PROPOSAL PENELITIAN

Ekstraksi Fitur Statistik untuk Deteksi Derau pada Dokumen Spesifikasi Kebutuhan Perangkat Lunak

Ahmad Mustofa
NRP. 5116201054

DOSEN PEMBIMBING

Daniel Oranova Siahaan, S.Kom, M.Sc, P.D.Eng

NIP. 197411232006041001

PROGRAM MAGISTER

DEPARTEMEN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI & KOMUNIKASI

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN
PROPOSAL TESIS

Judul	: Ekstraksi fitur statistik untuk deteksi noise pada dokumen Spesifikasi Kebutuhan Perangkat Lunak
Oleh	: Ahmad Mustofa
NRP	: 5116201054

Telah diseminarkan pada:

Hari	:
Tanggal	:
Tempat	:

Mengetahui / menyetujui

Dosen Penguji:	Dosen Pembimbing:
1.	1.
_____	
2.	2.
_____	
3.	
_____	
NIP.	

[Halaman ini sengaja dikosongkan]

Nama Mahasiswa : Ahmad Mustofa
NRP : 5116201054
Pembimbing : Daniel Oranovora Siahaan, S.Kom. PD.Eng

ABSTRAK

Tahap spesifikasi kebutuhan adalah tahap pertama yang dilakukan dalam proses pengembangan perangkat lunak. Sehingga jika terjadi kesalahan pada tahap ini, secara otomatis akan terjadi kesalahan pada tahap-tahap selanjutnya. Kesalahan dalam pernyataan kebutuhan perangkat lunak diantaranya berupa *noise* (derau), ambigu, konflik, serta inkonsistensi. Beberapa penelitian sebelumnya telah berhasil mendeteksi ambigu, konflik, dan inkonsistensi dalam pernyataan kebutuhan secara otomatis. Akan tetapi pada saat proposal ini ditulis, belum ada penelitian yang dilakukan untuk mendeteksi derau dalam pernyataan kebutuhan perangkat lunak secara otomatis.

Penelitian ini mengajukan suatu metode untuk deteksi derau dalam pernyataan kebutuhan perangkat lunak secara otomatis. Metode yang diajukan berfokus pada bagaimana melakukan ekstraksi fitur lokal dari masing-masing pernyataan kebutuhan dalam sebuah dokumen Spesifikasi Kebutuhan Perangkat Lunak (SKPL) sehingga kemudian bisa digunakan secara bersamaan dengan pernyataan kebutuhan dari dokumen SKPL yang lain dalam membangun model klasifikasi. Ekstraksi fitur lokal dari sebuah pernyataan kebutuhan dilakukan dengan memanfaatkan fitur statistik dari kemiripan pernyataan kebutuhan tersebut dengan pernyataan kebutuhan yang lain dalam dokumen SKPL yang sama.

Metode yang diajukan akan diuji dengan menggunakan data pernyataan kebutuhan yang telah dilabeli secara manual yang kemudian akan diukur performanya dengan menggunakan metode *k-fold cross validation*.

Kata kunci*:* pernyataan kebutuhan, deteksi derau, fitur statistik

[Halaman ini sengaja dikosongkan]

Nama Mahasiswa : Ahmad Mustofa
NRP : 5116201054
Pembimbing : Daniel Oranovora Siahaan, S.Kom. PD.Eng

ABSTRACT

Requirement specification is the first step of a software development cycle. If errors occured in this step, errors will automatically occur in the next step. Errors in software requirements consist of noise, ambiguous, conflict, and inconsistency. Some research has been successfully detecting ambiguous, conflict, and inconsistency in software requirements automatically. But when this document is written, there was no research done to detect noise in software requirements automatically.

This research proposes a method to detect noise in software requirements automatically. Proposed method focuses on how to extract local features of a requirement statement in a Software Requirement Specification (SRS) document so that this feature can be used globally with other requirement statements from another SRS document to build a classification model. Local feature extraction of a requirement statement is done by using statistical feature of the requirement statement's similarities with other requirement statement in the same SRS document.

Proposed method will be validated by using requirement statements data that have been labeled manually and it's performance will be measured later by using k-fold cross validation method.

***Keywords**:* requirements, noise detection, statistical feature

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Tahap spesifikasi kebutuhan adalah tahap pertama yang dilakukan dalam proses pengembangan perangkat lunak. Sehingga jika terjadi kesalahan pada tahap ini, secara otomatis akan terjadi kesalahan pada tahap-tahap selanjutnya. Bertrand meyer mengelompokkan kesalahan dalam spesifikasi kebutuhan menjadi tujuh kelompok yang kemudian dikenal dengan istilah *Meyer's seven sins* [1]. Beberapa peneliti mengajukan metode untuk mendeteksi ambiguitas (salah satu kesalahan dalam *Meyer's seven* sins) dalam pernyataan kebutuhan perangkat lunak [2]. Noise (derau) adalah salah satu kesalahan dalam *Meyer's seven sins* yang disebabkan oleh adanya suatu elemen dalam teks yang memberikan informasi yang tidak relevan dengan domain masalah yang hendak diselesaikan [1]. Meyer membagi derau menjadi dua jenis, yaitu *remorse* dan *redundancy*. *Remorse* dapat didefinisikan sebagai pernyataan kebutuhan yang memberikan informasi yang tidak relevan dengan domain masalah, sedangkan *redundancy* adalah pernyataan kebutuhan yang memberikan informasi yang sama dengan pernyataan kebutuhan yang lain.

Di sisi lain, perkembangan yang pesat pada bidang *machine learning* telah mampu melakukan klasifikasi teks secara otomatis. Hal yang paling penting dalam pengolahan teks terletak pada tahap pembobotan. Hao Xu dan Bo Yu menggunakan metode pembobotan term frequency – invers document frequency (tf-idf) untuk melakukan klasifikasi email spam [3]. Bruno Trstenjak, Sasa Mikac, dan Dzenana Donko juga menggunakan tf-idf sebagai metode pembobotan untuk melakukan pengelompokan dokumen [4].

Derau dalam dokumen Spesifikasi Kebutuhan Perangkat Lunak (SKPL) hanya dapat dikenali secara lokal. Pernyataan kebutuhan yang tergolong derau dalam sebuah dokumen SKPL akan memiliki kriteria yang berbeda dengan peryataan kebutuhan yang tergolong derau di dokumen SKPL yang lain. Hal ini bertolak belakang dengan metode tf-idf yang melakukan pembobotan teks secara global. Karenanya dibutuhkan sebuah

metode untuk ekstraksi fitur lokal dari masing-masing pernyataan kebutuhan dalam sebuah dokumen SKPL.

Di sisi lain, B. Chandra, dan Manish Gupta menggunakan pendekatan statisitik untuk seleksi fitur yang akan digunakan dalam klasifikasi gen [5]. Wujie Zhou, Lu Yu, Weiwei Qiu, Yang Zhou, dan Mingwei Wu juga menggunakan nilai statistik untuk mendapatkan fitur lokal dari sebuah citra yang kemudian digunakan untuk menentukan kualitas dari citra tersebut [6]. Pendekatan secara statistik inilah yang kemudian akan digunakan oleh penulis untuk ekstraksi fitur lokal dari masing-masing pernyataan kebutuhan dalam sebuah dokumen SKPL. Setelah didapatkan fitur dari masing-masing pernyataan kebutuhan, deteksi pernyataan kebutuhan yang termasuk derau dalam sebuah dokumen SKPL dapat dicapai dengan melakukan klasifikasi masing-masing pernyataan kebutuhan dengan memanfaatkan fitur-fitur yang berhasil didapat.

1.2 Perumusan Masalah

Rumusan masalah yang diangkat dalam penelitian ini adalah:

1. Fitur apa saja yang tepat untuk klasifikasi derau dalam pernyataan kebutuhan perangkat lunak
2. Bagaimana mengekstraksi fitur-fitur dari pernyataan kebutuhan dalam dokumen spesifikasi kebutuhan perangkat lunak

1.3 Tujuan

Tujuan yang akan dicapai dalam pembuatan tesis ini adalah membangun sebuah metode yang dapat mendeteksi pernyataan kebutuhan yang termasuk derau dalam sebuah dokumen spesifikasi kebutuhan perangkat lunak dengan akurat.

1.4 Manfaat

Manfaat dari penelitian ini adalah untuk membantu perekayasa kebutuhan meningkatkan kualitas dokumen spesifikasi kebutuhan perangkat lunak yang akan dibuat dengan mengurangi pernyataan kebutuhan yang termasuk derau.

1.5 Kontribusi Penelitian

Kontribusi yang diharapkan dari penelitian ini adalah mengembangkan sebuah metode ekstraksi fitur untuk klasifikasi pernyataan kebutuhan yang termasuk derau dalam dokumen spesifikasi kebutuhan perangkat lunak.

1.6 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

1. Dokumen spesifikasi kebutuhan perangkat lunak yang digunakan hanya menggunakan Bahasa Inggris.
2. Pernyataan kebutuhan diekstrak secara manual dari dokumen spesifikasi kebutuhan perangkat lunak

[Halaman ini sengaja dikosongkan]

BAB 2
KAJIAN PUSTAKA

2.1 TF-IDF

Term frequency – invers document frequency adalah metode pembobotan dalam pengolahan teks. Nilai term frequency (tf) didapat dengan menghitung jumlah kemunculan sebuah term dalam sebuah dokumen. Sedangkan nilai invers document frequency didapat dengan rumus

$$\mathrm{idf_i} = \log_2 \frac{N}{\mathrm{df_i}} \tag{2.1}$$

dimana $\mathrm{df_i}$ adalah jumlah dokumen yang mengandung term i. Nilai tf-idf dari term i kemudian dapat dihitung dengan mengalikan nilai tf dari term i dengan nilai idf dari term i [4].

2.2 Cosine Similarity

Dalam *text processing*, sebuah dokumen biasanya direpresentasikan sebagai sebuah vektor. Cosine similarity adalah sebuah metode yang digunakan untuk menghitung kemiripan antara dua vektor. Nilai cosine similarity dari vektor A dan vektor B didapat dengan menggunakan rumus

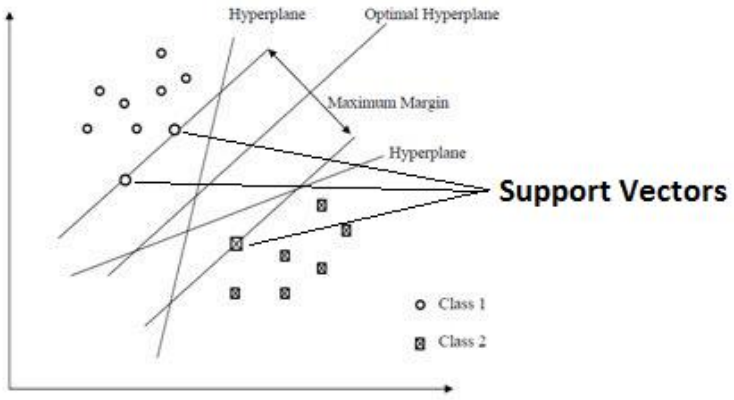
$$\mathrm{sim}(A,\ B) = \frac{\sum_{k=1}^{N} \mathrm{A_k} \ * \ \mathrm{B_k}}{\sqrt{\sum_{k=1}^{N} \mathrm{A_k}^{\,2}} \ * \sqrt{\sum_{k=1}^{N} \mathrm{B_k}^{\,2}}} \tag{2.2}$$

dimana $\mathrm{A_k}$ dan $\mathrm{B_k}$ adalah komponen dari vektor A dan B [7].

2.3 Support Vector Machine

Support vector machine (SVM) adalah metode klasifikasi *supervised* (data latih sudah diketaui kelasnya) yang mengklasifikasikan dua kelas. Pada metode ini, setiap data akan di-*plotting* ke dalam ruang *n*-dimensi (dimana *n* adalah jumlah fitur) dengan nilai masing-masing fitur menjadi nilai tertentu pada koordinat. Kemudian akan dilakukan klasifikasi dengan mencari *hyperplane* (bidang pembatas) untuk membedakan antara dua kelas sebaik mungkin. Gambar 2.1 merupakan beberapa contoh dari kemungkinan *hyperplane* yang digunakan untuk memisahkan kelas satu dengan lainnya. Metode SVM

akan mencari h*yperplane* yang memiliki *margin* yang maksimal. *Support vector* adalah data-data yang dijadikan pembatas dengan kelas lain.



Sumber gambar : www.sine.ni.com

Gambar 2.1 Contoh alternatif hyperplane

Diberikan data masukan $(x_{1,}x_{2,}x_3,\ldots,x_n)$ dan masing-masing kelas dinotasikan $y_i \in \{-1,+1\}$ untuk $i = 1,2,3,\ldots,n$ dimana *n* adalah banyaknya data. Fungsi *hyperplane* dibuat dengan persamaan

$$w\,.x + b = 0 \tag{2.3}$$

dengan batasan yang ditulis dalam persamaan

$$y_i(wx_i + b) + \ t_i \ \geq 1 \tag{2.4}$$

untuk mencari nilai *w* dan *b* yang optimal digunakan persamaan

$$\min \frac{1}{2}\ ||w||^2 + C\sum_{i=1}^{l} t_i \tag{2.5}$$

dimana nilai *C* adalah nilai pinalti dari kesalahan klasifikasi. Fungsi tujuan persamaan di atas berbentuk kuadrat, sehingga untuk menyelesaikannya, bentuk tersebut ditransformasikan ke dalam bentuk *dual space*. Persamaan *dual space* dapat ditulis menggunakan persamaan

$$\max \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j x_i x_j \tag{2.6}$$

dengan batasan dalam persamaan berikut.

$$\alpha_i \geq 0 , \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{2.7}$$

Untuk mencari nilai α_i, digunakan *sequential minimal optimization* (SMO). Persamaan *hyperplane* dilakukan dengan persamaan:

$$f = w^T z + b = \sum_{i=1}^{s} \alpha_i y_i \, {x_i}^T z + b \tag{2.8}$$

dimana *z* adalah data masukan *support vector machine*. Pada banyak kasus, data yang diklasifikasikan tidak bisa langsung dipisahkan dengan garis yang linear. Oleh karena itu, digunakan metode kernel untuk mengatasi permasalahan tersebut. Dengan metode kernel, suatu data x di *input space* dipetakan ke fitur *space* F dengan dimensi yang lebih tinggi. Salah satu kernel yang biasa dipakai adalah kernel RBF dan *Polynomial*. Persamaan kernel RBF dan *Polynomial* berurutan dapat dilihat pada persamaan di bawah ini:

$$k(x, y) = \exp(-\gamma |x - y|^2) \tag{2.9}$$

$$k(x, y) = (\gamma \langle x^T y \rangle + r)^p \tag{2.10}$$

Penggunaan fungsi kernel mengubah persamaan *dual space* menjadi

$$\max \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j k(x_i, x_j) \tag{2.11}$$

dan juga mengubah persamaan *hyperplane* menjadi

$$f = \sum_{i=1}^{s} \alpha_i y_i \, k(x_i, z) + b \qquad (2.12)$$

[8].

2.4 Synthetic Minority Over-sampling Technique (SMOTE)

Untuk mengatasi adanya *imbalanced* pada data latih, SMOTE dapat digunakan untuk melakukan sintesa data sehingga data yang digunakan untuk melatih *classifier* dapat menjadi seimbang. SMOTE merupakan sebuah algoritma *oversampling* (menambahkan data baru dengan kelas minoritas) yang memanfaatkan kemiripan dari masing-masing data [9]. Berikut adalah pseudocode dari algoritma SMOTE

```
1  SMOTE(T, N, k)
2  #   Input: Number of minority class samples T; Amount of SMOTE N%;
       Number of nearest neighbors k
3  #   Output: (N/100)* T synthetic minority class samples
4      if N <100
5         then Randomize the T minority class samples
6         T = (N/100) ∗ T
7         N = 100
8      endif
9      N = (int)(N/100)
10     k = Number of nearest neighbors
11     numattrs = Number of attributes
12     Sample[ ][ ]: array for original minority class samples
13     newindex: 0
14     Synthetic[ ][ ]: array for synthetic samples
15     for i ← 1 to T
16           Compute k nearest neighbors for i, and save the indices in the nnarray
17           Populate(N, i, nnarray)
18     endfor
19 return
20 Populate(N, i, nnarray)
21     while N != 0
22           for attr ← 1 to numattrs
23                 Compute: dif = Sample[nnarray[nn]][attr] − Sample[i][attr]
24                 Compute: gap = random number between 0 and 1
25                 Synthetic[newindex][attr] = Sample[i][attr] + gap ∗ dif
26           endfor
```

27	newindex++
28	N = N − 1
29	endwhile
30	**return**

2.5 Presisi dan Recall

Untuk menghitung tingkat performansi suatu sistem dapat digunakan perhitungan presisi dan *recall.* Secara matematis, rumus untuk menghitung presisi dan recall dapat dilihat pada persamaan berikut

$$precision = \frac{TP}{TP + FP} \tag{2.13}$$

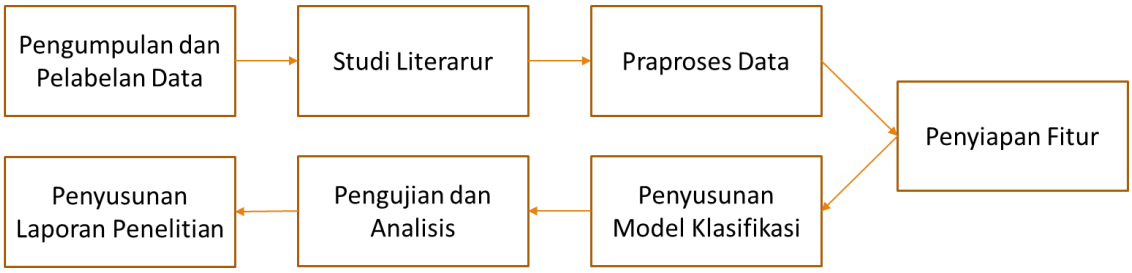
$$recall = \frac{TP}{TP + FN} \tag{2.14}$$

dimana TP adalah kebutuhan yang tergolong derau dan terklasifikasi sebagai derau, FP adalah kebutuhan yang bukan tergolong derau akan tetapi terklasifikasi sebagai derau, serta FN adalah kebutuhan yang tergolong derau akan tetapi tidak terklasifikasi sebagai derau.

[Halaman ini sengaja dikosongkan]

BAB 3
METODOLOGI PENELITIAN

Bab ini akan memaparkan tentang metodologi penelitian yang digunakan pada penelitian ini, yang terdiri dari (1) pengumpulan dan pelabelan data, (2) studi literatur, (3) praproses data, (4) ekstraksi fitur, (5) penyusunan model klasifikasi, (6) pengujian dan analisis, dan (7) penyusunan laporan penelitian. Ilustrasi alur metodologi penelitian dapat dilihat pada berikut.



Gambar 3.1 Alur Penelitian

3.1 Pengumpulan dan Pelabelan Data

Penelitian diawali dengan proses pengumpulan data pernyataan kebutuhan yang kemudian dilakukan pelabelan secara manual untuk menentukan pernyataan kebutuhan yang tergolong derau atau tidak. Sebagai contoh pada sistem informasi penemuan barang hilang dan sisterm informasi ketersediaan dosen, F adalah singkatan dari Fungsionalitas seperti yang ditunjukkan pada tabel berikut.

Tabel 3.1 Pernyataan kebutuhan SI Penemuan Barang Hilang

Kode	Pernyataan Kebutuhan	Label
F01	Register a new user account	0
F02	Update user profile	0
F03	Report on finding thing	0
F04	Report on lost thing	0
F05	Asking help in form of question	1
F06	Delete user account	0
F07	Give reward	1

Tabel 3.2 Pernyataan kebutuhan SI Ketersediaan Dosen

Kode	Pernyataan kebutuhan	Label
F11	Register a new lecturer's account	0
F12	Delete lecturer's account	0
F13	Update lecturer's account information	0
F14	Add lecturer's availability status	0
F15	Update lecturer's availability status	0
F16	Delete lecturer's availability status	0
F17	Show lecturer's information and availability status	0
F18	Find lecturer's account	0
F19	Lecturer can approve student's plan	1

Dari total 405 data yang berhasil dikumpulkan dan dilabeli oleh 3 orang annotator, total persentase data derau hanya sekitar 10%. Hal ini menunjukkan bahwa persebaran data derau tidaklah seimbang. Karenanya dibutuhkan sebuah proses untuk melakukan penyeimbangan data. Proses penyeimbangan data dilakukan dengan menggunakan algoritma SMOTE.

Tabel 3.3 Persebaran derau dalam SKPL yang digunakan

Nama File	Jumlah Derau	Total	Persentase
1_libra_srs-uji1	1	13	7.6
24_srsv6-uji24	3	24	12.5
18_SoftwareRequirementsSpecification	9	86	10.46
7_16hlmnExemplu_cerinte_software	1	6	16.67
12_15hlmnSRS4.0_uji12	14	106	13.2
4.argos_urd-uji4	1	39	2.56
13_T1-Req-uji13	8	64	12.5
15_bpms-uji15	2	17	11.76
2_SRS_sample V1.2	1	17	5.88
14_yh-rr-ss-jw1-uji14	1	33	3.03

3.2 Studi Literatur

Proses kedua adalah pengkajian yang berkaitan dengan topik penelitian yang diambil. Pada penelitian ini, referensi yang digunakan adalah jurnal-jurnal yang berkaitan dengan metode-metode yang digunakan.

3.3 Praproses Data

Pada tahap ini akan dilakukan praproses pada data pernyataan kebutuhan yang telah berhasil dikumpulkan. Tahap ini terbagi menjadi 3 proses, yaitu *stopword removal*, *stemming*, dan tokenisasi.

3.3.1 Stopword Removal

Stopword removal adalah penghapusan kata penghubung dan tanda baca dari masing-masing data pernyataan kebutuhan. Sebagai contoh, sebuah pernyataan kebutuhan F17 “show lecturer’s information and availability status” akan menjadi “show lecturer information availability status” setelah melalui tahap ini.

3.3.2 Stemming

Stemming adalah proses pengubahan setiap kata dalam pernyataan kebutuhan menjadi kata dasar. Sebagai contoh, “show lecturer information availability status” akan menjadi “show lecture information available status” setelah melalui tahap ini.

3.3.3 Tokenisasi

Tokenisasi adalah proses pemecahan masing-masing pernyataan kebutuhan menjadi kumpulan kata yang unik. Sebagai contoh, “show information available lecture available student” akan menjadi {“show", “information”, “available”, “lecture”, “student”} setelah melalui tahap ini.

3.4 Penyiapan Fitur

Tahap ini terdiri dari dua proses utama, yaitu tahap pembobotan term dan tahap ekstraksi fitur.

3.4.1 Pembobotan Term

Masing-masing pernyataan kebutuhan yang telah melalui tahap praproses akan dihitung bobotnya dengan menggunakan rumus tf-idf. Setelah proses ini, masing-masing pernyataan kebutuhan akan direpresentasikan sebagai sebuah vektor.

Sebagai contoh, 2 pernyataan kebutuhan F17 “show lecturer’s information and availability status” dan F13 “update lecturer’s account information” akan menjadi {“show", “lecture”, “information”, “available”, “status”} dan {“update", “lecture”, “account”, “information”} setelah melalui tahap praproses. Token-token dari F17 dan F13 kemudian digabungkan menjadi {“show", “lecture”, “information”, “available”, “status”, “update”, “account”} untuk kemudian digunakan sebagai fitur dalam penghitungan bobot tf-idf.

Untuk mendapatkan bobot tf-idf, pertama yang dilakukan adalah menghitung *term frequency* dari masing-masing kata fitur dalam masing-masing pernyataan kebutuhan sebagai berikut.

Tabel 3.4 Contoh Hasil Penghitungan Term Frequency

Kode	Show	Lecture	Information	Available	Status	Update	Account
F17	1	1	1	1	1	0	0
F13	0	1	1	0	0	1	1

Setelah itu, dilakukan penghitungan idf dari masing-masing kata fitur sebagai berikut.

Tabel 3.5 Contoh Hasil Penghitungan Invers Document Frequency

Kode	Show	Lecture	Information	Available	Status	Update	Account
Idf	0.3	0	0	0.3	0.3	0.3	0.3

Tahap akhir adalah mengalikan nilai tf dari masing-masing pernyataan kebutuhan dengan nilai idf.

Tabel 3.6 Contoh Hasil Penghitungan TF-iDF

Kode	Show	Lecture	Information	Available	Status	Update	Account
F17	0.3	0	0	0.3	0.3	0	0
F13	0	0	0	0	0	0.3	0.3

3.4.2 Ekstraksi Fitur

Untuk masing-masing vektor pernyataan kebutuhan, akan dihitung nilai similaritasnya dengan vektor pernyataan kebutuhan yang lain dalam dokumen SKPL

yang sama dengan menggunakan *cosine similarity*. Setelah itu diambil tiga fitur yang kemudian akan digunakan dalam tahap klasifikasi. Tiga fitur tersebut adalah sebagai berikut.

a. Maximum of Similarity Measure

   Fitur ini adalah nilai maksimum dari nilai similaritas yang sudah didapatkan sebelumnya.

b. Mean of Similarity Measure

   Fitur ini adalah nilai rata-rata dari nilai similaritas yang sudah didapatkan sebelumnya.

c. Standard Deviation of Similarity Measure

   Fitur ini adalah nilai standar deviasi dari nilai similaritas yang sudah didapatkan sebelumnya.

Berikut adalah pseudocode dari proses ekstraksi fitur yang dilakukan untuk masing-masing pernyataan kebutuhan dalam sebuah dokumen SKPL.

```
1  ExtractFeature(DokumenSKPL)
2      feature_matrix = new matrix
3      foreach k = pernyataan_kebutuhan in DokumenSKPL
4          similarity = cosineSimilarity(k,
               rest_of_pernyataan_kebutuhan)
5          feature_matrix.add([mean(similarity), std(similarity),
               max(similarity)])
6      end foreach
7      return feature_matrix
```

Berikut adalah hasil ekstraksi fitur dari contoh kasus sistem informasi penemuan barang hilang dan sisterm informasi ketersediaan dosen.

Tabel 3.7 Hasil Ekstraksi Fitur pada Contoh Kasus

Kode	Mean	Standard Deviation	Maximum
F01	0.062	0.119	0.3
F02	0.03	0.047	0.1
F03	0.076	0.185	0.5
F04	0.076	0.185	0.5
F05	0	0	0
F06	0.066	0.119	0.3

Kode	Mean	Standard Deviation	Maximum
F07	0	0	0
F11	0.049394	0.075	0.197
F12	0.149962	0.219	0.642
F13	0.049394	0.075	0.197
F14	0.081858	0.117	0.281
F15	0.081858	0.117	0.281
F16	0.17665	0.228	0.642
F17	0.066294	0.093	0.21
F18	0.042397	0.063	0.164
F19	0	0	0

3.4.3 Penyeimbangan Data Menggunakan SMOTE

Setelah proses ekstraksi fitur dari semua data berhasil dilakukan, akan dilakukan penyeimbangan data derau dengan menggunakan algoritma SMOTE. Berikut adalah hasil penyeimbangan data dari data fitur yang berhasil diekstrak dari contoh kasus sistem informasi penemuan barang hilang dan sisterm informasi ketersediaan dosen.

Tabel 3.8 Hasil Balancing Data Menggunakan Algoritma SMOTE dengan N=300

Kode	Mean	Standard Deviation	Maximum
F01	0.062	0.119	0.3
F02	0.03	0.047	0.1
F03	0.076	0.185	0.5
F04	0.076	0.185	0.5
F05	0	0	0
F06	0.066	0.119	0.3
F07	0	0	0
F11	0.049394	0.075	0.197
F12	0.149962	0.219	0.642
F13	0.049394	0.075	0.197
F14	0.081858	0.117	0.281

Kode	Mean	Standard Deviation	Maximum
F15	0.081858	0.117	0.281
F16	0.17665	0.228	0.642
F17	0.066294	0.093	0.21
F18	0.042397	0.063	0.164
F19	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0
SMOTE	0	0	0

3.5 Penyusunan Model Klasifikasi

Pada tahap ini data-data pernyataan kebutuhan yang sudah direpresentasikan sebagai vektor fitur akan dibagi menjadi 2 kelompok, yaitu data latih dan data uji. Data latih digunakan untuk menyusun model klasifikasi yang kemudian akan digunakan untuk mengklasifikasi data uji.

3.6 Pengujian dan Analisis

Tujuan dari pengujian adalah untuk membuktikan bahwa metode yang diajukan dapat mendeteksi derau pada pernyataan kebutuhan dalam dokumen SKPL secara akurat. Data uji yang telah melalui tahap ekstraksi fitur seperti pada tabel 3.6 akan menjadi masukan yang akan diklasifikasi dengan menggunakan model klasifikasi yang sudah dilatih menggunakan data latih. Label output dari hasil klasifikasi ini kemudian akan dibandingkan dengan label hasil penilaian secara manual untuk kemudian dihitung nilai akurasi, recall, dan presisinya.

Skenario uji coba yang akan dilakukan dalam penelitian ini adalah sebagai berikut.

1. Membandingkan metode penghitungan kemiripan dengan metode yang lain sehingga dapat diketahui metode penghitungan kemiripan yang terbaik untuk digunakan dalam kasus ini.
2. Menambahkan fitur statistik yang lain pada tahap ekstraksi fitur untuk melihat perbedaan performa hasil klasifikasi.
3. Menggunakan metode klasifikasi *supervised* lain selain SVM seperti k-Nearest Neighbor dan Neural Network untuk membuktikan bahwa metode ekstraksi fitur yang diajukan dapat dipakai untuk klasifikasi tanpa terikat dengan metode klasifikasi yang digunakan.
4. Melakukan variasi persentase data derau yang digunakan untuk kemudian dibandingkan performanya.

3.7 Penyusunan Laporan Penelitian

Pada tahap ini, semua hasil penelitian akan didokumentasikan dalam bentuk laporan penelitian.

3.8 Jadwal Penelitian

Jadwal penelitian yang dilakukan dapat dilihat pada tabel beritkut

Tabel 3.9 Jadwal Kegiatan Penelitian Tesis

No.	Kegiatan	Bulan															
		Maret 2018				April 2018				Mei 2018				Juni 2018			
1.	Pengumpulan dan Pelabelan Data																
2.	Studi Literatur																
3.	Praproses Data																
4.	Penyiapan Fitur																
5.	Penyusunan Model Klasfikasi																
6	Pengujian dan Analisis																
7	Penyusunan Laporan Penelitian																

DAFTAR PUSTAKA

[1] B. Meyer, "On Formalism in Specification," *IEEE Software,* 1985.

[2] D. M. Berry and E. Kamsties, "Ambiguity in Requirements Specification," in *Perspectives on Software Requirements*, Kluwer Academic Publishers, 2004, pp. 7-44.

[3] H. Xu and B. Yu, "Automatic thesaurus construction for spam filtering using revised back propagation neural network," *Expert Systems with Applications,* no. 37, pp. 18-23, 2010.

[4] B. Trstenjak, S. Mikac and D. Donko, "KNN with TF-IDF Based Framework for Text Categorization," *Procedia Engineering*, no. 69, pp. 1356-1364, 2014.

[5] B. Chandra and M. Gupta, "An efficient statistical feature selection approach for classification of gene," *Journal of Biomedical Informatics,* vol. 44, pp. 529-535, 2011.

[6] W. Zhou, L. Yu, W. Qiu, Y. Zhou and M. Wu, " Local Gradient Patterns (LGP): An Effective Local-Statistical-Feature Extraction Scheme for No-Reference Image Quality Assessment," *Information Sciences,* Vols. 397-398, pp. 1-14, 2017.

[7] A. Singhal, "Modern Information Retrieval: A Brief Overview," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2001.

[8] J. C. Platt, "Sequential Minimal Optimization : A Fast Algorithm for Training Support Vector Machines," Technical Report MSR-TR-98-14, 1998.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall and P. W. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence,* no. 16, pp. 321-357, 2002.