attingere informazioni (social network o testate giornalistiche) ma lo loro analisi risulta essere difficile e vi è la necessita di appoggiarsi a strumenti di analisi semantica per l'estrazione delle keywork che vanno ad identificare i topic del momento.

Un esempio di raccomandazione post-filtering per questa categoria contestuale è il push di item che sono associati al trend analizzato.

Coming Soon

Dato un insieme di film in uscita, l'obiettivo di questa raccomandazione contestuale è quello di riuscire a sponsorizzare item che abbiano delle similitudini con quelli al momento presenti nelle sale. Per similitudine intendiamo film delle stesso genere, oppure che abbiano un attore famoso in comune oppure che facciano riferimento alla stessa serie (Alien, Alien 2, ecc ...).

- **Use Case:** anche in questo caso abbiamo degli use case significativi e per questo il voto assegnato è pari a 5. Se per esempio osservassimo il calendario delle uscite e trovassimo un nuovo film di Batman, il

sistema dovrebbe proporre i vecchi film di Batman oltre ad alcuni film sui supereroi.

- **Sorgente:** andando ad analizzare siti di informazione cinematografica è facile individuare i film in uscita e tutte le loro informazione, di conseguenza è altrettanto facile identificare film simili. Per questi motivi il voto assegnato all'Osservabilità è 5.

4.1.4 Modal

In questa categoria sono stati inseriti quei contesti relativi allo stato dell'utente e le sue attività.
Sono state individuati anche in questo caso due sottoinsiemi: nel primo abbiamo contesti definiti dalle caratteristiche dell'utente (stato d'animo, capacità, intenti) mentre nel secondo gruppo definiamo un tipo di contesto in base a quello che l'utente fa(attività recenti, navigazione internet).

Categoria	Tipo	Contesto	Esempi	Util.	Oss.
Modal	Subjective	Stato d'animo	Felice Triste	4	0
		Esperienze e capacità dell'utente	Esperto di cinema	2	0
		Scopo	Ricerca e studio	0	2
	Objective	Attività recenti	Cronologia	5	5

Tabella 4.4: Modal Context

Stato d'animo

L'inserimento di questa categoria nell'elenco dei contesti, nasce dall'idea che lo stato d'animo di un utente posso influenzare il tipo di film da raccomandare.

- **Use Case:** in questo caso vengono proposti film che siamo in un qualche modo "compatibili" con lo stato d'animo dell'utente al momento della raccomandazione. L'idea alla base di questa categoria è molto valida per questo viene assegnato un voto pari a 4 all'utilità.

- **Sorgente:** è praticamente impossibile riuscire ad individuare in maniera implicita lo stato d'animo di un utente, per questo è stato assegnato 0 come voto all'osservabilità.

Di seguito sono mostrati due esempi di raccomandazione (si noti che sono possibili più interpretazioni e qui vengono inseriti solo due esempi significativi):

- **pre-filtering:** considero i rating rilasciati da persone con lo stesso stato d'animo dell'utente.
- **post-filtering:** push di item aventi un certo genere.

Esperienze e capacità dell'utente

Lo scopo di questa categoria, che può essere adattata a più domini, è quello di fornire item agli utenti in base al loro background conoscitivo. Questo genere di informazioni di solito possono essere trovate o dedotte dal profilo utente.

- **Use Case:** ad un esperto di cinema suggerisco film di nicchia piuttosto che film commerciali. Come detto precedentemente questo tipo di contesto si può utilizzare in più domini e il dominio film non è uno di quelli giù significativi, per questo il voto assegnato corrisponde a 2.
- **Sorgente:** in questo caso come il precedente è impossibile ricavare in maniera implicita le informazioni contestuali per questo il voto assegnato è 0.

Di seguito sono mostrati due esempi di raccomandazione (si noti che sono possibili più interpretazioni e qui vengono inseriti solo due esempi significativi):

- **pre-filtering:** considero solo utenti "esperti" per generare il modello oppure escludo item popolari dal modello.
- **post-filtering:** push di item particolari rispetto a quelli commerciali.

Ricerca e Studio

Questo tipo di categoria contestuale non è compatibile con il dominio considerato.
Nel caso prendessimo in considerazione un altro dominio, per esempio i libri di una biblioteca, potrebbe aumentare il suo valore di osservabilità. Infatti nel caso dovessimo suggerire dei testi per una ricerca, il risultato produrrà item differenti nel caso le ricerche siano per un progetto di una scuola elementare o per una tesi di laurea.

Attività recenti

In questa sezione vengono prese in considerazione tutte le attività recenti dell'utente, secondo il criterio per cui se un utente cerca e legge qualche informazione sulla rete allora ne è interessato.

- **Use Case:** in questa categoria abbiamo use case significativi, per questo viene assegnato il voto 5. Nel caso di attività recenti possiamo proporre un film simile ad uno appena visto che è piaciuto all'utente oppure analizzando la sua navigazione all'interno della nostra piattaforma, possiamo suggerire film che siano inerenti ai topic a cui si è interessato.

- **Sorgente:** in questo caso è possibile analizzare la cronologia dell'utente sulla piattaforma. Avendo tutte le informazioni disponibili l'osservabilità riceve come voto 5.

In questo capitolo sono stati elencati i vari contesti possibili nel dominio film, dopo un attenta analisi è stato scelto di sviluppare una piattaforma che sia in grado di filtrare contenuti in base all'analisi contestuale delle *attività recenti*.
Verranno analizzate delle news, corrispondenti alla cronologia recente dell'utente. Da queste news verranno estratte delle parole chiave che ne identificheranno i contenuti, a questo punto queste parole chiave verranno messe in relazione con il catalogo di film a nostra disposizione per individuare delle corrispondenze significative. Il tutto verrà analizzato dettagliatamente nel capitolo successivo.
Il capitolo si conclude con un riepilogo dei contesti individuati nelle due tabelle seguenti.

Category	Type	Context	Examples	IDENTIFICATION OF RELEVANT CONTEXT VARIABLES		CONTEXT ELICITATION		CONTEXTUALIZED RECOMMENDATIONS	
				Usefulness	Use case	Osservability	Source	Pre-filtering	Post-filtering
Physical	Environment	location	Milan/Rome	3	Propongo film nei cinema vicini alla posizione dell'utente	5	Dispositivi dotati di gps, identificazione della cella wifi.	Considero solo rating della location a me più vicina.	Classifica con i film "a me più vicini" suddivisi, per esempio, per genere
			home/theatre	5	In base alla posizione dell'utente, proponiamo film al cinema o film da poter vedere a casa.	5	Dispositivi dotati di gps, identificazione della cella wifi. Va integrata con poi (points of interest)	Considero solo rating della location a me più vicina.	Classifica con i film "a me più vicini" suddivisi, per esempio, per genere
		weather, light, temperature	sunny/rainy/cloudy, hot/cold	0	---	4	Dopo aver rilevato la posizione con gps o celle wifi, invio le coordinate ad un sito di meteorologia per ricavare il tempo.	---	---
		time	9pm, night, summer	5	Nel week-end si vogliono proporre film differenti rispetto ai workday	5	timestamp della richiesta	Considerare solo rating nella fascia oraria relativa	Push di item aventi un certo genere
	User	user activity	jogging/watching TV, working/relaxing	3	Sono in viaggio, consigliare film che: ►durino meno del viaggio ►abbiano un nesso con il viaggio (es., film camper se sto viaggiando in camper)	2	utente informa il sistema Dispositivi come cellulari o televisori ci possono dare informazioni sulle attività dell'utenza	Considero solo i rating dei film con durata inferiore alla durata del viaggio.	---
Interaction media		device	tablet, smartTV, laptop	5	In base al device usato vengono proposti contenuti in diversa definizione. Esempio: televisore Full HD film in 1080p	5	Nella comunicazione con il server viene identificato il dispositivo (caller)	Considero i rating solo per il dispositivo corrente	Push/Filtraggio di item aventi una risoluzione compatibile con il mio device corrente.
		type of media	book, movie						

Figura 4.1: Contesti individuati

Category	Type	Context	Examples	IDENTIFICATION OF RELEVANT CONTEXT VARIABLES		CONTEXT ELICITATION		CONTEXTUALIZED RECOMMENDATIONS	
				Usefulness	Use case	Osservability	Source	Pre-filtering	Post-filtering
Social	internal factors	presence/role of other people around user	alone, with friends, with partner	4	In base ai gruppi di utenza propongo film differenti. Esempio coppia film romantici, gruppo di amici film di azione o demenziali.	1	► utente informa il sistema ► utilizzo di software e hardware per il riconoscimento facciale (implementazione complessa)	Considero i rating solo per il gruppo corrente.	Business Rules predefinite per fare push di item (es., romantici) in base alla presenza di altri utenti (es. partner)
	external factors	latest trend	Satellite launch, event recurrence	5	►Un asteroide sta passando vicino alla terra, propongo film su asteroidi e catastrofici. ►Ricorrenza di un certo evento (es., la morte di un certo attore), propongo i suoi film	3	Analisi di stream di news, tweed, Facebook status. Analisi semantica per individuare gli hot topic, ricerca di film in base di dati semantica, confronto con catalogo.	---	Push di item associati con l'evento/news
		coming-soon movies	movies to be released	5	Sta uscendo un nuovo film su Batman, propongo i vecchi film di Batman o film sui supereroi.	5	Analisi di siti di informazione cinematografica	---	Push di item per genere, regista, attore. (Ricerca di un sistema di ranking ottimale)
Modal	subjective	mood	happy/sad	4	Propongo film il cui genere sia "compatibile" con lo stato d'animo dell'utente	0	► utente esprime umore ► derivato da "recent activity"	Considero rating solo per uno stato d'animo	Push di item aventi un certo genere
		experience/cognitiv capabilities	informatics expert buying a pc	2	Ad un utente esperto di cinema propongo film di nicchia e non popolari	0	Questionario all'utente.	►Considero solo utenti "esperti" per generare il modello ►Escludo item popolari da modello	Push di film di nicchia rispetto a quelli popolari
		intent	research for school/entertaineme	0	---	2	Analisi della cronologia dell'utente	---	---
	objective	recent activity: e. g., search/navigation	movie just seen	5	Propongo un film simile al film appena visto (raccomandazione content based)	5	Cronologia utente sulla nostra piattaforma.	---	Raccomandazioni con profilo utente in cui gli item associati all'attività recente pesano di più rispetto agli altri

Figura 4.2: Contesti individuat

Capitolo 5

Progettazione Use Case: "Attività Recenti"

In questo capitolo verrà analizzato dettagliatamente il processo di realizzazione del lavoro di tesi.
E' stata presa la decisione di implementare l'analisi contestuale delle attività recenti dell'utente, immaginando che questo navighi in alcuni siti di notizie. L'obiettivo del lavoro di tesi è quello di trovare delle relazioni tra gli item news e gli item film presi in considerazione.
Partendo di una descrizione generale si andrà ad analizzare approfonditamente ogni processo nel flusso dati movie e news.

Come possiamo vedere in Figura 5.1, il flusso schematizzato, valido per

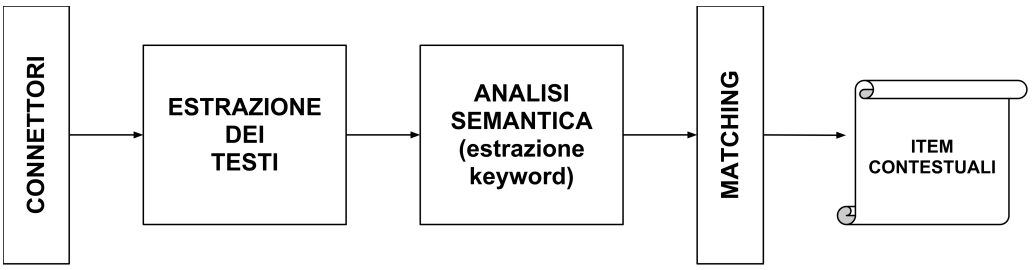

Figura 5.1: Schema del lavoro di tesi

entrambi gli item, è costituito da quattro fasi:

- *Connettori:* in questa fase iniziale vengono recuperati gli item dalle varie fonti definite in fase di progettazione;
- *Estrazioni dei testi:* gli item appena recuperati vengono elaborati, vedremo come in seguito, e ne vengono estratti i testi da analizzare;

- *Analisi Semantica*: qui i testi vengono parsati con strumenti di analisi semantica con l'obiettivo di produrre un elenco di parole chiave per ogni item;

- *Matching*: come passo finale viene individuata una relazione tra gli item di tipo news e di tipo film.

Nei due paragrafi successivi verranno discussi rispettivamente il flusso dati dei film e il flusso dati delle news.

5.1 Flusso Film

Il flusso film si occupa del recupero di informazione per la compilazione dell'oggetto "MovieBean", questo avviene attraverso le fasi di connessione alle fonti, di arricchimento e di riconciliazione.
L'oggetto MovieBean, mostrato in Figura 5.2, rappresenta l'oggetto film dove risiedono tutte le informazioni recuperato durante il processo. Della classe estesa MovieBeanSemantic si parlerà più approfonditamente nel capitolo successivo.
Per ogni oggetto sono disponibili le seguenti informazioni:

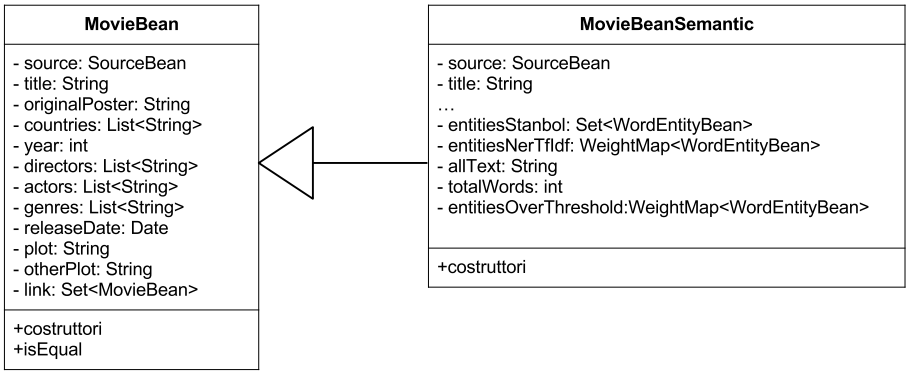

Figura 5.2: Classe MovieBean e classe estesa MovieBeanSemantic

- source: viene definita la fonte;

- title: il titolo del film;

- originalPoster: il link alla locandina del film;

- countries: la lista degli stati in cui è stato prodotto il film;

- year: l'anno di produzione;
- directors: la lista dei registi del film;
- actors: la lista degli attori del film;
- genres: la lista dei generi del film;
- releaseDate: la data in cui il film è stato rilasciato nei cinema;
- plot: la tram del film prelevata dalla fonte;
- otherPlot: altre trame del film prelevate da fonti esterne;
- link: un gruppo di altri oggetti MovieBean prelevati da altre fonti per arricchire a livello semantico i contenuti.

5.1.1 Connettori

I connettori si occupano di recuperare le informazioni dei film e di creare un set di MovieBean.
In primo luogo viene definita la sorgente, o le sorgenti, da cui prelevare i film con le relative informazioni.
Attraverso la classe connettori possiamo definire le nostre sorgenti che, pos-

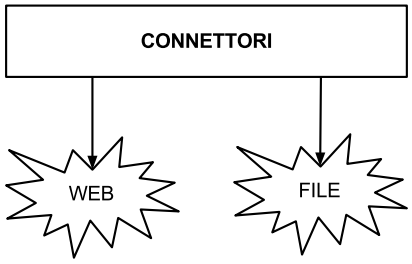

Figura 5.3: Connettori e Fonti

sono trovarsi nel web, come per esempi siti di informazione cinematografica, o possono essere cataloghi che risiedono sui file ben ordinati (cdv, ecc ...).
Come possiamo vedere in Figura 5.4, sono state definite due classi astratte (IConnector e IMovieConnector). La prima definisce i metodi per recuperare le informazioni di configurazione, ovvero tutte quelle informazioni dove risiedono indirizzi web, percorsi dei file ecc ..., mentre la seconda definisce il metodo retrieveMovie che si occupa del recupero dei film. Ogni classe che viene estesa da IMovieConnector si occuperà di implementare il metodo retrieveMovie secondo le specifiche della sorgente. Nel nostro caso sono state

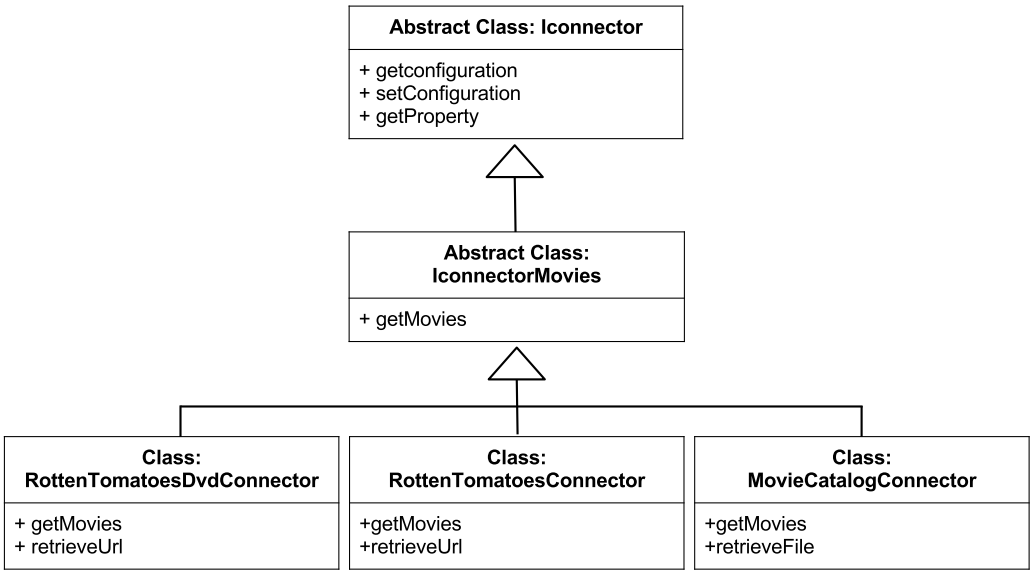

Figura 5.4: Definizione delle classi di Connettori

utilizzate due sorgenti: il sito web di informazione cinematografica "Rotten Tomatoes" e un catalogo cinematografico disponibile su un file.

Rotten Tomatoes

Rotten Tomatoes è un sito di informazione cinematografica che mette a dispozione degli utenti della api gratuite per l'interrogazione dei suoi database. Nel lavoro di tesi è stato scelto di appoggiarsi a due api che restituiscono i film americani in uscita rispettivamente al cinema e in dvd.
Attraverso la chiamata:

$http://api.rottentomatoes.com/api/public/v1.0/lists/movies/upcoming.json$

è possibile ricevere un file json, mostrato in Figura 5.5, che permette in maniera molto semplice di recuperare tutte le informazioni necessarie per ogni film in uscita al cinema.

Si può notare come sia possibile ricavare tutte le informazioni dell'oggetto MovieBean dal json restituito dall'api. Il json mantiene lo stesso tipo di formattazzione anche nel caso dei film in uscita in dvd, in questo caso l'unica modifica da fare è nel'url della chiamata, nella quale viene sostituita la parola movies con la parola dvds.

```
▼"movies": [
  ▼{
    ▼"abridged_cast": [
      ▼{
        ▶"characters": [...],
         "id": "162652472",
         "name": "Vin Diesel"
       },
      ▼{
        ▼"characters": [
             "Vaako"
         ],
         "id": "162654704",
         "name": "Karl Urban"
       },
      ▶{...},
      ▶{...},
      ▶{...}
    ],
   ▶"alternate_ids": {...},
    "id": "771267761",
   ▶"links": {...},
    "mpaa_rating": "R",
   ▼"posters": {
      "detailed": "http://content8.flixster.com/movie/11/17/20/11172082_det.jpg",
      "original": "http://content8.flixster.com/movie/11/17/20/11172082_ori.jpg",
      "profile": "http://content8.flixster.com/movie/11/17/20/11172082_pro.jpg",
      "thumbnail": "http://content8.flixster.com/movie/11/17/20/11172082_mob.jpg"
    },
   ▶"ratings": {...},
   ▼"release_dates": {
      "theater": "2013-09-06"
    },
    "runtime": 119,
   ▶"synopsis": "...",
    "title": "Riddick",
    "year": 2013
  },
```


Figura 5.5: Rotten Tomatoes: esempio di file Json

Catalogo

Attraverso un catalogo di circa 5000 film è stato possibile recuperare le informazione per ogni MovieBean anche offline, senza doversi connettere a servizi esterni.
Nel file a nostra disposizione, per ogni film erano disponibili le seguenti informazioni: id, genere, country, actors, directors, title, year e plot.

5.1.2 Reconciliation

Dopo aver recuperato un set di MovieBean attraverso la fase di connessione alla fonti, il set di film viene riconciliato con una base semantica. Nel nostro caso è stato utilizzato il sito www.freebase.com.
Anche in questo caso sono state utilizzate api gratuite che permettono in primo luogo di riconoscere in maniera univoca il film da noi cercato e in seguito di arricchire il nostro oggetto MovieBean con informazioni testuali aggiuntive.
Con una prima chiamata viene interrogata l'api di ricerca di freebase, l'output aspettato sono oggetti di tipo film. In questa chiamata vengono passate due variabili per cercare di identificare in maniera univoca i film ricercati: il titolo e l'anno di produzione.
Facciamo notare che in alcuni casi è stato necessario passare come parametro l'anno precedente a quello in nostro possesso (per esempio, invece di 2002 è stato necessario passare 2001) in quanto per molti film nel database semantico è presente la versione europea, quindi, nel caso di film americani, viene proposta una data di uscita successiva a quella recuperata nelle fonti.
La risposta a questa interrogazione restituirà un oggetto di tipo film caratterizzato da un codice univoco.
A questo punto sarà possibile ricercare l'oggetto specifico all'interno del database con un ricerca per codice(quello appena ricevuto), in maniera tale da poter estrarre le informazioni aggiuntive di cui necessitiamo.

Nelle figure 5.6 e 5.7 è presentato un breve esempio dove verranno illustrati i parametri utilizzati nelle chiamate.
In Figura 5.5, viene effettuata la prima chiamata al database di freebase per il recupero dell'id del film da riconciliare. I parametri della chiamata sono:

- name: il nome del film;
- kind: il tipo di oggetto che il sistema deve restituire, nel nostro caso filtriamo solo i film;

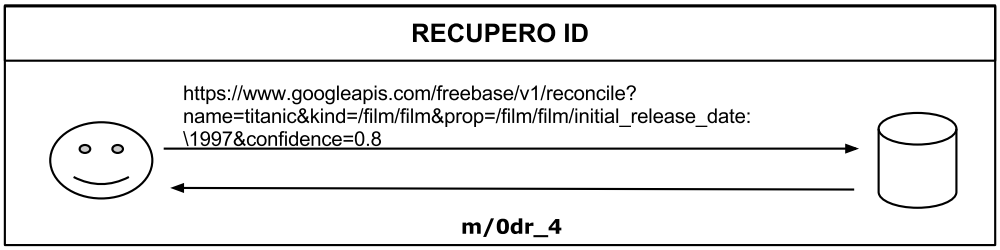

Figura 5.6: Prima chiamata al database di freebase

- prop: definiamo un ulteriore filtro per la ricerca passando l'anno di produzione
- confidence: in confidence viene definita una soglia di confidenza per filtrare gli item non idonei, nel nostro caso fissiamo una soglia molto alta in maniera tale che venga restituito solo un oggetto (quello da noi ricercato).

Figura 5.7: Seconda chiamata al database di freebase

Come risultato viene restituito il codice univoco dell'oggetto, che, come si può vedere in Figura 5.7, sarà utilizzato per recuperare le informazioni del film.

5.1.3 Enrichment

Nella fase dell'enrichment l'obiettivo è quello di connettersi a siti web dove è possibile reperire informazioni da aggiungere ai nostri oggetti MovieBean.

Nel lavoro di tesi si è deciso di appoggiarsi al colosso di informazione cinematografico imdb.
Anche in questo caso è stato possibile utilizzare delle api messe a disposizione gratuitamente dal sito http://mymovieapi.com, che permettono di interfacciarsi con il database di imdb.
La fase di enrichment va a creare un nuovo set di oggetti di tipo moviebean che andranno a far parte della variabile Link dell'oggetto principale a cui sono collegati (Figura 5.8).

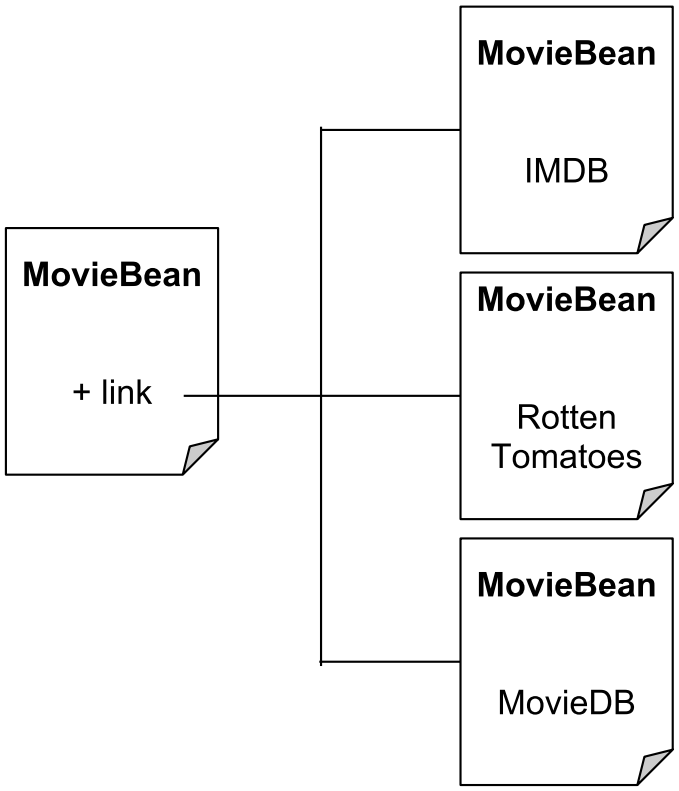

Figura 5.8: Schema Link Moviebean

La richiesta di enrichment all'api può avvenire in due modi:

- con id imdb: è sufficiente passare come parametro l'id imdb e verrà restituito l'item film corrispondente. I film che in precedenza sono stati caricati da Rotten Tomatoes presentano tra i campi l'id imdb.

- senza id imdb: è necessario effettuare una interrogazione che recuperi in maniera univoca un film. Come nel caso della reconciliation, viene

effettuata la richiesta passando come parametri il titolo e l'anno di produzione del film.

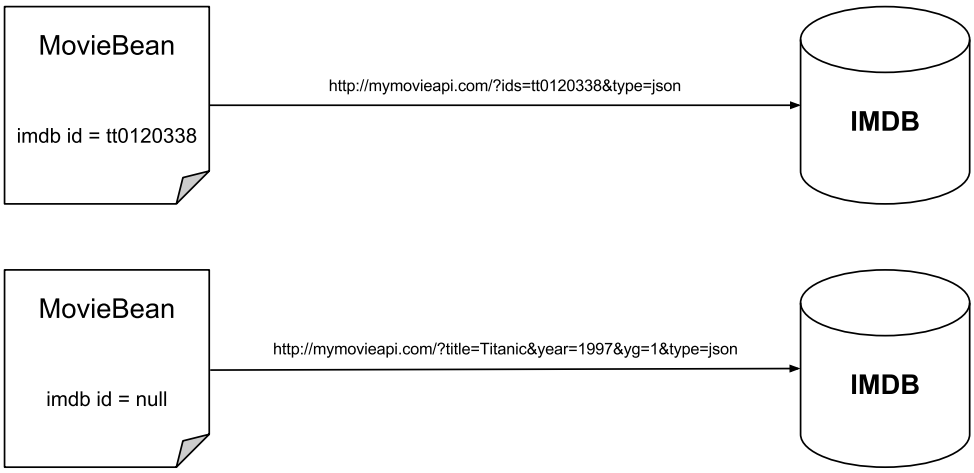

Figura 5.9: Chiamata Imdb

Nel caso non venissero trovati film passando come parametro l'id di imdb è stata implementata la possibilità di effettuare una seconda richiesta passando titolo e anno di produzione. Se anche in questo caso non fosse possibile ricavare nessuna informazione, l'enrichment non viene effettuato.
In Figura 5.9 vengono mostrate le due chiamate, la prima con il passaggio del parametro id e la seconda con il passaggio di titolo e anno di produzione. Anche in questo caso il formato restituito dall'api è un json di facile interpretazione (Figura 5.10).

```
{
▼"actors": [
        "Leonardo DiCaprio",
        "Kate Winslet",
        "Billy Zane",
        "Kathy Bates",
        "Frances Fisher",
        "Gloria Stuart",
        "Bill Paxton",
        "Bernard Hill",
        "David Warner",
        "Victor Garber",
        "Jonathan Hyde",
        "Suzy Amis",
        "Lewis Abernathy",
        "Nicholas Cascone",
        "Anatoly M. Sagalevitch"
  ],
▶"also_known_as": [...],
▼"country": [
        "USA"
  ],
▶"directors": [...],
  "filming_locations": "Santa Clarita, California, USA",
▶"genres": [...],
  "imdb_id": "tt0120338",
  "imdb_url": "http://www.imdb.com/title/tt0120338/",
▶"language": [...],
  "plot_simple": "A seventeen-year-old aristocrat, expecting to be married to a rich claimant by her mother,
    aboard the luxurious, ill-fated R.M.S. Titanic.",
▶"poster": {...},
  "rated": "PG-13",
  "rating": 7.6,
  "rating_count": 489028,
  "release_date": 19980403,
▶"runtime": [...],
  "title": "Titanic",
  "type": "M",
▶"writers": [...],
  "year": 1997
```


Figura 5.10: Imdb: file Json

5.1.4 Consolidator

A questo punto, prima di passare all'analisi semantica degli oggetti, viene effettuato un controllo di unicità sul set di MovieBean per evitare che vi siano oggetti duplicati.
Questo controllo viene effettuato in due fasi: nella prima si cercano di eliminare oggetti che abbiano l'indirizzo di reconciliation uguale, ovvero che facciano riferimento allo stesso film, nella seconda vengono confrontati per ogni film la coppia titolo e anno. A questo punto per ogni MovieBean è possibile creare un variabile allText, in cui vengono concatenati tutti i plot(plot della fonte, plot della reconciliation e plot dei vari enrichment).

5.2 Flusso News

Il flusso news si occupa del recupero di informazione per la compilazione dell'oggetto "NewsBean". In questo caso, a differenza del flusso Film, è presente solo la fase di connessione alle fonti. Le fonti in questo caso sono i siti web che si occupano di cronaca a livello locale e mondiale.
L'oggetto NewsBean, mostrato in Figura 5.11, rappresenta l'oggetto news dove risiedono tutte le informazioni recuperate durante la fase di connessione alle fonti. Della classe estesa NewsBeanSemantic si parlerà più approfonditamente nel capitolo successivo.
Per ogni oggetto sono disponibili le seguenti informazioni:

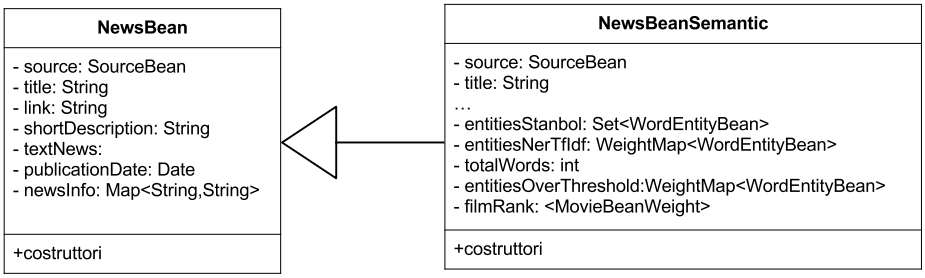

Figura 5.11: Classe NewsBean e classe estesa NewsBeanSemantic

- source: viene definita la fonte, ovvero da che quotidiano o sito è presa la notizia
- title: il titolo della notizia;
- link: il link alla notizia;

- shortDescription: una breve descrizione della notizia. Di solito in questo campo vengono inserite le informazioni prelevate dai feed rrs che racchiudono le prime 15-25 parole della notizia;
- textNews: il testo della notizia ricavato da feed rrs completi oppure dall'estrazione del testo dalla pagina html;
- publicationDate: la data di pubblicazione della news;
- newsInfo: informazioni aggiuntive come per esempio il genere della news(sport, world news, dcc ...);

5.2.1 Connettori

I connettori si occupano di recuperare le news dal web e di creare un set di NewsBean del giorno corrente.
In primo luogo viene definita la sorgente, o le sorgenti, da cui prelevare le news con le relative informazioni. Per il lavoro sono stati scelte come fonti sei quotidiani online: quattro americani, uno inglese e uno asiatico.
Come possiamo vedere in Figura 5.12, sono state definite due classi astratte

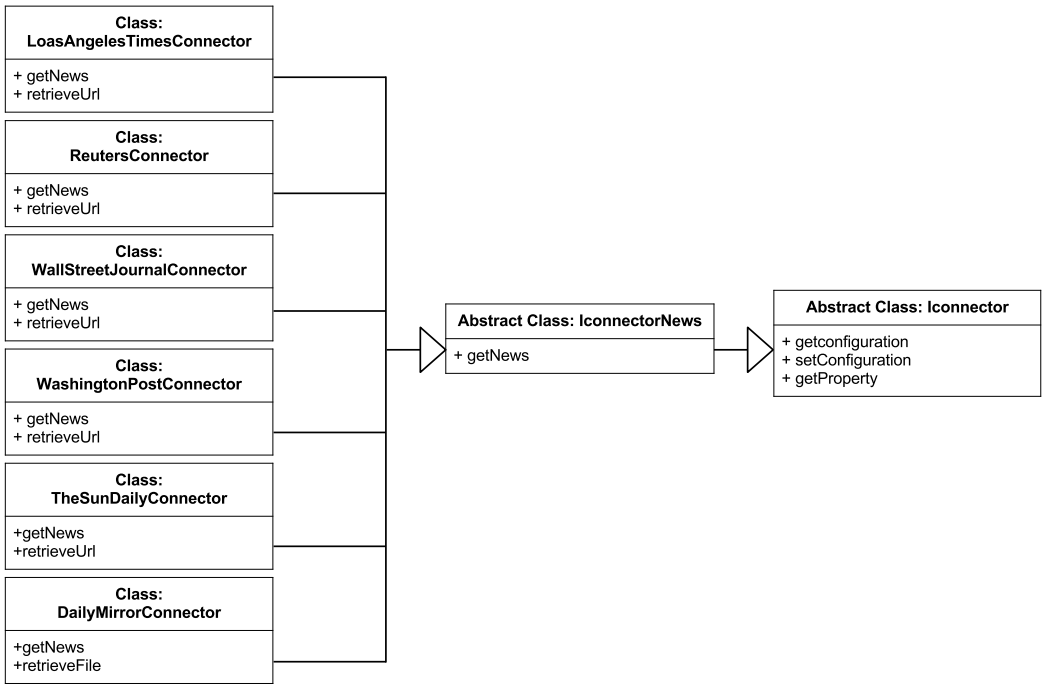

Figura 5.12: Definizione delle classi di Connettori

(IConnector e INewsConnector). La prima definisce i metodi per recuperare

le informazioni di configurazione, ovvero tutte quelle informazioni dove risiedono gli indirizzi web delle fonti sia nel formato xml sia nel formato html, mentre la seconda definisce il metodo getNews che si occupa del recupero delle news. Ogni classe che viene estesa da INewsConnector si occuperà di implementare il metodo getNews secondo le specifiche della sorgente. Per ogni quotidiano web sono state prelevate le notizie in formato XML, nei siti gratuiti è stato anche possibile ricavare il testo completo della news parsando la pagina html al link ricavato dal parsing dell'XML. Dopo la compilazione del set, gli oggetti NewsBean, a differenza dei MovieBean, sono pronti per la fase di analisi semantica.

In questa parte del capitolo si è vista una prima fase del progetto di tesi, dalla decisione delle fonti alla realizzazione di due Set di oggetti, NewsBean e MovieBean, pronti per essere utilizzati come input nella fase successiva di analisi semantica.
Prima di proseguire nell'analisi e nel parsing delle parti testuali individuate è necessario presentare una classe Java di fondamentale importanza per il processo di analisi semantica: la classe WordEntityBean.

5.3 WordEntityBean

La classe WordEntityBean, mostrata in Figura 5.13, rappresenta le entità estratte dai testi presenti negli oggetti MovieBean e NewsBean.
Le variabili che compongono la classe sono:

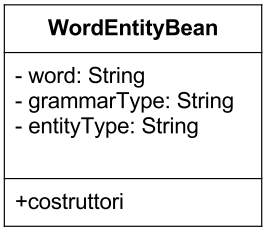

Figura 5.13: Classe WordEntityBean

- Word: all'interno di questa variabile viene salvata la parola estratta;
- grammarType: in questo campo viene identificato il tipo grammaticale della parola. Nel nostro caso di studio i tipi grammaticali tenuti in considerazione sono nomi propri e nomi comuni plurali o singolari;

- entityType: in questo campo viene salvato il tipo di entità definito dalla parola. Nel nostro caso di studio vengono estratte entità quali nomi di persone, nomi di luoghi, nomi di organizzazioni e nomi comuni.

Come è facilmente intuibile ogni oggetto semantico di tipo MovieBean e NewsBean sarà caratterizzato da un set di WordBean contenente le proprie parole chiave estratte. Nel prossimo paragrafo verranno illustrati dettagliatamente gli oggetti semantici che espandono le classi MovieBean e NewsBean.

5.4 Classi Semantiche

Le classi di oggetti MovieBean e NewsBean, prima di essere elaborate dal blocco semantico del software, vengono estese e acquisiscono nuove proprietà(Figura 5.14).

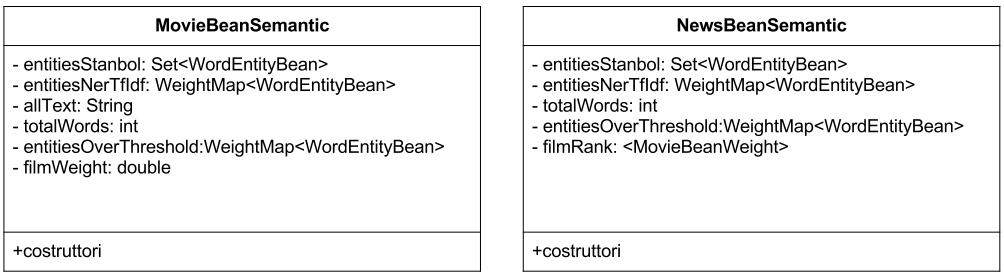

Figura 5.14: Classi semantiche

Iniziamo ad elencare e spiegare le variabili comuni ad entrambe le classi:

- entitiesStanbol: in questo Set vengono raccolte tutte le entità ricevute dalla chiamata all'api di stanbol;
- entitiesNerTfIdf: in questo Set vengono raccolte tutte le entità estratte dal software attraverso gli strumenti messi a disposizione da Stanford;
- totalWords: viene indicato il numero di parole che vengono parsate. Questo numero viene utilizzato per i calcoli del Tf-Idf per determinare il peso delle parole;
- entitiesOverThreshold: in questo Set vengono individuate le parole che hanno un peso sufficiente per essere considerate significative nel processo di matching;

Inoltre abbiamo due variabili esclusive per l'oggetto MovieBeanSemantic:

- allText: in questa stringa vengono concatenate tutte le trame relative al film;
- filmWeight: il peso del film relativo ad una certa news, ovvero la somma delle parole chiave che corrispondono alle parole chiavi trovate nel testo di una news;

Infine, abbiamo una variabile esclusiva anche per l'oggetto NewsBeanSemantic:

- filmRank: in questa variabile abbiamo una lista ordinata di film con cui è stata trovata almeno una corrispondenza, ovvero dove il filmWeight è diverso da zero.

5.5 Analisi Semantica

Dopo aver illustrato gli oggetti che andranno in input al blocco di analisi semantica (Figura 5.15), passiamo ad analizzare i vari passaggi che porteranno alla creazione degli item semantici.
Il blocco di analisi semantica che elaborerà i set di film e news è composto

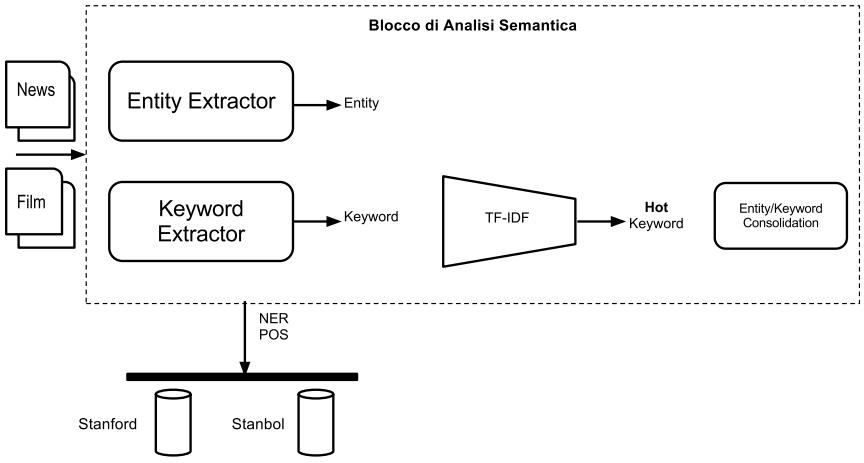

Figura 5.15: Blocco di analisi semantica

principalmente da tre fasi:

- Connessione Stanbol:
- Analisi Stanford:

- Calcolo Tf-Idf:

Durante l'esecuzione di queste fasi gli oggetti semantici verranno compilati e saranno pronti per il matching finale che porterà ad avere corrispondenze tra film e news.

5.5.1 Connessione a Stanbol

Come già illustrato nel capitolo 3, in questa fase per ogni elemento del Set viene selezionata la variabile contenente il testo da cui estrarre le parole chiave (per le News textNews e per i film allText).
I testi vengono passati all'api di Stanbol che, dopo averli parsati, restituirà un Json contenente le entità estratte. Ricordiamo che Stanbol non lavora sulle occorrenze delle parole nel testo ed estrae esclusivamente nomi propri di persone, luoghi e organizzazioni.

5.5.2 Analisi Stanford

L'analisi Stanford differisce dall'analisi Stanbol perchè, in primo luogo, non filtra le entità, ma filtra i nomi, sia propri che comuni, e solo in seguito identifica le entità nel set di nomi estratti. Più tecnicamente, come primo procedimento viene effettuata un'analisi PoS (part of speech) che identifica le parti del discorso per ogni parola e solo in seguito, sulle parole prese in considerazione, vengono identificate le entità tramite gli strumenti NER (Name Entity Recognition). Naturalmente i nomi comuni non avranno nessuna entità associata. Nel lavoro di tesi si è deciso di estrarre i nomi propri e comuni, in quanto attraverso essi è possibile identificare i topic del testo preso in considerazione.
Per capire meglio la differenza tra i due procedimenti viene utilizzato il seguente esempio. Consideriamo una notizia in cui viene descritta l'ultima conferenza di Barack Obama tenuta alla Casa Bianca riguardante il sistema sanitario nazionale. L'analisi Stanbol riconoscerà le entità presenti nel testo, quindi probabilmente restituirà come risultati "Barack Obama", "White House", "Washington" e "United States". Da queste entità però non possiamo comprendere pienamente il topic della news. Invece, attraverso un'analisi che utilizza gli strumenti di Stanford, si sarà in grado di estrarre anche i nomi comuni che occorrono nel testo. Quindi probabilmente verranno individuate parole come "assicurazione sanitaria" e "ospedali", oltre a tutti nomi propri elencati precedentemente.
A livello di sviluppo software è stato implementato del codice in grado di utilizzare le classi PoS e NER di stanford, in maniera tale da poter associare

ad ogni parola le sue caratteristiche grammaticali e semantiche. Ricordiamo che è stato necessario anche utilizzare un file vocabolario di supporto alla lingua inglese, sempre messo a disposizione da stanbol.

5.5.3 Calcolo Tf-Idf

A questo punto è necessario valutare effettivamente il peso delle parole estratte all'interno dei testi presi in considerazione. E' stato scelto di utilizzare come metodo di valutazione l'algoritmo Tf-idf.
Nel Tf-Idf 'importanza della parola cresce proporzionalmente con il numero delle occorrenze in un documento ma diminuisce se le occorrenze aumentano all'interno di tutta la collezione [49].
Per ogni parola estratta è stata implementata una mappa che tenesse conto delle occorrenze della parola all'interno del documento,inoltre, per ogni documento è stato individuato il numero totale di parole.
Con questi dati è stato possibile calcare il TF delle parole secondo la formula:

$$TF_{i,j} = \frac{n_{i,j}}{\sum_{k=1}^{n} n_{k,j}}$$

dove $n_{i,j}$ è il numero di occorrenze del termine T_i considerato nel documento D_j e il denominatore è la somma delle occorrenze di tutti i termini all'interno del documento D_j.
Per quanto riguarda l'IDF è stato necessario creare una mappa di tutte le entità estratte e di tutte le loro occorrenze all'interno della collezione dei documenti, quindi è stato possibile calcore l'IDF:

$$IDF_i = \frac{log\,|D|}{nD_i}$$

dove D è il numero totale dei documenti del set preso in considerazione e il denominatore è il numero dei documenti dove compare il termine T_i.
A questo punto possiamo ricavare il peso della parola con:

$$TFIDF_i = TF_{i,j} \times IDF_i$$

Bisogna notare che quando parliamo di collezione di item, per i film ci riferiamo alla totalità degli oggetti presi in considerazione, mentre per le news utilizziamo il set di news prese in considerazione dall'utente, più un catalogo di news che viene prelevato e salvato da tutte le fonti giornalmente.
Prima di procedere alla fase finale di matching è necessario associare ad ogni keyword estratta il peso TF-IDF ricavato ed eliminare eventuali parole duplicate all'interno di ogni item.

Inoltre, come vedremo nel Capitolo 6, dalla fase di analisi risulterà più efficiente eliminare parole con un peso inferiore a soglie definite in fase di testing. Per esempio, ipotizzando di fissare un soglia con un valore di 0.8 verrano eliminate tutte le parole con peso TF-IDF inferiore a 0.8.

5.6 Matching

Nella fase conclusiva del flusso, per ogni news presa in considerazione, vengono individuati tutti i film in cui è possibile trovare almeno una corrispondenza di parole chiave.
Per fare questo, per ogni set di parole chiave di ogni news, vengono ricercate delle corrispondenze in ogni set di parole chiave di ogni film.
Il peso del film per la news presa in considerazione sarà calcolato come la somma di tutti i prodotti tra i pesi delle parole che hanno un riscontro positivo.
A questo punto è possibile ricavare una classifica di film per ogni news, dal film più rilevante semanticamente o quello meno rilevante.

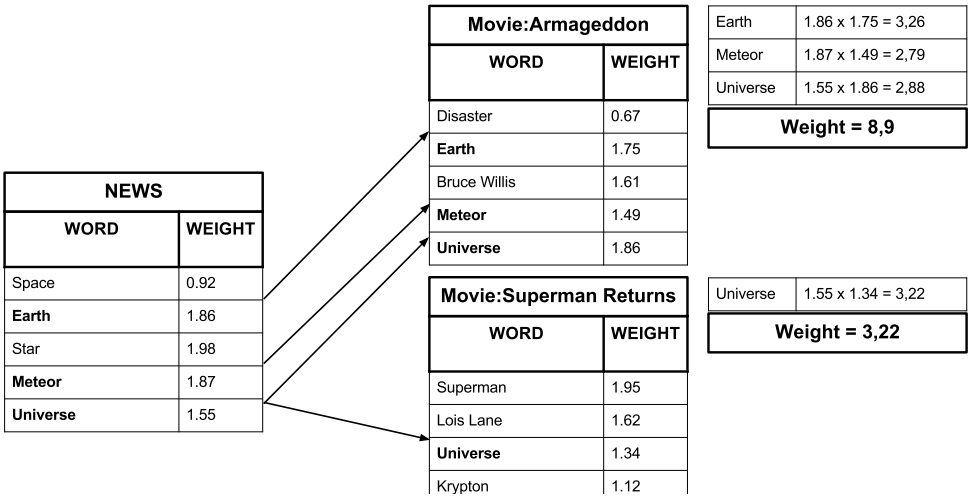

Figura 5.16: Esempio di Matching

In Figura 5.16, è possibile vedere un esempio di matching. Data una news generica con un set di parole chiave ognuna con un peso troviamo tre riscontri con il film "Armageddon" e un riscontro con il film "Superman Returns", in base al peso finale determiniamo la posizione in classifica dei film coerenti con la news. Bisogna notare che un maggior numero di parole chiave non implica sempre una posizione migliore in classifica, in quanto

due parole chiave con un peso molto basso andranno a pesare meno di una parola chiave con un peso elevato.

Concludiamo il capitolo con lo schema concettuale completo del lavoro di testi svolto (Figura 5.17).

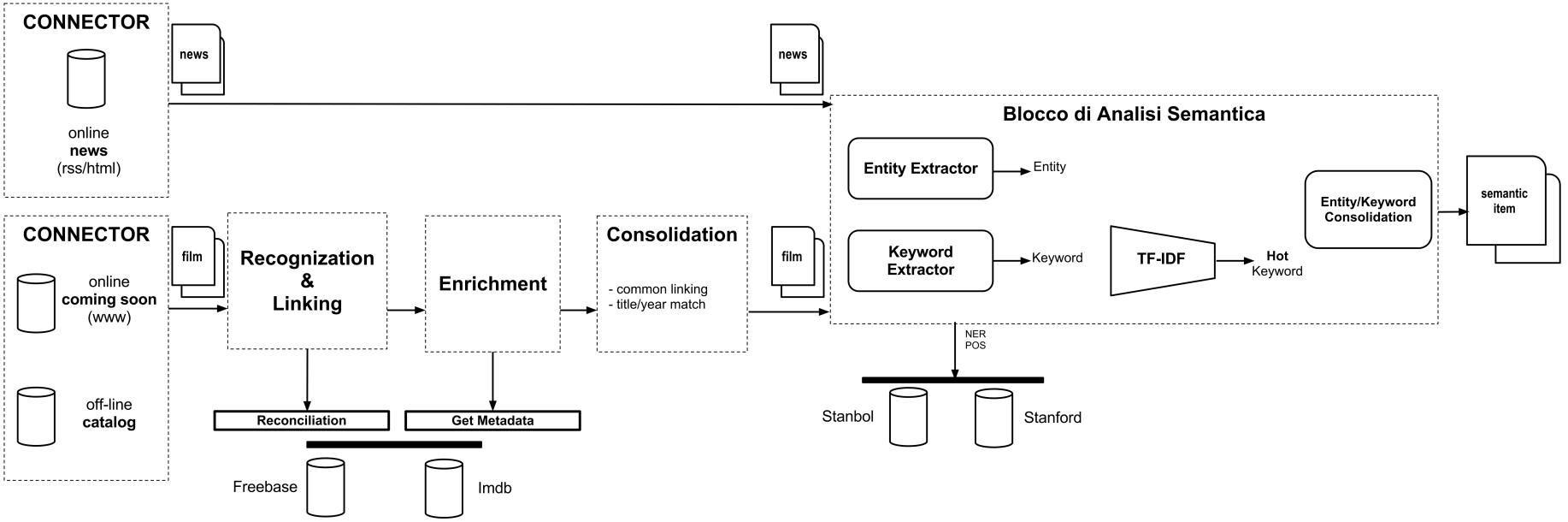

Figura 5.17: Schema concettuale

Capitolo 6

Testing e Analisi

In questo capitolo, utilizzando algoritmi content-based presenti in letteratura, verrà valutata l'efficacia delle parole chiave estratte durante il lavoro di tesi. Verranno testate, attraverso due tipi di algoritmi (KNN e LSA), quattro set diversi di parole chiave e verranno prodotti e analizzati risultati secondo classiche metriche quali recall e fallout.

6.1 Algoritmi Utilizzati

Per effettuare i test è stato utilizzato un catalogo di 4489 film con 2 milioni di rating rilasciati da 5709 utenti; un sottoinsieme del dataset Netflix[1] arricchito con metadati estratti mediante web crawling. Il test è basato sulle modalità descritte in [26] e utilizzate in letteratura per valutare le qualità delle raccomandazioni $Top-N$ (cioè la capacità di generare N raccomandazioni rilevanti). Dal catalogo sono stati estratti per ogni film: titolo, genere, registi, attori, luogo, anno di produzione e trama. Dopo aver effettuato l'estrazione dei dati, il set di film è stato riconciliato con freebase e arricchito con le informazioni di imdb.
A questo punto è stato deciso di creare quattro set diversi di parole chiave:

- All (stem): in questo set sono state eliminate le parole di lunghezza inferiore a 3 caratteri, superiore a 20 caratteri e incluse nella lista delle stopwords del vocabolario inglese(es., the). Alle parole restanti è stato applicato un algoritmo di stemming, ovvero un algoritmo che si occupa di identificare la radice delle parole prese in considerazione(es., la parola *playing* ha come radice *play*. Questo tipo di set è il set che di solito viene utilizzato per le raccomandazioni di tipo content-based, è verra quindi usato come baseline.

- Nomi Propri: in questo set sono state considerate solo le entità estratte da Stanbol(nomi, luoghi e organizzazioni);
- Nomi (no stem): in questo set vengono considerate le parole estratte dal software sviluppato nel lavoro di tesi(utilizzando sia Stanbol che Stanford), ovvero nomi propri e nomi comuni plurali e singolari;
- Nomi (stem): come nel caso precedente, anche in questo set vengono prese in considerazione le parole estratte dal lavoro di tesi, la differenza si riscontra nell'applicazione dell'algoritmo di stemming per i nomi comuni.

Il primo passo per la definizione degli input è stata la creazione di file che associasse un id numerico ad ogni parola chiave e film. Quindi sono stati generati cinque file: quattro per le parole chiave(*dictionary*) e uno per film(*itemCache*).
In Figura 6.1 sono mostrati due esempi: il primo relativo alle parole chiave e il secondo relativo ai film.

Nomi (no stem)	
Rhode Island	1
Ministry	2
Victor Hugo	3
answer	4
swords	5
...	...

Film	
The Flintstones1994	1
Kug Fu2004	2
Taxi Driver1976	3
Dirty Dancing1987	4
Helboy2004	5
...	...

Figura 6.1: File: entità-identificativo

Come possiamo notare dalla Figura 6.1, per individuare in maniera univoca i film si è scelto di concatenere l'anno di produzione al titolo.
Dopo la creazione di questi file è stato necessario modificare il file dei rating, in maniera tale che venisse associato ad ogni coppia utente-rating l'id ricavato dal file itemCache.
In questo modo è stato possibile creare il file *urm raw* che include 2 milioni di record nella forma: id numerico utente - id numerico film - rating (valore numerico compreso tra 1 e 5).
Come ultimo passo, è stato necessario importare tutti i valori di peso associati ad ogni parola dei quattro dictionary e creare quattro file (*icm raw*) nei

quali vengono associate ad ogni film le proprie parole chiave con il relativo peso.

6.1.1 Valutazione raccomandazioni

Importando i file appena creati in Matlab è stato possibile generare le matrici necessarie all'esecuzione degli algoritmi.
Come prima cosa è stata creata la matrice dei rating, nella quale troviamo gli utenti (5709) nelle righe e i film (4489) nelle colonne. Questa matrice, chiamata *urm test* è la matrice che contiene i dati reali dei rating rilasciati dagli utenti sui film del catalogo e verrà usata per testare i risultati degli algoritmi.
Per lo svolgimento del test si è deciso di prelevare il 5% dei rating della matrice urm test ed eliminarli, andando a creare un nuova matrice chiamata *urm train*.
Il secondo passo è stato quello di creare quattro matrici dei pesi, relative ai set di parole chiave passati. Queste matrici sono costituite dall'identificativo della parola nelle righe e dall'identificativo dei film nelle colonne. Quindi, il valore dell'elemento (1,3) della matrice, rappresenterà il peso della parola con identificativo 1 nel film con identificativo 3.
Per completezza, nella Figura 6.2 sono riportate tutte le matrici create e le loro dimensioni.
Con tutti i dati a disposizione è stato possibile eseguire l'analisi di recall e

NOME MATRICE	DIMENSIONE	DESCRIZIONE
URM_TEST	5709 x 4489	Matrice dei rating reali
URM_TRAIN	5709 x 4489	Matrice dei rating reali con 5% di dati eliminati
ICM_ALL_STEM	33638 x 4489	Insieme dei pesi delle parole ricavate esclusivamente con metodo di stemming
ICM_NOMI_PROPRI	14155 x 4489	Insieme dei pesi delle parole ricavate da Stanbol
ICM_NOMI_NOSTEM	37787 x 4489	Insieme dei pesi delle parole ricavate dai tools di Stanford senza metodo di stemming
ICM_NOMI_STEM	34919 x 4489	Insieme dei pesi delle parole ricavate dai tools di Stanford con metodo di stemming

Figura 6.2: Matrici MatLab

fallout utilizzando gli algoritmi KNN e LSA.
Tenendo conto della matrice di dati passata in ingresso come parametro, l'obiettivo degli algoritmi è quello di utilizzare la matrice urm train per

effettuare delle raccomandazioni sui valori nulli(quelli eliminati precedentemente) e confrontarli con quelli reali presenti nella matrice urm test. Naturalmente ad un numero elevato di riscontri positivo corrisponderà una maggiore efficacia delle parole chiave prese in considerazione.

Prima di illustrare gli algoritmi utilizzati e le discussioni dei risultati, definiamo il concetto di recall, fallout e grafici ROC.

6.1.2 Recall

La *recall* è una classificazione statistica frequentemente utilizzata in diversi ambiti del sapere. In statistica è definita come il numero di veri positivi diviso il numero totale di elementi che attualmente appartengono alla classe. Nell'*information retrieval* la recall è definita come il numero di item attinenti recuperati da una ricerca diviso il numero totale di item esistenti. Un valore di recall pari a 1.0 indica che tutti gli item attinenti sono stati recuperati dalla ricerca. Nell'ambito dei sistemi di raccomandazione la recall è calcolata in funzione del numero di item mostrati all'utente (n). Si procede calcolando le predizioni dei rating per tutti gli item non valutati da ogni utente, quindi si selezionano i primi n item ordinati secondo il rating predetto ($top - n$ *item*), infine se l'item compare nella lista delle raccomandazioni si ha una *hit*.
Quindi si calcola come[27]:

$$recall(n) = \frac{\#hits(T^+)}{|T^+|}$$

Dove T^+ è il set degli item rilevanti. Generalmente in un sistema di raccomandazione il set degli item rilevanti è composto da item a cui è stato assegnato un rating positivo, e viceversa si considerano non rilevanti gli item con un rating negativo.

6.1.3 Fallout

L'interpretazione della recall può di per sè essere fuorviante poichè, sebbene rappresenti in maniera efficace il numero di predizioni corrette, non da alcuna informazione sui *falsi positivi*. Per questo motivo si accompagna allo studio della recall quello del *fallout*, così da dare una misura di quanti item irrilevanti siano stati erroneamente predetti. Il calcolo è analogo a quello della recall, differisce esclusivamente nell'insieme utilizzato come test. Se per il calcolo della recall si utilizza l'insieme dei rating positivi T^+, per il fallout si utilizza l'insieme dei rating negativi T^-. Si procede calcolando le

predizioni dei rating per tutti gli item non valutati da ogni utente, quindi si selezionano i primi n item ordinati secondo il rating predetto ($top-n\ item$), infine se l'item compare nella lista delle raccomandazioni si ha una *hit*. Quindi si calcola come[27]:

$$fallout(n) = \frac{\#hits(T^-)}{|T^-|}$$

Dove T^- è il set degli item non rilevanti.

6.1.4 ROC

L'analisi di recall e fallout è facilitata dalle curve ROC (*Receiver Operating Characteristic* o anche note come *Relative Operating Characteristic*[30]). Si tratta di schemi grafici per un classificatore binario. Lungo i due assi sono rappresentate la *recall* e il *fallout* (o in un contesto piu' generale rispettivamente veri positivi e falsi positivi). La curva di ROC per un sistema di raccomandazione casuale è una retta diagonale che unisce gli estremi del grafico. Per confrontare le curve di ROC viene spesso utilizzata l'area sotto la curva: la bontà di un algoritmo di raccomandazione è proporzionale all'area sotto la sua curva di ROC; un algoritmo di raccomandazione casuale ha un'area sotto la curva di 0.5, che costruisce il limite minimo per la qualità del sistema di raccomadazione. Valori minori di 0.5 indicano che è preferibile un sistema di raccomandazione casuale e, quindi, che l'algoritmo non è molto efficace.

6.2 Algoritmi

Gli algoritmi KNN e LSA sono algoritmi content-based. L'obiettivo degli algoritmi Content-Based, nel nostro dominio, è quello di generare delle liste di raccomandazione basandosi sui profili utente e sulle informazioni contenute nei film(nel nostro caso le parole estratte).

Questo tipo di algoritmi necessita di una fase preliminare di creazione del modello(Figure 6.3). Nella creazione del modello, come visto precedentemente, vengono create le matrici icm e le matrici urm.

6.2.1 Algoritmo KNN

L'obiettivo dell'algoritmo KNN, come è possibile vedere in Figura 6.4, è quello di riuscire ad ottenere una matrice URM dove sono contenuti i rating

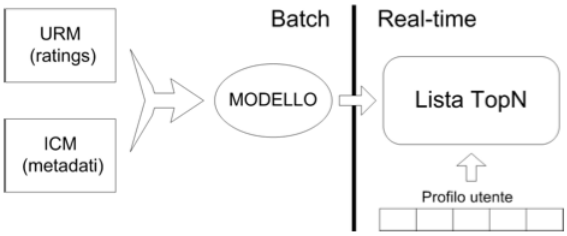

Figura 6.3: Modello per analisi

previsti per ogni coppia item utente ancora non esistente.

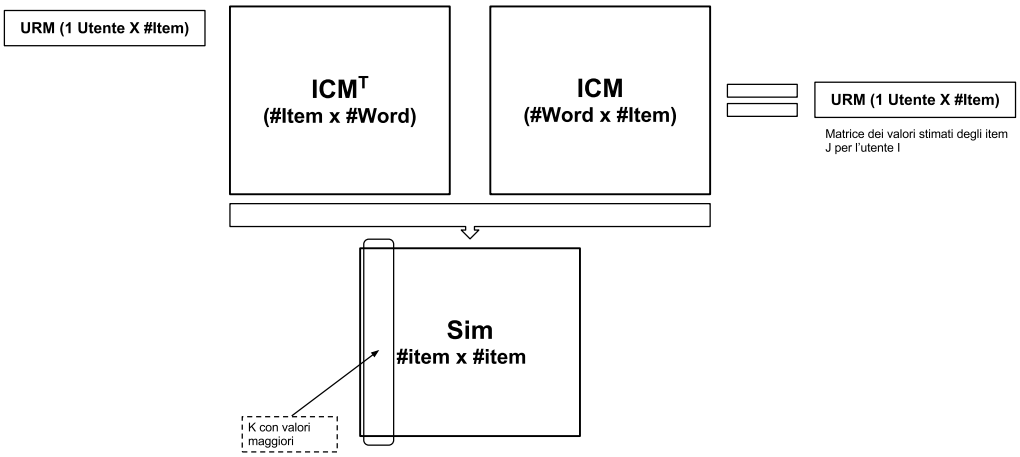

Figura 6.4: Algoritmo KNN

Per raggiungere l'obiettivo è necessario creare una matrice di similitudine definita dal prodotto delle matrice ICM trasposta e ICM. Per ogni colonna della matrice di similitudine vengono tenuti in considerazione i K elementi con valore maggiore.
La matrice URM, contenente i valori dei rating noti degli utenti per ogni film, viene moltiplicata per la matrici di similitudine per generare la matrice URM dei rating stimati.
Bisogna notare che la selezione dei k elementi con valore maggiore non corrisponde all'identificazione degli item più significati per ogni utente.

6.2.2 Algoritmo LSA

L'algoritmo LSA porta agli stessi risultati presentati nel paragrafo precedente, la differenza principale risiede nei metodi che ci portano alla creazione della matrice di similitudine.
Prima di presentare l'algoritmo LSA è necessario illustrare un procedimento matematico che ci permette di decomporre una qualsiasi matrice M di dimensioni $a \times b$ in tre matrici (SVD)

$$M = U \times \Sigma \times V^T$$

dove:

- M: è una matrice di dimensioni $a \times b$
- U: è una matrice ortonormale di dimensioni $a \times r$
- Σ: è una matrice diagonale di dimensioni $r \times r$ ordinata per valori decrescenti
- V^T: è la matrice trasposta coniugata di una matrice ortonormale di dimensioni $r \times b$

Le matrici U e V hanno colonne ortonormali, ovvero godono della seguente proprietà:

$$U^T \times U = V^T \times V = I$$

quindi il prodotto di U e V per le loro trasposte è uguale alla matrice di identità.
Nell'algoritmo LSA, come primo passo, viene applicato il procedimento SVD alla matrice ICM e alla sua trasposta.In questo modo definiamo la matrice di similitudine, non più come prodotto tra ICM e la sa trasposta, ma come prodotto una serie di prodotti delle 6 matrici generate.
Prima di procedere al calcolo della matrice URM dei rating previsti, viene selezione un campione K(con $k << item$) all'interno della matrice Σ che tiene conto degli elementi con peso maggiore e quindi più significativi.
A questo punto possiamo definire la matrice di similitudine come:

$$S = (\Sigma \times V^T)^T (\Sigma \times V^T)$$

6.3 Analisi dei risultati

Per lo svolgimento dei test sono state effettuate 20 analisi per ogni icm, si è partiti con un livello di soglia uguale a 0 per arrivare ad una soglia

di 2.0. Per ogni valore di soglia sono stati prodotti due risultati: uno per l'algoritmo KNN e uno per l'algoritmo LSA. Un livello di soglia superiore non sarebbe stato significativo in quanto si avrebbe avuto a che fare con un numero esiguo di parole (poche decine). Per ogni soglia sono stati rilevati tre valori: il valore di recall, il valore di fallout e lo score.
Lo score è calcolato come:

$$Score = \frac{Recall}{Fallout}$$

in questo modo possiamo stimare il rapporto tra i valori di recall e fallout e individuare un indice di bontà per i rating predetti.

ICM	**Algoritmo**	**Soglia**	**Recall**	**Fallout**	**Score**
All (Stem)	KNN	0.3	**0.0741**	0.0240	3.0880
		0.7	0.0674	0.0218	**3.0942**
	LSA	2.0	**0.0221**	0.0158	**1.4035**
Nomi Propri	KNN	0.6	0.0250	0.0162	**1.5412**
		1.5	**0.0296**	0.0193	1.5303
	LSA	0	**0.0263**	0.0191	1.3767
		1.0	0.0206	0.0100	**2.0596**
Nomi (No Stem)	KNN	0.2	**0.0683**	0.0222	3.0752
		0.4	0.0680	0.0204	**3.3251**
	LSA	0.2	0.0069	0.0036	**1.9421**
		1.9	**0.0165**	0.0180	0.9161
Nomi (Stem)	KNN	0	**0.0671**	0.0187	**3.5940**
	LSA	0.4	0.0073	0.0036	**2.0425**
		1.9	**0.0152**	0.0169	0.9023

Tabella 6.1: Risultati

Nella Tabella 6.1 sono state selezionate le migliori performance ricavate per ogni combinazione. In grassetto sono stati evidenziati i risultati in cui vi è una recall maggiore o uno score maggiore, nel caso di recall e score maggiori alla stessa soglia è presenta un solo record. Per esempio, nella prima riga della tabella possiamo osservare che viene presa in considerazione la matrice ICM All (stem) relativa all'algoritmo KNN. Il valore 0.0741 alla soglia 0.3 è in grassetto in quanto è il valore massimo di recall individuato in questo test. Se prendiamo in considerazione l'algoritmo LSA sempre relativo all'ICM All (stem) notiamo una solo riga con due valori in grassetto, in quanto sia i valori di recall che quelli di score risultano massimi alla soglia 2.0.

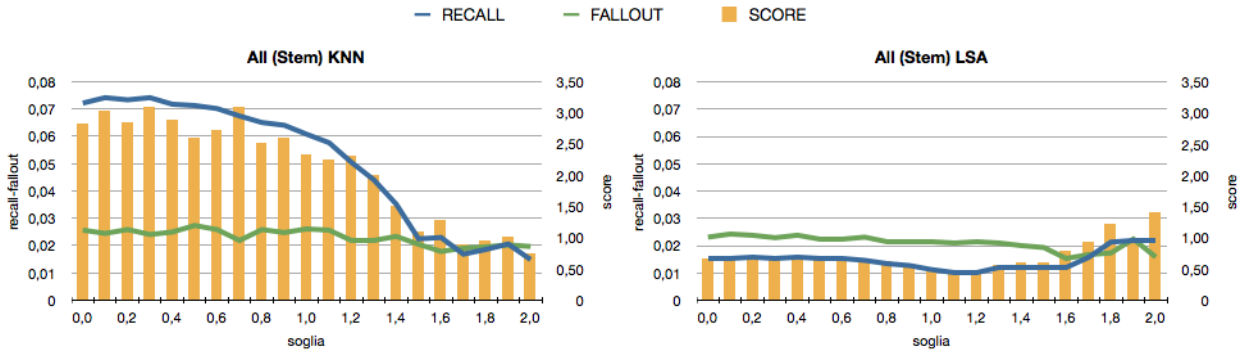

Figura 6.5: All (stem)

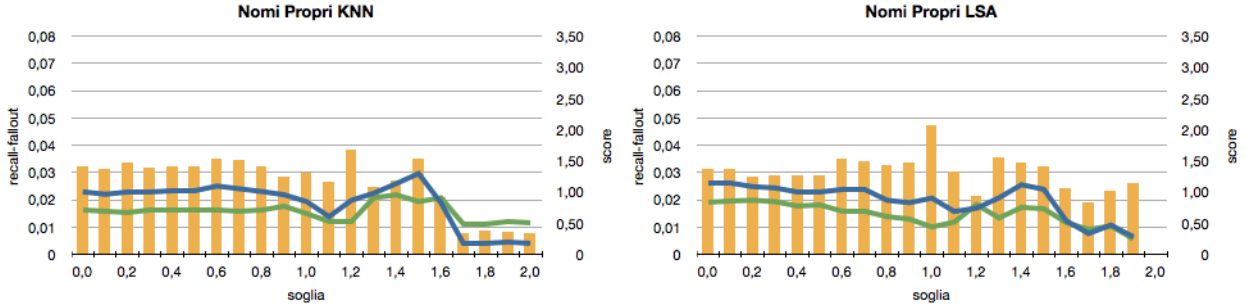

Figura 6.6: Nomi Propri

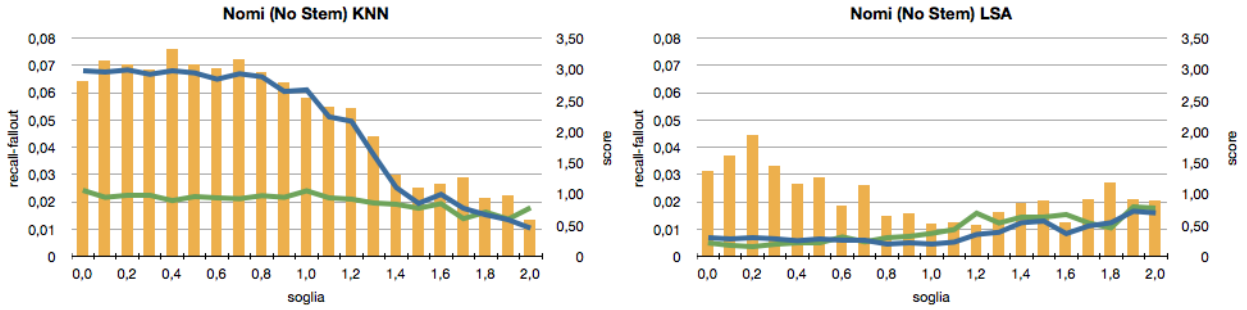

Figura 6.7: Nomi (No Stem)

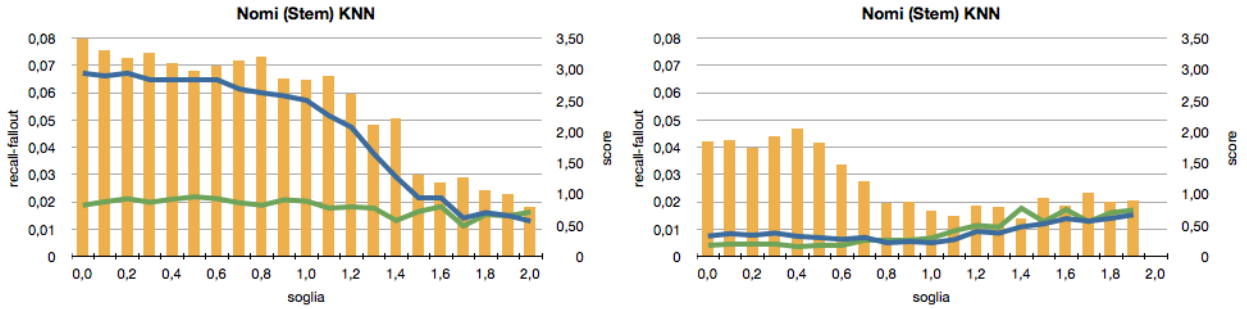

Figura 6.8: Nomi (Stem)

Nei grafici in Figura 6.5, 6.6, 6.7, 6.8, invece, è possibile osservare l'andamento delle curve di recall, di fallout e i valori di score per tutti i test effettuati. Sull'asse delle x troviamo tutti i valori di soglia per cui sono stati effettuati i test (dal valore 0 al valore 2.0), mentre sulle due assi delle y troviamo: a sinistra i valori di recall e fallout mentre a destra i valori di score (rapporto tra recall e fallout).In blu e in verde sono mostrati rispettivamente gli andamenti delle curve di recall e fallout al variare della soglia, mentre in giallo abbiamo i valori di score per ogni soglia.

6.3.1 Analisi Algoritmi

Da un'analisi generale dei dati osservati, possiamo evidenziare un prestazione nettamente inferiore dell'algoritmo LSA rispetto all'algoritmo KNN.
Questo si riscontra principalmente nei valori di recall individuati che in media risultano essere fino a sette volte inferiori rispetto ai valori ricavati attraverso l'algoritmo KNN.
Considerando che i valori di fallout non subiscono questo drastico ridimensionamento, dai risultati viene messo in risalto uno score molto basso, che in media è inferiore alla metà dello score del KNN.

6.3.2 Analisi ICM

In questo paragrafo cercheremo di individuare il miglior set di parole per le raccomandazioni. Per fare questo verranno confrontati i risultati ottenuti attraverso l'algoritmo KNN, che, come visto nel paragrafo precedente, risulta essere il più performante.
Nel set di parole "All (stem)" ritroviamo dei dati in linea con quelli presenti in letteratura. Il metodo utilizzato attualmente nei sistemi di raccomandazione content-based si presenta con una recall intorno al 7% e una fallout media intono al 2.5%. Questo ci porta ad avere uno score massimo di circa 3.09 alla soglia 0.7.
Considerando questi dati come dati di confronto, possiamo notare che l'estrazione dei soli nomi propri (Stanbol) non porta nessun miglioramento, anzi, i valori di recall diminuiscono di circa due terzi rispetto ai valori illustrati precedentemente. Nonostante ci sia anche un lieve diminuzione dei dati di fallout, dovuta probabilmente all'eliminazione di molti tipi grammaticali che portano rumore, lo score massimo individuato si attesta intorno a valori di 1.54, quasi la metà del precedente.
Le considerazioni che possiamo trarre da questi risultati, ci portano a dire che l'estrazione delle sole entità non è sufficiente ad identificare un topic e

quindi a trovare delle correlazioni significative tra item.
Le ultime due matrici prese in considerazione solo composte dai set di parole estratti attraverso il lavoro di tesi.
Il primo set corrisponde all'insieme di tutti i nomi senza stem, dai risultati di questo set vengono trovati dei valori di recall che si attestano intorno a valori di 6.8%, quindi inferiori al 7% della recall del set di parole "All (stem).
La differenza sostanziale si riscontra in un abbassamento del fallout di circa 0.5%, questo porta ad avere un miglioramento dello score, il cui valore massimo, a soglia 0.4, raggiunge un valore di 3.33.
Questo valore risulta essere migliore del valore di score del caso di confronto, questo porta ad avere un aumento delle performance nella raccomandazione.
L'ultimo set va a migliorare ulteriormente i risultati ottenuti dal set di parole di nomi senza stem, questo ci porta a dire che effettivamente l'algoritmo di stem porta dei miglioramenti nell'individuazione delle parole chiave.
In particolare, anche in questo caso, abbiamo un recall i cui valori migliori si attestano intorno a 6.7%, ma riusciamo ad ottenere un valore di fallout di circa 1.85%, inferiore rispetto al caso base di ben 0.65%. Questo porta ad avere il rapporto di score migliore dei quattro test effettuati che, a soglia 0, vale 3.59.

6.3.3 Considerazioni finali

Dai risultati si evince che l'utilizzo di nomi come entità semantiche per individuare il contesto di un testo è stata una scelta più che corretta, infatti abbiamo risultati molto simili al set di "All words", questo significa che effettivamente solo i nomi concorrono all'individuazione degli hit positivi nei rating predetti.
Inoltre, il set di parole estratte dal software sviluppato abbassa notevolmente il fallout portando quindi all'eliminazione di film con rating negativi durante il processo di raccomandazione.
Questo porterebbe a scegliere come algoritmo per un sistema reale, l'algoritmo KNN con matrice ICM corrispondente alla matrice contenente le parole estratte dal software a cui è stata applicato l'algoritmo di stem.
In questo modo avremmo un sistema che, abbassa leggermente la recall, quindi va a perdere qualche film potenzialmente raccomandabile, ma che diminuisce in maniera considerevole i film che l'utente non gradirebbe di sicuro, a causa del fallout molto basso.

Capitolo 7

Conclusioni

Nel corso del lavoro di tesi sono stati presentati i sistemi di raccomandazione contestuale. Inizialmente, sono stati definiti e proposti multipli contesti studiandone osservabilità e utilità nel dominio film.
Dopo aver definito un contesto con buoni valori di osservabilità e utilità è stato deciso di sviluppare un framework che riuscisse a proporre un insieme di item (film) che avessero delle relazioni semantiche in comune con un set di notizie lette dall'utente preso in considerazione. Per fare questo è stato necessario l'utilizzo e lo studio di strumenti semantici che ci hanno permesso di individuare un set di parole chiave (in particolare entità) per ogni item preso in considerazione.
A questo punto è stato necessario testare le parole chiave estratte per valutarne la bontà.
I set di parole estratti dal nostro software sono stati confrontati con il set di parole che di solito vengono estratte per le raccomandazioni di tipo content-based. Nonostante nessun set introduca un miglioramento a livello di valori di recall, i risultati hanno mostrato un miglioramento di performance (rapporto tra recall e fallout) per quanto riguarda le parole estratte dai set di nomi sia con e senza l'applicazione dell'algoritmo di stem. Invece, per quanto riguarda il set di soli nomi propri, non vi sono stati miglioramenti nella raccomandazione.

Alla luce degli esperimenti e dei risultati descritti, non possiamo però effettivamente determinare se la relazione news-film abbia realmente una valenza positiva per la raccomandazione. Il fatto di leggere una news non implica necessariamente che un utente sia interessato a vedere film che siano in relazione con essa. Prendiamo per esempio una notizia che parli dello scoppio di una guerra, la notizia ha una valenza talmente sensazionalistica

che verrà letta da un alto numero di utenti ma probabilmente molti di essi non saranno interessati a vedere film che trattino guerre.

Sviluppi Futuri

Per quanto appena detto si aprono due possibili esperimenti di estensione del lavoro di tesi.
Un primo lavoro, ha l'obiettivo di trovare un collegamento tra le news lette e i film, ovvero riuscire ad individuare se esiste una relazione tra news e film che soddisfi la raccomandazione contestuale. Per fare questo vanno effettuati dei test, con utenti reali, nei quali vengono proposte set di notizie e set di film per cercare di individuare delle relazioni tra i due gruppi di oggetti.
Un secondo lavoro, invece, si focalizza sull'identificazione dello User Profile attraverso le sue attività recenti. In altre parole, rimanendo sempre nel contesto news, bisognerebbe capire se attraverso un set di notizie è possibile individuare dei tratti comuni negli utenti, in maniera tale da inquadrarli in determinati gruppi sociali di cui sono noti in precedenza i gusti cinematografici. Ad esempio, se l'utente legge solo notizie di borsa e di sport potrebbe essere un giovane interessato alla finanza di un'età compresa tra i trenta e quarant'anni, quindi gli proponiamo film che hanno ricevuto rating alti da persone con un profilo simile al suo. In questo esempio si è visto come il contesto partecipi alla raccomandazione non individuando gli oggetti da proporre ma individuando il profilo dell'utente.

Bibliografia

[1]

[2] Apache stanbol semantic engine.

[3] The stanford natural language processing group.

[4] *Multilingual Single Document Keyword Extraction For Information Retrieval*, 2005.

[5] Gregory D. Abowd, Christopher G. Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide: a mobile context-aware tour guide. *Wirel. Netw.*, 3(5):421–433, October 1997.

[6] Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen, and Alexander Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Trans. Inf. Syst.*, 23(1):103–145, January 2005.

[7] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.

[8] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 335–336, New York, NY, USA, 2008. ACM.

[9] Rakesh Agrawal, Ralf Rantzau, and Evimaria Terzi. Context-sensitive ranking. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, SIGMOD '06, pages 383–394, New York, NY, USA, 2006. ACM.

[10] Giorgos Akrivas, Manolis Wallace, Giorgos Andreou, Giorgos Stamou, and Stefanos Kollias. Context-sensitive semantic query expansion. In

Proceedings of the 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS'02), ICAIS '02, pages 109–, Washington, DC, USA, 2002. IEEE Computer Society.

[11] Rasim M. Alguliev and Ramiz M. Aliguliyev. Effective summarization method of text documents. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '05, pages 264–271, Washington, DC, USA, 2005. IEEE Computer Society.

[12] Sarabjot Singh Anand and Bamshad Mobasher. From web to social web: Discovering and deploying user and content profiles. chapter Contextual Recommendation, pages 142–160. Springer-Verlag, Berlin, Heidelberg, 2007.

[13] Asim Ansari, Skander Essegaier, and Rajeev Kohli. Internet recommendation systems. pages 363–375, 2000.

[14] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[15] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, March 1997.

[16] Nicholas J. Belkin and W. Bruce Croft. Information filtering and information retrieval: two sides of the same coin? *Commun. ACM*, 35(12):29–38, December 1992.

[17] Michael J. Berry and Gordon Linoff. *Data Mining Techniques: For Marketing, Sales, and Customer Support*. John Wiley & Sons, Inc., New York, NY, USA, 1997.

[18] Daniel Billsus and Michael J. Pazzani. Learning collaborative information filters. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 46–54, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[19] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, UAI'98, pages 43–52, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[20] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, November 2002.

[21] Robin Burke. The adaptive web. chapter Hybrid web recommender systems, pages 377–408. Springer-Verlag, Berlin, Heidelberg, 2007.

[22] I. Cantador and P. Castells. Semantic contextualisation in a news recommender system. In *Workshop on Context-Aware recommender System*, 2009.

[23] Federica Cena, Luca Console, Cristina Gena, Anna Goy, Guido Levi, Sonia Modeo, and Ilaria Torre. Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. *AI Commun.*, 19(4):369–384, December 2006.

[24] S. Chatterjee and A. S. Hadi. *Regression Analysis by Example*. John Wiley and Sons, 2000.

[25] Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and olap technology. *SIGMOD Rec.*, 26(1):65–74, March 1997.

[26] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 39–46, New York, NY, USA, 2010. ACM.

[27] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 39–46, New York, NY, USA, 2010. ACM.

[28] J. Delgado. Memory-Based WeightedMajority Prediction for Recommender Systems, 1999.

[29] Paul Dourish. What we talk about when we talk about context. *Personal Ubiquitous Comput.*, 8(1):19–30, February 2004.

[30] Jeffrey D. Blume et al. Estimation and covariate adjustment of roc curves and underlying test score distributions by.

[31] P. Kotler G. L. Lilien and K. S. Scherle. *Marketing Models*. Prendice Hall, 1992.

[32] L. Getoor and M. Sahami. Using Probabilistic Relational Models for Collaborative Filtering. In *Working Notes of the KDD Workshop on Web Usage Analysis and User Profiling*, 1999.

[33] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Inf. Retr.*, 4(2):133–151, July 2001.

[34] Jonathan L. Herlocker and Joseph A. Konstan. Content-independent task-focused recommendation. *IEEE Internet Computing*, 5(6):40–47, November 2001.

[35] Will Hill, Larry Stead, Mark Rosenstein, and George Furnas. Recommending and evaluating choices in a virtual community of use. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 194–201, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[36] Xinghua Hu and Bin Wu. Automatic keyword extraction using linguistic features. In *Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, ICDMW '06, pages 19–23, Washington, DC, USA, 2006. IEEE Computer Society.

[37] Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, EMNLP '03, pages 216–223, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[38] M. F. Luce J. R. Bettman and J. W. Payne. Consumer decision making: A constructive perspective. pages 1–42, 1991.

[39] Gareth J. F. Jones. Challenges and opportunities of context-aware information access. In *Proceedings of the International Workshop on Ubiquitous Data Management*, UDM '05, pages 53–62, Washington, DC, USA, 2005. IEEE Computer Society.

[40] E. Pitoura K. Stefanidis and P. Vassiliadis. A context-aware preference database system. *International Journal of Pervasive Computing and Communications*, pages 3(4):439–600, 2007.

[41] Jasmeen Kaur and Vishal Gupta. Effective approaches for extraction of keywords. *IJCSI Internationa Journal of COmputer Science Issues*, 7(6), 2010.

[42] Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*, 37(2):18–28, September 2003.

[43] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Inc., New York, NY, USA, 2nd edition, 2002.

[44] D. Koller and M. Sahami. Toward optimal feature selection. In Morgan Kaufmann, editor, *In Proceedings of the 13th International Conference on Machine Learning*, pages 284–292, 1996.

[45] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. Grouplens: applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, March 1997.

[46] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 426–434, New York, NY, USA, 2008. ACM.

[47] Ken Lang. NewsWeeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995.

[48] David B. Leake and Ryan Scherle. Towards context-based search engine selection. In *Proceedings of the 6th international conference on Intelligent user interfaces*, IUI '01, pages 109–112, New York, NY, USA, 2001. ACM.

[49] Sungjick Lee and Han-Joon Kim. News keyword extraction for topic tracking. In *Proceedings of the 2008 Fourth International Conference on Networked Computing and Advanced Information Management - Volume 02*, NCM '08, pages 554–559, Washington, DC, USA, 2008. IEEE Computer Society.

[50] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, February 1994.

[51] Huan Liu and Hiroshi Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[52] Benjamin Marlin. Modeling user rating profiles for collaborative filtering. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

[53] Nikolaos F. Matsatsinis and Andreas P. Samaras. Mcda and preference disaggregation in group decision support systems. *European Journal of Operational Research*, 130(2):414 – 429, 2001.

[54] Y. Matsuo and M. Ishizuka. Keyword extraction from a single document using word co-occurence statistical information. *International Journal of Artificial Intelligence Tools*, 13(1):157–169, 2004.

[55] User Modeling and User-Adapted Interaction staff. User modeling and user-adapted interaction. *User Modeling and User-Adapted Interaction*, 12(1):97–104, February 2002.

[56] Mohamed F. Mokbel and Justin J. Levandoski. Toward context and preference-aware location-based services. In *Proceedings of the Eighth ACM International Workshop on Data Engineering for Wireless and Mobile Access*, MobiDE '09, pages 25–32, New York, NY, USA, 2009. ACM.

[57] Masahiro Morita and Yoichi Shinoda. Information filtering based on user behavior analysis and best match text retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '94, pages 272–281, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[58] J. Pascoe N. Ryan and D. Morse. Enhanced reality fieldwork: the context-aware archeological assistant. *Computer Applications in Archaeology*, 1997.

[59] Atsuyoshi Nakamura and Naoki Abe. Collaborative filtering using weighted majority prediction algorithms. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 395–403, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[60] Yukio Ohsawa, Nels E. Benson, and Masahiko Yachida. Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In *Proceedings of the Advances in Digital Libraries Conference*, ADL '98, pages 12–, Washington, DC, USA, 1998. IEEE Computer Society.

[61] Kenta Oku, Shinsuke Nakajima, Jun Miyazaki, and Shunsuke Uemura. Context-aware svm for context-dependent information recommendation. In *Proceedings of the 7th International Conference on Mobile Data Management*, MDM '06, pages 109–, Washington, DC, USA, 2006. IEEE Computer Society.

[62] J. D. Bovey P. J. Brown and X. Chen. Context-aware applications: from the laboratory to the marketplace. *IEEE Personal Communications*, pages 4:58–64, 1997.

[63] Cosimo Palmisano, Alexander Tuzhilin, and Michele Gorgoglione. Using context to improve predictive modeling of customers in personalization applications. *IEEE Trans. on Knowl. and Data Eng.*, 20(11):1535–1549, November 2008.

[64] Umberto Panniello, Alexander Tuzhilin, Michele Gorgoglione, Cosimo Palmisano, and Anto Pedone. Experimental comparison of pre- vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 265–268, New York, NY, USA, 2009. ACM.

[65] Dmitry Y. Pavlov and David M. Pennock. A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In *In Proceedings of Neural Information Processing Systems*, pages 1441–1448. MIT Press, 2002.

[66] Michael Pazzani and Daniel Billsus. Learning and revising user profiles: The identification of interesting web sites. *Mach. Learn.*, 27(3):313–331, June 1997.

[67] Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, 13(5-6):393–408, December 1999.

[68] David M. Pennock, Eric Horvitz, Steve Lawrence, and C. Lee Giles. Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, UAI '00, pages 473–480, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[69] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer*

supported cooperative work, CSCW '94, pages 175–186, New York, NY, USA, 1994. ACM.

[70] Elaine Rich. Readings in intelligent user interfaces. chapter User modeling via stereotypes, pages 329–342. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[71] Stephen Robertson. Understanding inverse document frequency: on theoretical arguments for idf. *Journal of Documentation*, 60(5):503–520, 2004.

[72] J. Rocchio. *Relevance Feedback in Information Retrieval*, pages 313–323. 1971.

[73] Gerard Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[74] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of dimensionality reduction in recommender systems: A case study. In *WebKDD Workshop at the ACM SIGKKD*, 2000.

[75] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM.

[76] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, pages 253–260, New York, NY, USA, 2002. ACM.

[77] B. N. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *Netwrk. Mag. of Global Internetwkg.*, 8(5):22–32, September 1994.

[78] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating ‘word of mouth’. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 210–217, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[79] B. Sheth and P. Maes. Evolving agents for personalized information filtering. pages 345–352, March 1993.

[80] Ahu Sieg, Bamshad Mobasher, and Robin Burke. Representing context in web search with ontological user profiles. In *Proceedings of the 6th international and interdisciplinary conference on Modeling and using context*, CONTEXT'07, pages 439–452, Berlin, Heidelberg, 2007. Springer-Verlag.

[81] Loren Terveen, Will Hill, Brian Amento, David McDonald, and Josh Creter. Phoaks: a system for sharing recommendations. *Commun. ACM*, 40(3):59–62, March 1997.

[82] Peter D. Turney. Learning algorithms for keyphrase extraction. *Inf. Retr.*, 2(4):303–336, May 2000.

[83] L. Ungar and D. Foster. Clustering methods for collaborative filtering. In *Proceedings of the Workshop on Recommendation Systems*. AAAI Press, Menlo Park California, 1998.

[84] Zhiwen Yu, Xingshe Zhou, Daqing Zhang, Chung-Yau Chin, Xiaohang Wang, and Ji Men. Supporting context-aware media recommendations for smart phones. *IEEE Pervasive Computing*, 5(3):68–75, July 2006.

[85] Yi Zhang, Jamie Callan, and Thomas Minka. Novelty and redundancy detection in adaptive filtering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, pages 81–88, New York, NY, USA, 2002. ACM.

[86] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 22–32, New York, NY, USA, 2005. ACM.