

Pengembangan Aplikasi Menghitung *Spam* Berbasis *Tag* pada Situs *Social Bookmarking* Menggunakan Metode *Spam Factor*: Studi Kasus *del.icio.us*

Application Development for Calculate Spam Based on Tag in Social Bookmarking site Using Spam Factor Method: Case Study del.icio.us

Arie Kurniawan, Dana Sulisty Kusumo, S.T., M.T., Ph.D., Indra Lukmana S., S.T., M.T.

¹²³ Prodi S1 Teknik Informatika, Fakultas Informatika, Universitas Telkom.

¹akarie@students.telkomuniversity.ac.id, ²danakusumo@telkomuniversity.ac.id,

³indraluk@telkomuniversity.ac.id

Abstrak

Social Bookmarking merupakan salah satu jenis *social tagging* yang digunakan untuk mengkategorisasikan sebuah tautan web. Dalam situs web *social bookmarking* seperti *del.icio.us* terdapat banyak *tag* yang digunakan oleh *user* untuk mengkategorisasikan atau merepresentasikan sebuah situs blog, *URL* dan tautan. Pada penelitian ini, sebuah *tag* mengandung *spam* apabila *tag* tersebut digunakan pada *bookmark* tetapi tidak mendeskripsikan konten / situs web *bookmark* tersebut. Ketika terjadi *spam* maka dapat terjadi ambiguitas karena penggunaan *tag* dalam sebuah *bookmark* yang tidak merepresentasikan *bookmark* tersebut. Dalam penelitian ini diimplementasikan metode *spam factor* untuk menghitung *spam* dari sebuah *tag* yang direpresentasikan dalam sebuah nilai. Untuk implementasi metode *spam factor*, maka dibentuk daftar *tag* yang benar untuk setiap *bookmark*, membentuk *posting random good user* dan *posting random bad user*, mengimplementasikan *trusted moderator* untuk mendeteksi dan menghilangkan *posting* yang mengandung *spam*, dan mengurutkan pemakaian *tag* dan *bookmark* dengan *occurrence-based search*. Metode *spam factor* menghitung nilai *spam* sebuah *tag* dari jumlah dan kesesuaian penggunaan *tag* tersebut dengan *bookmark* pada setiap *posting*. Nilai yang dihasilkan dari *spam factor* berkisar 0-1 dimana semakin besar nilai yang didapatkan, maka semakin tinggi *tag* tersebut terindikasi *spam*. Dari penelitian ini dapat disimpulkan bahwa nilai *spam factor* yang dihasilkan lebih baik ketika menggunakan *trusted moderator* karena menghilangkan *spam* dari sistem.

Kata kunci : tag, spam, social bookmarking, spam factor.

Abstract

Social bookmarking is a kind of *social tagging* that used for categorize a website. *Social Bookmarking* website such as *del.icio.us*, there's a lot of *tag* that user used to categorize or represent a blog, *URL* and link. In this research, a *tag* indicated *spam* if the *tag* used in a *bookmark* that not described *bookmarks* content or websites. If *tag spam* happened then ambiguities could happen too because the *tag* used but not represent the correct *bookmarks*. This research implement *spam factor* method that could count *spam* in a *tag* which is represented in score. Before implement *spam factor* method, first generate list of correct *tags* in each *bookmark*, then generate random good user *posting* and random bad user *posting*, implement *trusted moderator* which detected and removed *posting* that infected a *spam*, then rank each *tag* and *bookmark* with *occurrence-based search* method. This research use *spam factor* method for measure *spam* score on *tag*, score based on quantity and compatibility of using a *tag* on a *bookmark* for every *posting*. Score for *spam factor* is between 0-1, the bigger score then the more *tag* indicated as *spam*. From this research we can conclude *trusted moderator* makes *spam factor* scores better because eliminating *spam* from system.

Keywords: tag, spam, social bookmarking, spam factor

1. Pendahuluan

1.1 Latar Belakang

Tag merupakan sebuah kata kunci, nama kategori ataupun *metadata* [1]. *Social Tagging* merupakan kegiatan pelabelan (*tagging*) bersama secara publik dan mengkategorisasikan sumber daya yang dapat dibagi (*resources sharing*) pada ruang lingkup daring [2]. Manfaat dari *tagging* juga sebagai alat komunikasi tentang informasi kontekstual dalam sebuah objek. Situs web *social bookmarking* merupakan salah satu contoh dalam pemanfaatan teknologi *social tagging* [3].

Dalam sebuah situs web *social bookmarking* seperti *del.icio.us* terdapat banyak *tag* yang digunakan oleh *user* untuk mengkategorisasikan atau merepresentasikan sebuah situs blog, *URL* dan tautan [2]. Apabila pengguna menggunakan *tag* berulang kali pada setiap situs yang sama maka akan menimbulkan *spam* pada *social tagging*. *Spam* pada *social tagging* juga dapat terjadi apabila pengguna memakai *tag* yang tidak mendeskripsikan konten dari sebuah situs web [4]. Apabila terjadi masalah dalam *tag* seperti *spam*, maka dapat terjadi ambiguitas karena penggunaan *tag* dalam sebuah *bookmark* yang tidak merepresentasikan *bookmark* tersebut. Hal tersebut dapat membuat *user* membutuhkan usaha yang lebih banyak ketika melakukan pencarian pada mesin pencari [3][5].

Dalam penelitian ini dikembangkan sebuah aplikasi yang dapat menghitung *spam* sebuah *tag* berdasarkan nilai *spam factor*. Penelitian ini menggunakan *dataset* situs *social bookmarking*, *del.icio.us*. *Dataset* yang digunakan berasal dari situs *Hetrec* dan tercantum pada *2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011)* [6].

Pada penelitian ini, sebuah *tag* mengandung *spam* apabila *tag* tersebut tidak digunakan pada *bookmark* yang benar karena setiap *bookmark* mempunyai daftar *tag* yang benar untuk *bookmark* tersebut [7][8]. Untuk mengetahui sebuah *tag* terindikasi *spam* atau tidak, akan dibentuk sebuah daftar kumpulan *tag* yang benar untuk setiap dokumen (*bookmark*) yang disebut dengan $S(d)$ [7][8]. Setelah membentuk $S(d)$, maka selanjutnya dibentuk *posting* dari *random good user model* dan *random bad user model*. Selanjutnya digunakan metode *trusted moderator* untuk menghilangkan *posting* yang terindikasi *spam* [7][8]. Setelah mengimplementasikan *trusted moderator*, kemudian *tag* dan *bookmark* diurutkan dengan metode *occurrence-based search*. Untuk perhitungan nilai *spam*, digunakan metode *spam factor* yang menghitung nilai *spam* sebuah *tag* dari kesesuaian penggunaan *tag* tersebut dengan *bookmark* pada setiap *posting* [7][8]. Apabila *tag* tersebut termasuk dalam kumpulan *tag* yang benar pada *bookmark* / $S(d)$ maka tidak dianggap sebagai *spam* namun jika tidak termasuk dalam $S(d)$ ($T - S(d)$) maka dianggap sebagai *spam*. Nilai yang dihasilkan dari *spam factor* berkisar 0 - 1 dimana semakin besar nilai yang didapatkan, maka *tag* semakin terindikasi *spam* [7][8].

1.2 Topik dan Batasannya

Terkait permasalahan yang dipaparkan, dirumuskan masalah yang diangkat dalam penelitian ini, yang pertama adalah bagaimana cara menghitung nilai *spam* pada sebuah *tag* ?. Yang kedua adalah bagaimana pengaruh *trusted moderator*, *tag budget*, fraksi *trusted moderator*, dan persentase *good user* terhadap nilai *spam factor* yang dihasilkan ?.

Untuk batasan masalah pada penelitian ini saat pembentukan daftar *tag* yang benar pada setiap *bookmark* / $S(d)$ dibentuk secara acak dan *Dataset* yang digunakan merupakan *dataset* yang diunduh dari situs web <http://grouplens.org/datasets/hetrec-2011/> dan menggunakan 3 file yaitu file *usertaggedbookmarks-timestamps.csv* untuk *dataset user*, file *tags.csv* untuk *dataset tag*, dan file *bookmarks.csv* untuk *dataset bookmark*.

1.3 Tujuan

Tujuan yang dicapai dalam penelitian ini, yaitu mengimplementasikan metode untuk mengetahui cara menghitung nilai *spam* pada sebuah *tag* dan mengetahui pengaruh *trusted moderator*, *tag budget*, fraksi *trusted moderator*, dan persentase *good user* terhadap nilai *spam factor* yang dihasilkan.

1.4 Metoda Penelitian

Dalam pengerjaan penelitian ini terdapat beberapa metodologi penyelesaian masalah, yaitu :

1. Identifikasi Masalah

Pada tahap ini diidentifikasi permasalahan yang dipecahkan dalam penelitian ini, diantaranya membangun set *tag* yang benar ($S(d)$) pada sebuah dokumen (*bookmark*), menerapkan metode *trusted moderator*, dan menghitung *spam* dengan metode *spam factor*. Setelah masalah ditemukan kemudian menentukan rancangan sistem untuk mengatasi permasalahan tersebut.

2. Pengumpulan Data

Pada tahap ini penugumpulan data dilakukan dengan cara mencari *dataset* yang sesuai dengan kebutuhan pada sistem yang dibangun. Data yang dibutuhkan adalah data *user*, *tag* dan dokumen. Untuk data dokumen, karena studi kasus yang diangkat pada penelitian ini adalah situs *social bookmarking del.icio.us* maka dokumen dalam penelitian ini adalah sebuah *bookmark*.

3. Perancangan Sistem

Pada tahap ini dirancang sebuah sistem yang sesuai untuk menyelesaikan permasalahan yang telah diidentifikasi sebelumnya. Sistem yang dibangun meliputi $S(d)$, *random good user model* dan *random bad user model*, *join random user*, *trusted moderator*, *occurrence-based search* dan *spam factor*.

4. Pengujian Sistem

Pada tahap ini sistem yang telah dirancang diuji sesuai dengan kebutuhan dan parameter – parameter yang telah ditetapkan untuk pengujian. Mengubah parameter pada sistem yang telah dibangun, implementasi metode *trusted moderator* dan menghitung nilai *spam* pada kumpulan *posting* dengan metode *spam factor*.

5. Analisis Hasil

Untuk analisis maka dibandingkan nilai *spam factor* pada kumpulan *posting* yang mengimplementasikan *trusted moderator* dan kumpulan *posting* yang tidak mengimplementasikan *trusted moderator*. Selain itu, analisis hasil juga membandingkan nilai *spam factor* apabila parameter – parameter pada *tagging system* diubah.

2. Dasar Teori dan Metodologi

2.1 Tag

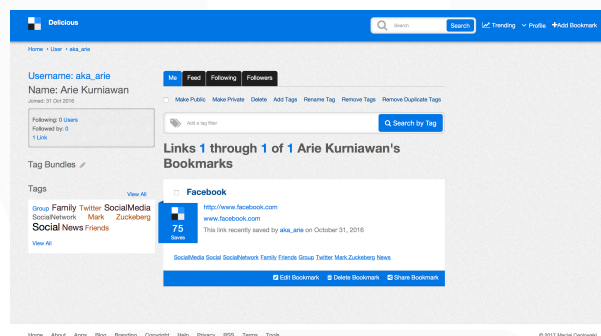
Tag atau label merupakan sebuah kata kunci, nama kategori atau metadata. *Tag* merupakan kata kunci tekstual yang dibentuk secara bebas [1]. *Tag* dibentuk oleh seorang *user* dan tidak dibentuk oleh seseorang yang ahli pada bidang tersebut, sehingga tidak mengikuti acuan formal manapun.

2.2 Social Tagging dan Social Bookmarking

Social Tagging merupakan sebuah proses pengguna menambahkan sebuah *metadata* dalam bentuk kata kunci (*tag*) untuk membubuhi keterangan dan mengkategorisasikan kumpulan *item* informasi seperti lagu, gambar, tautan web, produk dan sebagainya [9]. *Social bookmarking* merupakan salah satu jenis *social tagging* yang digunakan untuk mengkategorisasikan sebuah tautan web atau *URL* [3].

2.3 Situs Social Bookmarking del.icio.us

Situs *del.icio.us* merupakan salah satu contoh *social tagging* dengan memanfaatkan situs *social bookmarking*. Situs *del.icio.us* digunakan untuk melakukan *bookmark* terhadap sebuah situs yang akan kita kunjungi kembali [10]. Untuk tampilan situs *del.icio.us* dapat dilihat pada gambar 2-1.



Gambar 2-1 Tampilan Home Del.icio.us

2.4 Metadata

Metadata merupakan pendefinisian data yang menyediakan informasi atau dokumentasi dari data lainnya yang diatur dalam aplikasi atau *environment* yang sama. Sebagai contoh, *metadata* akan mendokumentasikan data tentang elemen data atau atribut (nama, ukuran, tipe data, dan sebagainya) [11].

2.5 Spam dan Spamming

Spam adalah penyalahgunaan sistem kirim pesan pada elektronik untuk mengirimkan pesan dalam jumlah yang besar yang tidak diinginkan dan mengirimkan pesan tersebut secara acak [12]. Sedangkan *spamming* adalah kegiatan dalam menyebarkan konten yang tidak diinginkan dan tidak bersangkutan pada *e-mail*, *instant messaging*, halaman web, dan lainnya [13]. *Spam* dan *spamming* dapat terjadi pada media dan sarana apapun [12][13].

2.6 Spam pada Social Tagging

Spam pada *social tagging* merupakan salah satu masalah dalam pemanfaatan teknologi *social tagging* pada situs web *social bookmarking*. Situs *social bookmarking* seperti *del.icio.us* sangat rentan terhadap *spam* [10]. *Spam* dapat terjadi ketika terdapat *tag* yang menimbulkan ambiguitas karena pemakaian *tag* tersebut yang tidak merepresentasikan konten dari sebuah situs web [4]. Pada penelitian ini, sebuah *tag* mengandung *spam* apabila *tag* tersebut tidak digunakan pada *bookmark* yang benar karena setiap *bookmark* mempunyai daftar *tag* yang benar untuk *bookmark* tersebut [7][8].

2.7 Tagging System Model (Definisi Posting dan $S(d)$)

Untuk *Tagging System Model* terbuat dari kumpulan \mathcal{D}/d yang berisikan dokumen (*bookmark*), kumpulan \mathcal{T}/t atau *tag* dan kumpulan \mathcal{U}/u yang berarti pengguna atau *user*, serta \mathcal{P} untuk relasi *posting* dalam kumpulan elemen atau *tuple* $[u, d, t]$ dalam P yang menunjukkan bahwa *user* u terdaftar pada *tag* t untuk *bookmark* d [7][8]. Contoh sebuah *posting* dideskripsikan pada tabel 2-1.

Tabel 2-1 Contoh *Posting*

User	Bookmark	Tag
1	San Francisco Symphony Keeping Score: Revolutions in Music	Quality

Untuk setiap dokumen $d \in \mathcal{D}$ mempunyai set $S(d) \subseteq \mathcal{T}$ untuk *tag* yang yang betul-betul mendeskripsikan dan sesuai dengan *bookmark* tersebut, selain itu untuk *spam* dapat didefinisikan dalam $\mathcal{T} - S(d)$ dalam dokumen d [7][8]. Untuk *tag* yang benar / $S(d)$ setiap dokumen (*bookmark*) dibentuk secara acak dan digunakan parameter untuk menentukan jumlah *tag* dalam setiap $S(d)$ [8]. Untuk contoh $S(d)$ dapat dilihat tabel 2-2.

Tabel 2-2 Contoh $S(d)$

Bookmark	Tag
San Francisco Symphony Keeping Score: Revolutions in Music	Quality, share, dropbox, music

Setiap *posting* yang dihasilkan tiap *random user model* sesuai dengan *tag budget* yang telah ditentukan pada parameter. *Tag Budget* digunakan untuk membatasi seberapa banyak *posting* yang dihasilkan oleh seorang *user* pada setiap *random user model*. *User* pada penelitian ini dibedakan menjadi dua jenis yaitu pengguna baik atau *good user* (menggunakan *tag* sesuai dengan *bookmark* / $S(d)$) dan pengguna buruk atau *bad user* (menggunakan *tag* tidak sesuai dengan *bookmark* / $\mathcal{T} - S(d)$). Untuk sistem ini maka *tag budget good user* disimbolkan dengan pg dan *tag budget bad user* disimbolkan dengan pb [8].

2.8 Random User Model

Random user model merupakan model dalam membangun sebuah *posting*. Untuk *random user model* diciptakan dua model, yaitu *Random Good User model* dan *Random Bad User model*. Untuk membangkitkan populasi *random good user* maka *tag* yang digunakan pada saat membentuk *posting* merupakan *tag* yang sesuai dengan *bookmark* / $S(d)$ pada *posting* tersebut sedangkan untuk membangkitkan populasi *random bad user* maka saat membuat *posting* akan digunakan *tag* yang tidak sesuai dengan *bookmark* yang telah dipilih / $\mathcal{T} - S(d)$ pada *posting* tersebut [8]. Untuk penggunaan *random user model* pengguna baik disimbolkan oleh G dan pengguna buruk disimbolkan oleh B , maka $\mathcal{U} = G \cup B$ tetapi $G \cap B = \emptyset$. Untuk menghasilkan / membangkitkan *posting* digunakan algoritma pembangkitan seperti yang dijelaskan pada algoritma 2.1 dan 2.2.

Random Good User Model

for each user $u \in G$ do

 for each posting $j = 1$ to pg do

 select at random a document d from D ;

 select at random a tag t from $S(d)$;

 record the posting: user u tags d with t .

(2.1)

Random Bad User Model :

for each user $u \in B$ do

 for each posting $j = 1$ to pb do

 select at random a document d from D ;

 select at random an incorrect tag t from $\mathcal{T} - S(d)$;

 record the posting: user u tags d with t .

(2.2)

Keterangan :

$u = User$, $B = Bad User$, $G = Good User$, $pg = tag budget$ untuk *good user*, $pb = tag budget$ untuk *bad user*, $d =$ dokumen yang dipilih, $D =$ kumpulan dokumen, $t = tag$, $S(d) = tag$ yang sesuai dengan *bookmark* pada *posting* tersebut, $\mathcal{T} - S(d) = tag$ yang tidak sesuai dengan *bookmark* pada *posting* tersebut

Untuk *Random Bad User Model* menghasilkan *posting* yang mengandung *spam* pada sistem [7][8].

2.9 Algoritma Trusted Moderator

Trusted Moderator dapat mengurangi *posting* buruk dengan mengecek *posting* dari pengguna. Sebuah *posting* dikatakan buruk jika merupakan *posting* yang dihasilkan oleh *random bad user model*. Algoritma ini

akan mendeteksi dan menghilangkan *spam* pada kumpulan *posting* (gabungan *posting random user*) sesuai dengan fraksi (f) yang telah ditetapkan pada parameter [7][8]. Fungsi dari algoritma *trusted moderator* dideskripsikan pada algoritma 2.3.

let $D_f \subseteq D$ containing a fraction f of D 's documents;
for each document $d \in D_f$ do
 for each incorrect posting $[u, d, t]$
 eliminate all entries $[u, *, *]$. (2.3)

2.10 Algoritma Occurrence-Based Search

Algoritma ini mengurutkan *tag* dan *bookmark* berdasarkan jumlah *tag* dan *bookmark* yang paling banyak digunakan pada kumpulan *posting* kemudian algoritma menampilkan semua hasil pencarian [7][8]. Selain itu pada penelitian ini, untuk algoritma *occurrence-based search* akan menampilkan detail dari *tag* dan *bookmark* yang dipakai. Model pencarian ini dideskripsikan pada algoritma 2.4.

Occurrence-Based Search:

rank documents by decreasing number of postings in P that contain t ;
return top K documents. (2.4)

2.11 Spam Factor

Spam Factor digunakan untuk mengukur *spam* untuk setiap *tag* (t) [7][8]. Cara menghitung *spam* direpresentasikan dalam bentuk nilai yang dihitung pada persamaan 2.5.

$$\text{SpamFactor}(t) = \frac{\sum_{d_i \in D_K} w(d_i) \cdot \frac{1}{i}}{H_K} \quad (2.5)$$

dimana

$$D_K = [d_1, d_2, \dots, d_K], \text{ dengan peringkat } (d_i - 1, t), 2 \leq i \leq K$$

$K = \text{Jumlah Document}$

dan

$$w(d_i) = \begin{cases} 1 & \text{ketika } d_i \text{ merupakan dokumen yang buruk} \\ 0 & \text{ketika } d_i \text{ merupakan dokumen yang baik} \end{cases}$$

dan H_K merupakan K^{th} *harmonic number* yang merupakan urutan dari penjumlahan jumlah parsial deret harmonik [14] dari jumlah kebalikan dari nilai awal K seperti dijelaskan pada persamaan 2.6.

$$H_K = \sum_{i \in [1..K]} \frac{1}{i} \quad (2.6)$$

Untuk perhitungan *spam factor* untuk setiap *tag*, diurutkan terlebih dahulu *bookmark* yang paling banyak menggunakan *tag* tersebut (D_K). Apabila terdapat penggunaan *tag* pada jumlah yang sama tetapi pada dokumen / *bookmark* yang berbeda maka akan diurutkan dokumen baik terlebih dahulu kemudian dokumen buruk [8]. Sebuah dokumen d dikatakan buruk jika t bukan *tag* yang benar dari *bookmark* atau dengan kata lain $t \notin S(d)$ dan sebaliknya. *Spam Factor* memberikan nilai *spam* setiap *tag*. Nilai dari *Spam Factor* berkisar antara 0 sampai 1 [7].

3. Pembahasan

3.1 Gambaran Umum Sistem

Sistem yang dibangun pada penelitian tugas akhir ini adalah sebuah sistem yang dapat menghitung nilai *spam* untuk setiap *tag* pada kumpulan *join posting*. Tahapan untuk membangun sistem ini terbagi atas enam proses utama, yaitu membentuk $S(d)$, membangkitkan *posting random good user model* dan *random bad user model*, *join random user model*, *trusted moderator*, *occurrence-based search*, dan *spam factor*. Pada gambar 3-1 merupakan *flow chart* dari gambaran umum sistem.

3.2 Parameter Pengujian

Parameter yang digunakan pada penelitian ini dapat dilihat pada tabel 3-1. Parameter ini merupakan parameter yang mengacu pada jurnal "*Combating spam in tagging systems: An evaluation*" [8]. Persentase *good user* ditentukan untuk menentukan seberapa banyak *good user* dari total *user*. *tag budget* ditentukan untuk jumlah *posting* setiap *user* pada jenis *posting*. Parameter *fair average* digunakan untuk menentukan apakah memukulratakan nilai $S(d)$ atau tidak. Untuk *fair average* dapat bernilai *true* atau *false*.

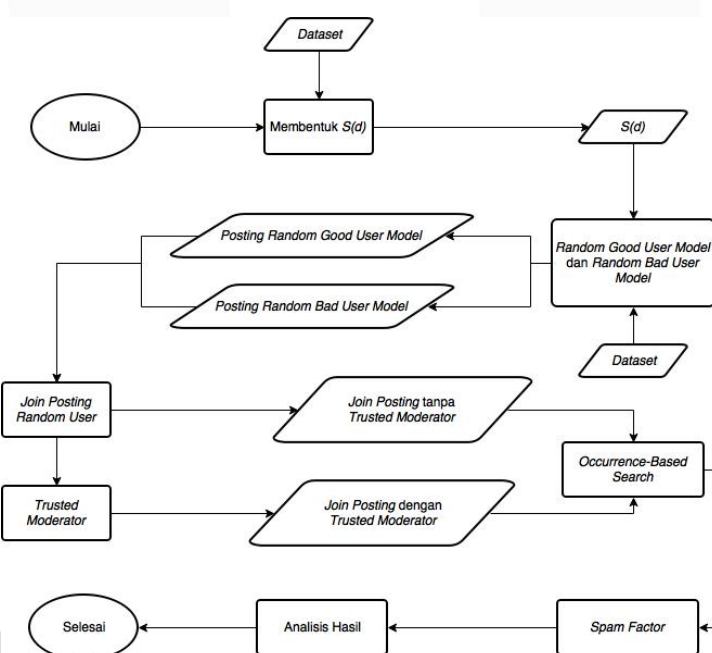
Pada penelitian ini beberapa parameter bersifat *random* dalam implementasi maka model yang dibangun akan menggunakan iterasi. Iterasi dilakukan untuk mendapatkan nilai rata-rata dari semua nilai yang didapatkan, untuk penetapan jumlah iterasi sendiri dilakukan agar nilai analisis yang didapatkan lebih dari 1 kali. Pada penelitian ini tidak dihitung akurasi karena data yang digunakan pada penelitian sesuai dengan data yang dipilih pada *dataset* sehingga tidak akan terjadi kemunculan *data* diluar dari *dataset*.

Tabel 3-1 Parameter Pengujian

Simbol	Deskripsi
\mathcal{D}	Jumlah Dokumen / <i>bookmark</i>
T	Jumlah <i>tag</i>
\mathcal{U}	Jumlah <i>user</i>
\mathcal{G}	Persentase <i>good user</i>
pg	<i>Tag budget good user (posting)</i>
pb	<i>Tag budget bad user (posting)</i>
$s(d)$	Ukuran $S(d)$ tiap dokumen (<i>tag</i>)
<i>fair average</i>	Penentuan jumlah $S(d)$ adil atau tidak (<i>true</i> atau <i>false</i>)
f	Persentase fraksi yang dicek oleh <i>trusted moderator</i>
i	iterasi / jumlah percobaan

3.3 Hasil Pengujian

Untuk hasil pengujian, diberikan beberapa skenario dengan nilai parameter yang berbeda. Untuk beberapa skenario diimplementasikan dua sistem, yaitu sistem yang menggunakan metode *trusted moderator* dan tidak menggunakan *trusted moderator*. Berikut beberapa pengujian dan skenario yang dihasilkan :



3-1 Gambaran Umum Sistem

3.3.1 Skenario 1 (Pengaruh *Trusted Moderator*)

Pada skenario pertama ini akan ditampilkan hasil *spam factor* sistem yang menggunakan *trusted moderator* dan sistem yang tidak memakai *trusted moderator*. Selain itu, tidak diperlihatkan hasil tiap prosesnya tetapi langsung ke *file summary* dan *file avg summary* dimana pada *file avg summary* tersebut akan menampilkan hasil akhir, seperti parameter yang digunakan, nilai maksimum dari semua iterasi, nilai minimum dari semua iterasi dan nilai *mean* dari jumlah nilai *mean* tiap iterasi. Untuk file *avg_summary* dijelaskan pada tabel 3-2

Tabel 3-2 *File Avg Summary* Skenario 1

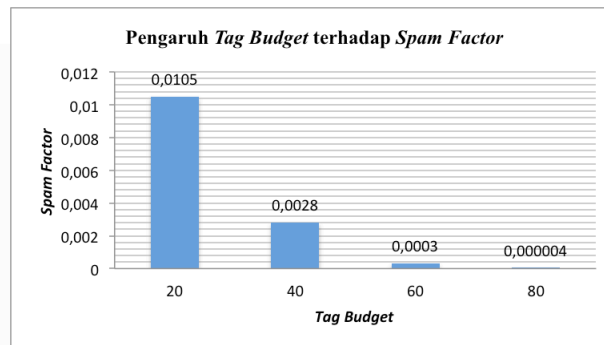
Parameter	dengan <i>trusted moderator</i>	tanpa <i>trusted moderator</i>
Total User	1.000	1.000
Total Document	10.000	10.000
Total Tag	500	500
Set Tag / $S(d)$	25	25
Fair Average	<i>True</i>	<i>True</i>

<i>Good User Percentage</i>	90 %	90 %
<i>Tag Budget Good User</i>	10	10
<i>Tag Budget Bad User</i>	10	10
<i>Trusted Moderator Fraction</i>	5 %	5 %
<i>Max Score</i>	0.16	0.16
<i>Min Score</i>	0	0
<i>Mean</i>	0.0193	0.0299

Ketika sistem menggunakan *trusted moderator* maka nilai rata-rata *spam factor* dari tiap iterasi adalah 0.0193 sedangkan sistem tanpa *trusted moderator* 0.0299. Dapat disimpulkan bahwa dengan menggunakan metode *trusted moderator* nilai *spam factor* yang dihasilkan akan berkurang sehingga indikasi sebuah *tag* bersifat *spam* dapat dikurangi.

3.3.2 Skenario 2 (Pengaruh *Tag Budget*)

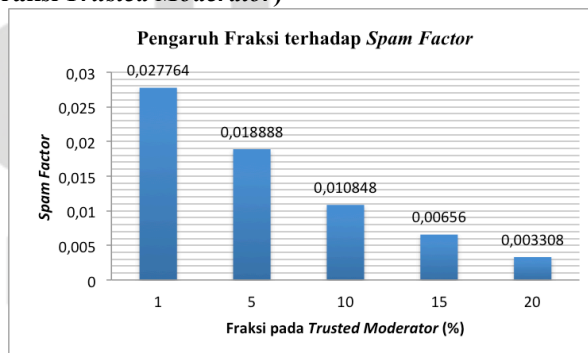
Pada skenario ini menguji nilai *spam factor* ketika sistem menggunakan *trusted moderator* tetapi menggunakan *tag budget* yang berbeda. Dimana nilai *tag budget* yang dipakai adalah 20, 40, 60, dan 80. Selain itu *tag budget* ini berlaku untuk *good user* dan *bad user*. Nilai *spam factor* yang dibandingkan adalah nilai rata – rata *spam factor* 5 kali iterasi. Untuk hasil yang didapatkan, terdapat pada gambar 4-15. Untuk hasilnya dapat dilihat pada gambar 3-2.



Gambar 3-2 Grafik nilai *spam factor* skenario 2

Pada hasil diatas memperlihatkan bahwa semakin besar nilai *tag budget* maka semakin kecil pula nilai *spam factor* yang dihasilkan, ini terjadi akibat *posting* yang dihasilkan oleh kedua jenis *user* bertambah dan khususnya untuk *good user* dimana semakin banyak *posting* yang *good user* hasilkan maka semakin banyak *posting* yang bersifat *good posting (spam free)*. Sebagai contoh pada parameter ini, digunakan *user* sebanyak 1.000, dimana persentase *good user* sebanyak 90 % (900 *user*). Apabila *tag budget* sebesar 20 maka total *posting good user* adalah sebanyak $900 * 20 = 18.000$ *posting*. Sedangkan untuk *posting bad user* hanya sebesar 2.000 *posting* ($100 * 20 = 2.000$). Sehingga ketika melakukan pencarian menggunakan *occurrence-based search* maka setiap *tag* lebih banyak berperan sebagai *posting good user*. Ketika populasi *posting good user* lebih banyak maka ketika melakukan perhitungan *spam factor* nilai yang dihasilkan lebih kecil karena *good document* lebih banyak ditemukan.

3.3.3 Skenario 3 (Pengaruh fraksi *Trusted Moderator*)



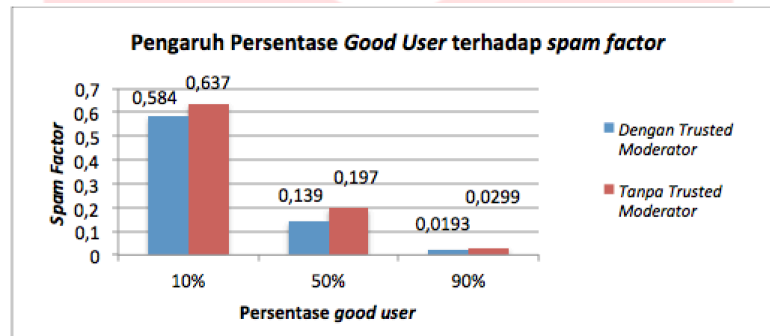
Gambar 3-3 Grafik nilai *spam factor* skenario 3

Skenario ketiga ini menggunakan parameter skenario pertama dan menggunakan fraksi 1 %, 5 %, 10 %, 15 %, 20 %, dan 50 %. Hasil nilai *spam factor* yang didapatkan dipaparkan pada gambar 3-3.

Semakin besar persentase fraksi maka semakin kecil nilai *spam factor* yang dihasilkan atau dengan kata lain, sistem akan semakin terbebas dari *spam*. Hal tersebut membuktikan bahwa semakin besar nilai fraksi yang “dicek” oleh *trusted moderator* maka semakin besar peluang *posting bad user* akan ditemukan dan dihapus dari sistem sehingga membuat nilai *spam factor* semakin kecil.

3.3.4 Skenario 4 (Pengaruh persentase *good user*)

Untuk skenario keempat akan diuji ketika nilai persentase *good user* sebesar 90%, sebesar 50 % atau dengan kata lain persentase *good user* dan *bad user* sama dan ketika persentase *good user* hanya sebesar 10 %.

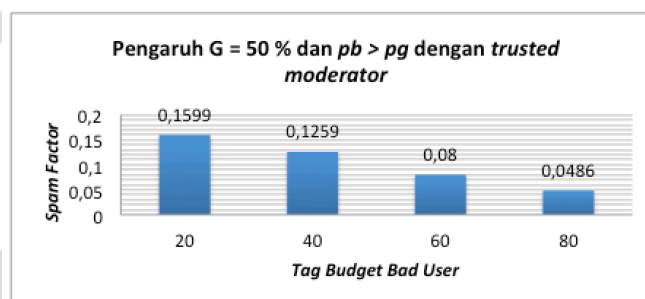


Gambar 3-4 Grafik nilai *spam factor* skenario 4

Dari gambar 3-4 diketahui bahwa semakin kecil persentase *good user* maka nilai *spam factor* yang dihasilkan semakin besar. Untuk penjelasan hal tersebut diberikan contoh ketika sistem menggunakan *trusted moderator* dan persentase *good user* sebesar 50 %. Ketika persentase *good user* 50 % maka jumlah *bad user* dan *good user* sama yaitu 500 user (total 1000 user) sehingga setiap *good user* dan *bad user* mempunyai jumlah *posting* yang sama yaitu sebesar 5.000 *posting* ($user * tag\ budget$). Pada iterasi - 1 didapatkan 218 *bad user* berbeda dalam fraksi, jika 1 *bad user* menghasilkan 10 *posting* maka sistem menghilangkan 2.180 *posting* dari total 5.000 *posting bad user* (43,6 %). Berbeda ketika persentase *good user* 90 % (skenario 1), pada iterasi - 1 ditemukan 35 *bad user* berbeda dalam fraksi, jika 1 *bad user* menghasilkan 10 *posting* maka dihilangkan 350 *posting* dari total 1.000 *posting bad user* (35 %). Untuk penentuan jumlah *bad user* yang didapatkan, dilakukan dengan cara mengurangi jumlah awal *posting* pada *file join posting random user* dengan *file result trusted moderator* yang merupakan *posting* setelah sistem melalui proses *trusted moderator*.

Meskipun secara persentase, ketika persentase *good user* 50 %, lebih besar dalam mengurangi *bad user posting* tetapi secara keseluruhan berbeda karena pada sistem dengan persentase *good user* 50 %, jumlah *bad user posting* setelah menggunakan *trusted moderator* tersisa sebesar 2.820 *posting bad user* (282 *bad user*) dari total 7.820 *posting* (36 % dari total *posting*) sedangkan pada skenario pertama tersisa sebesar 650 *posting bad user* (65 *bad user*) dari total 9.650 *posting* (6,75 % dari total *posting*). Secara keseluruhan jumlah *posting bad user* sistem dengan persentase *good user* 50 % lebih banyak sehingga menghasilkan nilai *spam factor* lebih besar.

3.3.5 Skenario 5 (Percobaan $G = 50\%$ & jumlah $pb > pg$)



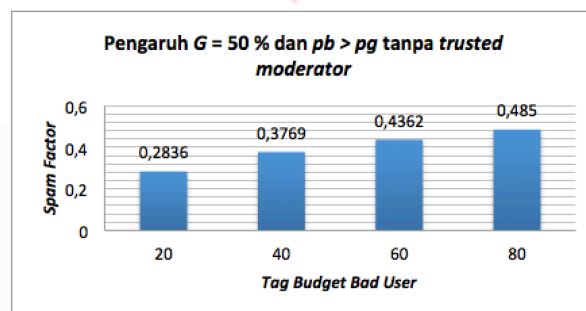
Gambar 3-5 Grafik nilai *spam factor* skenario 5 sistem dengan *trusted moderator*

Pada gambar 3-5 diketahui bahwa nilai *spam factor* dari sistem yang menggunakan *trusted moderator* semakin kecil ketika *tag budget bad user* semakin besar meskipun persentase *good user* sebesar 50 % (*bad user* 50 %). Sebagai contoh ketika $pb = 20$ pada iterasi - 1, dari 500 *bad user* maka jumlah *posting* yang didapatkan sebesar 10.000 *posting* ($500\ bad\ user * 20\ tag\ budget$). Meskipun jumlah *posting bad user* 66,66 % dari total *posting* (15.000 *posting*) tetapi pada saat digunakan *trusted moderator* terpilih sebanyak 329 *bad user* berbeda

dalam fraksi. Jika $pb = 20$, maka terdapat 6.580 *posting* yang dihilangkan atau sekitar 65,8 % dari total *posting bad user*. Setelah melalui *trusted moderator* maka tersisa sekitar 8.420 *posting*. Dari total *posting* tersebut, tersisa 3.420 *posting bad user* dari 171 *bad user* (40,6 % *posting bad user* dari seluruh *posting*).

Selanjutnya ketika $pb = 40$ pada iterasi - 2, dari 500 *bad user* maka jumlah *posting* yang didapatkan sebesar 20.000 *posting* ($500 \text{ bad user} * 40 \text{ tag budget}$). Jumlah *posting bad user* 80 % dari total *posting* (25.000 *posting*) tetapi pada saat digunakan *trusted moderator* terpilih sebanyak 438 *bad user* berbeda dalam fraksi. Jika $pb = 40$, maka terdapat 17.520 *posting* yang dihilangkan atau sekitar 87,6 % dari total *posting bad user*. Setelah melalui *trusted moderator* maka tersisa sekitar 7.480 *posting*. Dari total *posting* tersebut, tersisa 2.480 *posting bad user* dari 62 *bad user* (33,15 % *posting bad user* dari seluruh *posting*).

Dari persentase yang didapatkan dari kedua *tag budget* tersebut maka dapat disimpulkan ketika *tag budget* semakin besar maka persentase hilangnya *posting bad user* dalam sistem semakin besar. Hal ini terjadi karena metode *trusted moderator* mendeteksi *bad user* lebih banyak ketika *tag budget* bertambah atau secara keseluruhan persentase *posting bad user* yang dihilangkan lebih besar dari total *posting*, terbukti pada saat *tag budget* = 20, maka sistem memiliki 40,6 % *posting* yang bersifat *spam* (65,8 % *posting bad user* dihilangkan) dan pada saat *tag budget* = 40, sistem hanya memiliki 33,15 % *posting* bersifat *spam* (87,6 % *posting bad user* dihilangkan). Nilai *spam factor* pada saat tidak menggunakan *trusted moderator* dideskripsikan pada gambar 3-6.



Gambar 3-6 Grafik nilai *spam factor* skenario 5 sistem tanpa *trusted moderator*

Ketika tidak menggunakan *trusted moderator* nilai yang dihasilkan bertambah seiring bertambahnya *tag budget*. Ini diakibatkan karena populasi dari *posting bad user* semakin banyak. Sebagai contoh ketika *tag budget bad user* = 20 maka *posting bad user* sebanyak 10.000 *posting* atau sekitar 66,66 % dari total *posting* (total *posting* 15.000). Ketika *trusted moderator* tidak digunakan maka *posting bad user* tidak dapat dikurangi sehingga nilai *spam factor* yang dihasilkan sesuai dengan persentase *posting bad user* dari total *posting*.

4. Kesimpulan

Berdasarkan hasil pengujian dan analisis pada penelitian ini, maka dapat diambil beberapa kesimpulan sebagai berikut :

1. Untuk menghitung nilai *spam* untuk setiap *tag* maka digunakan *spam factor*. Untuk mengimplementasikan *spam factor* maka dibangun sebuah sistem yang membangun $s(d)$ untuk setiap *bookmark*. Setelah membentuk $s(d)$ maka dibentuk *posting* dari *random good user* dan *random bad user*. Setelah membentuk *posting* kemudian dilakukan *join posting*. Kemudian digunakan metode *trusted moderator* dimana setiap *posting* yang terdeteksi dan dihasilkan oleh *bad user* dihilangkan dari *join posting*. Setelah itu dilakukan *occurrence-based search* dimana metode tersebut melakukan pemeringkatan untuk setiap *tag* dan *bookmark* berdasarkan intensitas pemakaiannya.
2. Nilai *spam factor* yang dihasilkan bervariasi dan bergantung pada pemakaian *trusted moderator* dan parameter yang digunakan. Ketika sistem menggunakan *trusted moderator* maka nilai *spam factor* yang dihasilkan lebih kecil dibandingkan sistem yang tidak menggunakan *trusted moderator*. Semakin besar nilai fraksi *trusted moderator* yang digunakan maka semakin kecil nilai *spam factor* yang dihasilkan.
3. Untuk parameter *tag budget*, ketika sistem menggunakan *trusted moderator*, persentase *good user* sebanyak 90 % dan *tag budget good user* dan *bad user* sama, maka menghasilkan nilai yang semakin kecil seiring bertambahnya nilai dari *tag budget*.
4. Untuk persentase *good user* ketika *tag budget good user* sama dengan *tag budget bad user* maka nilai *spam factor* yang dihasilkan akan semakin kecil seiring dengan bertambahnya persentase *good user* baik sistem dengan *trusted moderator* maupun tidak.
5. Nilai *spam factor* ketika *tag budget good user* < *tag budget bad user* dan persentase *good user* 50 % bergantung pada pemakaian *trusted moderator*. Ketika sistem menggunakan *trusted moderator* maka nilai yang dihasilkan akan semakin kecil seiring dengan bertambahnya *tag budget*. Tetapi jika tidak menggunakan *trusted moderator*, nilai *spam factor* bertambah seiring dengan bertambahnya *tag budget*.

5.2 Saran

Berikut ini adalah beberapa saran dari penulis untuk pengembangan penelitian ini ke depannya :

1. Sebaiknya menggunakan *dataset* yang mempunyai data $S(d)$ untuk setiap *bookmark* (tidak dibangun secara acak) sehingga hasil nilai *spam factor* yang didapatkan dapat dianalisis untuk rekomendasi pemakaian *tag* yang benar untuk sebuah *bookmark*.
2. *Dataset* (*user*, *bookmark*, dan *tag*) dapat ditambahkan ke dalam skala yang lebih besar.

Daftar Pustaka

- [1] Guy, M., & Tonkin, E. (2006). Tidying up tags. *D-lib Magazine*, 12(1), 1082-9873.
- [2] Trant, J. (2009). Studying social tagging and folksonomy: A review and framework. *Journal of Digital Information*, 10(1).
- [3] Gupta, M., Li, R., Yin, Z., & Han, J. (2010). Survey on social tagging techniques. *ACM Sigkdd Explorations Newsletter*, 12(1), 58-72.
- [4] Heymann, P., Koutrika, G., & Garcia-Molina, H. (2008, February). Can social bookmarking improve web search?. In *Proceedings of the 2008 International Conference on Web Search and Data Mining* (pp. 195-206). ACM.
- [5] Song, Y., Zhuang, Z., Li, H., Zhao, Q., Li, J., Lee, W. C., & Giles, C. L. (2008, July). Real-time automatic tag recommendation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 515-522). ACM.
- [6] Cantador, I., Brusilovsky, P. L., & Kuflik, T. (2011). Second workshop on information heterogeneity and fusion in recommender systems (HetRec2011).
- [7] Koutrika, G., Effendi, F. A., Gyöngyi, Z., Heymann, P., & Garcia-Molina, H. (2007, May). Combating spam in tagging systems. In *Proceedings of the 3rd international workshop on Adversarial information retrieval on the web* (pp. 57-64). ACM.
- [8] Koutrika, G., Effendi, F. A., Gyöngyi, Z., Heymann, P., & Garcia-Molina, H. (2008). Combating spam in tagging systems: An evaluation. *ACM Transactions on the Web (TWEB)*, 2(4), 22.
- [9] Symeonidis, P., Nanopoulos, A., & Manolopoulos, Y. (2008, October). Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM conference on Recommender systems* (pp. 43-50). ACM.
- [10] Wetzker, R., Zimmermann, C., & Bauckhage, C. (2008, July). Analyzing social bookmarking systems: A del.icio.us cookbook. In *Proceedings of the ECAI 2008 Mining Social Data Workshop* (pp. 26-30).
- [11] Rosenfeld, L., & Morville, P. (2002). *Information architecture for the world wide web*. " O'Reilly Media, Inc."
- [12] Whitworth, B., & Whitworth, E. (2004). Spam and the social-technical gap. *Computer*, 37(10), 38-45.
- [13] Hayati, P., Potdar, V., Talevski, A., Firoozeh, N., Sarenche, S., & Yeganeh, E. A. (2010, April). Definition of spam 2.0: New spamming boom. In *4th IEEE International Conference on Digital Ecosystems and Technologies* (pp. 580-584). IEEE.
- [14] Rochowicz Jr, J. A. (2015). Harmonic Numbers: Insights, Approximations and Applications. *Spreadsheets in Education (eJSiE)*, 8(2), 4.