Indice

Elenco delle figure

Capitolo 1

Introduzione

Questa tesi ha come obbiettivo lo studio e l'analisi delle tecniche di spam detection attualmente esistenti ed in particolare delle tecniche online. Nella prima parte le tecniche verranno classificate sulla base dei segnali che utilizzano. Successivamente verranno eseguiti dei test per valutare alcuni algoritmi di spam detection offline eseguiti durante la fase di crawling. Ed infine verranno presentati e discussi i risultati ottenuti. Attualmente sono poche le tecniche online di spam detection, ovvero tecniche che rilevano lo spam durante la fase di crawling. Infatti quasi tutti i metodi tentano di fare il crawling dell'intera porzione di web di interesse e successivamente classificare le pagine in classi (dove di norma le classi sono due: *spam* oppure *non spam*).

Il fenomeno del web spam è sempre più presente all'interno del web: questo è dovuto al fatto che gli utenti tendono ad esaminare solo i primi risultati calcolati dai motori di ricerca e quindi se un sito compare tra i primi n risultati può avere un ritorno economico maggiore, legato alla quantità di traffico che viene generata per quel sito. Da uno studio del 2005 [3] si stima che la perdita finanziaria mondiale causata dallo spam è di circa 50 miliardi di dollari mentre nel 2009 tale perdita è salita a 130 miliardi di dollari, come descritto in [4]. Risulta ovvio, quindi, perché recentemente tutte le più grandi compagnie di motori di ricerca hanno identificato nel recupero di informazioni non pertinenti una delle priorità da risolvere.

Le conseguenze del web spam possono essere riassunte come segue [5]:

- la qualità delle ricerche è compromessa penalizzando i siti web legittimi;
- un utente potrebbe perdere la fiducia sulla qualità di un motore di ricerca e passare con facilità all'utilizzo di un altro;
- inoltre i siti spam possono essere usati come mezzo per malware, pubblicazione di contenuto per adulti e attacchi di tipo "fishing". Una prova tangibile si può vedere in [6], dove gli autori hanno eseguito l'algoritmo di *PageRank* su 100 milioni di pagine e hanno notato che 11 sui primi 20 risultati erano composti da siti con contenuto per adulti.

Queste considerazioni evidenziano che nella progettazione di un motore di ricerca occorre tenere conto delle pagine che potrebbero portare al mal funzionamento del motore stesso.

Il lavoro prodotto da questa tesi sarà utilizzato per essere integrato all'interno di un web crawler distribuito ad alte prestazioni per il futuro sviluppo di un modulo di spam detection. L'esigenza di tale modulo è sorta a seguito dello sviluppo, presso il Dipartimento, di un crawler chiamato *BUbiNG*, altamente configurabile ma privo al momento di qualunque forma di rilevazione di siti e contenuti malevoli. Il problema è estremamente interessante sia dal punto di vista teorico che da quello pratico: infatti, sebbene siano numerose le tecniche descritte in letteratura per la determinazione di spam (usando come segnali sia il contenuto che la struttura dei link), è sorprendentemente scarso l'insieme di tali tecniche che possono essere usate on-line, cioè durante il crawl. Il problema diventa ancora più complesso se si aggiungono considerazioni legate ai vincoli di spazio di memoria disponibile e al tempo di calcolo.

In letteratura il processo di spam detection viene eseguito quasi sempre offline, subito dopo la fase di crawling. Tale processo è integrato nella fase di ranking dei motori di ricerca:

- crawling dell'intero web;
- fase di spam detection;
- indicizzazione.

Il motivo per cui la fase di spam detection viene eseguita dopo la fase di crawling è perché si utilizza il grafo risultante per determinare le pagine spam.

Partendo da queste considerazioni, in questa tesi verranno effettuate delle analisi per determinare se il processo di spam detection possa essere eseguito durante la fase di crawling ovvero al momento in cui il crawler esegue il "fetch" di una pagina per determinare "on the fly" se la pagina è buona o ha un contenuto malevolo.

1.1 Ranking dei motori di ricerca

Prima di spiegare i vari metodi con cui si possono creare pagine web spam e i vari metodi utili ad identificarlo, è necessario capire come i motori di ricerca siano capaci di valutare la rilevanza di una pagina web per una determinata query.

In linea di massima un sistema di reperimento di informazioni, ovvero un motore di ricerca, è dato da una collezione documentale D (un insieme di documenti) di dimensione N, da un insieme Q di interrogazioni e da funzione di ranking ($r : Q \times D \rightarrow R$) che assegna ad ogni coppia formata da un'interrogazione e un documento un numero reale. L'idea è che a fronte di un'interrogazione a ogni documento venga assegnato un punteggio reale: i documenti con punteggio nullo non sono considerati rilevanti, mentre quelli a punteggio non nullo sono tanto più rilevanti quanto più il loro punteggio è alto. In particolare i metodi di ranking si dividono in *endogeni* ed *esogeni*. I primi metodi fanno uso del contenuto del documento per valutarne la rilevanza mentre i secondi fanno uso di un struttura esterna, ad esempio il grafo composto dai collegamenti ipertestuali tra le pagine web; questo non implica che i metodi esogeni non possano fare uso del contenuto della pagina (per esempio il testo delle ancore). I criteri si dividono ulteriormente in statici (o indipendenti dall'interrogazione) e dinamici (o dipendenti dall'interrogazione). Nel primo caso il punteggio assegnato a ciascun documento è fisso e indipendente da un'interrogazione q mentre nel secondo il punteggio assegnato a ciascun documento e dipendente da un'interrogazione q.

Tra i metodi endogeni sono di maggiore importanza *tf-idf* e *BM25* mentre tra quelli esogeni i più diffusi in letteratura sono *PageRank* e *HITS*.

1.1.1 Metodi di ranking endogeno

I metodi di ranking endogeno utilizzano il contenuto di una pagina per assegnarle un punteggio. Possono essere anch'essi statici o dinamici (cioè dipendere o meno da un'interrogazione). L'algoritmo usato dai motori di ricerca per fare il rank delle pagine web basandosi sui campi di testo usa varie forme del *tf-idf*. Il *tf-idf* è un metodo di ranking endogeno dinamico che utilizza il contenuto di una pagina per assegnarle un punteggio. Il *tf-idf* è composto da due misure più semplici: la *Term Frequency* e la *Inverse Document Frequency*. Il primo metodo assegna a un documento d il punteggio dato dalla somma dei conteggi dei termini t dell'interrogazione che compaiono nel documento stesso. In questo modo documenti in cui i termini dell'interrogazione compaiono più frequentemente avranno un punteggio più elevato. Utilizzare solo questo metodo non conviene in quanto è facilmente manipolabile. Inoltre non tiene conto del fatto che alcuni termini occorrono più frequentemente non perché rilevanti, ma perché altamente frequenti all'interno di *ogni* documento (ad esempio le congiunzioni). Il secondo metodo è definito come l'inverso del numero di documenti nella collezione che contengono il termine t [7]. Più precisamente:

$$idf_t = \log \frac{N}{df_t} \tag{1.1}$$

dove N è il numero totale di documenti nella collezione e df_t il numero di documenti nella collezione dove il termine t occorre. La combinazione del *tf* ed dell'*idf* produce una misura composta che permette di normalizzare il peso dei termini. Il *tf-idf* di un documento d rispetto a una query q è calcolato su tutti i termini t in comune come:

$$tf - idf(d, q) = \sum_{t \in d \; and \; t \in q} tf(t, d) \cdot idf(t) \tag{1.2}$$

Con il *tf-idf* gli spammer possono avere due obbiettivi: o creare pagine rilevanti per un gran numero di query o creare pagine molto rilevanti per una specifica query. Il primo obbiettivo può essere ottenuto includendo un gran numero di termini distinti in un documento; il secondo, attraverso la ripetizione di determinati termini nel documento.

Un altro metodo di ranking endogeno è *BM25* [8], basato sul *modello probabilistico*.

1.1.2 Metodi di ranking esogeno

Uno dei metodi esogeni statici è *PageRank* descritto in [9]. PageRank usa le informazioni portate dai link in entrata (*inlink*) per determinare un punteggio globale di importanza di una pagina. Esso assume che esista un legame tra il numero di *inlink* di una pagina p e la popolarità della pagina p. L'importanza di una pagina web p, utilizzando *PageRank*, è legata al numero di pagine web che puntano ad essa e all'importanza di tali pagine web. Questo concetto è mutualmente rinforzante ovvero l'importanza di una certa pagina influenza ed è influenzata dall'importanza delle altre pagine [10].

PageRank è basato sulla passeggiata naturale del grafo del web G. Più precisamente, la passeggiata viene perturbata nel seguente modo: fissato un parametro α tra 0 e 1, a ogni passo con probabilità α si segue un arco uscente, e con probabilità $1 - \alpha$ si sceglie un qualunque altro nodo del grafo utilizzando una qualche distribuzione v, detta vettore di preferenza (per esempio, uniforme). Assumendo che non esistano pozzi, la matrice di transizione della catena è quindi rappresentata dalla combinazione lineare:

$$\alpha G + (1 - \alpha) 1 v^T \tag{1.3}$$

dove G è la matrice della passeggiata naturale su G. Il fattore α è detto fattore di attenuazione di norma è impostato a un valore di 0,85.

Un altro metodo esogeno usato per il ranking delle pagine è *HITS (Hyperlink Induced Topic Distillation)* introdotto in [11]. Differentemente da *PageRank* esso assegna due punteggi di importanza a ogni pagina: uno di *hubbines* e uno di *autorevolezza*. L'intuizione dietro a HITS è che invece di un singolo punteggio di importanza esista un concetto di pagina *autorevole*, cioè pagina con contenuto pertinente e interessante, e di *hub*, cioè pagina contenente numerosi collegamenti a pagine autorevoli. I due concetti si rinforzano *mutuamente*: una pagina autorevole è pun-

tata da molte pagine centrali, e una buona pagina centrale punta a molte pagine autorevoli.

Questo approccio considera che nel web ci sono due tipi di pagine: quelle che contengono dei contenuti per un determinato argomento (*authoritative*) e quelle che contengono tanti link a delle pagine *authoritative* che sono chiamate pagine *hub*. Le pagine *hub* sono utili per scoprire le pagine *authoritative* [12].

L'algoritmo lavora su un sottografo del web ottenuto a partire da un'interrogazione. La selezione del sottografo può essere fatta in vari modi, un modo è quello di prendere un certo insieme di risultati ottenuto da un motore di base e generare un sottografo sulla base di una query e delle pagine che puntano a quelle ottenute dalla query. Per questo sottoinsieme di pagine otteniamo un matrice di adiacenza A. I punteggi di *hub* e *authority* per tutte le pagine del sottoinsieme possono essere formalizzate dalla seguente coppia di equazioni:

$$\begin{cases} \vec{a_{t+1}} = A^T \vec{h_t} \\ \vec{h_{t+1}} = A \vec{a_{t+1}} \end{cases} \tag{1.4}$$

Può essere dimostrato che la soluzione ottenuta applicando iterativamente il sistema 1.4 converge rispettivamente al principale autovettore di AA^T e A^TA [12][5].

1.2 Web spam

Con il termine web spamming si fa riferimento a tutti i metodi che tentano di manipolare gli algoritmi di ranking dei motori di ricerca per aumentare il valore di alcune pagine rispetto ad altre [10]. Dato il numero esorbitante di pagine che vengono create e pubblicate sul web, gli utenti competono per far comparire le proprie pagine tra le prime dei risultati di una query. Il fenomeno dello spamming o spamindexing ricade sulla qualità delle ricerche causando diversi problemi: indicizzazione di pagine che non sono utili, aumento del costo delle operazioni di query, malware e reindirizzamento verso contenuto per adulti; inoltre questo spinge gli utenti ad utilizzare altri motori di ricerca [5].

L'obbiettivo del motori di ricerca è di ottenere ottimi risultati per identificare tutte le pagine web che sono rilevanti per una specifica query e presentarle secondo

l'importanza che esse hanno. Di norma la rilevanza viene misurata attraverso la similarità testuale tra la query e le pagine mentre l'importanza è definita come la popolarità globale della pagina e a volte è inferita dalla struttura dei link [10]. Ci sono due categorie di tecniche associate al web spam [10]:

- tecniche boost che cercano di far avere più importanza o rilevanza a delle pagine
- tecniche hiding che sono metodi per nascondere le tecniche di boost all'utente dal browser, anche se alcuni autori incorporano queste tecniche fra quelle di boost.

1.2.1 Tecniche di boost

Le tecniche di boosting si dividono in: *Term Spamming* e *Link Spamming*. Con l'avvento degli algoritmi di ranking basati sulla struttura del grafo il *Term Spamming* è stato trascurato. In figura 1.1 è rappresentata una possibile tassonomia delle tecniche boost [10].

1.2.2 Term Spamming

Nel valutare la rilevanza testuale i motori di ricerca considerano dove i termini di una query compaiono in una pagina. Il tipo di punto all'interno della pagina è chiamato *campo*. I più comuni campi di testo per una pagina p sono: il body della pagina, il titolo, i meta tag nell'header HTML e l'URL della pagina. Inoltre viene considerato anche come *campo*, il testo delle ancore (il tag a) associate all'URL che puntano alla pagina p dato che descrive molto bene il contenuto della pagina . I campi di testo di p sono utilizzati per determinare la rilevanza di p rispetto ad una query (alcune volte i campi vengono pesati sulla base della loro importanza) e perciò chi fa *term spamming* utilizza tecniche di pesatura dei contenuti dei campi di testo in modo tale da aumentare l'efficacia dello spam [10]. Le tecniche di spamming possono essere raggruppate in base ai *campi* di testo dove viene fatto spamming. In base a questo distinguiamo [10]:

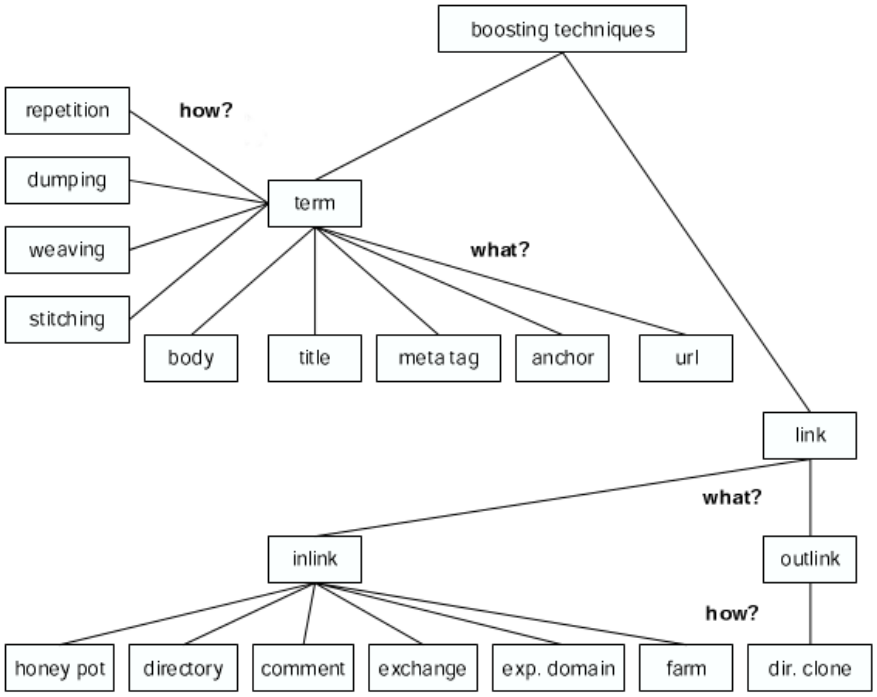

Figura 1.1: Tassonomia delle tecniche boost

- *Body Spam.* In questo caso lo spam è nel corpo del documento. Questo è lo spam più diffuso.

- *Title Spam.* Molti motori di ricerca danno molta importanza ai termini che compaiono nel titolo. Quindi ha senso includere termini di spam all'interno del titolo della pagina.

- *Meta Tag Spam.* I tag che compaiono nell'header sono molto frequentemente soggetti a spam. Per questo i motori di ricerca danno poca importanza a questi campi o non li considerano. Di seguito viene mostrato un esempio di questo tipo di spam.

```
<meta name="keyword" content="buy, cheap, cameras, lens,
  accessories, nikon, canon">
```

- *Anchor Text Spam.* I motori di ricerca assegnano un peso maggiore al testo nelle ancore perché pensano che esse contengano un riassunto del contenuto

della pagina. Perciò del testo di spam è incluso nel testo delle ancore dei collegamenti HTML di una pagina. In questo caso lo spamming non viene fatto sulla pagina cui si vuole far avere un rank più alto ma sulle pagine che puntano ad essa.

```
<a href="target.html">free, great deals, cheap,
  inexpensive, cheap, free</a>
```

- *URL Spam.* Alcuni motori di ricerca dividono l'URL delle pagine in un insieme di termini che sono usati per determinare la rilevanza di una pagina. Per sfruttare questo metodo di ranking, gli spammer creano lunghi URL che includono una grande sequenza di termini spam, un esempio può essere: *buy-canon-rebel-20d-lens-case.camerasx.com.*

Queste tecniche possono essere utilizzate insieme o separatamente. Un altro modo per raggruppare queste tecniche si basa sul tipo di termini che vengono utilizzati nei campi di testo [10], possiamo avere:

- Ripetizione di uno o più specifici termini.
- Inclusione di molti termini generici per creare pagine rilevanti per molte query.
- Intreccio di vari termini all'interno della pagina.
- Creazione di frasi di senso compiuto per l'elaborazione di contenuti generati velocemente attraverso la concatenazione di frasi da fonti diverse.

1.2.3 Link Spamming

Il *link spamming* è un tipo di spam che fa uso della struttura dei link tra le pagine web per favorire il rank di una pagina target *t*. Come descritto in [10], per uno spammer ci sono tre tipi di pagine web: inaccessibili, accessibili (blog) e proprietarie (fig. 1.2). Le inaccessibili sono quelle che uno spammer non può modificare. Le accessibili sono pagine gestite da altri ma che possono essere modificate lievemente dallo spammer attraverso l'immissione di un post in un forum, in un blog o in

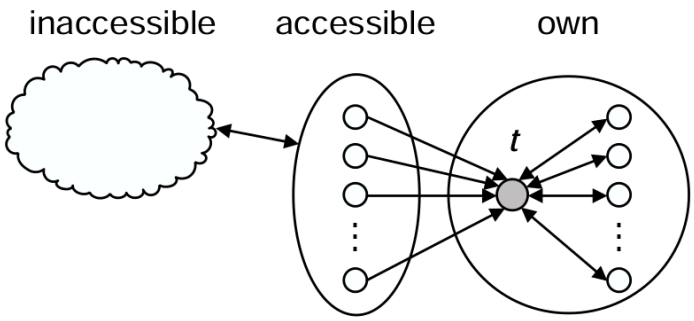

Figura 1.2: Tipi di pagine nel web per uno spammer

portali di questo genere. Le proprietarie sono pagine su cui gli spammer hanno il pieno controllo. Il gruppo di pagine proprietarie è chiamato *spam farm*.

Molti motori di ricerca utilizzano due algoritmi per aumentare l'importanza basandosi sulle informazioni dei link: PageRank e HITS; sulla base di questi due tipi di algoritmi vengono definite due categorie principali di *link spamming*: *outgoing link spam* e *incomign link spam*. L'*outgoing link* è uno dei metodi più facili da implementare in quanto basta aggiungere alla propria pagina dei link ad altre pagine, che sono considerate buone, al fine di aumentare il punteggio di *hub*. Per la ricerca di link da includere nella pagina per cui si vuole incrementare il punteggio di *hub* si possono utilizzare delle directory che contengono liste di siti come DMOZ o Yahoo!. Queste directory organizzano i contenuti web in contenuti e in liste di siti relativi. Per quanto riguarda *incoming link*, ci sono diverse strategie che si possono adottare in modo tale da avere un numero elevato di link in entrata [10]:

- *Honeypot*: consiste nella creazione di un insieme di pagine aventi un contenuto interessante (per esempio una documentazione Linux) ma che contengono link nascosti alla pagina o alle pagine di cui si deve aumentare il valore di rilevanza.
- *Infiltrarsi in una directory web*: tecninca con cui i webmaster inseriscono nelle directory web link ai loro siti.
- *Postare link nei blog, forum e wiki*: consiste nell'includere URL a pagine di spam come parte di un commento.

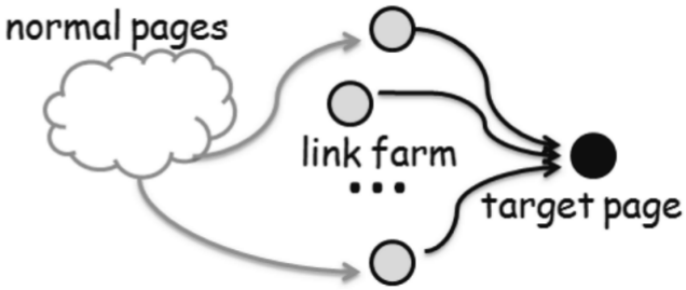

Figura 1.3: Esempio di una spamfarm

- *Scambio di link*: tecnica che consiste nello scambiare link con altre pagine di spam. Rappresenta una pratica comune tra chi fa spam ed infatti esistono blog completamente finalizzati all'incontro di spammer per lo scambio dei link.

- *Comprare domini scaduti*: tecnica che consiste nell'acquisto di domini scaduti e nella manipolazione delle pagine che puntano ancora ad esso al fine di aumentare il rank di una pagina target.

- *Creare una spam farm*: consiste nella creazione di un insieme di pagine che puntano ad una pagina spam, detta *target page*, e che hanno come obbiettivo l'aumento della rilevanza di quest'ultima. In questo caso il valore di page rank aggregato delle pagine è propagato alla pagina target. Molte volte tale tecnica si integra con quella *honeypot*. Una delle forme più aggressive di integrazione honeypot - spamfarm è l'*hijacking* [5], dove gli spammer prima attaccano un sito con una buona reputabilità e poi usano questo come parte della loro link farm.

1.2.4 Tecniche di hiding

Le tecniche di hiding si possono classificare in: *content hiding*, *cloaking*, *redirection* (fig. 1.4) [10]. Nel *Content hiding* i termini o i link di spam vengono nascosti quando il browser visualizza una pagina, ad esempio utilizzando per i termini lo stesso colore dello sfondo e per i link non inserendo il testo all'interno delle ancore che indirizzano ad una pagina. Un'altra tecnica è quella di utilizzare degli script

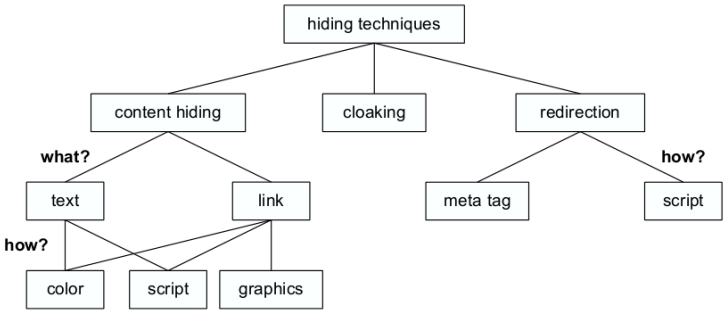

Figura 1.4: Tecniche di hiding

per nascondere il contenuto. Il *Cloaking* sfrutta la possibilita di identificare se la richiesta di una pagina è fatta da un crawler o da un browser: dato un URL, il server spam restituisce un documento HTML diverso a seconda che la richiesta venga fatta da un crawler o da un browser. Quindi vengono distribuiti due contenuti diversi in base alla parovenienza della richiesta al server spam. La rilevazione di un crawler può essere effettuata in due modi: o si mantiene in memoria una lista di indirizzi di crawler oppure si usa l'header della richiesta HTTP, controllando il campo user-agent e verificando che sia diverso dai più comuni browser (si iportizza quindi che sia un crawler). Nell'esempio sotto, lo user-agent della richiesta HTML indica l'uso del web browser Chrome.

```
Mozilla/5.0 (X11; Linux x86_64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/32.0.1700.102 Safari/537.36
```

La *Redirection* è un'altra tecnica che reindirizza il browser ad un altro URL appena la pagina è caricata. Un esempio di redirection server-side è mostrato di seguito.

```
header("Location: http://www.example.com/");
```

1.2.5 Click Spamming

Un ultimo metodo per fare web spam è il *Click Spamming* [5]. I motori di ricerca utilizzano dati sul flusso dei click per regolare le funzioni di ranking, quindi gli

spammer generano clik fraudolenti per manipolare il comportamento di queste funzioni in modo tale da fare avere un rank migliore ai loro siti. Il metodo prevede che vengano fatte delle query e si clicchi sulla pagina di cui si vuole aumentare il rank. Tale metodo viene eseguito in modo automatico attraverso script che girano su diverse macchine per non fare sospettare il motore di ricerca delle numerose richieste provenienti da un unica macchina [5].

Capitolo 2

Tecniche basate sul contenuto

Il capitolo illustra le tecniche di spam detection presenti in letteratura basate sul contenuto. Nella prima parte del capitolo verranno illustrate le prime tecniche basate sul contenuto che sono state sviluppate, nella seconda parte menrte verrano presentate le tecniche che fanno del contenuto della pagina in modo più soffisticato.

2.1 Prime feature per identificare lo spam

Un metodo per identificare lo spam basandosi sul contenuto di una pagina web è quello di analizzare alcune proprietà (feature) delle pagine spam e confrontarle con le medesime proprietà di quelle non spam al fine di ottenere dei valori con cui stimare la natura della pagina web (spam o non spam). Alcune di queste proprietà, come descritto in [13], sono le seguenti:

- *Proprietà degli URL*: alcune analisi sulle proprietà dei link mostrano che gli URL di un host sono delle buone feature per identificare lo spam. In particolare l'URL di un host con molti caratteri, punti, slash e numeri è un buon indicatore di spam. Un modo semplice per classificare le pagine è quindi quello di usare un valore di soglia che definisca il numero massimo di tali caratteri per identificare una pagina come spam.

- *Host name resolution*: gli spammer possono popolare gli URL delle pagine spam con termini contenuti in query molto frequenti, che sono rilevanti per

un certo settore, e impostare un DNS per risolvere questi host name. Gli spammer cercano di manipolare il meccanismo usato da alcuni motori di ricerca (ad esempio Google) che data una query q, assegnano un rank più alto a un URL u se i termini che compongono il nome dell'host di u combaciano con i termini della query. Perciò gli spammer creano tanti URL rilevanti per le diverse query in modo tale da far risultare la pagina di web spam tra i primi risultati di ricerca per molte query. Per determinare tale forma di spam, uno spam detector dovrebbe controllare quanti URL vengono risolti da uno stesso indirizzo IP.

- *Proprietà del contenuto*: le pagine generate automaticamente hanno tutte lo stesso template, ad esempio numerosi siti di spam generano dinamicamente pagine con uno stesso numero di parole. Una tecnica per determinare lo spam è quella di clusterizzare le pagine in base alla somiglianza dei template. Considerando che le pagine di spam hanno strutture molto simili tra loro, se si identificano gruppi con molte pagine aventi la stessa struttura è probabile che esse siano spam.

Oltre a queste proprietà base, in [1] vengono descritti ulteriori metodi e proprietà per l'individuazione dello spam. In tale studio i metodi e le proprietà descritti sono il frutto di analisi effettuate dagli autori su un dataset di pagine HTML , che è stato ricavato utilizzando MSN Search crawler nell'agosto del 2004. Dalle analisi su tale dataset risulta che i domini con maggiore contenuto di spam sono: ".biz", ".us" e ".com" mentre le pagine contenenti più spam sono: francesi, tedesche e inglesi. I risultati sono rappresentati nei due grafici in figura 2.1 e in figura 2.2. Nel primo grafico l'asse orizzontale rappresenta i domini e l'asse verticale mostra la frazione di spam all'interno di un dominio. I valori nel grafico sono riportati con un intervallo di confidenza del 95% rappresentato dalla linea verticale sopra ogni barra. L'intervallo di confidenza varia in dimensione a causa del numero differente di campioni raffigurati nei deiversi domini. Nel secondo grafico l'asse orizzontale rappresenta la lingua della pagina e l'asse verticale rappresenta la frazione di spam

per le pagine scritte in una particolare lingua. Anche nel secondo grafico i valori sono riportati con un intervallo di confidenza del 95%.

Una proprietà che consente di identificare una pagina spam è il numero di parole all'interno della pagine stessa. Infatti una pratica molto comune nel costruire pagine spam è la cosiddetta "Keyword stuffing", un processo per cui al contenuto della pagina vengono aggiunte molte parole popolari che sono, tuttavia, irrilevanti rispetto ai contenuti; lo scopo è di incrementare le probabilità della pagina di essere in cima ai risultati di molteplici query. In figura 2.3 viene plottato la distribuzione del numero di parole per ogni pagina del dataset (rappresentata dall'istogramma blu) correlata con la probabilità che una pagina sia spam (rappresentata dalla linea viola). Dal grafico si nota che la prevalenza di spam è più alta per le pagine contenenti molte parole. Perciò c'è una correlazione tra prevalenza di spam e numero di parole. Ma si può notare anche che il conteggio delle parole da solo non è una buona euristica visto che porta un alto tasso di falsi positivi.

La tecnica "Keyword stuffing" viene utilizzata anche per la scelta dei titoli, tenendo in considerazione che alcuni motori di ricerca assegnano un peso maggiore ai termini della query presenti all'interno del titolo della pagina. Il grafico in figura 2.4 rappresenta la distribuzione del numero di parole all'interno dei titoli delle pagine correlata con la probabilità che una pagina sia spam. Dal grafico si evince che un eccesso di parole all'interno del titolo è un indicatore di spam. Le parole che vengono utilizzate nel processo di "keyword stuffing" vengono selezionate casualmente o da un ristretto gruppo di query comuni. Per esaminare il comportamento con cui sono selezionate e costruite le frasi innanzitutto vengono identificate le n parole più comuni all'interno del corpus; successivamente viene calcolata, per ogni pagina, la frazione delle parole comuni. Tale processo viene ripetuto per ogni scelta di n (in figura 2.5 n=*200*). Il grafico rappresenta la frazione di parole comuni per ogni pagina; esso ha una caratteristica gaussiana e suggerisce che la maggior parte delle pagine di spam sono generate tessendo parole da un dizionario con una scelta casuale. Un metodo che può essere adottato per identificare le pagine spam che sono generate automaticamente è descritto in [14].

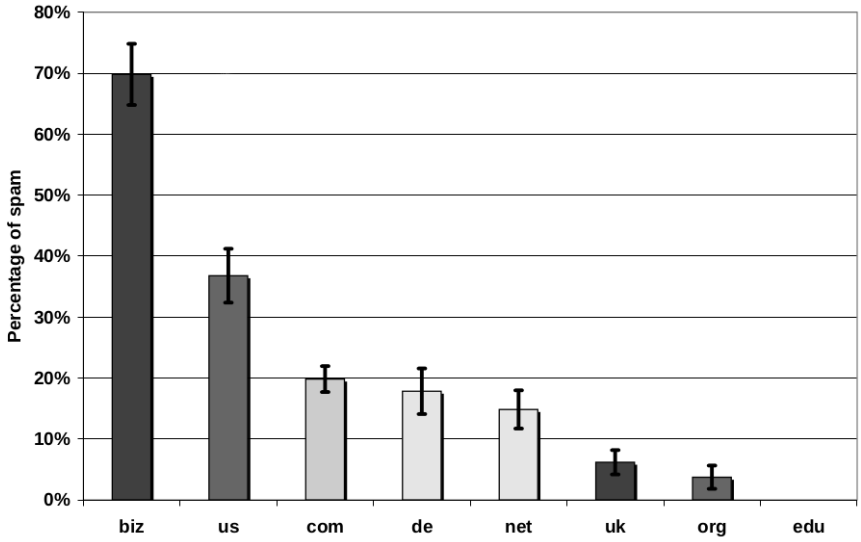

Figura 2.1: Occorrenze dello spam classificate per dominio all'interno del dataset descritto in [1]

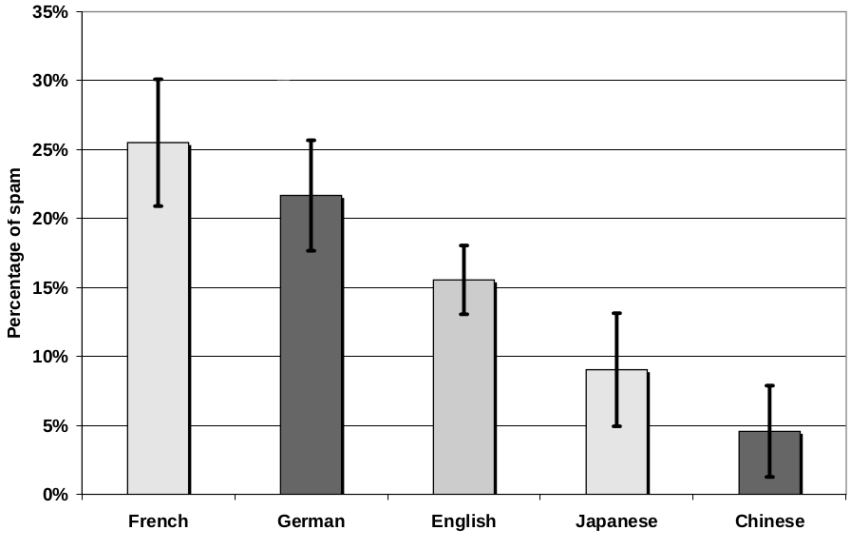

Figura 2.2: Occorrenze dello spam classificate per lingua all'interno del dataset descritto in [1]

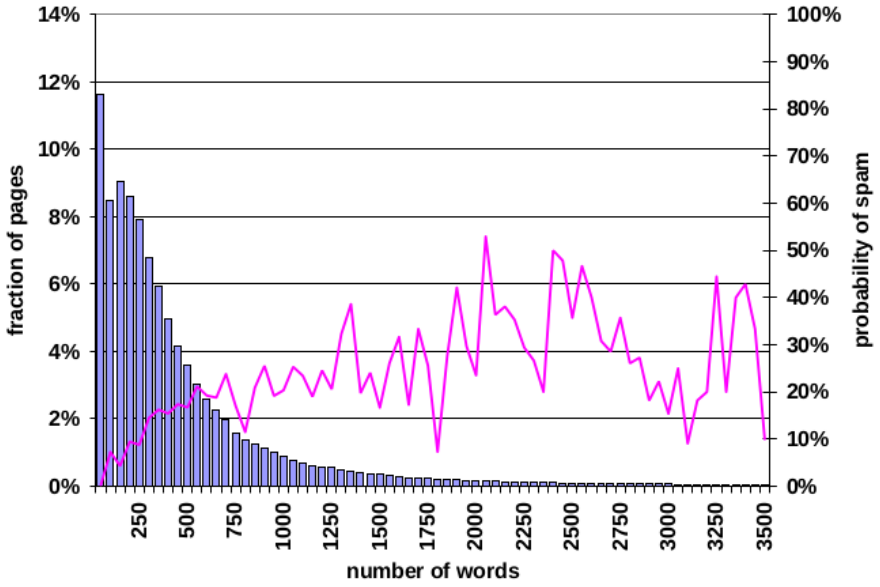

Figura 2.3: Prevalenza di spam sulla base del numero di parole per pagina

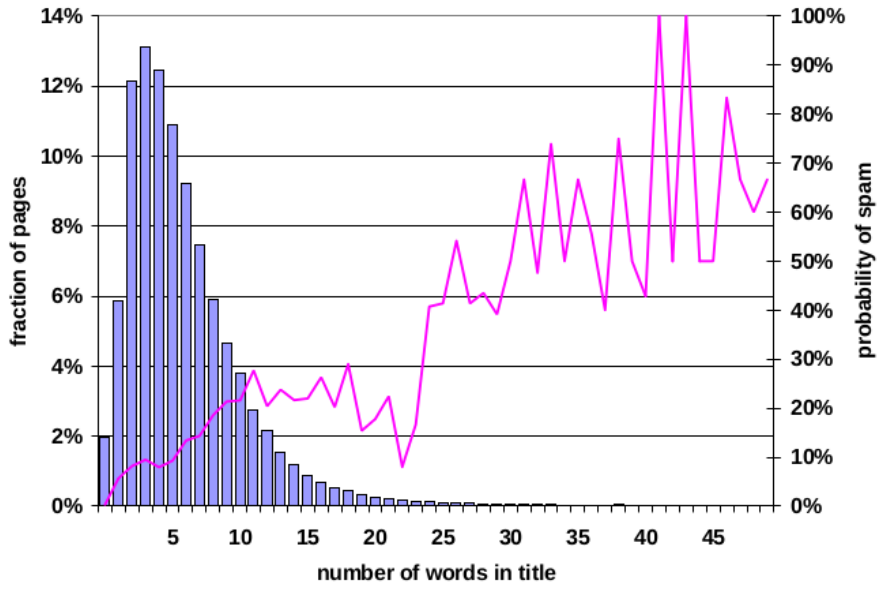

Figura 2.4: Prevalenza di spam sulla base del numero di parole all'interno dei titoli delle pagine

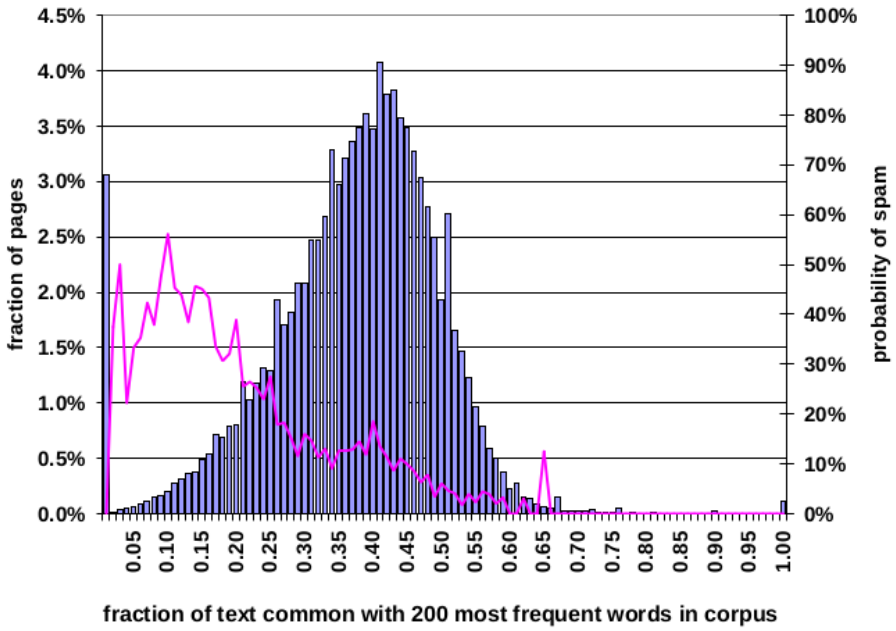

Figura 2.5: Prevalenza di spam sulla base della frazione di parole di una pagina che sono tra le 200 più frequenti parole nel corpus

Un altro dato utilizzato per determinare se una pagina è spam è la lunghezza media delle parole delle pagine. Dal dataset preso in considerazione in [1] (grafico in figura 2.6) si nota che la distribuzione della lunghezza media delle parole è simile a una gaussiana con moda e mediana corrispondenti a 5.0 e le parole con lunghezza media uguale a 10 sono certamente spam.

Un'altra proprietà delle pagine web che consente di stimare se una pagina è spam è la quantità di testo che è contenuta all'interno delle ancore (il tag "$< a >$" delle pagine web). Infatti una pratica comune dei motori di ricerca è considerare il testo delle ancore dei link in una pagina come annotazioni che descrivono il contenuto della pagina che viene puntata. L'idea principale è che se la pagina a ha un link alla pagina b con testo dell'ancora, ad esempio, "computer" allora potremmo concludere che b parli di computer, anche se questa keyword non compare all'interno della pagina b. Pertanto durante il ranking tali motori di ricerca potrebbero considerare la pagina b come risultato di una query contenente la keyword "computer". Sfruttando questo meccanismo si creano pagine di spam contenenti solo del testo all'interno delle

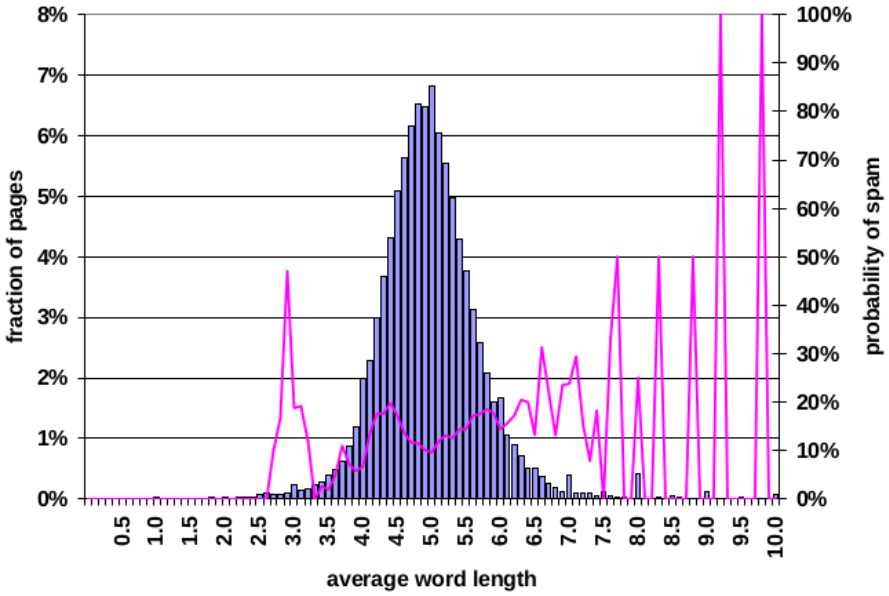

Figura 2.6: Prevalenza di spam sulla base della lunghezza media delle parole per pagina

ancore per valorizzare il ranking di altre pagine.; queste pagine di norma sono solo cataloghi di link ad altre pagine. Per capire meglio il fenomeno è stato calcolata la frazione di tutte le parole del testo delle ancore all'interno di una pagina, escludendo i markup rispetto al contenuto della pagina. In figura 2.7 viene visualizzato il grafico risultante. Si nota che un'alta frazione di testo delle ancore aumenta la probabilità che la pagina sia spam ma usare questa euristica da sola potrebbe portare un alto numero di falsi positivi [1].

Calcolando la frazione di contenuto visibile all'interno di una pagina (definita come la lunghezza in termini di byte di tutte le parole non di markup) rispetto all'intera dimensione della pagina si nota dal grafico, in figura 2.8, che la distribuzione (delle frazioni di contenuto visibile) evidenzia che le pagine di spam hanno meno markup delle pagine normali. Questo fa intendere che molte pagine spam hanno il solo scopo di dover essere indicizzate dai motori di ricerca e non di essere fruite da un utente.

Come detto in precedenza i motori di ricerca possono dare un peso maggiore a pagine che contengono ripetutamente le keyword contenute nella query (ad esempio utilizzano come metodo di ranking il "term-frequency"). Le pagine spam hanno,

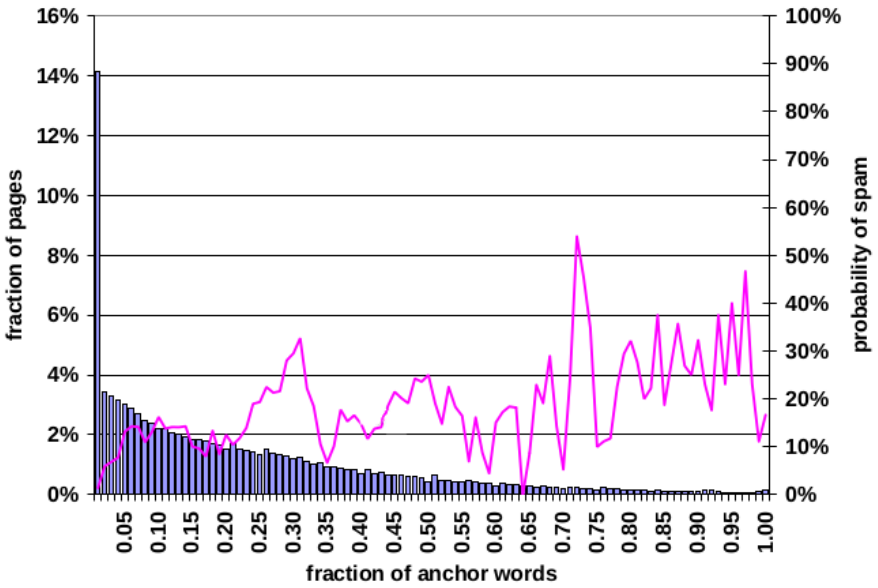

Figura 2.7: Prevalenza di spam sulla base della quantità di testo delle ancore delle pagine

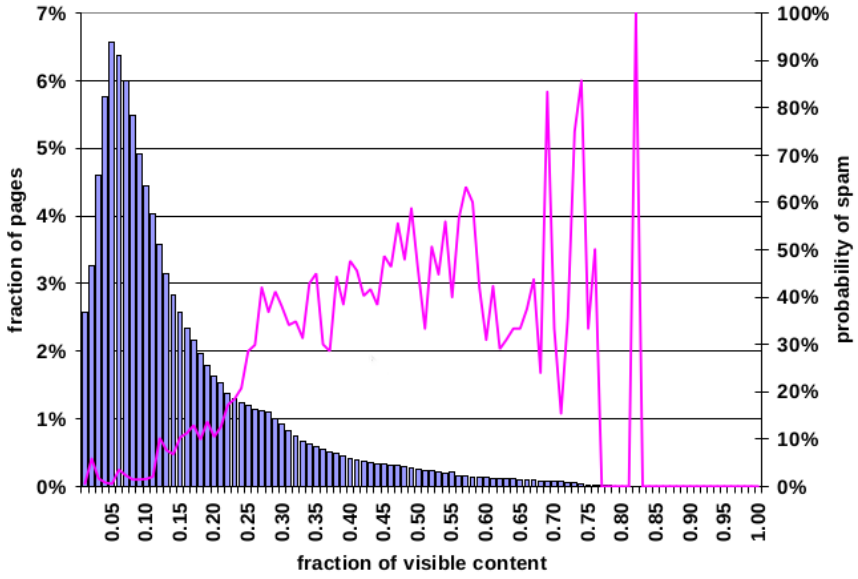

Figura 2.8: Prevalenza di spam sulla base della frazione di contenuto visibile

quindi, contenuti replicati molte volte per aumentare il rank. Per rilevare la ridondanza di contenuti (ottenuta dal processo di replica), viene calcolato il rapporto di compressione ovvero la dimensione della pagina non compressa divisa per la dimensione della pagina compressa. In figura 2.9 è rappresentata la distribuzione del rapporto di compressione e la likelihood che la pagina sia spam. Dal grafico si nota che quanto più il valore di compressione è elevato tanto più probabilmente la pagina può essere considerata spam; il 70 per cento delle pagine con un rapporto di compressione maggiore di 4.0 sono giudicate spam. Questo è dovuto al fatto che una compressione su una pagina spam che ha contenuti ridondanti, sarà più efficace di una compressione su una pagina non spam che è caratterizza da contenuti non ridondanti. E perciò il rapporto di compressione tenderà a crescere quanto più la pagina sarà caratterizzata da contenuti ridondanti.

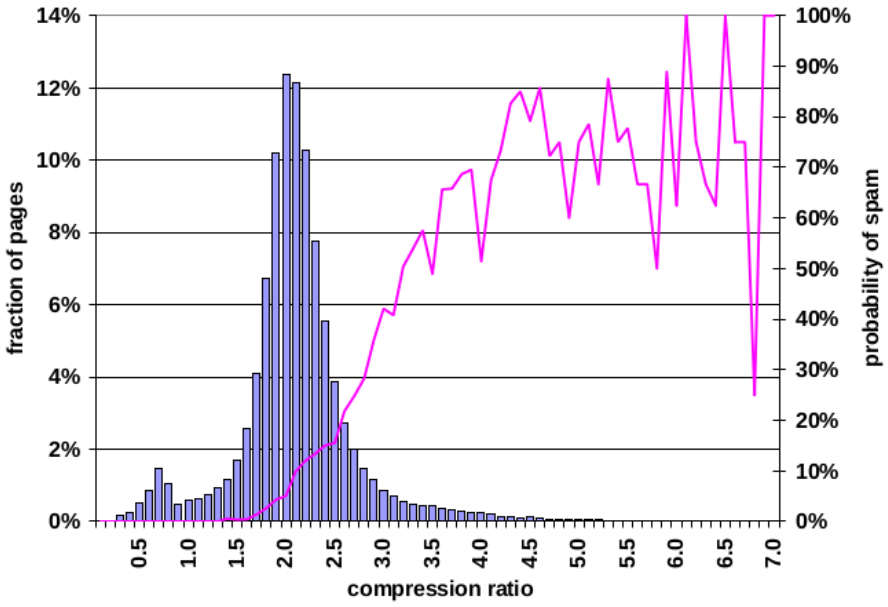

Figura 2.9: Prevalenza di spam sulla base del rapporto di compressione

2.1.1 Utilizzo di un classificatore per combinare le feature

Sempre in [1] le euristiche descritte fino a questo punto sono combinate considerando il problema di rilevamento dello spam come un problema di classificazione. Quindi

viene creato un classificatore, che usando le caratteristiche di una pagina web la classificherà in una delle due classi: spam o non spam. Costruire un classificatore richiede una fase di training durante la quale i parametri del classificatore sono determinati e una fase di testing durante la quale le performance del classificatore sono valutate. Per ogni pagina all'interno del dataset viene calcolato il valore di ogni feature (ad esempio, usando le euristiche discusse sopra) e si utilizzano questi valori per istruire il classificatore. Il classificatore usato è un albero di decisione che dato un insieme di dati da traning e un insieme di feature crea un diagramma di flusso ad albero. Ogni nodo dell'albero corrisponde al test da valutare per una particolare feature mentre ogni arco è un valore di uscita del test ed infine le foglie corrispondono alle classi che verrano assegnate alle pagine. Un esempio del classificatore è rappresentato in figura 2.10. Per migliorare il classificatore si possono usare tecniche come bagging o boosting. Queste tecniche creano un insieme di classificatori che vengono combinati.

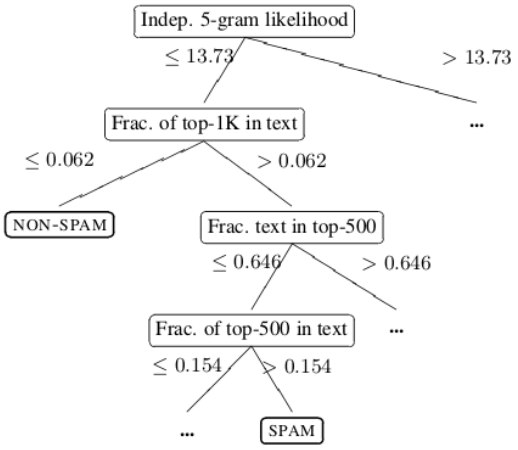

Figura 2.10: Esempio di classificatore

2.2 Language model per rilevare lo spam

Oltre alle feature presentate in precedenza, in [2] sono descritte nuove tipologie di feature e metodi per rilevare lo spam che fanno uso dei *language model* per analizzare

le sorgenti estratte da ogni sito in un dataset. Le sorgenti sono dei pezzi di contenuto presenti all'interno delle pagine web di partenza (quelle che contengono i link ad altre pagine) e all'interno delle pagine web di destinazione (quelle che sono linkate dalle pagine di partenza). Per ogni sorgente si definisce un modello e si calcola quanto siano differenti i modelli individuati; viene utilizzata la *Kullback-Leibler Divergence (KLD)* per misurare la divergenze tra le distribuzioni di probabilità dei termini di pagine web, applicandola a unità di testo della pagina di partenza e di quella linkata. La Kullback-Leibler (KL) è una misura asimmetrica della divergenza che misura quanto male una distribuzione di probabilità M_q riesce a modellare M_d:

$$KLD(T_1||T_2) = \sum_{t \in T_1} P_{T_1}(t) \log \frac{P_{T_1}(t)}{P_{T_2}(t)} \tag{2.1}$$

dove in 2.1 $P_{T_1}(t)$ è la probabilità del termine t nella prima unità di testo e $P_{t_2}(t)$ è la probabilità del termine t nella seconda unità di testo. In basso sono rappresentati due esempi di KLD applicata tra il testo delle ancore della pagina sorgente e i titoli delle pagine puntate dai link (esempio preso da WEBSPAM-UK2006). Dagli esempi si deduce che nel primo caso c'è una maggiore correlazione tra il testo dell'ancora della pagina sorgente e il titolo della pagina puntata rispetto al secondo esempio dove il valore di KLD e maggiore.

```
KLD(Free Ringtones || Free Ringtones for Your Mobile Phone from
   PremieRingtones.com) = 0.25


KLD(Best UK Reviews || Findabmw.co.uk - BMW Information
  Resource) = 3.89
```

Per determinare se una pagina è spam si cerca di identificare una relazione tra due pagine collegate sulla base del valore di divergenza. I valori sono ottenuti calcolando le divergenze con KLD tra una o più sorgenti di informazioni da ogni pagina. In particolare si usano tre tipi di informazione di una pagina sorgente: testo delle ancore, testo intorno alle ancore, termini nell'URL mentre per la pagina che viene linkata dalla pagina sorgente: titolo, contenuto della pagina e meta tag. Combinando queste sorgenti di informazione si determina la divergenza tra due pagine; di seguito vengono descritte alcune combinazioni base:

- testo delle ancore - contenuto. Una pagina che ha un link verso un'altra pagina ha solo un modo per convincere l'utente a visitare la pagina collegata: mostrare in maniera concisa le informazioni relative alla pagina collegata. Perciò una grande divergenza tra questi due frammenti di testo indica che la pagina può essere spam. In figura 2.11 è illustrata la divergenza KL tra le due sorgenti di informazione, come si vede la curva delle pagine normali è più compatta di quella delle pagine spam.

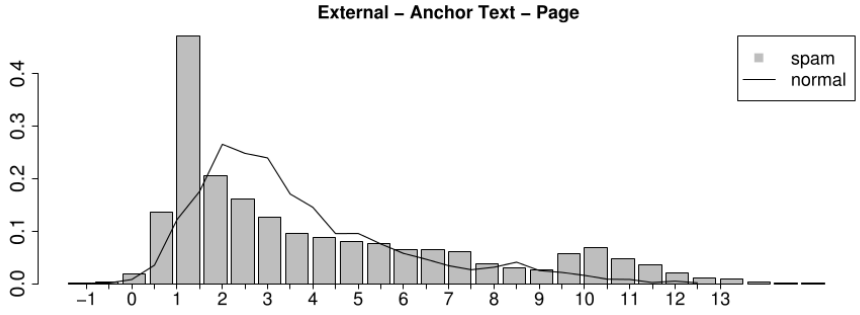

Figura 2.11: Istogramma della divergenza KL tra testo delle ancore e il contenuto della pagina puntata basato sul dataset WEBSPAM-UK2006 utilizzato in [2]

- testo vicino alle ancore - contenuto. Il testo delle ancore, a volte, è un valore poco descrittivo e per ovviare al problema viene usato il testo che circonda le ancore. Nell'esperimento vengono utilizzate 7 parole per lato. In figura 2.12 viene mostrato che le pagine spam hanno alti valori di divergenza mentre le normali sono concentrate intorno $KL \approx 2.5$, .

- termini nell'URL - contenuto. I motori di ricerca danno molta importanza agli URL perché i termini che li compongono possono dare un'idea del contenuto della pagina a cui l'URL fa riferimento. Un metodo di spam che sfrutta questo meccanismo consiste nella creazione di URL contenenti molti termini comuni (ad esempio: `www.domain.com/viagra-youtube-free-download-poker-online.html` che in realtà è solo uno store online e non ha alcuna attinenza con i nessuno dei termini utilizzati nell'URL). Al fine di identificare questo tipo di spam, vengono prelevati i termini più rilevanti da un URL e calcolata la divergenza col contenuto della pagina di destinazione. La distribuzione finale,

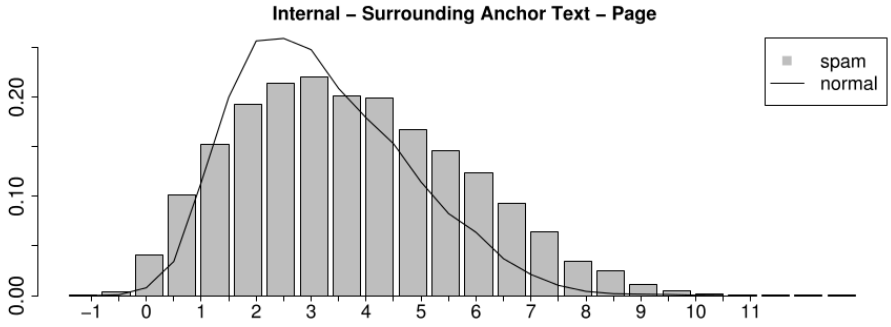

Figura 2.12: Istogramma della divergenza KL tra testo intorno alle ancore e il contenuto della pagina puntata basato sul dataset WEBSPAM-UK2006 utilizzato in [2]

rappresentata in figura 2.13, dimostra una grande differenza tra l'istogramma delle pagine normali con quello delle pagine spam.

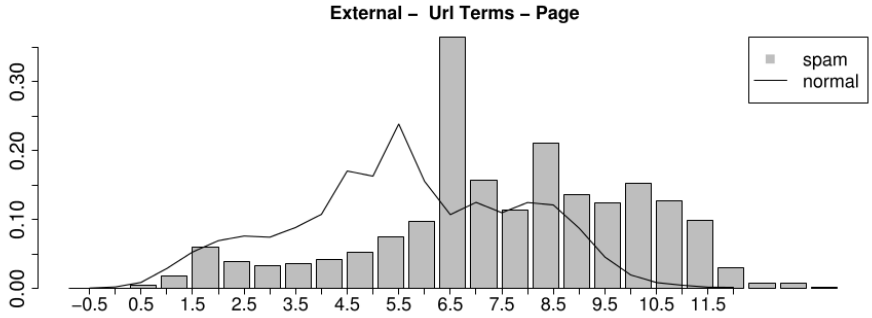

Figura 2.13: Istogramma della divergenza KL tra termini degli URL e il contenuto della pagina puntata basato sul dataset WEBSPAM-UK2007 utilizzato in [2]

- testo delle ancore - titolo. Questa feature si compone di due sorgenti che sono molto simili in quanto descrivono la pagina con poche parole. Tuttavia la prima sorgente può essere scritta anche da chi non è il proprietario della pagina di destinazione. In figura 2.14 notiamo che questa feature da sola non discrimina bene le pagine spam ma è abbastanza efficace se utilizzata in congiunzione con altre feature.

- testo intorno alle ancore - titolo. Dal grafico in figura 2.15 si può notare come tale feature rileva meglio lo spam rispetto alla precedente, infatti molti valori

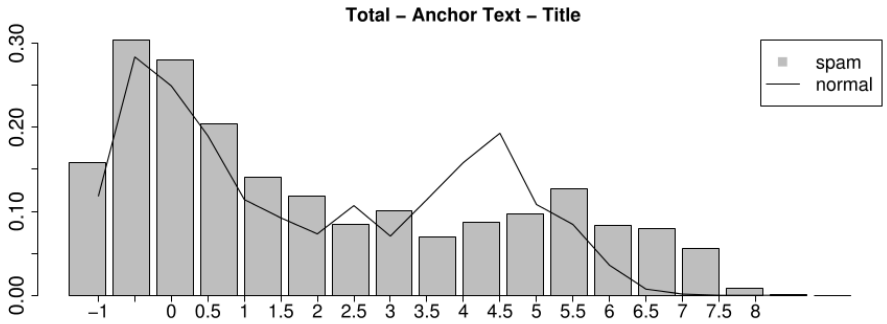

Figura 2.14: Istogramma della divergenza KL tra testo delle ancore e titolo della pagina puntata basato sul dataset WEBSPAM-UK2007 utilizzato in [2]

di spam sono concentrati per valori di $KL > 3$.

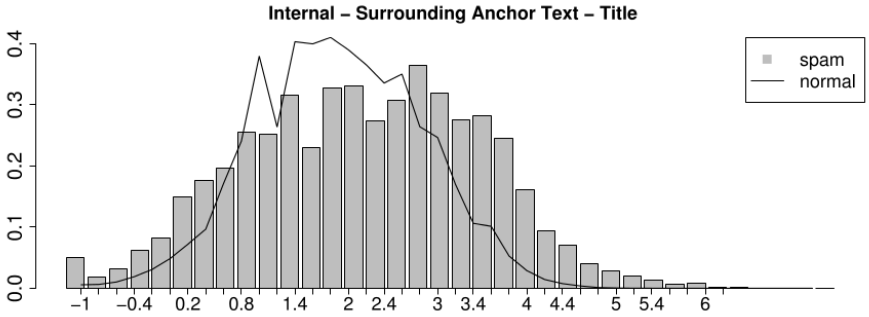

Figura 2.15: Istogramma della divergenza KL tra testo intorno alle ancore e titolo della pagina puntata basato sul dataset WEBSPAM-UK2006 utilizzato in [2]

- termini nell'URL - titolo. In questa feature entrambe le sorgenti sono generate dal proprietario della pagina, a differenza della precedente in cui la sorgente di informazione della pagina di partenza poteva essere generata da un'altra persona. In questo vi dovrebbe essere maggiore coerenza tra le due sorgenti. In figura 2.16 sono rappresentate le distribuzioni per le pagine spam e non spam.

- titolo - contenuto. I motori di ricerca danno un peso maggiore ai termini della query se presenti nel titolo della pagina. Sfruttando tale meccanismo gli spammer perfezionano i loro processi in modo tale da impostare termini chiave nel titolo anche creando però una divergenza tra titolo e contenuto

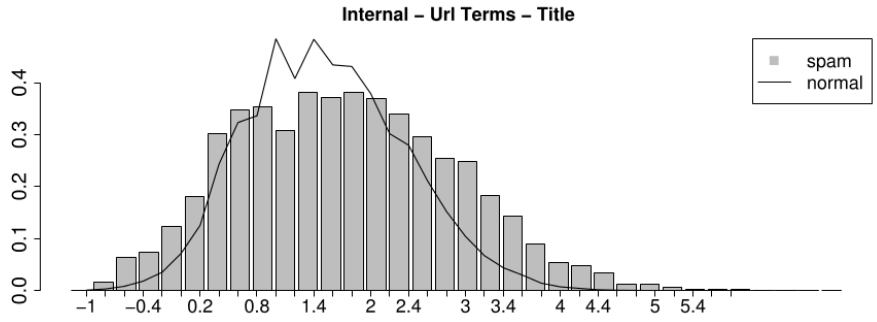

Figura 2.16: Istogramma della divergenza KL tra termini nell'URL e titolo della pagina puntata basato sul dataset WEBSPAM-UK2006 utilizzato in [2]

della pagina. La feature titolo - contenuto consente la rilevazione di spam quando non vi è alcuna relazione tra il titolo e il contenuto della pagina. In figura 2.17 è rappresentata la divergenza tra le due distribuzioni spam e non spam.

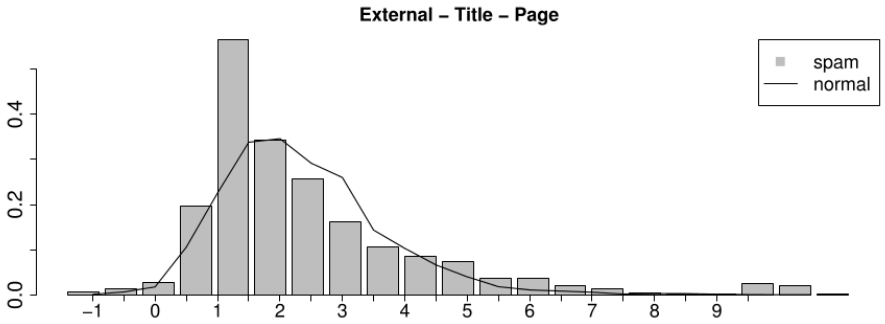

Figura 2.17: Istogramma della divergenza KL tra titolo e contenuto della pagina basato sul dataset WEBSPAM-UK2006 utilizzato in [2]

- metatag. Vengono usati per calcolarne la divergenza con altre sorgenti di informazioni della pagina di partenza (come il testo delle ancore e il testo intorno alle ancore) e della pagina destinazione (come il contenuto o i temini dell'URL). In figura 2.18 viene visualizzata la divergenza tra il testo delle ancore e i metatag.

Oltre alle feature descritte si possono ottenere delle feature più articolate combinando le feature base tra di loro; ad esempio per le pagine sorgenti si possono definire:

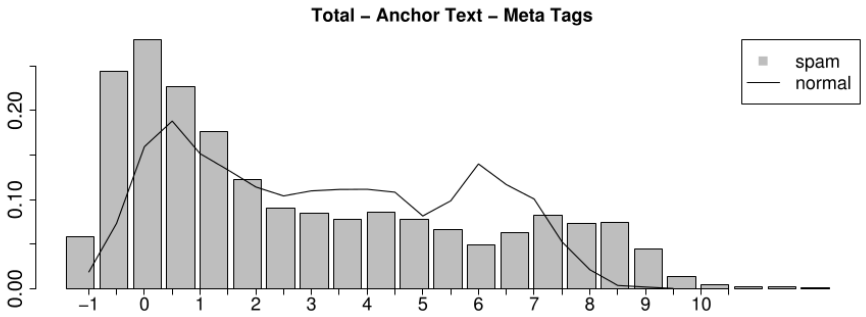

Figura 2.18: Istogramma della divergenza KL tra testo delle ancore e i meta tag della pagina basato sul dataset WEBSPAM-UK2006 utilizzato in [2]

testo delle ancore e termini nell'URL , testo intorno alle ancore e URL (per maggiori dettagli [2]).

Infine tali feature possono essere usate per istruire un classificatore.

2.3 Spam detection sulla base degli argomenti di una pagina web

Dong et al. [15] propongono un metodo basato su statistiche effettuate sugli argomenti delle pagine. Le statistiche non si basano sulle parole della pagina ma sugli argomenti, in modo da non ignorare la semantica delle parole e di catturare le feature linguistiche nascoste nel testo per capire se la pagina è spam. Le analisi vengono fatte usando i topic model, modelli statistici che scoprono gli argomenti latenti presenti in una collezione di documenti. Un topic model utilizzato è la *Latent Dirichlet Allocation (LDA)* che modella ogni argomento latente come una distribuzione probabilistica su un vocabolario e ogni documento come una distribuzione probabilistica sugli argomenti latenti. L'intuizione di usare i topic model per analizzare il contenuto delle pagine web nasce dal fatto che analizzando le feature nascoste di una pagina spam, gli autori hanno notato che i contenuti di queste pagine, generati automaticamente, sono differenti dai contenuti delle pagine non spam. Vengono definiti tre tipi di misure.

La prima misura per determinare le pagine spam utilizzando LDA sfrutta la

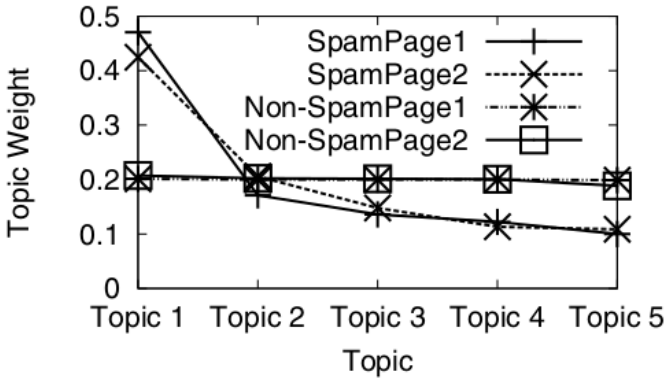

Figura 2.19: Distribuzione degli argomenti pesati per pagine spam e normali

caratteristica che tali pagine sono molto topic-centric, ovvero hanno uno specifico insieme di argomenti. In figura 2.19 sono rappresentate quattro distribuzioni degli argomenti presenti in quattro pagine (due spam e due non spam) scelte casualmente da un dataset. A supporto della tesi degli autori che le pagine spam sono topic centric si può notare che presentano una distribuzione esponenziale, in quanto esse vengono sviluppate per avere un alto ranking per un insieme di specifiche query di ricerca, al contrario delle pagine non spam che presentano una distribuzione uniforme (nell'articolo gli autori sostengono che le pagine non spam come una homepage contengono vari argomenti come ad esempio: i contatti, chi e cosa fa, altre informazioni). Per classificare le pagine sulla base di questa caratteristica, basata sulle distribuzioni degli argomenti, è stato proposto una misura della diversità degli argomenti basata sulla varianza. Data una pagina web d, la sua distribuzione degli argomenti è $T(d) = \{t_1, t_2, ..., t_m\}$, dove ogni argomento $t_i (1 \leq i \leq m)$ è associato con un peso δ_{t_i}. La misura della diversità degli argomenti basata sulla varianza per d, denotata con $TopicVar(d)$, è calcolata come:

$$TopicVar(d) = \frac{\sum_{i=1}^{m} (\delta_{t_i} - u)^2}{m} \tag{2.2}$$

dove $u = \frac{\sum_{i=1}^{m} \delta t_i}{m} = \frac{1}{m}$. In figura 2.20 è illustrata la distribuzione risultante. Dalla distribuzione si nota che le pagine spam sono più topic-centric ovvero la varianza dei pesi degli argomenti è piu grande rispetto alle pagine non spam. Questo è dovuto al fatto che le pagine non spam avendo una distribuzione quasi uniforme hanno una

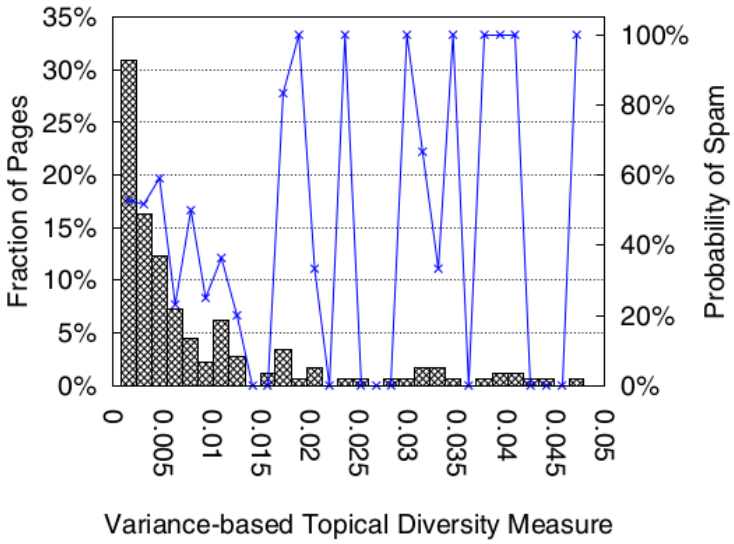

Figura 2.20: Prevalenza di spam relativa alla misura della diversità degli argomenti basata sulla varianza

la varianza che è più piccola rispetto alle pagine spam che hanno una distribuzione più concentrata di quella uniforme, come quella esponenziale. Il valore di *TopicVar* è proporzionale alla probabilità che una pagina sia spam, ovvero all'aumento della probabilità di una pagina di essere spam consegue un aumento del valore di *TopicVar* per quella pagina. Questa misura è un ottimo indicatore per rilevare lo spam.

La seconda misura per identificare le pagine spam utilizzando LDA che permette di misurare la relazione semantica tra gli argomenti è la semantica delle parole. Gli autori definiscono che: data una pagina web d, la sua distribuzione degli argomenti è $T(d) = \{t_1, t_2, ..., t_m\}$. La probabilità che un parola w appartenga a un argomento $t_i (1 \leq i \leq m)$ è definita come $\phi(w|t_i)$. Ogni argomento t_i è rappresentato come un insieme di parole denotate come $W(t_i)$. Intuitivamente due argomenti $t_i, t_j (1 \leq i, j \leq m)$ sono semanticamente correlati se le parole $W(t_i)$ e $W(t_j)$ sono semanticamente legate. In [15] per ottenere le relazioni semantiche tra le due parole è utilizzata una funzione di similarità $Sim(w_i, w_j)$ che è quella di Wordnet. Quindi per misurare la relazione semantica tra due argomenti t_i, t_j, si calcolano le similarità tra ogni coppia di parole dei due argomenti moltiplicate con le loro probabilità

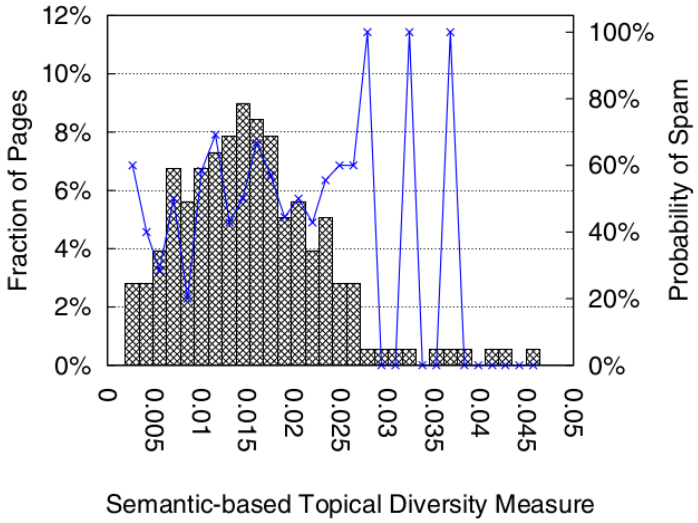

Figura 2.21: Prevalenza di spam relativa alla misura di diversità degli argomenti basata sulla semantica

rispetto agli argomenti.

$$Sim(t_i, t_j) = \frac{\sum_{w_k \in W(t_i), w_l \in W(t_j)} Sim(w_k, w_l) X \phi(w_k|t_i) X \phi(w_l|t_j)}{\frac{|W(t_i)|X|W(t_j)|}{2}} \tag{2.3}$$

Usando quindi un modello degli argomenti otteniamo m argomenti latenti. Dall'equazione 2.3 si deriva una misura della diversità degli argomenti basata sulla semantica per gli m argomenti latenti di una collezione [15]: data una pagina web d, la sua distribuzione degli argomenti è $T(d)$ e quindi la misura della diversità degli argomenti basata sulla semantica per tale pagina d è:

$$TopicSim(d) = \frac{\sum_{1 \leq i \leq j \leq m} Sim(t_i, t_j)}{\frac{1}{2}m(m-1)} \tag{2.4}$$

In figura 2.21 è illustrata la distribuzione della misura di diversità degli argomenti basata sulla semantica. Dalla distribuzione si nota che quando la misura cresce la probabilità che una pagina sia spam aumenta, ovvero le pagine spam hanno argomenti che sono in forte relazione semantica tra loro e cioè sono icentrate su pochi argomenti.

L'ultima misura è basata sulla massima semantica ed è definita come:

$$TopicSimMax(d) = max\{Sim(t_i, t_j) | 1 \leq i \leq j \leq m\} \tag{2.5}$$

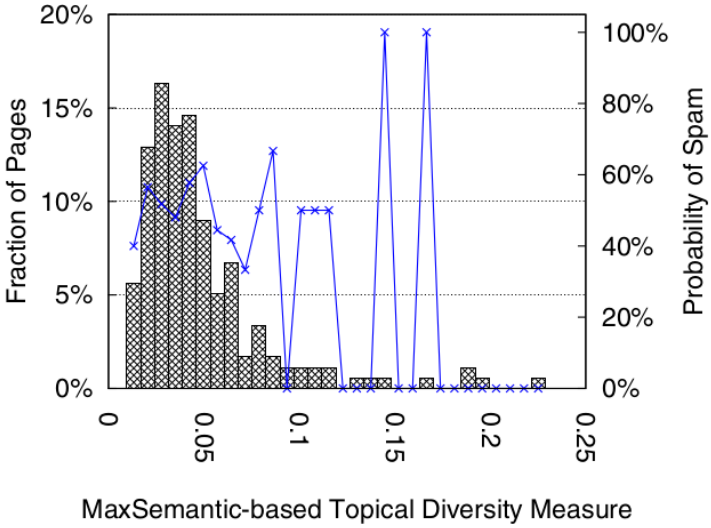

Figura 2.22: Prevalenza di spam relativa alla misura di diversità sulla massima semantica

La distribuzione della misura di diversità basata sulla massima semantica mostra che questa è più alta per le pagine che contengono spam. La distribuzione è rappresentata in figura 2.22.

Anche in questo caso per determinare se una pagina è spam oppure non spam, vengono utilizzati algoritmi supervisionati di apprendimento per istruire un classificatore di pagine spam usando misure di diversità degli argomenti. Con LDA possiamo impostare parametri come il numero di argomenti e il numero di parole per ogni argomento che incidono sulle le prestazioni della classificazione.

2.4 Altre tecniche

Nello studio in [16] viene presentato l'algoritmo WITCH (Web Spam Identification Through Content and Hyperlinks), un algoritmo ibrido che utilizza sia il contenuto della pagina che la struttura dei link per identificare le pagine spam. Come descritto in precedenza nel sotto capitolo 2.1, le differenti proprietà tra le pagine spam e non spam possono essere sfruttate per costruire un classificatore. Per identificare lo spam l'algoritmo WITCH utilizza le feature descritte in precedenza e analizza la struttura dei collegamenti tra le pagine. In particolare viene istruito un classificatore lineare nello spazio delle feature usando la SVM (Support Vector Machine) come funzione

obbiettivo. I collegamenti tra le pagine sono utilizzati in modo da regolarizzare il grafo, che produce una predizione che varia leggermente tra le pagine dei link. Il metodo SVM associato alla regolarizzazione del grafo è efficiente per il rilevamento di web spam.

Un altro metodo, proposto in [17], è denominato "Hidden style similarìty measure" ed è basato su feature extra testuali appartenenti alle pagina HTML. Infatti gli autori sostengono che le pagine spam generate automaticamente non sono facili da rilevare utilizzando metodi classici di classificazione basati solamente sul contenuto; gli autori quindi utilizzano la struttura HTML di una pagina per classificare le pagine simili.

Capitolo 3

Tecniche basate sul grafo

In questo capitolo verrano presentate le tecniche presenti in letteratura che si avvalgono del grafo derivato dalla fase di crawling per rilevare lo spam. In particolare tali tecniche fanno uso del grafo del web ricavato dai collegamenti ipertesuali tra le pagine. Il web, quindi, può essere rappresentato come un grafo diretto $G = (V,E)$, dove V è l'insieme dei nodi del grafo e rappresenta le pagine web mentre E è l'insieme degli archi diretti tra i nodi e rappresenta l'insieme dei link diretti tra le pagine: assumendo che ci siano due pagine web a_p e b_p queste saranno rappresentate da due nodi del grafo a e b; se esiste un collegamento ipertestuale dalla pagina a_p alla pagina b_p allora sarà associato un arco diretto dal nodo a al nodo b. Ogni pagina può avere link in uscita (detti anche outlink) che sono i link presenti nella pagina che referenziano altre pagine, link in entrata (inlink) ovvero tutti i riferimenti della pagina fatti da altre pagine. Inoltre il grafo può essere astratto e rappresentato da una matrice di transizione cosi formata:

$$T(p,q) = \begin{cases} 0 & if(q,p) \in E \\ 1/\omega(q) & if(q,p) \in E \end{cases} \tag{3.1}$$

dove $\omega(q)$ è il grado di link in uscita della pagina q. Possiamo anche definire la matrice di transizione inversa U:

$$U(p,q) = \begin{cases} 0 & if(p,q) \in E \\ 1/l(q) & if(p,q) \in E \end{cases} \tag{3.2}$$

dove $l(q)$ è il grado di link in ingresso della pagina q. Bisogna notare che $U \neq T^T$, ovvero la matrice di transizione inversa U non è uguale alla matrice di transizione trasposta.

3.1 Metodi classici per identificare lo spam web usando il grafo

Uno dei primi metodi adottati per identifiare lo spam web usando il grafo è *Trustrank* [18]. *Trustrank* fa uso di un insieme di pagine di partenza S (detto anche seedset) che sono valutate da degli esperti e che vengono classificate in due sottoinsiemi: pagine non spam S^+ e pagine spam S^-; questo fase è chiamata funzione *Oracle*. Per determinare le pagine non spam senza invocare la funzione *Oracle* su tutto il grafo derivato dalla fase di crawling, viene fatta un'assunzione empirica chiamata *isolazione approssimata dell'insieme delle pagine buone* la quale afferma che le pagine non spam raramente punteranno a delle pagine spam dato che gli sviluppatori di pagine non spam hanno poco interesse nel linkare pagine spam. A quest'ultima affermazione ci sono delle eccezioni che riguardano il caso in cui gli sviluppatori web sonoingannati tramite ad esempio l'uso di tecniche come l'*honeypot*. Quindi fissato un numero limitato di chiamate della funzione *Oracle* sul seedset di partenza e sfuttando l'assunzione fatta precendentemente viene definita una funzione, denominata *funzione di verità ignorante* T_0, per ogni pagina pagina p del grafo:

$$T_0(p) = \begin{cases} O(p) & if \quad p \in S \\ 1/2 & altrimenti \end{cases} \tag{3.3}$$

dove la funzione O è la funzione *Oracle*. Dal momento che le pagine buone dovrebbero puntare ad altre pagine buone assegnamo 1 a tutte le pagine che possono essere raggiunte da una pagina in S^+ in M step. La funzione di verità T_M è definita come:

$$T_M(p) = \begin{cases} O(p) & if \quad p \in S \\ 1 & if \quad p \notin S \quad and \quad \exists q \in S^+ : q \rightarrow_M p \\ 1/2 & altrimenti \end{cases} \tag{3.4}$$

```
function TrustRank
input
        T       transition matrix
        N       number of pages
        L       limit of oracle invocations
        α_B     decay factor for biased PageRank
        M_B     number of biased PageRank iterations
output
        t*      TrustRank scores
begin
        // evaluate seed-desirability of pages
(1)     s = SelectSeed(...)
        // generate corresponding ordering
(2)     σ = Rank({1,...,N}, s)
        // select good seeds
(3)     d = 0_N
        for i = 1 to L do
                if O(σ(i)) == 1 then
                        d(σ(i)) = 1
        // normalize static score distribution vector
(4)     d = d/|d|
        // compute TrustRank scores
(5)     t* = d
        for i = 1 to M_B do
                t* = α_B · T · t* + (1 − α_B) · d
        return t*
end
```

Figura 3.1: Algoritmo di trustrank

Il percorso dalla pagina q a p nell'equazione non comprende pagine spam incluse nell'insieme S^-.

Il problema della funzione di verità T_M è che non esiste la sicurezza che le pagine raggiungibili da pagine buone siano effetivamente della stessa carattesistica. Infatti più lontana una pagina p si trova dal seedset S^+ minore è la certezza che quella pagina sia buona. Un modo per non incorrere in questo errore è ridurre il valore della funzione di verità ogni qual volta ci si allontana dal seedset S^+. In figura 3.1 è possibile vedere in dettaglio l'algoritmo. L'algoritmo calcola il valore di verità di ogni pagina dell'intero grafo. I valori di input sono il grafo descritto dalla matrice di transizione T e il numero di pagine N e i parametri di controllo dell'esecuzione dove: L è il numero di chiamate della funzione *Oracle*, α_b è il fattore di decadimento per il calcolo di *Pagerank* ed infine M_b è il numero di iterazioni per il calcolo di *Pagerank*.

Al primo passo viene invocata la funzione *SelectSeed()* che calcola l'insieme delle pagine con un relativo punteggio di pertinenza per essere incluse nel seedset di partenza.

Gli autori consigianto due metodi per implementare questa funzione tenenendo conto che il risultato, ovvero il seedset, deve essere il migliore è il più piccolo possibile. Il primo approccio, molto simile a PageRank con la differenza che si è interessati ai link in uscita e non a quelli in ingresso, è dare una preferenza alle pagine dalle quali si possono raggiungere molte altre pagine. Perciò per calcolare la desideribilità di una pagina viene calcolato il PageRank sul grafo $G' = (V, E')$, dove

$$(p, q) \in E' \leftrightarrow (q, p) \in E \tag{3.5}$$

Questo approccio è chiamato *Inverse PageRank*. Il secondo metodo detto *High PageRank*, da una alto valore di pertinenza per fare parte del seedset di partenza dell'altgoritmo di *Trustrank* a pagineche hanno un alto valore di pagerank.

Ritornando all'algoritmo di *Trustrank* in figura 3.1 nel secondo punto la funzione $Rank(x, s)$ ordina gli elementi di x in modo decrescente sulla base dello score di s. Il punto tre invoca la funzione *Oracle* su L pagine. I valori del vettore d che corrispondono alle pagine buone del seed sono imposate a 1. Nel punto (4) il vettore viene normalizzato in modo tale che la somma faccia 1. Infine al punto (5) viene calcolato *Trustrank* usando *Pagerank* personalizzato dal vettore d che rimpiazza la distribuzione uniforme. Dall'algoritmo si nota che *Trustrank* è una versione modificata di *Pagerank* dove il vettore di teletrasporto è il seed set S^+ calcolato al punto 3 e 4.

Un altro algoritmo che è stato progettato per identificare lo spam usando come input il grafo delle pagine web è *Anti-Trust Rank* [19]. Partendo dalla stessa intuizione di *Trustrank* dell'isolamento approssimato ovvero che pagine non spam molto raramente punteranno a pagine malevoli, *Anti-trust rank* popola un seedset formato da pagine spam e propaga la funzione Anti Trust (che sarebbe la funzione di verità di Trustrank) sul grafo trasposto con l'obbiettivo di rilevare le pagine spam, le quali successivamente potranno essere filtrate da un motore di ricerca. Più precisamente a differenza di quanto avviene in *Trustrank* dove la funzione *Trust* è propagata dal

seedset composto da pagine non spam lungo tutto il grafo, in *Anti-Trust Rank* la funzione (in questo caso la funzione *Anti Trust*) è propagata nella direzione inversa ai link in entrata ad ogni pagina del grafo, partendo da un insieme di pagine del seed set composto da pagine spam. L'obbiettivo è assegnare un rank maggiore alle pagine spam e successivamente eliminarle dalle ricerche usando un valore di soglia traminte cui le pagine sono considerate spam oppure ritornando le n pagine che hanno valore di *Anti-Trust Rank* più alto.

Trustrank e *Anti-Trust rank* sono ottimi algoritmi per identificare lo spam, ma hanno il problema relativo alla dimensione dell'insieme seed perché potrebbe non essere sufficientemente rappresentativo per coprire bene tutti gli argomenti del web. Per tentare di risolvere questo problema è stato implementato un metodo [20] che per ottenere una vasta copertara delle pagine web fa uso degli argomenti delle pagine come segnale di ingresso ovvero invece di usare un singolo valore di *trustrank* per un sito, vengono estrapolati per ogni pagina gli argomenti che essa contiene e successivamente viene calcolato *trustrank* per tutti gli argomenti di ogni pagina. L'algoritmo consiste nel partizonare il seedset sulla base dei vari argomenti che esso contiene e usare ognuna di queste partizioni come seedset per calcolare il valore di *trustrank* per ogni pagina.

Un altro metodo per l'identificazione di pagine spam è descritto in [21]. Questo metodo separa la credibilità di una pagina dalla credibilità del link per quella pagina. Al contrario di *pagerank* tale metodo è difficilmente manipolabile tramite tecniche come *hoenypot*. La credibilità viene definita in termini di credibilità *k-scope*. Data una funzione C essere una funzione di credibilità che istantaneamente valuta la qualità di un link di un pagina p al tempo t, un valore di $C(p,t) = 0$ indica che p non è credibile mentre $C(p,t) = 1$ indica che p è credibile. Dato un percorso in un grafo diretto G dalla pagina p alla pagina q essere la sequenza di nodi: $path(p,q) = (n_0, n_1, ..., n_j)$ dove $p = n_0, q = n_j$ tale che esiste un arco diretto tra nodi successivi nel percorso $n_i, n_{i+1} \in L$ per $0 \leq i \leq j-1$, diciamo che un percorso in un grafo diretto G dalla pagina p alla pagina q è un *bad path* se la pagina di destinazione è una pagina spam $q \in P_b$ (dove P_b è l'insieme delle pagine spam) e nessuna altra pagina nel percorso è una pagina spam. $path(p,q) = (n_0, n_1, ..., n_j)$ e $q \in P_b$ e

$n_i \notin P_b (0 \leq i \leq j-1)$. La probabilità che una camminata casuale passi, lungo un percorso di lunghezza k, da una pagina p è denotata con $Pr(path_k(p))$ ed è determinata con i pesi degli archi per ogni hop nel percorso:

$$PR(path_k(p)) = \prod_{i=0}^{k-1} w(n, n_{i+1}) \tag{3.6}$$

Quindi credibilità *k-scope* di una pagina è definita in termini di probabilità che una camminata casuale eviti le pagine spam dopo aver superato k hop dalla pagina di origine. La credibilità *k-scope* di una pagina p al tempo t, denotata con $C_k(p,t)$ è definita come segue:

$$C_k(p,t) = 1 - \sum_{l=1}^{k} \left(\sum_{path_l(p) \in BPath_l(p)} Pr(path_l(p)) \right) \tag{3.7}$$

Nel caso $p \in P_b$ allora $C_k(p,t) = 0$. Nel caso in cui non ci siano pagine spam all'interno di k hop di pagine allora p è credibile con un valore $C_k(p,t) = 1$ se lei è un pagina spam o nel caso in cui tutti i percorsi originati da p colpiscono una pagina p all'intenro di k hop, allora p no è credibile $C_k(p,t) = 0$. Ma dato che non che non è possibile avere tutto il grafo e non c'è nessuna sicurezza sulla conoscenza totale dei nodi spam è stato introdotto il concetto di cerdibilità tunable k-Scope, la quale aumenta il calcolo della credibilità k-scope includendo un fattore di penalità di credibilità. GLi obbiettivi sono approssimare al meglio la credibilità k-scope sotto limiti reali e capire come parametri differenti protrebbero influire sulla qualità delle varie funzioni usate. Sia $G = (P, L)$ essere un grafo diretto, k il raggio massimo di camminata e $\gamma(p)$ il fattore di penalità di credibilità di una pagina $p \in P$ dove $0 \leq \gamma(p) \leq 1$. Definiamo la credibilità tunable k-scope di una pagina p, denotata con $C_k(p)$, in due fasi, quando $p \notin P_b$:

$$C_k(p) = \left(1 - \sum_{l=1}^{k} \left(\sum_{path_l(p) \in BPath_l(p)} Pr(path_l(p)) \right) \right) \cdot \gamma(p) \tag{3.8}$$

e quando $p \in P_b$ allora: $C_k(p) = 0$.

Oltre al metodo per definire la credibilità di un link gli autori in [21] propongono un algoritmo, denominato *CredibleRank* di ranking basato sulla credibilità. *CredibleRank* definisce che la qualità di una pagina è determinata da due criteri: la qualità

delle pagine che puntano ad essa e la credibilità di ogni pagina puntata. Un link da un alta-qualità/alta-credibilità conta più di un link da alta-qualità/bassa-credibilità. Definendo con $In(p)$ l'insieme di pagine che puntano a p. Calcoliamo *CredibleRank* $r_c(p)$ per una pagina p

$$r_c(p) = \sum_{q \in In(p)} C(q) \cdot r_c(q) \cdot w(q,p) \tag{3.9}$$

Questa formula dice che il valore di *CredibleRank* di una pagina p è determinato dalla qualità $r_c(q)$ e dalla credibilità dei link $C(q)$ delle pagine che la puntano cosi come la forza del link $w(q,p)$.

Alcuni metodi si avvalgono dei sottografi ricavati a partire dal grafo ottenuto dalla fase di crawling per rilevare lo spam. Ad esempio in [22] lo spam viene neutralizzato tramite l'identificazione dei sottografi composti da pagine spam e i sottografi composti da pagine non spam. In particolare per rilevare i sottografi spam si analizza la struttura *bow-tie* del grafo del web, che si ottiene identificando le componenti fortemente connesse. In figura 3.2 si nota che la struttura *bow-tie* è composta da 5 elementi: il *Core* che è la componente fortemente connessa più che contiene la maggior parte di siti non spam che sono facilemte accessibili agli utenti; la componente *IN* è formata da pagine che puntano al *Core* e OUT da pagine che che sono puntate dal *Core*, quindi le pagine più facilemente accessibili sono quelle che si trovano all'intenroo della componente OUT; infine ci sono le componenti *Tendrils* che sono connesse alle componenti *IN* o OUT e la componente *Tube*. Alcuni studi riportano che dense componenti fortemente connesse vicino al *Core* del grafo del web, sono potenziali indicatori di spam quindi un modo di identificare i sottografi composti da pagine spam è analizzare le componenti fortemente connesse lungo *In*, *Out*, *tendrils* e *tube*. Tutte le componenti saranno valutate da un classificatore per identificare gli host di spam. L'identificazione dei sottografi non spam avviene all'interno della sezione *Core* del grafo del web. Dato il massimo sottografo dove ogni nodo è connesso ad almeno k altri nodi nel sottografo definto come *k-core* e dato il valore di *coreness* di un nodo essere il massimo parametro k in modo che il nodo faccia parte del sottografo con un dato *k-core* allora impostando un certo valore di *coreness* si possono identificare i sottografi con una certa robustezza. Host

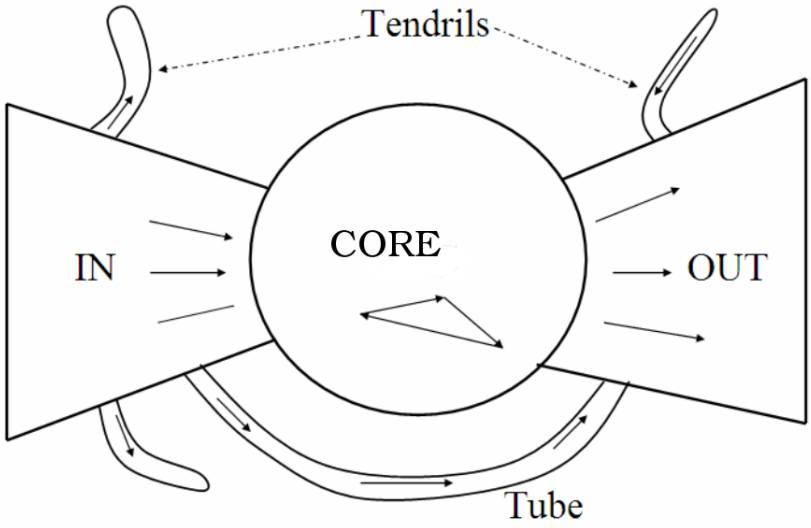

Figura 3.2: Struttura bow-tie.

importanti, ovvero quelli che hanno molti inlink, sono connessi a altri host altamente connessi, formando sottografi robusti con alta *coreness*, mentre gli host spam hanno una bassa *coreness*. Quindi per identificare gli host non-spam può impostare una valore di soglia di *coreness* per gli host che fanno parte della sezione *Core* del grafo del web usando una variante di *k-core* dove ogni nodo ha almeno k inlink.

Il metodo elabora il grafo del web a due livelli di granularità, uno a livello di pagine web e l'altro a livello di host ed è composto dalle segueni fasi:

- vengono rilevati gli host e le pagine spam partizionando il grafo in sottografi densi e calcolando delle feature, per ogni sottografo, sia sulla base degli URL quali: statistiche basate sulla lunghezza, statistiche basate sulla posizione della pagina rispetto all'homepage e statistiche basate sulla lunghezza del nome dell'host; e feature basate sul grafo come ad esempio la struttura all'interno dell grafo;

- i valori di spam sia a livello di pagine e di host vengono ricondotti ad un unico livello che è quello di host;

- vengono identificate le strutture fortemente connesse all'interno del textitCore del grafo del web per rilevare gli host non-spam;

- infine per aumentare il numero di host spam e non spam vengono propagati simultaneamente *trustrank* e *anti-trust rank*, dagli host che siamo sicuri siano non spam e spam agli host vicini in modo da valorizzare gli host non spam e penalizzare quelli di spam.

Un altro metodo per rilevare lo spam utilizza combina un classificatore automatico che combina un insieme di feature basate su link e contenuto [23]. Dato che i link tra le pagine non sono piazzati in modo casuale ovvero pagine simili tendono a linkarsi tra di loro più frequentemente di pagini diverse, si può sfruttare tale meccanismo per rilevare le pagine spam, perché tali pagine tendono a raggrupparsi in cluster. Una spiegazione per questo fenomeno è che le pagine spam utilizzano delle tecniche per aumenteare il rank basato sui link attraverso le link farm. Gli autori assumono che gli host che sono ben collegati tra di loro sono molto probabilmente della stessa classe: spam o non spam.

3.2 Metodi per identificare link farm

Per riconoscere una spam farm (o link farm) si parte dal presupposto che i nodi della spam farm avranno dei link uscenti verso delle pagine target t per aumentarne il rank. In [24] per identificare le spam farm viene introdotta una misura, denominata *spam mass*, dell'impatto dello spam (basandosi sulla struttura del grafo) sul rank di una pagina. Usando questa misura tutte le pagine web avranno assegnate due valori: quello di *pagerank* e quello relativo alla *spam mass*, in particolare le pagine target delle spam farm ricevono un alto valore di *pagerank* e un alto valore di *spam mass* mentre le pagine non spam anche possono avere un alto valore di *pagerank* ma un basso valore di *spam mass*. Partendo dal presupposto che le il web può essere partizionato in nodi non spam V^+ e nodi spam V^- dove la loro unione forma il grafo del web, per una data partizione $\{V^+, V^-\}$ di V e per dei nodi x il pagerank di x è la somma dei contribbuti di nodi non spam e dei nodi spam. Quindi per stimare la *spam mass* per ogni nodo del grafo si utilizzano due misure di *spam mass*:

- La *spam mass assoluta* di x, denotata con M_x, è il *pagerank* che x riceve dai

nodi spam è che uguale a:

$$M_x = q_x^{V^-} \tag{3.10}$$

dove $q_x^{V^-}$ è appunto il *pagerank* di x derivato dai nodi spam.

- La *spam mass relativa* di x, denotata da m_x, è la frazione del *pagerank* di x dovuto dal contribbuto dei nodi di spam cioè:

$$m_x = q_x^{V^-}/p_x \tag{3.11}$$

dove $q_x^{V^-}$ è il *pagerank* di x derivato dai nodi spam e p_x il *pagerank* derivato da tutti i nodi.

Dal momento che non è possibile conoscere le proprietà (spam o non spam) per tutti i nodi del grafo ma solo un sottoinsieme di nodi buoni $(\tilde{V})^+$ le misure precedenti vengono calcolate nel seguente modo:

- la stima assoluta di *spam* mass di un nodo x è:

$$\tilde{M}_x = p_x - p'_x \tag{3.12}$$

- la stima relativa di *spam mass* di x è:

$$\tilde{m}_x = (p_x - p'_x)/p_x = 1 - p'_x/px \tag{3.13}$$

dove $p = PR(v)$ è il *pagerank* dei nodi basato su una distribuzione uniforme mentre $p' = PR(v^{\tilde{V}^+})$ è *pagerank* basato sull'insieme $(\tilde{V})^+$ con una distribuzione di salto $v^{\tilde{V}^+}$. Nel caso in cui si conoscesse $\tilde{V}^-$ lo *spam mass* può essere stimato con $M = PR(v^{\tilde{V}^-})$. Mentre se si conoscerro entrambi i sottoinsiemi V^+, V^- la stima dello *spam mass* può essere fatta attraverso $(\tilde{M} + \tilde{M})/2$. Perciò è possibile utilizzare un valore di soglia tramite la quale una pagina è considerata facente parte di una spam farm se il valore di *spam mass* supera la soglia.

Le pagine all'interno delle spam farm sono densamente connesse tra di loro e hanno molti link in entrata e uscita. Usando le pagine facenti parte dalle spam farm come seedset, ogni nuova pagina, può può fare parte della spam farm se questa ha molti link in entrata e uscita, da e per, il seedset e quindi si può allargare il seedset

Let p to denote the URL for a web page and $d(p)$ represents the domain name of p. Suppose we are given N pages initially. $IN(p)$ and $OUT(p)$ represent the sets of incoming and outgoing links of p, respectively.

1. For each URL i in $IN(p)$, if $d(i) \neq d(p)$ and $d(i)$ is not in $INdomain(p)$, then add $d(i)$ to the set $INdomain(p)$.
2. For each URL k in $OUT(p)$, if $d(k) \neq d(p)$ and $d(k)$ is not in $OUTdomain(p)$, then add $d(k)$ to the set $OUTdomain(p)$.
3. Calculate the intersection of $INdomain(p)$ and $OUTdomain(p)$. If the number of elements in the intersection set is equal to or bigger than the threshold T_{IO}, mark p as a bad page.
4. Repeat 1 to 3 for every page in the data set.
5. For all pages that have been marked bad during 1 to 4, place a 1 in the initial value array $A[N]$. Return A.

Figura 3.3: Algoritmo di ricerca dei seed set

di partenza aggiungendo la nuova pagina. Il processo è iterato fin quando nessuna altra pagina potrà essere aggiunta. Un metodo che applica questo principio per identificare le spam farm è descritto in questo modo è descritto in [25]. Per decidere se una pagina deve fare parte di un seedset, si parte dall'assunzione che le pagine all'interno della link farm normalmente hanno molti nodi in comune tra l'insieme dei link in entrata e quello dei link in uscita. Se ci sono solo uno o due nodi in comune non etichettiamo queste pagine come pagine problematiche ma se ci sono molti nodi in comune è probabile che queste pagine facciano parte di una spam farm, l' algoritmo è presentato in dettaglio in figura 3.3. Se il numero di incoming link in comune o outgoing link in comune è uguale o maggiore a una soglia T_{IO} allora le pagine sono etichettate come spam. Dall'intuizione che se una pagina punta a un insieme di pagine cattive è probabile che anche essa sia cattiva viene allargato il seed set usando un altra soglia T_{PP} per giudicare una pagina: se il numero di outgoing link a pagine spam è uguale o supera la soglia, la pagina sarà giudicata spam e perciò facente parte del seed set della spam farm. Il metodo descritto è formalizzato nell'algoritmo di *ParentPenality* in figura 3.4. Una volta trovate le pagine spam bisogna utilizzare queste informazioni per il ranking. Un modo è quello di eliminare

ParentPenalty:
Suppose we already have an array $A[N]$ from the initial step in which bad pages have value 1 in it and other pages have value 0, and a threshold T_{PP},

1. For each member p s.t. $A[p] = 0$, fetch its outgoing links set $OUT(p)$.
2. Set $badnum$=0.
3. For each element k in $OUT(p)$, if $A[k]$ is 1, then increase $badnum$ by 1.
4. If $badnum \geq T_{PP}$, set $A[p] = 1$.

Repeat 1 to 4 until the values of A do not change.

Figura 3.4: Algoritmo per aumentare il seedset

queste pagine direttamente dal grafo del web, un altro modo può essere quello di penalizzare i link invece che le pagine, facenti parte della spam farm, con un fattore di decadimento o infine potrebbe essere utile eliminare direttamente i link che fanno parte della spam farm.

3.3 Link dai forum

Un metodo utilizzato dagli spammer per creare delle grandi link farm è lo scambio di link che molto spesso avviene attraverso l'utilizzo di forum SEO (Search Engine Optimization). Questi forum contengo varie tipologie di discussioni, come ad esempio siegazioni su come manipolare gli algoritmi dei motori di ricerca per incrementare il rank delle pagine e sezioni dedicate anche allo scambio dei link. Un modo utile sarebbe quello di utilizzare le informazioni contenute in questi forum per identificare i siti spam e qundi azzerare o attenuare i loro valore di rilevanza delle ricerche in un motore di ricerca. Tale processo non è semplice in quanto nei post dei forum ci sono altre informazioni che producono rumore nel rilevamento dei link. Un modo efficace per implementare questa tecnica consiste nel seguire le seguenti fasi [26]:

- si estrggono tutti i link contenuti nei post.

- vengono estratte le feature dai link sulla base delle loro relazioni con gli utenti del forum e della struttura dei loro link nel grafo del web. Le feature vengono catalogate in tre tipi: feature del forum SEO (quali la frequenza di URL nel forum, numero di thread che il proprietario dell'URL ha discusso, numero di post autorizzati dal proprietario dell'URL, numero di URL inseriti dal proprietario dell'URL, media degli URL per post di un utente, numero di post che contengono l'URL del proprietario), feature del grafo (fanno parte il numero di link in ingresso, numero di link in uscita, media dei link in uscita dei vicini in ingresso, media dei link in entrata dei vicini in uscita) e feature del sito (la lunghezza dell'URL).

- viene utilizzato un framework per calcolare il valore di spam dei siti.

Questi metodi sono di particolare aiuto nell'incrementare il numero di pagine spam che i metodi convenzionali (sia basati sul contenuto che sulla struttura del grafo) non riescono a identificare e perciò è possibile usarlo come metodo complementare ai metodi classici.

3.4 Metodi per migliorare la classificazione

Invece di cercare di migliorare la rilevazione delle pagine spam utilizzando nuove feature, o nuove tecniche altri metodi sono stati sviluppati con l'obbiettivo di migliorare la fase di classificazione . In [27] viene presentato un metodo per migliorare la classificazione utilizzando un classificatore di base per etichettare le pagine e delle euristiche, basate sulla tipologia dei nodi vicini a un nodo v, in modo da determinare se tale nodo v dovrebbe essere rietichettato basandosi sulla basa della prima classificazione o sull'infromazione portata dai nodi vicini. Gli autori di tale metodo ipotizzano che per un sito la struttura dei vicini è un buon indicatore per classificarlo in spam o non spam. In particolare vengono analizzate alcune distribuzioni delle proprietà dei vicini:

- la distribuzione dello spam in entrata: in figura 3.5 viene rappresentata la distribuzione dello spam in entrata: ogni sito andrà a finire in uno dei settori

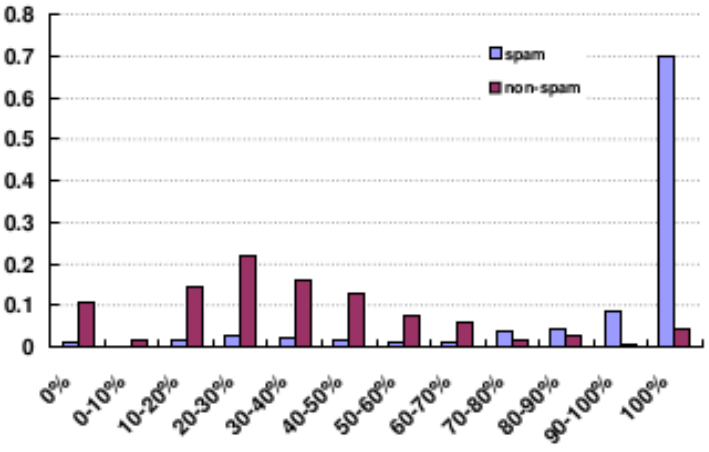

Figura 3.5: Distribuzione dello spam in entrata

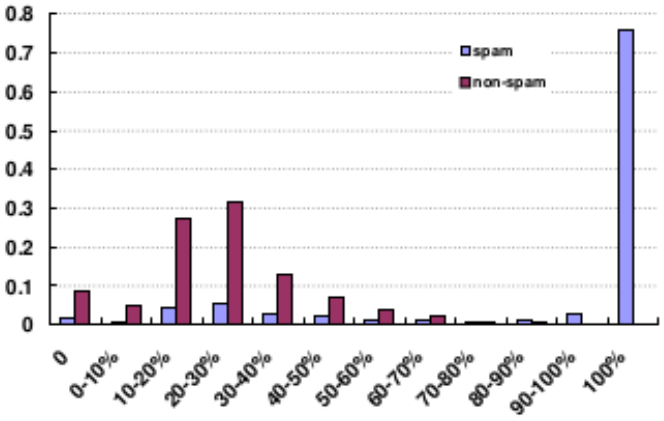

Figura 3.6: Distribuzione dello spam in uscita

sull'asse x in base alla frazione di nodi spam tra i sui vicini in entrata. L'asse y rappresenta la percentuale di spam/non spam all'interno del settore. Una grande porzione di siti spam ha molti vicini che sono spam.

- la distribuzione dello spam in uscita: in figura 3.6 viene osservato una distribuzione simile a quella per lo spam in entrata.

- distribuzione entrante pesata:vegono esaminati gli in-lin pesati con pagerank (figura 3.7).

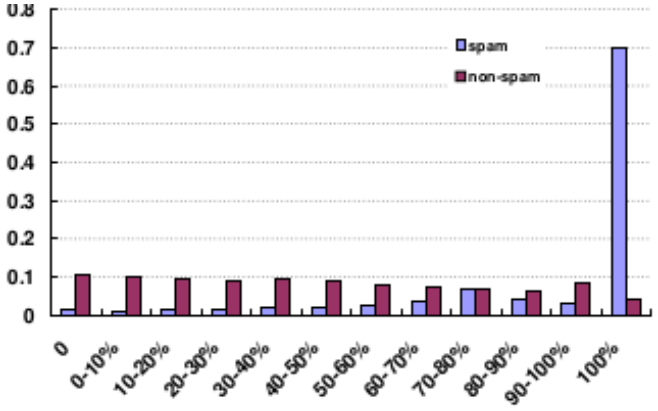

Figura 3.7: Distribuzione entrante pesata

Per rietichettare i nodi vengono cambiate le etichette che sono assegnate dal primo classificatore sulla base dei nodi vicini, in particolare prima viene definita un etichetta per i nodi vicini di un sito, con un certo livello di confidenza, dopo viene confrontata questa etichetta con la prima (assegnata dal classificatore) se le due etichette sono molto diverse tra loro e l'etichette dei vicini sono moltoo affidabili in termini di confidenza allora l'etichetta del sito viene cambiata ooppure si può usare un altro classificatore che usa le euristiche definite prima.

Un altro metodo che fa uso della riassegnazione delle etichette associate a ogni nodo dopo una prima fase di classificazione è descritto in [28]. Tale metodo fa uso di una strategia di riestrazione delle feature facendo delle analisi basate sui clustering dei nodi, analisi sulla propagazione e analisi basate sulla struttura del grafo dei nodi vicini. Le analisi leate ai cluster vengono eseguite nel seguente modo: dato $G = (V, E, w)$ essere un grafo non diretto dove V è l'insieme degli host, $w = f(n)$ la funzione di pesatura dove n è il numero di link tra le pagine dell'host e E è l'insieme degli archi con peso non zero. Il grafo dopo essere stato clusterizzato in k cluster, la feature basata sul clustering può essere calcolata.

$$cf(H) = \frac{\sum_{h \in C(H)} spamicity(H)}{|C(H)|} \tag{3.14}$$

dove $C(H)$ è il cluster al quale l'host H appartiene e la $spamicity(h)$ è il valore di spam rilevato durante la prima classificazione. Mentre la feature di propagazione si

ottiene nel seguente modo:

$$pf(H)^{(t)} = \sum_{h:h->H} \frac{pf(h)^{t-1} \times weight(h,H)}{\sum_{g:h->g} weight(h,g)} \tag{3.15}$$

dove t è il numero di iterazioni, $pf(h)^{(0)} = spamicity(h)$ e $weigth(h,H)$ è il peso degli host h e H. Infine l'ultima feaure quella relativa alla struttura dei vicini si ottiene:

$$nf(H) = \frac{\sum_{h \in N(H)} (sapmicity(h) \times weight(H,h))}{|N(H)|} \tag{3.16}$$

dove $N(H) \in \{inlink(H), outlink(H), outlink(outlink(H)), inlink(inlink(H)), inlink(outlink(H)), outlink(inlink(H))\}$ e quindi rappresenta l'insieme delle relazioni con i nodi vicini del nodo H . L'algoritmo è suddiviso in quattro fasi:

- vengoono estratte delle feature base per ogni nodo;
- viene eseguita una prima classificazione;
- successivamente vengono riestratte le atre feature basate sui cluster, sulla propagazione e sulla struttura dei vicini;
- viene rieseguita una seconda fase di classificazione;

Nelle prime due fasi vengono estratte le feature di base basate sul contenuto e sulla struttura dei link e viene fatta una prima classificazione; nella terza e quarta fase vengono estratte i nuovi tipi di feature e successivamente viene rieseguita la classificazione.

Il problema di molti algoritmi di classificazione è il riperimento di dati etichettati ovverò per avere un buon risultato nella classificazione i dati di training devono essere consistenti. Per risolvere questo problema in [29] viene proposto un algoritmo di apprendimento supervisionato per migliorare le performance di un classificatore. Tale algoritmo è basato sul tradizionale self training e sull'apprendimento dai link ovvero la dipendenza topologica, l'algoritmo è denominato *Link-training*. L'algoritmo di apprendimento si può riassumere nei seguenti processi:

- Prima viene istruito un classificatore con un piccolo dataset.

- Successivamente viene utilizzato il classificatore per categorizzare e assegnare un valore di spam (PS) ai dati non etichettati; il calcolo di PS avviene nel seguente modo:

$$PS(x) = \frac{P_{spam}(x)}{P_{spam}(x) + P_{normal}(x)} \tag{3.17}$$

  dove, $P_{spam}(x)$ è la probabilità di x di essere un nodo spam.

- Il passo successivo consiste nell'assegnare a tutti i nodi il valore di spam calcolato.

- Per istruire il classificatore oltre al valore di spam di base viene calcolato anche quello relativo ai vicini (LS):

$$LS(h) = \frac{\sum_{v \in N(h)} (PS(v) \times weight(h, v))}{\sum_{v \in N(h)} weight(h, v)} \tag{3.18}$$

  dove v e h sono gli host, $weight(h, v)$ è il peso dell'host h e v , $weight(h, v \in 1, \log(n))$, dove n è il numero di link tra i due nodi. $N(h) \in inlink(h) \cup outlink(h)$, dove $inlink(h)$ rappresenta l'insieme dei link in entrata di h e $outlink(h)$ rappresenta l'insieme dei link in uscita di h. Queste fasi del processo sono cicliche per un numero stabilito di iterazioni.

Gli algoritmi di spam detection basati sul grafo che viene ottenuto dalla fase di crawling, cercano di sfruttare le caratteristiche dei grafi per ottenere delle informazioni riguardo i nodi spam. Quasi tutti i metodi descritti si basano sulla stessa intuizione dell'algoritmo di *trustrank* (quella realativa all'isolazione approssimata delle pagine buone) sfruttando tale intuizione per determinare le pagine spam. Ad esempio *Anti-trust rank* sfrutta tale intuizione ma non andando a manipolare i link in uscità di ogni nodo ma i link dei nodi entranti. Altri metodi mentre si basano sulla ricerca delle spam farm mentre altri ancora si focalizzano sul miglioramento dei vari algoritmi di classificazione. Oltre ai metodi e feature descitte fino adesso ci sono altri metodi che utilizzano criteri diversi dal contenuto delle pagine web e dalla struttura del grafo per determinare le pagine spam, tali metodi sono nati per andare incontro alla crescita di nuovi tipi di spam web.

Capitolo 4

Tecniche che fanno uso di altri segnali

Come già discusso nei capitoli precedenti le tecniche basate sul contenuto utilizzano delle feature determinate empiricamente su alcuni dataset di pagine web a disposizione dei ricercatori mentre le tecniche che basate sul grafo diminuiscono o eliminano del tutto l'impatto dello score dei nodi spam attraverso la determinazione di alcuni pattern strutturali all'interno del grafo. Quindi queste nuove tecniche sono la risposta alla continua evoluzione delle tecniche di spam web. Mentre in questo capitolo saranno illustrate le tecniche di spam detection che utilizzano come segnale di partenza oggetti al di fouri del contenuto delle pagine web e della struttura del grafo. Questi nuovi tipi di approcci sono state sviluppati per identificare tutti i vari tipi di spam che hanno caratteristiche tali da non essere identificate con i metodi tradizional oppure per essere usate in modo complementare alle tecniche già presenti in letteratura ovvero quelle basate su contentuto e quelle basate sulla struttura del grafo.

4.1 Rilevamento dello spam di tipo cloacking

Ci sono pochi metodi che tentano di rilevare il cloaking. Gli spammer possono rilevare un crawler dal suo indirizzo IP o dal campo *user-agent* all'interno di una

richiesta HTTP e quindi fornire due versioni di una pagina a seconda di chi effettua la richiesta. Molti dei metodi per rilevare il cloaking prendono in considerazioni due copie di una stessa pagina, la prima è ottenuta tramite un richiesta HTTP al server (che ha la pagina) da parte di un crawler e la seconda è ottenuta tramite la richiesta HTTP al server da parte del browser, quindi le due copie vengono confrontate per verificare se le due pagine siano identiche e perciò che non si tratti di cloacking. I metodi che fanno uso di questa tecnica non sono efficaci per il fatto che con l'avvento del WEB 2.0 le pagine sono generate dinamicamente e possono variare nei contenuti. Per superare l'incertezza legata alla natura dinamica delle pagine si può utilizzare un metodo [30] che si basa sul confronto dei termini tra le due copie di una pagina usando delle funzioni hash per aumentare la velocità di confronto. Il metodo definiche come C_i l'insieme delle copie di una pagina che sono ottentute tramite le richieste fatte da parte di un crawler al server, B_i l'insieme delle copie di una pagina che sono conseguite attraverso delle richieste da parte del browser al server e sf una funzione hash allora $f(B_i)$ e $f(C_i)$ sono i valori di hash rispettivamente delle copie del browser e del crawler da questo ne consegue che se i valori delle due funzioni hash sono identici le pagine sono identiche in caso contrario le due pagine sono differenti. L'algoritmo contradistingue il cloacking in statico e dinamico. Nel primo caso ci si trova nella situazione in cui le copie di una pagina del browser e del crawler sono diverse mentre nel cloacking dinamico si ha esattamente la seguente situazione $B_1 = B_2 = C_2 \neq C_1$ ovvero la due copie di una pagina del browser sono identiche a una copia della pagina del crawler ma sono differenti da un altra copia del crawler. I vari casi di cloacking statico si possono esaminare come segue:

- La prima fase valuta $f(C_1)$ e $f(B_1)$ se i due valori di hash sono differenti vengono calcolati anche $f(B_2)$ e $f(C_2)$. Valori di hash differenti implicano una buona probabilità che le due pagine derivino da un meccanismo di cloacking, ma queste considerazioni non sono sufficienti perciò bisogna eseguire il passo successivo.

- In questa fase i valori di hash vengono valutati nel seguente modo: $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) = f(C_2)$, $f(B_1) = f(B_2)$. Tale valutazione

suggerisce che la probabilità di cloacking è elevata ma data l'alta natura dinamica delle pagine web per essere sicuri di essere di fronte ad un meccanismo di cloacking si calcola la differenza dei termini tra C_1 e B_1 (denominata D_{C1B1}) che se è minore di una certa soglia allora indica che non si tratta di cloacking.

- La terza fase si contradistingue dal caso in cui $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) = f(C_2)$, $f(B_1) \neq f(B_2)$. Questo caso si differenzia dal precedente per il fato che le due copie del browser sono diverse questo suggerisce che la probabilità di cloacking è bassa e che si tratti di un contentuto altamente dinamico. Per prevenire falsi positivi vengono viene calolata D_1 come le differenze dei termini tra C_1 e B_1 e D_2 come le differenze dei termini tra C_2 e B_2 e successivamente D_{TOTAL} come la differenza tra $D_1 \cup D_2$ e $D_1 \cap D_2$. Se D_{TOTAL} è più grande di una certa soglia allora ci si trova davanti un meccanismo di cloacking.

- La quarta fase si ha con $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) \neq f(C_2)$, $f(B_1) \neq f(B_2)$, dal momento che $f(C_1), f(C_2), f(B_1), f(B_2)$ sono differenti queste pagine cambiano molto velocemente sono pagine dinamiche.

- Infine $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) \neq f(C_2)$, $f(B_1) = f(B_2)$, in questo caso esiste una buona probabilità di cloacking. Per determinare se la differenza tra le due copie del crawler e l'identicità tra le copie del browser indicano che si tratta di cloacking viene calcolato D_{C1C2} come la differenza dei termini tra le copie del crawler e D_{B1C1} come la differenza dei termini tra B_1 e $C1$. Se D_{C1C2} e maggiore di D_{B1C1} allora non si tratta di cloacking perchè la differenza tra le copie del crawler e maggiore della differenza tra la copia del crawler e quella del browser; questo sta a sottolineare la natura dinamica della pagina.

Per quanto riguarda il cloacking dinamico si hanno due casi: il caso in cui $f(C_1) \neq f(B_1)$, $f(C_2) = f(B_2)$, $f(B_1) \neq f(B_2)$ ma non abbiamo a che fare con meccanismi di cloacking in quanto $f(B_1)$ è diverso di $f(B_2)$ ovvero già le due copie del browser sono diverse e quindi si tratta della pagina che cambia i contenuti a causa della sua

natura dinamica ma non per fare cloacking. Mentre nel secondo caso $f(C_1) \neq f(B_1)$, $f(C_2) = f(B_2)$, $f(B_1) = f(B_2)$ si nota che la copia C_1 del crawler e quella B_1 del browser sono diverse mentre nel secondo caso la copia del crawler C_2 è uguale alla copia del browser B_2. Questo significa o che il server spam sceglie quando effettuare il cloacking o che c'è una relazione con la natura dinamica delle pagine. Per questo l'algoritmo prevede il calcolo di D_{C1C2} che è uguale alla differenza dei termini tra le due copie del crawler. Se D_{C1C2} è maggiore di una certa soglia allora si tratta di cloacking.

Nella progettazione di una componente di spam detection all'interno di un crawler si protrebbe utilizzare questo metodo per rilevare il cloacking affiancandolo ad altri metodi che rilevano lo spam di altro tipo anche se questo implicherebbe che per ogni pagina ottenuta durante la fase di fetching del crawler bisognerebbe eseguire la stessa richiesta attraverso un browser. Un metodo molto più efficiente sarebbe quello di determinare il cloacking direttamente nella sola sessione del crawler.

4.2 Rilevare lo spam tramite l'header HTTP

Un metodo innovativo per rilevare lo spam è quello descritto in [31]. Tale metodo può essere usato come supporto ad altri metodi descritti in precedenza e può essere utilizzato in modo dinamico durante la fase di downloading delle pagine, utile per risparmiare il numero di richieste inutili verso pagine spam. Diversamente dai metodi classici (basati sull'uso del contenuto della pagina o sulla struttura del grafo) questo metodo utilizza le informazioni racchiuse all'interno dell'header HTTP per determinare le pagine spam. Oltre all'utilizzo lato server (crawler) può essere usato anche lato client per migliorare la qualità dei contenuti protegendo da malware e permettendo di riparmiare banda e quantità di memoria. Infatti prima viene effetuata una richiesta HTTP, al server, per una pagina ma durante la lettura della pagina ci si ferma al solo header; successivamente viene azionato un classificatore per valuare l'header come spam o non spam; se l'header viene classificato come non spam allora si continua con la lettura del resto della pagina. I vari campi all'interno delle sessioni HTTP possono essere usate come feature che sono poi valutate dal

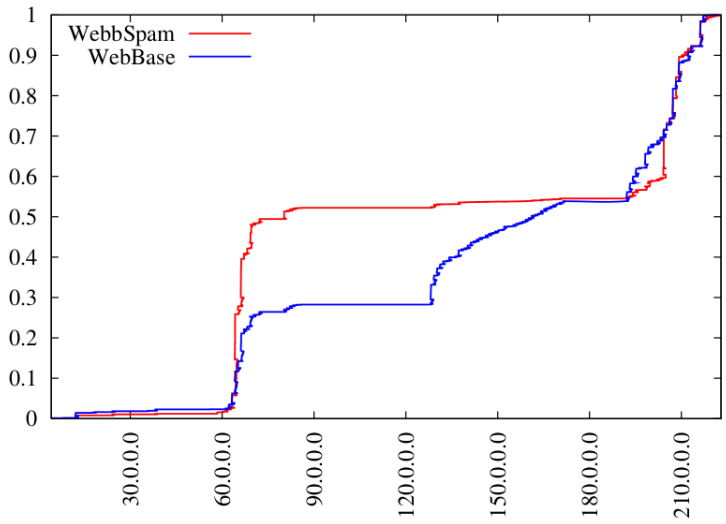

Figura 4.1: Distribuzione degli indirizzi IP di due dataset: WebSpam che contiene pagine spam e WebBase che contiene pagine non spam.

classificatore. Analizzando i vari campi si nota che le pagine spam hanno molto frequentemente alcuni campi rispetto alle pagine non spam che hanno gli stessi campi con minore probabilità. Ad esempio in figura 4.1 vengono confrontate le distribuzioni degli indirizzi IP per il corpus di pagine WebSpam (che contiene delle pagine spam) e il corpus WebBase (che contiene pagine non spam); dal grafico si nota che gli indirizzi IP nel corpus WebSpam sono concentrati principalmente nel range tra $63.* - 69.*$ e $204.* - 216.*$. Perciò tale metodo tenta di fare uso dell'header HTTP in modo tale da classificare le pagine basandosi sull'osservazione che pagine spam e pagine non spam hanno valori (campi dell'header) che hanno distribuzioni distinte. Questo metodo comunque non è molto affidabile se usato da solo perciò è un ottimo strumento da usare in modo complementare ad altri metodi più tradizionali. Un uso sensato sarebbe quello di utilizzare questa tecnica come processo di preselezione in modo tale da sfoltire il numero di pagine che gli altri metodi (che si basano o sull'uso del contenuto delle pagine o del grafo del web) devono utilizzare e quindi richiedendo un numero minore di risorse.

4.3 Altri metodi

Nuovi metodi di spam detection fanno uso di pattern basati sul comportamento dell'utente per identificare le pagine spam come in [32] un metodo che identifica le pagine spam che fa uso di tre feature ricavate da pattern comportamentali basati sulle azioni dell'utente in presenza di pagine spam e pagine non spam. Le feature sono cosi calcolate:

- La prima feature è denominata SEOV (Search Engine Oriented Visit) ed è definita come:

$$SEOV(p) = \frac{\#(Search\ engine\ oriented\ visit\ of\ p)}{\#(Visit\ of\ p)} \tag{4.1}$$

  dove p rappresenta la pagina web per cui viene calcolata la feature. La misura al numeratore indica la visita della pagina p per mezzo dei motori di ricerca mentre la misura denominatore indica la visita alla pagina p senza il bisogno di utilizzare i motori di ricerca. Dato che un utente non andrebbe mai su pagine spam se non ingannato dai risultati dei motori di ricerca, le pagine spam hanno valori alti per questa feature. In figura 4.2 è rappresenta la distribuzione delle pagine (del dataset usato dagli autori) sulla base del valore della feature SEOV; in rosso sono rappresentate le pagine spam mentre in blu sono rappresentate le pagine non spam. Molte pagine web spam hanno valori SEOV più alti delle pagine non spam perchè i motori di ricerca sono gli strumenti, e in alcuni casi sono gli unici, tramite cui le pagine spam possono essere raggiunte.

- Visto che esiste una grande differenza tra chi progetta una pagina spam e chi ne progetta una non spam, ci si avvale del comportamento dell'utente per etichettare le pagine. Un utente rimane su una pagina spam solo fino a al punto in cui capisce di essere su un sito con contenuti non pertinenti, mentre nel caso in cui si trovi a navigare su una pagina non spam l'utente è stimolato a rimanrci. Quindi la seconda feature è definita come SP (Start Point Visiting rate):

$$SP(p) = \frac{\#(user\ click\ a\ hyperlink\ on\ p\ while\ visiting\ p)}{\#(Visit\ of\ p)} \tag{4.2}$$

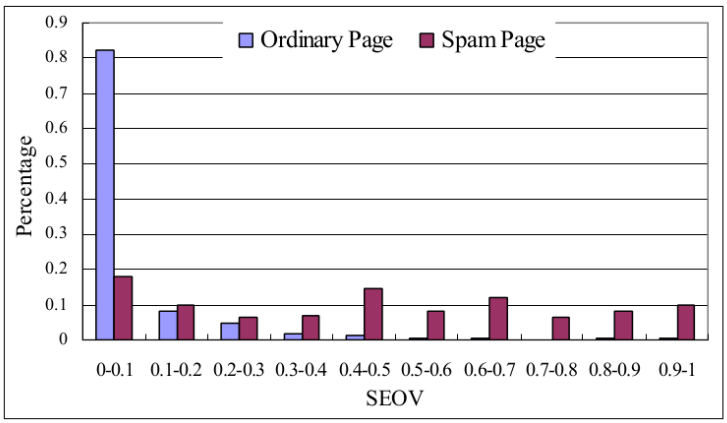

Figura 4.2: Distribuizione delle pagine sulla base della feature SEOV.

questa feature indica quanti click sono fatti su una certa pagina p.

- Infine l'ultima feature è denominata SN (Short-term Navigation rate) e indica quante pagine di un sito w saranno visitate una volta che un utente visita il sito w. Questa misura è definita in questo modo:

$$SN(w) = \frac{\#(Session\ in\ wich\ users\ visit\ less\ than\ N\ pages\ in\ w)}{\#(Session\ in\ wich\ user\ visit\ w)} \quad (4.3)$$

Qundi questo metodo utilizzando pattern comportamentali per determinare le pagine spam è molto più flessibile dei metodi tradizionali, i quali sono dipendenti dalla struttura della pagina o del grafo. La novità di questo metodo sta nel fatto che mentre gli altri metodi sono basati sullo studio di determinate proprietà che le pagine web hanno (nel caso dei metodi basati sul contenuto delle pagine) o sullo studio del grafo del web e la tipologia che le pagine web assumo all'interno quest'ultimo metodo si basa sul comportamento dell'utente. Questo rende il problema dell'identificazione dello spam scalabile ovvero protrebbe riuscire ad arginare e rilevare anche le nuove tipologie di spam web che potrebbero essere implementate mentre con i metodi classici è più difficile in quanto si basano su delle euristiche riscontrate su alcuni dataset e quindi sono dipendenti dalla tipologia di spam che si riscontra e non sono mutabili ovvero non possono essere flessibili per apprendere i nuovi tipi di tecniche spam.

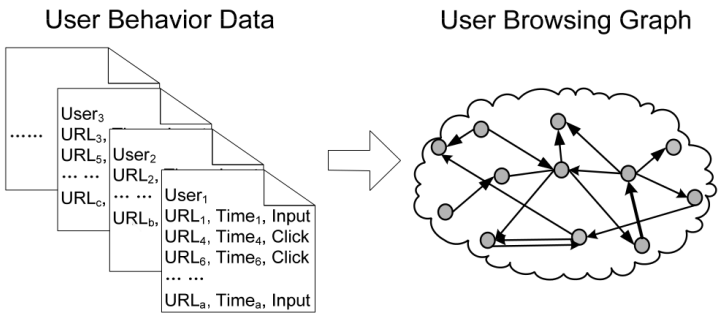

Figura 4.3: Dal comportamento dell'utente al grafo.

Ci sono altri metodi che fanno uso del comportamento dell'utente durante la navigazione web per determinare le pagine spam molto innovativo e interessante è *BrowseRank* [33]. *BrowseRank* determina l'importanza di una pagina web dal grafo ricavato dal comportamento dell'utente durante la navigazione web con il browser. Il grafo è costituito dai vertici che rappresentano le pagine. da archi orientati che rappresentano le transizioni da una pagina all'altra da parte dell'utente e anche dal tempo di permanenza su una pagina. Infine come per *Pagerank* viene utilizzato il grafo ricavato per determinare l'importanza delle pagine. In figura 4.3 è rappresentato uno schema esplicativo del modo con cui si ricava il grafo. Il grafo è ricavato dai browser degli utenti (ad esempio tramite l'utlizzo di toolbar) che raccolgono diversi dati come l'URL, il tempo di permanenza, la tipologia della visita (ad esempio se l'utente ha inserito l'URL nella barra degli indirizzi del browser oppure se arrivato a ad una pagina per mezzo di un link) e vegono poi recuperati da un server che integra i dati provenienti da milioni di utenti. Questo algoritmo ha due vantaggi principali rispetto ai metodi tradizionali sui link quali:

- Dato che il grafo è ricavato durante la fase di navigazione è più accurato di quello ricavato da un crawler perché i link tra le pagine possono cambiare continuamente.

- Questo metodo inoltre tiene conto anche del tempo in cui ci si sofferma su una pagina, caratteristica che può fare capire se si è in presenza di una pagina spam, infatti un utente non avrebbe nessun vantaggio a rimanere a lungo su una

pagina con poca pertinezza e qualità rispetto al suo bisogna di informazione.

Dagli esperimenti gli autori hanno notato che *BrowseRank* è più efficiente rispetto a *Trustrank* e quindi dimostra che i metodi che si basano su segnali differenti dal contenuto o dal grafo del web possono essere in grado di lavorare autonomamente e non solo in modo complementare ai metodi classici.

Un ultimo metodo [34] per rilevare lo spam web affronta il problema da un altro punto di vista: gli autori hanno osservato che le query più comuni (quelle che hanno un alta frequenza nei file log dei motori di ricerca) sono quelle che genereranno più spam. Tali query hanno le seguenti caratteristiche: sono molto comuni e quindi riflettono una grande quantità di richiesta da parte degli utenti, e i risultati per queste query sono composti da pochi risultati utili in quanto sono prese di mira per fare spam ovvero le pagine spam molte volte sono piene di pubblicità e per attirare l'utente (o meglio per ingannarlo) cercano di manipolare i motori di ricerca per andare in cima ai risultati quindi le query più popolari saranno prese di mira più frequentemnte da chi fa spam. Il metodo, tenendo conto di queste considerazioni, fa uso dei file log dei click dei motori di ricerca per prevenire lo spam web.

Bibliografia

[1] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly, "Detecting spam web pages through content analysis," in *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, (New York, NY, USA), pp. 83–92, ACM, 2006.

[2] J. Martinez-Romo and L. Araujo, "Web spam identification through language model analysis," in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, AIRWeb '09, (New York, NY, USA), pp. 21–28, ACM, 2009.

[3] N. R. Jennings, "The global economic impact of spam.," *Ferris Research*, 2005.

[4] N. R. Jennings, "Cost of spam is flattening - our 2009 predictions.," *Ferris Research*, 2009.

[5] N. Spirin and J. Han, "Survey on web spam detection: Principles and algorithms," *SIGKDD Explor. Newsl.*, vol. 13, pp. 50–64, May 2012.

[6] N. Eiron, K. S. McCurley, and J. A. Tomlin, "Ranking the web frontier," in *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, (New York, NY, USA), pp. 309–318, ACM, 2004.

[7] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. Pages 117–119.

[8] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Found. Trends Inf. Retr.*, vol. 3, pp. 333–389, Apr. 2009.

[9] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[10] Z. Gyongyi and H. Garcia-Molina, "Web spam taxonomy," Technical Report 2004-25, Stanford InfoLab, March 2004.

[11] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM*, vol. 46, pp. 604–632, Sept. 1999.

[12] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. Pages 474–476.

[13] D. Fetterly, M. Manasse, and M. Najork, "Spam, damn spam, and statistics: Using statistical analysis to locate spam web pages," in *Proceedings of the 7th International Workshop on the Web and Databases: Colocated with ACM SIGMOD/PODS 2004*, WebDB '04, (New York, NY, USA), pp. 1–6, ACM, 2004.

[14] D. Fetterly, M. Manasse, and M. Najork, "Detecting phrase-level duplication on the world wide web," in *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, (New York, NY, USA), pp. 170–177, ACM, 2005.

[15] C. Dong and B. Zhou, "Effectively detecting content spam on the web using topical diversity measures," in *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '12, (Washington, DC, USA), pp. 266–273, IEEE Computer Society, 2012.

[16] J. Abernethy, O. Chapelle, and C. Castillo, "Web spam identification through content and hyperlinks," in *Proceedings of the 4th International Workshop on*

Adversarial Information Retrieval on the Web, AIRWeb '08, (New York, NY, USA), pp. 41–44, ACM, 2008.

[17] T. Urvoy, T. Lavergne, and P. Filoche, "Tracking web spam with hidden style similarity," in *AIRWeb*, pp. 25–31, 2006.

[18] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen, "Combating web spam with trustrank," in *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB '04, pp. 576–587, VLDB Endowment, 2004.

[19] V. Krishnan, "Web spam detection with anti-trust rank," in *In AIRWEB*, pp. 37–40, 2006.

[20] B. Wu, V. Goel, and B. D. Davison, "Topical trustrank: Using topicality to combat web spam," in *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, (New York, NY, USA), pp. 63–72, ACM, 2006.

[21] J. Caverlee and L. Liu, "Countering web spam with credibility-based link analysis," in *Proceedings of the Twenty-sixth Annual ACM Symposium on Principles of Distributed Computing*, PODC '07, (New York, NY, USA), pp. 157–166, ACM, 2007.

[22] Y. I. Leon-Suematsu, K. Inui, S. Kurohashi, and Y. Kidawara, "Web spam detection by exploring densely connected subgraphs," in *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '11, (Washington, DC, USA), pp. 124–129, IEEE Computer Society, 2011.

[23] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri, "Know your neighbors: Web spam detection using the web topology," in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, (New York, NY, USA), pp. 423–430, ACM, 2007.

[24] Z. Gyongyi, P. Berkhin, H. Garcia-Molina, and J. Pedersen, "Link spam detection based on mass estimation," in *Proceedings of the 32Nd International Conference on Very Large Data Bases*, VLDB '06, pp. 439–450, VLDB Endowment, 2006.

[25] B. Wu and B. D. Davison, "Identifying link farm spam pages," in *Proceedings of the 14th International World Wide Web Conference*, pp. 820–829, ACM Press, 2005.

[26] Z. Cheng, B. Gao, C. Sun, Y. Jiang, and T.-Y. Liu, "Let web spammers expose themselves," in *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, (New York, NY, USA), pp. 525–534, ACM, 2011.

[27] Q. Gan and T. Suel, "Improving web spam classifiers using link structure," in *Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web*, AIRWeb '07, (New York, NY, USA), pp. 17–20, ACM, 2007.

[28] G. Geng, C. Wang, and Q. Li, "Improving web spam detection with re-extracted features," in *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, (New York, NY, USA), pp. 1119–1120, ACM, 2008.

[29] G.-G. Geng, Q. Li, and X. Zhang, "Link based small sample learning for web spam detection," in *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, (New York, NY, USA), pp. 1185–1186, ACM, 2009.

[30] S. Ghiam and A. N. Pour, "Detecting cloaking web spam using hash function," in *Computer Science and Information Technology*, vol. 1, pp. 33–40, Horizon Research, 2013.

[31] S. Webb, J. Caverlee, and C. Pu, "Predicting web spam with http session information," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, (New York, NY, USA), pp. 339–348, ACM, 2008.

[32] Y. Liu, M. Zhang, S. Ma, and L. Ru, "User behavior oriented web spam detection," in *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, (New York, NY, USA), pp. 1039–1040, ACM, 2008.

[33] Y. Liu, B. Gao, T.-Y. Liu, Y. Zhang, Z. Ma, S. He, and H. Li, "Browserank: Letting web users vote for page importance," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, (New York, NY, USA), pp. 451–458, ACM, 2008.

[34] C. Wei, Y. Liu, M. Zhang, S. Ma, L. Ru, and K. Zhang, "Fighting against web spam: A novel propagation method based on click-through data," in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, (New York, NY, USA), pp. 395–404, ACM, 2012.