Scientific Journal of Informatics
Vol. 2, No. 2, November 2015
p-ISSN 2407-7658 http://journal.unnes.ac.id/nju/index.php/sji e-ISSN 2460-0040

Implementasi *Vector Space* Model dalam Pembangkitan *Frequently Asked Questions* Otomatis dan Solusi yang Relevan untuk Keluhan Pelanggan

Abdul Aziz[1], Ristu Saptono[2], Kartika Permatasari Suryajaya[3]

[1,2,3]Program Studi Informatika, FMIPA, Universitas Negeri Sebelas Maret Surakarta
Email: [1]aaziz@staff.uns.ac.id,[2]ristu.saptono@staff.uns.ac.id,[3]kpsuryajaya@student.uns.ac.id

Abstrak

Salah satu keunggulan dari sebuah lembaga/unit pelayanan adalah seberapa cepat dan akurat dalam menangani keluhan pelanggan. Keluhan yang disampaikan pelanggan umumnya memiliki kesamaan dengan keluhan-keluhan sebelumnya, sehingga solusi dari keluhan baru dapat didasarkan pada solusi yang diberikan pada keluhan lama. *Vector Space Model* (VSM) merupakan salah satu model yang digunakan untuk mengetahui kemiripan dokumen, yang digunakan dalam membangkitkan FAQ otomatis. Pembobotan *term* dilakukan dengan teknik *Term Frequency-Inverse Document Frequency* (TF-IDF). Kombinasi notasi TF-IDF yang dibandingkan adalah TF-IDF itu sendiri, modifikasi logaritmik TF dan modifikasi logaritmik IDF. *Similarity measure* yang digunakan adalah *cosine similarity*. Hasil dari penelitian ini adalah algoritma VSM dengan pembobotan TF-IDF dapat digunakan untuk membangkitkan FAQ otomatis dan solusi yang relevan. Berdasarkan hasil perhitungan *accuracy* pada masing-masing percobaan dapat disimpulkan bahwa pada *threshold* 0.5, kombinasi notasi TF-IDF yang memiliki nilai rata-rata *accuracy* dan *precision* tertinggi adalah modifikasi pertama, yaitu masing-masing sebesar 62.09% dan 55.15%. Sedangkan untuk *threshold* 0.65 yang memiliki nilai rata-rata *accuracy* dan *precision* tertinggi adalah TF-IDF, yaitu masing-masing sebesar 83.18% dan 68.35%. Selain itu percobaan dengan menggunakan 171 data, TF-IDF dan *threshold* 0.65 dapat membangkitkan 27 FAQ, yaitu dengan persentase 70.37% relevan.

Kata Kunci: *Cosine Similarity, Term Frequency-Inverse Document Frequency, Text Mining, Vector Space Model*

1. PENDAHULUAN

Pelanggan merupakan salah satu elemen penting dalam sebuah perusahaan [1]. Ketidakpuasan yang diakibatkan adanya perbedaan antara harapan dan kemampuan dari sebuah produk atau jasa yang diterima oleh konsumen akan menimbulkan *negative effect* yang diyakini akan berpengaruh terhadap kesetiaan konsumen [2]. Sehingga diperlukan penangan keluhan pelanggan di sebuah perusahaan.

UPT PUSKOM UNS merupakan salah satu unit pelaksana teknis di UNS, yang memiliki tugas pokok untuk memberikan fasilitas laboratorium komputer, *training* dan *technical support* [3]. Dalam menjalankan fungsinya, UPT PUSKOM UNS memerlukan sebuah penangan keluhan pelanggan yang berasal dari civitas akademika UNS (selanjutnya disebut pelanggan). Keluhan-keluhan yang disampaikan pelanggan akan diberi solusi yang di dasarkan pada keluhan-keluhan sebelumnya yang memiliki kemiripan dengan keluhan yang baru. Oleh karena itu diperlukan metode untuk menghitung kemiripan antara keluhan baru dengan keluhan-keluhan yang telah lampau. Hasil perhitungan kemiripan tersebut dapat digunakan dalam pembangkitan

Frequently Asked Questions (FAQ) otomatis dan solusi yang relevan untuk keluhan pelanggan di UPT PUSKOM UNS.

Beberapa pemodelan yang dapat digunakan untuk menghitung kemiripan antar dokumen, antara lain adalah *Levenshtein Distance* [4], *Latent Semantic Analysis* (LSA) [5], *Exact Match* [6] dan *Vector Space Model* (VSM) [7]. Pemodelan VSM memiliki cara kerja yang efisien, mudah dalam representasi dan dapat diimplementasikan dalam *document-matching* [8]. Artikel ini menggunakan model VSM dalam pembangkitan FAQ otomatis dan solusi yang relevan untuk keluhan pelanggan di UPT PUSKOM UNS dengan pembobotan *term* dilakukan dengan teknik *Term Frequency-Inverse Document Frequency* (TF-IDF). Selain itu akan digunakan 3 kombinasi notasi TF-IDF yang akan dibandingkan tingkat akurasinya, yaitu TF-DF itu sendiri, modifikasi logaritmik TF dan modifikasi logaritmik IDF. Sedangkan *similarity measure* yang digunakan adalah *cosine similarity*.

2. METODE

2.1. *Text Mining*

Text mining digunakan untuk mengolah dokumen sebelum dilakukan proses *similarity*. Proses *text mining* dibagi menjadi kedalam tiga buah proses, yaitu *text preprocessing*, *text transforming* dan *pattern discovery* [9].

Text preprocessing merupakan tindakan menghilangkan karakter-karakter tertentu yang terkandung dalam dokumen, seperti titik, koma, tanda petik dan lain-lain serta mengubah semua huruf kapital menjadi huruf kecil. Selain itu, dalam tahap *text preprocessing* ini dilakukan *tokenization*. *Tokenization* merupakan proses pengolahan token yang terdapat dalam rangkaian teks [10], sehingga dokumen akan dipecah-pecah menjadi *term.*

Text transforming merupakan proses *stopwords removal* dan *stemming* pada dokumen. *Stopword* merupakan kata yang sering muncul dalam dokumen tetapi tidak memiliki makna yang berarti [9]. Adapun *Stemming* adalah proses penghilangan *prefix* dan *suffix* dari kata untuk mendapatkan kata dasar [10]. Dalam artikel ini algoritma yang akan digunakan pada proses *stemming* adalah algoritma Nazief & Adriani [11].

Pattern discovery bertujuan untuk mengukur kemiripan antar dokumen. Proses diawali dengan menghitung bobot *term* menggunakan algoritma TF-IDF. *Term* yang telah memiliki bobot tersebut akan direpresentasikan ke dalam model ruang vektor (*vector space model)*. Elemen-elemen vektor inilah yang kemudian akan digunakan untuk menghitung kemiripan antar dokumen. Pengukuran kemiripan dokumen akan dilakukan dengan menggunakan algoritma *cosine similarity*, yaitu menghitung nilai *cosine* yang dibentuk oleh vektor antar dokumen.

2.2. *Term Frequency – Inverse Document Frequency (TF-IDF)*

TF-IDF merupakan suatu cara untuk memberikan bobot hubungan suatu kata atau *term* terhadap suatu dokumen [12]. Algoritma ini menggabungkan dua konsep untuk perhitungan bobot, yaitu frekuensi kemunculan sebuah kata di dalam sebuah dokumen tertentu atau TF dan *inverse* frekuensi dokumen yang mengandung kata tersebut atau IDF [13].

TF merupakan *term frequency* pada sebuah dokumen [14]. Biasanya, *term frequency* pada sebuah dokumen langsung digunakan untuk nilai TF. Sehingga nilai TF adalah jumlah *term* i tersebut.

$$\boldsymbol{TF_i = tf_{ij}} \quad (1)$$

Dimana TF_i merupakan frekuensi dari *term* i pada sebuah dokumen j. Konsep dasar IDF telah diperkenalkan oleh Robertson [15, 14]. Kemudian, diskusi antara Sparck Jones & Robertson menghasilkan rumus berikut [15]:

$$\boldsymbol{IDF_i = \left(log\left(\frac{N}{n_j}\right)\right) + 1} \quad (2)$$

Dimana N merupakan jumlah total dokumen dan n_j merupakan jumlah dokumen yang mengandung *term* i.

Selain itu, terdapat beberapa notasi yang dapat digunakan untuk menghitung TF-IDF. Tabel 1 merupakan beberapa notasi pada TF-IDF [13]:

Tabel 1. Beberapa notasi TF-IDF [13]

Abjad Pertama	Persamaan	Deskripsi
Term Frequency		
N	TF	*Raw term frequency*
L	$TF = 1 + (\log\ TF)$	*Logarithm term frequency*
B	$1/0$	*Binary term frequency*
A	$TF = 0.5\ (0.5\ x\ \frac{TF}{\max TF})$	*Augmented term frequency*
Inverse Document Frequency		
N	1	IDF tidak diperhitungkan
T	$IDF = \left(\log\left(\frac{D}{DF}\right)\right) + 1$	Nilai logaritmik dari IDF

Notasi-notasi pada Tabel 1 dapat dikombinasikan untuk menghitung pembobotan TD-IDF. Perhitungan menggunakan kombinasi notasi untuk pembobotan TF-IDF, yaitu:

1. Kombinasi N.T

$$\boldsymbol{TF\ x\ IDF = TF\ x\ \left(\left(\log\left(\frac{D}{DF}\right)\right) + 1\right)} \quad (3)$$

2. Kombinasi L.T

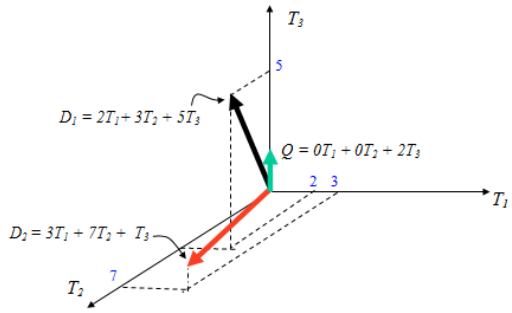
$$\boldsymbol{TF\ x\ IDF = (1 + \log\ TF)\ x\ \left(\left(\log\left(\frac{D}{DF}\right)\right) + 1\right)} \quad (4)$$

3. Kombinasi L.N

$$\boldsymbol{TF\ x\ IDF = (1 + \log\ TF)\ x\ 1} \quad (5)$$

2.3. *Vector Space Model (VSM)*

VSM merupakan model *Information Retrieval* yang merepresentasikan dokumen dan *query* sebagai vektor pada ruang multidimensi. Kesamaan suatu dokumen dengan *query* dapat diukur dengan vektor dokumen dan vektor *query* [16]. Gambar 1 merupakan representasi dokumen dan *query* pada ruang vektor [17]*:*



Gambar 1. Representasi dokumen dan *query* pada ruang vektor [17]

Perhitungan kemiripan antara vektor dokumen dan vektor *query* dilihat dari sudut yang paling kecil. Sudut yang dibentuk oleh dua buah vektor dapat dihitung dengan melakukan *inner product* [13]. Kemiripan antara vektor dokumen dan vektor *query* akan dihitung dengan pendekatan *cosine similarity*. Perumusan *cosine similarity* adalah sebagai berikut [18]:

$$\boldsymbol{Sim\ (D, D_i)\ = \cos\theta}$$

$$= \frac{\boldsymbol{D \cdot D_i}}{\boldsymbol{|D||D_i|}} = \frac{\boldsymbol{\sum W_{q,j}\ W_{i,j}}}{\boldsymbol{\sqrt{\sum W_{q,j}^2}\sqrt{\sum W_{i,j}^2}}} \quad (6)$$

Dimana*:*

D = dokumen acuan
D_i = dokumen ke-i
$W_{q,i}$ = bobot *term* j pada dokumen acuan
$W_{i,i}$ = bobot *term* j pada dokumen i

3. HASIL DAN PEMBAHASAN

3.1. Implementasi

Data keluhan pelanggan UPT PUSKOM UNS didapat mulai dari bulan Februari 2013 sampai September 2014, dengan jumlah seluruh data yang digunakan yaitu 190 data. Pembagian data *training* dan data *testing* ditunjukkan oleh Tabel 2.

Tabel 2. Pembagian data *training* dan data *testing*

Percobaan ke-	**Data *Training***	**Data *Testing***
1	114 data	19 data
2	133 data	19 data
3	152 data	19 data
4	171 data	19 data

Data yang digunakan sebagai contoh adalah keluhan dengan ID 172-180. Keluhan ID 172-179 digunakan sebagai data *training* dan keluhan ID 180 digunakan sebagai data *testing*. Hasil dari proses *text mining* yaitu *text preprocessing & text transforming* untuk keluhan ID 180 dapat dilihat pada Tabel 3.

Tabel 3. Hasil *Text Preprocessing & Text Transforming* untuk keluhan ID 180

ID	Keluhan	Hasil *Text Preprocessing*	Hasil *Text Transforming*
180	tidak bisa *login* siakad karena *username* dan *password* tidak tepat	tidak // bisa // *login* // siakad // karena // *username* // dan // *password* // tidak // tepat	~~tidak~~ // ~~bisa~~ // *login* // siakad // ~~karena~~ // *username* // ~~dan~~ // *password* // ~~tidak~~ // ~~tepat~~

Keterangan: tanda // digunakan sebagai batas antar term

Langkah berikutnya yakni menghitung frekuensi kata atau *term* dari setiap dokumen atau menghitung TF. Pada kombinasi N.T yang dimaksud TF adalah frekuensi *term* itu sendiri, sedangkan pada kombinasi L.T dan L.N nilai TF didapatkan dengan menggunakan persamaan 4 dan 5. Tabel 4 menunjukkan perhitungan TF menggunakan kombinasi N.T, L.T dan L.N*:*

Tabel 4. Perhitungan TF pada keluhan ID 180 menggunakan kombinasi N.T, L.T dan L.N

ID	*Term*	Nilai TF Setiap kombinasi		
		N.T	**L.T**	**L.N**
180	*login*	1	1	1
	siakad	1	1	1
	username	1	1	1
	password	1	1	1

Proses selanjutnya dari *pattern discovery* menghitung frekuensi *term* dari seluruh dokumen. Tabel 5 menunjukkan perhitungan frekuensi beberapa kata atau *term* dari seluruh dokumen.

Tabel 5. Perhitungan frekuensi kata dari seluruh dokumen

ID *Term*	*Term*	DF	ID *Term*	*Term*	DF	ID *Term*	*Term*	DF
1	*Add*	3	11	Hasil	2	21	*password*	1
2	Aktif	1	12	Ipk	1	22	*Pin*	1
3	Ambil	1	13	Juli	1	23	registrasi	3
4	Arah	1	14	jurusan	1	24	salah	1
5	Cek	1	15	Kuliah	3	25	siakad	6
6	Cetak	1	16	Kuota	1	26	Sks	1
7	*Edit*	1	17	*Login*	5	27	status	1
8	Email	1	18	Nama	1	28	*student*	1
9	*Error*	1	19	Nilai	1	29	transkrip	1
10	Gagal	1	20	*Online*	2	30	*username*	1

Langkah selanjutnya adalah membuat model ruang vektor (VSM) dari seluruh *term* dan seluruh dokumen. Model ruang vektor pada kombinasi L.T dan L.N adalah sama karena perhitungan TF pada kedua kombinasi tersebut menggunakan persamaan yang sama. Tabel 6 menunjukkan model ruang vektor pada kombinasi N.T, L.T dan L.N.

Tabel 6. Model ruang vektor pada kombinasi N.T, L.T & L.N

ID	DF	ID 172		ID 173		ID 174		ID 175		ID 176	
term		NT	LT/LN	NT	LT/LN	NT	LT/LN	NT	LT/LN	NT	LT/LN
1	3	1	1	0	0	0	0	0	0	1	1
2	1	0	0	1	1	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	0
4	1	0	0	1	1	0	0	0	0	0	0
5	1	0	0	1	1	0	0	0	0	0	0
6	1	0	0	0	0	0	0	2	1.3	0	0
7	1	0	0	0	0	0	0	0	0	0	0
8	1	0	0	0	0	1	1	0	0	0	0
9	1	0	0	0	0	1	1	0	0	0	0
10	1	1	1	0	0	0	0	0	0	0	0
11	2	2	1.3	0	0	0	0	0	0	0	0
12	1	0	0	0	0	0	0	0	0	0	0
13	1	0	0	1	1	0	0	0	0	0	0
14	1	0	0	0	0	0	0	1	1	0	0
15	3	1	1	0	0	0	0	0	0	2	1.3
16	1	0	0	0	0	0	0	0	0	1	1
17	5	1	1	1	1	1	1	0	0	0	0
18	1	0	0	0	0	0	0	0	0	0	0
19	1	0	0	0	0	0	0	3	1.48	0	0
20	2	1	1	1	1	0	0	0	0	0	0
21	1	0	0	0	0	0	0	0	0	0	0
22	1	0	0	0	0	0	0	0	0	0	0
23	3	1	1	2	1.3	0	0	0	0	0	0
24	1	0	0	0	0	0	0	0	0	0	0
25	6	1	1	1	1	0	0	1	1	0	0
26	1	0	0	0	0	0	0	0	0	0	0
27	1	0	0	1	1	0	0	0	0	0	0
28	1	0	0	0	0	1	1	0	0	0	0
29	1	0	0	0	0	0	0	2	1.3	0	0
30	1	0	0	0	0	0	0	0	0	1	1

Langkah selanjutnya adalah menghitung nilai IDF dari setiap *term*. Pada kombinasi N.T dan L.T nilai IDF dihitung dengan menggunakan persamaan T pada Tabel 1. Pada kombinasi L.N, nilai IDF tidak diperhitungkan sehingga IDF bernilai 1 berlaku untuk semua *term*. Tabel 7 menunjukkan hasil perhitungan nilai IDF pada masing-masing kombinasi.

Tabel 7. Hasil perhitungan nilai IDF

ID *Term*	DF	Nilai IDF			ID *Term*	DF	Nilai IDF		
		N.T	L.T	L.N			N.T	L.T	L.N
1	3	1.48	1.48	1	16	1	1.95	1.95	1
2	1	1.95	1.95	1	17	5	1.26	1.26	1
3	1	1.95	1.95	1	18	1	1.95	1.95	1
4	1	1.95	1.95	1	19	1	1.95	1.95	1
5	1	1.95	1.95	1	20	2	1.65	1.65	1
6	1	1.95	1.95	1	21	1	1.95	1.95	1
7	1	1.95	1.95	1	22	1	1.95	1.95	1
8	1	1.95	1.95	1	23	3	1.48	1.48	1
9	1	1.95	1.95	1	24	1	1.95	1.95	1

10	1	1.95	1.95	1	25	6	1.18	1.18	1
11	2	1.95	1.95	1	26	1	1.95	1.95	1
12	1	1.95	1.95	1	27	1	1.95	1.95	1
13	1	1.95	1.95	1	28	1	1.95	1.95	1
14	1	1.95	1.95	1	29	1	1.95	1.95	1
15	3	1.47	1.47	1	30	1	1.95	1.95	1

Langkah selanjutnya adalah menghitung bobot dari setiap *term*, yaitu mengalikan nilai TF dengan IDF. Tabel 8 menunjukkan hasil perhitungan bobot dari setiap *term* pada kombinasi N.T.

Tabel 8. Hasil perhitungan bobot setiap *Term* (TF*IDF) pada kombinasi N.T, L.T & L.N

ID	**ID 172**			**ID 173**			**ID 174**			**ID 175**			**ID 176**		
term	**NT**	**LT**	**LN**	**NT**	**LT**	**LN**	**NT**	**LT**	**LN**	**NT**	**LT**	**LN**	**NT**	**LT**	**LN**
1	1.48	1.48	1	0	0	0	0	0	0	0	0	0	1.48	1.48	1
2	0	0	0	1.95	1.95	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1.95	1.95	1	0	0	0	0	0	0	0	0	0
5	0	0	0	1.95	1.95	1	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	3.91	2.54	1.3	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	1.95	1.95	1	0	0	0	0	0	0
9	0	0	0	0	0	0	1.95	1.95	1	0	0	0	0	0	0
10	1.95	1.95	1	0	0	0	0	0	0	0	0	0	0	0	0
11	3.91	2.54	1.3	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	1.95	1.95	1	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	1.95	1.95	1	0	0	0
15	1.48	1.48	1	0	0	0	0	0	0	0	0	0	2.95	1.92	1.3
16	0	0	0	0	0	0	0	0	0	0	0	0	1.95	1.95	1
17	1.26	1.26	1	1.26	1.26	1	1.26	1.26	1	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	5.86	2.89	1.48	0	0	0
20	1.65	1.65	1	1.65	1.65	1	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	1.48	1.48	1	2.95	1.92	1.3	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	1.18	1.18	1	1.17	1.18	1	0	0	0	1.18	1.18	1	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	1.95	1.95	1	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	1.95	1.95	1	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	3.91	2.54	1.3	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	1.95	1.95	1

Langkah selanjutnya adalah menghitung kemiripan atau *similarity* masing-masing dokumen atau keluhan ID 172-179 terhadap keluhan ID 180. Tabel 9 menunjukkan hasil perhitungan kemiripan antara keluhan ID 172-179 dengan keluhan ID 180 pada ketiga kombinasi.

Tabel 9. Hasil perhitungan kemiripan antara keluhan ID 172-179 dengan keluhan ID 180

ID	Nilai Kemiripan dengan ID 180		
	Kombinasi N.T	**Kombinas L.T**	**Kombinasi L.N**
172	0.1624	0.1915	0.4123
173	0.1570	0.1703	0.3905
174	0.1341	0.1341	0.3138
175	0.0507	0.0826	0.2138
176	0	0	0
177	0	0	0
178	0.1415	0.1415	0.3395
179	0.2792	0.2792	0.6078

Berdasarkan Tabel 9, nilai kemiripan yang paling besar adalah antara keluhan ID 180 dengan 179, baik pada kombinasi N.T, L.T maupun L.N. Hal tersebut berarti bahwa diantara keluhan ID 172 sampai 179, yang paling mirip dengan keluhan ID 180 adalah 179.

3.2. *Analisa Hasil*

1) Pengujian Akurasi

Pengujian dilakukan 4 kali dengan jumlah data *training* yang berbeda-beda. Pada masing-masing percobaan dilakukan 3 kali proses dengan 3 kombinasi notasi pembobotan TF-IDF yang berbeda, yaitu kombinasi N.T, kombinasi L.T dan kombinasi L.N. Selain itu, *threshold* yang digunakan pada masing-masing percobaan yaitu 0.5 dan 0.65. Sehingga keluhan yang dianggap mirip oleh sistem adalah keluhan yang memiliki kemiripan diatas 0.5 dan 0.65.

Dari nilai TP, FP, FN dan TN dapat dihitung nilai *accuracy*, *precision* dan *recall*-nya. Tabel 10 merupakan hasil perhitungan nilai *accuracy*, *precision* dan *recall* untuk *threshold* 0.5.

Tabel 10. Hasil perhitungan nilai *accuracy*, *precision* dan *recall* untuk *threshold* 0.5 & 0.65

Kombinasi Notasi TF-IDF	Percobaan ke-	*Accuracy* (%)		*Precison* (%)		*Recall* (%)	
		T:0.5	***T:0.65***	***T:0.5***	***T:0.65***	***T:0.5***	***T:0.65***
N.T	1	45.71	80.77	32.14	61.54	100	100
	2	67.93	86.84	64.58	83.87	100	100
	3	68.75	79.41	64.29	70.83	100	100
	4	52	85.71	46.67	57.14	100	100
L.T	1	45.71	76.92	32.14	53.85	100	100
	2	69.81	86.49	65.96	83.33	100	100
	3	70.83	82.86	66.67	76	100	100
	4	62	80.95	55.81	50	100	100
L.N	1	30.36	56.25	26.41	36.36	100	100
	2	51.19	74.47	50	70	100	100
	3	60.32	81.58	56.9	75	100	100
	4	40.81	76.47	38.95	68	100	100

Berdasarkan Tabel 10 mengenai hasil perhitungan *accuracy*, *precision* dan *recall* untuk *threshold* 0.5, dapat diketahui bahwa *accuracy* yang paling rendah adalah pada

percobaan pertama untuk ketiga kombinasi notasi TF-IDF, yaitu masing-masing 45.71%, 45.71% dan 30.36%. Hal ini disebabkan jumlah keluhan yang mirip dan relevan atau nilai *true positive* jauh lebih sedikit bandingkan dengan keluhan yang mirip namun tidak relevan atau nilai *false positive*.

Adapun hasil perhitungan *accuracy*, *precision* dan *recall* untuk *threshold* 0.65 dapat diketahui bahwa kombinasi notasi TF-IDF N.T memiliki *accuracy* sebesar 80.77% pada percobaan ke-1 dan mengalami peningkatan pada percobaan ke-2 yaitu sebesar 86.84%. Kemudian mengalami penurunan pada percobaan ke-3 yaitu sebesar 79.41% dan kembali mengalami peningkatan pada percobaan ke-4 yaitu sebesar 57.14%.

Pola peningkatan dan penurunan *accuracy* tersebut juga berlaku untuk kombinasi notasi TF-IDF L.T, namun tidak berlaku untuk kombinasi notasi TF-IDF L.N. Kombinasi notasi TF-IDF L.N memiliki *accuracy* sebesar 36.36% pada percobaan ke-1 dan mengalami peningkatan pada percobaan ke-2 dan ke-3, yaitu memiliki *accuracy* sebesar 74.47% dan 81.58%. Kemudian mengalami penurunan pada percobaan ke-4 yaitu dengan *accuracy* sebesar 76.47%.

2) Membandingkan Kombinasi Notasi TF-IDF

Pada Tabel 11 ditunjukkan bahwa nilai rata-rata *accuracy*, *precision* dan *recall* pada masing-masing kombinasi notasi TF-IDF.

Tabel 11. Nilai rata-rata *accuracy*, *precision* dan *recall* ketiga kombinasi notasi TF-IDF

Kombinasi Notasi TF-IDF	*Accuracy* (%)		*Precision* (%)	
	0.5	**0.65**	**0.5**	**0.65**
N.T	58.6	83.18	51.92	68.35
L.T	62.09	81.81	55.15	65.8
L.N	45.67	72.19	43.07	62.34

Berdasarkan Tabel 11, dilihat dari nilai rata-rata *accuracy* dan *precision*-nya dapat diketahui bahwa pada *threshold* 0.5 kombinasi notasi pembobotan TF-IDF yang terbaik adalah kombinasi L.T yaitu sebesar 62.09% untuk *accuracy* dan 55.15% untuk *precision*. Sedangkan pada *threshold* 0.65 kombinasi notasi pembobotan TF-IDF yang terbaik adalah kombinasi N.T yaitu sebesar 83.18% untuk *accuracy* dan 68.35% untuk *precision*.

3) Pembangkitan FAQ

Untuk membangkitkan FAQ dilakukan percobaan dengan menggunakan data sebanyak 171 data. Selain itu kombinasi notasi pembobotan TF-IDF yang digunakan adalah kombinasi N.T serta menggunakan *threshold* 0.65.

Dari percobaan tersebut dapat dibangkitkan 27 FAQ. 19 diantaranya relevan dan 8 lainnya tidak relevan. Yang dimaksud relevan dalam hal ini adalah FAQ tersebut dibentuk oleh 3 atau lebih keluhan yang betul-betul mirip. Sedangkan tidak relevan adalah FAQ tersebut dibentuk oleh 3 atau lebih keluhan yang sebetulnya tidak mirip.

4. SIMPULAN

Algoritma VSM dengan pembobotan TF-IDF dapat digunakan untuk membangkitkan FAQ otomatis dan solusi yang relevan. Berdasarkan hasil perhitungan *accuracy* pada masing-masing percobaan dapat disimpulkan pada *threshold* 0.5, kombinasi notasi TF-IDF yang memiliki nilai rata-rata *accuracy* dan *precision* tertinggi adalah modifikasi pertama, yaitu masing-masing sebesar 62.09% dan 55.15%. Sedangkan untuk *threshold* 0.65 kombinasi notasi TF-IDF yang memiliki nilai rata-rata *accuracy* dan *precision* tertinggi adalah TF-IDF, yaitu masing-masing sebesar 83.18% dan 68.35%. Sehingga dapat dikatakan bahwa kombinasi yang paling tepat untuk digunakan di UPT PUSKOM UNS adalah kombinasi N.T dengan *threshold* 0.65. Selain itu percobaan yang dilakukan dengan menggunakan data sebanyak 171 data, TF-IDF dan *threshold* 0.65 dapat membangkitkan 27 FAQ, yaitu dengan persentase 70.37% relevan.

5. REFERENSI

[1] Metehan, T., Yasemin, Z. A. 2011. Demographic Characteristics and Complaint Behavior: An Empirical Study Concerning Turkish Customers. *International Journal of Business and Social Science*, Vol. 2, No. 9.

[2] Andreassen, T.W. 1999. What Drives Customer Loyalty with Complaint Resolution?. *Journal of Service Research*, Vol. 1, No. 4.

[3] Anonim. 2014. *Profil UPT Puskom*. [Online] Available at: http://puskom.uns.ac.id/profil-upt-puskom/. Diakses pada tanggal 11 Agustus 2015.

[4] Junedy, R.. 2014. Perancangan Aplikasi Deteksi Kemiripan Isi Dokumen Teks dengan Menggunakan Metode Levenshtein Distance. *Pelita Informatika Budi Darma*, Vol. 7, No. 2.

[5] Wicaksono, D. W., Irawan, M. I., Rukmi, A. M.. 2014. Sistem Deteksi Kemiripan Antar Dokumen Teks Menggunakan Model Bayesian Pada *Term* Latent Semantic Analysis (LSA). *Jurnal Sains dan Seni POMITS*, Vol. 3, No. 2.

[6] Heriyanto. 2011. Penggunaan Metode Exact Match untuk Menentukan Kemiripan Naskah Dokumen Teks. *Telematika*, Vol. 8, No. 1, 43-52.

[7] Isa, T. M., Abidin, T. F.. 2013. Mengukur Kesamaan Paragraf Menggunakan Vector Space Model untuk Mendeteksi Plagiarisme. *Seminar Nasional dan Expo Teknik Elektro.*

[8] Amin, F.. 2011. Implementasi Search Engine (Mesin Pencari) Menggunakan Metode Vector Space Model. *Dinamika Teknik*, Vol. 5, No. 1, 45-48.

[9] Asshidiq, A., Saptono, R., Sulistyo, M. E.. 2013. *Penilaian Ujian Bertipe Essay Menggunakan Metode Text Similarity. Informatika*. Universitas Sebelas Maret. Surakarta.

[10] Grossmann, D., Fieder, O.. 2004. *Information Retrieval: Algorithm and Heuristic, Second Edition*. Springer, Dordrecth, The Netherlands.

[11] Agusta, L.. 2009. Perbandingan Algoritma Stemming Porter dengan Algoritma Nazief & Adriani untuk Stemming Dokumen Teks Bahasa Indonesia. *Konferensi Nasional Sistem dan Informatika*, 196-201.

[12] Robertson, S.. 2005. Understanding Inverse Document Frequency*:* On Theoretical Arguments for IDF. *England: Journal of Documentation*, Vol. 60, 502-520.
[13] Karmayasa, O., Mahendra, I. B.. 2012. Implementasi Vector Space Model dan Beberapa Notasi Metode *Term* Frequency Inverse Document Frequency (TF-IDF) pada Sistem Temu Kembali Informasi. *Jurnal Elektronik Ilmu Komputer Universitas Udayana*, Vol. 1, No. 1
[14] Xia, T., Chai, Y.. 2011. An Improvement to TF-IDF: *Term* Distribution based *Term* Weight Algorithm. *Journal of Software*, Vol. 6, No. 3.
[15] Robertson, S.. 2005. Understanding Inverse Document Frequency*:* On Theoretical Arguments for IDF. *England: Journal of Documentation*, Vol. 60, 502-520.
[16] Usharani, J., Iyakutti, K.. 2013. A Genetic Algorithm Based on Cosine Similrity for Relevant Document Retrieval. *International Journal of Engineering Research & Technology (IJERT)*, Vol. 2, Issue 2.
[17] Mandala, R., Setiawan, H.. 2002. *Peningkatan Performansi Sistem Temu-Kembali Informasi dengan Perluasan Query Secara Otomatis*. Departemen Teknik Informatika. Institut Teknologi Bandung. Bandung.
[18] Muhajir, R. B.. 2012. *Metode Similarity-Mashup untuk Framework Modul Relevant Content pada Content Management System* (CMS). Informatika. Universitas Sebelas Maret. Surakarta.