Sistemi intelligenti per Internet

Psw: sii2024

Slides 1

Information need -> retrieval -> recommendation (in push o pull)

Recommendation thru recommender system RS

Motori di ricerca - search

- Una search su/per un documento può cercare un contenuto significativo oppure una struttura significativa desiderata del documento stesso
- Il confronto tra campi di un doc è più facile del confronto su NL a causa del vocabulary mismatch problem (problemi di polisemia, sinonimia, iperonimia, iponimia)
- Indicizzazione del web
- Task (vedi slides)
- Progettazione di sistemi di raccomandazione: rilevanza, valutazione, information need

Task IR

- Ricerca ad hoc
- Filtraggio (identificazione doc rilevanti ad esempio in uno stream)
- Classificazione (etichette rilevanti)
- Question answering

Rilevanza: topical relevance / user relevance

Valutazione (della rilevanza): servono benchmark di riferimento, ad esempio dei doc di rifermento (corpus) con query e doc rilevanti associati (i più famosi sono i TREC). Le metriche usate sono precision e recall.

Sora, midjourney, prompt engineering

Tip of the tongue track

Medline

Problematiche dei motori di ricerca:

- Performance
- Incorporazione nuovi dati (recency, freshness)
- Scalability
- Adaptability
- Specific problems (spam: misleading or non relevant info in doc designed for some commercial benefit, adversarial IR)

Slides 2

Processo di interrogazione:

- User interaction
- Evaluation
- Rating

User interaction

- Query input
- Query transformation include tecniche per migliorare la query iniziale (tokenising, stopping, stemming), sia prima che dopo la creazione del ranking, come anche trasformazione del testo, spell checking, query suggestion, query expansion, relevance feedback
- Result output

il result output include task come la generazione di snippet che riassumono i doc recuperati, evidenziazione di parole o passaggi rilevanti in tali doc, raggruppamento dei doc in gruppi correlati, visualizzazione di pubblicità rilevanti, traduzione in linguaggio opportuno

- Scoring: è la componente che calcola il punteggio per i doc recuperati utilizzando un algoritmo di ranking, in cui la forma di base di un punteggio è $\sum q_i d_i$ con qi e di pesi della query e del documento per un termine i.
- Performance optimisation: si vuole diminuire il response time e aumentare il query throughput
- Distribution: si vuole processare le query in ambienti distribuiti, per poi assemblare i risultati con un query broker.

Evaluation

- Logging (log delle query, analisi dei click, query caching, ranking, ricerca di pubblicità)
- Ranking analysis
- Performance analysis

Slides 3

Crawler (o spider)

Oggetto software che trova e scarica pagine web automaticamente, dal momento che il web è costantemente aggiornato ed ampliato. I crawler vengono gestiti dai motori di ricerca per raccogliere le informazioni richieste.

Ogni pagina web ha un suo unico URL (uniform resource locator) che ha tre parti: scheme, host name e resource name, ad esempio http://www.cs.umass.edu/csinfo/people.html . Le pagine web sono memorizzate su web server che usano il protocollo http (hypertext transfer protocol) per scambiare info con i software client.

Browser web e crawler web sono due diversi tipi di client web. Il browser web è ciò che permette all' user di interagire con le pagine web. I crawler web si connettono ad un server DNS che risolve i nomi, cioè traduce l' host name in un indirizzo IP. A questo punto il crawler si connette col server attraverso una porta specifica (http ha la porta 80) e si connette quindi con la pagina. Inizia a seguire i link della pagina, anche quelli che portano in altre pagine.

La richiesta HTTP più comune è la GET: GET /csinfo/people.html HTTP/1.0 , con la quale si chiede al server di inviare la pagina richiesta al client, utilizzando la versione 1.0 del protocollo HTTP. Il server invia l'header e il contenuto di quel file al client. Se il client desidera più pagine, invia richieste aggiuntive, quindi scarica pagine web e aggiunge in coda/frontiera gli url di altre pagine da visitare (queste possono essere scelte in modo casuale o focused). Il processo continua finchè non ci sono più url nella coda o si è esaurito lo spazio per memorizzare le pagine. Le pagine scaricate sono lette e analizzate attraverso parsing per trovare i nuovi url da aggiungere.

Dal momento che il crawler passa molto tempo in attesa di risposta alle sue richieste http, è un software multithread. Inoltre, dal momento che potenzialmente i crawler possono creare problemi di flooding alle pagine web, sono sottoposti a politiche di politeness. Il crawling può essere controllato dagli amministratori di web server tramite il file robots.txt, attraverso il quale si può vietare o meno il crawling di alcune pagine e risorse (es disallow: /private/). Il file sitemap invece è un file zippato che fornisce al crawler la topologia della pagina.

(CODICE)

Strategie di crawling: breadth first, depth first

Una metrica della qualità è la freshness: i crawler devono rivisitare pagine già visitate per capire se ci sono stati cambiamenti che quindi renderebbero la versione della pagina visitata precedentemente "stale". Il protocollo HTTP ha un tipo speciale di richiesta detta HEAD che rende facile il controllo sui cambiamenti delle pagine, richiedendo solo l'intestazione delle pagine e non la pagina stessa. Il problema della freshness (la frazione di pagine che sono fresh) è che seguendola si rischiano scelte sbagliate, per questo si considera l'age.

Il focused crawling si concentra (e quindi scarica) sulle pagine che trattano di un particolare argomento, basandosi sul fatto che pagine che trattano di un argomento conterranno link ad altre pagine che parlano dello stesso argomento. Le pagine vengono conservate solo se sono on topic (questo viene valutato tramite classificazione dei testi) e se lo è si utilizzano i link che contiene per continuare l'esplorazione. Questi link vengono giudicati sulla base della topicality, soprattutto basandosi sul loro anchor text.

Il deep o hidden web è lo spazio del web difficilmente analizzabile. La maggior parte dei siti del deep web rientrano nelle categorie di: private site (volutamente privato, nessun link di provenienza, può chiedere login), form result (siti che richiedono di inserire dati in un form), scripted page (pagine che usano script/linguaggi client side per generare link).

Il crawling distribuito consiste nell'utilizzo di molteplici calcolatori per fare crawling. In questo modo il crawler può essere più "vicino" ai siti che analizza, deve ricordarne meno e può risparmiare tempo di calcolo per l'analisi delle risorse. Si usa una funzione di hash per assegnare gli URL ai calcolatori che si occupano di crawling.

Il desktop crawling è utilizzato per la ricerca su desktop e la ricerca aziendale. È molto più facile trovare dati (si può immaginare un file system come un web server, ma con un site map generato automaticamente). Quando si parla di documenti pubblicati, cioè creati in un dato istante e raramente aggiornati, e sono pubblicati dalla stessa risorsa, questi possono essere ordinati in una sequenza ordinata chiamata document feed, facile da accedere per il web crawler perché questo dovrà analizzare solo la parte finale del feed. Ci sono due tipi di document feed: il push feed che avverte il sottoscrittore quando viene pubblicato un nuovo documento, e il pull feed che richiede che l'utente controlli periodicamente i nuovi documenti pubblicati (il cui formato più comune è detto RSS, rich site summary o RDF site summary; una delle caratteristiche più interessanti di questo formato è il tag ttl che indica il tempo espresso in minuti entro il quale il contenuto dovrebbe essere memorizzato. Un'informazione disponibile da più di un'ora dovrebbe quindi essere considerata stale).
Slides 4
Si memorizzano i documenti trasformati in testo per risparmiare risorse di crawling e per avere a disposizione un accesso efficiente al testo. I requisiti funzionali per il sistema di memorizzazione sono:

- Accesso casuale
- Compressione dei file di grande dimensione
- Aggiornamento (necessità di lavorare su grandi volumi di documenti nuovi e modificati)

L'idea fondamentale per la gestione di questa mole di dati è quella di memorizzare documenti in grandi file, anziché usare un documento per ogni file, comprimendo i file (algoritmi DEFLATE, LZW) e sfruttando le informazioni semi strutturare da raccogliere/mantenere disponibili. Un sistema di memorizzazione di dati deve processare continuamente differenti porzioni di dato, eseguire operazioni di scrittura e lettura con altissima frequenza, scansionare efficientemente i tutti i sottoinsiemi di dati interessanti, esaminare i cambiamenti nel tempo..

Big table è una mappa multilivello distribuita scalabile e self managing: è un sistema di database distribuito, creato per la memorizzazione di pagine web. Un'istanza è una grande tabella (una sola per ogni database) divisa poi in piccole porzioni chiamate tablet, servite da migliaia di macchine. Una "BigTable" è una mappa multilivello, ordinata, distribuita, persistente e multidimensionale. La mappa è indicizzata mediante tre parametri: una "row key", una "column key" e un "timestamp". Questa tripletta identifica il contenuto memorizzato (es una pagina html). Ogni valore nella mappa è un array di byte non interpretato. La tabella può avere un numero illimitato di colonne. Le column key (sintassi: family:qualifier) sono raggruppate in insieme chiamati column family.

L’architettura logica è composta da tre moduli

- Library code (per ogni client)
- Master Server
- Tablet Server

Ogni Tablet Server parte con una singola tablet. Quando la dimensione di questa tablet supera una soglia, viene suddivisa in due tablet. Le locazioni delle tablet sono memorizzate utilizzando un B-Tree (per modellare la gerarchia). BigTable usufruisce di un sistema di alta affidabilità per i lock distribuiti chiamato Chubby.

Ricerca di duplicati: i duplicati sono un elemento critico perché consumano risorse durante crawling, indicizzazione e ricerca. Le tecniche per la ricerca di duplicati sono diverse a seconda che si cerchi un duplicato esatto o simile. Per il duplicato esatto il ritrovamento è più semplice e la ricerca si basa su tecniche di checksum. La ricerca dei near duplicate è un task più complicato e si basa sulla definizione di un valore di soglia su una misura di similarità tra coppie di documenti. Il rilevamento di near duplicate si divide in

- Search: trovare tutti i near duplicate di un documento D O(N)
- Discovery: trovare tutte le coppie di near duplicate nella collezione O(N^2)

Per la gestione della discovery di near duplicates lo sviluppo di nuove tecniche ha portato alla definizione di rappresentazioni più compatte per i documenti, come le fingerprint, generate come segue:

1. Il documento è letto e analizzato (parsato) in parole. Il contenuto non composto da parole, come punteggiatura, tag HTML e spazi aggiuntivi, è rimosso.
2. Le parole sono raggruppate in n-grammi contigui per un certo n. Di solito si tratta di sequenze di parole sovrapposte, sebbene alcune tecniche utilizzino sequenze non sovrapposte.
3. Alcuni degli n-grammi sono selezionati per rappresentare il documento.
4. Gli n-grammi selezionati sono sottoposti ad hashing per migliorare l'efficienza del processo di retrieval e ridurre ulteriormente le dimensioni della rappresentazione
5. I valori hash sono memorizzati, in genere in un indice inverso.
6. I documenti sono confrontati fra loro utilizzando la sovrapposizione di fingerprint

Le comparazioni di similarità che sfruttano rappresentazioni word-based sono molto efficaci ma molto costosi. Simhash combina i vantaggi della similarità basata sulla misura Word-based con l’efficienza del fingerprint basato su hashing. Ha la proprietà insolita che documenti simili hanno valori si hash simili. Più precisamente, la similarità fra due pagine misurata dalla misura di correlazione del coseno (similarità basata sul contenuto) risulta proporzionale al numero di bit che sono gli stessi nei fingerprint generati da simhash.

Fingerprint Simhash:

1. Elaborare il documento in un insieme di feature con pesi associati. Assumeremo il caso semplice in cui le feature sono parole pesate in base alla loro frequenza.
2. Generare un valore hash con b bit (la dimensione desiderata del fingerprint) per ogni parola. Il valore hash dovrebbe essere univoco per ogni parola.
3. Nel vettore b-dimensionale V, aggiornare le componenti del vettore aggiungendo il peso di una parola a ogni componente per il quale il bit corrispondente nel valore hash della parola è 1 e sottraendo il peso se il valore è 0.
4. Dopo che tutte le parole sono state elaborate, generare un fingerprint di b-bit impostando l'i-esimo bit a 1 se l'i-esima componente di V è positiva, o 0 altrimenti?

Le pagine web contengono spesso testo, link, immagini non direttamente collegate al contenuto principale della pagina. Questo è detto rumore, che viene rilevato per poter essere ignorato, concentrandosi sui blocchi di contenuto rilevante nella pagina. L’identificazione di content block viene fatta per esempio tramite analisi della distribuzione dei tag HTML, oppure esaminando l’albero DOM (document object module) che rappresenta la struttura della pagina HTML.

Slides 5

- L'ordine delle chiavi nel modello dati big table è: row key, column family, column qualifier, timestamp
- Le seguenti affermazioni su big table sono corrette:

1. I dati sono mantenuti in ordine lessicografico
2. Sono supportate le transazioni su singola riga
3. I file sono memorizzati utilizzando Google File System (GFS)

- la struttura dati utilizzata per memorizzare le informazioni sulla posizione del tablet è detta B-Tree

HBase vedi slides

Slides 6

Elaborazione del testo

Risulta utile convertire il documento in index term, così da migliorare l'efficacia del retrieval (che non può basarsi esclusivamente sulla rigida corrispondenza tra la stringa di caratteri della query e testo originale del documento), gestendo il fatto che le parole non sono tutte di ugual valore nella ricerca e che spesso può non essere chiaro dove le parole iniziano e finiscono.

Inoltre i modelli di retrieval e di ranking dipendono fortemente dalle proprietà statistiche delle parole (le parole importanti occorrono spesso all'interno di un documento ma non sono frequenti nell'intera collezione). La distribuzione delle frequenze delle parole all'interno di un testo è skewed/distorta: ci sono poche parole che hanno frequenze molto alte e molte parole con frequenze basse. Questa distribuzione è descritta dalla legge di Zipf: la frequenza della r-esima parola più comune è inversamente proporzionale a r o, in alternativa, il rank di una parola moltiplicato per la sua frequenza (f) è all'incirca una costante (k): r*f=k.

La tokenizzazione o analisi lessicale è il task che si occupa di riconoscere ogni occorrenza di una parola nella sequenza di caratteri di un documento (parole sono di >2 caratteri, terminano con spazio o caratteri speciali, e tutti i caratteri sono trasformati in minuscoli)

Problemi della tokenizzazione:

- Parole di 1-2 caratteri non vengono considerate parole ma possono essere importanti
- Sillabazioni (presenza o meno di trattino può portare confusione)
- Caratteri speciali
- Parole maiuscole con significato diverso dalle minuscole
- Apostrofi
- Numeri
- Punti per abbreviazioni

Il processo di tokenizzazione più generale comporta prima l'identificazione della struttura del documento e quindi l'identificazione delle parole nel testo come una qualsiasi sequenza di caratteri alfanumerici, terminata da uno spazio o da un carattere speciale, tutta convertita in minuscolo, quindi non troppo diverso dal processo di tokenizzazione dei primi sistemi IR. In molti casi vengono aggiunte regole al tokeniser per gestire alcuni dei caratteri speciali.

Stopping: alcuni termini (preposizioni, articoli) non costituiscono informazione se presi singolarmente quindi possono essere rimossi; a volte però possono essere importanti in particolari combinazioni. Esistono liste standard per specifiche applicazioni di termini che possono essere rimossi, oppure se ne creano di personalizzate. Le parole che possono essere rimosse sono comunque da scegliere tra quelle più frequenti nella collezione di documenti.

Stemming: ha come scopo quello di ridurre le variazioni morfologiche di una parola riportandola ad una radice comune (es rimozione dei suffissi). La realizzazione di stemmer segue generalmente due approcci: algoritmico (si utilizza un programma per decidere se due parole sono correlate tra loro in base alla conoscenza dei suffissi della lingua specifica)

oppure dictionary based (non segue una particolare logica ma si basa su dizionari pre-creati di termini correlati per memorizzare relazioni tra i termini. Siccome questi elenchi possono essere creati da esperti del dominio, il tasso di falsi positivi può essere molto basso. Il problema è che la dimensione di questi dizionari non può essere infinitamente grande). Lo stemmer algoritmico più noto è lo stemmer di Porter: consiste in una serie di passi, ciascuno contenente un insieme di regole per rimuovere i suffissi. Ad ogni passo è eseguita la regola per il suffisso più lungo applicabile. Produce quindi stem e non parole. Lo stemmer di Krovetz è invece uno stemmer ibrido: fa uso costante di un dizionario per verificare se la parola è valida, e produce parole complete anziché stem.

Le frasi sono aggregazioni di termini, più precise di singole parole e meno ambigue. Il problema delle frasi riguarda la loro identificazione, che prevede tre possibili approcci: uso di un Part-Of-Speech (POS), uso di n-grammi di parola, memorizzazione delle posizioni delle parole degli indici e uso di operatori di prossimità nelle query.

POS tagging: Un POS tagger contrassegna le parole in un testo con tag corrispondenti alla parte del discorso della parola in quel contesto, cioè fa l'analisi logica del testo. I POS tagger usano approcci statistici o basati su regole per la predizione dei tag sintattici delle parole e sono addestrati utilizzati ampi corpora etichettati manualmente.

N-grammi: il POS tagging è troppo lento per grandi collezioni quindi viene semplificata la definizione di frase: la frase è una qualunque sequenza di n parole. Gli n-grammi frequenti hanno maggiori probabilità di essere frasi significative, e seguono la distribuzione di Zipf.

Slides 7
Alcune parti di un documento sono più importanti di altre, per questo i parser ne riconoscono la struttura utilizzando i markup come i tag di html e rendono quindi indicizzabili le varie parti.

Modello random surfer: è il modello secondo il quale l'utente segue sul web un percorso casuale, governato da due regole di base: nonostante la casualità l'utente visiterà più spesso le pagine più popolari, mentre l'opzione "surprise me" garantirà la visita di tutte le pagine. Il pagerank di una pagina è la probabilità che un random surfer la visiti.

La link analysis (es algoritmo PageRank) sfrutta la codifica di un link in html secondo la codifica: Example website in cui Example website è l'anchor text mentre http.//example.com è il link di destinazione. L'anchor text fornisce una descrizione breve e sintetica della destinazione del link ed è molto utile per l'efficacia della ricerca.
L'algoritmo PageRank si basa sul modello random surfer: l'utente naviga senza fine e metà tra le pagine web. Ogni volta che è su una pagina web, ha due alternative: seguire uno dei link presenti a caso o scegliere l'opzione surprise me che lo porterà su una nuova pagina web a caso. Il calcolo del PageRank corrisponde a trovare la distribuzione di probabilità stazionaria di un random walk sul web graph. Il random walk è un caso particolare di Markov chain in cui lo stato successivo (pagina visitata successiva) dipende solo dallo stato corrente (pagina corrente), in cui le transizioni consentite tra gli stati (quelle date dai link) sono tutte equi probabili.
Modello random surfer (scrittura più formale):

```
while true
    r <- numero casuale tra 0 e 1
    se r < lambda (lambda è solitamente molto basso)
        surprise me (va in una pagina casuale)
    se r >= lambda
        clicca su un link a caso nella pagina corrente
```

L’opzione surprise me impedisce di rimanere bloccati in pagine senza link, o con link che non puntano ad altre pagine o formano loop (loop con queste due problematiche sono detti dangling links).
Il page rank per qualsiasi pagina u è calcolato come segue: PR(u) = $\frac{\lambda}{N} + (1-\lambda) \cdot \sum_{v \in B_u} \frac{PR(v)}{L_v}$
Dove N è il numero di pagine totali, lambda vale tipicamente 0.15, Bu è l’insieme di pagine che puntano a u e Lv è il numero di link in uscita da una pagina v senza contare i duplicati. Il fattore 1-lambda è detto damping factor.
In conclusione, PageRank è un importante esempio di metadato indipendente dalle query che può migliorare il ranking delle pagine web: le pagine infatti possono avere lo stesso valore di pagerank indipendentemente dalla query elaborata, e a parità di query i motori di ricerca che impiegano PageRank preferiranno pagine con un PageRank più alto.

HITS: Hypertext Induced Topic Search è un algoritmo per la link analysis che stima il valore del contenuto di una pagina (valore di authority) e il valore dei link ad altre pagine (valore di hub), calcolati tramite un algoritmo iterativo basato solo sulla struttura dei link, come PageRank. HITS calcola tali valori solo per un sottoinsieme di pagine recuperate da una data query, che può renderlo computazionalmente impossibile per motori di ricerca con alto traffico di query.

La qualità dei link può essere affetta da spam o da altri fattori: linkfarm per incrementare pagerank, link di traceback nei blog per creare loop, link di spam inseriti nella sezione commenti di blog popolari. Due soluzioni possibili sono rendere i motori di ricerca in grado di individuare le sezioni di commenti e ignorarne il contenuto, e chiedere ai proprietari di siti web di modificare i link non importanti in modo che i motori di ricerca possano ignorarli (funzione dell’attributo ai link rel=nofollow).

Information extraction: sono algoritmi che permettono di estrarre in maniera automatica strutture dal testo. Sono utilizzati in vari contesti, specialmente di text data mining. Nelle applicazioni di search sono utilizzati per identificare features da impiegare poi nei motori di ricerca per migliorare il ranking. Tra questi vi sono gli algoritmi di Named Entity Recognition (NER) che identificano named entity, ossia una parola/sequenza di parole nel testo usate per far riferimento a qualcosa di interessante in una specifica applicazione. I due approcci principali per la realizzazione di NER sono quello ruled based e quello statistico.

NER ruled based: usano uno o più lexicon (elenchi) di parole e frasi che categorizzano nomi, e utilizzano regole per trovare nuove entità non presenti negli elenchi. Tali regole possono essere sviluppate manualmente da un esperto del dominio o tramite un processo di machine learning a partire da un seed di regole precedentemente decise.
NER statistici: usano un modello probabilistico della frequenza delle parole in ed intorno a una entity. La probabilità è stimata usando un insieme di dati di training annotati manualmente. HMM hidden markov model è un possibile approccio.

HMM è un modello che utilizza il contesto per identificare le entity, sfruttando le caratteristiche generative della sequenza di parole e quindi la proprietà di Markov: la parola successiva in una sequenza dipende solo da un numero ridotto delle parole precedenti. HMM è un modello che studia le probabilità e le utilizza nell’analisi di un grafo, in cui si analizzano in sequenza i vari token e il livello di probabilità che li fa spostare da un evento all’altro. Questo modello descrive un processo come un insieme di stati con transizioni tra essi, ogni transizione con un livello di probabilità associato. Ogni stato ha un insieme di possibili output che possono essere generati, anch’essi con una probabilità associata.

Slides 9
Evoluzione del web: dal primo sito nel 1991 al miliardo e otto di siti attivi (dato 2021), anche grazie alla trasformazione della telefonia dal fisso al mobile, il web si è esponenzialmente ampliato. Si è passati dal web 1.0 (in cui i siti erano collezioni di documenti html non interattivi, prodotti direttamente dal publisher per avere un’interazione diretta con l’utente con l’obiettivo di un servizio chiuso, es corporate web site, portali e motori di ricerca, e-commerce come eBay) al Web 2.0

in cui gli utenti passano da essere fruitori passivi di informazioni digitali a produttori e consumatori di conoscenza in rete, protagonisti attivi e partecipativi che creano UGC User Generated Content e lo scambiano grazie alla forte interazione tra utenti del tipo uno a molti, es blog, e molti a molti, es social media delle tipologie social networking site o content sharing site. Caratteristiche fondamentali del web 2.0 sono quindi la creazione collettiva, lo user rating nell'ecommerce, il tagging, l'organizzazione di materiale tra preferiti, cartelle ecc).

Slides 10

Social search: è una ricerca in un ambiente sociale, definito come una comunità di utenti che partecipano attivamente nel processo di search. È una definizione ampia che copre tutte le applicazioni che richiedano di definire gli interessi dei singoli utenti, interagire con altri utenti, modificare le rappresentazioni degli oggetti di ricerca. Si differenzia da un classico paradigma di search perché l'utente interagisce con il sistema e con altri utenti, in maniera implicita o esplicita, avendo quindi un ruolo attivo che non è più limitato solamente alla sottomissione di query. La social search si basa sul mondo online, che è un ambiente fortemente socialmente molto ricco di interazioni. Questo genera sia enormi quantità di dati che si basano su queste interazioni sia ovviamente le questioni di privacy che l'uso di queste risorse solleva. Gli aspetti della social search che ci interessano sono quelli collegati allo sviluppo di search engine e di information retrieval. Utilizzi:

- Tag utenti, indicizzazione manuale
- Ricerca all'interno di comunità
- Document filtering (DF)
- Recommender system (RS)
- Peer to peer (P2P) e Metasearch
- Personalization

Tag utenti e indicizzazione manuale: molti siti web (es social media) consentono agli utenti di assegnare tag ad item. L'insieme di questi tag non sono una tassonomia (cioè una classificazione definita da regole ben precise) ma riflettono comunque un punto di vista di chi li definisce, in ciò che è stata definita folksonomia, una tassonomia basata su criteri individuali, le scelte e i gusti delle persone che taggano i contenuti, senza la struttura rigida e gerarchica della tassonomia e senza controllo di autorità. La folksonomia si collega al concetto di grafo, la tassonomia a quello di albero. La folksonomia può essere broad (alla stessa risorsa sono attribuiti più tag da più persone, si avranno più tag per la stessa risorsa) o narrow (le risorse sono aggregate da poche persone con un unico termine, sempre lo stesso. Sono più precise e i dati sono più facilmente recuperabili). I tag inclusi in una folksonomia possono essere espliciti (quelli che un utente assegna dichiarando un commento o una valutazione) o impliciti (quelli che derivano dalla navigazione in Internet), e possono essere a singolo o multi termine.

La ricerca all'interno di comunità riguarda delle comunità online, gruppi virtuali di entità (utenti) che condividono interessi comuni e interagiscono socialmente in vari modi all'interno dello stesso ambiente virtuale, a cui si accede solitamente senza alcun meccanismo formale (questo vuol dire che spesso l'appartenenza è implicita). In questo tipo di ricerca è importante determinare automaticamente quali comunità esistono all'interno di un ambiente, e definire quali utenti appartengono a ciascuna di essere e come essi interagiscono tra loro, attraverso interazioni passive (es lettura di pagine web) o attive (scrittura su blog o forum). Un tipo di interazione attiva è il Community Based Question Answering.

Il filtraggio di documenti (DF), insieme ai recommender system, non ricadono in tipiche applicazioni del web 2.0 ma entra e si basano su rappresentazioni individuali di utenti dette profili e rientrano quindi nella definizione più ampia di social search. Entrambi combinano elementi di document retrieval e classificazione. Nella search standard i sistemi restituiscono documenti i risposta a molte differenti query, che corrispondono tipicamente a short term information need, mentre nel filtering esiste una query fissa (il profilo) che rappresenta un certo long term information need. Il sistema di search

monitora i documenti in ingresso e recupera solo quelli rilevanti per l'information need. In questo modo la ricerca di informazioni non avviene in modalità pull (l'utente cerca attivamente articoli di interesse) ma push (al sistema di ricerca è assegnato il compito di trovare eventuali documenti rilevanti che matchano l'information need).

Sistemi di raccomandazione (RS): simili ai sistemi di DF ma l'obiettivo consiste nel predire come un utente valuterebbe un certo item (di cui ancora non ha preso visione) sulla base dei rating forniti da utenti simili (collaborative filtering) di una stessa comunità virtuale con interessi correlati.

Peer-to-peer: è una ricerca che ha come task quello di interrogare una comunità di "nodi" (individui, organizzazioni, macchine, motori di ricerca) rispetto ad un certo information need. Quando un utente sottomette una query questa è trasmessa attraverso la rete P2P ed eseguita su uno o più nodi, e poi i risultati sono restituiti.

Metasearch: è un caso speciale di ricerca P2P in cui tutti i nodi sono motori di ricerca. Un metasearch engine esegue query su più motori di ricerca per poi collezionare tutti i risultati e fonderli tra loro per arrivare ad un risultato più accurato rispetto a quello fornito da un singolo search engine.

La personalisation è un'altra area parte della social search che copre un ampio spettro di tecniche che hanno l'obiettivo di migliorare la ricerca rappresentando preferenze e interessi del singolo utente.

Tag utenti e indicizzazione manuale:

L' indicizzazione manuale è il processo attraverso il quale da una collezione di items, ad esempio libri, vengono estratte manualmente delle features, es titolo, autore ecc, che poi vengono sfruttate per compilare cataloghi che possono essere consultati sulla base della query e delle informazioni che abbiamo per eseguire la nostra ricerca. Questo tipo di ricerca si basa sul concetto informatico di ontologia, ovvero la "rappresentazione formale, condivisa ed esplicita di una concettualizzazione di un dominio di interesse". Un'ontologia è l'unione di una tassonomia e di un insieme di relazioni, vincoli e regole. Una tassonomia è a sua volta l'insieme di un vocabolario e di una struttura. Un ontologia può essere per esempio, come forma di rappresentazione della conoscenza, una rete semantica di concetti di un dominio.

Ad oggi, vista l'impossibilità di indicizzare manualmente le collezioni di media digitali disponibili online, i motori di ricerca si avvalgono di tecniche di automatic indexing, che consistono nell'assegnare identificatori ai documenti durante la costruzione di indici. L'automatic indexing può essere meno accurato di quello manuale ma risulta essere esaustivo e consistente, dando ottimi risultati quando poi viene applicata la search. Come compromesso tra l' indexing manuale e automatico, i social media site danno la possibilità agli utenti di taggare manualmente gli item. Questi tag, in quanto generati da utenti non necessariamente esperti (scelta basata su folksonomie) e privi di un controllo di qualità, generano tag necessariamente noisy e possibilmente inesatti. Nonostante la presenza del semantic web, o web 3.0, il cui obiettivo è quello di taggare in maniera semantica tutto il contenuto del web con tag estratti da un'ontologia fissa e standardizzata di metadati, il manual tagging ha riscosso maggiore successo. Il semantic web ha comunque assunto nuova rilevanza in seguito all'avvento dei Linked Open Data (LOD), cioè dati pubblicati in forma strutturata così da poter essere collegati tra loro, basati su tecnologie machine readable che quindi li rendono consistent e facilmente utilizzabili. Alcune tecnologie utilizzate in questo campo sono la URI Uniform Resource Identifier, una sequenza di caratteri che identifica in maniera univoca una risorsa generica, o l'RDF Resource Description Framework, uno strumento per la codifica, lo scambio e il riutilizzo di metadati strutturati che consente l'interoperabilità tra applicazioni che condividono informazioni sul web.

I tag non sono una tassonomia ma riflettono il punto di vista di chi li associa ad una certa risorsa. I tag possono essere content based, context based, attribute, subjective, organisational. Possono essere applicati a molti differenti tipi di risorsa, e utilizzati per cercare, organizzare, condividere, scoprire nuova informazione. Dal momento che sono associati sia da chi crea la risorsa che da chi in seguito vi accede, fanno emergere nel tempo una visione della comunità. Inoltre, essendo quasi unicamente keyword testuali, possono fornire una dimensione testuale di item che non lo sono (es foto, video)e risultare quindi fondamentali nella search. Eseguire una search basata sui tag è tuttavia complesso a causa della brevità dei tag, che spesso sono presenti in numero molto ridotto per ogni item. Va inoltre risolto il Vocabulary Mismatch Problem, per il quale è possibile che non ci sia overlap tra i termini dei tag e quelli della query nonostante alcuni tra essi esprimano lo stesso concetto. É possibile superare questo problema ampliando la rappresentazione dei tag grazie a della conoscenza esterna. La search rimane complessa a causa della possibilità di noise, inesattezza, o mancanza di tag. Nel caso in cui degli item di una collezione siano privi di tag ci sono tecniche che possono associare dei tag a tali item, soprattutto nel caso di item testuali. (Esempi di queste tecniche)

Slides 11

Ricerca all'interno di comunità:

Una comunità online è un gruppo di entità che interagiscono in un ambiente online e condividono obiettivi comuni, caratteristiche o interessi. Non tutte le comunità sono costituite da esseri umani (es web comunity sono collezioni di pagine web che trattano di uno stesso argomento). Ci sono molti modi in cui un utente può partecipare ad una community, alla quale appartiene anche implicitamente. Un utente può anche essere parte di più comunità distinte tra loro. Le comunità online sono definite implicitamente dalle interazioni fra un set di entità con tratti comuni, ma possono essere identificate da algoritmi appositi che cercano caratteristiche che sono desiderate come quelle condivise (Community Detection Algorithms). Questi algoritmi prendono in input un set di entità, informazioni su ciascuna entità e dettagli sulle interazioni o correlazioni tra le entità. Questi dati sono rappresentabili tramite grafo, in cui i nodi sono le entità e gli archi diretti o indirette sono le interazioni (asimmetriche/casuali o simmetriche). I criteri per l'individuazione di comunità all'interno del grafo sono due: le entità devono essere simili tra loro secondo una qualche metrica, e le entità di una comunità devono interagire tra loro più di quanto non interagiscono con altre. L'algoritmo HITS può essere impiegato anche per trovare comunità: nell'algoritmo ciascuna entità ha un authority score (stima del valore del contenuto della pagina) e un hub score (stima del valore dei link ad altre pagine), l'algoritmo, che è iterativo e termina convergendo sempre dopo un numero finito e ridotto di iterazioni, è basato su un insieme circolare di assunzioni:

- Buoni hub puntano a buone authority
- Buone authority sono puntate da buoni hub

Dato un grafo di entità, occorre prima identificare un sottoinsieme di entità che potrebbero essere membri della comunità, dette entità candidate. Date le entità candidate, HITS può essere usato per trovare il core della comunità, cioè le entità più autorevoli (i.e., entità con authority score più elevato).

HITS prende in input un grafo G con set di nodi V e set di archi E (e il numero fissato di iterazioni K). Per trovare le comunità, il set V consiste di entità candidate, il set E di tutti gli archi fra entità candidate. Per ogni entità candidata (nodo) p nel grafo, HITS calcola un authority score A(p) e un hub score H(p). Si assume che buoni hub siano quelli che puntano a buone authority, e che buone authority siano quelle che sono puntate da buoni hub. Questo significa che esiste una interdipendenza, cioè l'authority score dipende dal hub score, che a sua volta dipende dall'authority score. Dato un set di authority score e hub score, HITS aggiorna gli score secondo le seguenti equazioni

- A(p) = $\sum_{q \to p} H(p)$ (che puntano a p)
- H(p) = $\sum_{p \to q} A(q)$ (che sono puntati da p)

Ne consegue che per essere una buona authority, una entità deve avere molti archi entranti, tutti con relativamente alti hub score, o avere relativamente pochi link entranti che hanno però alti hub score.

Algorithm 1 HITS

```
1: procedure HITS(G = (V, E), K)
2:     A_0(p) ← 1 ∀p ∈ V
3:     H_0(p) ← 1 ∀p ∈ V
4:     for i = 1 to K do
5:         A_i(p) ← 0 ∀p ∈ V
6:         H_i(p) ← 0 ∀p ∈ V
7:         Z_A ← 0
8:         Z_H ← 0
9:         for p ∈ V do
10:            for q ∈ V do
11:                if (p, q) ∈ E then
12:                    H_i(p) ← H_i(p) + A_{i−1}(q)
13:                    Z_H ← Z_H + A_{i−1}(q)
14:                end if
15:                if (q, p) ∈ E then
16:                    A_i(p) ← A_i(p) + H_{i−1}(q)
17:                    Z_A ← Z_A + H_{i−1}(q)
18:                end if
19:            end for
20:        end for
21:        for p ∈ V do
22:            A_i(p) ← A_i(p) / Z_A
23:            H_i(p) ← H_i(p) / Z_H
24:        end for
25:    end for
26:    return A_K, H_K
27: end procedure
```

1. inizializzazione a 1 di tutti gli hub e authority score delle entità candidate;

2. aggiornamento di hub e authority score secondo le equazioni precedenti;

3. gli hub score sono normalizzati in modo che la loro somma sia pari a 1;

4. gli authority score sono normalizzati in modo che la loro somma sia pari a 1;

5. l'intero processo è poi ripetuto sugli score normalizzati per un numero fisso di iterazioni indicato da *K;*

6. l'algoritmo restituisce gli authority score e hub score di tutte le entità candidate.

E' dimostrato che l'algoritmo converge sempre, e solitamente questo avviene dopo un numero ridotto di iterazioni.

Per trovare comunità online possono essere utilizzati algoritmi di clustering (data la natura del problema chiaramente appartenente all'unsupervised learning) che sia clustering gerarchico agglomerativo o clustering k-means.
La valutazione dell'efficacia di algoritmi di community detection è complessa perché non è chiaro come determinare se una certa entità dovrebbe essere considerata parte di una community o meno. Le informazioni raccolte dall'identificazione automatica di comunità online sono poi impiegabili in molti modi (estensione della search, identificazione di esperti, suggerimenti ecc) e sfruttabili per rispondere ad information need complessi. Il Community Based Question Answering si pone infatti come obiettivo quello di combinare informazioni provenienti da sorgenti multiple ed eterogenee con un significativo bagaglio di esperienze umane, sfruttando quella che viene detta wisdom of crowds. Questa tipologia di approccio ad un information need prevede aspetti positivi (possibilità di trovare risposte varie e difformi risposte a problemi complessi che necessitano di pareri di esseri umani e non di algoritmi, possibilità di consultare un archivio di risposte precedentemente date) e negativi (possibilità di mancata risposta o risposta dopo molto tempo, risposte potenzialmente errate, inopportune, spam ecc).
Nel caso in cui l'utente scelga di consultare le risposte precedenti sottomette una query che viene confrontata con le sole domande, utilizzando modelli di information retrieval come i language model, che sfruttano informazioni relative alla frequenza di occorrenza di una parola nel testo, e informazioni contestuali (es frequenza di co-occorrenza). Un language model è uno strumento statistico in grado di assegnare un valore di probabilità ad una porzione di testo in input basandosi sulle informazioni ricevute in fase di training, cioè una distribuzione di probabilità sulle possibili sequenze di termini. Esso si basa su testi conosciuti per poter prevedere sequenze di parole in testi sconosciuti. Inoltre si possono utilizzare modelli di cross language retrieval che sono basati su tecniche di machine translation che richiedono l'apprendimento delle probabilità di traduzione che legano una parola nel linguaggio source ad una nel linguaggio target. Questi modelli possono anche essere estesi per affrontare l'Intra-Language vocabulary mismatch problem. (Formule)

Slides 12

<u>Collaborative search</u>

A contrario della search classica che prevede un unico searcher, la collaborative search implica un gruppo di utenti, con un obiettivo comune, che cercano insieme in un quadro di collaborazione tra di loro. Esistono due possibili scenari: la Co-Located Collaborative Search (i partecipanti sono nello stesso luogo) e la Remote Collaborative Search (i partecipanti sono in luoghi differenti). Le criticità da affrontare sono: come i singoli utenti interagiscono tra di loro e con il sistema? Come sono condivisi i dati e quali dati persistono tra le sessioni?
CoSearch System è un esempio di co-located collaborative search system che prevede un sistema principale monitor+tastiera+mouse controllati da un driver che conduce il search task mentre gli altri partecipanti (observer) sono collegati con i loro dispositivi. Il driver sottopone la query e i risultati vengono visualizzati contemporaneamente sul monitor del driver e su tutti i dispositivi degli observer. Tutti i partecipanti possono cliccare sui risultati che vengono aggiunti in una page queue condivisa.
SearchTogether System è un esempio di remote collaborative search che non fa alcuna assunzione sulla presenza online contemporanea di tutti i partecipanti. Gli utenti del sistema possono sottomettere query che sono loggate e condivise con tutti gli altri partecipanti della ricerca. Gli utenti possono aggiungere rating e commenti alle pagine che sono state visionate durante la sessione. Un utente può raccomandare esplicitamente una pagina ad un altro.

Slides 13

Peer-to-peer e metasearch:

Abbiamo descritto social search application che implicassero network/comunità di individui; esistono invece numerosi tool di ricerca implementati usando comunità di nodi, in cui ciascun nodo può gestire informazioni (cercare, memorizzare e condividere). La forma più semplice di questo tipo di distributed search environment è un metasearch engine, in cui ogni nodo è un search engine completo i cui risultati vengono poi fusi con quelli degli altri search engine per migliorare l'efficacia del ranking complessivo. Un'altra forma di distributed search environment è la peer-to-peer search application, tipicamente costituita da un numero molto grande di nodi, ciascuno con una quantità relativamente piccola di informazioni ed una conoscenza limitata degli altri nodi.

La distributed search (DS) detta anche distributed information retrieval o federated search è una ricerca su un network di nodi ciascuno dei quali contiene dei dati su cui è possibile effettuare una ricerca. A differenza delle tradizionali applicazioni di search, i sistemi di DS devono eseguire tre ulteriori funzioni fondamentali:

- Rappresentazione delle risorse
- Selezione delle risorse
- Fusione dei risultati

Tali funzioni sono eseguite da nodi designati che dipendono dall'architettura dell'applicazione. La disposizione più semplice è quella che prevede uno speciale nodo che fornisce directory service (gestisce le informazioni riguardanti reti di computer e le risorse condivise) di selezione e fusione, e che ogni altro nodo sia responsabile del fornire la sua propria rappresentazione. La query sottoposta alla metasearch application è trasmessa dal metasearch engine a tutti i search engine usati dall'applicazione, trasformandola nel formato appropriato per ciascun search engine utilizzando le sue API. I risultati sono poi fusi tra loro e il metasearch engine fornisce un unico ranking come risultato.

Rappresentazione: I documenti in un singolo nodo sono rappresentati secondo un unico language model (distribuzione di probabilità sulle possibili sequenze di termini) che rappresenta tutti i documenti in quel nodo, con le probabilità che sono stimate usando le frequenze delle parole sommate su tutti i documenti (cioè i documenti memorizzati in un nodo sono trattati come un unico grande documento per stimare il modello del linguaggio).

Selezione: si classificano (ranking) i nodi sfruttando la loro rappresentazione (data una query si classificano le risorse secondo la probabilità che il loro language model abbia generato quella query) e poi si selezionano i top k ranked nodi o tutti i nodi con un ranking superiore ad una certa soglia.

Fusione: dopo la selezione sono eseguite ricerche locali su ciascun nodo selezionato; i risultati di queste ricerche sono poi normalizzati (per rendere possibili la comparazione) e fusi per produrre un singolo ranking, tenendo conto del fatto che lo stesso documento può apparire in più set di risultati.

Un approccio euristico alla normalizzazione degli score:

$$s'_d = s_d(\alpha + (1-\alpha)R'_d)$$

$$R'_d = (R_{score} - R_{min})/(R_{max} - R_{min})$$

Con S'd lo score normalizzato, Sd score locale (nel nodo), Rd ranking locale, R'd ranking normalizzato.

Lo score combinato complessivo dopo la fusione è:

$$s'_d = n_d^{\gamma} \sum_{i=1}^{k} s_{d,i}$$

Con nd numero di search engine che hanno restituito il documento d nella lista dei risultati, k numero di search engine che ha restituito risultati, γ costante che può assumere valori -1,0,+1. La scelta γ=+1 è nota come CombMNZ ed è risultata essere molto efficace per la combinazione di score.

Nell'architettura di un sistema P2P i nodi possono essere client, server o possono coprire entrambi i ruoli. Architetture possibili sono la Central Hub (Napster, hub centrale che fornisce directory services, nodi collegati direttamente ad esso ma non tra loro), la pure P2P (assenza di hub, la query generata da un nodo consumer è trasmessa agli altri nodi della rete tramite flooding. Nonostante le query abbiano un horizon limitato, cioè un numero massimo di hop che possono attraversare prima che scadano, questo tipo di architettura non è molto scalabile in quanto il traffico di query può crescere esponenzialmente a causa del flooding),la Hierarchical P2P(presenta una gerarchia a due livelli con nodi hub e nodi foglia). Il flooding può essere migliorato in efficienza tramite l'uso di network neighbourhoods.

Slides 14

Recommender system

Il task di un RS è quello di supportare l'utente nel processo di decision making. Tradizionalmente, per risolvere problemi del genere l'essere umano ha a disposizone una vasta gamma di strategie che lo aiutano nel processo decisionale. L'obiettico dei RS è quindi quello di fornire raccomandazioni facilmente accessibili, di alta qualità, ad un'ampia comunità di utenti (infatti questi sistemi mirano alle decisioni individuali ma, a causa della loro applicazione di massa, hanno impatto significativo in senso più ampio).

Internet e i mercati a coda lunga:

il principio di Pareto è il principio empirico formulato in ambito economico che afferma che "la maggior parte degli effetti è dovuta ad un numero ristretto di cause"; è anche noto come la legge dell'80/20, "generalmente l'80% dei risultati dipende dal 20% delle cause".

questo principio è strettamente collegato al concetto di Long Tail, per cui esiste una correlazione per cui esistono pochi prodotti (top seller) a cui è associato il maggiore volume di popolarità, mentre esistono molti prodotti con ridotti volumi di vendita. La prima categoria di prodotti è conveniente per i retail, mentre la seconda tipologia può occupare una quota di mercato equivalente o superiore a patto di avere un canale di distribuzione opportuno. questo canale di distribuzione deve poter disporre di un catalogo molto vasto a costi marginali pressochè nulli, deve poter raggiungere una popolazione di potenziali acquirenti enormemente vasta a costi marginali pressochè nulli, e questi potenziali acquirenti devono poter disporre di strumenti di selezione efficaci. il web è il canale in cui tutto questo è realizzabile, per cui si è passati nel tempo dalla mass market economy (blockbuster culture, per cui si considera solo una ristretta varietà di prodotti da vendere al maggior numero di persone) alla long tail economy.

Descrizione del dominio:i RS aiutano ad associare user e item, riducendo per lo user il problema dell'information overload e rendendo disponibili diversi paradigmi di RS a seconda del tipo e della quantità di dati disponibili, dei feedback, del dominio

Obiettivi e fattori di successo:

Punto di vista del retrieval:

- riduzione dei costi di ricerca
- produzione di raccomandazioni "corrette"
- gli utenti conoscono anticipatamente cosa vogliono

Punto di vista della racommandation:

- serendipity
- gli utenti non ne conoscono in anticipo l'esistenza
- identificazione item dalla long tail

Punto di vista della predizione:

- predicono il rating con cui ad un certo utente piace/non piace un item
- scenario di valutazione più comune nella ricerca

Punto di vista della interazione:

- "good feeling" per gli utenti
- istruiscono gli utenti sul dominio dei prodotti
- forniscono explanation

Punto di vista della conversione:

- aumento dei tassi di hit, clickthrough, lookers to bookers
- ottimizzazione di margini di vendita e profitto

Ruolo di un RS:
Per il service provider: incrementare il numero di item venduti, vendere più item differenti tra loro, incrementare la user satisfaction e la user fidelity, comprendere la domanda dell'utenza
Per l'end user: trovare item utili per l'utente e trovarli tutti, raccomandare sequenze di item correlati o gruppi di item, facilitare il browsing, aiutare ed influenzare gli altri utenti, permettendogli di esprimere la propria opinione

Pardigmi dei RS:
- Collaborative Filtering
- Content-Based Filtering
- Knowledge-Based Recommendation
- Hybridization Strategy

Tutti i diversi sistemi raccolgono dati inerenti ad item (prodotti/servizi da raccomandare caratterizzati da valore di utilizzo positivo o negativo in base all'utilità che ha per l'utente target, e complessità basata sulla qualità e disponibilità dei dati), user (vari interessi, preferenze, necessità, gli user data vanno a costituire lo user profile), transaction (interazione registrata fra utente e RS, basati sui log data a loro volta basati su feedback espliciti come un rating o impliciti come inferenze sul comportamento dell'utente).
Un RS può essere visto come una funzione che prende in input item e profilo utente e fornisce in output un relevance score usato per il ranking.
Un RS è fortemente legato all'information retrieval (IR, identificazione di informazione rilevante in una collezione di dati generalmente non strutturate) e all'information filtering (IF, selezione dell'informazione rilevante rispetto ad un utente in modo da evitare information overload). queste due aree di ricerca sono correlate anche tra loro, in quanto condividono l'obiettivo di rendere ottimizzato l'accesso a sorgenti di dati non strutturati. la convergenza tra le due aree di ricerca è la ricerca personalizzata.
Per fornire raccomandazioni personalizzate, il RS deve poter disporre di informazioni relative a ciascun utente, quindi creare e mantenere aggiornato lo user profile, acquisendo ed utilizzando dati sulla base della tecnica di raccomandazione del sistema specifico, e sfruttando poi eventuali informazioni supplementari.

Paradigma Collaborative: si basa sull'idea di considerare il comportamento, le opinioni e i gusti di una larga comunità di altri utenti, sfruttando l'assunzione che utenti che hanno condiviso interessi simili in passato avranno interessi simili anche in futuro (es bookstore online)
Paradigma Content-Based: si basa sulle descrizioni degli item (create manualmente o estratte automaticamente) e sull'importanza che un profilo assegna a queste feature (purchase history).
non esistono molti domini applicativi per questo approccio, e spesso l'assenza di purchase history rende inutilizzabii questi primi due approcci. è necessario quindi che un sistema sia in grado di sfruttare conoscenza supplementare per generare raccomandazioni
Paradigma Knowledge-Based: il RS fa uso di informazioni supplementari, spesso estratte manualmente, riguardanti sia l'utente che gli item (es Constraint-Based RS, mettono l'utente davanti ad un constraint su cui deve esprimere una preferenza che influenzerà la raccomandazione).
Paradigma Hybrid: sfrutta la combinazione di vari input e/o combinazioni di differenti meccanismi al fine di generare raccomandazioni migliori.

la distinzione tra le varie tecniche riguarda più che altro la sorgente dei dati utilizzati per la raccomandazione (background data, input data, combinazione)

Slides 15

Collaborative Filtering (indice)

- Origini storiche
- Pure CF approach
- User-based nearest-neighbor
- Pearson Correlation similarity measure
- Memory-based e model-based approach
- Item-based nearest-neighbor
- Cosine similarity measure
- Data sparsity problem
- Valutazione dei RS
- Metodi recenti (Matrix Factorization, Association Rule Mining, Slope One, RF-Rec, ...)
- Google News personalization engine
- Discussione e ricapitolazione
- Letteratura

Origini storiche: nasce dall'idea di sfruttare le opinioni delle milioni di persone online con l'obiettivo di aiutare gli utenti nel trovare contenuti più utili e interessanti. Alcuni dei primi sistemi ispirati a questa idea:
PARC Tapestry System (idea di base: i lettori entusiasti leggono tutti i documenti immediatamente, i lettori casuali attendono che i lettori entusiasti li annotino, era un sistema di mail che valutava le reazioni degli utenti nel leggere le mail per poi fornire filtri di mailing list), GroupLens (idea di base: persone che si sono trovate d'accordo nelle valutazioni in passato è più probabile che lo saranno nuovamente in futuro, era un browser di newsgroup con funzionalità di rating), MIT Ringo System (eseguiva un filtraggio collaborativo automatizzato per album e artisti musicali), Bellcore Video Recommender (filtraggio collaborativo automatizzato per film)

Slides 16

Pure CF approach: Il pure CF approach non richiede e non sfrutta alcuna conoscenza riguardante gli item. Prende in input una matrice di rating user-item e fornisce in output una predizione numerica che indica il grado con il quale un item piacerà o meno all'utente attuale e una lista dei top-N item raccomandati (recommendation list).

User-based nearest-neighbor: L'idea di base è quella di prendere in input la rating matrix e l'ID dell'utente attuale u, identificare altri utenti (detti nearest neighbors) che hanno avuto in passato preferenze simili a quelle dell'utente attuale, calcolare, per ogni item p del catalogo che non sia già stato visionato dall'utente attuale una predizione sulla base dei rating per lo stesso item p espressi dai nearest neighbors di u e poi raccomandare i best-rated. Questa tecnica si basa sulle assunzioni che le preferenze degli utenti rimangano stabili e coerenti nel tempo e che utenti che hanno avuto opinioni simili in passato le avranno anche in futuro.

Pearson Correlation similarity measure: Il Pearson Correlation coefficient è un coefficiente di correlazione tra due utenti che può assumere valori tra -1(strong negative correlation) e +1(strong positive correlation), basato su a,b utenti, r(a,p) rating dell'utente a per l'item p, P set di item valutati sia da a che da b, r(a) rating medio dell'utente a, R matrice di rating

sim(a,b) = $$\frac{\sum_{p\in P}(r_{a,p}-\bar{r}_a)(r_{b,p}-\bar{r}_b)}{\sqrt{\sum_{p\in P}(r_{a,p}-\bar{r}_a)^2}\sqrt{\sum_{p\in P}(r_{b,p}-\bar{r}_b)^2}}$$

Il coefficiente di Pearson tiene conto del fatto che gli utenti interpretino la scala di rating in maniera differente (es utenti che non assegnano mai il valore massimo), sottraendo le medie di rating degli utenti nel calcolo, in modo da rendere gli utenti comparabili tra loro.

Una possibile equazione per la predizione relativa ad un certo item è: $$pred(a,p) = \overline{r_a} + \frac{\sum_{b\in N} sim(a,b) * (r_{b,p} - \overline{r_b})}{\sum_{b\in N} sim(a,b)}$$

Le problematiche in un'applicazione real-world sono:

- Complessità computazionale dovuta alla dimensione dei db
- Rating matrix generalmente sparsa in quanto ogni utente valuta di norma un sottoinsieme molto piccolo degli item disponibili
- Difficoltà di gestione di nuovi utenti e di nuovi item a causa della mancanza di dati
- Nonostante il coefficiente di Pearson ottiene le prestazioni migliori nel caso di User-Based RS, il suo uso esclusivo può non risultare la scelta migliore. Occorre infatti considerare che nella maggior parte dei domini ci saranno alcuni item che piacciono a tutti, quindi un accordo tra due utenti su un item univocamente apprezzato è meno rilevante a livello informativo che un accordo su un item controverso
- Il numero di item co-rated (valutati sia dall'utente a che dall'utente b)può essere basso

Miglioramenti:

- Uso di una funzione di trasformazione detta Inverse User Frequency, riduce l'importanza relativa ad item che sono univocamente graditi da tutti gli user
- Uso di un Variance Weighting Factor, incrementa l'influenza degli item che hanno alto valore di varianza nel rating e sui quali quindi vengono espressi rating controversi
- Uso di un fattore che riduce linearmente il peso quando il numero di co-rated items è basso (Significance Weighting)
- Fine Tuning dei pesi di predizione, ovvero dare più peso a neighbors molto simili (per i quali il valore di similarità è vicino a -1 o +1)
- Selezione del neighborhood: definizione di un valore minimo specifico di soglia per la similarità tra utenti, limite della dimensione del neighborhood a k utenti simili

Approcci Memory-Based e Model-Based: Le tecniche di raccomandazione collaborativa sono spesso classificate in Memory-Based (a cui appartengono i modelli User-Based poichè la rating matrix è conservata interamente in memoria ed utilizzata direttamente per generare le raccomandazioni) e Model-Based (i dati grezzi sono prima processati offline e impiegati per apprendere un modello, utilizzato poi a runtime per fare predizioni). Nonostante i modelli Memory Based siano teoricamente più precisi poichè basati su dati completi e disponibili, sono poco scalabili.

Slides 17

Item-based nearest neighbor CF: Sebbene gli approcci User-Based CF siano stati applicati con successo su vari domini, rimangono alcune criticità che li rendono impossibili da usare su larga scala (siti di e-commerce su larga scala). Questi siti/ domini implementano quindi una tecnica differente detta Item-Based raccomendation, più adatta per il preprocessing online, che rende quindi possibile il calcolo di raccomandazioni real time anche per matrici di rating di grandi dimensioni. L'idea di base per questo tipo di algoritmi è calcolare la predizione usando la similarità tra item e non la similarità tra utenti.

Cosine Similarity Measure: Per trovare item simili occorre sfruttare una misura di similarità, una metrica che nel caso standard è la Cosine Similarity, che misurà la similarità tra due vettori n-dimensionali sulla base del coseno del loro angolo. La similarità tra due item a e b visti come vettori è così calcolata (con prodotto scalare e lunghezza euclidea/norma, cioè la radice quadrata del prodotto scalare del vettore con sè stesso):

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$

Questa misura prende valori tra 0(weak similarity) e 1(strong similarity), ma non tiene conto delle differenze nel comportamento di rating medio tra utenti. Questo problema è risolto con la Adjusted Cosine Measure che sottrae la media dell'utente dai rating. Questa misura assume valori che vanno da -1 a +1. L'ACM è così calcolata:

$$sim(a, b) = \frac{\sum_{u \in U}(r_{u,a} - \overline{r_u})(r_{u,b} - \overline{r_u})}{\sqrt{\sum_{u \in U}(r_{u,a} - \overline{r_u})^2}\sqrt{\sum_{u \in U}(r_{u,b} - \overline{r_u})^2}}$$

Si può predire, per l'utente u, un rating per l'item p come segue:

$$pred(u, p) = \frac{\sum_{i \in ratedItems(u)} sim(i, p) * r_{u,i}}{\sum_{i \in ratedItems(u)} sim(i, p)}$$

Preprocessing per Item-Based filtering: Per rendere applicabili gli algoritmo Item-Based per siti in larga scala senza sacrificare l'accuratezza delle informazioni è normalmente adottato un approccio basato sul preprocessing offline dei dati. L'idea è quella di costruire a priori la item similarity matrix che descrive la similarità a coppie (pairwise similarity) di tutti gli item in catalogo. A runtime una predizione per un prodotto p e un utente u è fatta considerando solo gli item che sono più simili a p sulla base dei rating di tutti gli altri utenti e calcolando la somma pesata dei rating di u per gli item del neighborhood. Il numero di neighbors da considerare è limitato al numero di item che l'utente target ha valutato in precedenza, e dal momento che tale neighborhood è quindi tipicamente piuttosto piccolo, il calcolo della predizione può essere rapidamente svolto in realtime.

Riguardo ai requisiti di memoria, una item similarity matrix completa per N item può teoricamente avere fino a N^2 elementi ma ne ha nella pratica molti meno. Inoltre ci sono altre tecniche che possono essere applicate per ridurre la complessità.

In linea teorica sarebbe possibile applicare questo tipo di preprocessing anche negli approcci User-Based; tuttavia nella pratica uno user profile contiene tipicamente meno rating rispetto ad un item profile, e quindi si avrebbero meno rating per calcolare la similarità tra due utenti di quanti non se ne abbiano per calcolare la similarità tra due item. Inoltre la similarità tra utenti risulta essere più dinamica (quindi il calcolo a priori dello user neighborhood può risultare poi poco accurato) rispetto alla similarità tra item che rimane invece abbastanza statica.

Esiste poi la possibilità di sfruttare soltanto una certa frazione della rating matrix (es tramite subsampling casuale o basato su un qualche criterio), così da ridurre la complessità computazionale.

Rating espliciti: Tra le alternative possibili per raccogliere opinioni dagli utenti, richiedere item rating espliciti è probabilmente la più precisa. Nella maggior parte dei casi sono utilizzate Five-Point o Sevent-Point Likert Response Scale che vanno da "strongly dislike" a "strongly like" e che vengono poi convertite in valori numerici per applicare le misure di similarità.
Argomenti di ricerca attuale sono come varia il rating behaviour degli utenti sulla base della scala utilizzata, come varia la recommendation quality all'aumentare della granularità della scala e al variare della multidimensionalità del rating.
Le problematiche principali relative ai rating espliciti sono che tali rating richiedono uno sforzo aggiuntivo all'utente che può non essere disposto a fornire a causa della mancanza di ricezione di benefici immediati. Questa argomentazione è però ancora argomento di dibattito e studio, motivo per cui si chiede una maggiore ricerca per lo sviluppo di modelli di rating da impiegare per persuadere gli utenti a partecipare sempre più attivamente.
Rating impliciti: Per rating impliciti si intendono una molteplicità di comportamenti online quali l'acquisto di un item, il browsing behaviour di un utente ecc. Questo tipo di rating sono raccolti continuamente e non richiedono sforzo attivo da parte dell'utente. Di contro però possono essere più difficili da interpretare, anche se si sostiene che potrebbero determinare modelli addirittura più accurati di quelli basati su rating espliciti.

Data sparsity problem: Il problema delle applicazioni real-world sta nell'avere matrici di rating sparse poichè gli utenti di solito forniscono rating solo per una piccolissima frazione di item del catalogo, il che rende più difficile calcolare buone predizioni. Una semplice opzione per affrontare il problema è quella di considerare anche informazioni aggiuntive riguardo agli utenti, basando così i neighborhood non solo sull'analisi dei rating espliciti e impliciti ma anche su informazioni esterne alla matrice di rating. Tuttavia, sistemi basati su informazioni di questo tipo non sono più "collaborativi" in senso stretto.
Algoritmi per sparse dataset:
Recursive CF: assumiamo di avere un neighbor n molto vicino all'utente corrente u che tuttavia non ha ancora valutato l'item i. Questo algoritmo è basato sull'idea di applicare il metodo CF ricorsivamente e predire un rating del neighbor n per l'item i, e usare questo rating predetto invece del rating di un neighbor diretto più distante.
Spreading Activation (metodo graph-based): l'idea di base è quella di sfruttare la supposta proprietà di "transitivity" dei gusti degli utenti ed estendere così la matrice con informazioni addizionali.

- Si considera la rating matrix binaria (1 se user ha dato rating a item, 0 altrimenti) e si genera un user-item relationship graph bipartito (user e item sono collegati da arco se nella matrice la coppia è segnata con 1).
- Se negli approcci standard CF verrebbero considerati percorsi di lunghezza 3 che collegano lo user corrente e nuovi item che potrebbero essergli consigliati, in un approccio basato su uno sparse dataset si considerano percorsi di lunghezza maggiore, ad esempio 5, o in generale >3, per generare raccomandazioni.

Default Voting: l'idea è quella di assegnare valori di default ad item che solo uno dei due utenti che si stanno comparando ha valutato (e possibilmente anche ad alcuni item aggiuntivi). Questi voti artificiali agiscono come un meccanismo di smorzamento (damping) che riduce gli effetti di similarità dovuti a rating individuali e fortuiti.
Similarity Fusion: l'idea di base è combinare fra loro similarità fra utenti e similarità tra item al fine di migliorare la prediction accuracy, sfruttando nella funzione di predizione anche un terzo tipo di informazione non considerato in approcci precedenti: "similar item ratings made by similar users".
Cold-Start Problem: può essere visto come un caso particolare del data sparsity problem, in cui si hanno le seguenti problematiche:

- Come fare raccomandazioni a nuovi utenti che non hanno ancora espresso alcun rating
- Come trattare nuovi item che non sono ancora stati valutati o acquistati

Questi problemi vengono affrontati con l'ausilio di approcci ibridi, cioè con l'uso di informazioni supplementari ed esterne (es algoritmo di Eigentaste).

Slides 18

Valutazione dei RS: La maggior parte delle valutazioni sperimentali analizza l'accuratezza del RS nel supportare l'utente nel task di "trovare buoni item". La valutazione può avvenire in più modi:

- Misurare quanto il sistema è in grado di predire il valore esatto di rating dell'utente target (Value Comparison)
- Misurare quanto il sistema è in grado di predire se l'item è rilavante o meno per l'utente target (Relevant vs Not Relevant)
- Misurare quanto il ranking degli item predetto dal sistema è vicino al ranking reale dell'utente target (Ordering Comparison)

Per valutare un RS occorre eseguire i seguenti passi:

1. Raccogliere i dati, cioè i valori di rating user-item
2. Suddividere i dati disponibili in train set e test set
3. Costruire un modello sui dati di training
4. Confrontare i valori predetti dal RS con
    - Rating su ciascun item del test set
    - Raccomandazioni con le reali buone raccomandazioni (quali sono?)
    - Ranking con il ranking reale (qual è?)

Occorre quindi una metrica per confrontare i valori predetti di rating con i valori reali di training.

Value Comparison: L'obiettivo è quello di misurare quanto i rating predetti dal sistema sono vicini ai rating reali espressi dagli utenti, per tutti i rating nel test set R(test). Questa misura che si vuole valutare è quella della predictive accuracy, con la metrica del Mean Absolute Error (MAE), dove r*(ui) è il rating predetto dal sistema per l'utente u appartenente a U e l'item i appartenente a I, e r(ui) è quello reale dell'utente:

$$MAE(r^*) = \frac{1}{|R_{test}|} \sum_{r_{ui} \in R_{test}} \left| r^*_{ui} - r_{ui} \right|$$

Variazione 1: ci saranno sicuramente utenti con un set di rating molto più ampio rispetto ad altri user. Per risolvere questo problema si può modificare la metrica di rating, ad esempio enfatizzando gli errori più elevati, nel caso della Root Mean Square Error (RSME):

$$RMSE(r^*) = \sqrt{\frac{1}{|R_{test}|} \sum_{r_{ui} \in R_{test}} \left(r^*_{ui} - r_{ui} \right)^2}$$

Variazione 2: per rendere possibile in confronto tra risultati ottenuti su dataset differenti, che hanno quindi potenzialmente differenti scale di rating, si può usare il Normalized MAE:

$$NMAE(r^*) = \frac{1}{|R_{test}|(r_{\max} - r_{\min})} \sum_{r_{ui} \in R_{test}} \left| r^*_{ui} - r_{ui} \right|$$

Relevant vs Not Relevant: Per misurare queste qualità si usano Precision e Recall. Per calcolarle la scala di rating deve essere binaria o deve essere trasformata in una che lo sia. La Precision per un singolo utente è il rapporto tra gli item rilevanti restituiti dal RS e il numero totale di item restituiti dal RS (N(selected,relevant)/N(selected)), mentre la Recall per un singolo utente è il rapporto tra gli item rilevanti restituiti e il numero totale di item rilevanti presenti nel catalogo (N(selected,relevant)/N(relevant)). Precision e Recall sono le metriche più utilizzate per valutare RS.

Vogliamo ora calcolare i valori di Precision e Recall per l'intero insieme di utenti U. Suddividiamo l'insieme di rating in training set e test set, siano T(u) gli item che sono stati valutati "high"/relevant da u e sono nel test set. L(u) è la recommendation list per u (usando train), e l'intersezione di L(u) e T(u) è detto Hit Set.

$$P(L) = \frac{1}{|U|}\sum_{u \in U} |L(u) \cap T(u)| / |L(u)|$$

$$R(L) = \frac{1}{|U|}\sum_{u \in U} |L(u) \cap T(u)| / |T(u)|$$

La combinazione dei valori di Precision P e Recall R è data dalla F1:

$$F_1 = \frac{2PR}{P+R}$$

Problematiche dell'uso di Precision e Recall:
-Occorre conoscere quali item sono rilevanti e quali no
-Difficile sapere cosa è rilevante e cosa non per un utente in un RS che gestisce migliaia/milioni di item
-La Recall è più difficile da stimare

Ordering Comparison: La metrica utilizzata in questo caso è il Normalized Discounted Cumulative Gain (NDCG). Dato un set di query Q(user), sia R(j,m) il valore di rilevanza (che vale 1 se rating>3, 0 altrimenti) che gli user hanno attribuito al documento (item nel test set) in posizione di indice m per la query (user)j. Il ranking è calcolato ordinando gli item per valore decrescente del rating predetto:

$$NDCG(Q,k) = \frac{1}{|Q|}\sum_{j=1}^{|Q|} Z_{kj} \sum_{m=1}^{k} \frac{2^{R(j,m)}-1}{\log_2(1+m)}$$

dove Z(k,j) è un fattore di normalizzazione calcolato in modo tale che l'NDCG di un ranking perfetto sui k top search result per la query j sia 1. Per gli utenti per i quali k'<k item sono nel test set, l'ultima sommatoria è eseguita fino a k'.

Metriche beyond accuracy: novelty, serendipity, diversity, coverage, learning rate, confidence, metriche di user satisfaction, metriche di site performance.

Slides 19

Metodi recenti:

Matrix Factorization/Latent Factor Models: I metodi di matrix factorization possono essere impiegati nei RS per individuare un set di latent(hidden) factors dai modelli di rating e caratterizzare sia gli utenti sia gli item tramite vettori di tali fattori. Viene poi fornita una raccomandazione di un certo item i quando l'utente target e l'item risultano simili rispetto a questi fattori.

In algebra lineare la Singular Value Decomposition (SVD) è una fattorizzazione di una matrice tramite autovalori e autovettori. Data una matrice M reale o complessa di dimensioni nxn, essa può essere decomposta nel prodotto matriciale di tre matrici M=UxAxVT (VT è la matrice V trasposta), in cui ogni colonna della matrice U è un autovettore sinistro, ogni colonna della matrice V è un autovettore destro, e gli elementi della diagonale di A sono gli autovalori di M. In termini formali, si può affermare che la SVD rende possibile approssimare la matrice completa osservando soltanto le feature più importanti, cioè quelle con autovalori più elevati, riducendo la dimensionalità.

La matrix factorization genera approssimazioni low-rank/a basso rango della matrice user-item, individua fattori latenti e proietta user e item nello stesso spazio n-dimensionale. La qualità dei rating può diminuire perchè non si tiene conto dei rating originali, ma può aumentare perchè viene filtrato del noise dai dati e perchè sono individuate delle correlazioni non banali nei dati.

Association Rule Mining: è una tecnica usata per individuare rulelike relationship pattern in transazioni di vendita in larga scala e calcolare una misura della qualità per tali regole (es un'applicazione tipica è l'individuazione di coppie di generi di prodotti in un supermercato che sono spesso acquistati insieme).

Nella notazione formale, una transazione T è un sottoinsieme dell'insieme di prodotti P={p1,...,pm} e descrive un insieme di prodotti che sono stati acquistati insieme. Le regole di associazione sono spesso scritte nella forma X->Y (X implica Y), con X e Y entrambi sottoinsiemi disgiunti di P. Una regola di associazione X->Y esprime che ogni volta che gli elementi di X (il body della regola) sono inclusi in una transazione T, è probabile che gli elementi di Y (l'head della regola) siano elementi della stessa transazione.

Le misure per la qualità dei pattern individuati sono il Support e la Confidence. Il Support di una regola X->Y è definito come il rapporto tra il numero di transazioni che contengono tutti gli item di XUY e il numero di transazioni complessive (la probabilità di co-occorrenza di X e Y in una transazione):

$$\text{support} = \frac{\text{number of transactions containing } X \cup Y}{\text{number of transactions}}$$

La Confidence di una regola X->Y è definita come il rapporto tra il numero di transazioni che contengono tutti gli item di XUY e il numero di transazioni che contengono X (la probabilità condizionata di Y dato X):

$$\text{confidence} = \frac{\text{number of transactions containing } X \cup Y}{\text{number of transactions containing } X}$$

Algoritmi standard di rule-mining sono utilizzati per analizzare una matrice di rating binaria o resa tale, individuare un elenco di regole di associazione e calcolare i loro corrispondenti valori di support e di confidence, per poi selezionare le regole effettivamente rilevanti come quelle che hanno valori di Support e Confidence superiori ad una certa soglia scelta. Il calcolo del set di regole di associazioni interessanti può essere svolto offline, mentre a runtime si determinano le regole interessanti per l'utente corrente, se ne calcolal'insieme unione di item Y coinvolti in queste regole, gli item vengono ordinati secondo il valore di Confidence della regola che li ha predetti e si restituiscono i primi n elementi di questa lista ordinata come una raccomandazione.

Probabilistic Recommendation Approach: è l'approccio che sfrutta i formalismi della teoria della probabilità. Si può vedere il problema di predizione come un problema di classificazione. Una tecnica standard è quella basata sui Classificatori di Bayes.
Si considera una tabella con n user e m item, per la quale vogliamo determinare la predizione per l'm-esimo item di un certo utente u. Con questo metodo calcoliamo le probabilità condizionate per ogni possibile valore di rating dati gli altri rating dell'utente u, e poi selezioniamo come predizione quello col più alto valore di probabilità. Per il calcolo di tali probabilità condizionate (un calcolo per ogni rating possibile) si può usare il Teorema di Bayes che consente di calcolare tale probabilità a posteriori P(YIX) tramite la class conditional probability P(XIY), la probabilità di X e la probabilità di Y:

$$P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)}$$

Sotto la condizione che gli attributi diano conditionally independent possiamo calcolare la probabilità a posteriori per ciascun valore di Y con un classificatore Bayesiano naive:

$$P(Y|X) = \frac{\prod_{i=1}^{d} P(X_i|Y) \times P(Y)}{P(X)}$$

con d numero di attributi in ogni X.
Approccio Cluster-Based:è un altro approccio probabilistico in cui si assume che gli utenti ricadano in un piccolo numero di cluster e si fanno predizioni sulla base della probabilità che un utente u ricada in un cluster c e che all'utente u piaccia l'item i dato un certo cluster e i suoi precedenti rating.
Slope One Predictor: l'idea di base è quella della popularity differential fra item e user. In generale, il problema consiste nel trovare funzioni della forma f(x)=x+b che predicono, per una coppia di item, il rating per un item dal rating dell'altro. Può risultare computazionalmente pesante.
Raccomandazione come problema di ottimizzazione: l'operazione di raccomandazione può essere vista anche come un problema di ottimizzazioine in cui l'operazione di raccomandazione consiste nell'apprendimento da osservazioni noisy (x,y) in cui la funzione di stima deve essere determinata in modo tale che l'errore quadratico tra valore predetto e valore reale sia minimo.
Google News: è un portale online di informazioni che aggrega news article provenienti da diverse migliaia di fonti, visualizzandoli in modo personalizzato per gli utenti registrati. Sfrutta il CF Recommendation Approach basato sulla click history dell'utente target e sulla click history della comunità allargata. Le criticità principali sono: la presenza di un vasto numero di articoli e utenti, la necessità di generare liste di raccomandazioni in real time, lo stream costante di nuovi news article.. per cui è impiegata una combinazione di tecniche memory based e model based ed è fortemente potenziata la parallelizzazione. Per la parte model-based sono applicate due tecniche di clustering: la PLSI (Probabilistic Latent Semantic Indexing) e MinHash come metodo di hashing per inserire due utenti nello stesso cluster sulla base della possibile sovrapposizione di item che entrambi gli utenti hanno visionato.