

Filtraggio collaborativo personalizzato nelle folksonomie

Università degli Studi di Firenze



Facoltà di Ingegneria
Tesi di Laurea in Ingegneria Informatica

Relatori: Paolo Frasconi

Candidato: Matteo Bertini

Alessandro Fantechi

A/A 2005-2006

Prologo

La quantità di informazioni che ogni giorno appare nel web è di gran lunga troppo grande per essere consultata da un utente senza l'aiuto di sistemi di filtraggio personalizzati. Per questa ed altre ragioni la ricerca nell'ambito del filtraggio collaborativo è molto attiva e trova molto spesso applicazione su dati diversi con obiettivi diversi.

Questa ricerca è mirata allo studio dell'informazione presente nelle folksonomie, sempre più diffuse grazie a quello che viene definito software sociale. In questo lavoro viene studiata una particolare folksonomia originata da un sistema di bookmarking sociale, e vengono proposti e confrontati alcuni algoritmi di suggerimento collaborativo personalizzato.

Il testo presenta la definizione di un generale concetto di similarità nelle folksonomie ed due nuove misure di valutazione della qualità di previsione specifiche per folksonomie.

1 Abstract

The great amount of new information appearing every day in Internet is too much for users to navigate without the help of some personalized information filtering system. Research in the area of recommender systems is quite active, focusing on disparate datasets. My research is focused in extracting knowledge from folksonomies growing in social software. In this thesis I aim to explain some relations inside that folksonomies, and present a way to improve a personalized recommendation system using discovered informations.

This work presents a clean approach to define similarity measures inside folksonomies using the Tag-Space vision and two quality measures for the prediction specialized on folksonomies.

Indice

Prologo	3
1 Abstract	3
Indice	5
1 Introduzione	7
1.1 Struttura della tesi	8
1.2 Reperire informazioni in internet	9
1.3 Ricerche collegate	10
1.3.1 Sistemi di raccomandazione	10
1.3.2 Data Mining	11
2 Folksonomie	13
2.1 Esempi di folksonomie in internet	15
2.2 Sistemi di bookmarking collettivo	15
2.3 Interpretazione matriciale	16
2.4 Eredità dall'Information Retrieval	18
2.4.1 Nuovi dati, vecchi metodi	19
3 Analisi statistica della folksonomia di del.icio.us	21
3.1 Interfaccia di del.icio.us	21
3.2 Raccolta dei dati per l'esperimento	23
3.3 Sintesi statistica dei dati	25
3.3.1 Proiezioni primarie	27
3.3.2 Proiezioni secondarie	30
3.4 Evoluzione temporale dei dati	32
4 Misure di similarità nelle folksonomie	35
4.1 Apprendimento automatico	35
4.2 Misure dirette	36
4.2.1 Alcuni esperimenti qualitativi	38
4.3 Misure trasversali	39
5 Algoritmi di suggerimento	41
5.1 Ordinamenti impersonali	41

5.2	Ordinamenti basati sugli utenti	42
5.3	Ordinamento basato sulla misura trasversale	45
5.4	Algoritmi di espansione semantica	45
6	Valutare la qualità della predizione	47
6.1	Precisione e Recupero	47
6.2	Dalla Precisione alla Similarità	48
6.3	Dal Recupero alla RevCall	48
7	Esperimenti e risultati	51
7.1	Selezione dei dati con $n = 6$	51
7.2	Selezione dei dati con $n = 4$	57
7.3	Confronto tra i due migliori ordinamenti	59
7.4	Algoritmo di espansione semantica	61
8	Conclusioni e possibili sviluppi	63
	Bibliografia	65

Capitolo 1

Introduzione

Le dimensioni di internet e la velocità di afflusso di nuove informazioni hanno stimolato la nascita di sistemi che cercano di selezionare un sottoinsieme limitato di informazioni ritenute importanti per il particolare utente.

Negli ultimi 10 anni abbiamo assistito ad una rapida evoluzione di internet ed in particolare del web. Alla nascita il web era uno strumento asimmetrico, il numero degli utenti in grado di pubblicare informazioni era estremamente ridotto rispetto al numero di utenti in grado solo di consultare la rete.

Negli anni '90 abbiamo assistito ad una rivoluzione, che possiamo riassumere in due eventi strettamente correlati:

1. nascono le applicazioni internet (Hotmail^{1.1} è forse il primo esempio di trasposizione di un programma di posta elettronica in una pagina web interattiva),
2. esplose il numero degli utenti in grado di pubblicare informazioni in internet (nascono i primi diari online, i primi blog^{1.2}).

Il web si evolve, e da risorsa da consultare, diventa un importante mezzo con cui parlare al mondo. Infatti accanto ai programmi di posta nel web, nascono molte applicazioni che sono divenute parte essenziale della rete: nascono i forum^{1.3} di discussione ed i già citati blog.

In questo periodo assistiamo infatti alla nascita di una nuova forma di competizione. I giornali e le riviste si trovano a dover conquistare il pubblico del web non solo in competizione tra loro come in passato, ma anche in competizione con uno sterminato esercito di estemporanei giornalisti, critici e commentatori, i “blogger”. La forza di questi disinteressati “avversari” sta semplicemente nel numero enorme di nuovi articoli che insieme possono pubblicare ogni giorno. Come recita la prima pagina del famoso aggregatore di blog Technorati^{1.4}:

50 milioni di blog... qualcuno *deve* essere interessante.

1.1. Hotmail è un servizio di posta elettronica via web (detta in genere webmail) fondato da Jack Smith e Sabeer Bhatia nel 1995: it.wikipedia.org/wiki/Hotmail

1.2. I primi diari in internet, antenati degli odierni blog, risalgono al 1994: en.wikipedia.org/wiki/Blog

1.3. I primi forum di discussione nascono nel 1995: en.wikipedia.org/wiki/Internet_forum

1.4. Technorati: www.technorati.com/

Questa nuova massa di informazioni ha però un'enorme carenza, manca di struttura. I blog, come diari, non hanno nessuna pretesa di uniformità tematica. È normale che due articoli consecutivi dello stesso blog affrontino argomenti completamente diversi. Per questa ragione molti software per gestire blog offrono oggi la possibilità di associare ad un articolo delle parole chiave, delle etichette con cui l'autore può classificare gli articoli per argomento.

Questa soluzione è diventata uno standard *de facto*, mentre ufficialmente la ricerca era ed è tuttora mirata verso la creazione di una struttura semantica generica che permetta di esprimere tutte le relazioni possibili nel web, il cosiddetto web semantico.

Anche se la potenza espressiva di una ontologia ben progettata è sicuramente superiore alla potenza espressiva dell'associazione di una parola chiave ad un contenuto, è anche vero che il numero di utenti in grado di associare una parola chiave o una semplice "etichetta" ad un sito web, è molto superiore al numero di utenti in grado di sfruttare gli strumenti ed i linguaggi messi a disposizione dalla ricerca nel campo del web semantico.

Possiamo quindi assumere che, fino a quando non esisteranno interfacce ad elevata usabilità per la creazione di contenuti con informazione semantica, il web si presenterà con un'esigua porzione "semanticizzata" ed una ben più grande porzione semplicemente "etichettata".

Quando più utenti classificano delle risorse con l'uso di etichette, si viene a creare una struttura che è stata definita col neologismo *folksonomia* (vedi capitolo 2).

1.1 Struttura della tesi

La tesi si pone come obiettivo l'ordinamento personalizzato delle risorse usando le informazioni sulla personalità dell'utente presenti in una folksonomia.

Per far questo il lavoro è stato diviso in più fasi:

1. il primo capitolo presenta un panorama molto sintetico di tutte le scienze o tecnologie che sono entrate in gioco nelle varie fasi del lavoro,
2. il secondo capitolo presenta le folksonomie e ne formalizza una definizione matriciale,
3. il terzo capitolo presenta il sito origine dei dati e continua con una sintesi statistica dei dati stessi,
4. il quarto capitolo presenta le misure di similarità definite in una folksonomia,
5. il quinto capitolo presenta gli algoritmi di suggerimento personalizzato,
6. il sesto capitolo presenta le misure di valutazione della qualità di predizione,
7. il settimo capitolo presenta e commenta i risultati degli esperimenti,
8. l'ottavo ed ultimo capitolo conclude il lavoro con riflessioni e spunti per il futuro.

1.2 Reperire informazioni in internet

Internet è essenzialmente uno strumento per la condivisione delle informazioni, ogni utente accede ad internet per trovare informazioni e per scambiarle con altri.

In estrema sintesi è possibile classificare il reperimento di informazioni in due classi:

1. il reperimento attivo di informazioni, cioè la ricerca di informazioni specifiche per affrontare una situazione contingente (ad esempio, cercare il significato di una parola sul dizionario),
2. il reperimento passivo di informazioni, cioè la normale prassi di tenersi aggiornati su argomenti di interesse generale (ad esempio, leggere il quotidiano o guardare la televisione).

Esistono anche fonti di informazione passiva specializzata, come ad esempio canali tematici o riviste specializzate. Questo tipo di informazione è comunque indirizzata ad un pubblico più ampio del singolo utente. Per questo nella grande maggioranza dei casi, l'utente cerca attivamente le informazioni che lo interessano.

La ricerca di informazioni si basa, in sintesi, su tre tipologie di fonti:

1. l'esperienza personale,
2. il consiglio di conoscenti (scelti in base all'argomento),
3. le fonti impersonali (giornali, enciclopedie).

Fonti che possono essere interpretate nel ristretto ambito della navigazione web come:

1. i propri bookmark,
2. il consiglio di conoscenti,
3. i motori di ricerca ed i portali internet.

Possiamo notare che l'evoluzione più marcata si è verificata nell'ambito dei portali e dei motori di ricerca. I motori di ricerca si sono evoluti molto negli ultimi anni e la qualità della ricerca è cresciuta sensibilmente [17]'98, [16]'2000; parallelamente sono nati portali (tematici o meno) che forniscono un'abbondante quantità di informazioni per l'utente che intende tenersi aggiornato.

Gli sviluppi in internet delle prime due fonti sono invece molto più limitati, e non godono ancora pienamente dei vantaggi della condivisione di informazioni e della comunicazione a distanza che internet offre. Ad esempio ci potremmo aspettare una migliore "memoria" dei siti visitati in passato (e poi regolarmente dimenticati e cercati nuovamente), oppure un qualche metodo automatico per trovare persone con interessi simili ai nostri, anche se non le conosciamo personalmente.

Guardandoci attorno però possiamo vedere che molti degli ingredienti che potrebbero permettere una visione personalizzata del web sono già in uso. Il dubbio che deve essere chiarito è: sono già questi i germogli della futura evoluzione del web, oppure i veri germogli devono ancora spuntare?

1.3 Ricerche collegate

I sistemi di filtraggio collaborativo sono oggetto di ricerca da molti anni. Forse tra le prime ricerche effettuate la più famosa sono Tapestry (1992) [10] e GroupLens (1994) [20]; poi sicuramente è da citare il lavoro “Recommender systems” (Resnick e Varian, 1997) [21] a cui dobbiamo la definizione di “Sistemi di Raccomandazione”, cioè di sistemi disegnati per prevedere le preferenze di un utente usando le preferenze espresse dagli altri utenti del sistema.

Al crescere del numero delle pubblicazioni si è resa necessaria la scelta di una terminologia più precisa [2]’97 e di un’identificazione più chiara delle problematiche di questa tipologia di sistemi [18]’01. Presto si sono rese necessarie anche delle ricerche comparative (Herlocker 2004) [13], mirate a confrontare i risultati sperimentali. Infatti spesso ogni lavoro ha introdotto una nuova misura di prestazione e questo rende molto difficile un confronto tra algoritmi diversi.

Possiamo riassumere che la ricerca nell’ambito del "suggerimento collaborativo" (termine usato in questo testo per definire genericamente i sistemi di suggerimento collaborativo basati sugli utenti, sui contenuti o misti) ha esplorato una vasta gamma di approcci, e trovato applicazioni su una vasta gamma di archivi: dai primi suggerimenti di articoli in Usenet di GroupLens [20]’94 al suggerimento di film sull’archivio Bellcore [14]’95 fino al recente suggerimento di comunità interessanti in Orkut [23]’05.

Vediamo brevemente prima una panoramica sui sistemi di raccomandazione e poi una breve presentazione del processo di scoperta del sapere (*Knowledge Discovery o Data Mining*) che sempre più spesso caratterizza le prime fasi dell’approccio ad una base di dati ignota.

1.3.1 Sistemi di raccomandazione

I sistemi di raccomandazione (Resnick & Varian, 1997) [21] sono sistemi progettati per predire le preferenze di un utente, usando in modo collettivo le informazioni acquisite su tutti gli utenti del sistema.

Esistono sistemi che fanno uso delle sole raccomandazioni degli utenti, sistemi che usano le sole caratteristiche estratte dall’oggetto suggerito e sistemi che operano in modo misto.

Indipendentemente dal tipo di sistema, il processo di suggerimento presenta dei problemi comuni [18].

Il problema dei nuovi utenti. Il sistema deve essere in grado di fare predizioni "interessanti" anche dopo un breve periodo d’uso per evitare che l’utente smetta di usare il sistema.

Il problema del rinnovo continuo. Nuovi oggetti vengono aggiunti con continuità al sistema. Un sistema basato solo sulle preferenze degli utenti rischia di non essere in grado di fare predizioni accurate sui nuovi ingressi.

Il problema della sparsità delle preferenze. In ogni sistema di raccomandazione il numero di oggetti con molte preferenze espresse è molto inferiore al numero complessivo degli oggetti presenti e raccomandabili.

Il problema delle prestazioni. Sistemi di questo tipo sono spesso destinati ad operare in tempo reale con un numero molto alto di richieste ed archivi in continua crescita.

I sistemi che usano solo le valutazioni degli utenti si dicono "sistemi di raccomandazione collaborativa". In questo tipo di sistemi non vengono suggeriti oggetti simili a quelli scelti dall'utente nel passato, ma vengono suggeriti gli oggetti scelti nel passato da utenti simili. Esempi di questo tipo sono GroupLens [20] ed il sistema di raccomandazione di film Bellcore [14]. Questo tipo di sistemi "puri" è il più sensibile al problema del rinnovo continuo.

Al contrario i sistemi basati esclusivamente sulle caratteristiche degli oggetti catalogati, "sistemi di raccomandazione basati sul contenuto", sono meno sensibili al problema del rinnovo continuo, ma rischiano di essere affetti dal problema dei nuovi utenti. Inoltre sistemi di questo tipo hanno problemi nel catalogare informazioni di tipo non testuale e rischiano nel tempo di specializzarsi troppo sul profilo dell'utente [2].

Il sistema presentato in questa tesi è un sistema basato sulle sole preferenze degli utenti, ma come nel lavoro di Han e Karipys del 2005 [12] gli oggetti non sono entità atomiche ma hanno degli attributi, però a differenza di [12]'05 gli attributi sono specifici per ogni utente.

1.3.2 Data Mining

Si definisce "Data Mining" (DM) o anche "Knowledge Discovery in Databases" (KDD) il processo molto euristico che si avvale dell'uso simbiotico di varie scienze e tecnologie al fine di estrarre conoscenza da grandi archivi di dati.

Secondo Frawley *et al.* [9]'92 è possibile definire lo "Knowledge Discovery" come *il processo di estrazione di informazioni implicite, precedentemente ignote e potenzialmente utili, dai dati.*

Nel processo di scoperta sono necessarie come minimo competenze in quattro ambiti.

Conoscenza delle tecnologie di archiviazione. Quando si presenta un problema di KDD, i dati da analizzare sono in archivi che raramente hanno dimensioni inferiori al milione di record. Senza un'adeguata padronanza delle tecnologie di archiviazione esistenti, il problema può risultare ingestibile fin dalle prime fasi dell'analisi.

Nozioni di statistica. Una volta organizzati i dati in un archivio interrogabile, il primo passo è quello di comprendere la struttura dei dati. Una volta comprese le relazioni fondamentali, sarà possibile identificare i dati interessanti e scartare i dati non adatti.

Nozioni di apprendimento automatico. Spesso una conoscenza dei fondamenti dell'apprendimento automatico può essere molto utile, sia per affiancare agli strumenti della statistica degli strumenti più specifici al particolare dominio, sia per impostare la base di dati fin dalle prime fasi della scoperta e renderla adatta ad un futuro utilizzo nel campo dell'intelligenza artificiale.

Conoscenza delle tecnologie di visualizzazione. Quando la dimensione dei dati o il numero di dimensioni delle variabili supera un certo limite, una corretta visualizzazione grafica può dare ottimi suggerimenti sulle direzioni verso cui dirigere l'esplorazione.

L'uso combinato di queste scienze o tecnologie permette di speculare informazioni sintetiche (conoscenza) da archivi precedentemente opachi alla vista del ricercatore.

In questo lavoro ho usato ben pochi tra gli strumenti offerti dall'evoluzione del campo del Data Mining avvenuta negli ultimi anni, ma comunque ritengo importante sottolineare come molte delle competenze richieste siano in effetti parte della formazione ingegneristica offerta dall'università italiana.

Capitolo 2

Folksonomie

Folksonomia è un neologismo che descrive il risultato di classificazione collaborativa di informazioni realizzata da più utenti con l'uso di parole chiave scelte liberamente.

Strutture di questo tipo si vengono a creare quando un gruppo di utenti utilizza una stessa applicazione per assegnare delle parole chiave (dette “tag”) a delle risorse. Solitamente le applicazioni sono accessibili in internet da un semplice browser.

Questa struttura si è diffusa come alternativa alle tassonomie, cioè alle organizzazioni gerarchiche con cui vengono classificate ad esempio le specie animali.

La profonda differenza tra le due organizzazioni è basata su due aspetti:

1. La tassonomia prevede l'organizzazione preventiva dei concetti in classi via via più specifiche.

La folksonomia non prevede la definizione di alcuna gerarchia tra le classi.

2. In una tassonomia ogni elemento è generalmente membro di una sola classe.

In una folksonomia ogni elemento è generalmente membro di più classi.

La costruzione di una tassonomia prevede lo studio del dominio degli elementi, finalizzato all'identificazione di attributi che permettano l'inserimento di un elemento in una singola classe.

Quindi spesso la creazione di una tassonomia viene delegata a degli esperti del dominio, che progettano la struttura che poi verrà usata dagli utenti.

Il processo di creazione di una folksonomia è invece completamente diverso. Gli organizzatori dell'informazione sono di solito gli utenti finali; infatti ogni utente ha la possibilità di classificare liberamente gli elementi, indipendentemente dalla classificazione effettuata da altri. In una folksonomia non esistono a priori delle relazioni gerarchiche tra classi, questo comporta un certo disordine nella struttura che però è compensato dalla semplicità con cui la classificazione può adattarsi a sfaccettature linguistiche e culturali molto varie.

Il termine, coniato da Thomas Vander Wal^{2.1}, ha origine dall'unione delle parole *folk* (o *folks*) e *tassonomia*. Tassonomia deriva dalle parole in lingua greca *taxis* e *nomos*. Il termine *taxis* significa *ordine*, il termine *nomos* (oppure *nomia*) significa *amministrazione*, infine *folk* significa *persone*.

Un'aggregazione con queste caratteristiche - che solo in seguito è stata definita folksonomia - è apparsa per la prima volta verso la fine del 2003 nel sito di bookmarking collettivo del.icio.us ed è stata rapidamente replicata nel cosiddetto *social software*.

2.1. Thomas Vander Wal: en.wikipedia.org/wiki/Thomas_Vander_Wal

I sistemi di *tagging* collaborativo, data l'elevata dinamicità e la continua interazione con gli utenti sono un interessante oggetto di studio; tra le pubblicazioni che trattano l'argomento si può citare il lavoro di Golder e Huberman del 2005 [11].

Esempio 2.1. Un esempio di classificazione possibile sia in una folksonomia che in una tassonomia è la classificazione bibliotecaria.

In ambito bibliotecario è molto diffusa la classificazione gerarchica Dewey^{2.2}.

Le dieci classi principali di questa tassonomia sono:

- 000.** Generalità
- 100.** Filosofia e psicologia
- 200.** Religione
- 300.** Scienze sociali
- 400.** Scienze del linguaggio
- 500.** Scienze naturali e matematica (Scienze pure)
- 600.** Tecnologia (Scienze applicate)
- 700.** Le arti
- 800.** Letteratura e retorica
- 900.** Geografia e storia

L'uso di una tassonomia è consigliato tutte le volte che il materiale da classificare ha una consistenza fisica, e quindi deve essere collocato in archivi e reperito con facilità.

La classificazione folksonomica ha senso invece solo per entità immateriali, dove la presenza contemporanea in più classi è facilmente ottenibile.

Ad esempio il testo *A vector space model for automatic indexing* (Salton 1975, *Comm. ACM*) può essere classificato come:

classificazione decimale Dewey. *D004.05 Elaborazione dei dati. Scienza degli elaboratori. Pubblicazioni in serie.*

classificazione in una folksonomia. *automatic compsci cs information information-retrieval informationretrieval ir linalg algebra linearalgebra math model no-tag retrieval review seminal space tf-idf tfidf vector vector-space vector-space-model vectors.*^{2.3}

Possiamo riassumere che la classificazione tassonomica permette di collocare e trovare facilmente il testo in una biblioteca, mentre la classificazione folksonomica permette di trovare più facilmente articoli di argomenti specifici, presentando una vasta gamma di possibili categorie, ognuna opinione di un particolare utente.

2.2. it.wikipedia.org/wiki/Classificazione_decimale_Dewey sviluppata da Melvil Dewey nel 1876.

2.3. i tag presentati provengono da CiteULike.

2.1 Esempi di folksonomie in internet

Oltre al già citato del.icio.us, le folksonomie si sono diffuse anche in altri campi diversi dalla sola gestione dei bookmark. Esistono infatti folksonomie di immagini, video o pubblicazioni accademiche.

Alcuni tra i più famosi siti che permettono la creazione di folksonomie sono:

Social Bookmarking.

- del.icio.us: del.icio.us
- Simpy: www.simpdy.com
- RawSugar: www.rawsugar.com

Condivisione video.

- YouTube: www.youtube.com

Condivisione immagini.

- Flickr: www.flickr.com^{2.4}

Siti di aggregazione e condivisione notizie.

- Digg: www.digg.com
- Technorati: www.technorati.com^{2.5}

Calendario eventi sociali.

- Upcoming: upcoming.org

Condivisione di riferimenti a pubblicazioni.

- CiteULike: www.citeulike.org
- BibSonomy: www.bibsonomy.org

2.2 Sistemi di bookmarking collettivo

La diffusione di internet ad alta velocità e basso costo ed in generale i crescenti tempi di connessione degli utenti hanno permesso lo sviluppo di sistemi per conservare i bookmark online.

Molti di questi sistemi stimolano (lasciando l'opzione come predefinita) la condivisione dei bookmark raccolti, e offrono varie soluzioni per navigare in modo trasversale tra raggruppamenti per utente (che ha inserito il link nel sistema) e raggruppamenti tematici. La maggior parte dei sistemi di bookmarking collettivo offre infatti la possibilità di associare ad un bookmark delle parole, etichette, "tag" con cui l'utente può classificare la risorsa inserita in archivio.

2.4. Flickr (sistema di condivisione di foto online) non costituisce una folksonomia completa, perché i tag sono associati alle foto e non sono personali per ogni utente.

2.5. Anche Technorati (aggregatore di blog) non costituisce una folksonomia completa, perché i tag sono opzione del solo autore dell'articolo.

Si viene quindi a creare una struttura molto complessa di collegamenti tra utenti, tag e risorse nella rete, struttura che prende il nome di *folksonomia* e che va idealmente a sovrapporsi alla *tassonomia*, l'organizzazione ad albero gerarchico, che ancora caratterizza le directory di siti web^{2.6}.

Anche se la ricerca è molto attiva riguardo a metodi innovativi per la navigazione delle folksonomie, attualmente il principali filtri per la selezione delle risorse (tipo GroupLens [20]) e dei tag^{2.7} sono essenzialmente basati sulla popolarità.

2.3 Interpretazione matriciale

Una folksonomia è una rete di connessioni tra utenti, tag e risorse. Ogni utente può inserire una risorsa ed associare alla risorsa delle parole che la caratterizzano.

Caratteristiche essenziali di una folksonomia sono:

- la possibilità di assegnare personalmente^{2.8} dei tag ad una risorsa,
- la possibilità di accedere a tutte le risorse classificate con un dato tag ed a tutti gli utenti che hanno usato tale tag.

Possiamo rappresentare una folksonomia come un cubo di valori binari, dove le tre dimensioni sono gli utenti, i tag e le risorse, e dove la cella corrispondente ad ogni intersezione è 1 se la relazione esiste, mentre è 0 se la relazione non esiste (come presentato in [5]'06).

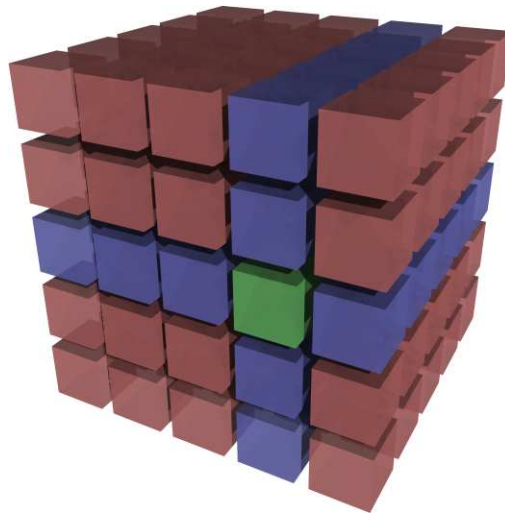


Figura 2.1. Interpretazione della folksonomia come cubo di valori binari.

Nota 2.1. D'ora in avanti la descrizione della folksonomia sarà limitata all'archivio di bookmark di del.icio.us, pertanto le generiche “risorse” da questo punto in poi saranno definite “href”.

^{2.6.} Directory gerarchiche di siti web come: dmoz.org o dir.yahoo.com

^{2.7.} Ad esempio la prima pagina di del.icio.us, vedi Sezione 3.1

^{2.8.} Per poter rappresentare una folksonomia in tre dimensioni l'associazione risorsa-tag deve essere riferita ad un particolare utente e non può essere collettiva.

Una volta trasposta la folksonomia nel cubo, è possibile immaginare tre proiezioni:

1. la proiezione che conteggia il numero di utenti in una matrice href - tag,
2. la proiezione che conteggia il numero di href in una matrice utente - tag,
3. la proiezione che conteggia il numero di tag in una matrice href - utente.

Definite queste proiezioni è facile vedere la somiglianza con le ben note matrici dell'occorrenza dei termini nei documenti usate nell'Information Retrieval (IR), ed in particolare viene naturale pensare alle interpretazioni vettoriali già usate con successo nell'ambito dei motori di ricerca [22]'75, come vedremo nel capitolo 4.

Definizione 2.1. *Siano:*

- *users* l'insieme degli utenti che hanno inserito almeno un href in archivio,
- *hrefs* l'insieme degli href inseriti in archivio da almeno un utente,
- *tags* l'insieme dei tag associati ad un href da almeno un utente.

Definizione 2.2. *Si definisce $\mathcal{S} \in \{0, 1\}^{|users| \times |hrefs| \times |tags|}$ la matrice binaria tridimensionale che associa utenti ad indirizzi e tag.*

Definita la matrice binaria che rappresenta la folksonomia completa, possiamo quindi definire le matrici di interi che derivano dalla somma nelle tre possibili proiezioni.

La **prima proiezione** è per numero di utenti:

Definizione 2.3. *Si definisce la matrice $\mathcal{U} \in \mathbb{N}^{|hrefs| \times |tags|}$*

come matrice che associa ad ogni coppia (href, tag) il numero di user presenti in archivio.

Definizione 2.4. *Si definisce il vettore $\mathcal{U}_{(h_i, \cdot)} \in \mathbb{N}^{1 \times |tags|}$ la riga i -esima della matrice \mathcal{U} .*

Analogamente si definisce il vettore $\mathcal{U}_{(\cdot, t_j)} \in \mathbb{N}^{|hrefs| \times 1}$ la colonna j -esima della matrice \mathcal{U} .

La **seconda proiezione** è per numero di indirizzi:

Definizione 2.5. *Si definisce la matrice $\mathcal{H} \in \mathbb{N}^{|users| \times |tags|}$*

come matrice che associa ad ogni coppia (tag, user) il numero di href presenti in archivio.

Definizione 2.6. *Si definisce il vettore $\mathcal{H}_{(u_i, \cdot)} \in \mathbb{N}^{1 \times |tags|}$ la riga i -esima della matrice \mathcal{H} .*

Analogamente si definisce il vettore $\mathcal{H}_{(\cdot, t_j)} \in \mathbb{N}^{|users| \times 1}$ la colonna j -esima della matrice \mathcal{H} .

La **terza proiezione** è per numero di tag:

Definizione 2.7. *Si definisce la matrice $\mathcal{T} \in \mathbb{N}^{|users| \times |hrefs|}$*

come matrice che associa ad ogni coppia (*user*, *href*) il numero di tag presenti in archivio.

Definizione 2.8. Si definisce il vettore $\mathcal{T}_{(u_i, \cdot)} \in \mathbb{N}^{1 \times |\text{hrefs}|}$ la riga *i*-esima della matrice \mathcal{T} . Analogamente si definisce il vettore $\mathcal{T}_{(\cdot, h_j)} \in \mathbb{N}^{|\text{users}| \times 1}$ la colonna *j*-esima della matrice \mathcal{T} .

2.4 Eredità dall'Information Retrieval

Tutte le volte che il fattore umano partecipa all'evoluzione di una quantità, conviene sempre aspettarsi un comportamento guidato dal principio di Pareto.

In particolare, come vedremo nel capitolo successivo, è facile aspettarsi che i dati risentano in qualche modo di una distribuzione tipo Zipf, distribuzione che spesso caratterizza dati creati dall'operato congiunto di più persone.

Questa situazione presenta una felice analogia con una altra configurazione già affrontata nel campo dell'IR, e per cui sono noti vari approcci ben documentati [3].

Uno di questi è l'uso di un filtro TF-IDF sui dati.

Sia n_t il numero di occorrenze del termine t all'interno del documento h di lunghezza $|h|$, si definisce "term frequency", TF, la frequenza del termine all'interno del documento. L'uso della frequenza relativa al posto del numero di occorrenze permette di confrontare documenti di dimensione diversa.

$$\text{TF}(h, t) \doteq \frac{n_t}{|h|}$$

Si definisce "inverse document frequency", IDF, la misura della rarità del termine nell'insieme dei documenti, misurata come logaritmo del rapporto tra il numero totale di documenti H e la popolarità del termine p_t .

$$\text{IDF}(t) \doteq \log \frac{H}{p_t}$$

Applichiamo adesso il filtro TF-IDF alla matrice $\mathcal{U} \in \mathbb{N}^{|\text{hrefs}| \times |\text{tags}|}$ e consideriamo come documento la riga (cioè l'href).

- Sia t il tag considerato,
- sia h l'href considerato,
- n_t è pari al numero di utenti $\mathcal{U}_{(h,t)}$ che hanno associato il tag t alla risorsa h ,
- sia $|h|$ la somma delle diverse occorrenze dei tag nell'href considerato $\|\mathcal{U}_{(h,\cdot)}\|_1$,
- sia p_t la popolarità del termine pari al numero di elementi diversi da zero del vettore $\mathcal{U}_{(\cdot,t)}$.

Possiamo quindi scrivere:

$$\text{TF}_{\mathcal{U}}(h, t) \doteq \frac{\mathcal{U}_{(h,t)}}{\|\mathcal{U}_{(h,\cdot)}\|_1}$$

Si definisce invece IDF:

$$\text{IDF}_{\mathcal{U}}(t) \doteq \log \frac{|\text{hrefs}|}{p_t}$$

Quindi per ogni matrice \mathcal{U} , \mathcal{H} , \mathcal{T} è possibile definire una matrice \mathcal{U}_{tf} , \mathcal{H}_{tf} , \mathcal{T}_{tf} dove ad ogni elemento originario è stato sostituito il valore filtrato:

$$\mathcal{U}_{\text{tf}} \leftarrow \text{TF}_{\mathcal{U}}(h, t) \cdot \text{IDF}_{\mathcal{U}}(t)$$

$$\mathcal{H}_{\text{tf}} \leftarrow \text{TF}_{\mathcal{H}}(u, t) \cdot \text{IDF}_{\mathcal{H}}(t)$$

$$\mathcal{T}_{\text{tf}} \leftarrow \text{TF}_{\mathcal{T}}(u, h) \cdot \text{IDF}_{\mathcal{T}}(h)$$

Sono in realtà possibili anche le tre matrici che interpretano termine e documento nell'altro senso, ma non sono state studiate, anzi la sola matrice filtrata su cui sono stati compiuti dei test è la matrice \mathcal{U}_{tf} .

L'applicazione del filtro TF-IDF ha l'effetto di mettere in evidenza i termini che caratterizzano in particolare un numero esiguo di documenti e di smorzare la presenza dei termini molto comuni. Gli effetti di questo filtro sulla qualità della previsione (vedi capitolo 7) sono molto difficili da interpretare e necessiterebbero di uno studio più approfondito.

2.4.1 Nuovi dati, vecchi metodi

Anche se a prima vista l'interpretazione di una folksonomia in forma di cubo può apparire solo gradevole dal punto di vista estetico, essa permette in realtà di accedere a metodi e strumenti per l'accesso alle informazioni ben studiati ed ottimizzati (ad esempio [4]'98 nell'ambito della gestione di strutture multidimensionali specializzate e [8]'01 nell'ambito del Data Mining). È quindi possibile che in poco tempo sia possibile portare una struttura per ora solo teorica ad operare in produzione semplicemente usando in simbiosi i nuovi dati con i vecchi metodi.

Capitolo 3

Analisi statistica della folksonomia di del.icio.us

3.1 Interfaccia di del.icio.us

Il sito internet del.icio.us è stato il primo esempio di folksonomia nel web, anzi, è stata la prima manifestazione di una struttura che solo in seguito ha preso il nome di folksonomia.

Del.icio.us offre la possibilità di associare dei tag a delle risorse, tipicamente indirizzi internet, ed è nato come strumento di uso personale per conservare in ordine i propri bookmark online.

L'interfaccia di del.icio.us è molto semplice. Appena si accede al sito viene visualizzata una prima pagina con i link più attivi al momento, ordinati secondo un ignoto algoritmo.

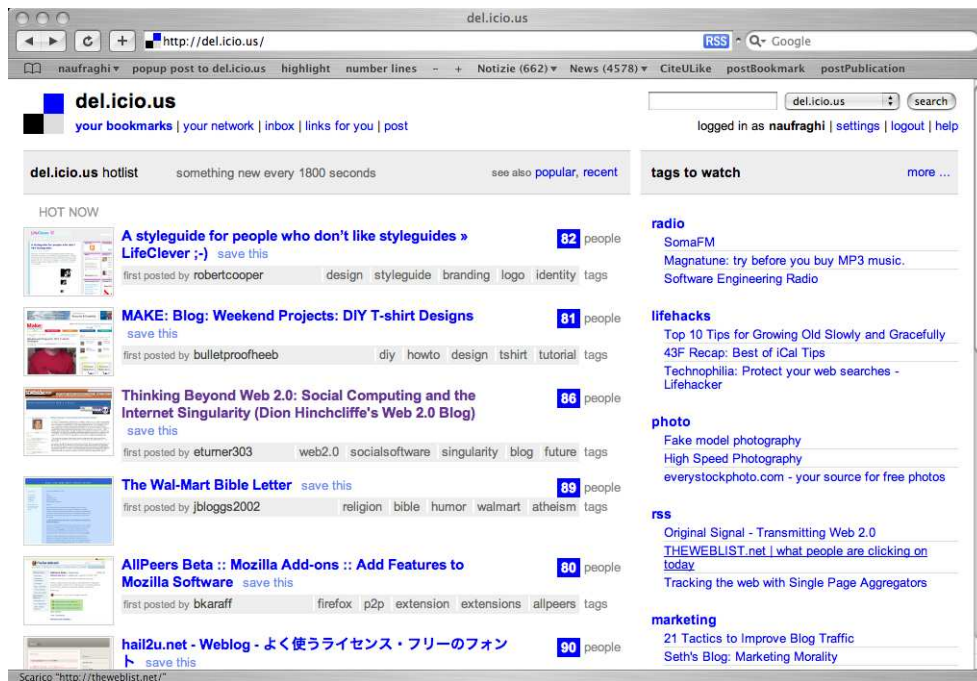


Figura 3.1. Prima pagina di del.icio.us (catturata il 2-9-06)

In alto nella pagina si possono notare due link, uno alla pagina dei popolari, “popular” ed uno alla pagina delle ultime modifiche, “recent”.

La pagina dei popolari presenta i link più diffusi (ordinati con un ignoto algoritmo che privilegia i link popolari ma anche attivi negli ultimi tempi).

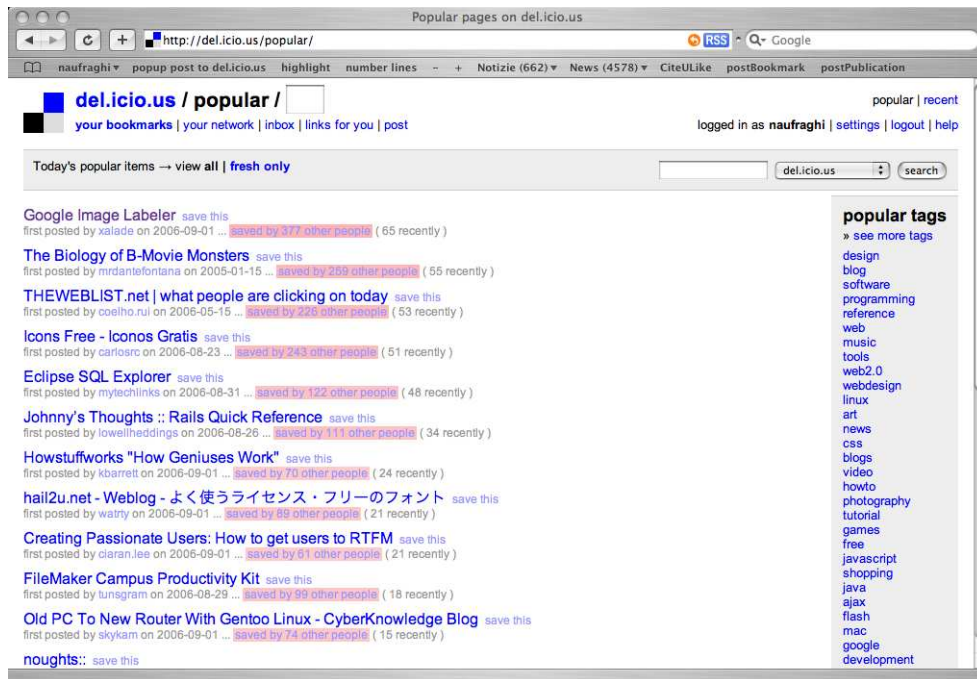


Figura 3.2. Pagina dei popolari di del.icio.us (catturata il 2-9-06)

La pagina delle ultime modifiche invece elenca gli ultimi indirizzi su cui c'è stata attività, indipendentemente dalla popolarità del link.

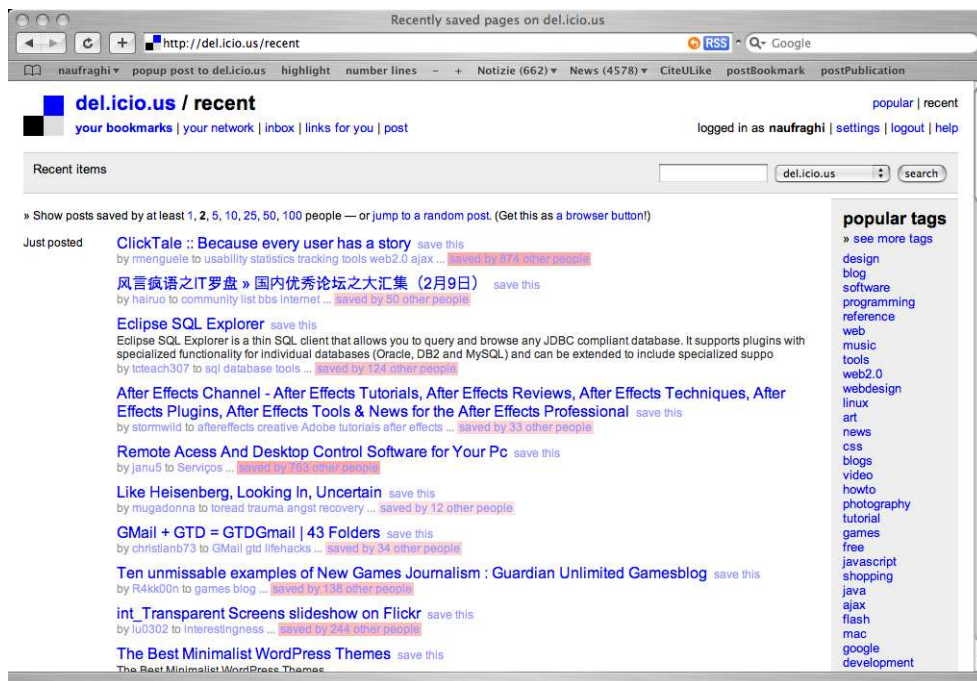


Figura 3.3. Pagina dei recenti di del.icio.us (catturata il 2-9-06)

Una volta iscritti al sistema è possibile creare un archivio personale di link con tag associati, e consultarlo tramite browser, usando per la navigazione i tag presentati sulla destra.

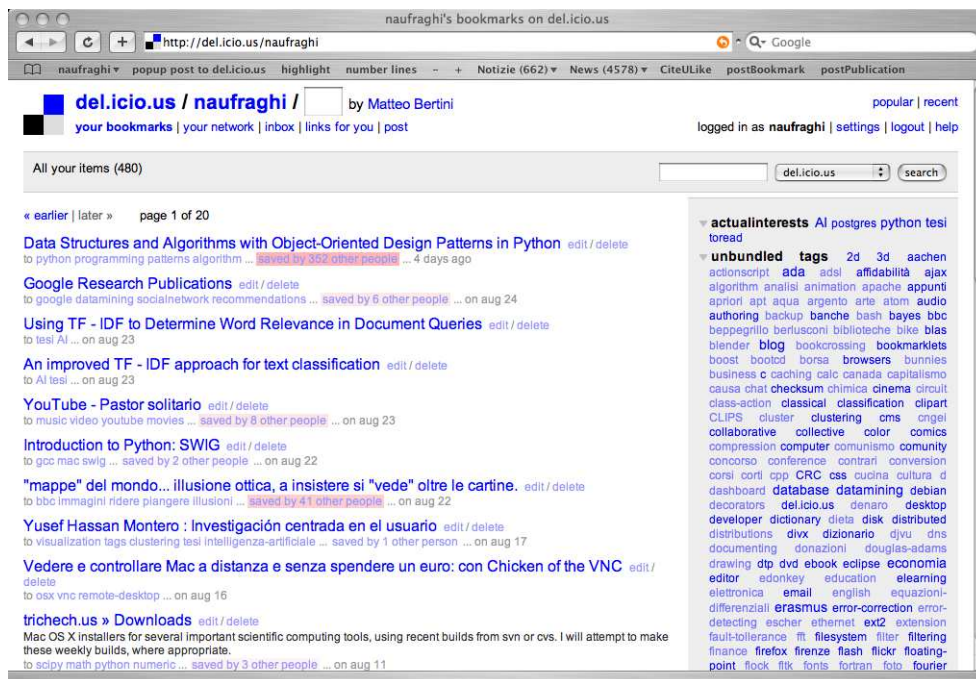


Figura 3.4. Pagina di un utente di del.icio.us (catturata il 2-9-06)

Anche se poco visibile in figura, i tag sono visualizzati in modo diverso in base al numero di documenti che “contengono”, cioè al numero di documenti a cui l’utente ha associato il tag.

Questo è un espediente per limitare la proliferazione di tag diversi, dovuti semplicemente ad una capitalizzazione diversa o all’uso di un plurale al posto del singolare; in tali casi infatti la stessa parola verrebbe interpretata dal sistema come due tag diversi.

Infatti l’interfaccia del software e l’usabilità in genere in questo campo sono fattori determinanti per il successo di un’applicazione.

Una particolarità da notare è che l’archiviazione dei siti in del.icio.us non è statica. Infatti è comune trovare il tag “toread”. Si suppone che questo tag rimanga associato ad una risorsa per un limitato lasso di tempo, concluso il quale:

- il tag è convertito in una classificazione più accurata, oppure
- la risorsa viene esclusa dall’archivio nel caso in cui l’interesse dell’utente non venga confermato.

3.2 Raccolta dei dati per l’esperienza

L’esperienza si basa sui dati scaricati per un mese (dal 15 giugno al 15 luglio 2006) dal sito di bookmarking collettivo del.icio.us. Il processo di download dei profili degli utenti è basato su tre programmi a linea di comando scritti in Python. Il primo programma (getusers.py) è programmato per scaricare ogni 30 minuti i nomi degli utenti presenti nella pagina dei popolari e nella pagina delle ultime aggiunte. Il secondo programma

(descrapper3.py) è programmato per operare in continuo scaricando il completo profilo degli utenti o aggiornando un profilo già scaricato. Il terzo programma (xml2pg.py) è programmato per concludere la sequenza inserendo in un database relazionale i dati scaricati.

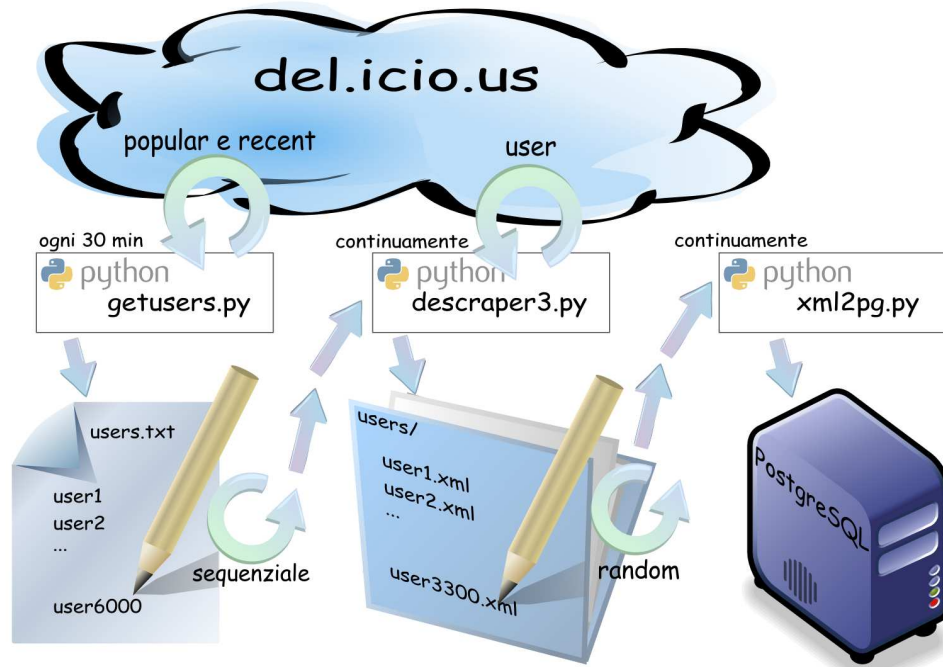


Figura 3.5. Processo di raccolta dei dati.

Le funzioni (API) di del.icio.us permettono il download facilitato di tutti i bookmark in un particolare formato XML, però limitano l'accesso ai bookmark al solo utente che li ha inseriti. Non permettono quindi ad un utente qualsiasi di scaricare i bookmark di altri utenti, ma solo di visualizzarli online in un browser.

Usando un analizzatore di testo programmato ad hoc viene ricreato un file XML con la stessa sintassi del file scaricabile mediante le API, contenente tutti i post degli utenti di cui avevo raccolto i nomi.

Per rispetto delle condizioni del servizio di del.icio.us, la velocità di download è stata limitata ad una pagina ogni 25 secondi (il limite inferiore suggerito per l'uso delle API è di una query al secondo). Il programma comunque è progettato per gestire il messaggio di errore 503 (il messaggio inviato dal server per rallentare le richieste troppo insistenti) e raddoppiare il tempo di attesa fino ad un massimo prestabilito in cui il programma termina.

Tutto il software sviluppato è scritto in Python^{3.1}; questo mi ha permesso di spostare agilmente il software tra i sistemi operativi con cui lavoro solitamente (Mac OS X e GNU/Linux), e di accedere ad una vasta gamma di librerie, in particolare Beautiful

3.1. È interessante notare come grazie alle potenti librerie disponibili tutto il software sviluppato superi di poco le 1000 righe. La sintesi e la flessibilità di un linguaggio come Python facilitano il veloce prototipaggio necessario nel campo dell'informatica.

Soup^{3.2} per gestire l'estrazione di informazioni dal codice HTML e scipy^{3.3} per l'elaborazione dei dati in forma di matrici.

A causa delle risorse limitate e dei vincoli imposti da del.icio.us, degli oltre 6000 username raccolti, nel mese di test sono stati scaricati i profili XML di circa 3300, di cui circa 2500 sono stati inseriti in archivio (PostgreSQL).

I dati presenti nel file XML dei post di un utente di del.icio.us sono:

1. il nome utente (univoco)
2. la risorsa inserita
3. i tag associati dall'utente alla risorsa

I dati sono organizzati in una semplice struttura osservabile in un estratto:

Estratto del file XML che rappresenta il profilo di un utente
<pre><?xml version='1.0' encoding='utf-8'?> <posts tag="" user="username"> <post href="http://blog.plasticsfuture.org/2006/04/23/mac-backup-software-harmful/" description="plasticsfuture >> Mac Backup Software Harmful" extended="" tag="apple macosx backup article comparisson" time="2006-04-25T08:43:22Z" /> ... </posts></pre>

Come si può osservare dai caratteri asiatici presenti nelle figure 3.2 e 3.3 il sito del.icio.us è completamente internazionalizzato. È stato infatti necessario gestire la codifica UTF-8^{3.4} per importare correttamente i caratteri particolari.

3.3 Sintesi statistica dei dati

Al momento dei test l'archivio comprende:

- circa 2.500 utenti,
- circa 200.000 tag,
- circa 1.000.000 di href.

Questo significa che i dati studiati in questo lavoro sono di un ordine di grandezza più consistenti dei dati studiati in altri lavori precedenti [11].

3.2. www.crummy.com/software/BeautifulSoup/

3.3. www.scipy.org

3.4. UTF-8 (Unicode Transformation Format, 8 bit) è una codifica a lunghezza variabile dei caratteri Unicode: it.wikipedia.org/wiki/UTF-8

Da osservare che la tabella di collegamento UHT, la tabella che rappresenta il cubo \mathcal{S} nella base di dati, contiene circa 4 milioni di record ($4 \cdot 10^6$), valore estremamente ridotto rispetto ai possibili $5 \cdot 10^{14}$ elementi del cubo. Si conferma quindi l'estrema sparsità presentata come problema dei sistemi di suggerimento collaborativo.

Vediamo di cogliere alcuni aspetti dei dati presenti in archivio.

Abbiamo detto che ogni utente può inserire in archivio degli href classificandoli con dei tag.

Quindi ad ogni utente è associato un insieme di href ed un insieme di tag.

Lo stesso vale per gli href, che hanno associato un insieme di utenti ed un insieme di tag, e per i tag, che hanno associati un insieme di utenti ed un insieme di href.

Vediamo di comprendere le caratteristiche di questi insiemi.

I dati in tabella sono il minimo, il massimo, la media e la deviazione standard di:

- a) numero di href inseriti da un utente,
- b) numero di tag distinti usati da un utente,
- c) numero di utenti che hanno in archivio un dato href,
- d) numero di tag distinti associati ad un dato href,
- e) numero di href associati ad un dato tag,
- f) numero di utenti che usano un dato tag.

		min	max	media	std
a)	href per user	1	16.621	615	1.152
b)	tag per user	1	9.418	306	589
c)	user per href	1	415	1,5	3,4
d)	tag per href	1	333	3,9	5,5
e)	href per tag	1	33.899	21	296
f)	user per tag	1	1.507	4	27

Tabella 3.1. Relazioni tra user, href e tag.

Dobbiamo ricordare però che la stima sintetica per queste quantità non è significativa, dato che non ci sono garanzie che la distribuzione sia di tipo normale. Fortunatamente anche solo osservando alcuni grafici è possibile comprendere come i dati assumano la forma di altre distribuzioni note.

Nota 3.1. Per ridurre il rumore, in particolare nelle zone finali della scala logaritmica, i dati sono stati filtrati con la funzione:

$$\text{loground}(x) = \text{round}(10^{\text{round}(\log_{10}(x), 1)}, 0)$$

Funzione che in pratica varia la dimensione della base dell'istogramma in funzione della distanza dall'origine in modo da compensare la scala logaritmica dell'asse.

3.3.1 Proiezioni primarie

Come prima analisi procediamo conteggiando il numero di href per utente. La figura che segue presenta in ascissa il numero degli href ed in ordinata il numero di utenti che hanno in archivio tale numero di href.

Possiamo assumere che la distribuzione sia di tipo logonormale (l'asse delle ascisse è in scala logaritmica) e con una stima molto approssimativa possiamo dire che la maggior parte degli utenti ha un numero di href tra i 100 ed i 1000. L'area sottesa alla curva (non tracciata ma sufficientemente esplicita in figura) è pari al numero totale di utenti, cioè circa 2500.

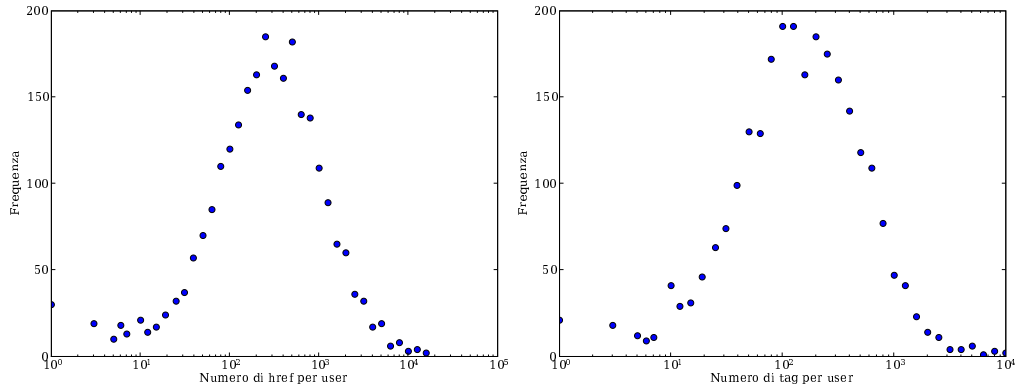


Figura 3.6. Distribuzioni del numero di href e di tag per utente

Operando in modo analogo per i tag, possiamo osservare che la distribuzione ha un aspetto molto simile alla distribuzione degli href ma stavolta la maggioranza degli utenti ha un numero di tag che è concentrato tra i 50 ed i 500.

Importante 3.1. Possiamo riassumere che le distribuzioni guidate dall'operato di un singolo utente sono sostanzialmente di tipo lognormale.

Nel caso invece di distribuzioni guidate dall'operato combinato di molti utenti, la situazione è molto diversa e le distribuzioni tendono ad assomigliare alla distribuzione di Zipf.

Le prossime immagini rappresentano i dati raccolti per href.

La figura 3.7 rappresenta l'istogramma della popolarità degli href. In ascissa abbiamo il numero di utenti che hanno inserito in archivio l'href (la popolarità dell'href) mentre in ordinata abbiamo il numero di href che hanno la popolarità appena trovata.

Possiamo osservare come vi siano moltissimi (10^6) href con popolarità pari ad 1, cioè presenti nell'archivio di un singolo utente, mentre vi sono pochi href molto popolari, cioè presenti nell'archivio di molti utenti.

I 10 href più popolari dell'archivio sono:

1. pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html

2. www.adaptivepath.com/publications/essays/archives/000385.php
3. wellstyled.com/tools/colorscheme2/index-en.html
4. www.goodfonts.org/
5. dietrich.ganx4.com/foxylicious/
6. pro.html.it/esempio/nifty/
7. www.beelerspace.com/index.php?p=890
8. developer.apple.com/internet/webcontent/xmlhttpreq.html
9. www.datsi.fi.upm.es/~frosal/docs/25mdq.html
10. www.techsupportalert.com/best_46_free_utilities.htm

Questa lista differisce profondamente dalla lista presente nella prima pagina di del.icio.us in quanto in questa classifica non viene considerato il fattore tempo.

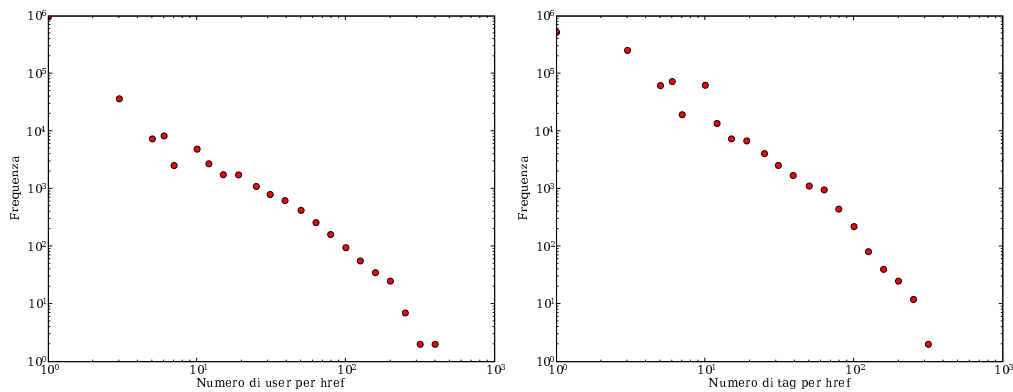


Figura 3.7. Distribuzioni del numero di utenti e di tag per href

L'area sottesa alla curva è pari al numero totale di href, cioè circa 1 milione.

Analizziamo adesso il grafico in figura 3.7. Questo è il grafico del numero complessivo di tag diversi per href. In ascissa abbiamo il numero di tag usati per definire un dato href (tutti i tag distinti usati da tutti gli utenti) ed in ordinata abbiamo il numero di href caratterizzati con lo stesso numero di tag.

Possiamo osservare che ci sono pochi siti caratterizzati con un gran numero di tag diversi, mentre la maggior parte dei siti è caratterizzata da un numero molto limitato di tag diversi.

Prendiamo ad esempio un sito piuttosto noto, come ad esempio, www.techsupportalert.com/best_46_free_utilities.htm, sito che presenta una raccolta di programmi gratuiti di vario genere. Osserviamo come il contributo di ogni utente tenda ad evidenziare un particolare aspetto di una complessa classificazione, fino ad arrivare ad una lunga lista di termini (in questo caso 92 termini diversi, ottenuti considerando solo i tag condivisi da almeno 6 diversi utenti):

viewer, zip, reference, Windows, rootkit, replace, osx, notes, trojan, list, linux, screenshot, spam, general, editor, email, internet, security, inventory, unix, antivirus, files, protection, upload, ripper, interesting, ftp, web, anonymous, pictures, misc, multimedia, Free, application, search, processes, images, Freeware, bittorrent, spyware, registry, clipboard, p2p, Utilities, lists, win32, Personal, downloads, bestof, os, screencapture, organize, resources, unread, html, notepad, adware, firewall, desktop, technology, synchronization, bookmarks, browser, tcp, programs, find, mac, popular, file, popup, tweaks, Software, archive, toread, photos, outline, productivity, utility, IT, resource, download, computers, best, tool, computer, useful, tools, free, utilities, windows, freeware, software

Vi è quindi un effetto combinato tra la popolarità e l'argomento trattato che porta anche il numero di tag ad assumere una distribuzione di tipo Zipf.

Vediamo adesso le distribuzioni organizzate per tag. Per prima osserviamo la distribuzione del numero di utenti per tag, cioè conteggiamo la popolarità di ogni tag (in ascissa) e contiamo quanti tag hanno la popolarità trovata. In questo caso la curva si allontana in modo più evidente dalla curva teorica sia nella parte iniziale che nella parte finale, ma in questa fase non sono state approfondite le ragioni di tale comportamento.

L'area sottesa alla curva è pari al numero di tag, cioè circa 200.000.

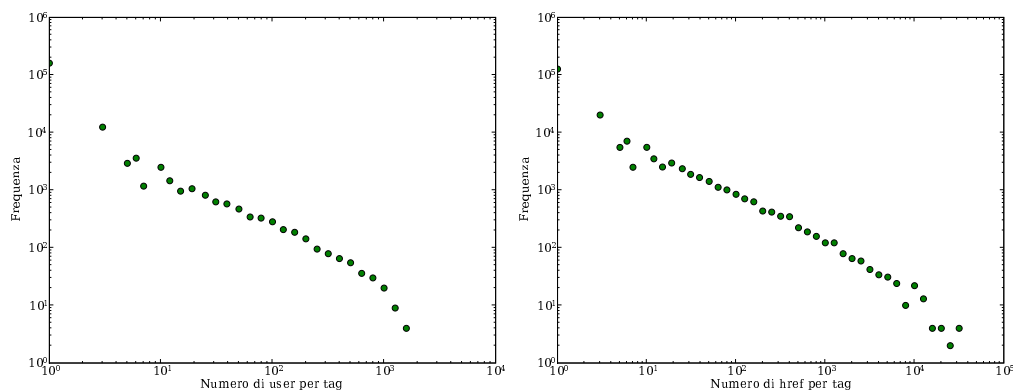


Figura 3.8. Distribuzioni del numero di utenti e di href per tag

In generale, come quasi di consueto in questo archivio, vi sono molti tag poco popolari tra gli utenti (si pensi ad esempio all'uso dei termini in lingua non inglese), mentre vi sono pochi tag particolarmente popolari.

A titolo di esempio l'elenco dei 20 tag più popolari in ordine decrescente di popolarità è:

css, web, design, music, google, software, javascript, firefox, blog, programming, news, search, art, del.icio.us, rss, linux, reference, php, tools, flash

Altra proiezione è quella della diffusione dei tag all'interno dei documenti. In questo caso abbiamo in ascissa il numero di documenti associati al tag, ed in ordinata il numero di tag con la stessa diffusione. Un'alta diffusione caratterizza spesso termini generici, mentre tag a bassa diffusione sono spesso termini più specifici, per ragioni semantiche o linguistiche.

A titolo di esempio l'elenco dei 20 tag più diffusi in ordine decrescente di diffusione è:

web, design, tools, reference, software, webdev, programming, css, blog, webdesign, fun, howto, tool, art, weblog, html, toread, cool, internet, tech

Importante 3.2. Possiamo riassumere che le distribuzioni guidate dall'operato congiunto di più utenti sono sostanzialmente di tipo zipfiano.

Questa è la ragione fondamentale dei problemi che si presentano nei sistemi di suggerimento collaborativo.

3.3.2 Proiezioni secondarie

Oltre alle sei proiezioni primarie presentate fino a questo punto è possibile anche studiare le tre diverse proiezioni del cubo \mathcal{S} .

Se consideriamo la proiezione \mathcal{U} che conteggia il numero di utenti che hanno associato un dato tag ad un dato href e rappresentiamo l'istogramma di queste quantità, possiamo osservare che esistono associazioni molto diffuse ed associazioni molto rare, secondo una legge tipo Zipf.

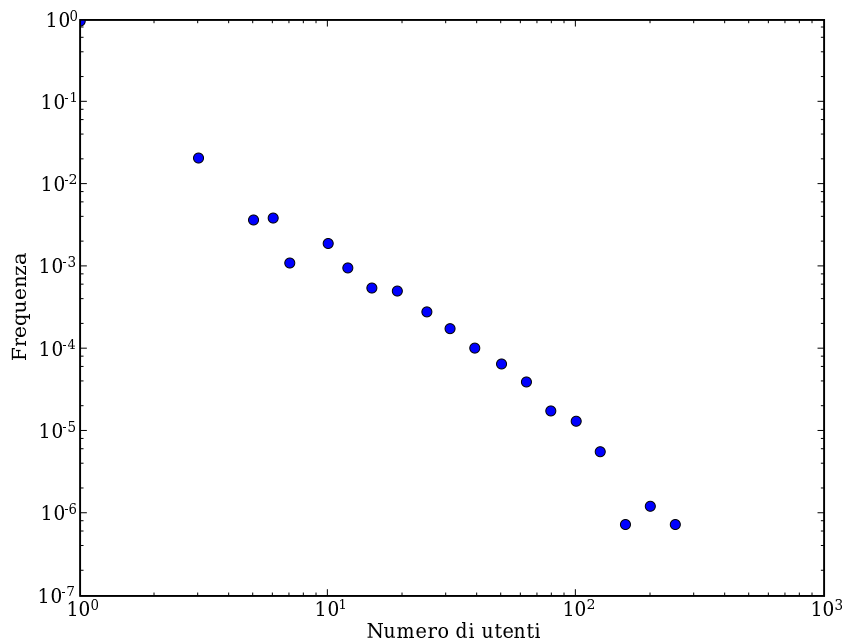


Figura 3.9. Numero di utenti per coppia (tag, href)

Le prime 10 coppie più popolari sono, in ordine decrescente di popolarità:

del.icio.us. pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html

css. pro.html.it/esempio/nifty/

ajax. www.adaptivepath.com/publications/essays/archives/000385.php

fonts. www.goodfonts.org/

firefox. dietrich.ganx4.com/foxylicious/

design. wellstyled.com/tools/colorscheme2/index-en.html

google. www.google.com/help/cheatsheet.html

javascript. developer.apple.com/internet/webcontent/xmlhttpreq.html

delicious. pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html

cooking. www.hertzmann.com/articles/2005/fables/

Una situazione analoga si presenta anche andando a contare il numero degli href associati ad un tag da un utente nella proiezione \mathcal{H} . Possiamo vedere che anche in questo caso la distribuzione globale della diffusione dei tag si ripropone anche considerando separatamente i vari utenti.

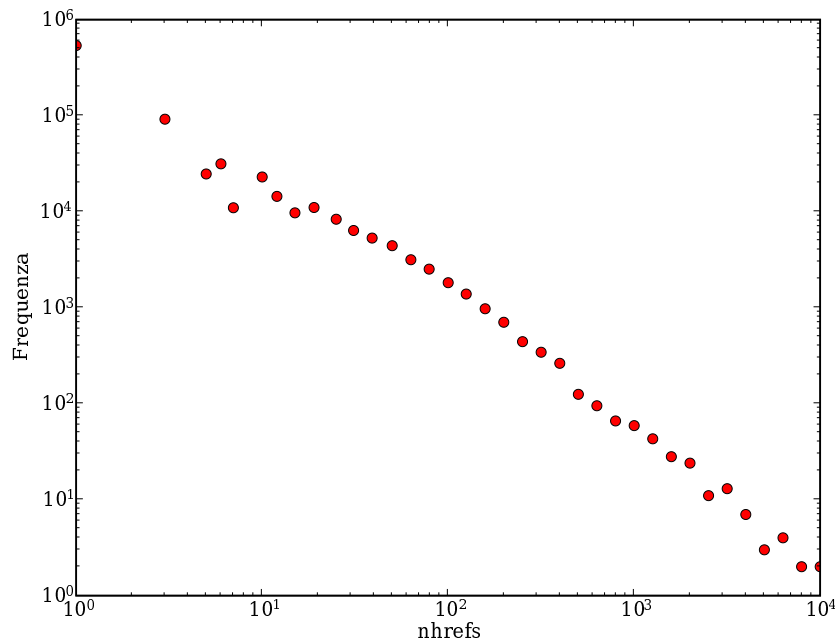


Figura 3.10. Numero di href per coppia (user, tag)

La prossima figura non è stata filtrata con la funzione $\text{logground}(x)$ per evidenziarne alcuni particolari.

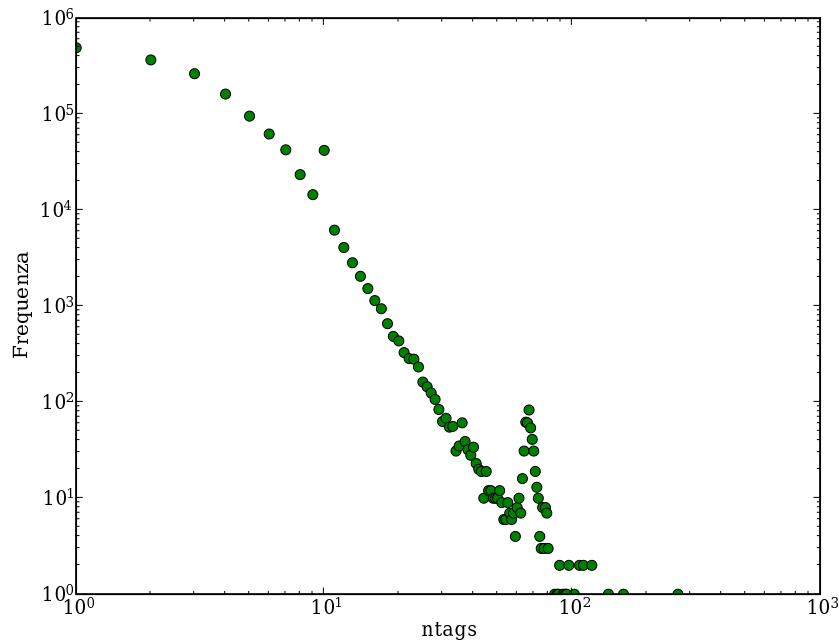


Figura 3.11. Numero di tag per coppia (user, href)

La curva delle occorrenze della coppia (user, href), cioè il numero di tag con cui un dato utente descrive un dato href è molto particolare.

La curva presenta infatti delle caratteristiche molto marcate che non permettono di interpretare l'istogramma come rappresentativo di una distribuzione tipo Zipf.

Possiamo comunque riassumere che in modo ancora più marcato che nel caso di una distribuzione zipfiana, il numero di tag che un utente associa mediamente ad un href è condensato tra 1 e 10.

La curva presenta uno strano picco nella zona dei 70 tag; non è chiara la ragione di questa caratteristica della curva, ma è plausibile che dato il numero relativamente limitato di utenti inseriti in archivio, la distorsione possa essere dovuta anche a pochi utenti che hanno inserito 100 siti usando 70 tag per definirli.

3.4 Evoluzione temporale dei dati

Una caratteristica tipica di questo archivio è il rinnovo continuo dei dati. Nelle distribuzioni appena presentate possiamo osservare che solo circa il 20% degli href di un utente è condiviso con almeno un altro utente. Il restante 80% è composto da href di cui l'unica opinione esistente è quella dell'utente che li ha inseriti.

Questo fattore non è stato studiato approfonditamente per via della limitatezza del campione raccolto. Dalle distribuzioni assunte dai dati si può comunque ipotizzare che la proporzione tra gli elementi comuni a più utenti e gli elementi supportati da un singolo utente sia strutturale e quindi più o meno stabile nel tempo.

Possiamo dedurre che, se si opera esclusivamente all'interno della folksonomia, è possibile fare previsioni anche basate sull'opinione di un singolo utente, ma non è possibile verificare la qualità della previsione senza ricorrere ad un esperimento dal vivo.

Per questa ragione gli esperimenti sono stati compiuti sulla sola parte comune dell'archivio.

Possiamo in parte giustificare la sensatezza di questo approccio considerando i numeri assoluti invece delle proporzioni. Infatti anche se il base di dati comune è una parte ridotta della base di dati completa, il numero di indirizzi distinti presenti è comunque molto elevato.

Nei dati di esempio, con soli 2500 utenti inseriti la base di indirizzi comuni è di circa 140.000 indirizzi internet (su un totale di circa 1 milione), un numero comunque ingestibile dall'utente senza un applicativo di supporto.

Per avere un'immagine dell'evoluzione temporale dei dati possiamo osservare l'evoluzione mensile del numero di nuovi href, tag ed utenti.

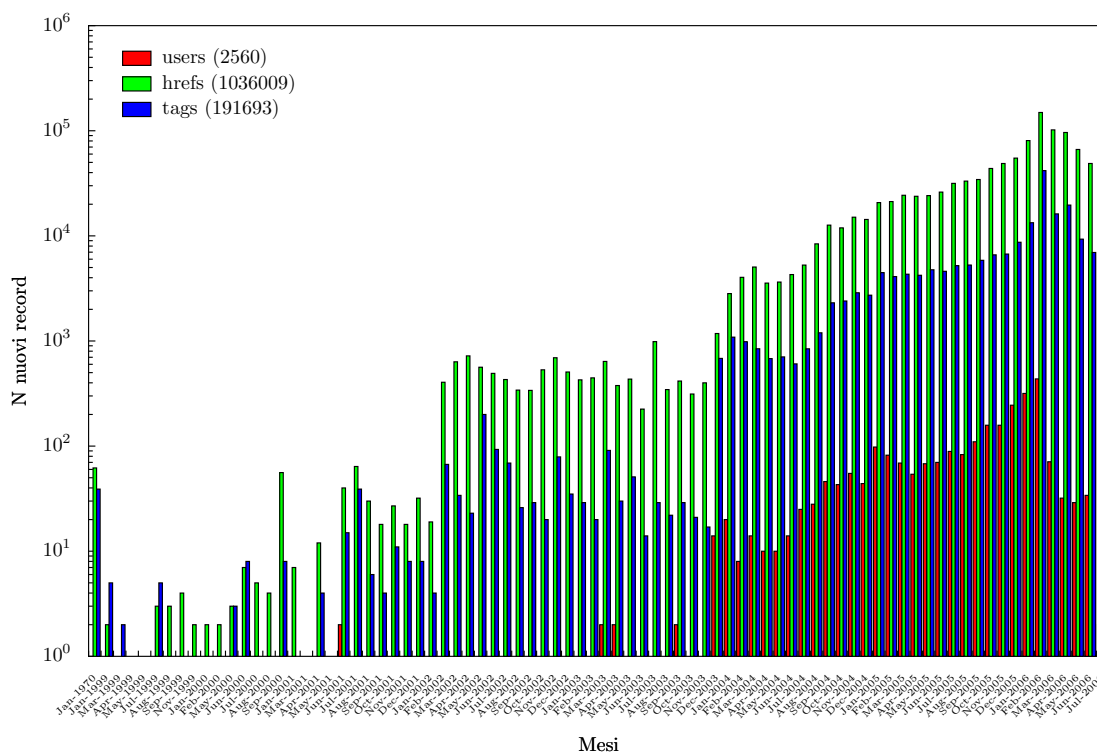


Figura 3.12. Nuovi ingressi per mese

È possibile osservare come il grafico dei nuovi ingressi sia visibilmente in crescita, anche usando una rappresentazione logaritmica dei dati (la parte finale del grafico è quasi sicuramente disturbata dall'interruzione del campionamento).

La crescita continua dell'archivio è quindi un aspetto fondamentale della struttura delle folksonomie.

È ragionevole immaginare che gli interessi di un utente varino nel tempo. Ogni utente avrà una base di interessi più o meno stabili ed un numero variabile di interessi temporanei.

Quindi un sistema di suggerimento evoluto deve essere abbastanza flessibile da gestire cambiamenti di questo tipo nella personalità dell'utente.

In questa fase dei lavori il fattore temporale è stato escluso dalle valutazioni, ma, grazie alla flessibilità della struttura proposta, non è difficile immaginare una modifica all'operatore di proiezione che pesi in modo diverso gli elementi binari in \mathcal{S} in funzione della data associata.

Capitolo 4

Misure di similarità nelle folksonomie

Prima di definire le misure di similarità vediamo brevemente come è definito un problema di apprendimento automatico.

4.1 Apprendimento automatico

Nell'apprendimento automatico forniamo ad una *learning machine* un insieme di dati ed un corrispondente insieme di etichette.

$$\mathcal{D}_\ell = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_\ell, y_\ell)\} \subseteq (X \times Y)^\ell$$

Solitamente lo spazio dei dati in ingresso è un sottospazio di \mathbb{R}^d , $X \subseteq \mathbb{R}^d$ e quindi un vettore di ingresso ha la forma:

$$\mathbf{x}_i = (x_1, x_2, \dots, x_d)$$

Una macchina che operi sull'ipotesi di avere una relazione lineare tra ingresso ed uscita viene definita *learning machine* lineare.

In genere siamo interessati a predire y per configurazioni di \mathbf{x} non presenti in archivio. Per molti problemi intuitivamente l'approccio è: vogliamo che ingressi simili producano risultati simili.

Per formalizzare questo approccio dobbiamo definire cosa intendiamo per *simili*.

Riguardo ai risultati, di solito si ricorre ad una *funzione di loss*; nel semplice caso della classificazione binaria, l'elemento può risultare classificato correttamente o classificato erroneamente, altre funzioni di loss possono prevedere la presenza di diversi gradi di errore.

Per gli ingressi invece la nozione di similarità è più complessa.

Una nozione di similarità particolarmente semplice ma sorprendentemente utile è la nozione di similarità che deriva dal considerare gli elementi parte di uno spazio Euclideo ed utilizzare concetti geometrici. Ad esempio nelle *Support Vector Machine* [24], la similarità è misurata come prodotto scalare in uno spazio delle caratteristiche (*feature space*) $\mathfrak{F}\mathfrak{s}$, solitamente caratterizzato da un'alta dimensionalità. Formalmente gli attributi vengono mappati in $\mathfrak{F}\mathfrak{s}$ mediante una funzione $\phi: X \rightarrow \mathfrak{F}\mathfrak{s}$, $\mathbf{x} \mapsto \phi(\mathbf{x})$, ed in seguito confrontati mediante il prodotto scalare $\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$. Per evitare di lavorare in spazi ad elevata dimensionalità, è stato trovato un trucco, il cosiddetto *trucco del kernel* [19], cioè una funzione (anche non lineare) che permette di calcolare il prodotto scalare nello spazio delle caratteristiche direttamente dallo spazio degli attributi:

$$K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \tag{4.1}$$

Spesso si definisce un kernel ignorando la forma assunta dalla funzione ϕ , anche perché è dimostrato che (4.1) è ben definita per ogni kernel che rispetti le condizioni di Mercer [7], o equivalentemente che sia un kernel definito positivo [6].

Definizione 4.1. *Kernel definito positivo.*

Una funzione simmetrica $K: X \times X \rightarrow \mathbb{R}$ che per ogni $m \in \mathbb{N}$, $\mathbf{x}_i \in X$ origini una matrice di Gram positiva, o analogamente che per ogni $c_i \in \mathbb{R}$ valga

$$\sum_{i,j=1}^m c_i c_j G_{ij} \geq 0 \text{ con } G_{ij} \doteq K(\mathbf{x}_i, \mathbf{x}_j),$$

è detta kernel definito positivo.

4.2 Misure dirette

Una volta definite le tre matrici proiezione è possibile definire due misure per ogni entità rappresentata nel cubo.

Rispetto ai tag è possibile definire.

- Una misura nello spazio degli href che usa come modulo il numero di utenti.

Questa misura confronta i tag in base al numero di utenti che hanno associato un tag ad un particolare sito internet. Per esempio, se utenti diversi hanno usato le parole “cat” e “gatto” riferite allo stesso sito, il sistema considera le due parole più simili di “gatto” e “politica” che magari non compaiono mai associate allo stesso sito.

- Una misura nello spazio degli utenti che usa come modulo il numero di href.

Questa misura indica quanto un dato argomento viene accomunato ad un altro negli interessi di una persona, ma non è stata studiata.

Rispetto agli utenti è possibile definire.

- Una misura nello spazio degli href che usa come modulo il numero di tag.

Questa misura indica quanto sono simili due utenti considerando i loro interessi (i siti web che hanno salvato in archivio).

- Una misura nello spazio dei tag che usa come modulo il numero di href.

Questa misura indica quanto sono simili due utenti in base alle parole con cui hanno definito i siti archiviati.

Rispetto agli href è possibile definire.

- Una misura nello spazio degli utenti che usa come modulo il numero di tag.

Questa misura indica quanto un dato sito viene accomunato ad un altro negli interessi di una persona, ma non è stata studiata.

- Una misura nello spazio dei tag che usa come modulo il numero di utenti.

Questa misura indica quanto due siti sono simili in base alla descrizione fatta dagli utenti.

Mentre alcune di queste misure hanno una possibile interpretazione e sono state usate con successo in questo lavoro, altre misure sono di difficile interpretazione e non hanno attualmente un chiaro significato.

Anche se in questo lavoro non sono state utilizzate, le misure di similarità sono state definite in forma di kernel [1] per permettere un facile passaggio ad algoritmi di apprendimento automatico più evoluti.

Definizione 4.2. *Si definisce kernel tra user nello spazio degli href:*

$$K_{u\mathcal{H}u}(u, u') = \sum_{h \in \text{hrefs}} \mathcal{T}_{(u,h)} \cdot \mathcal{T}_{(u',h)} = \langle \mathcal{T}_{(u,\cdot)}, \mathcal{T}_{(u',\cdot)} \rangle$$

Analogamente si definisce kernel tra user nello spazio dei tag:

$$K_{u\mathcal{T}u}(u, u') = \sum_{t \in \text{tags}} \mathcal{H}_{(u,t)} \cdot \mathcal{H}_{(u',t)} = \langle \mathcal{H}_{(u,\cdot)}, \mathcal{H}_{(u',\cdot)} \rangle$$

Definizione 4.3. *Si definisce kernel tra href nello spazio dei tag:*

$$K_{h\mathcal{T}h}(h, h') = \sum_{t \in \text{tags}} \mathcal{U}_{(h,t)} \cdot \mathcal{U}_{(h',t)} = \langle \mathcal{U}_{(h,\cdot)}, \mathcal{U}_{(h',\cdot)} \rangle$$

Analogamente si definisce kernel tra href nello spazio degli user:

$$K_{h\mathcal{U}h}(h, h') = \sum_{u \in \text{users}} \mathcal{T}_{(u,h)} \cdot \mathcal{T}_{(u,h')} = \langle \mathcal{T}_{(\cdot,h)}, \mathcal{T}_{(\cdot,h')} \rangle$$

Definizione 4.4. *Si definisce kernel tra tag nello spazio degli href:*

$$K_{t\mathcal{H}t}(t, t') = \sum_{h \in \text{hrefs}} \mathcal{U}_{(h,t)} \cdot \mathcal{U}_{(h,t')} = \langle \mathcal{U}_{(\cdot,t)}, \mathcal{U}_{(\cdot,t')} \rangle$$

Analogamente si definisce kernel tra tag nello spazio degli user:

$$K_{t\mathcal{U}t}(t, t') = \sum_{u \in \text{users}} \mathcal{H}_{(u,t)} \cdot \mathcal{H}_{(u,t')} = \langle \mathcal{H}_{(\cdot,t)}, \mathcal{H}_{(\cdot,t')} \rangle$$

Nota 4.1. Ricordiamo che dato un kernel $K(x, y)$ è possibile calcolare il kernel normalizzato $\bar{K}(x, y)$ come:

$$\bar{K}(x, y) = \frac{K(x, y)}{\sqrt{K(x, x) \cdot K(y, y)}}$$

Per ogni kernel non normalizzato appena definito, mediante la formula di normalizzazione si definisce l'analogo kernel normalizzato.

Nota 4.2. Osserviamo che dato un kernel $K(x, y)$, è possibile ricavare la distanza in norma 2 (nel feature space indotto dal kernel) mediante la relazione:

$$d(x, y) = \sqrt{K(x, x) - 2 \cdot K(x, y) + K(y, y)}$$

4.2.1 Alcuni esperimenti qualitativi

Prima di procedere con degli esperimenti quantitativi ho preferito condurre qualche rapido esperimento qualitativo. Anche se questi esperimenti non sono utili per dimostrare la validità di una misura di similarità, sono stati di supporto per comprendere alcune caratteristiche delle misure utilizzate.

Prendendo spunto dalla valutazione comparativa presentata in [23]'05 ho scelto di sfruttare due misure, una misura di similarità basata sul prodotto scalare normalizzato, equivalente alla "distanza coseno" definita in [22]'75, ed una misura basata sul prodotto scalare non normalizzato, misure formalmente definite nella sezione 4.2.

Scartate le misure tra utenti ed href per la problematica valutazione del concetto di similarità, l'unica misura che permette una analisi qualitativa è la misura tra tag.

Negli esempi che seguono possiamo osservare che l'ordinamento per similarità proposto dal kernel non normalizzato $K_{\mathcal{H}t}(t, t')$ appare ragionevole:

- *blog, blogs, Blogs, blogging, weblog, weblogs, daily, links, Blog, blogtool*
- *news, daily, News, headlines, slashdot, World, media, technology, geek, tech*

È possibile anche notare che se operiamo sulla folksonomia nel suo insieme, la polisemia (termini con significato diverso in base al contesto) non viene gestita; infatti tendono ad emergere i significati più comuni. Riflettendo si può ipotizzare che la polisemia possa essere gestita solamente facendo riferimento ad una particolare risorsa o ad un particolare utente. Solo in questo modo è possibile dedurre che per un affezionato lettore del NYTimes, la parola "news" è più simile a "politics" che a "tech".

Procedendo con altri test qualitativi ho notato che la misura normalizzata tende a promuovere le parole che hanno delle strette relazioni con la parola cercata, mentre la misura non normalizzata ha l'effetto di generalizzare il significato, promuovendo le parole sì correlate ma nello stesso tempo anche più diffuse.

La ricerca di parole meno frequenti mette in evidenza questo fattore (in grassetto la parola cercata):

misura normalizzata.

***environment**, climate, Energy, green, eco, Environment, environmentalism, earthquake, environmental, sustainability*

misura non normalizzata.

***environment**, science, blog, politics, design, green, future, weblog, humor, blogs*

Questo effetto è più evidente cercando una parola ancora meno diffusa, come "environmentalism" (ambientalismo). In questo caso possiamo vedere amplificato l'effetto della popolarità, infatti con la misura non normalizzata la parola si presenta solo in 13ma posizione ed emergono invece gli ambiti più comuni in cui il termine viene usato:

misura normalizzata.

environmentalism, earthquake, Environment, asia, eco, green, change, energy, future, disaster

misura non normalizzata.

*future, green, blog, blogs, politics, environment, technology, change, community, world, (news, culture, **environmentalism**)*

Questi sono appunto esperimenti qualitativi, ma molto utili per comprendere il funzionamento delle varie misure di similarità ed interpretare i risultati quantitativi presentati nei prossimi capitoli.

4.3 Misure trasversali

In modo analogo alle misure dirette è possibile definire tre misure trasversali, tra entità diverse confrontate in uno spazio comune. Avendo tre possibili spazi è possibile definire:

1. una misura tra utenti e href nello spazio dei tag,
2. una misura tra tag ed href nello spazio degli utenti,
3. una misura tra tag ed utenti nello spazio degli href.

Anche in questo caso, solo la misura tra href ed utenti è stata usata con successo in questo lavoro, le altre due misure, anche se hanno interpretazioni plausibili, non sono state studiate a fondo.

Per abuso di linguaggio viene definito kernel una funzione asimmetrica operante su spazi distinti, ma è possibile definire un kernel valido semplicemente operando sulle due matrici concatenate.

Definizione 4.5. *Si definisce kernel tra user e href nello spazio dei tag:*

$$K_{uTh}(u, h) = \sum_{t \in \text{tags}} \mathcal{H}_{(u,t)} \cdot \mathcal{U}_{(h,t)} = \langle \mathcal{H}_{(u,\cdot)}, \mathcal{U}_{(h,\cdot)} \rangle$$

Questa misura interpreta un utente come un meta-documento e quindi permette di confrontare direttamente gli interessi di un utente con gli argomenti degli href in archivio.

Definizione 4.6. *Si definisce kernel tra href e tag nello spazio degli user:*

$$K_{hUt}(h, t) = \sum_{u \in \text{users}} \mathcal{T}_{(u,h)} \cdot \mathcal{H}_{(u,t)} = \langle \mathcal{T}_{(\cdot,h)}, \mathcal{H}_{(\cdot,t)} \rangle$$

Questa misura dovrebbe suggerire i migliori tag per un dato href e i migliori href per un dato tag, ma non è stata implementata.

Definizione 4.7. *Si definisce kernel tra user e tag nello spazio degli href:*

$$K_{uHt}(u, t) = \sum_{h \in \text{hrefs}} \mathcal{T}_{(u,h)} \cdot \mathcal{U}_{(h,t)} = \langle \mathcal{T}_{(u,\cdot)}, \mathcal{U}_{(\cdot,t)} \rangle$$

Questa misura dovrebbe suggerire i tag più rappresentativi per un dato user e gli user più competenti per un dato tag, ma non è stata implementata.

Come nella sezione precedente, per ogni misura non normalizzata è possibile definire una analoga misura normalizzata.

Nota 4.3. Per completezza mostriamo che è possibile definire un kernel che operi in un singolo spazio unione di due matrici:

Siano $\mathcal{U} \in \mathbb{N}^{|\text{hrefs}| \times |\text{tags}|}$ e $\mathcal{H} \in \mathbb{N}^{|\text{users}| \times |\text{tags}|}$ le matrici delle caratteristiche di utenti ed href.

Le due matrici hanno lo stesso numero di colonne, pertanto è possibile definire la matrice:

$$\mathcal{UH} \in \mathbb{N}^{(|\text{users}| + |\text{hrefs}|) \times |\text{tags}|}$$

Su questa matrice è possibile definire il kernel:

$$K_{\mathcal{T}}(x, y) = \sum_{t \in \text{tags}} \mathcal{UH}_{(x,t)} \cdot \mathcal{UH}_{(y,t)} = \langle \mathcal{UH}_{(x,\cdot)}, \mathcal{UH}_{(y,\cdot)} \rangle$$

operante nello spazio dei tag.

In modo analogo è possibile definire i kernel operanti nello spazio degli href e nello spazio degli utenti.

Al momento dell'implementazione è risultata preferibile l'opzione delle funzioni specializzate, e la formalizzazione tende a ricalcare dove possibile le scelte implementative.

Capitolo 5

Algoritmi di suggerimento

Una volta definite le misure di similarità tra elementi è possibile definire delle funzioni di ordinamento che in base a diverse ipotesi scelgono i migliori elementi candidati per il suggerimento.

Nell'esperimento vengono confrontate varie predizioni, alcune impersonali, cioè l'insieme suggerito è stabile tra utente ed utente, ed altre personali, cioè predizioni in cui l'insieme suggerito varia tra utenti diversi.

5.1 Ordinamenti impersonali

Le misure impersonali implementate ed usate nei test sono essenzialmente due: la prima è il suggerimento casuale, che rappresenta il limite inferiore della qualità di previsione, la seconda è l'ordinamento basato sulla popolarità della risorsa suggerita.

Nel caso degli href la popolarità può essere calcolata con un semplice loop che incrementa il punteggio di un href di una costante ogni volta che questo è stato archiviato da un utente, indipendentemente dal numero di tag usati:

Popolarità basata sulla distanza nello spazio dei tag

```
popularity = zeros(|hrefs|)
for u in users:
    for h in hrefs:
        if  $\mathcal{T}_{(u,h)} > 0$ :
            popularity[h] += 1.0
```

La popolarità è un fattore che determina una maggiore visibilità della risorsa e quindi una più facile diffusione tra gli utenti. Vedremo infatti come sia difficile avere delle prestazioni in previsione che si distacchino in modo sensibile dalla previsione basata sulla popolarità.

In modo analogo è possibile definire la popolarità per tag:

Popolarità basata sulla distanza nello spazio degli href

```
popularity = zeros(|tags|)
for u in users:
    for t in hrefs:
        if  $\mathcal{H}_{(u,t)} > 0$ :
            popularity[t] += 1.0
```

Accanto alla popolarità è possibile definire anche la diffusione del tag, cioè il numero di volte che il tag è stato usato per definire un href:

```

Diffusione dei tag
diffusion = zeros(|tags|)
for u in users:
    for t in hrefs:
        diffusion[t] +=  $\mathcal{H}_{(u,t)}$ 

```

5.2 Ordinamenti basati sugli utenti

Osservando la struttura di `del.icio.us`, ed in particolare la prima pagina del sito che aggiorna quasi in tempo reale i siti popolari del momento, è facile intuire che la probabilità che un elemento passato in prima pagina venga salvato da qualche utente è piuttosto elevata. Gli ordinamenti basati sugli utenti sfruttano questa caratteristica dei dati e cercano di interpretare in modo diverso le opinioni di persone diverse.

Il comportamento dell'algoritmo non è dissimile da ciò che facciamo ad esempio per scegliere quale film andare a vedere al cinema. Possiamo chiedere opinioni a vari amici che hanno visto il film, ma valuteremo in modo diverso l'opinione dei vari amici in base alle esperienze passate di accordo o disaccordo in altri giudizi cinematografici.

L'algoritmo, grazie alla rete di relazioni presente nella folksonomia, valuta in modo empirico la similarità tra utenti e cerca di compiere lo stesso tipo di operazione.

Date le relazioni tra utenti è possibile definire tre ordinamenti che si ispirano all'algoritmo di suggerimento usato in `GroupLens` [20]. L'ipotesi è di suggerire ad un utente ciò che in passato è risultato interessante ad utenti a lui simili, senza considerare le informazioni (eventualmente note) relative all'oggetto suggerito.

In verità le informazioni relative all'oggetto suggerito sono in effetti implicite nella misura di similarità tra utenti, questo perché in tutte le misure interne ad una folksonomia sono sempre coinvolte tutte le tre entità esistenti.

Possiamo quindi modificare l'algoritmo che calcola la popolarità di un elemento e sostituire alla costante additiva una variabile, in modo da pesare diversamente le opinioni di utenti più o meno simili.

La similarità tra utenti può essere calcolata con uno dei due kernel tra utenti presentati nella sezione 4.2. Sperimentalmente ed intuitivamente ha più senso usare il kernel normalizzato che dà lo stesso peso a tutti gli utenti (senza premiare gli utenti molto attivi come farebbe il kernel non normalizzato).

È possibile definire almeno tre misure, la prima che calcola la similarità nello spazio dei tag, la seconda che opera nello spazio degli href ed una terza che media le due misure.

In realtà anche la scelta del tipo di media da usare (aritmetica, geometrica o armonica ad esempio), o la scelta di un iperparametro che pesi in modo diverso le due similarità (ad esempio $0.2 \cdot \bar{K}_{\mathcal{T}}(u, u') + 0.8 \cdot \bar{K}_{\mathcal{H}}(u, u')$) richiederebbe uno studio più approfondito che non è stato affrontato in questa fase dei lavori.

Negli esperimenti fatti la media armonica ha condotto a prestazioni soddisfacenti, ed è quindi stata scelta senza dedicare ulteriore tempo alla ricerca di una soluzione ottimale.

Vediamo l'algoritmo che ordina gli href usando la misura di similarità tra utenti nello spazio dei tag:

```

Popolarità tra utenti, similarità nello spazio dei tag
u' ∈ users
popularity_on_user_tag_similarity = zeros(|hrefs|)
for u in users:
    for h in hrefs:
        if  $\mathcal{T}_{(u,h)} > 0$ :
            popularity_on_user_tag_similarity[h] +=  $\bar{K}_{\mathcal{T}u}(u, u')$ 

```

In questa ottica risultano simili utenti che hanno usato tag simili e che hanno un numero di href per ogni tag simile, cioè si suppone un'analogia esperienza nell'argomento definito dal tag.

In modo complementare è possibile definire l'algoritmo operante nello spazio degli href:

```

Popolarità tra utenti, similarità nello spazio degli href
u' ∈ users
popularity_on_user_href_similarity = zeros(|hrefs|)
for u in users:
    for h in hrefs:
        if  $\mathcal{T}_{(u,h)} > 0$ :
            popularity_on_user_href_similarity[h] +=  $\bar{K}_{u\mathcal{H}u}(u, u')$ 

```

In questo caso sono considerati simili gli utenti che hanno in archivio un insieme di href simile e che usano i tag con lo stesso stile, cioè usano un numero simile di tag per definire gli href inseriti. Questo potrebbe essere un vincolo eccessivo, le risorse potrebbero essere valutate in modo costante ed indipendente dal numero di tag usati, ma questa estensione non è stata implementata, e quindi non si hanno risultati sperimentali per mettere a confronto le due opzioni.

Definito l'operatore media armonica (nel caso di soli due valori) come:

$$\text{hmean}: \mathbb{R}^2 \rightarrow \mathbb{R}, \text{hmean}(x, y) = \frac{2 \cdot x \cdot y}{x + y}$$

possiamo definire l'algoritmo che media le due misure:

```

Popolarità tra utenti, similarità media
u' ∈ users
popularity_on_user_taghref = zeros(|hrefs|)
for u in users:
    for h in hrefs:
        if T(u,h) > 0:
            popularity_on_user_taghref[h] += hmean(K_uT_u(u, u'), K_uH_u(u, u'))

```

Questo ordinamento è il più configurabile, infatti sarebbe possibile pesare in modo diverso le due misure di similarità tra utenti, e scegliere con un parametro nella visualizzazione dei risultati se essere più tag-centrici o href-centrici. Queste opzioni non sono state sperimentate e richiederebbero un test dal vivo per valutare le impressioni degli utenti.

Un altro aspetto che necessiterebbe di essere approfondito è legato al comportamento del kernel. Se due utenti risultano ortogonali nello spazio indotto dal kernel, cioè se non hanno tag o href in comune, la loro similarità risulta nulla. Questo per alcuni utenti che hanno pochi tag o pochi href potrebbe limitare in modo significativo il numero di opinioni ricevute. Sarebbe ragionevole cercare un limite inferiore non nullo alla misura di similarità che permetta di mantenere prestazioni accettabili anche nel caso di utenti che usano il sistema da poco tempo. Ad esempio questa modifica porta a prestazioni leggermente maggiori nel caso della similarità calcolata nello spazio dei tag.

L'algoritmo implementato è simile a quello presentato precedentemente ma con una piccola aggiunta:

```

Popolarità tra utenti, similarità nello spazio dei tag corretta
u' ∈ users
popularity_on_user_tag_similarity = zeros(|hrefs|)
for u in users:
    for h in hrefs:
        if T(u,h) > 0:
            popularity_on_user_tag_similarity[h] += K_T(u, u') or mean(K_T(·, u'))

```

Dove l'operatore `or` opera come definito nel linguaggio Python, cioè viene restituito il primo valore se diverso da zero, altrimenti viene restituito il secondo.

Da notare che lo pseudocodice presentato è stato de-ottimizzato per fini esplicativi. Ovviamente è inutile iterare su tutti gli href ma è sufficiente iterare solo su quelli salvati dall'utente attuale nel loop.

5.3 Ordinamento basato sulla misura trasversale

Anche se sono state definite tre misure trasversali, ai fini dell’esperimento, cioè volendo predire gli href che l’utente inserirà in archivio nel futuro, vi è una unica misura applicabile direttamente.

Il kernel $K_{u\mathcal{T}h}(u, h)$ tra utenti ed href restituisce direttamente una misura di similarità tra un utente e un href. Applicando la misura a tutti gli href, abbiamo direttamente l’ordinamento per similarità che cercavamo:

Similarità trasversale tra utenti ed href
<pre> u ∈ users user_href_similarity = zeros(hrefs) for h in hrefs: user_href_similarity[h] = K_{uTh}(u, h) </pre>

Questa misura è concettualmente azzardata ma piuttosto efficace. Permette infatti di considerare l’utente come un meta-documento summa di tutti i documenti da lui classificati.

Intuitivamente, è ragionevole che un utente che usa molto il tag “storia” risulti simile ad un sito web che molti altri utenti ritengono trattare l’argomento “storia”.

5.4 Algoritmi di espansione semantica

Abbiamo visto che è possibile operare all’interno di una folksonomia per avere dei suggerimenti su termini che hanno significato analogo. Abbiamo anche visto che operando in modo generale non è possibile risolvere la polisemia dei termini se non nel senso maggiormente diffuso.

Ad esempio il tag “tiger” oggi viene interpretato come la versione omonima del sistema operativo Apple. Questo ad esempio è l’elenco dei primi 10 termini simili a “tiger” secondo una delle misure implementate:

- *tiger, osx, mac, dashboard, spotlight, automator, software, apple, OSX, macosx, howto, tips, widgets.*

Anni fa probabilmente la parola sarebbe stata interpretata come il felino predatore “tigre”.

In un’ottica globale (computer centrica ovviamente), il termine “tiger” ha subito una profonda mutazione semantica, ed in pratica a rimetterci dal punto di vista dell’accesso alle informazioni sono stati coloro che prima della mutazione usavano il tag “tiger” riferendosi all’animale, ed adesso vedono comparire solo pagine che parlano di computer.

Purtroppo nel limitato dataset di esempio non ci sono utenti che interpretano “tiger” nel senso zoologico, e quindi non è possibile fare un esempio su questo termine.

È però possibile fare un esempio in un ambito ben più comune in *delicio.us*, la programmazione.

È evidente che anche il termine “programming” è molto ambiguo, essenzialmente non chiarisce se un utente è interessato ad un linguaggio di programmazione in particolare.

Infatti cercando i termini simili a “programming” senza specificare un particolare utente abbiamo:

- *programming, development, dev, code, cs, coding, Programming, reference, webdev, javascript*

Vediamo come si comporta il semplice algoritmo di disambiguazione sviluppato in questo lavoro con alcuni esempi qualitativi.

Ad esempio vediamo i tag simili a “programming” nella visione di due utenti diversi:

user 207. *programming, web, javascript, design, css, ajax, php, xml, reference, software*

user 208. *programming, news, linux, php, tech, mysql, development, apache, web, javascript*

L’algoritmo disambigua la parola dando un’interpretazione ben diversa e implicitamente suggerisce che il primo utente sia probabilmente un webdesigner e che il secondo sia più probabilmente un sistemista.

La disambiguazione non avviene esclusivamente all’interno dei tag dell’utente, infatti in questo caso non potrebbero essere suggeriti tag non usati dall’utente, ma avviene fondendo l’ordinamento proposto dalla funzione impersonale di similarità tra tag con un ordinamento dei tag derivato dalla popolarità analogo a quello presentato per gli href nella sezione 5.2.

Quella che segue è la lista dei 10 tag più popolari nella visione degli utenti 207 e 208 basata sulla popolarità pesata con la similarità nello spazio degli href tra utenti:

user 207. *web, css, design, javascript, php, ajax, search, rss, google, firefox, xml, music, programming*

user 208. *news, linux, php, tech, mysql, apache, geek, mozilla, macosx, bsd, oreilly, design, music*

La fusione avviene nell’attuale implementazione semplicemente facendo la media dei due vettori normalizzati.

Ovviamente anche in questo caso i possibili approcci sono molti, anche solo il fatto di usare una misura normalizzata o meno, o la scelta di usare come base la tabella originale oppure la tabella filtrata con TF-IDF conduce a risultati sensibilmente diversi. Dato il numero eccessivo di parametri e la difficile valutazione della qualità, è evidente che questo ambito necessiterebbe di uno studio più approfondito.

Capitolo 6

Valutare la qualità della predizione

Il problema di valutare le prestazioni di un sistema di suggerimento collettivo è relativamente nuovo, esistono molte misure che in qualche modo si affiancano alla storica misura di Precisione e Recupero, ma per ora nessuna è stata eletta come misura di confronto standard.

Questo spesso non è dovuto alle sole manie di protagonismo dei ricercatori ma bensì a difficoltà effettive. Spesso infatti le ricerche sono fatte su basi di dati che poi non vengono (o non possono essere) rese disponibili, quindi l'esperimento non sarebbe replicabile anche se la misura usata fosse stata standard. Altre volte invece l'obiettivo delle ricerca è quello di evidenziare alcuni particolari aspetti della qualità della previsione che non vengono messi in evidenza dalle misure già note.

Per questa seconda ragione in questo lavoro ho scelto di affiancare alla misura di prestazione classica Precisione e Recupero [?], due nuove misure. Una misura di similarità che opera come una Precisione “morbida”, che considera gli errori meno gravi se l'elemento suggerito è simile ad un elemento corretto, ed un Recupero modificato che valuta in modo positivo la presenza di risorse poco note.

6.1 Precisione e Recupero

Sia *prevision* l'insieme degli href suggeriti e sia *to_predict* l'insieme degli href scelti nel futuro dall'utente.

Definizione 6.1. *Si definisce Precisione:*

$$\text{Precisione} = \frac{|\text{prevision} \cap \text{to_predict}|}{|\text{prevision}|}$$

il rapporto tra il numero elementi corretti suggeriti e il numero di elementi suggeriti.

Definizione 6.2. *Si definisce Recupero:*

$$\text{Recupero} = \frac{|\text{prevision} \cap \text{to_predict}|}{|\text{to_predict}|}$$

il rapporto tra il numero di elementi corretti suggeriti ed il numero di elementi che potevano essere predetti.

Si può intuire che le due misure sono tipicamente legate da una relazione di proporzionalità inversa. Infatti suggerendo un gran numero di elementi è facile avere un buon Recupero, ma al prezzo di una bassa Precisione. In modo analogo anche un ottimo algoritmo di predizione che ha Precisione massima, avrà un basso Recupero fino a quando l'insieme predetto avrà cardinalità inferiore all'insieme degli elementi predicibili.

Spesso è utile valutare le misure di Precisione e Recupero al variare dell'orizzonte di predizione, per questo è stata definita una misura combinata che sintetizzi in un solo valore le prestazioni della previsione.

Definizione 6.3. *Si definisce F1 la media armonica tra Precisione e Recupero:*

$$F1 = \text{hmean}(P, R) = \frac{2 \cdot P \cdot R}{P + R}$$

6.2 Dalla Precisione alla Similarità

La previsione in un dataset complesso come una folksonomia non è semplice, le prestazioni in Precisione e Recupero sono molto basse. Per questo è nata la necessità di trovare fattori che potessero discriminare meglio la qualità delle previsioni fatte. Ad esempio disponendo di grandi risorse sarebbe possibile effettuare dei test dal vivo e chiedere a degli utenti di valutare i link proposti. Poiché le risorse necessarie a svolgere un esperimento del genere esulano delle possibilità di un lavoro di tesi, in questo esperimento è stata scelta la misura di Similarità.

Sia `prevision` l'insieme degli href suggeriti e sia `to_predict` l'insieme degli href scelti nel futuro dall'utente.

Sia $\bar{K}_T^f(h, h')$ il kernel normalizzato tra href nello spazio dei tag basato sui dati futuri.

La Similarità valuta ogni elemento predetto $p \in \text{prevision}$ con il massimo della funzione kernel tra l'elemento p e tutti i possibili elementi da predire in `to_predict`. Dividendo per il numero di elementi predetti abbiamo una funzione che corrisponde alla Precisione quando gli elementi predetti non corretti sono tutti ortogonali a quelli corretti (kernel nullo). Negli altri casi la Similarità è più "accomodante" della Precisione valutando gli errori con un valore $k \in (0, 1)$ evidentemente non nullo.

Possiamo riassumere che la Similarità estende la Precisione ammettendo un criterio di valutazione della gravità dell'errore.

6.3 Dal Recupero alla RevCall

Un altro aspetto della qualità di predizione è la quantità di predizioni corrette realizzate, considerata indipendentemente dal numero di tentativi effettuati, misurata dal Recupero. Dobbiamo osservare che anche a parità di prestazioni in Recupero, due suggerimenti ugualmente corretti potrebbero risultare molto diversi nell'ottica dell'utente.

Esempio banale potrebbe essere un sistema di suggerimento per supermercati. Anche l'algoritmo più banale che suggerisce gli oggetti più comprati, come "pane" e "latte" potrebbe avere prestazioni discrete, ma risultare inutile all'utente.

Un fattore che sta sempre più affermandosi come indice di qualità di un sistema di suggerimento è la capacità di suggerire elementi adatti all'utente anche se poco popolari tra tutti gli utenti. I fattori che vengono considerati sono la "novelty" e la "serendipity" dei suggerimenti, cioè un misto di novità ed inaspettatezza che si ritiene interessare positivamente l'utente.

Osservando le statistiche di uso dei motori di ricerca [15]'06, possiamo vedere come il 60% degli intervistati si fermi alla prima pagina dei risultati. Questo quando l'utente sta cercando attivamente qualcosa. In questo caso, quando l'utente aspetta passivamente di vedere notizie interessanti, si può supporre che la legge del minimo sforzo sia ancora più inclemente.

Pertanto la misura che propongo si basa su due ipotesi:

- è ragionevole interpretare la popolarità di un sito in `del.icio.us` come campione della popolarità dei siti web in internet,
- è ragionevole quindi interpretare la popolarità in `del.icio.us` proporzionale alla probabilità che l'utente acceda al sito navigando in rete.

Con queste ipotesi e noti i comportamenti degli utenti nella ricerca, è ragionevole valutare in modo molto più significativo il fatto di avere suggerito correttamente un sito poco noto invece di un sito molto diffuso.

La misura `RevCall` è una misura normalizzata che risulta pari a 0 quando la predizione è errata e pari ad 1 quando gli N elementi predetti sono gli N elementi meno popolari scelti dall'utente nell'insieme di test. L'importanza del riordino decresce in modo logaritmico con la distanza dalla prima pagina.

Sia `pophrefs` l'insieme ordinato per popolarità decrescente di tutti gli href.

Sia `prevision` l'insieme degli href suggeriti e sia `to_predict` l'insieme degli href scelti nel futuro dall'utente.

Definizione 6.4. *Si definisce `MaxPopSum` il numero risultante dalla somma degli indici in `pophrefs` degli elementi di `to_predict`.*

Definizione 6.5. *Si definisce `TruePopSum` il numero risultante dalla somma degli indici in `pophrefs` degli elementi di `prevision` \cap `to_predict`.*

Definizione 6.6. *Si definisce `RevCall` il rapporto:*

$$RevCall = \frac{\log(TruePopSum)}{\log(MaxPopSum)}$$

Possiamo riassumere che la `Revcall` estende la `Recall` inserendo un criterio di preferenza tra gli elementi suggeribili.

Capitolo 7

Esperimenti e risultati

I dati per l'esperimento sono stati selezionati secondo due criteri:

1. un criterio temporale che limita gli elementi inclusi a quelli attivi da almeno tre mesi prima, e ancora attivi per almeno tre mesi dopo la data del 01/06/2005 presa come linea di demarcazione tra passato e futuro,
2. un criterio di attività che limita la selezione agli elementi che hanno in archivio almeno n elementi di supporto negli altri due spazi. Cioè utenti con almeno n href ed n tag, indirizzi con almeno n utenti ed n tag, tag con almeno n utenti ed n href.

Gli esperimenti sono stati condotti per $n \in [4, 6]$, l'archivio è stato diviso in due parti, una parte di addestramento che contiene tutta l'attività avvenuta fino al giorno limite ed una parte di test che contiene tutta l'attività avvenuta oltre il giorno limite.

Il test consiste nel suggerire q href ad ogni utente e verificare quanti di questi q href sono stati effettivamente inseriti in archivio dall'utente in futuro.

I grafici presentano la misura F1 per Precisione e Recupero e la misura F1 per Similarità e RevCall su un orizzonte di predizione di dimensione q , con q elemento di uno spazio logaritmico di 20 elementi compresi tra 10^1 e 10^2 .

7.1 Selezione dei dati con $n = 6$

La selezione dei dati con $n = 6$ limita i dati ad un insieme di dati piuttosto diffusi, e rimuove tutti gli elementi che non rispettano il criterio di attività. In pratica stiamo operando nella parte alta della curva della popolarità. Con questi vincoli l'archivio è ridotto a:

- 502 utenti
- 5732 href
- 3813 tag

Osserviamo che gli ordinamenti intelligenti superano quasi sempre le prestazioni dell'ordinamento per popolarità, ed in particolare possiamo notare che l'ordinamento basato sulla misura di similarità nello spazio dei tag, *utu*, ha prestazioni leggermente peggiori degli altri due algoritmi.

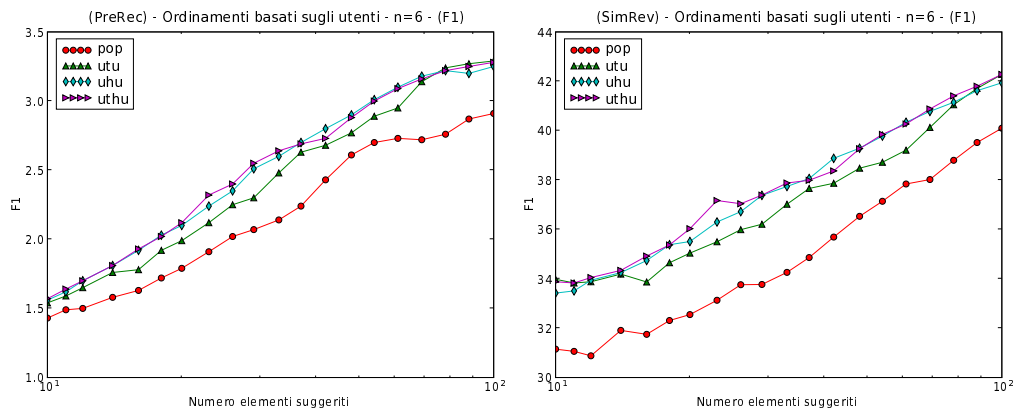


Figura 7.1. F1 per i tre algoritmi di suggerimento basati sugli utenti.

Questo è probabilmente effetto della diversa cardinalità degli insiemi di utenti “vicini” indotti dalla due misure. Infatti i tag sono molto più caratterizzati dallo sfondo culturale dell'utente (ad esempio tag in lingua italiana creano un intorno di utenti probabilmente di lingua italiana), mentre gli href sono più trasversali, è frequente che persone di lingue e culture diverse abbiano in archivio uno stesso sito in lingua inglese.

Da notare come le prestazioni sui primi 10 elementi vedano invertite le misure *utu* ed *uhu*. Probabilmente la migliore misura tra le misure basate sugli utenti è la misura mediata *uthu*.

Seconda osservazione, la misura F1 di Similarità e RevCall ($F1(S, R)$) mantiene sostanzialmente l'ordinamento del giudizio di qualità ma premia in modo più marcato le misure personalizzate.

Analizziamo anche separatamente le prestazioni in Precisione e Recupero ed in Similarità e RevCall.

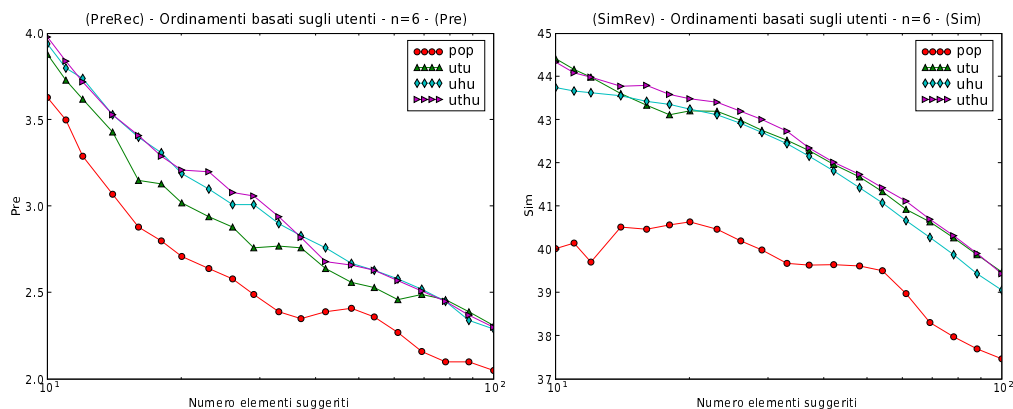


Figura 7.2. Precisione e Similarità per i tre algoritmi di suggerimento basati sugli utenti.

Possiamo osservare come la Similarità penalizzi in modo particolarmente visibile i primi suggerimenti basati sulla popolarità, suggerimenti che probabilmente non sono molto coerenti con quelli che sono gli interessi dell'utente.

Da osservare che mentre la misura di Similarità è basata direttamente sul confronto nello spazio dei tag, l'ordinamento che genera i suggerimenti è basato sull'opinione cumulata degli utenti, opinione che evidentemente contiene in modo implicito informazioni sugli argomenti trattati.

Se confrontiamo separatamente le prestazioni di Recupero e RevCall possiamo notare che il distacco in Recupero tende a crescere leggermente mentre il distacco in RevCall tende a diminuire, sintomi di una buona capacità sia di promuovere elementi improbabili ai primi posti dell'ordinamento sia in grado di mantenere prestazioni assolute migliori.

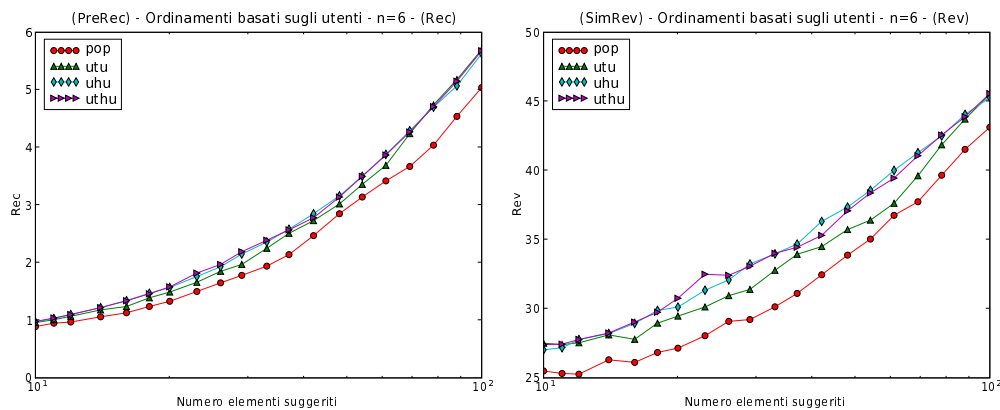


Figura 7.3. Recupero e RevCall per i tre algoritmi di suggerimento basati sugli utenti.

Se osserviamo invece il comportamento della qualità di previsione per gli algoritmi basati sulla misura trasversale tra user e href, osserviamo che l'applicazione della normalizzazione e del filtro TF-IDF ha effetti molto particolari. Infatti le prestazioni iniziali sono sensibilmente peggiori delle prestazioni dell'ordinamento basato sulla popolarità, ma oltre un certo valore dell'orizzonte di previsione le prestazioni salgono e si avvicinano alle prestazioni originate dalla popolarità.

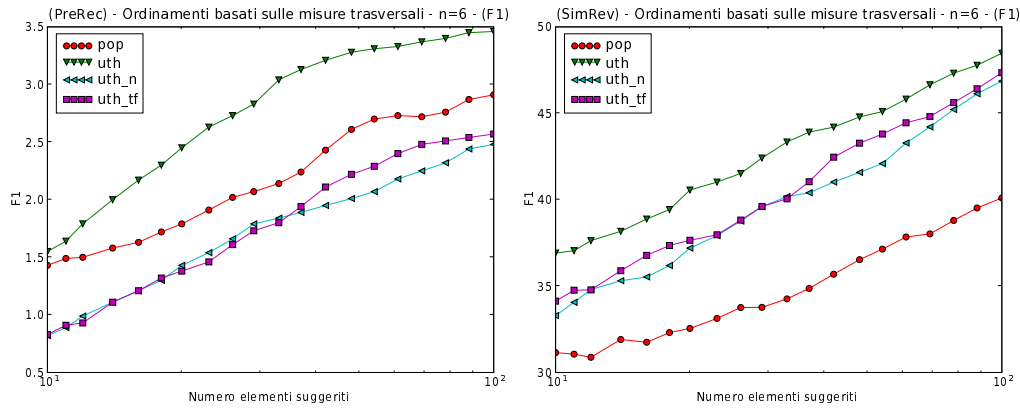


Figura 7.4. F1 per i tre algoritmi di suggerimento basati sulle misure trasversali.

Questo effetto è probabilmente dovuto alla proprietà del filtro TF-IDF di caratterizzare i documenti per gli attributi meno diffusi. Quindi i primi elementi suggeriti sono elementi caratterizzati da attributi molto poco comuni ed è quindi molto improbabile che molti utenti li abbiano in archivio. Esauriti gli strani suggerimenti iniziali, l'effetto moderatore del filtro TF-IDF sulla popolarità perde peso e la differenza tra le prestazioni diminuisce. Situazione analoga si presenta con l'uso del kernel normalizzato.

È interessante notare come il giudizio delle due misure di qualità sia sensibilmente diverso, infatti con $F1(P, R)$ le misure personalizzate hanno prestazioni peggiori della misura impersonale, mentre con $F1(S, R)$ tutte le misure personalizzate hanno prestazioni migliori.

Se confrontiamo separatamente le prestazioni di Precisione e Similarità possiamo notare che buona parte delle ragioni dello scambio in F1 sono proprio basate sulla coerenza degli argomenti proposti con gli interessi dell'utente valutati dalla Similarità.

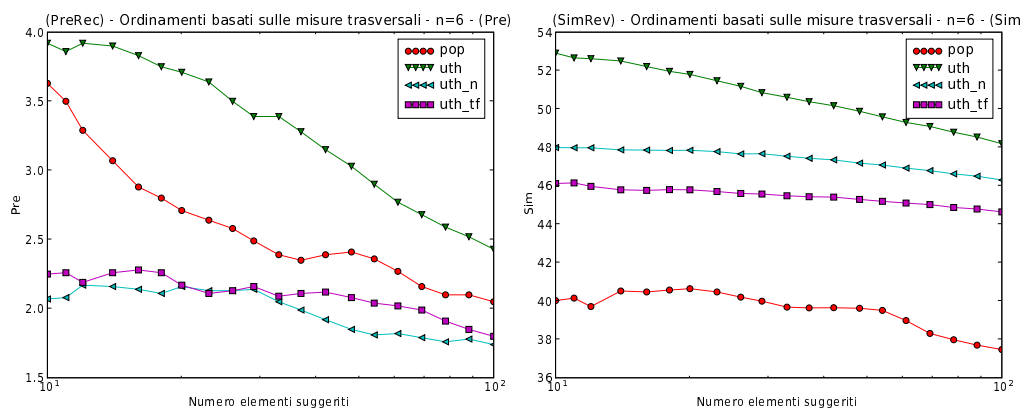


Figura 7.5. Precisione e Similarità per i tre algoritmi di suggerimento basati sulle misure trasversali.

Questo probabilmente induce una distorsione che premia in modo eccessivo gli ordinamenti personalizzati. Fortunatamente la misura uth ha prestazioni migliori anche in termini assoluti di Precisione e Recupero, situazione che lascia aperta la strada ad uno studio più approfondito sulla qualità percepita, ed in particolare su quale ordinamento risulti in effetti più gradevole ad un utente.

Possiamo anche visualizzare separatamente le prestazioni in Recupero e RevCall. Anche in questo caso vi è un'inversione nel giudizio di qualità rispetto all'ordinamento per popolarità, che risulta penalizzato dalla RevCall. Altra caratteristica evidenziata dal grafico è il tasso di crescita della qualità dell'ordinamento che usa la TF-IDF che è più alto rispetto agli altri ordinamenti (in modo più evidente usando la RevCall).

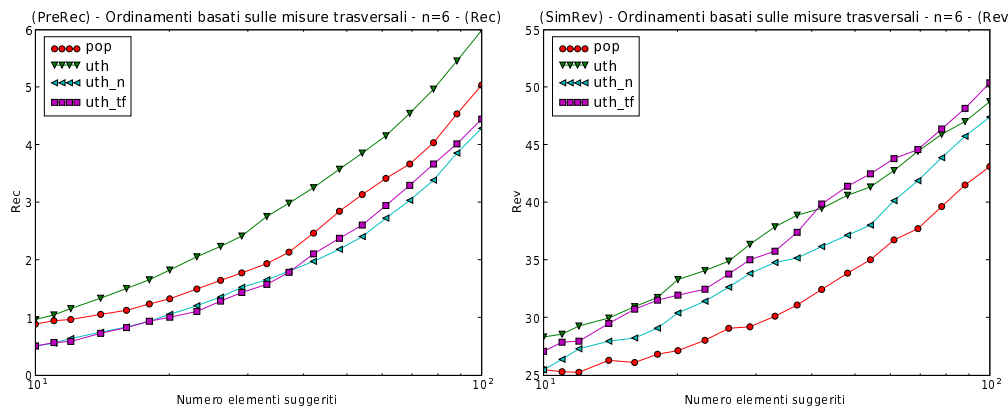


Figura 7.6. Recupero e RevCall per i tre algoritmi di suggerimento basati sulle misure trasversali.

Per tentare di comprendere qualitativamente la differenza tra i vari ordinamenti, presentiamo i primi 10 suggerimenti per gli utenti 207 e 208 già citati nella sezione 5.4.

Usando l'ordinamento per popolarità viene suggerito correttamente un solo href per l'utente 208:

- developer.apple.com/internet/webcontent/xmlhttpreq.html

Considerando invece gli ordinamenti personalizzati abbiamo questa situazione (in grassetto i suggerimenti corretti):

utente 207. Ordinamento uth:

1. pro.html.it/esempio/nifty/
2. www.adaptivepath.com/publications/essays/archives/000385.php
3. developer.apple.com/internet/webcontent/xmlhttpreq.html

4. www.webpasties.com/xmlHttpRequest/
5. tool-man.org/examples/edit-in-place.html
6. www.walterzorn.com/dragdrop/dragdrop_e.htm
7. www.bobbyvandersluis.com/articles/goodpractices.php
8. jpspan.sourceforge.net/wiki/doku.php?id=javascript+xmlhttprequest
9. www.xml.com/pub/a/2005/02/09/xml-http-request.html
10. leftjustified.net/site-in-an-hour/

utente 208. Ordinamento uth (nessun suggerimento corretto):

1. www.modernmethod.com/sajax/
2. www.marumushi.com/apps/newsmap/newsmap.cfm
3. www.getlucky.net/freetag/
4. www.engadget.com/
5. news.google.com/
6. www.theregister.co.uk/
7. www.gizmodo.com/
8. sputnik.pl/cake/
9. www.wired.com/
10. firepages.com.au/php_usb.htm

Le prestazioni assolute dell'ordinamento uthu sono ancora peggiori, però la profonda differenza tra gli insiemi suggeriti apre una serie di interessanti domande sul possibile gradimento della lista da parte dell'utente.

utente 207. Ordinamento uthu (nessun suggerimento corretto):

1. www.adaptivepath.com/publications/essays/archives/000385.php
2. www.goodfonts.org/
3. www.webpasties.com/xmlHttpRequest/

4. pro.html.it/esempio/nifty/
5. pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html
6. jpspan.sourceforge.net/wiki/doku.php?id=javascript+xmlhttprequest
7. www.datsi.fi.upm.es/~frosal/docs/25mdq.html
8. www.tiddlywiki.com/
9. 37signals.com/papers/introtopatterns/
10. dietrich.ganx4.com/foxylicious/

utente 208. Ordinamento uthu (nessun suggerimento corretto):

1. www.adaptivepath.com/publications/essays/archives/000385.php
2. pchere.blogspot.com/2005/02/absolutely-delicious-complete-tool.html
3. www.theregister.co.uk/
4. news.google.com/
5. www.w3schools.com/
6. www.boingboing.net/
7. dietrich.ganx4.com/foxylicious/
8. www.gizmodo.com/
9. home.tampabay.rr.com/bmerkey/cheatsheet.htm
10. www.dnsstuff.com/

Non bisogna lasciarsi sconcertare dalla bassa Precisione delle previsioni, infatti osservando i siti proposti in particolare all'utente 208 è possibile che l'utente conosca e visiti regolarmente i siti proposti, così tanto regolarmente da non averli inseriti nei propri bookmark in del.icio.us.

7.2 Selezione dei dati con $n = 4$

Vediamo brevemente come si modificano le prestazioni nel caso in cui venga abbassato il limite di attività ad $n = 4$, criterio per cui ogni entità deve avere almeno 4 entità colle-

gate negli altri due spazi.

In questo caso le dimensioni dell'archivio divengono:

- 507 utenti, (+2 rispetto ad $n = 6$),
- 10764 href, (+5032 rispetto ad $n = 6$),
- 5293 tag, (+1480 rispetto ad $n = 6$).

Quindi, per un numero di utenti simile, il numero di href è quasi raddoppiato ed il numero di tag è cresciuto di circa un terzo.

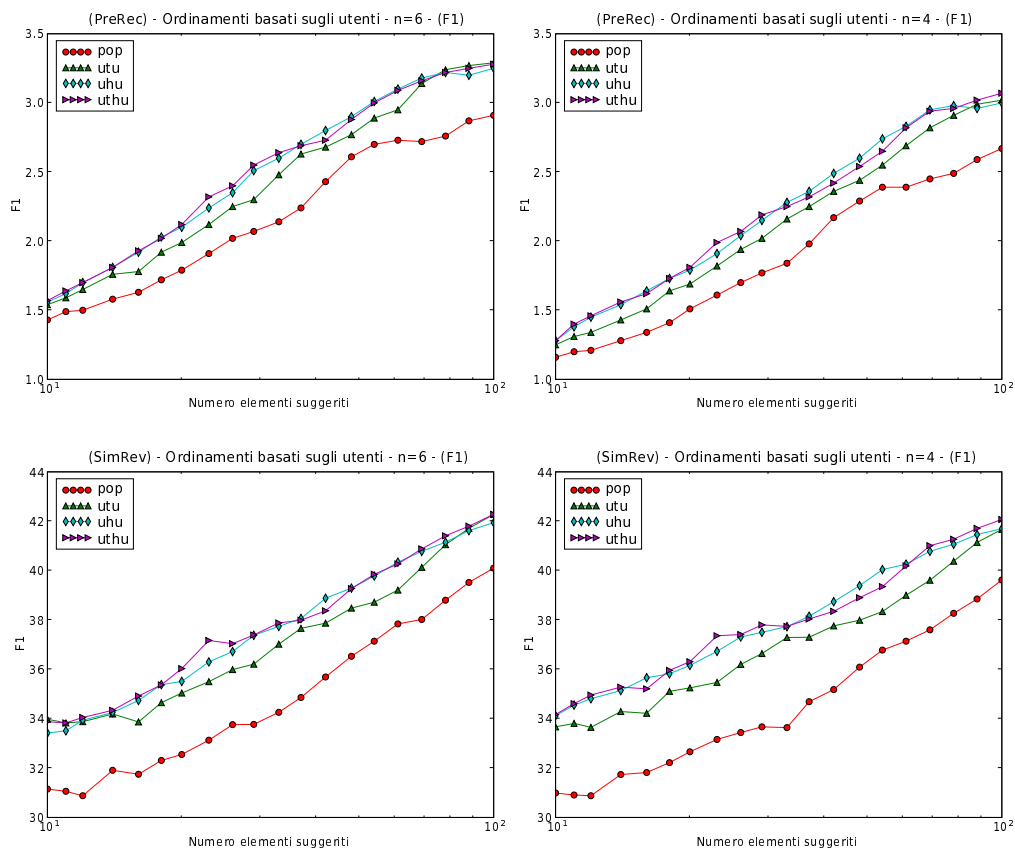


Figura 7.7. F1 per i tre algoritmi di suggerimento basati sugli utenti.

Il comportamento delle misure basate sugli utenti è sostanzialmente invariato, questo è ragionevole dato lo stretto legame con la popolarità. Interessante notare che le prestazioni assolute vengono penalizzate molto più delle prestazioni “morbide” al crescere della base di dati.

Nel caso degli ordinamenti basati sulle misure trasversali possiamo osservare come al crescere della dimensione della base di dati, il guadagno sull'ordinamento per popolarità, più evidente con $n = 6$, sia nei dati con $n = 4$ più limitato.

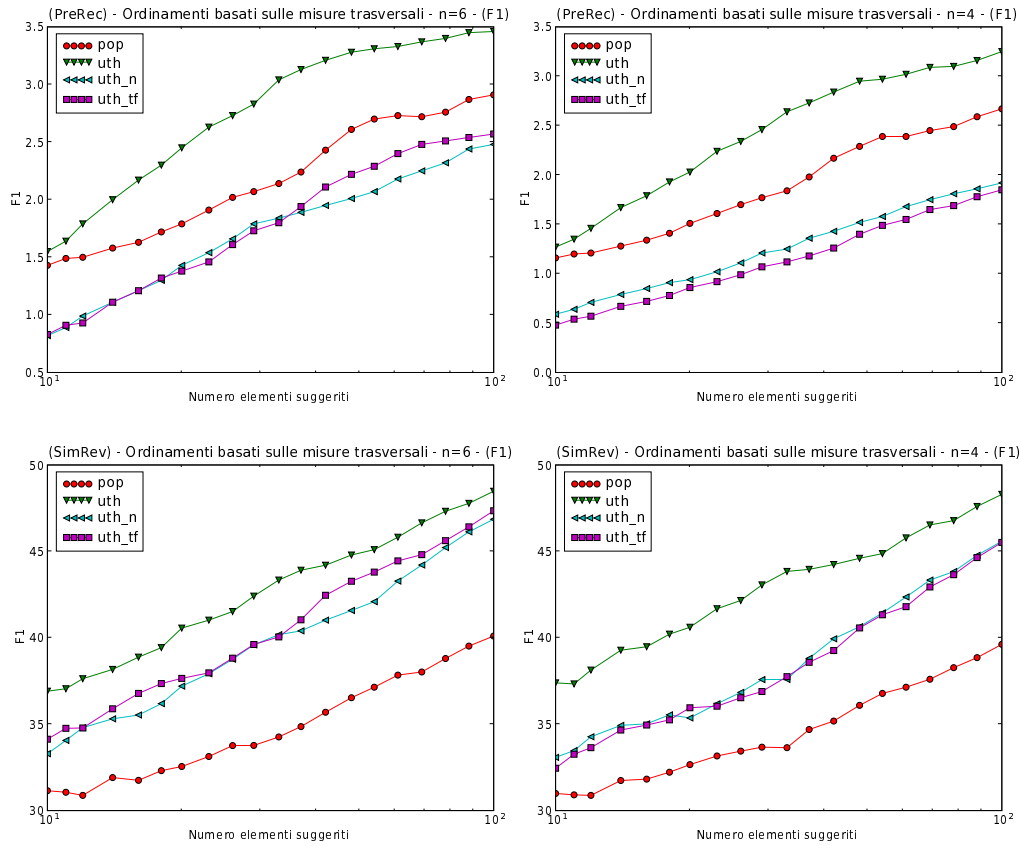


Figura 7.8. F1 per i tre algoritmi di suggerimento basati sulle misure trasversali.

7.3 Confronto tra i due migliori ordinamenti

Abbiamo visto che in modo abbastanza stabile anche al variare della dimensione dell'archivio, i due ordinamenti che mantengono le prestazioni migliori sono l'algoritmo basato sugli utenti uthu e l'algoritmo basato sulla misura trasversale uth.

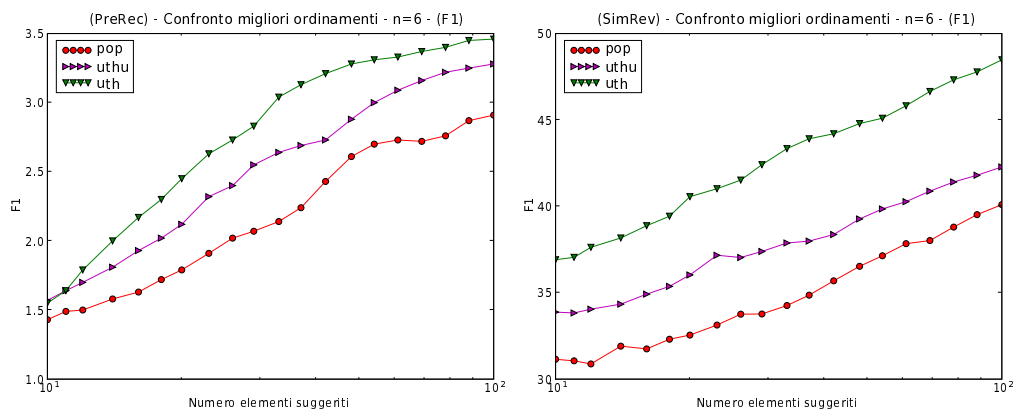


Figura 7.9. F1 per i due migliori algoritmi di predizione.

Confrontando le prestazioni possiamo osservare che nella fase iniziale della predizione a prestazioni assolute coincidenti corrispondono prestazioni morbide ben differenziate. Per meglio comprendere le ragioni di questa differenza vediamo i grafici delle misure di Precisione e di Similarità.

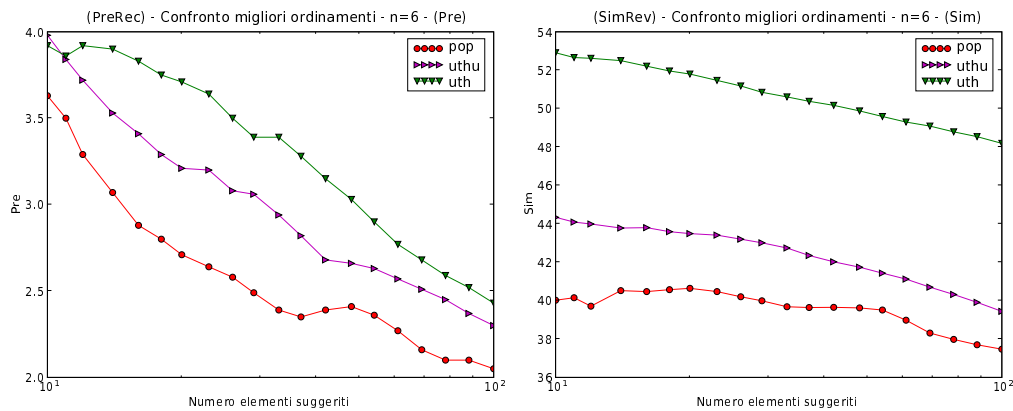


Figura 7.10. Precisione e Similarità per i due migliori algoritmi di predizione.

In questo caso è evidente che il guadagno in prestazioni della misura *uth* è principalmente dovuto ad un guadagno nella valutazione della similarità.

Va comunque detto le prestazioni di *uth* sono leggermente migliori anche secondo le misure di Recupero e RevCall.

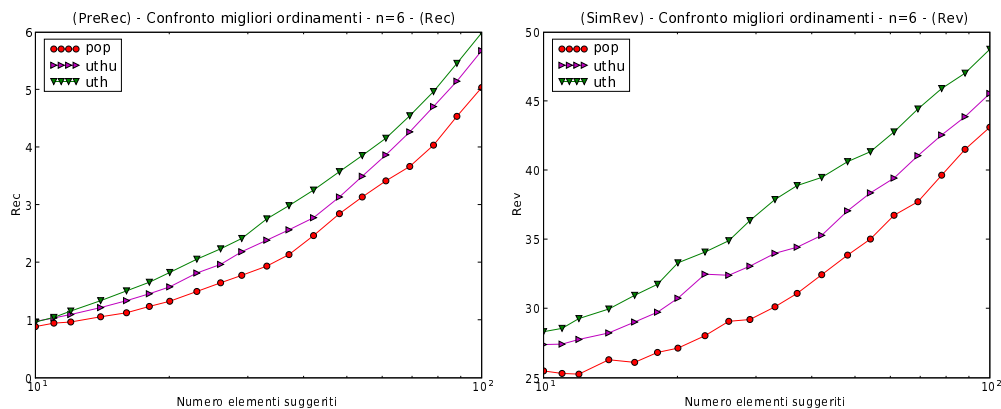


Figura 7.11. Recupero e RevCall per i due migliori algoritmi di predizione.

Quindi in assenza di test dal vivo possiamo concludere che le migliori prestazioni provengono dall'algoritmo più semplice che confronta direttamente l'utente con gli href.

Questa situazione ha implicazioni abbastanza complesse, infatti permette di affermare che utenti che usano tag simili sono anche interessati ad argomenti simili, indipendentemente dal rumore presente nella folksonomia dovuto ai "sinonimi" (cioè alle parole che differiscono per capitalizzazione o per numero).

7.4 Algoritmo di espansione semantica

L'ultimo algoritmo sviluppato opera espandendo in modo personalizzato i tag dell'utente. Ogni tag viene espanso in modo da includere anche i primi k tag vicini.

L'idea dell'algoritmo è quella di compensare la presenza di un tag poco diffuso con l'aggiunta di tag sinonimi più diffusi.

Le prestazioni in predizione di questo algoritmo non sono soddisfacenti e quindi non vengono neanche presentate.

Questo probabilmente è dovuto al grande numero di variabili in gioco, e si prospetta quindi necessario un test sistematico delle prestazioni al variare dei parametri. Completare un test di questo tipo avrebbe richiesto risorse sproporzionate rispetto agli obiettivi di questo studio. Resta da sottolineare che i risultati qualitativi presentati nella sezione 4.2.1 presentano caratteristiche semantiche di qualità non indifferente, e quindi per approfondire l'argomento si potrebbe prospettare necessaria la collaborazione di esperti nel campo della linguistica applicata.

Un problema che però rimane irrisolto nell'approccio all'espansione semantica è quello computazionale. Mentre gli altri algoritmi richiedono il calcolo della distanza tra un elemento e tutti gli altri elementi di uno spazio, l'algoritmo ad espansione semantica richiede che questa operazione venga compiuta per ogni tag dell'utente.

Resta quindi aperta la domanda sulla qualità di questa misura, e se la qualità fosse sensibilmente migliore di quella ottenibile con altri metodi, si aprirebbe un problema di prestazioni non indifferente.

Capitolo 8

Conclusioni e possibili sviluppi

Le folksonomie esistono da pochi anni ma sono oggi probabilmente la più ricca miniera di informazione presente sul web. Il grande vantaggio delle folksonomie rispetto ad altri approcci all'organizzazione del web è sicuramente la semplicità con cui queste strutture permettano ad un gran numero di utenti di cooperare.

Mentre il web semantico emette i primi vagiti, le folksonomie crescono, si espandono e si aggiornano molto rapidamente. Questo avviene senza un impegno particolare, ma grazie all'effetto combinato di tanti piccoli contributi.

Queste riflessioni mi hanno portato a pensare: possibile che le folksonomie racchiudano molta più informazione di quella che è visibile oggi?

Grazie ad una versatile interpretazione matriciale ed all'eredità di anni di evoluzione informatica, è bastata una intuizione perché le folksonomie cominciarono a mostrare i loro tesori nascosti.

La struttura che è stata presentata in questo lavoro ha delle gradevoli caratteristiche di simmetria che permettono di usare metodi ed algoritmi analoghi per trattare elementi diversi.

Abbiamo visto come sia possibile operare in una folksonomia per creare un sistema di filtraggio collaborativo, vediamo come il sistema creato risponde ai problemi presentati nella sezione 1.3.1:

Il problema dei nuovi utenti. Abbiamo visto che è possibile costruire un ordinamento che nel caso peggiore si comporta come la previsione basata sulla popolarità. Una interessante questione tuttora aperta è lo studio dell'evoluzione temporale dell'archivio.

Il problema del rinnovo continuo. Per fronteggiare questo problema è stata definita una misura di qualità specifica. Massimizzando la RevCall, gli effetti del rinnovo continuo sono molto limitati.

Il problema della sparsità delle preferenze. Questo problema in un sistema molto dinamico è fittizio. Infatti il rinnovo continuo delle informazioni è guidato dal principio di Pareto, ed è quindi garantita la presenza di una base di elementi sufficientemente densi ed in continua evoluzione su cui operare.

Il problema delle prestazioni. Anche se la dimensionalità degli insiemi è molto elevata, l'elevata sparsità delle matrici permette di mantenere le prestazioni piuttosto elevate. Ci sono buone garanzie che l'uso di strutture dati specializzate possa permettere di operare in tempo reale anche su archivi di produzione.

Interessante notare che la stessa struttura usata per il filtraggio collaborativo può operare in campo semantico operando tra tag o in ambito sociologico operando tra utenti.

In altro importante aspetto emerso è che le folksonomie si differenziano dalle basi di dati comunemente studiate nell'apprendimento automatico. Infatti i problemi posti all'interno di una folksonomia sono spesso una via di mezzo tra i problemi di apprendimento supervisionato ed i problemi di apprendimento non supervisionato, e mancano attualmente di una formalizzazione completa.

Aver scoperto che la struttura interna di una folksonomia rispetta determinate caratteristiche è più importante di quanto possa sembrare a prima vista. Infatti sia nel campo del Recupero di Informazioni (Information Retrieval o IR) che nel campo dei Sistemi Esperti negli ultimi anni sono stati studiati metodi di accesso alle informazioni specializzati ed ottimizzati, metodi che possono essere utilizzati quasi direttamente in un nuovo dominio come le folksonomie.

In questo lavoro è stato mostrato un approccio euristico che dà già risultati soddisfacenti ed inoltre lascia intuire che ciò che è stato trovato ha probabilmente solo scalfito la superficie di una struttura molto più complessa e ricca.

Bibliografia

- [1] M. A. Aizerman, E. A. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. In *Automation and Remote Control*, number 25 in *Automation and Remote Control*, pages 821–837, 1964.
- [2] Marko Balabanovic and Yoav Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, March 1997.
- [3] Pierre Baldi, Paolo Frasconi, and Padhraic Smyth. *Modeling the Internet and the Web: Probabilistic Methods and Algorithms*, chapter 4. Wiley, April 2003.
- [4] D. Barbara. Quasi-cubes: A space-efficient way to support approximate multidimensional databases, 1998.
- [5] Grigory Begelman, Philipp Keller, and Frank Smadja. Automated tag clustering: Improving search and exploration in the tag space. In *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland, 2006*.
- [6] C. Berg, J. P. R. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups*. Springer, Berlin, 1984.
- [7] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learning Theory*, pages 144–152, 1992.
- [8] Kaushik Chakrabarti. Managing large multidimensional datasets inside a database system, 2001.
- [9] W. J. Frawley, G. Piattetsky Shapiro, and C. J. Matheus. Knowledge discovery in databases - an overview. *Ai Magazine*, 13:57–70, 1992.
- [10] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, December 1992.
- [11] Scott Golder and Bernardo A. Huberman. The structure of collaborative tagging systems, Aug 2005.
- [12] Eui-Hong Han and George Karypis. Feature-based recommendation system. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 446–452, New York, NY, USA, 2005. ACM Press.
- [13] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, January 2004.
- [14] Will Hill, Larry Stead, Mark Rosenstein, and George Furnas. Recommending and evaluating choices in a virtual community of use. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 194–201, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.
- [15] iProspect. iProspect search engine user behavior study (april 2006). Technical report, iProspect, 2006.
- [16] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing & Management*, 36(2):207–227, March 2000.

- [17] S. Lawrence and C. Giles. Searching the world wide web. *Science*, 280:98–100, 04 1998.
- [18] Wee S. Lee. Collaborative learning and recommender systems. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 314–321, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.
- [19] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. 209:415–446, 1909.
- [20] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994. ACM Press.
- [21] Paul Resnick and Hal R. Varian. Recommender systems. *Commun. ACM*, 40(3):56–58, March 1997.
- [22] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, November 1975.
- [23] Ellen Spertus, Mehran Sahami, and Orkut Buyukkokten. Evaluating similarity measures: a large-scale study in the orkut social network. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 678–684, New York, NY, USA, 2005. ACM Press.
- [24] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995.