Un Metodo per il Riconoscimento di Duplicati in Collezioni di Documenti*

Federica Mandreoli, Riccardo Martoglia e Paolo Tiberio

Università di Modena e Reggio Emilia,
Dip. di Ingegneria dell'Informazione, Modena, Italy
{tiberio.paolo, mandreoli.federica, martoglia.riccardo}@unimo.it

Sommario I recenti avanzamenti nella potenza di calcolo e nelle telecomunicazioni hanno creato le giuste condizioni per la diffusione globale di enormi moli di informazioni elettroniche e di nuovi strumenti per l'analisi del loro contenuto, sollevando problemi di information overload e, in particolare, di duplicate detection. I duplicati, cioè documenti molto simili che contengono approssimativamente le stesse informazioni, degradano l'efficacia e l'efficienza delle ricerche e, spesso, costituiscono anche violazioni di copyright.
In questo articolo introduciamo DANCER (Document ANalysis and Comparison ExpeRt), un sistema completo di duplicate detection che sfrutta idee innovative nell'ambito dell'information retrieval per l'identificazione dei documenti duplicati, utilizzando algoritmi e misure di similarità inedite in questo campo e sufficientemente fini da ottenere una buona efficacia nella maggior parte delle applicazioni. Inoltre, il sistema propone diverse nuove tecniche di data reduction che permettono di ridurre sia il tempo di esecuzione che lo spazio richiesto per la memorizzazione dei dati, senza compromettere la buona qualità dei risultati.

1 Introduzione al problema del Duplicate Detection

I recenti avanzamenti nella potenza di calcolo e nelle telecomunicazioni, uniti al costante calo dei costi di accesso e gestione dei dati su Internet, hanno creato le giuste condizioni per la diffusione globale del Web e, più in generale, delle informazioni elettroniche e di nuovi strumenti per l'analisi del loro contenuto.

La notevole dimensione di un tale insieme di informazioni, costituito prevalentemente da dati testuali, insieme alla moltitudine delle diverse sorgenti da cui provengono le informazioni, sollevano problemi di duplicazione ed usabilità. In effetti, perché tale volume di informazioni possa creare un valore aggiunto nelle diverse aree di Internet e dell'Information economy, le tecniche di Information Retrieval devono poter rispondere alle esigenze informative degli utenti sia efficacemente che efficientemente, risolvendo problemi di information overload e, in particolare, di duplicate detection.

I duplicati, cioè documenti molto simili che contengono approssimativamente le stesse informazioni, sono largamente diffusi per varie ragioni: lo stesso documento può essere memorizzato in più luoghi in forme quasi identiche (ad esempio siti mirror, sorgenti di dati parzialmente sovrapposte), oppure possono essere disponibili diverse versioni di un documento (ad esempio revisioni e aggiornamenti, riassunti, formati diversi). Inoltre, ci possono essere duplicati intenzionali e illegali, come copie fraudolente, particolarmente sgradite ma altrettanto frequenti nell'era dell'accesso e della distribuzione elettronica delle informazioni.

Per risolvere queste importanti problematiche, numerose tecniche per l'identificazione di documenti duplicati sono state definite in diversi contesti e per diversi scopi [4], senza raggiungere risultati pienamente soddisfacenti. Semplici approcci quali la sola identificazione

* Il presente lavoro è parzialmente supportato dal progetto Fondo Speciale Innovazione 2000 "Tecnologie per arricchire e fornire accesso ai contenuti" e da Logos S.p.A.

di documenti strettamente identici o l'assegnazione di identificatori univoci ai documenti sono chiaramente non applicabili o, per lo meno, inadeguati. Ancora, servizi quali COPS [2], KOALA [7] e DSC [3] si basano su misure di similarità non sufficientemente fini per ottenere una efficacia accettabile nella maggior parte delle applicazioni.

Pertanto, appare chiara la necessità di progettare un sistema per document similarity detection che sia in grado di andare oltre la ricerca di match esatti, sfruttando una misura di similarità adeguatamente espressiva ma senza rinunciare all'efficienza. Un tale sistema dovrebbe essere in grado di:

- migliorare il knowledge management e, in particolare, sia l'efficacia che l'efficienza delle ricerche in vaste collezioni di documenti, come digital library e portali di dati, dove l'unione di dati ottenuti da più sorgenti porta ad un alto livello di duplicazione alla sorgente [9];
- migliorare l'efficacia di servizi elettronici avanzati, come newsletter e disseminazione automatica di informazioni, dove i problemi di duplicazione sussistono sia alla sorgente che alla destinazione (ad esempio per invii ripetuti di messaggi molto simili);
- migliorare l'efficacia, o sicurezza, di un sistema per il copy detection e l'identificazione di violazioni di copyright.

L'utilità di un tale strumento in questi scenari appare notevole. Per quanto riguarda il primo punto, il successo dei servizi di ricerca è principalmente dovuto alla qualità e alla consistenza dei risultati e dei servizi forniti. I documenti duplicati non forniscono all'utente nessuna informazione aggiuntiva e pertanto abbassano l'accuratezza dell'insieme di risposta. In effetti, il knowledge management e la ricerca non sono utili se non si provvede a ricercare e rimuovere le informazioni duplicate od obsolete dalle collezioni.

Inoltre, i search engine sono recentemente evoluti in portali di dati con elenchi di argomenti, servizi di informazione, servizi di mail elettroniche gratuiti, newsletter. Una efficace identificazione dei duplicati è ancor più necessaria per fare sì che questi servizi possano essere effettivamente tali: il sempre crescente flusso di messaggi che questi inviano all'utente fanno sì che il tempo sia divenuto una risorsa preziosissima e dunque da non sprecare [5].

Ultimo, non certo per importanza, è il problema della protezione del copyright. In [2], Garcia-Molina et al. affermano giustamente che il mezzo elettronico rende molto più facile la copia e la distribuzione illegali delle informazioni. In effetti il problema della violazione di copyright ha assunto oggigiorno una rilevanza fondamentale: ad esempio, nel 1998 è stato emanato il Digital Millennium Copyright Act (DMCA) al fine di proteggere le informazioni trasmesse, registrate e pubblicate in forma elettronica. Schemi di *copy prevention*, quali l'uso di watermark o di schemi di autorizzazione hardware, rendono difficile lo scambio di qualunque informazione e non offrono sufficiente usabilità all'utente comune. Il miglior trade-off tra il proteggere e il fornire accesso alle informazioni sono i sistemi per *copy detection* che, ancora, devono fondare il proprio engine di confronto su una solida metrica di similarità.

In questo articolo introduciamo DANCER (Document ANalysis and Comparison ExpeRt), un sistema completo di duplicate detection che sfrutta idee innovative nell'ambito dell'information retrieval per l'identificazione dei documenti duplicati. Inoltre, il sistema propone diverse nuove tecniche di data reduction che permettono di ridurre sia il tempo di esecuzione che lo spazio richiesto per la memorizzazione dei dati, senza compromettere la buona qualità dei risultati. Questo è possibile grazie a tecniche di *filtering*, che riducono il numero di confronti inutili, e di *intra* e *inter-document reduction*, che riducono le quantità di informazioni memorizzate.

Il resto dell'articolo è organizzato come segue: nella Sezione 2 descriviamo il sistema, inclusa l'architettura e le tecniche di ricerca di similarità e di data reduction utilizzate. Nella Sezione 3 mostriamo i risultati di alcuni degli esperimenti condotti. Infine, la Sezione 4 conclude l'articolo.

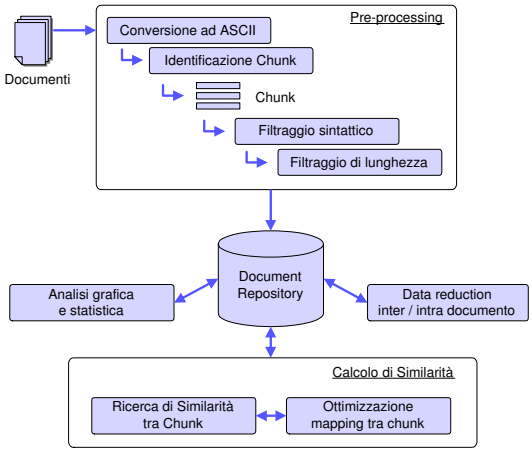


Figura 1. L'architettura del sistema DANCER

2 Il sistema DANCER

DANCER (Document ANalysis and Comparison ExpeRt) è un sistema in grado di fornire un servizio di duplicate detection tra i documenti presenti in un repository e i documenti query. Il servizio individua non solo copie esatte ma anche documenti contenenti sovrapposizioni significative.

DANCER è stato implementato in un prototipo, la cui architettura è riassunta in Figura 1. Il cuore del sistema è il *Document Repository*, il database che memorizza le rappresentazioni interne dei documenti. Parte integrante di tali rappresentazioni sono i *chunk*, estratti nella fase di *Pre-processing* e di cui si parlerà nella Sezione 2.1.

Gli altri moduli del sistema accedono a queste informazioni e svolgono su di esse le diverse operazioni disponibili, incluso il fondamentale *Calcolo di Similarità* tra documenti (Sezione 2.2) che realizza il servizio di duplicate detection. Altre funzionalità sono le operazioni di *Data Reduction*, cui è dedicata la Sezione 2.3, e l'*Analisi grafica e statistica* dei documenti, che per limiti di spazio non potrà essere trattata in questa sede.

2.1 Pre-processing

Lo scopo dei moduli di preprocessing è di tradurre i documenti da memorizzare in DANCER in una forma "interna" adatta ai calcoli di similarità (vedi Sezione 2.2). Alla base del preprocessing è il concetto di *chunk*. I chunk catturano l'informazione strutturale insita nei documenti, essendo sequenze consecutive delle loro parti fondamentali (parole, frasi o paragrafi). I chunk che utilizziamo sono unità che hanno un significato a sé stante e in grado di definire un contesto. A questo proposito non considereremo chunk più piccoli della singola frase, che rappresenta un contesto di parole il cui ordine ne determina il significato. Nella maggior parte dei casi, utilizzeremo quindi chunk costituiti da singole frasi o paragrafi.

In particolare, il Pre-processing dei documenti, il cui scopo è quello di memorizzare un insieme di chunk per ogni documento, consiste dei seguenti passi: conversione ad ASCII del contenuto dei documenti, identificazione dei chunk, filtraggio sintattico e filtraggio di lunghezza dei chunk. Per quanto riguarda l'*Identificazione dei Chunk*, il sistema è in grado di identificare correttamente le frasi e i paragrafi di un documento e fornisce all'utente la possibilità di scegliere la dimensione preferita del chunk in termini di un numero prefissato di frasi o paragrafi. Per identificare le corrette suddivisioni con precisione e flessibilità, DANCER fa uso di un semplice automa a stati finiti.

Prima che possano essere inseriti nel database, i chunk estratti vengono sottoposti a operazioni di *filtraggio*. Per aumentare la resilienza del nostro approccio rispetto a cambiamenti poco significativi apportati ai documenti, viene applicato un *filtraggio sintattico* che rimuove suffissi e stopword e pratica lo stemming, portando i termini rimasti nella loro forma base [1].

Con il *filtraggio di lunghezza*, infine, il sistema esegue una selezione dei chunk applicando una soglia di lunghezza che esprime la minima lunghezza in parole. Impostando tale soglia al valore di 2 o 3 parole, questo semplice filtro è in grado di migliorare notevolmente la performance del sistema senza intaccare il significato dei documenti.

2.2 Calcolo di Similarità

In questa sezione presentiamo il modulo principale che realizza il servizio di duplicate detection attraverso un confronto di documenti basato su una nuova misura di similarità. Diamo innanzitutto un'intuizione delle proprietà che, a nostro parere, una buona misura di similarità tra due documenti deve soddisfare. Tale misura di similarità deve essere in grado di quantificare con efficacia il livello di duplicazione dei due documenti e catturare la nozione informale di "sovrapposizione significativa". Analogamente a [13], si consideri un documento D_0 da confrontare con un generico documento D_i. Si consideri D_0 rappresentato dalla sequenza di chunk indicata dalle lettere ABC, e si considerino i seguenti documenti: $D_1 = ABC$, $D_2 = BAC$, $D_3 = AB$, $D_4 = ABCD$, $D_5 = A \dots A$ (n volte), $D_6 = A'BC$. Informalmente, ci si aspettano i seguenti risultati: D_0 è l'esatto duplicato di D_1 e D_2, essendo quest'ultimo costituito da una ri-disposizione dei chunk di D_0; D_3 e D_4 sono abbastanza simili a D_0 e D_3 (D_4) è tanto più simile quanto più è corto il chunk C (D) rispetto ad A e B; D_5 è abbastanza simile per bassi n e non molto simile per alti n; D_6 è tanto più simile quanto più A' è simile ad A.

Il modulo per il *Calcolo di Similarità* è in grado di calcolare le similarità tra i documenti appartenenti a due insiemi, S_i ed S_j in due fasi: la *Ricerca di Similarità tra Chunk* e la *Ottimizzazione del Mapping tra Chunk*. La *Ricerca di Similarità tra Chunk* esegue un match approssimato tra i chunk dei documenti di S_i e dei documenti di S_j. Per farlo occorre innanzitutto specificare una metrica di similarità tra chunk. A differenza della maggior parte dei sistemi di copy detection, proponiamo una misura che si spinge oltre la stretta uguaglianza, analizzando il contenuto dei chunk e calcolando *quanto* sono simili. Per fare questo, consideriamo un chunk come una sequenza di termini e introduciamo una misura di similarità che si basa su una delle metriche più sfruttate nell'ambito delle sequenze testuali: la nozione di *edit distance* [12].

Definizione 1 (Similarità tra chunk). *Dati due chunk $c_i^h \in D_i$ e $c_j^k \in D_j$, la similarità tra c_i^h and c_j^k è così definita:*

$$sim(c_i^h, c_j^k) = \begin{cases} 1 - \frac{ed(c_i^h, c_j^k)}{max(|c_i^h|, |c_j^k|))} & se \ \frac{ed(c_i^h, c_j^k)}{max(|c_i^h|, |c_j^k|))} < t \\ 0 & altrimenti \end{cases} \quad (1)$$

dove ed() rappresenta l'edit distance (in parole) tra i chunk, $|c_i^h|$ ($|c_j^k|$) rappresenta la lunghezza (in parole) di un chunk e t una soglia di distanza relativa.

Per ogni coppia di documenti $D_i \in S_i$ e $D_j \in S_j$, le cui rappresentazioni interne in termini di chunk sono rispettivamente $c_i^1 \dots c_i^n$ e $c_j^1 \dots c_j^m$, questa prima fase restituisce tutte le coppie di chunk $c_i^h \in D_i$ e $c_j^k \in D_j$ tali che $sim(c_i^h, c_j^k) > 0$. La ricerca di similarità tra chunk è implementata sul DBMS tramite espressioni SQL che, prima di calcolare l'edit distance, filtrano rapidamente le coppie di chunk che non fanno match utilizzando tecniche di filtering definite ad hoc (per maggiori dettagli fare riferimento a [11]). Si noti che un chunk

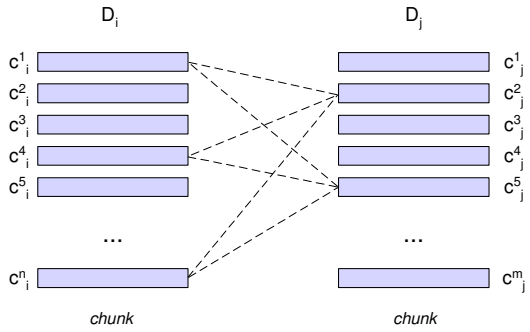


Figura 2. Il problema del mapping tra i chunk

nel documento D_i potrebbe essere simile a più di un chunk nel documento D_j e che questi chunk in D_j potrebbero, a loro volta, avere similarità con altri chunk di D_i (si veda Figura 2).

Fra tutti i possibili mapping fra chunk, l'idea che sta alla base della nostra misura è di ricercare il mapping che massimizza la similarità tra i due documenti sulla base delle similarità tra chunk corrispondenti.

Definizione 2 (Similarità tra documenti). *Dati due documenti D_i e D_j, la similarità tra essi, detta $Sim(D_i, D_j)$, è definita come il massimo della seguente funzione:*

$$\frac{\sum_{k=1}^{m}\sum_{h=1}^{n}\left(|c_i^h| + |c_j^k|\right)\cdot sim(c_i^h, c_j^k)\cdot x_{h,k}}{\sum_{k=1}^{n}|c_i^k| + \sum_{k=1}^{m}|c_j^k|}$$

dove $|c_i^h|$ e $|c_j^k|$ rappresentano la lunghezza (in parole) dei chunk e $x_{h,k}$ è una variabile booleana che assume i seguenti valori:

$$x_{h,k} = \begin{cases} 1 & \textit{se il chunk } c_i^h \textit{ è associato al chunk } c_j^k \\ 0 & \textit{altrimenti} \end{cases}$$

I vincoli sono i seguenti:

$$\forall k \quad \sum_h x_{h,k} \leq 1, \qquad \forall h \quad \sum_k x_{h,k} \leq 1 \tag{2}$$

cioè un chunk in un documento è accoppiato con al più un chunk dell'altro.

In DANCER, il problema di accoppiare ogni chunk di un documento con al più un chunk dell'altro, considerando le similarità reciproche, è implementato nel modulo di *Ottimizzazione del Mapping tra i Chunk* come un problema di programmazione lineare intera, per il quale è possibile utilizzare pacchetti standard per PLI. Si noti che la risoluzione di un tale problema deve essere richiamata solo per le coppie di chunk c_i^h nel documento D_i e c_j^k nel documento D_j tali che c_i^h sia associato a più di un chunk in D_j, includendo anche c_j^k, e viceversa. In tutti gli altri casi, il mapping tra i chunk può essere calcolato direttamente. Sottolineiamo che impostare la soglia di similarità t nel calcolo della similarità tra i chunk a $0,3$ o ad un valore più basso non ha impatto sull'efficacia della misura (si veda la Sezione 3) e permette di migliorare l'efficienza del sistema, attivando la risoluzione del problema PLI completo molto di rado.

2.3 Data reduction

Il modulo di Calcolo di Similarità può rappresentare il *core* di tecniche di analisi di documenti basate su distanze. In particolare, ci siamo concentrati su problemi di *ricerca di similarità* e *clustering*. In un contesto di ricerca di similarità, dato un documento query il sistema recupera i documenti dello spazio di ricerca le cui similarità con quello di query non eccedono una determinata soglia (*range query*) o sono le più elevate (*k-nearest neighbor query*). Con il clustering, invece, il sistema analizza grandi insiemi di documenti e li raggruppa in cluster, collezioni di documenti simili che possono pertanto essere trattati collettivamente come gruppo. Tale clustering può aiutare, ad esempio, a risolvere problemi di documenti duplicati o URL instabili su web [3].

In entrambi gli scenari, il confronto di documenti, in particolare per grandi collezioni, è un'operazione che richiede grandi quantità di tempo e di spazio. Per questa ragione, DANCER propone tre nuovi tipi di funzionalità di *Data reduction* che permettono di ridurre tali requisiti pur mantenendo una elevata efficacia nei risultati. Nel nostro contesto, *data reduction* significa ridurre lo spesso enorme numero di confronti richiesti dalle tecniche di analisi dello spazio dei documenti basate su funzioni di distanze. Gli approcci pensati sono i seguenti:

- *filtri*, per ridurre il numero di confronti inutili;
- *riduzione intra-documento*, per ridurre il numero di chunk in ogni documento;
- *riduzione inter-documento*, per ridurre il numero di documenti memorizzati.

Tali approcci sono ortogonali e, per questo, pienamente combinabili. L'impatto sullo spazio di ricerca dei documenti è il seguente: i filtri lo lasciano invariato, la riduzione intra-doc introduce un'approssimazione nella rappresentazione logica dei documenti, riducendo lo spazio occupato mediante chunk campione, mentre la riduzione inter-doc approssima lo spazio di ricerca eliminando documenti e mantenendone campioni significativi. In altre parole, i filtri assicurano la completezza dei risultati, diminuendo il numero di confronti necessari e quindi il tempo di risposta ma lasciando invariato il repository dei documenti, mentre le altre due classi di tecniche riducono anche lo spazio richiesto per la memorizzazione dei dati, pur introducendo una approssimazione dei risultati. Per queste ragioni, i filtri e le tecniche di riduzione intra-doc possono essere utilizzati sia per la ricerca di similarità sia per il clustering, mentre quelle di riduzione inter-doc possono essere applicate solo al problema di clustering.

I filtri presenti in DANCER sono diversi e riguardano sia l'identificazione dei chunk (filtri di *lunghezza*, *posizione* e *conteggio*) sia dei documenti simili (filtro *doc-pair*). Per motivi di spazio, rimandiamo la descrizione dei filtri e delle tecniche di riduzione inter-doc ad un futuro lavoro [10], soffermandoci in questa sede sulle sole tecniche di riduzione intra-doc.

Come già accennato, la riduzione intra-documento mira a ridurre il numero di chunk registrati e confrontati nelle elaborazioni. Proponiamo due diverse tecniche: *selezione length-rank* e *clustering* dei chunk. Entrambe agiscono selezionando una percentuale di chunk di ciascun documento specificata da un fattore di riduzione *chunkRatio*. Ognuna è completamente configurabile ed è in grado di ottenere buoni risultati, sia per efficacia sia per efficienza, in collezioni di documenti di diverso tipo e dimensione (si veda la Sezione 3 dedicata alle prove sperimentali).

La *selezione length-rank* è una alternativa all'approccio del sampling (campionamento a passo fisso o aleatorio) proposto in [2,7]. Opera selezionando per ogni documento una percentuale *sampRatio* di suoi chunk corrispondenti a quelli più lunghi. Pur essendo un'idea ugualmente semplice, funziona molto meglio del sampling poiché tende a selezionare chunk simili da documenti simili. Per questo tale tecnica si è dimostrata di grande efficacia, mantenendo risultati di buona qualità pur riducendo significativamente i tempi di risposta e i requisiti di spazio.

Il *clustering* dei chunk è il processo di clusterizzazione nello spazio dei chunk corrispondente alla rappresentazione di un documento. L'intuizione è semplice: se un documento contiene due (o più) chunk molto simili, ne viene memorizzato uno solo ma ne viene aumentato il peso relativo nel calcolo della similarità tra documenti, mantenendo così approssimativamente lo stesso risultato finale. Più precisamente, dato un documento D contenente n chunk, l'algoritmo di clustering, basato su un clustering agglomerativo gerarchico di tipo complete-link [8], produce $chunkRatio * n$ cluster di chunk. Di ogni cluster manteniamo alcune caratteristiche:

- il *centroide*, corrispondente al chunk che minimizza il massimo delle distanze tra se stesso e gli altri elementi del cluster;
- il *peso totale* (in parole) degli elementi (chunk) del cluster;
- il *numero totale* degli elementi del cluster.

Tali caratteristiche vengono poi sfruttate in una variante della similarità tra documenti standard presentata nella sezione precedente. Per motivi di spazio ci si limita in questa sede ad enunciare le principali modifiche necessarie: la similarità complessiva è formata da contributi relativi ai cluster (non più ai chunk), calcolati valutando le similarità tra i rispettivi rappresentanti e pesandole sulle lunghezze totali (o medie, in una ulteriore versione) dei chunk contenuti nei cluster stessi. L'efficacia di un tale approccio (si veda la Sezione 3) è dovuta al fatto che è basato sulla stessa metrica di similarità tra chunk usata nel calcolo della similarità finale tra documenti. Grazie alle similarità tra i chunk, si è in grado di scegliere i "giusti" rappresentanti, dando risultati particolarmente buoni per documenti con una notevole ripetitività interna.

3 Prove sperimentali

Per valutare sia l'efficacia che l'efficienza del sistema DANCER abbiamo condotto numerosi esperimenti utilizzando il prototipo descritto nella precedente sezione. In questa sezione descriveremo lo strumento di generazione automatica di documenti che abbiamo sviluppato per creare la maggior parte delle collezioni usate nei test, le collezioni e i risultati ottenuti in alcuni dei test eseguiti.

3.1 Il Generatore di Documenti

Per valutare più approfonditamente le prestazioni del sistema, abbiamo sviluppato un *Generatore di Documenti*, progettato per produrre automaticamente variazioni casuali da un insieme di documenti iniziali. L'algoritmo prende in input due insiemi di documenti: un insieme di documenti "seme", contenente i documenti a cui generare variazioni, e uno di "varianti", da cui estrarre il nuovo materiale necessario alla modifica dei documenti iniziali.

Il generatore di documenti è pienamente parametrizzabile e opera applicando trasformazioni casuali alle differenti parti di un documento. Le modifiche sono scelte a caso tra cancellazioni, scambi, inserimenti o sostituzioni e interessano le seguenti componenti di un documento, con una frequenza specificata dal relativo parametro:

- paragrafi (uno modificato ogni *parStep* paragrafi);
- frasi (una modificata ogni *sentStep* frasi);
- parole (una modificata ogni *wordStep* parole).

Infine, è disponibile un parametro aggiuntivo, il numero di documenti (varianti) da generare a partire da ogni documento seme.

Si noti che gli algoritmi del generatore sono basati su flussi indipendenti di numeri casuali, che determinano il tipo di modifica da operare e, in caso di inserimenti o sostituzioni, gli elementi dei documenti varianti da usare per la corrispondente modifica.

3.2 Le Collezioni di Documenti

Per i nostri test abbiamo creato ed utilizzato diversi insiemi di documenti, provenienti da diverse sorgenti e studiati per mettere alla prova le diverse caratteristiche e capacità del sistema in vari scenari:

- **NewsMix50, NewsMix100**: collezioni sintetiche di 50 e 100 documenti, contenenti variazioni a partire da 10 documenti seme, estratti da news distribuite in varie mailing list;
- **Times100L, Times100S, Times500S**: collezioni sintetiche di 100 documenti lunghi, 100 e 500 documenti più brevi, contenenti variazioni da 10, 10 e 50 documenti seme rispettivamente, estratti da articoli del Times;
- **News120**: collezione reale di 120 documenti, estratti da vari gruppi di mailing list (4 gruppi di 30 documenti ciascuno).

I primi due gruppi sono costituiti da collezioni sintetiche, create con il generatore di documenti per verificare accuratamente il comportamento del sistema. Per tali collezioni, per motivi di spazio, in questo articolo presenteremo solo i test più significativi svolti sulle collezioni NewsMix50, NewsMix100 e Times100L, sufficienti per mostrare le principali caratteristiche del sistema.

Per quanto riguarda le collezioni NewsMix50 e NewsMix100, contenenti rispettivamente circa 1800 e 3200 frasi, sono state ottenute impostando il generatore con i parametri che si sono rivelati più adatti a simulare uno scenario di modifiche ai documenti sufficientemente vario ma pur sempre realistico ($sentStep = 6$ e $wordStep = 8$). Nella collezione più grande, Times100L, di oltre 10000 frasi, abbiamo sperimentato anche una impostazione leggermente più "aggressiva" ($sentStep = 4$ e $wordStep = 5$).

La collezione News120 è invece studiata per verificare il comportamento del sistema con insiemi di documenti reali, quindi con meno correlazioni interne. Per valutare l'efficacia delle misure, questa collezione doveva presentare una caratteristica fondamentale: essere chiaramente clusterizzabile in diversi gruppi, utilizzando una similarità sul contenuto e non solo sugli argomenti. A questo proposito, particolarmente adatti si sono rivelati i messaggi raccolti da quattro mailing list di ricerca, cioè DBWORLD (DBW), Description Logics (DL), Semantic Web (SW), e Adaptive Hypermedia (AH), caratterizzati da un buon livello di ripetizione intra-gruppo e da una correlazione incrociata piuttosto bassa.

Si noti infine che le collezioni qui presentate portano ad un prodotto incrociato che varia da 2.500 (50*50) a 10.000 (100*100) confronti tra documenti e da più di 3.000.000 a quasi 120.000.000 di confronti tra chunk (frasi).

3.3 Risultati

La performance del sistema è stata verificata sperimentalmente sia per quanto riguarda l'efficacia che per l'efficienza delle tecniche proposte. I test sono stati eseguiti su una workstation Pentium 4 1.8Ghz su Windows XP Pro, utilizzando un soglia di distanza t compresa tra $0,2$ e $0,4$ per le tecniche di filtering su edit distance.

Test di Efficacia Per quanto riguarda l'efficacia, è stata verificata valutando i punteggi di similarità ottenuti sulle varie collezioni. Abbiamo anche messo alla prova la robustezza della misura di similarità rispetto alle diverse tecniche di data reduction descritte nella Sezione 2.3.

Il metodo elaborato è il seguente: si inseriscono nel sistema i documenti della collezione interessata, quindi si esegue un *self approximate join* tra i documenti eseguendo query per ogni documento rispetto all'insieme completo e si calcola una matrice di similarità tra tutte le coppie di documenti. Abbiamo analizzato tale matrice in vari modi:

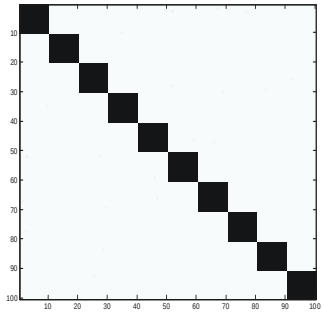


Figura 3. Efficacia: risultati sulla collezione Times100L, visualizzazione a matrice grafica

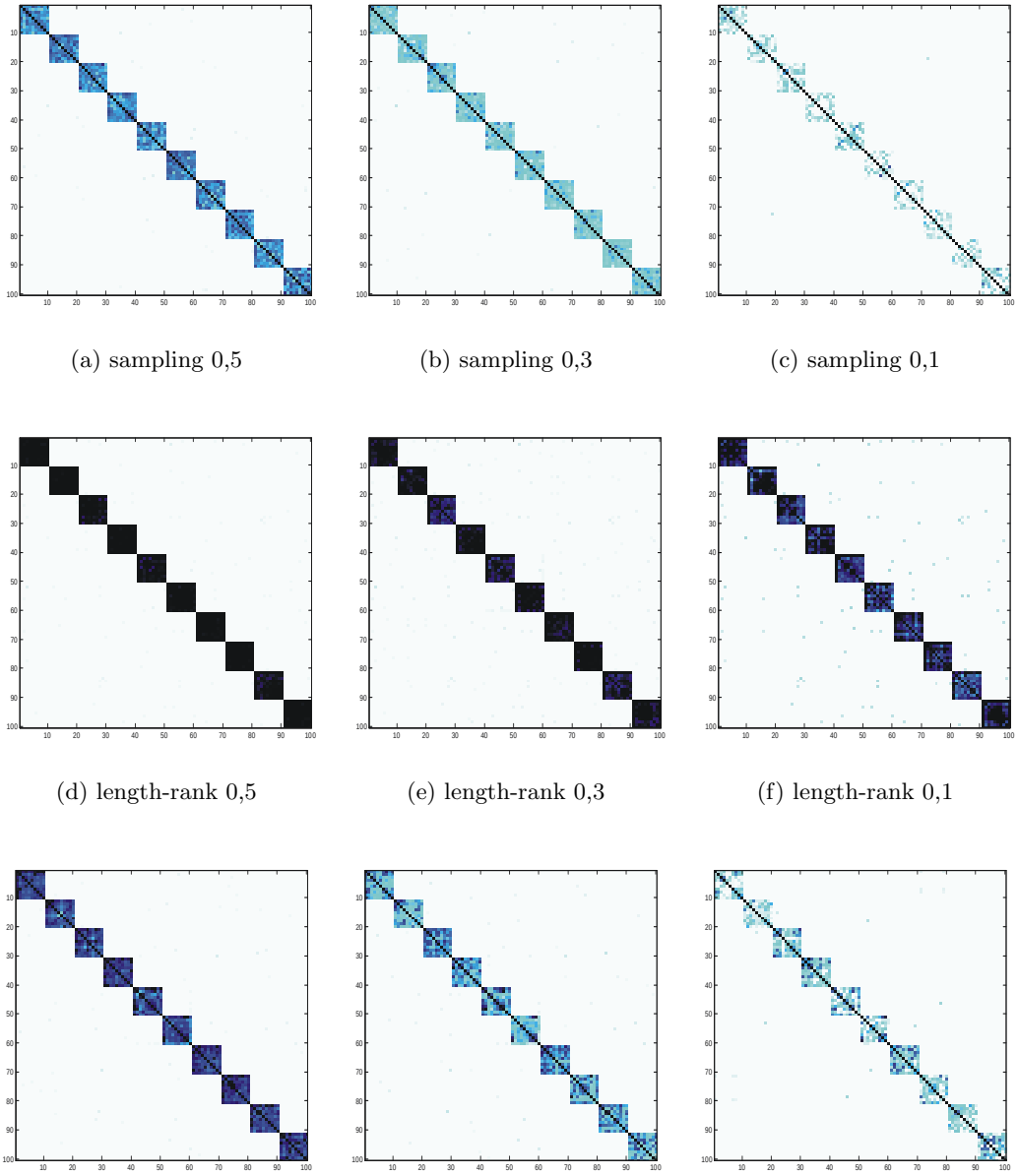
– visualizzazione grafica diretta dei valori della matrice di similarità;
– calcolo di media e deviazione standard nelle misure di similarità.

In una visualizzazione grafica diretta della matrice, l'immagine è generata direttamente dall'output, associando particolari sfumature di colore ai diversi valori di similarità. In questo modo, ogni pixel rappresenta la similarità tra una coppia di documenti: un colore scuro è associato a coppie poco simili, uno chiaro a coppie con maggiore somiglianza. Questa tecnica permette di mostrare e cogliere fin da un primo sguardo la qualità dei risultati di similarità, pur preservando molti dei dettagli richiesti per un'analisi approfondita.

La figura 3 mostra l'immagine generata da questa tecnica per la collezione Times100L senza applicare tecniche di data reduction. Si noti che, per rendere l'immagine leggibile a colpo d'occhio, tutti i documenti generati da uno stesso seme sono stati inseriti nel sistema con id consecutivi e compaiono pertanto in una zona specifica del grafico. La Figura 3 mostra che i 10 gruppi di 10 documenti simili sono chiaramente visibili e identificati dal sistema. I risultati ottenuti mostrano la robustezza (anche chiamata sicurezza in [2]) delle misure di similarità rispetto alle modifiche: i nostri punteggi di similarità sono proporzionali al numero di cambiamenti applicati ai documenti seme ed occorre apportare molte modifiche per renderli non più correlabili ai documenti originali.

Mostriamo ora quanto le diverse tecniche di data reduction presentate di tipo intra documento influenzino l'efficacia del sistema, presentando la matrice grafica della stessa collezione per diversi parametri di riduzione. I risultati dei nostri approcci vengono confrontati a quelli ottenibili mediante la tecnica di sampling utilizzata in [2]. Per tale confronto, abbiamo considerato sia un algoritmo di sampling a passo fisso, che mantiene un chunk ogni i, sia uno di tipo *sequential random* proposto da Vitter [14]. Poiché i due tipi di sampling hanno portato approssimativamente allo stesso livello di qualità, presentiamo solo i risultati relativi al sampling random. La Figura 4 confronta l'effetto sull'output di similarità dei diversi livelli di sampling (prima riga) rispetto alle nostre tecniche di selezione length-rank (seconda riga) e di clustering (terza riga). Ad esempio, un'impostazione di *chunkRatio* a $0,1$ mantiene un chunk su dieci. Con sampling a $0,5$ e $0,3$ i gruppi sono ancora distinguibili ma i punteggi di similarità sono molto ridotti (punti più scuri). Con sampling a $0,1$ la qualità dei risultati è molto bassa. La qualità fornita dalla selezione length-rank è, invece, molto alta e i punteggi di similarità ottenuti, anche al più alto livello di selezione $(0,1)$, sono sempre migliori di qualunque risultato del sampling. Come ci si poteva aspettare, un approccio di sampling random non è in grado di comportarsi in modo simile con documenti simili e pertanto non ha la stessa qualità del nostro approccio di selezione.

La terza riga di Figura 4 mostra i risultati ottenuti con l'altra tecnica di riduzione intradoc presentata, il clustering. Poiché il livello di ripetizione interno alla collezione è abbastanza



(a) sampling 0,5 (b) sampling 0,3 (c) sampling 0,1

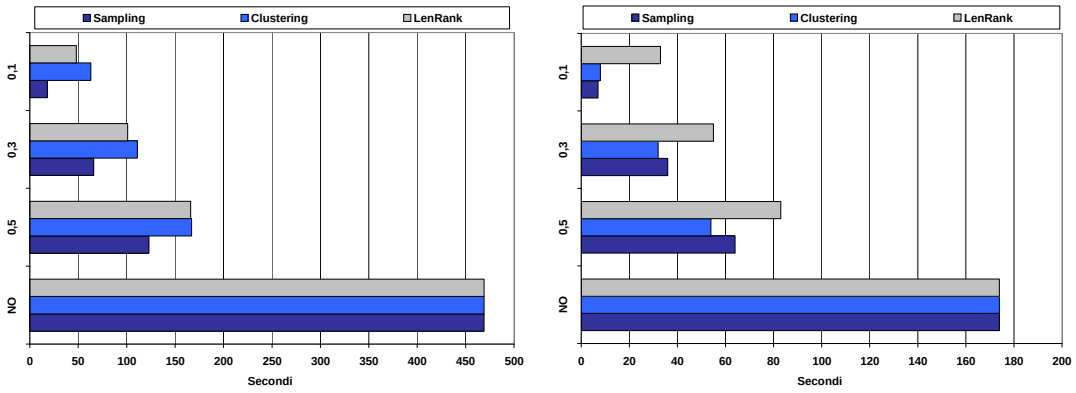
(d) length-rank 0,5 (e) length-rank 0,3 (f) length-rank 0,1

(g) clustering 0,5 (h) clustering 0,3 (i) clustering 0,1

Figura 4. Efficacia: risultati dei test intra-doc per la collezione Times100L

basso, i risultati non raggiungono i livelli di qualità mostrati per la selezione length-rank. Si noti che i risultati hanno una qualità accettabile fino a livelli di riduzione di $0,2$-$0,1$.

Infine, per verificare l'efficacia del sistema nel discernere documenti correlati anche in insiemi di documenti reali, abbiamo inserito i 4 gruppi di 30 documenti della collezione News120 e calcolato i valori di *affinità* e *rumore* rispetto alla nostra misura di similarità. Il livello di affinità è calcolato all'interno dei documenti di ogni gruppo e rappresenta quanto questi sono



(a) Collezione Times100L

(b) Collezione NewsMix100

Figura 5. Efficienza: risultati dei test

vicini in termini di similarità (tra 0, nessuna somiglianza e 1, documenti identici). Il livello di rumore, invece, è calcolato tra i documenti di un gruppo rispetto a quelli degli altri e rappresenta i match indesiderati tra documenti che appartengono a gruppi diversi. La media e la deviazione standard dei valori di similarità ottenuti sono mostrate in Tabella 1.

Group	**Correlati(Affinità)**	**Scorrelati(Rumore)**
DBW	$0,1269 \pm 0,2534$	$0,0005 \pm 0,0022$
AH	$0,1357 \pm 0,3335$	$0,0005 \pm 0,002$
SW	$0,0925 \pm 0,282$	$0,0005 \pm 0,0023$
DL	$0,1357 \pm 0,3198$	$0,0007 \pm 0,0028$
	$0,1227 \pm 0,2972$	$0,0005 \pm 0,0023$

Tabella 1. Media e deviazione standard di affinità e rumore per la collezione News120

Si notino i livelli estremamente bassi di rumore e gli alti valori di affinità, due ordini di grandezza più grandi. Questo conferma ancora una volta la bontà della nostra misura di similarità, anche con insiemi di documenti reali.

Test di efficienza Quanto all'efficienza delle tecniche proposte, abbiamo misurato le prestazioni a runtime del sistema nell'eseguire le operazioni descritte precedentemente.

La Figura 5 mostra i tempi di elaborazione sulle collezioni Times100L e NewsMix100. I tempi mostrati comprendono il calcolo di tutti i punteggi di similarità per tutte le coppie di documenti delle collezioni e sono mostrati sia senza che con ciascuna delle tecniche di data reduction intra-documento. Come si vede, in caso di attivazione di tali tecniche i tempi di calcolo vengono sensibilmente ridotti: ad esempio, una selezione length-rank impostata a $0,1$ permette di ridurre i tempi di un fattore 1:8, senza compromettere la buona qualità dei risultati (si veda la sezione precedente). In generale, le nostre tecniche di riduzione intra-doc offrono un buon trade-off tra efficacia ed efficienza, permettendo miglioramenti prestazionali comparabili all'approccio del sampling, ma con risultati di ben altra qualità.

Per motivi di spazio, anche in questo caso non vengono mostrati i risultati relativi alle altre tecniche di riduzione (filtri e inter-doc) ma si noti che l'attivazione e la combinazione di tali tecniche con quelle intra-doc permettono riduzioni di tempo di altrettanta entità.

Come nota finale, i tempi di pre-processing delle tecniche di data reduction non sono mostrati poiché, come il pre-processing di inserimento documenti, non sono parte del calcolo di similarità ma costituiscono una fase preparatoria iniziale da eseguire una volta per tutte. Comunque, in tecniche quali la selezione length-rank tale preparazione è praticamente istantanea, mentre il clustering richiede ovviamente un tempo maggiore.

4 Conclusioni

In questo articolo abbiamo presentato DANCER (Document ANalysis and Comparison ExpeRt), un sistema completo di duplicate detection che sfrutta una nuova metrica di similarità e innovative tecniche di data reduction avanzate per l'identificazione efficace ed efficiente dei documenti duplicati.

La metrica, flessibile e di buona precisione, permette di operare con chunk di dimensione non prefissata e di grandi dimensioni (es. frasi), non compromettendo la buona qualità dei risultati e fornendo un livello di sicurezza inedito in caso di utilizzo in ambito di copy e copyright-violation detection. Le diverse tecniche di data reduction, come mostrato, permettono infine di ridurre sia il tempo di esecuzione che lo spazio richiesto per la memorizzazione dei dati.

Riferimenti bibliografici

1. R. Baeza-Yates e B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.
2. Sergev Brin, James Davis, e Hector Garcia-Molina. Copy Detection Mechanisms for Digital Documents. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pp. 398–409, 1995.
3. A.Z. Broder, S.C. Glassman, M.S. Manasse, e G. Zweig. Syntactic clustering of the Web. *Computer Networks and ISDN Systems*, 29(8-13):1157–1166, 1997.
4. A. Chowdhury, O. Frieder, e D. Grossman. Collection statistics for fast duplicate document detection. *ACM Transactions on Information Systems*, 20(2):171–191, 2002.
5. P. Denning. Internet time out. *Communications of the ACM*, 45(3):15–18, 2002.
6. C. Faloutsos e K. Lin. FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pp. 163–174, 1995.
7. N. Heintze. Scalable Document Fingerprinting. *Second Usenix Workshop on Electronic Commerce*, pp. 191–200, 1996.
8. A.K. Jain, M.N. Murty, e P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.
9. S. Lawrence, K. Bollacher, e C. Lee Giles. Indexing and Retrieval of Scientific Literature. In *Proc.s of 8th Int'l Conf. on Information and Knowledge Management (CIKM)*, 1999.
10. F. Mandreoli, R. Martoglia, e P. Tiberio. A Method for Document Similarity Detection. To appear.
11. F. Mandreoli, R. Martoglia, e P. Tiberio. A Syntactic Approach for Searching Similarities within Sentences. In *Proc. of the 11th ACM Conference of Information and Knowledge Management (ACM CIKM)*, 2002.
12. G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.
13. N. Shivakumar e H. Garcia-Molina. SCAM: A Copy Detection Mechanism for Digital Documents. In *Proceedings of the 2nd International Conference on Theory and Practice of Digital Libraries*, 1995.
14. J.S. Vitter. An Efficient Algorithm for Sequential Random Sampling. *ACM Transactions on Mathematical Software*, 13(1):58–67, 1987.