Web spam detection: a survey

Un'introduzione alle varie tecniche di spam detection

Agenda

- Introduzione allo spam
- I vari tipi di spam detection
- Tecniche basate sul contenuto
- Tecniche basate sul grafo
- Tecniche che utilizzano altri segnali

Web spam

•Con il termine web spamming si fa riferimento alla volonta di ingannare i motori di ricerca in modo tale d’aumentare il ranking di alcune pagine rispetto ad altre.

•Questo fenomeno viene chiamato spamming o spamendixing.

Tecniche

- Ci sono due categorie di tecniche associate al web spam:
- **tecniche boost** che cercano di far avere più importanza o rilevanza a delle pagine
- **tecniche hiding** che sono metodi per nascondere le tencinche di boost all'utente dal browser

Tecniche di boost

Le tecninche di boosting si dividono in: **Term Spamming** e **Link Spamming**.
Con l’avvento degli algoritmi di ranking basati sulla struttura del grafo il Term Spaming è stato trascurato.

Term spaming

- Body spam
- Title spam
- Meta tag spam
- Anchor text spam
- URL spam

Link spamming

- Honeypot
- Infiltrarsi in una directory web
- Postare link nei blog
- Scambio di link
- Comprare domini scaduti
- Creare una spam farm

Click spamming

Dal momento che i motori di ricerca utilizzano i dati sul flusso di click per regolare le funzioni di ranking, gli spammers generano clik fraudolenti per manipolare il comportamento di queste funzioni in modo tale da fare avere un migliore rank ai loro siti.

Tecniche di hiding

Le tecninche di hiding si possono classificare in: content hiding, cloaking, redirection.

Tecniche basate sul contenuto

- Fetterly
- Martinez
- Castillo
- Benczur
- Urvoy

Fetterly: Euristiche

Faceno alcune analisi statistiche si vede che le pagine spam hanno proprietà diverse rispetto a quelle con contenuti.

Proprietà: URL

Il nome di un host con molti caratteri, punti, barre e numeri fanno molto probabiltmente riferimento a pagine spam. Un semplice classificatore può essere implementato attraverso l'utilizzo di una soglia.

Proprietà: Host name resolution

Alcuni motori di ricerca (Google) data una query, danno un rank più alto a un URL se contiene i termini della query. Gli spammers per sfruttare questo popolano le pagine con le URL le cui componenti contengono query popolari che sono rilevanti per un certo settore e impostano un DNS per risolvere questi host name.

Per determinare questa forma di spam basta vedere quanti nomi vengono risolti con lo stesso indirizzo.

Proprietà: Contenuto

Le pagine generate automaticamente hanno tutte lo stesso template, in particolare ci sono numerosi siti di spam che dinamicamente generano pagine che hanno uno stesso numero di parole.

Proprietà: Clustering

Una tecninca per determinare lo spam è quella di clusterizzare le pagine in base alla somiglianza dei template.

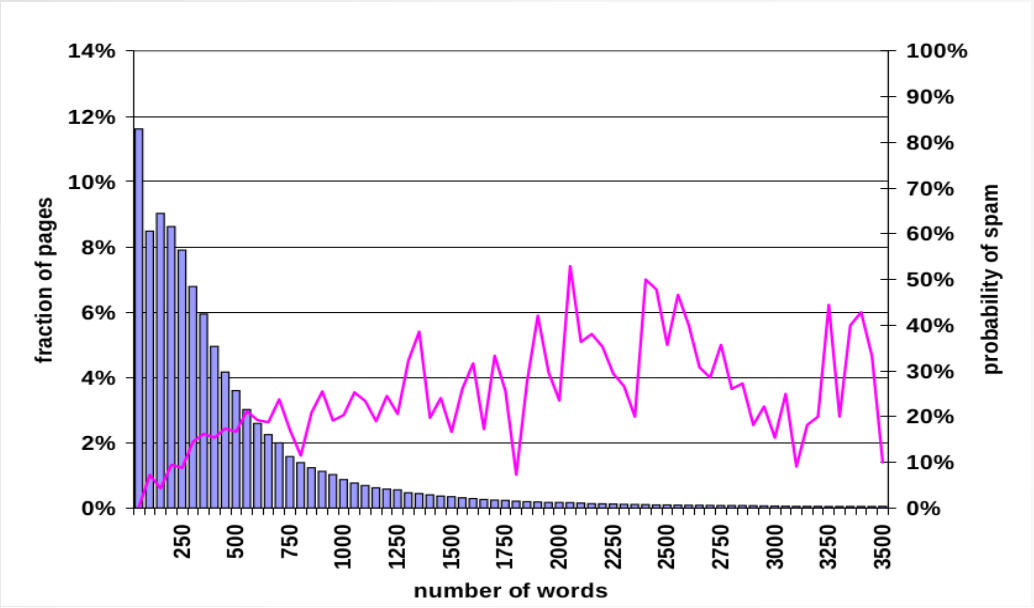
Detecting Spam Web Page

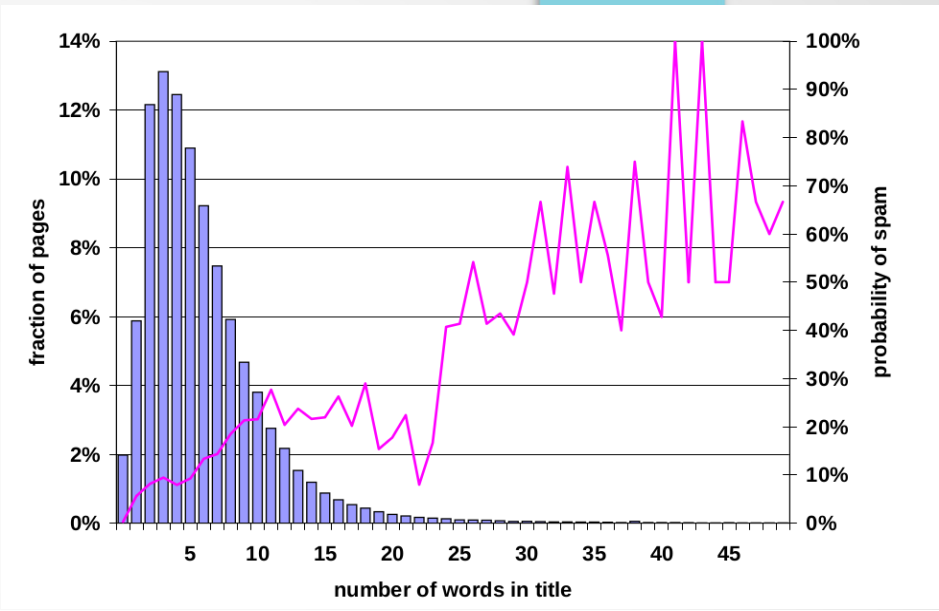
Nell'articolo vengono descritte delle altre tecniche per identificare lo spam. Inoltre viene presentato un metodo che combina le tecniche idividuali per creare un algoritmo più efficiente.

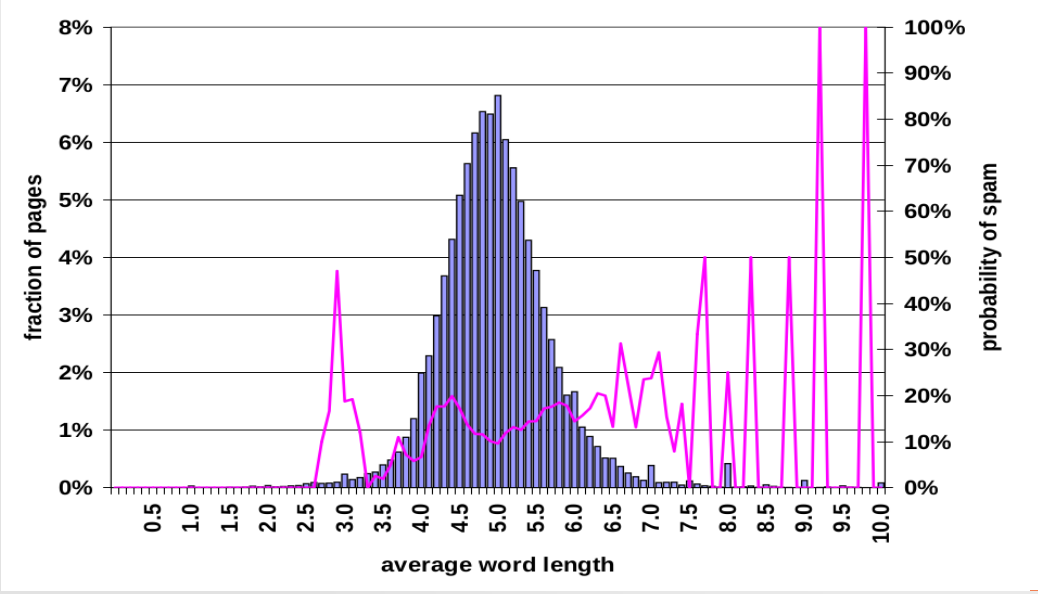
Spam categorizzato: Domini e Lingua

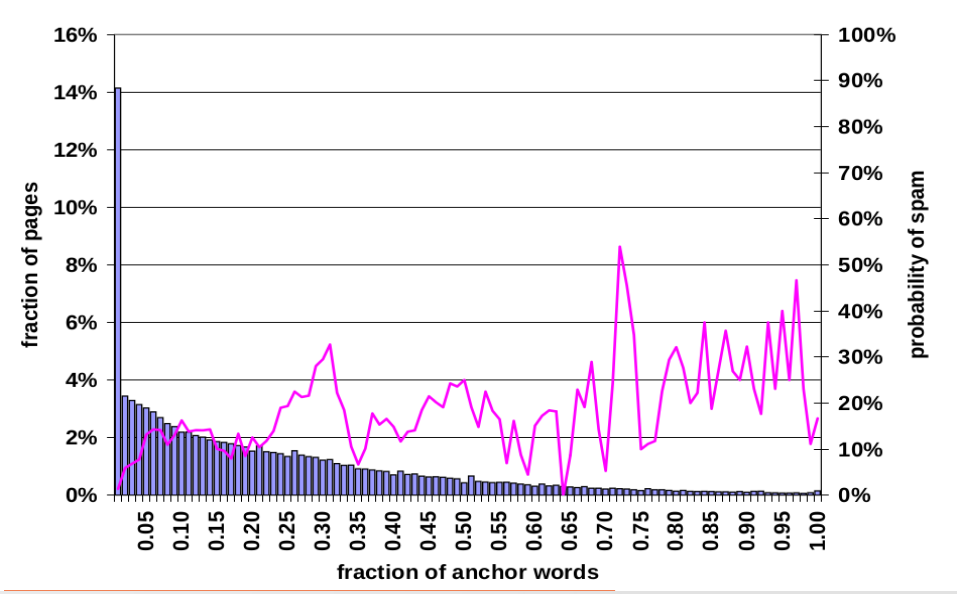
Dato il dataset ricavato da MSN crawler gli autori hanno identificato che i domini contenenti più spam sono il .biz, .us, .com. Per le lingue sono il francese,tedesco e inglese.

Altre euristiche

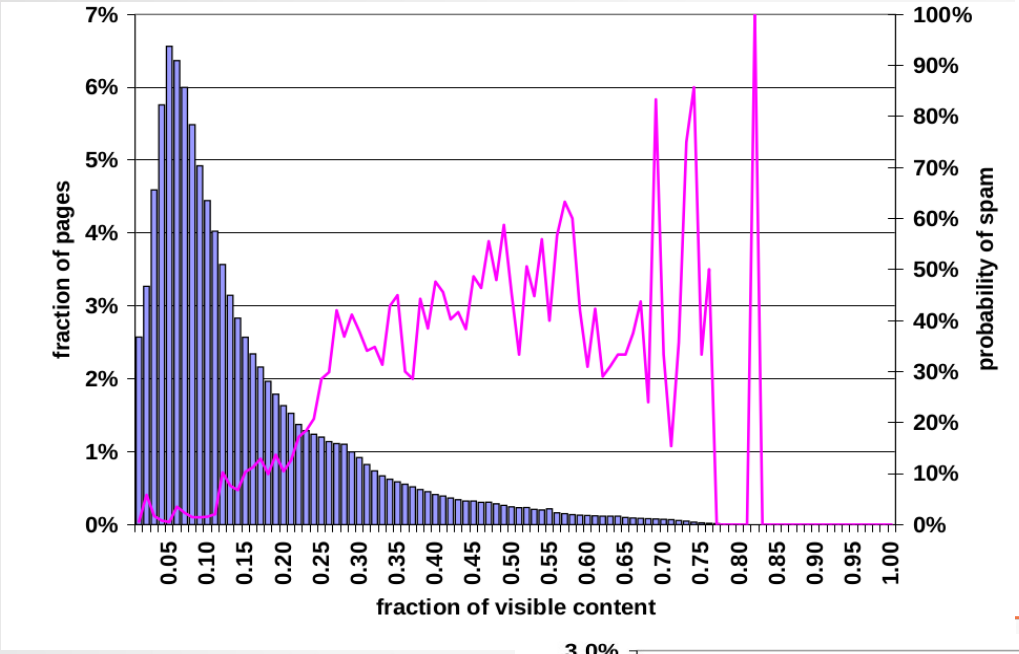


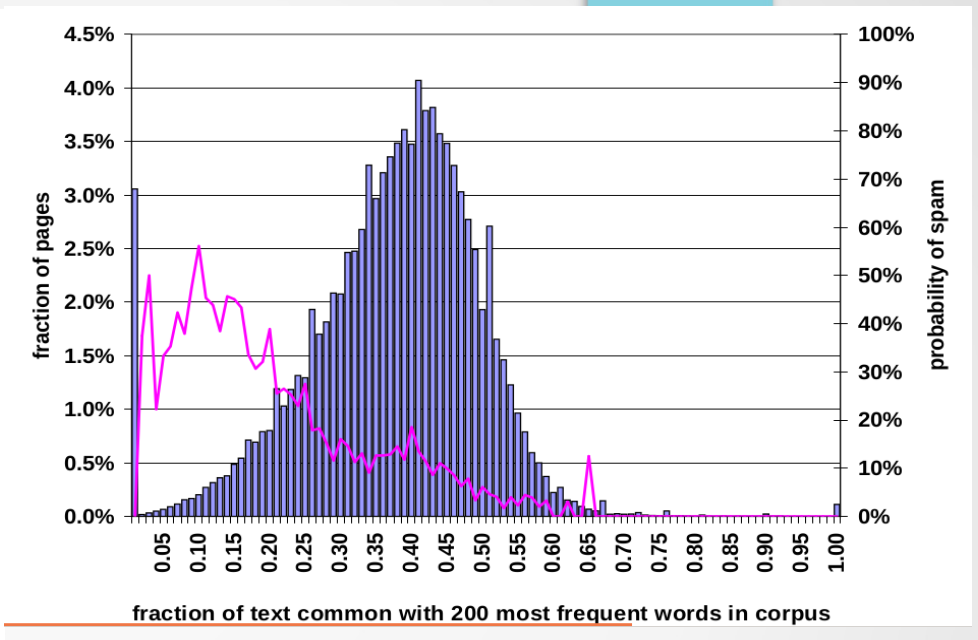


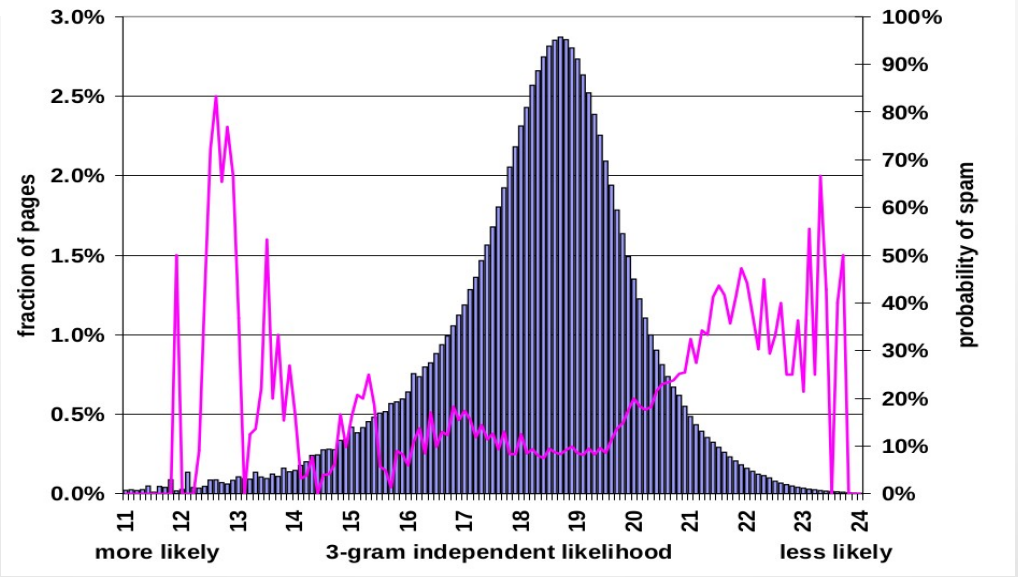




Altre euristiche



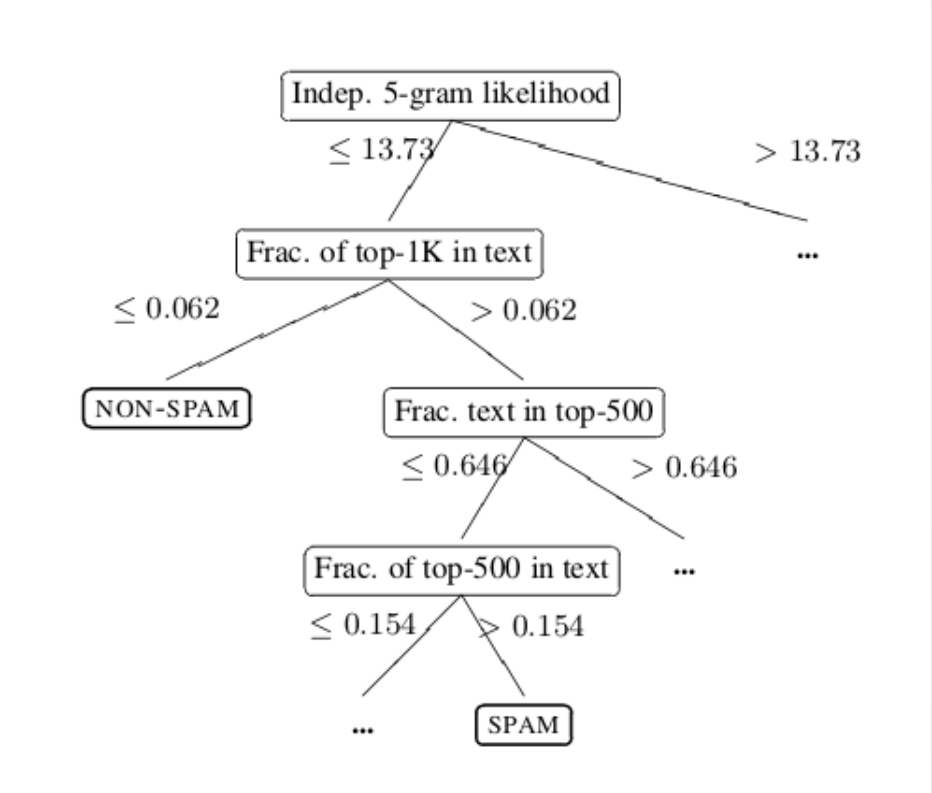




Classificatore

Viene utilizzato un classificatore per combinare le euristiche.

Viene utilizzato un classificatore di tipo “decision-tree”.



Risultati

class	recall	precision
spam	82.1%	84.2%
non spam	97.5%	97.1%

Ci sono delle tecniche per poter migliorare questi risultati come le tecninche bagging e boosting.

Rilevare la duplicazione delle frasi

Una delle euristiche menzionate era la ridondanza di contenuti che venia rilevata attraverso il rapporto di compressione.

Un'altro metodo è quello descritto nell'articolo “Detecting Phrase-Level Duplication on the World Wide Web”.

Rilevare la duplicazione delle frasi

Esaminando un dataset di pagine spam tedesche generate automaticamente, si è notato che sono create combinando contenuti presi da un corpus limitato.

Vegono definiti degli algoritmi per scoprire questo tipo di replica.

Le pagine create secondo questo metodo vengono generate al volo e cambiano compretamente a ogni download.

Dataset

Vengono utilizzati due dataset:

- Il primo dataset considerato consiste di 151 milioni di pagine HTML.
- il secondo consiste di 96 milioni di pagine HTML scelte in modo casuale dal insieme di pagine crawlate da MSN.

Campionamento

- Un documento viene ridotto ad un insieme di feature.
- Vengono rimpiazzati tutti i markup con spazi bianchi.
- Vegono definite parole del documento le sequenze massime di caratteri alfanumerici
- Vengono definite le k-frasi di un documento tutte le sequenze di k consecuti parole nel documento
- Il documento viene trattato come un cerchio: la sua ultima parola è seguita dalla sua prima.

Rabin fingerprint

- Il metodo di campionamento sfrutta alcune propriet`a della funzione Rabin fingerprint
- Una Rabin fingerprint tratta i bit di una string in input come coefficienti di un polinomio booleano
- Ci sono molte Rabin fingerprint ognuna delle quali è parametrizzata da un polinomio primitivo sopra l'anello dei booleani. Un polinomio primitivo *p* di grado *d* ha la proprietà che x^i *modulo p* non è uguale a x^j *modulo p* per tutti $0 < i < j < 2^d$

Algoritmo

L’algoritmo per creare i vettori di feature funziona in questo modo:

- vengono calolate la fingerprint per ogni parola nel documento (usando un polinomio primitivo p_a), in questo modo si riducono n-parole in n-token.
- calcoliamo le fingerprint di ogni frase di k-token (usando un polinomio primitivo p_b)
- Terzo applichiamo *m* differenti fingerprinting (con polinomi primitivi da p_1 a p_m) per ognuna delle frasi, tendendo le più piccole degli n risultati
- Questo processo infine crea un vettore di *m* fingerprinting che rappresentano il documento. Ci riferiamo al i-esimo elemento del vettore come l’i-esimo shingle del documento, e l’insieme degli elementi come shingles di tutto il vettore.

Duplicazione

- Vengono clasterizzate tutte le pagine dei data set in classi di equivalenza e ogni classe contiene tutte le pagine che sono esattamenente o quasi duplicate di un'altra
- In particolare nell'articolo sono state cercate le frasi più popolari cioè quelle che compaiono in più documenti. Queste frasi sono poco interessanti e riguardano termini legali, menu, frammenti javascritp
- Per determinare le frasi popolari si considerano le seguenti triple *i, s, d* dove *s* è l'i-esimo shingle del documento *d*. Vengono estratte tutte le triple corrispondenti all'interno della collezione. Vengono ordinate lessicografilmente e cercate le triple che matchano per i e s.
- Eseguendo alcuni studi sugli shingle più popolari si è notato che la distribuzione delle pagine spam tedesche mette in chiaro che esse hanno una maggiore concentrazione di shingle in comune con i dataset DS1 e DS2 presi in esame.
- Questa distribuzione caratterizza la replica delle frasi che abbiamo parlato in precedenza. Da qui segue un osservazione ovvero ci sono delle pagine web che sono donatrici delle loro parti per altre pagine

Bibliografia

- Spam, Damn Spam, and Statistics,Using statistical analysis to locate spam web pages, Dennis Fetterly, Mark Manasse, Marc Najork
- Detecting Spam Web Pages through Content Analysis, Alexandros Ntoulas, Marc Najork, Mark Manasse, Dennis Fetterly
- Detecting Phrase-Level Duplication on the World Wide Web, Dennis Fetterly, Mark Manasse, Marc Najork

Martinez

- Vengono proposte nuovi tipi di feature per rilevare lo spam.
- Vengono usati modelli di linguaggi per analizzare le sorgenti estratte da ogni sito nella collezione.
- Viene creato un modello del linguaggio per ogni sorgente e calcolata la differenza tra i due modelli da ogni altro modello
- Le sorgenti di informazioni usate sono:
  - testi delle ancore, testo vicino alle ancore della pagine sorgente
  - titolo e contenuto della pagina obbiettivo dello spam

Kullback-Leibler

- Per esaminare le differenze tra i modelli viene utilizzata la Kullback-Leibler (KL), una misura asimmetrica della divergenza la quale misura quanto male una distribuzione di probabilità M_q riesce a modellare M_d.

Classificazione

- I dataset utilizzati sono WEBSPAM-UK2006 e WEBSPAM-UK2007. Queste collezioni definiscono un’etichetta con cui si definisce una pagina facente parte della classe spam, non spam o non ben definita.
- Per quanto riguarda la classificazione delle pagine è stato usato Weka un software che contiene un insieme di algoritmi di machine learning e data mining.

Kullback-Leibler Divergence

- La Kullback-Leibler Divergence (KLD) viene utilizzata per misurare le divergenze tra le distribuzioni di probabilità dei termini di due documenti.
- Viene applicata a unità di testo della pagina sorgente e di quella linkata.

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

- dove in $P_{T1}(t)$ è la probabilità del termine *t* nella prima unità di testo e $P_{T2}(t)$ è la probabilità del tetmine *t* nella seconda unità di testo.
- Il modello del linguaggio usato stima la massima likelihood della probabilità dell'occorrenza dell'unigramma.

Feature

- Viene cercato di trovare una relazione tra due pagine collegate sulla base del loro 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 3 tipi di informazione dalla pagina che collega un’altra: testo delle ancore, testo intorno alle ancore, termini nell’URL.
- Inoltre vengono utilizzate 3 tipi di informazione per la pagina che viene linkata dalla pagina sorgente: titolo,contenuto della pagina, meta tag.

Feature usate

- testo delle ancore – contenuto

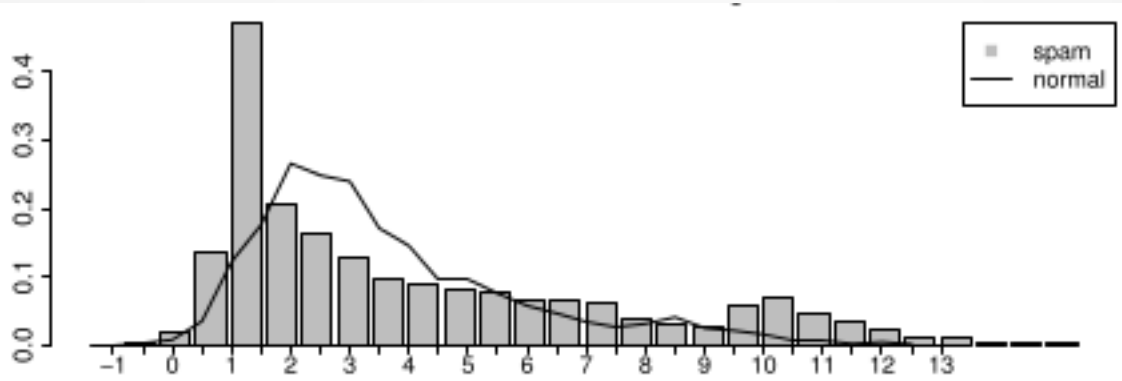


Figure 3: Histogram of KL divergence between Anchor Text and target Page Content. Reference collection is (WEBSPAM-UK2006) and external links have been used.

Feature usate

- testo vicino alle ancore – contenuto

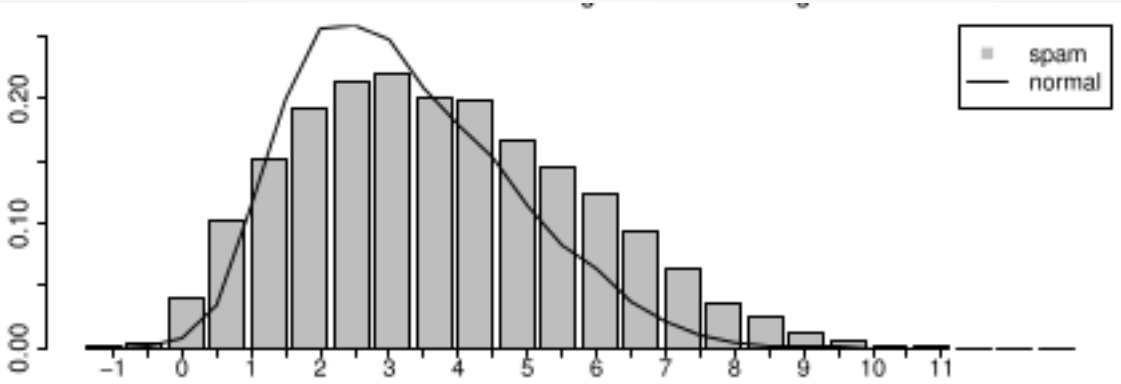


Figure 4: Histogram of KL divergence between Surrounding Anchor Text and target Page Content. Reference collection is (WEBSPAM-UK2006) and internal links have been used.

Feature usate

- termini nell'URL – contenuto

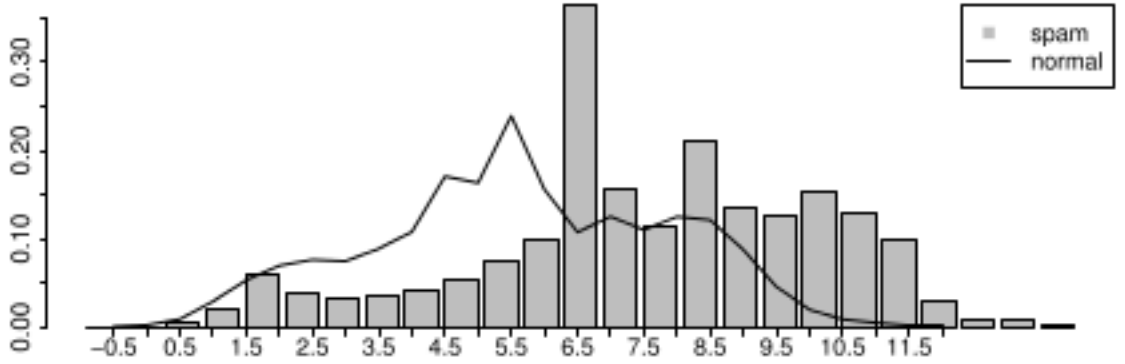


Figure 5: Histogram of KL divergence between Url Terms and target Page Content. Reference collection is (WEBSPAM-UK2007) and external links have been used.

Feature usate

- testo delle ancore – titolo

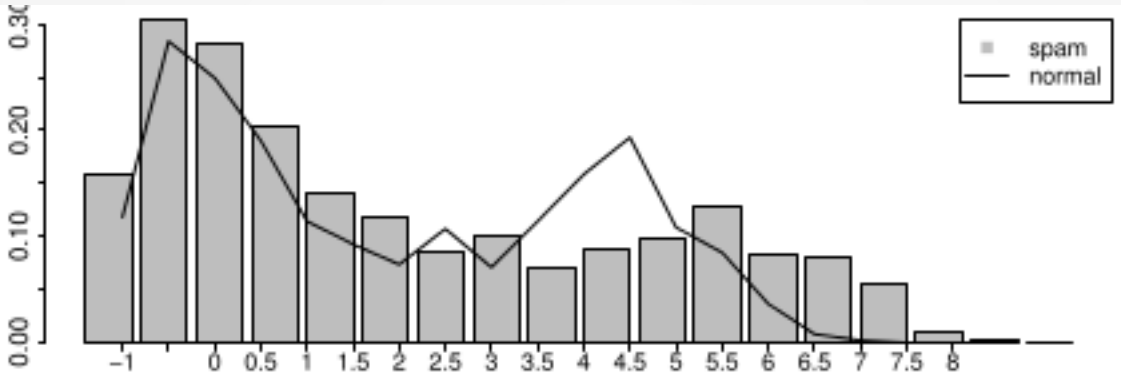


Figure 6: Histogram of KL divergence between Anchor Text and target Page Title. Reference collection is (WEBSPAM-UK2007) and both internal and external links have been used.

Feature usate

- testo intorno alle ancore – titolo

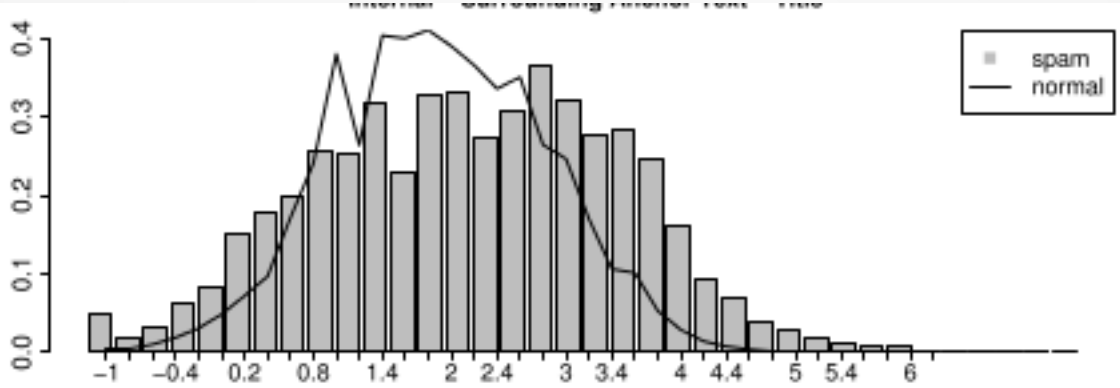


Figure 7: Histogram of KL divergence between Surrounding Anchor Text and target Page Title. Reference collection is (WEBSPAM-UK2006) and internal links have been used.

Feature usate

- termini nell'URL – titolo

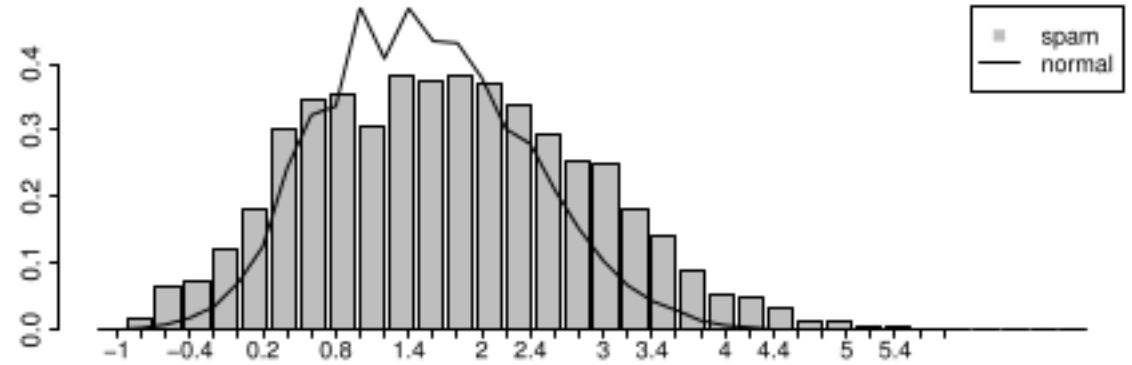


Figure 8: Histogram of KL divergence between Url Terms and target Page Title. Reference collection is (WEBSPAM-UK2006) and internal links have been used.

Feature usate

- titolo – contenuto

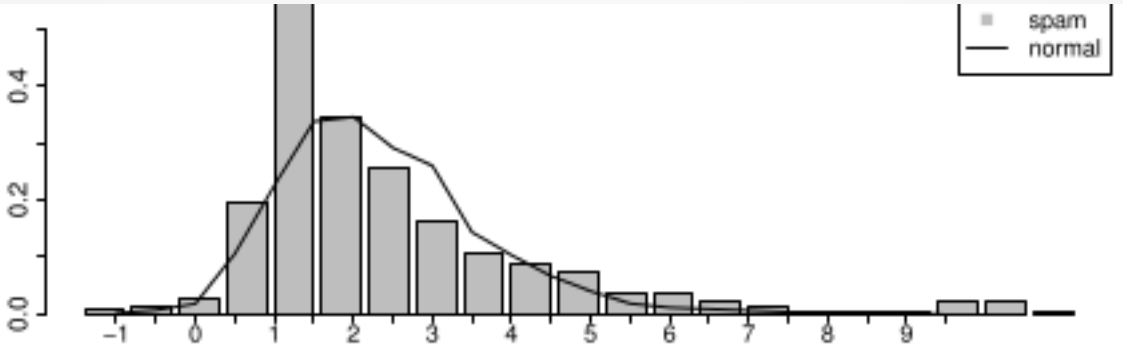


Figure 9: Histogram of KL divergence between both Title and Content from target Page. Reference collection is (WEBSPAM-UK2006) and external links have been used.

Feature usate

- titolo – contenuto

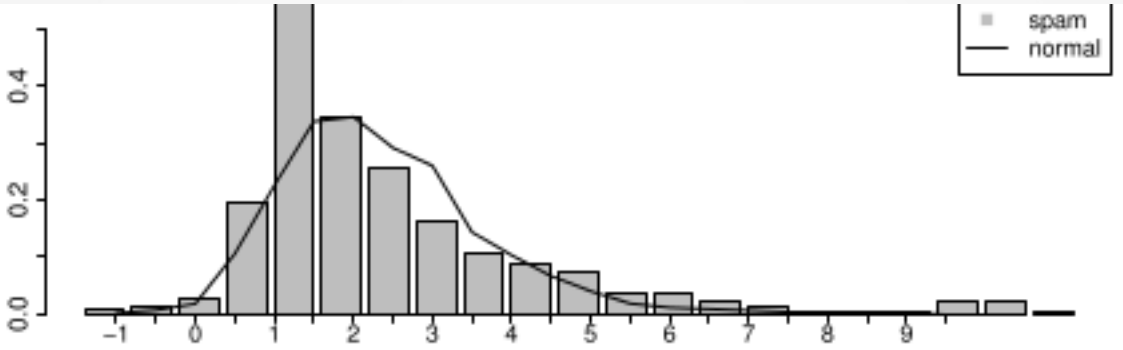


Figure 9: Histogram of KL divergence between both Title and Content from target Page. Reference collection is (WEBSPAM-UK2006) and external links have been used.

Feature usate

- meta-tag

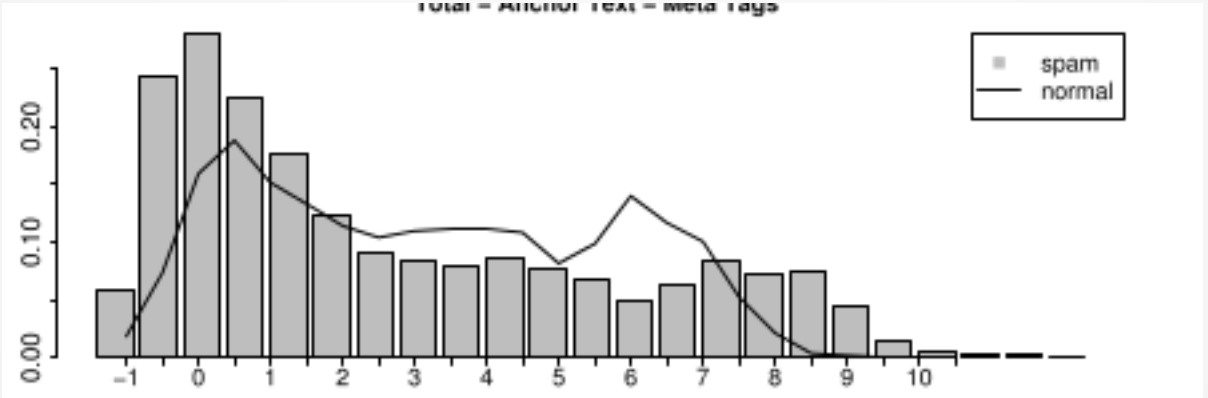


Figure 10: Histogram of KL divergence between Anchor Text and target Page Meta Tags. Reference collection is (WEBSPAM-UK2006) and both internal and external links have been used.

Risultati

WEBSPAM-UK2006					
Feature Set	Features	TP	FP	F	AUC
Content (C)	98	0.61	0.08	0.63	0.82
Link (L)	139	0.67	0.09	0.66	0.83
Lang. Models(LM)	42	0.43	0.05	0.55	0.76
C ∪ L	237	0.84	0.14	0.75	0.85
C ∪ LM	140	0.58	0.09	0.61	0.81
L ∪ LM	181	0.84	0.20	0.69	0.83
C ∪ L ∪ LM	279	0.87	0.11	0.81	0.86

Table 2: Features, True Positive rate (TP), False Positive rate (TP), F-measure (F) and Area Under Roc Curve (AUC) for Web Spam classifiers using different feature sets on UK-2006.

WEBSPAM-UK2007					
Feature Set	Features	TP	FP	F	AUC
Content (C)	98	0.33	0.04	0.30	0.72
Link (L)	139	0.39	0.12	0.20	0.68
Lang. Models(LM)	42	0.24	0.04	0.24	0.72
C ∪ L	237	0.31	0.03	0.31	0.73
C ∪ LM	140	0.37	0.05	0.30	0.72
L ∪ LM	181	0.42	0.12	0.22	0.70
C ∪ L ∪ LM	279	0.33	0.03	0.33	0.75

Table 3: Features, True Positive rate (TP), False Positive rate (TP), F-measure (F) and Area Under Roc Curve (AUC) for Web Spam classifiers using different feature sets on UK-2007.

Bibliografia

- Web Spam Identification Through Language Model Analysis, Juan Martinez-Romo, Lourdes Araujo

Zhou

- Viene fatta un’analisi su il contenuto spam usando dei modelli specifici e vengono proposte delle misure specifiche di diversità per identificare le pagine spam.
- Viene descritto un metodo su statistiche sulla base di argomenti.
- Non utilizza statistiche basate sulle parole della pagina le quali ignorano la semantica tra di esse.
- Il metodo basato su statistiche degli argomenti può catturare le feature linguistiche nascoste nel testo per capire se la pagina è buona.
- Le analisi vengono fatte usando dei modelli degli argomenti come la Latent Dirichlet Allocation (LDA) che sono modelli statistici dei linguaggi per scoprire argomenti nascosti che compaiono in una collezione di documenti.

Topic model

- Un topic model è un modello statistico che scopre gli argomenti latenti in una collezione di documenti.
- In generale LDA modella ogni argomento latente come una distribuzione probabilistica su un vocabolario e ogni documento come una distribuzione probabilistica sugli argomenti latenti.

Topical diversity measure

- Se analizziamo le feature nascoste di una pagina spam, possiamo notare che i contenuti di queste pagine che sono generati automaticamente, hanno delle differenze con altri contenuti delle pagine che non sono spam.
- Da questa intuizione è stato proposto di analizzare il contenuto delle pagine web usando i topic models.

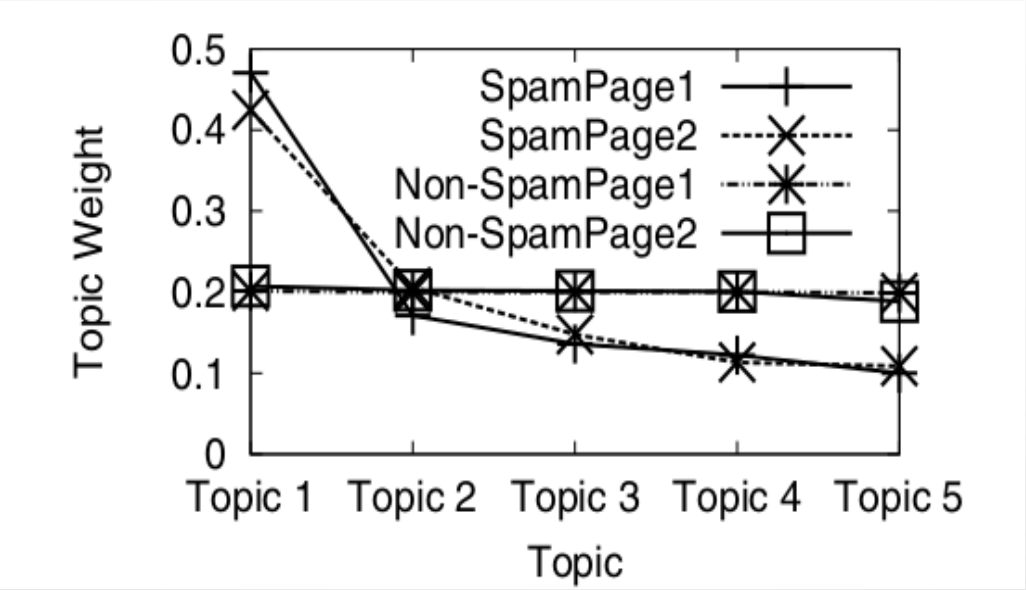
- Dopo avere usato un topic model su un testo come il contentuo di una pagina web possiamo scoprire e caratterizzare gli argomenti nascosti all’interno del testo.
- A ogni argomento latente è associato con un corrispondente peso che indica quanto frequentemente questo argomento compare nel corpus, in generale otteniamo una distribuzione di argomenti per il testo disponibile.
- Per rilevare il contentuto spam, l’obbiettivo è capire se le pagine spam e non spam hanno un’unica caratteristica in termini di distribuzione degli argomenti.

Processo

- Data una pagina *d* possiamo rappresentare il contentuto della pagina usando il modello di bag-of-word.
- Il vocabolario di *d* che consiste di tutte le parole in *d* è rappresentato come *W* . Allora si può applicare LDA direttamente su *d* per ottenere gli *m* argomenti latenti denotati come t_1 , t_2 , ..., t_m , dove ogni argomento t_i è una distribuzione di parole φ_{ti} .
- Per ogni parola $w \in W$ la probabilità che compaia in un argomento t_i viene definita come: $\varphi(w|t_i)$.
- Usiamo $W(t_i)$ per rappresentare l'insieme delle parole che hanno una probabilità rispetto a t_i più grande di 0.
- Il peso di ogni argomento t_i è uguale a $\sum_{w \in W(t_i)} \phi(w|t_i)$

Distribution-based Topical Diversity Measure

- le pagine spam sono molto topic-centric, ovvero loro hanno uno specifico insieme di argomenti centrali.
- Intuitivamente le pagine di spam e non spam hanno distribuzioni degli argomenti differenti

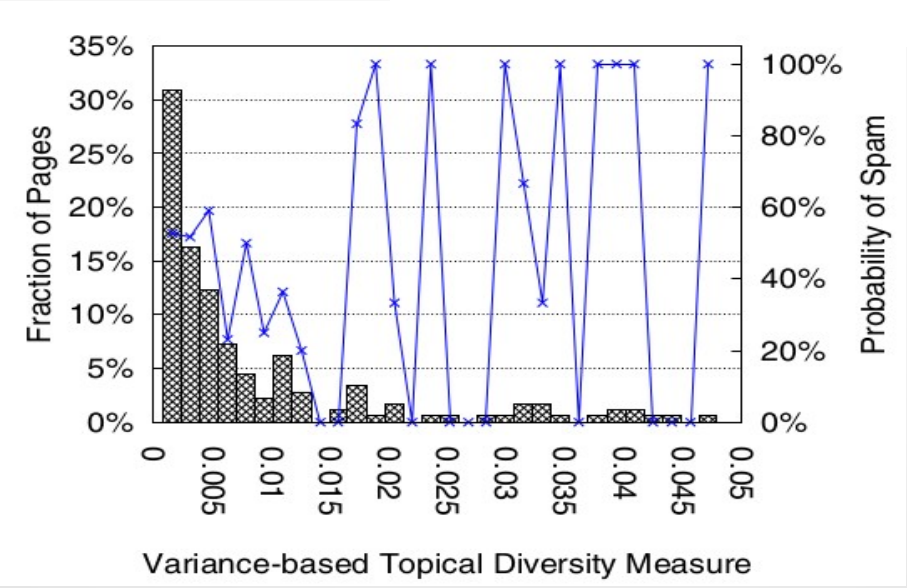


Distribution-based Topical Diversity Measure

- Per catturare questa caratteristica sulle distribuzioni dei pesi degli argomenti per le pagine spam e normali, è stato proposta una misura della diversità degli argomenti basata sulla varianza.

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

- u=1/m



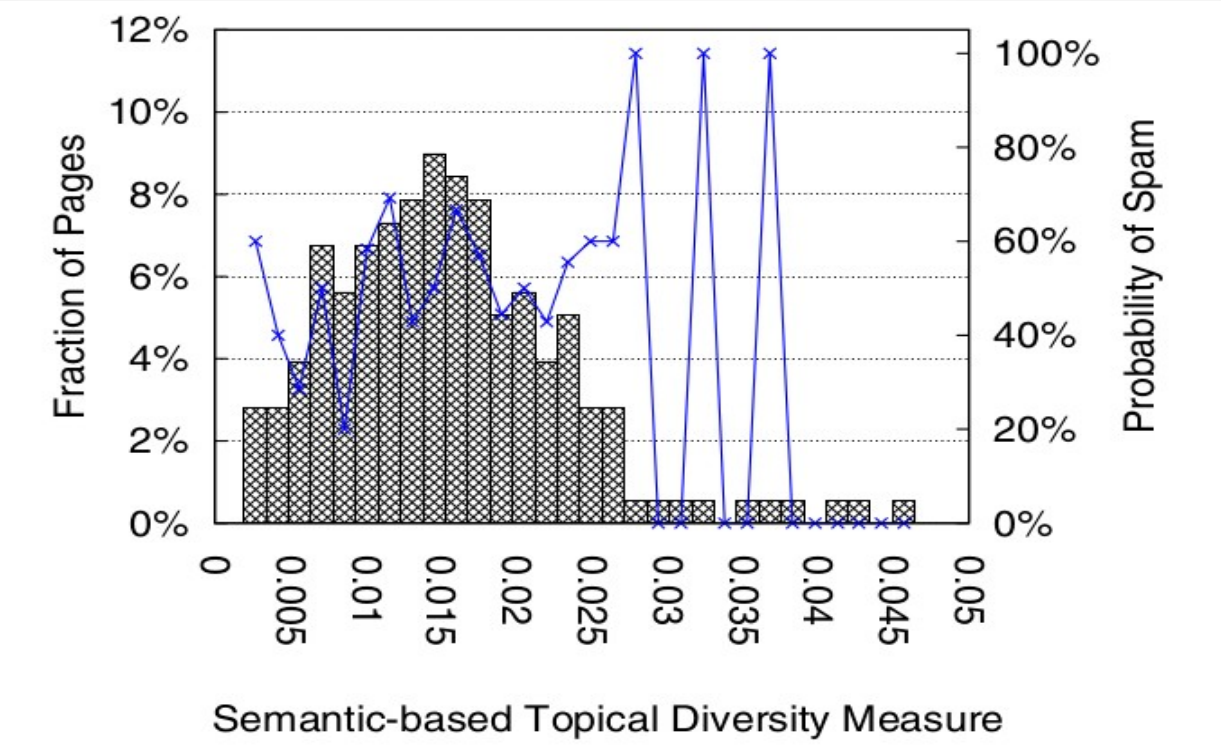
Semantic-based Topical Diversity Measure

- La misura della diversità degli argomenti basata sulla distribuzione considera solamente la distribuzione dei pesi degli argomenti e non se gli argomenti sono semanticamente relativi
- Chi crea le pagine spam ha come obiettivo di aumentare il rank delle pagine perciò è facile pensare che molte pagine spam sono semanticamente relative.
- Un modo per misurare la relazione semantica tra gli argomenti è misurare la semantica tra le parole.
- Viene utilizzata una funzione di similarità Sim(wi , wj) per ottenere le relazioni semantiche tra due parole. La funzione di similarità usata è quella di Wordnet.
- Per misurare la relazione semantica tra due argomenti ti , tj , possiamo ottenere le similarità tra ogni coppia di parole dei due argomenti moltiplicate con le loro probabilità rispetto agli argomenti, per ottenere la media delle similiratà.

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

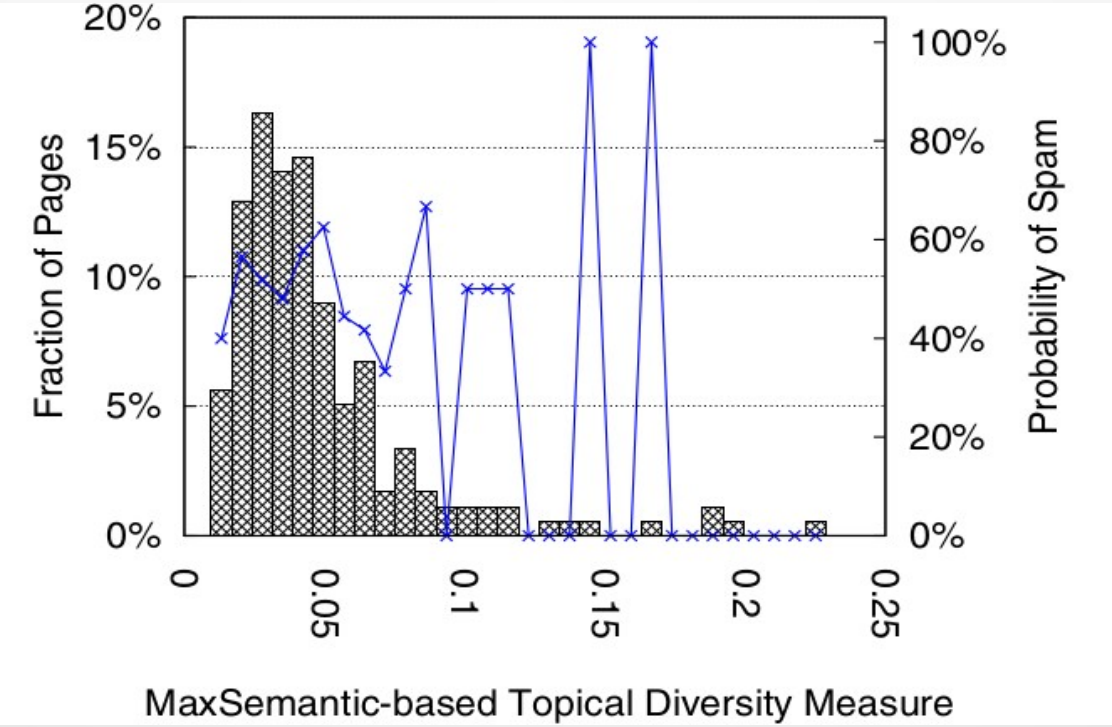
Misura della diversità degli argomenti basata sulla semantica

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



Misura di diversità basata sulla massima semantica

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



Valutazione

Parameters in LDA		TPR	FPR	F-Measure	AUC
#topics	#words				
2	3	0.933	0.188	0.779	0.784
2	5	0.893	0.185	0.796	0.838
3	3	0.943	0.179	0.787	0.852
3	5	0.908	0.151	0.838	0.884
5	3	0.954	0.176	0.791	0.898
5	5	**0.965**	**0.133**	**0.850**	**0.921**

Table I

Algorithm	Feature Set	TPR	FPR	F-Measure	AUC
RandomForest	Content	0.953	0.089	0.907	0.931
	Topic	**0.965**	0.133	0.850	0.921
	All	0.959	**0.078**	**0.918**	**0.949**
RandomTree	Content	0.965	0.151	0.825	0.835
	Topic	**0.977**	0.162	0.806	0.811
	All	0.966	**0.139**	**0.841**	**0.843**

Table II

Bibliografia

- Effectively Detecting Content Spam on the Web Using Topical Diversity Measures, Cailing Dong, Bin Zhou

Castillo

- Viene presentato un algoritmo (WITCH) che impara a rilevare lo spam degli host o di pagine sul web.
- Analizza la struttura del grafo del web e il contenuto della pagina.
- Le pagine spam e non spam hanno differenti proprietà statistiche le quali possono essere sfruttate per costruire un classificatore. Ma oltre a queste feature standard da analizzare bisogna analizzare la struttura dei collegameti tra le pagine.

Algoritmo

- Assumiamo che abbiamo le seguenti caratteristiche:
  - un insieme *l* di esempi etitchetati $(x_1 , y_1), ..., (x_l , y_l)$, dove x_i denota il vettore di feauter associato all i-esimo host e y_i è la sua etichetta: -1 per non spam e +1 per spam
  - un insieme *u* di esempi non etichettati, $x_l +1 , ..., x_n$, con $n = l + u$
  - un grafo diretto pesato con nodi $x_1, ..., x_n$
  - *E* l'insieme delle coppie (i, j) dove il nodo i è connesso al nodo j e a_{ij} il peso del link da x_i a xj

Algoritmo

- Supponiamo volessimo istruire un classificatore lineare f(x) = w · x

$$\Omega(w) = \frac{1}{l} \sum_{i=1}^{l} R(w \cdot x_i, y_i) + \lambda w \cdot w$$

Algoritmo

- Trarre vantaggio dai link tra le pagine che rappresentano il grado di similarità tra sorgente e destinazione

$$\Omega(w) = \frac{1}{l}\sum_{i=1}^{l} R(w \cdot x_i, y_i) + \lambda w \cdot w + \gamma \sum_{(i,j)\in E} a_{ij}\Phi(w \cdot x_i, w \cdot x_j)$$

- Nel caso in cui lo spazio delle feature non è abbastanza grande un sempice classificatore lineare non è abbastanza flessibile. Perciò viene introdotto un nuovo parametro z_i per ogni nodo i e il classificatore viene istruito: $f(x_i) = w \cdot x_i + z_i$.

$$\Omega(w) = \frac{1}{l}\sum_{i=1}^{l} R(w\cdot x_i + z_i, y_i) + \lambda_1 w\cdot w + \lambda_2 z\cdot z + \gamma \sum_{(i,j)\in E} a_{ij}\Phi(w\cdot x_i + z_i, w\cdot x_j + z_i)$$

Algorithm 1 WITCH

Params: $\lambda_1, \lambda_2, \gamma$, convex function $\Phi(\cdot, \cdot)$
Input: labeled training set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)$
Input: unlabeled set $\mathbf{x}_{l+1}, \ldots, \mathbf{x}_n$
Input: hyperlink graph E with edge weights $\{a_{ij}\}_{(i,j)\in E}$
Solve:

$$\mathbf{w}, \mathbf{z} \leftarrow \arg\min_{\mathbf{w},\mathbf{z}} \Omega(\mathbf{w}, \mathbf{z}),$$

with $\Omega(\cdot)$ defined in (3).
Predict: label node i as $\text{sign}(\mathbf{w} \cdot \mathbf{x}_i + z_i)$.

Risultati

- I parametri hanno un influenza sulle prestazioni dell'algoritmo.

Table 1: Results summary with two training sets.

Training Algorithm	AUC 10%	AUC 100%
SVM + stacked g.l.	0.919	0.953
Link based (no features)	0.906	0.948
Challenge winner	–	0.956
Only Features	0.859	0.917
Features + GR	0.874	0.917
Slack + GR	0.919	0.954
WITCH (Feat. + Slack + GR)	**0.928**	**0.963**

Bibliografia

- Web spam Identification Through Content and Hyperlinks, Jacob Abernethy, Olivier Chapelle, Carlos Castillo
- WITCH: A NEW APPROACH TO WEB SPAM DETECTION, Jacob Abernethy, Olivier Chapelle, Carlos Castillo
- Topical Locality in the Web: Experiments and Observations ∗, Brian D. Davison

Urvoy

- Le pagine spam che vengono generate automaticamente non sono facili da rilevare usando metodi di classificazione classici basati sul contentuto
- Viene presentato un nuovo metodo chiamato “(hidden) style similarity measure” basato su feature testuali extra al codice html
- Inoltre è descritto un metodo per clusterizzare una grande collezione di documenti sulla base di questa nuova misura
- Questo metodo clusterizza le pagine che hanno uno stile in comune, perciò risulta ottimo nell'identificare spam

Rilevare la similarità

- L'identificazione della similarità testuale normalmente fa uso di feature basate sulle parole
- Ma visto che le pagine sono generate automaticamente difficilemente condivideranno gli stessi vocaboli
- Per rilevare la similarità basandosi sul metodo di generazione abbiamo bisogno di feature più simili alla strttura interna html
- Questo porta a studiare lo stile di una pagina più che il contenuto

Algoritmo HSS

- Per catturare la similarità basandosi sul metodo di generazione delle pagine viene utilizzato un processo che esclude tutti i caratteri alfanumerici e tiene conto di quelli rimanenti attraverso l'uso degli n-grammi
- Analizzando le feautre html come: spazzi, tag ecc. riusciamo a modellizzare lo stile di un pagina
- Questo modello consente di confrontare e raggruppare insieme documenti che condividono molte feature nascoste

Algoritmo HSS

- Vengono considerati: HS-similarity (one to one) e HS-clustering (global)

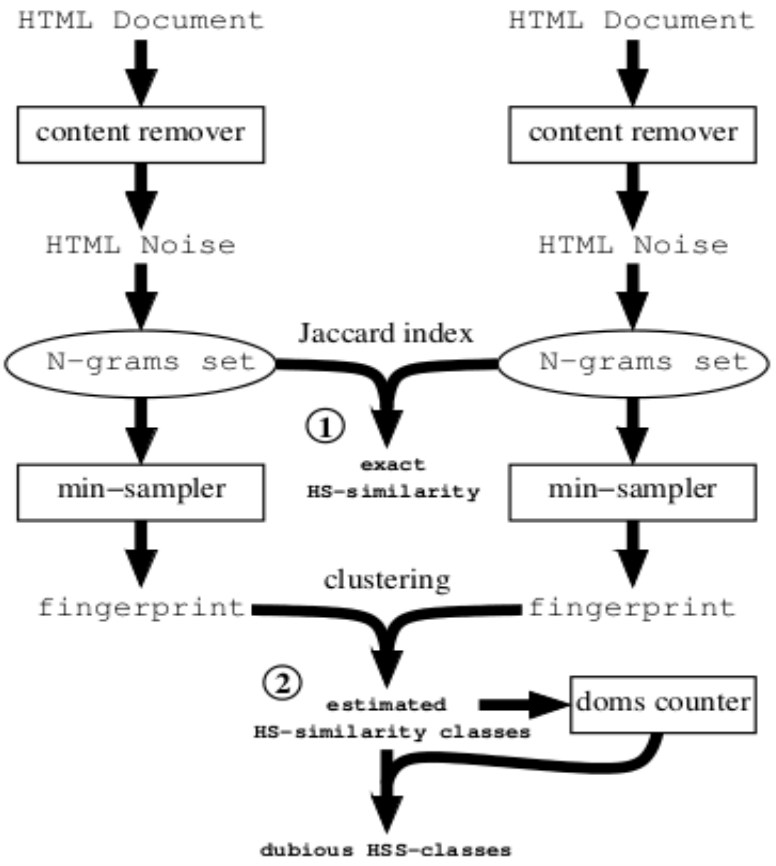


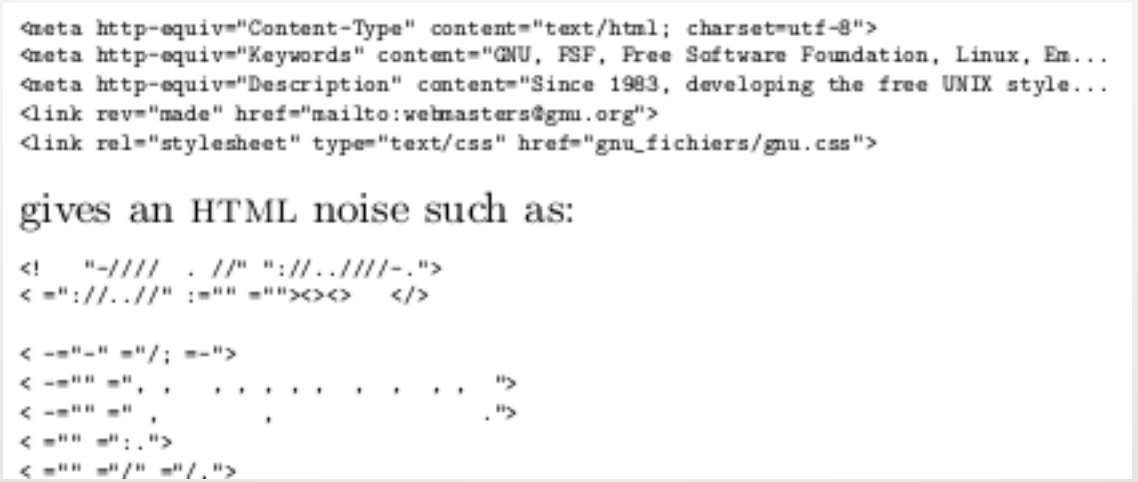
Figure 2: The HSS algorithm: step (1) describes one-to-one full document HS-similarity computation, step (2) describes large scale similarity classes calculus and link farm detection.

Algoritmo HSS

- L'algoritmo è efficiente per caratterizzare documenti che provengono dallo stesso sito senza avere informazioni sull'host o URL
- Un risultato importate è che le classi di similarità contentdono pagini di differenti domini

Preprocessing

- Il primo passo per confrontare due documenti è eliminare tutto ciò non riflette il loro conentuto
- Nel caso di documenti html: tag, spazi extra, stop word
- L'approccio descritto fa il contrario ovvero tiene tutto ciò che è rumore di un documento html ed elimina i caratteri alfanumerici



```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="Keywords" content="GNU, FSF, Free Software Foundation, Linux, Em...
<meta http-equiv="Description" content="Since 1983, developing the free UNIX style...
<link rev="made" href="mailto:webmasters@gnu.org">
<link rel="stylesheet" type="text/css" href="gnu_fichiers/gnu.css">
```

gives an HTML noise such as:

```
<!   "-////  . //" "://..////-.">
< =":///..//" :="" =""><><>   </>

< -="-" ="/; =-">
< -="" =", .   . . . . . .  . .  . .  ">
< -="" =" .        .                .">
< ="" =":.">
< ="" ="/" ="/.">
```

Similarità

- Per ragioni computazionali si è scelto di usare una distanza basata sull'intersezione degli insiemi con gli n-grammi che si sovrappongono
- Una semplice HS-similarity può essere calcolata anche con Jaccard

Fingerprint

- Viene utilizzata per ragioni computazionali

Procedure 1 Insert a string s by minsampling into a fingerprint $V \in \mathbb{N}^m$.

Require: $m > 0$ and V initialized

$h := preHash(s)$
$i := h \mod m$
$h' := \sigma_i(h)$
if $h' < V[i]$ **then**
$V[i] := h'$
end if

Clustering

- Grafi di similarita sono caratterizzati dalla proprietà quasi-transitiva: se xSy e ySz allora è probabile che xSz.
- Questa proprietà è di aiuto per velocizzare il processo
- Degli elementi potrebbero essere utilizzati per decidere se altri elementi sono nella stessa classe

Procedure 2 Search HS-similarity classes

Require: $0 < k, p, t \leq m$
  init similarity graph;
  for $i := 0$ to p **do**
    pick a k-subset $s \subseteq [m]$;
    for all pairs (x, y) of fingerprints matching according to s **do**
      if $Sim_b(x, y) > t$ **then**
        add edge (x, y) to similarity graph;
      end if
    end for
  end for
  compute *Clusters* from connected components of the graph;

Bibliografia

- Tracking Web Spam with Hidden Style Similarity, Tanguy Urvoy, Thomas Lavergne, Pascal Filoche
- Tracking Web Spam with HTML Style Similarities,TANGUY URVOY, EMMANUEL CHAUVEAU, and PASCAL FILOCHE

Benczur

- Raccoglie un insieme di feature per web spam
- Spiegazione di alcune tecniche di machine learning: “essemble selection”, “LoginBoost”, “RandomForest”

Dataset

	UK2006	UK2007	DC2010			
			en	de	fr	all
Hosts	10 660	114 529	61 703	29 758	7 888	190 000
Spam	19.8%	5.3%	8.5% of valid labels; 5% of all in large domains.			

Table 1: Fraction of Spam in WEBSPAM-UK2006 and UK2007 as well as in DC2010. Note that three languages English, German and French were selected for labeling DC2010, although Polish and Dutch language hosts constitute a larger fraction than the French.

Count	IP address	Comment
3544	80.67.22.146	spam farm `*-palace.eu`
3198	78.159.114.140	spam farm `*auts.eu`
1374	62.58.108.214	`blogactiv.eu`
1109	91.204.162.15	spam farm `x-mp3.eu`
1070	91.213.160.26	spam farm `a-COUNTRY.eu`
936	81.89.48.82	`autobazar.eu`
430	78.46.101.76	spam farm `77k.eu` and 20+ domains
402	89.185.253.73	spam farm `mp3-stazeni-zdarma.eu`

Table 2: Selection of IP addresses with many subdomains in the DC2010 data set.

Label	Yes	Maybe	No
Spam	423		4 982
News/Editorial	191		4 791
Commercial	2 064		2 918
Educational	1 791		3 191
Discussion	259		4 724
Personal-Leisure	1 118		3 864
Non-Neutrality	19	216	3 778
Bias	62		3 880
Dis-Trustiness	26	201	3 786
Confidence	4 933		49
Media	74		4 908
Database	185		4 797
Readability-Visual	37		4 945
Readability-Language	4		4 978

Table 3: Distribution of assessor labels in the DC2010 data set.

Feature

- Le feature usate sono quelle descritte in: C. Castillo, D. Donato, L. Becchetti, P. Boldi, S. Leonardi,M. Santini, and S. Vigna. A reference collection for web spam. SIGIR Forum, 40(2):11–24, December 2006

Ensemble Selection

- E' un metodo che consente di utilizzare grandi collezioni di diverisi classificatori
- Vantaggi:
  - Ottimizzazioni delle perfomance
  - Raffinamente dei precendeti overfitting
- Usato per combinare un gran numero di classificatori
- Instead of tuning various parameters of different classifiers, we can concentrate on finding powerful features and selecting the main classifier models which we believe to be able to capture the differences between the classes to be distinguished.

Learning Methods

- Vengono usati i seguenti tipi di modello per costruire la libreria per “essemble selection”: bagged and boosted decision tree, logistic regression, naive Bayes, random forest
- For most classes of features we use all classifiers and allow selection to choose the best ones

Risultati

Feature Set	No. of Features	UK2007 AUC	DC2010 AUC	DC2010 NDCG
content (A)	74	0.859	0.757	0.762
content (Aa)	24	0.841	0.726	0.732
content (B)	96	0.879	0.799	0.803
BM25 + (B)	10096	**0.893**	**0.891**	**0.893**

Table 5: Performance of ensembles built on content based features.

Feature Set	No. of Features	UK2007 AUC	DC2010 AUC	DC2010 NDCG
link	177	0.759	0.587	0.621
all	10 273	**0.902**	0.885	0.888

Table 6: Performance of ensembles built on link based and all features.

Bibliografia

- Web Spam Classification: a Few Features Worth More Miklós Erdélyi, András Garzó, András A. Benczúr
- A Reference Collection for Web Spam Carlos Castillo , Debora Donato, Luca Becchetti, Paolo Boldi, Stefano Leonardi, Massimo Santini and Sebastiano Vigna

Tecninche basate su link

- Gyongyi
- Caverlee
- Davison
- Krishan
- Wu
- Altri

Gyongy: trustrank

- Gli autori propongono una tecnica semi-automatica per separare le pagine buone da quelle di spam
- Prima viene selezionato un insieme di pagine seed per essere valutate da un esperto. Una volta identificate le pagine buone, viene utilizzata la struttura dei link per scoprire pagine che sono probabilmente buone

Funzione Oracle

- Gli autori formalizzano il lavoro fatto da una perona per determinare se una pagina è spam oppure no con una funzione binaria Oracle

$$O(p) = \begin{cases} 0 & if \quad p \quad is \quad bad \\ 1 & if \quad p \quad is \quad good \end{cases}$$

Funzione Oracle

- Per scoprire le pagine buone senza ivocare la funzione Oracle su tutte le pagine bisogna fare un osservazione empirica che gli autori chiamano isolazione approssimata dell’insieme delle pagine buone: le pagine buone raramente puntano a quelle cattive (oss. ma in caso di honey pot) perché le persone che creano pagine buone hanno poco interesse a linkare pagine chehanno un contenuto costruito per aumentare il rank
- Ma alcune volte i creatori di pagine buone sono ingannati

Funzione trust

- Per non usare la funzione Oracle su tutto ill web gli autori stimano la verosomiglianza che una pagina *p* sia normale
- La nuova funzione è la trust function *T* che ritorna un range di valori tra 0 (cattivo) e 1 (normali).
- Idealmente per delle pagine *p*, *T(p)* ritornerebbe la probabilità che *p* sia buona

Proprietà trust

- Proprietà Ideale TrusRank
  - $T(p) = P\,r[O(p) = 1]$
- Proprietà di ordinamento di TrustRank
  - $T(p) < T(q) \leftrightarrow P\,r[O(p) = 1] < P\,r[O(p) = 1]$
  - $T(p) = T(q) \leftrightarrow P\,r[O(p) = 1] = P\,r[O(p) = 1]$
- Proprietà di soglia di TrustRank
  - $T(p) > \delta \leftrightarrow O(p) = 1$

Calcolare la verità

- Gli autori denotano con S^+ il sottoinsieme di pagine dell'insieme seed che sono normali mentre con S^- le pagine cattive, ricavati tramite la funzione Oracle
- Le pagine che non sono controllare da una pesona sono constrassegnate con un valore di verità uguale a ½
- Questo schema è chiamato funzione di verità ignorante T_0 definita per ogni pagina $p \in V$

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

Propagazione della verità

- Come passo successivo nel calcolare il punteggio di verità prendiamo vantaggio nell'usare la proprietà dell'isolamento approssimata delle pagine normali
- Assegnamo 1 a tutte le pagine che possono essere raggiunte da una pagina in S^+ in M step
- Dagli esperimenti si nota che quanto più è grande M la precision e la recall diminuiscono
- La funzione di verità T_M è definita come:

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

Attenuazione della verità

- Per attenuare il valore di verità quanto più si va avanti durante il cammino si possono utilizzare:
  - Trust dampening
  - Trust splitting

Algoritmo

function TrustRank
input
- $\mathbf{T}$ 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
- $\mathbf{t}^*$ TrustRank scores

begin

// evaluate seed-desirability of pages

(1) $\mathbf{s} = \mathsf{SelectSeed}(\ldots)$

// generate corresponding ordering

(2) $\sigma = \mathsf{Rank}(\{1,\ldots,N\},\mathbf{s})$

// select good seeds

(3) $\mathbf{d} = \mathbf{0}_N$

for $i = 1$ to L do
  if $\mathsf{O}(\sigma(i)) == 1$ then
    $\mathbf{d}(\sigma(i)) = 1$

// normalize static score distribution vector

(4) $\mathbf{d} = \mathbf{d}/|\mathbf{d}|$

// compute TrustRank scores

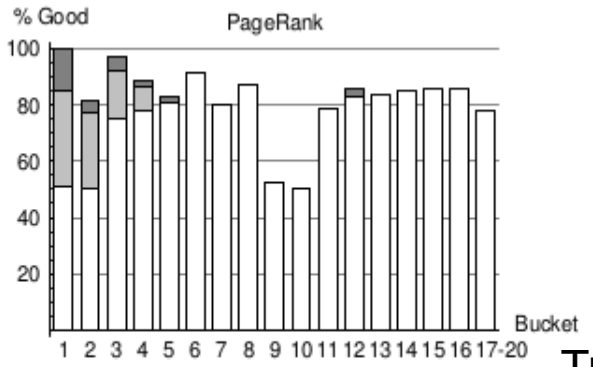
(5) $\mathbf{t}^* = \mathbf{d}$

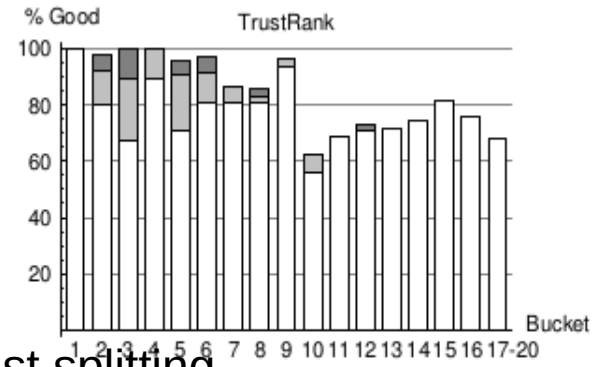
for $i = 1$ to M_B do
  $\mathbf{t}^* = \alpha_B \cdot \mathbf{T} \cdot \mathbf{t}^* + (1 - \alpha_B) \cdot \mathbf{d}$

return $\mathbf{t}^*$

end

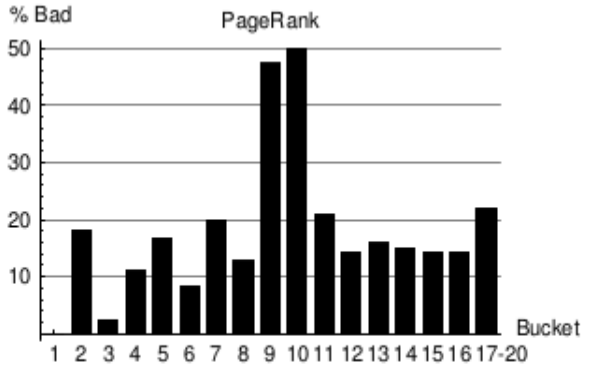
Risultati





Trust splitting

Figure 9: Good sites in PageRank and TrustRank buckets.



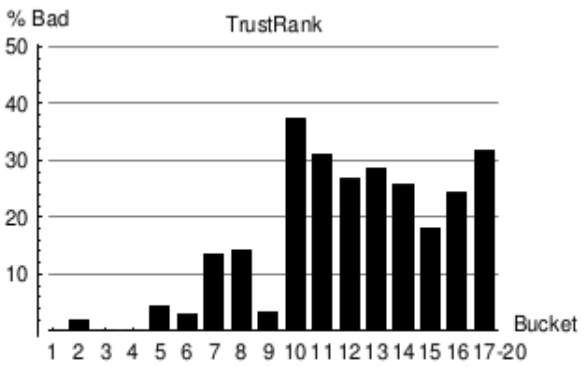


Figure 10: Bad sites in PageRank and TrustRank buckets.

Gyongy: Mass Estimation

- E’ introdotto il concetto di spam mass, una misura dell’impatto dello spam (da link) sul rank di una pagina
- Viene descritto un metodo per identificare spam farm
- Le pagine target di una farm spam sono quelle che ricevono un valore alto di pagerank e hanno una grande valore di spam mass mentre le pagine buone che hanno alto pagerank, derivato da altre pagine buone, hanno un basso spam mass
- Viene stimato lo spam mass di tutte le pagine calcolando e combinando due valori di pagerank:
  - il valore regolare di pagerank per ogni pagina
  - un valore personalizzato dove un grande gruppo di pagine reputaili (conosciute) ricevono un peso maggiore.
- La mass estimation può essere usata per identificare pagine che beneficiano di link di spam

Approccio Naive

- Il web può essere partizionato in nodi buoni V^+ e nodi spam V^- e la loro unione forma il grafo del web
- Trovare i nodi *x* che aumentano di molto il loro pagerank attraverso i nodi spam che puntano ad essi
- Per il momento consideriamo che i vicini che lo puntano sappiamo se sono ruputabili o spam
- Un modo semplice è quello che dato un nodo *x* andiamo a vedere i sui vicini che lo puntano e inferire se *x* è buono o spam

Approccio Naive

- In una prima approssimazione possiamo semplicemente guardare il numero di link in ingresso. Se la maggior parte dei inlink provengono da nodi spam, il nodo allora sarà etichettato come nodo target spam altrimenti buono.
- Questo approccio falisce quando ad esempio abbiamo un nodo con molti link in ingresso che provengono da nodi buoni rispetta a nodi spam ma il pagerank prodotto dipende molto di più dai nodi spam che da quelli buoni.
- Un'alternativa è non guardare solo gli inlink ma anche il pagerank che deriva dagli ognuno dei link. Il contribbuto di un link ammonta al cambiamento di pagarank che avviene se `e rimosso il link.
- La terza alternativa è etichettare tutti i contributi di pagerank dei nodi che sono direttamente e indirettamente conessi al nodo

Spam mass

- Per una data partizione { V^+, V^- } di *V* e per dei nodi *x* il pagerank di *x* è la somma dei contribbuti di nodi buoni e dei nodi spam.
- Teorema 1. Lo spam mass assoluto di *x*, denotata con M_x , è il pagerank che *x* riceve dai nodi spam che uguale a $M_x = q_x^{V^-}$
- Spam mass è la misura di quanto direttamente e indirettamente i nodi di spam vicini incrementano il pagerank di un nodo.
- Teorema 2. Lo spam mass relativo di *x*, denotara da m_x , è la frazione del pagerank di *x* dovuto al contribbuto dei nodi di spam cioè: $m_x = q_x^{V^-} / p_x$

Stimare spam mass

- Se non si conosce se i nodi seed sono spam oppure buoni bisogna fare una stima
- Assumimo che solo un sottoinsieme di nodi buoni è fornito ed è chiamato good core (V~)+
- Dato (V~)+ vengono calcolati due insiemi di valori di pagerank:
  - *p = PR(v)* il pagerank dei nodi basato su una distribuzione di salto uniforme
  - $p' = PR(v^{\tilde{V}^+})$ il pagerank basato sul good core con una distribuzione di salto $v^{\tilde{V}^+}$:

$$v_x^{\tilde{V}^+} = \begin{cases} 1/n & se \quad x \in \tilde{V}^+ \\ 0 & altrimenti \end{cases}$$

Stimare spam mass

- Teorema 3. Dati i valori di pagerank p_x e p'_x , la stima assoluta di spam mass di un nodo *x* è:

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

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

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

Algoritmo

input : good core $\tilde{\mathcal{V}}^+$, relative mass threshold τ,
PageRank threshold ρ
output: set of spam candidates $\mathcal{S}$

$\mathcal{S} \leftarrow \emptyset$
compute PageRank scores $\mathbf{p}$
construct $\mathbf{w}$ based on $\tilde{\mathcal{V}}^+$ and compute $\mathbf{p}'$
$\tilde{\mathbf{m}} \leftarrow (\mathbf{p} - \mathbf{p}')/\mathbf{p}$
for *each node* x *so that* $p_x \geq \rho$ **do**
  if $m_x \geq \tau$ **then**
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{x\}$
  end
end

Algorithm 2: Mass-based spam detection.

Bibliografia

- Combating Web Spam with TrustRank, Zoltan Gÿongyi, Hector Garcia-Molina, Jan Pedersen
- Link Spam Detection Based on Mass Estimation, Zoltan Gyongyi, Pavel Berkhin

Krishan: anti-trustrank

- Come per TrustRank gli autori propongono un algoritmo di selezione di un insieme seed di pagine che sono valutate da persone. Poi vengono usate la struttura dei link del web e le etichette inserite manualmente alle pagine dell'insieme seed per rilevare altre pagine
- L'algoritmo di trustranlk parte da un insieme di seed e poi utilizza pagerank personalizzato, usando un vettore che è un sottoinsieme di pagine dell'insieme seed, per manipolare pagerank in modo tale da visitare di più le pagine che sono buone
- Viene seguita la stessa intuizione di trustrank:le pagine spam non sono puntate da pagine buone.
- Perciò si parte con un insieme di pagine spam da un insieme seed e si propaga Anti Trust in direzione inversa con l'obbiettivo di rilevare le pagine spam le quali possono essere filtrate da un motore di ricerca.

Isolamento approssimato

- L’approccio su cui si basa anti trust rank è sull’isolamento approssimato cioè pagine buone molto raramente punteranno a pagine malevoli.
- Questo principio implica anche che le pagine che puntano a pagine spam sono molto probabilmente anche esse spam

Algoritmo

- L’algoritmo trustrank parte con insieme di pagine attendibili (seed) e si propaga la Trust lungo i link in uscita.
- In anti trust rank, l’anti trust viene propagata nella direzione inversa ai link in entrata partendo da un insieme di pagine dell’isieme seed che sono spam.
- Vengono classificate le pagine come spam se il valore di anti trust rank è più di una certa soglia.
- Un altro modo per classificare una pagina come spam è ritornare le *n* pagine che hanno valore di anti trust rank maggiore

Dettaglio

- Prelevare un insieme di pagine spam etichettate manualmente
- calcolare T per essere la matrice trasposta della matrice del grafo del web
- eseguire pagerank personalizzato sulla matrice T con seed set come insieme di teletrasporto
- calcolare il rank dellle pagine in ordine decrescente.

Dettaglio

- Visto che le pagine normali sono linkate da pagine nomrali se si calcola pagerank personalizzato con le pagine spam, sulla matrice trasposta, si ottiene pagerank al contrario.
- Perciò visto che il vettore di teletrasporto è costituito da pagine spam, e le pagine spam sono collegate da altre pagine spam, il rank al contrario ritorna tutte lo score delle pagine che puntano le pagine spam con più alto rank
- Ordinandole al contrario si hanno tutte le pagine che hanno un valore di spam più alto

Bibliografia

- Web Spam Detection with Anti-Trust Rank,Vijay Krishnan, Rashmi Raj

Badrank

- A differenza di pagerank, lalgoritmo di badrank non determina l’importanza delle pagine web ma misura le caratteristiche negative
- Badrank si basa sul principio del collegamento con vicini cattivi: se una pagina collega un’altra pagina con un alto Badrank, la prima pagina prende un alto badrank attraverso questo link
- Questo è molto simile a pagerank con la differenza che badrank non è basato sulla valutazione di link in entrata di una pagina web ma dai sui link in uscita

Badrank

- Badrank può essere formalizzato con la seguente formula:

$$BR(A) = E(A)(1 - d) + d(BR(T_1)/C(T_1) + ... + BR(T_n)/C(T_n))$$

- dove: BR(A) è il badrank di una pagina A, $BR(T_i)$ è il badrank delle pagine T_i le quali sono i link in uscita della pagina A_i , $C(t_i)$ è il numero di link in uscita della pagina T_i e *d* è il damping factor. Mentre E(A) rappresenta la valutazione di certe pagine web ovvero se sono spam o normali
- perciò prima di tutto le pagine devono essere valutate, un filtro assegna un valore E(A), il quale può essere basato sul grado di spam o il valore di pagerank

PR0

- Pagerank e badrank vengono combinati per penalizzare le pagine spam
- Un modo è calcolare il badrank prima è poi dividere il pagerank di ogni pagina con il propio badrank per ogni iterazione nel calcolo di pagerank.
- Questo ha dei vantaggi infatti una pagina con alto badrank passera un piccolo o nullo pagerank alle pagine a cui punta

Bibliografia

- http://pr.efactory.de/e-pr0.shtml

Davison: Topical trustrank

- Trustrank è un ottimo algoritmo per identificare lo spam, ma ha un problema cioè l'insieme seed usato potrebbe non essere sufficientemente rappresentativo per coprire bene tutti gli argomenti del web
- Un modo naturale di ottenere una grande copertura del web è usare informaini sugli argomenti
- Invece di usare un singolo trustrank score per un sito, gli autori propongono di calcolare il trustrank score per differenti argomenti

Davison: Topical trustrank

- Per fare questo bisogna partizionare l’insieme seed sulla base degli argomenti
- Usando ognuna di queste partizioni come seed set viene calcolato il trust score per ogni pagina

Combinazione

- Vengono mostrate due tecniche per combinare lo score di verità dei vari argomenti.
- Simple summation. In questa tecnica i vari punteggi sono sommati per generare il topical trust score.
- Quality bias. Viene introdotta una “quality” bias nella combinazione di score individuali locali di verità. Gli autori propongono di persare ogni valore di verità locale individuale con un fattore di influenza w_i per un argomento i

Bibliografia

- Topical TrustRank: Using Topicality to Combat Web Spam, Baoning Wu, Vinay Goel, Brian D. Davison

Caverlee:Credibility-Based Link Analysis

- Viene introdotto un concetto di credibilità di un link. Questa credibilità è definita tramite tr passaggi
- Prima vengono definite delle tecninche per assegnare semi automaticamente la credibilità ai link delle pagine web
- Secondo l'algoritmo di assegnazione della credibilità permette agli utenti di asserire la credibilità in maniera personalizzata
- Terzo algoritmo di ranking basato sulla credibilità (CredibleRank).
- Le prestazioni sono migliori di pagerank e trustrank nel cercare lo spam

Caverlee:Credibility-Based Link Analysis

- Viene fatta una separazione tra qualita della pagina e dei link
- La credibilità viene definita in termini di credibilità k-scope.
- Una funzione *C* di credibilità 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.

Credibilità naive

- In generale il web è troppo grande per essere etichetato tutto. Assumiamo che l'insieme *P* di tutte le pagine può essere diviso in tre insiemi: pagine conosciute essere buone P_w , pagine conosciute spam P_b , e pagine per le quali non si sa nulla P_u tale che P = P_w ∪ P_b ∪ P_u.

$$C_{naive}(p,t) = \begin{cases} 0 & if \quad p \in P_b \\ \theta & if \quad p \in P_u \\ 1 & if \quad p \in P_w \end{cases}$$

Credibilità k-scope

- La credibilità k-scope valuta la credibilità di un pagina in termini della qualità della camminata random originata dalla pagina fino a k step.
- Diciamo che un percorso dalla pagina *p* alla pagina *q* è un “bad path” se la pagina di destinazione è una pagina spam e nessuna altra pagina nel percorso è una pagina spam
- Indichiamo con $Path_k(p)$ l'insieme di tutti i percorsi dalla pagina *p* di lunghezza *k*
- Indichiamo con $BPath_k(p)$ l'insieme di tutti i percorsi “bad path” dalla pagina *p* di lunghezza *k*
- $PR(path_k(p))$ la probabilità che una camminata casuale passi lungo un percorso k-lenght da una pagina p

Credibilità k-scope

- Formalmente definiamo la credibilità k-scope di una pagina in termini di probabilità che una camminata casuale eviti le pagine spam dopo aver superato *k* hop dalla pagina di origine.

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

- 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 essa è un pagina spam o nel caso in cui tutti i percorsi originati da *p* colpiscono una pagina spam all'intenro di k hop, allora *p* non è credibile C_k(p, t)=0

Credibilità k-scope

- Visto che non che non è possibile avere tutto il grafo e non c’`e 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

Credibilità k-scope

- Visto che non che non è possibile avere tutto il grafo e non c'`e 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
- Definiamo la credibilità tunable k-scope di una pagina *p,* denotata con $C_k(p)$, in due fasi, quando p non appartiene a P_b:

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

- e quando *p* appartiene P_b allora: $C_k(p) = 0$

Algoritmo

- Il calcolo della credibilità di una pagina è un processo locale che richiede solo la pagina corrente e il crawiling di tutte le pagine all’interno di k-hop dalla pagina di origine.
- Il costo principale di calcolare la credibiltà tunable k-scope è il costo di identificare l’insieme dei percorisi cattivi per ogni pagina e il costo del calcolo delle probabilità dei percorsi.

CredibleRank

- CredibleRank definisce che la qualit`a di una pagina `e determinata da due criteri: la qualità delle pagine che puntano ad essa e la credibilità di ogni pagina puntata
- Definendo con *In(p)* l'insieme di pagine che puntano a *p*. Calcoliamo CredibleRank $r_c(p)$ per una pagina *p*

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

- 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)*

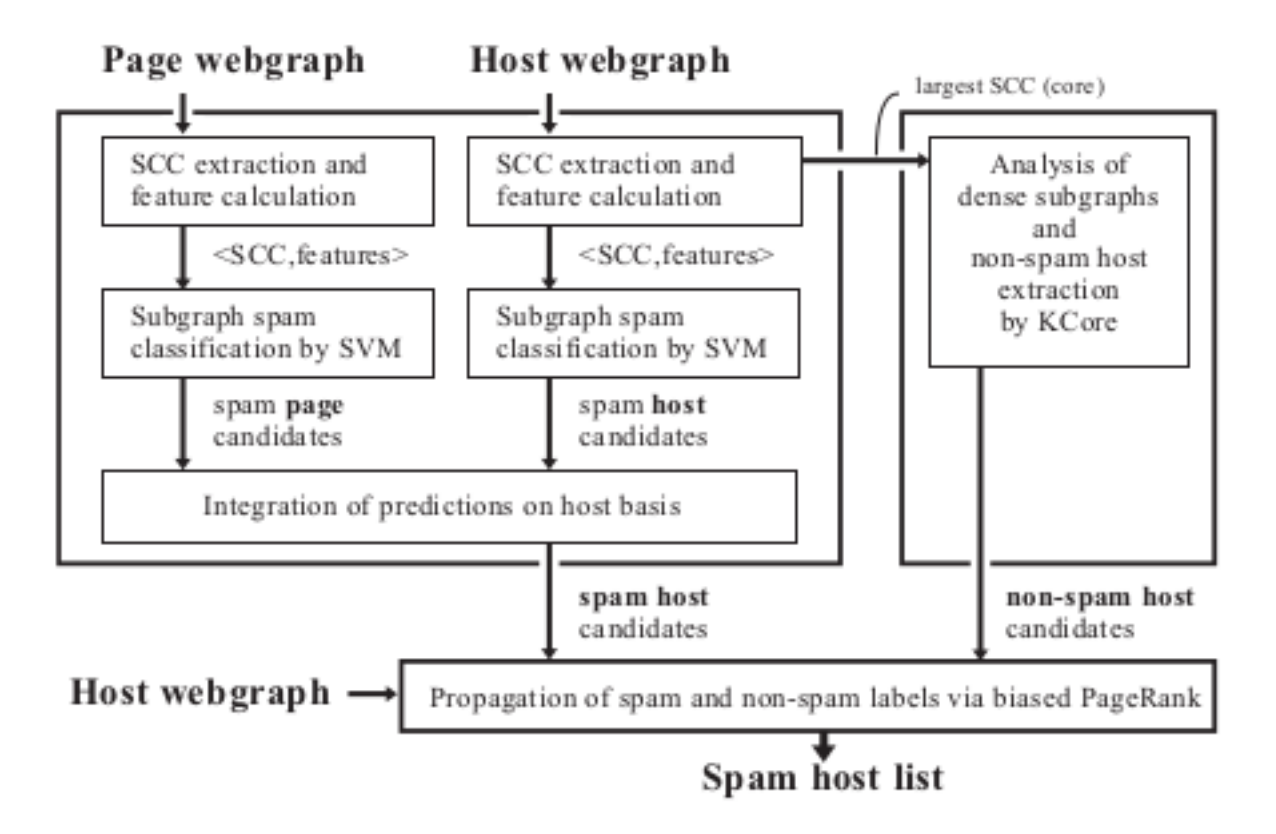
Bibliografia

- Countering Web Spam with, Credibility-Based Link Analysis, James Caverlee, Ling Liu

Exploring Densely Connected Subgraphs

- Metodo per rilevare spam basato sull’analisi dei link.
- Costituito da tre fasi:
  - (1) decomposizione del grafo in sottografi connessi densi e calcolo delle feature per ogni sottografo
  - (2) uso del classificatore SVM per identificare i sottografi composti da spam
  - (3) propagazione dellapredizione su grafo del web attraverso pagerank personalizzato

Framework


Page webgraph
Host webgraph
largest SCC (core)
SCC extraction and feature calculation
SCC extraction and feature calculation
Analysis of dense subgraphs and non-spam host extraction by KCore
<SCC,features>
<SCC,features>
Subgraph spam classification by SVM
Subgraph spam classification by SVM
spam page candidates
spam host candidates
Integration of predictions on host basis
spam host candidates
non-spam host candidates
Host webgraph
Propagation of spam and non-spam labels via biased PageRank
Spam host list

Sottografi

- Gli autori si focalizzano sui sottografi connessi densi per identificare sottografi spam e non spam usando due approcci
- Analisi della struttura bow-tie del grafo del web. Alcuni studi riportano che grandi SCC vicino al core nel grafo del web degli host, sono potenziali indicatori di spam. Questo motiva l'intuizione degli autori di sfruttare la struttura boe-tie e analizzare gli SCC lungo "In","Out","tendrils" e "tube"
- Identificazione dei sottografi connessi densi all'interno del core del grafo del web. Dal momento che K-core sono sottografi chiusi, possiamo inferire che host importanti, ovvero quelli che hanno molti inlink, sono connessi a altri host altamente connessi, formando sottografi robusti con alta coreness. Dall'altra parte per host che sono spam c'`e una limitazione della coreness

Calcolo

- Per ogni sottografo vengono calcolate le seguenti feature:
- URL feature, ovvero statistiche base (conteggio, somma, media) della lunghezza degli URL, lunghezza del percorso, lunghezza del hostname, e distanza hostname.
- Feature basate sulla struttura dei link: il tipo di sottografo nella struttura bow-tie, le statistiche di pagerank di nodi composti, il rapporto della media aritmetica e il massimo pagerank degli inlink di un nodo, il numero di outlink, inlink, e il loro rapporto e altre caratteristiche

Pagerank

- Per aumentare il numero di spam e non spam host rilevati viene utilizzato pagerank personalizzato come trustrank.
- Ma a differenza di trustrank che propaga la trust, vengono propagati simultaneamente trust e antitrust per valorizzare gli host non spam e penalizzare quelli di spam

Bibliografia

- Web Spam Detection by Exploring Densely Connected Subgraphs Yutaka I. Leon-Suematsu, Kentaro Inui, Sadao Kurohashi and Yutaka Kidawara

Biased Random Walks From Spam Seed Sets

- Il metodo parte da un piccolo seed set fornito dagli utenti e simultani random walk.
- La randon walk è basata sull’esplorazione dei vicini locali di un seed set.
- Il troncamento è utilizzato per tenere solo i nodi piùfrequentemente visitati. Alla finie del processo i nodi sono ordinati in ordine decrescente per la loro probabilità finale.

Bibliografia

- Extracting Link Spam using Biased Random Walks From Spam Seed Sets, Baoning Wu, Kumar Chellapilla

Identifying Link Farm Spam Pages

- In questo articolo viene presentato un algoritmo per il rilevamento automatico delle link farm.
- Per fare questo prima viene generato un seed set basato su un insieme di link comuni tra incoming link e outgoing link di pagine web
- Poi questo insieme viene allargato. I link tra pagine identificate sono ripesati cosi da modificare il grafo del web per usarlo nel ranking delle pagine

Identifying Link Farm Spam Pages

- Gli autori hanno osservato che le pagine all’interno delle link farm sono densamente connesse tra di loro
- Se possiamo le pagine all’interno di una link farm le impostiamo come seed set, allora per ogni nuova pagina, è possibile che essa faccia parte della stessa link farm se questa ha molti link in entrata e in uscita, da e per, il seed set

Identifying Link Farm Spam Pages

- Gli autori hanno osservato che le pagine all'interno delle link farm sono densamente connesse tra di loro
- Se possiamo le pagine all'interno di una link farm le impostiamo come seed set, allora per ogni nuova pagina, è possibile che essa faccia parte della stessa link farm se questa ha molti link in entrata e in uscita, da e per, il seed set
- Allora si può allargare il seed set aggiungendo la nuova pagina
- Questo processo può essere iterato e terminerà quando nessuna altra pagina potrà essere aggiunta

Algoritmo

- L’algoritmo per fare è composto dai seguenti passaggi:
  - generare un seed set da un insieme di dati
  - espandere il seed set
  - eseguire il ranking delle pagine attraverso la combinazione dei valori di badness con algoritmi di ranking base

Prima fase

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.

Prima fase

- 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 molti nodi in comune è probabile che queste pagine facciano parte di una link farm. 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.
- Il metodo non calcola che una pagina punti o è puntata da un'altra pagina ma fa riferimento al sito o dominio

---

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.

Seconda fase

- Per espandere il seed set viene usato la ParentyPenality l'intuizione è che se una pagina punta a un insieme di pagine cattive è probabile che anche essa sia cattiva.
- Viene usata un soglia T_{PP} per giudicare una pagina: se il numero di outgoing link a pagine cattive è uguale o supera la soglia, la pagina sarà giudicata cattiva.

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.

Terza fase

- Una volta rovate le pagine cattive bisogna utilizzare queste informazioni per il ranking.
- Un modo è cambiare la matrice di adiacenza del grafo del web con il data set ed eliminare le pagine all’interno della link farm

Bibliografia

- Identifying Link Farm Spam Pages, Baoning Wu and Brian D. Davison

Castillo:Web Spam Detection using the Web Topology

- Viene presentato un metodo per rilevare lo spam che combina feature basate sui link e contenuto e usa la struttura del grafo per vedere le dipendenze tra le pagine.
- Gli autori hanno trovato che gli host collegati appartengono alla stessa classe spam o no.

Link feature

- Degree-related measures.
- Pagerank.
- Trustrank.
- Truncated pagerank.
- Estimation of supporters.

Content feature

- Numero di parole nella pagina, titolo e media delle lunghezze delle parole.
- Frazione del testo delle ancore.
- Frazione di testo visibile.
- Frequenza di compressione.
- Corpus precisio e corpus recall.
- Query precision e query recall.
- Indipendent trigram likelihood.
- Entropia dei trigrammi.

Classificatore

- E’ stato usato uno dei classificatori di weka.
- Usando entrambi i tipi di feature descritti in precedenza viene costruito un albero con 45 feature.
- Durante la classificazione si tiene conto che l’errore di classificazione di pagine spam con normali non è uguale del contrario

Clustering

- Viene fatto un clusterign sul grafo ottenuto.
- Sia il clastering di *G* composto da *m* cluster *C 1 , C 2 , ..., C m* .
- Sia *p(h) ∈ [0...1]* la predizione di un algoritmo di classificazione C cosi che per ogni host *h* un valore di *p(h)* uguale a 0 indica non spam mentre un valore 1 indica spam.
- Per ogni cluster Cj , j = 1, ..., m calcoliamo la media dello spam.
- Vengono usate due soglie una bassa t_l e una alta t_u .
- Per ogni cluster Cj se $p(Cj) \leq t_l$ allora tutti gli host di Cj sono marcati come non spam e *p(h)* è impostata a 0 per tutti gli host appartenenti al cluster.
- Altrimenti il cluster viene confrontato con la sogli alta e se la supera o è uguale viene considerato spam

Bibliografia

- Know your Neighbors: Web Spam Detection using the Web Topology, Carlos Castillo, Debora Donato, Aristides Gionis, Vanessa Murdock, Fabrizio Silvestri

Improving Web Spam Classifiers Using Link Structure

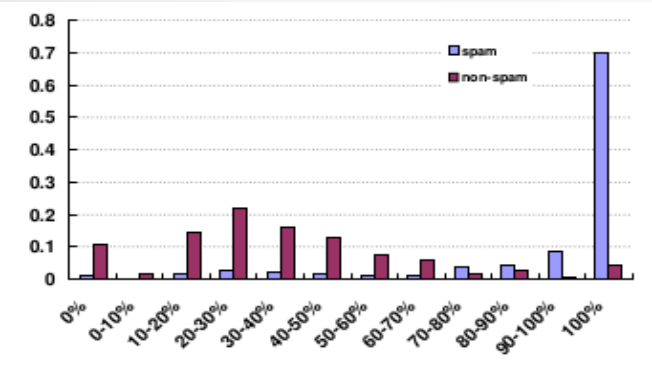
- Gli autori desrivono due approcci per migliorare la classificazione.
- Prima viene impementato un classificatore per catturare una grande porzione di spam dei dati.
- Dopo vengono descritte delle euristiche per decidere se un nodo dovrebbe esser rietichettato basandosi sul classificatore precendente e sulla conoscenza dei vicini

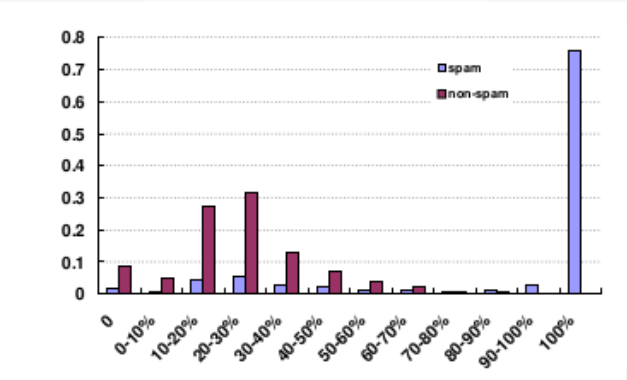
Improving Web Spam Classifiers Using Link Structure

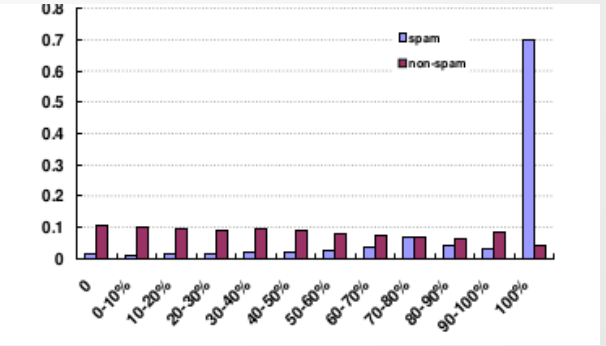
- Il primo metodo è chiamato relabeling. Questo metodo protrebbe cambiare l’etichetta assegnata ad un sito dal classificatore base utilizzando molte feature basate sui nodi vicini.
- Il secondo metodo è chiamato secondary classifier prende entrambi i risultati (classificatore base e classificatore basato sui vicini) come attribbuti di input.

Nodi vicini

- Gli autori ipotizzano che la struttura dei vicini è un buon indicatore per un sito per essere indicato spam o normale. In particolare sono interessati alle distribuzioni di particolari proprietà dei vicini







Miglioramento del classificatore

- Approccio relabeling: gli autori intendono il processo di rietichettatura di un sito da spam a non spam o viceversa attraverso determinate regole. In particolare prima viene definita un etichetta per i nodi vicini di un sito sulla base delle euristiche definite prima. Questa etichetta è assegnata con un valore di confidenza. Viene confrontata questa etichetta con quella del nodo assegnata dal classificatore. Se le due etichette sono molto diverse con ogni altro vicino, in termini di confidenza, allora l'etichetta del sito viene cambiata.
- Approccio classifier. Un semplice metodo è quello di adottare un altro classificatore.

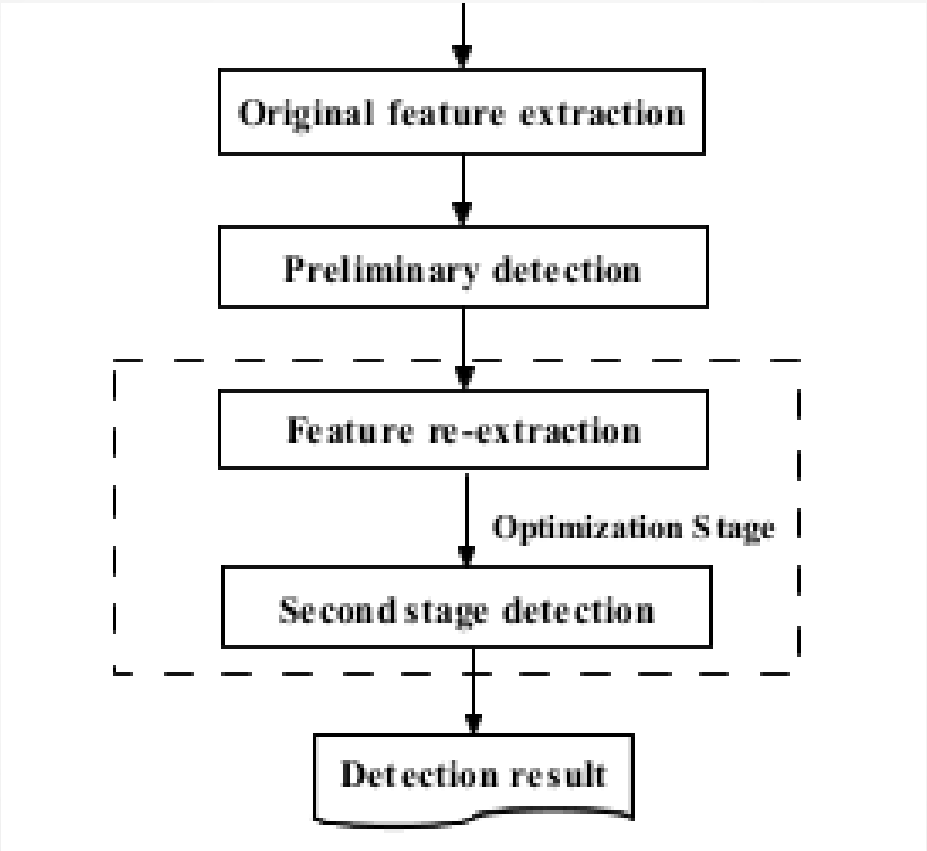
Bibliografia

- Improving Web Spam Classifiers Using Link Structure, Qingqing Gan and Torsten Suel

Re-Extracted Features

- Invece di utilizzare delle euristiche per rilevare lo spam, viene proposta una strategia di estrazione delle feature per ottimizzare la rilevazione.
- La riestrazione delle feature è basata sulla topologia del web e su risultati di rilevamento ottenuti in precedenza.

Strategia


Original feature extraction
Preliminary detection
Feature re-extraction
Optimization Stage
Second stage detection
Detection result

Riestrazione delle feature

- Clustering feature
- Feature di propagazione
- Feautre basate sui vicini

Risultati

Table 1. Web Spam Detection Performace with Different Strategies

Features	TP	FP	Precision	Recall	F-measure	AUC
of	0.832	0.0626	0.845	0.832	0.838	0.958
Sgl	**0.885 (6.41%)**	0.0645 (-2.98%)	**0.848 (0.47%)**	**0.885 (6.41%)**	**0.867 (3.38%)**	**0.967 (0.088%)**
of + cf + pf + nf	**0.900 (8.12%)**	**0.0606 (3.08%)**	**0.859 (1.63%)**	**0.900 (8.12%)**	**0.879 (4.80%)**	**0.971 (1.26%)**

Bibliografia

- Improving Web Spam Detection with Re-Extracted Features, Guang-Gang Geng, Chun-Heng Wang, Qiu-Dan Li

Link Based Small Sample Learning

- Per risolvere il problema di riperimento dei dati etichettati e di calcolo viene proposto questo metodo di classificazione.
- Il self-training è una tecnica comune di learnig semi supervisionato.
- Basandosi sul fatto che i nodi vicini hanno a volte le stesse proprietà, gli autori propongono che i classificatori che utilizzano i link, devono essere istruiti con dei dati etichettati per prevedere la spammacita PS.
- Durante la fase di learnig la spammacità dei link LS è calcolata in funzione dei vicini

Link Based Small Sample Learning

- Gli autori calcolano la spammacità dei link in questo modo

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

- dove *v, h* sono gli host, *weight(h, v)* è il peso dell'host *h, v, weight(h, v∈ 1,log(n))*, dove *n* è il numero di link tra i due nodi

Algoritmo

Algorithm 1 Link-Training Algorithm

Input :

L:	Labeled training set
U:	Unlabeled examples set
T:	Test set
C:	Classifier
K:	Iterations
G:	Host level hyperlink graph
n, p:	The number of selected non-spam and spam samples in each iteration

1: $i = 0$
2: **while** $i < K$ **do**
3: Train classifier C with L
4: Detect the examples in U with C, compute their PS values with Formula 1
5: Annotate G with the PS values of samples in U and L (the PS value of spam and non-spam samples in L are 1 and 0 respectively)
6: Perform link learning on annotated G; Compute the LS values with Formula 2
7: According to the values of LS, select p largest and n smallest examples as spam and normal respectively, put the $n + p$ samples to L, and delete them from U (The principle of choosing p and n is to keep the ratio of non-spam and spam unchanged in training set.)
8: $i = i + 1$
9: **end while**
10: Train classifier C on the train set L
11: Test the samples in T with the trained classifier C

Output : Web spam detection result on test set T

Bilbiografia

- Link Based Small Sample Learning for Web Spam Detection, Guang-Gang Geng, Qiu-Dan Li, Xin-Chang Zhang

Let Web Spammers Expose Themselves

- Gli autori propongono un approccio differente il quale rileva lo spam dei link andando a guardare come i link vengono creati
- Normalmente gli spammer si alleano tra di loro
- I forum SEO sono uno degli strumenti principali tramite i quali vengono fatte le alleanze
- Gli autori propongono perciò di andare a prendere i link sospetti all'interno di questi forum

Let Web Spammers Expose Themselves

- L'obbiettivo non è facile perché nei post ci sono altre informaizioni che producono un effetto negatico nella rilevazione
- Per rendere efficiente il metodo:
  - Vengono estratti tutti i link conentuti nei post
  - Vengono estratte le feature per i link dalle loro relazioni con gli utenti del forum e dalla struttura di essi nel grafo
  - Viene costrutito un framework semi-supervisionato per calcolare lo score dello spam dei siti

Feature

- Feature dal forum SEO:(1) numero di URL di un sito, (2) frequenza di un URL nel forum, (3) numero di thread che il proprietario dell'URL ha discusso (4) numero di post autorizzati dal proprietario dell'URL (5) numero di thread creati dal proprietario dell'URL, (6) numero di URL postati dal proprietario dell'URL, (7) media degli URL per post del proprietari dell'URL and (8) numero di post che contengono l'URL del proprietario dell'URL
- Feature del grafo: numero di inlink, numero di outlink, numero di mutal link, media degli outlink dei vincini in entrata, media degli inlink dei vicini in uscita
- Feature del sito: URL la sua lunghezza

Bibliografia

- Let Web Spammers Expose Themselves, Zhicong Cheng Bin Gao, Congkai Sun,Yanbing Jiang, Tie-Yan Liu

Detecting Cloaking Web Spam Using Hash Function

- Questo metodo cerca di rilevare le pagine cloaking basandosi sulla rilevazione delle differenze dei termini tra le copie delle pagine del crawler e quelle delle browser

Lavori precendenti

- Ci sono pochi articoli che descrivono come rilevare il cloaking.
- Gli spammer possono rilevare un crawler dal suo indirizzo IP o dal user agent.
- Najork aveva proposto un metodo per il rilevamento del cloaking confrontando la copia del crawler con quella del brawser e se le due pagine erano le stesse, allora la pagina non era un cloaking. Questo metodo non è buono per il fatto che oggi le pagine vegono generate dinamicamente e possono variare in contenuti.
- Wu usa un metodo che confronta la differenza tra termini e link. Se il numero dei temini diversi o dei link supera una certa soglia la pagina viene segnalata

Metodi

- Questo metodo si basa sulle differenze de termini tra le due copie di un crawler e un browser.
- Per confrontare le due copie vengono usate delle funzioni hash per aumentare la velocità di confronto.
- Se il valore hash delle due funzioni è lo stesso le pagine sono identiche

Tipi di cloacking

- Syntactic cloaking, contenuti diversi per crawler e utente
- Semantic cloaking, è un sotto iniseme del syntatic cloacking ed èun tentativo di fornire contenuti diversi al web crawler e al browser web, in modo che differenze di significato tra questi due copie di un URL possono ingannare gli algoritmi di ranking dei motori di ricerca

Rilevamento cloacking statico

- Il cloaking statico è una situazione nella quale le copie sono differenti
- Vengono definiti 5 step.
- $f(C_1) = f(B_1)$
- $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) = f(C_2)$, $f(B_1) = f(B_2)$
- $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) = f(C_2)$, $f(B_1) \neq f(B_2)$
- $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) \neq f(C_2)$, $f(B_1) \neq f(B_2)$
- $f(C_1) \neq f(B_1)$, $f(C_2) \neq f(B_2)$, $f(C_1) \neq f(C_2)$, $f(B_1) = f(B_2)$

Rilevamento cloacking dinamico

- Il cloaking dinamico è un nuovo tipo di cloacking, si comporta come una pagina normale ed è difficile da identificare
- La sistuazione può essere descritta come $B_1 = B_2 = C_2 \neq C_1$
- Vengono definiti 2 step.
- $f(C_1) \neq f(B_1)$, $f(C_2) = f(B_2)$, $f(B_1) \neq f(B_2)$
- $f(C_1) \neq f(B_1)$, $f(C_2) = f(B_2)$, $f(B_1) = f(B_2)$

Bibliografia

- Detecting Cloaking Web Spam Using Hash Function, Shekoofeh Ghiam, Alireza Nemaney Pour

Predicting Web Spam with HTTP Session Information

- In questo lavoro, viene presentato un approccio predittivo leggero per la classificazione web spam che si basa esclusivamente su informazioni delle sessioni HTTP come gli indirizzi IP e sessione headers HTTP

Innovazioni

- Nuovo approccio per identificare il web spam usando HTTP
- In particolare viene inserito un classificatore delle sessioni HTTP all'interno del processo di retrieval il quale predice se una pagian web è spam sulla base delle informazioni delle sessioni HTTP
- Questo approccio difende dai malware e salva la banda e spazio in memoria
- In questo modo l'indicizzazione del motore di ricerca è fatta solo su contentuti buoni
- Perciò può essere usato online

Innovazioni

- Nuovo approccio per identificare il web spam usando HTTP
- In particolare viene inserito un classificatore delle sessioni HTTP all'interno del processo di retrieval il quale predice se una pagian web è spam sulla base delle informazioni delle sessioni HTTP
- Questo approccio difende dai malware e salva la banda e spazio in memoria
- In questo modo l'indicizzazione del motore di ricerca è fatta solo su contentuti buoni
- Perciò può essere usato online

Innovazioni

- Gli autori indicano che quest approccio è complementare ad altri approcci basati su contentuto e link

Approccio

- Per migliorare l'esperienza browser (se usato lato client) e migliorare la qualità dei contentuti (se usato dal crawler) è stato proposto un nuvo approccio per ottenere le pagine web
- Il riperimento delle pagine seguen il normale flusso delle richieste HTTP
- Ma quando viene letta una pagina lo user agent legge solo l'header della pagina
- Successivamente lo user agent aziona un classificatore per valutare l'header e classificarlo come spam o normale
- Quando un header viene classificato come normale la pagina viene letta interamente

Approccio

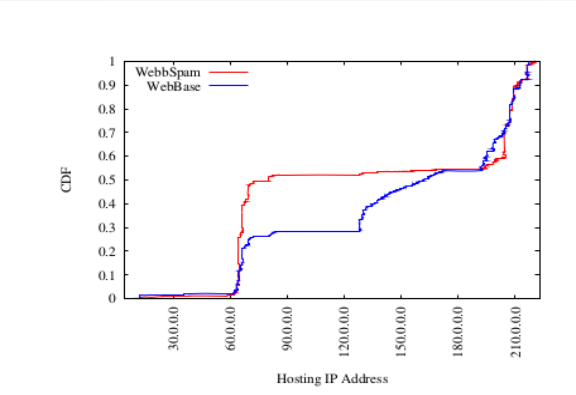
- Per migliorare l'esperienza browser (se usato lato client) e migliorare la qualità dei contentuti (se usato dal crawler) è stato proposto un nuvo approccio per ottenere le pagine web
- Il riperimento delle pagine seguen il normale flusso delle richieste HTTP
- Ma quando viene letta una pagina lo user agent legge solo l'header della pagina
- Successivamente lo user agent aziona un classificatore per valutare l'header e classificarlo come spam o normale
- Quando un header viene classificato come normale la pagina viene letta interamente

Benefici

- Scarica le pagine che sono legittime
- Metodo complementare ad altri metodi basati su link e contenuto

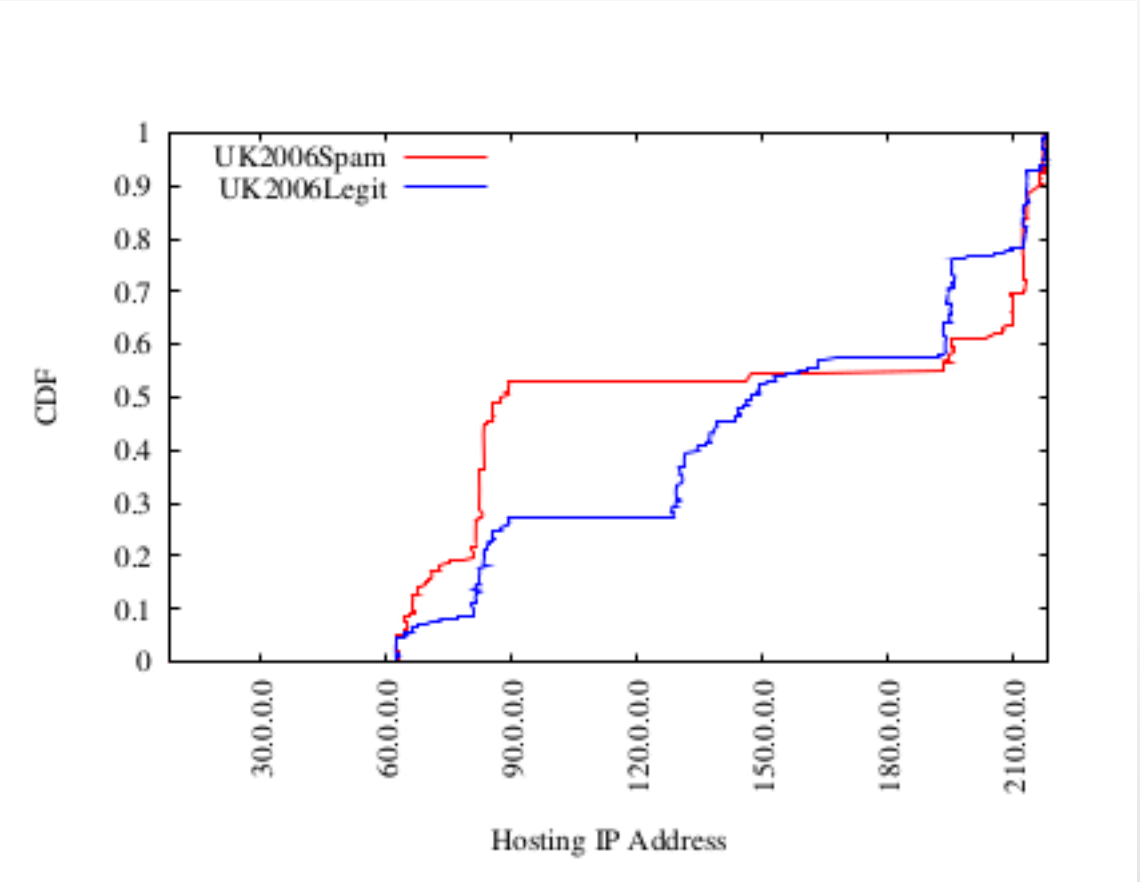
Feature

- Tutte le feature sono booleane se $f_i=1$ la feature è presente altrimenti la feature è assente
- Una delle feature importanti è l'IP
- Dal grafico si nota che gli IP degli host spam sono ragruppati in un piccolo range tra 63.* – 69.* e 204.* – 216.
- Ma il 50.6% degli host normali sono in questo range



Feature

- La distribuzione degli IP nel data set WEBSPAM-UK2006 e simile a quella descritta prima



Header

Table 2: Most popular HTTP session headers.

Header	WebbSpam Total Count (Unique Count)	WebBase Total Count (Unique Count)	UK2006Spam Total Count (Unique Count)	UK2006Legit Total Count (Unique Count)	Sample Value
Content-Type	348,878 (688)	392,600 (122)	1,338 (41)	3,542 (123)	text/html
Server	343,168 (6,513)	387,301 (2,505)	1,336 (196)	3,490 (704)	apache/2.0.52 (fedora)
Connection	327,478 (6)	354,837 (8)	1,322 (2)	3,230 (4)	close
X-Powered-By	209,215 (261)	114,181 (111)	862 (52)	1,753 (90)	asp.net
Content-Length	162,532 (31,232)	214,100 (58,417)	498 (447)	2,203 (1,987)	1470
Cache-Control	148,715 (548)	103,916 (1,461)	283 (27)	1,377 (117)	private
Set-Cookie	145,315 (140,431)	110,519 (105,636)	287 (287)	1,392 (1,376)	gx_jst=9fa7274e662d6164; path=/apps/system
Link	142,785 (15,573)	79 (64)	797 (421)	2,690 (2,215)	<style.css>; rel="stylesheet"; type="text/css"
Expires	93,477 (25,056)	55,991 (31,197)	183 (44)	745 (353)	mon, 26 jul 1997 05:00:00 gmt
Pragma	75,435 (32)	29,464 (9)	167 (5)	541 (10)	no-cache
Last-Modified	73,071 (50,206)	149,191 (125,573)	418 (380)	1,348 (1,245)	wed, 21 dec 2005 00:21:06 gmt
P3P	59,023 (816)	21,970 (205)	27 (14)	84 (54)	cp="noi cura adma our nor bus phy onl uni pur com nav sta"
Accept-Ranges	47,821 (4)	157,384 (4)	442 (1)	1,245 (2)	bytes
X-Meta-Robots	45,900 (241)	N/A	327 (25)	662 (64)	index, follow
Refresh	45,075 (8,556)	136 (6)	9 (8)	150 (137)	0; url=/index.asp
Etag	42,546 (24,783)	118,804 (115,122)	347 (317)	1,125 (1,039)	"110d66-112-24dfc0"
Vary	25,721 (45)	15,939 (27)	43 (5)	120 (12)	accept-encoding, user-agent
Content-Language	20,397 (163)	9,661 (49)	138 (7)	401 (18)	en-us

Osservazioni

- La popolarità relativa delle varie intestazioni di sessione HTTP varia notevolmente tra spam e pagine Web legittime
- Per esempio, il 60% delle istanze di spam nel Corpus Webb Spam possiedono una attributo di header "X-Powered-By " mentre solo il 29,1% delle legittime istanze nei dati WebBase contengono tale attributo.
- Questa disparità è ancora più grande per l'intestazione "Link", che si trova nel 40,9% dei casi di spam e solo il 0,02% dei legittimi
- La tabella mostra anche un fenomeno simile per lo spam e le istanze legittime in webspam-UK2006 corpus

Osservazioni

- Le distribuzioni dei valori degli header HTTP sono distinte tra pagine spam e normali
- Per esempio, il 59.023 delle istanze di spam nel Corpus Webb Spam Corpus possiedono un vaolore "P3P" Valore di header e solo il 34,5% di loro ha un valore è anche posseduto da almeno una istanza buona nel corpusi WebBase

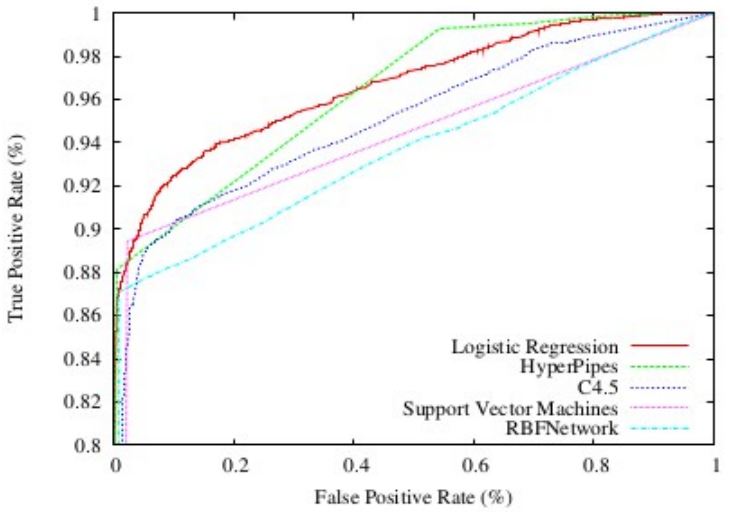
Osservazioni

- Per derivare le feature di classificazione dall'header HTTP vengono create tre rappresentazioni di un unico valore di header.
- Prima vien memorizzato il valore di header come una frase
- Dopo la frase viene tokenizzata sul carattere bianco e sulla base della punteggiatura
- Viene memorizzato il risultato con gli unici 2-ngram e 3-ngram
- Infine viene aggiunto all'inizio il nome dell'header

Risultati

Table 4: Top 10 features for Webb Spam and Web-Base.

Rank	Feature
1	accept-ranges_bytes
2	x-powered-by_php/4 3
3	x-powered-by_php/4
4	content-type_text/html; charset=utf-8
5	content-type_text/html; charset=iso-8859-1
6	expires_00 00 gmt
7	64.225.154.135
8	server_fedora
9	pragma_no-cache
10	p3p_cp=



Bibliografia

- Predicting Web Spam with HTTP Session Information, Steve Webb, James Caverlee, Calton Pu

Metodi basati sui clik o valutazioni

- Fighting against Web Spam: A Novel Propagation Method based on Click-through Data, Chao Wei, Yiqun Liu, Min Zhang, Shaoping Ma, Liyun Ru, Kuo Zhang
- BrowseRank: Letting Web Users Vote for Page Importance, Yuting Liu, Zhiming Ma, Tie-Yan Liu, Ying Zhang, Shuyuan He, Hang Li
- User Behavior Oriented Web Spam Detection, Yiqun Liu, Min Zhang, Shaoping Ma, Liyun Ru