PROPOSAL TESIS

PENYUSUNAN REKOMENDASI BERDASARKAN ESKTRAKSI FITUR PRODUK DAN JENIS OPINI DARI OPINI PENGGUNA

Divi Galih Prasetyo putri
NRP. 5114201045

DOSEN PEMBIMBING

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

NIP. 197411232006041001

PROGRAM MAGISTER

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2015

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN
PROPOSAL TESIS

Judul :

Oleh :

NRP :

Telah diseminarkan pada:

Hari :

Tanggal :

Tempat :

Mengetahui / menyetujui

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

[Halaman ini sengaja dikosongkan]

Nama Mahasiswa :

NRP :

Pembimbing :

ABSTRAK

Kata kunci*:*

[Halaman ini sengaja dikosongkan]

Nama Mahasiswa :

NRP :

Pembimbing :

ABSTRACT

***Keywords**:*

[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

Hal yang paling sulit dilakukan dalam pembangunan perangkat lunak adalah menentukan secara pasti apa saja fitur yang akan dibangun. Ketidakmampuan dalam menyusun kebutuhan perangkat lunak secara lengkap, sesuai kebutuhan, dan tidak ambigu masih dianggap sebagai penyebab terbesar dari kegagalan sebuah perangkat lunak . Rekayasa kebutuhan menekankan pada penggunaan teknik yang sistematis dan berulang untuk memastikan kelengkapan, konsistensi, dan kesesuaian dari kebutuhan perangkat lunak yang disusun . Oleh karena itu, proses rekayasa kebutuhan merupakan proses yang paling krusial dalam pembangunan perangkat lunak.

Beberapa proses dalam melakukan rekayasa kebutuhan antara lain elisitasi, analisis, spesifikasi, verifikasi, dan manajemen. Elisitasi kebutuhan dapat diartikan sebagai proses untuk menemukan, mereview, mendokumentasikan, dan memahami kebutuhan pengguna serta batasan dari sistem yang akan dibangun . Proses elisitasi kemudian dilanjutkan dengan proses analisa untuk menentukan kembali semua kebutuhan pengguna dan batasan sistem hasil dari proses elisitasi. Proses-proses ini sangat penting dilakukan pada tahap awal pengembangan perangkat lunak tidak terkecuali pada *Software Product Line* (SPL). Pada SPL, elisitasi dan analisa kebutuhan berfokus untuk menangkap kesamaan dan keberagaman dari rangkaian produknya. SPL memanfaatkan teknik *commonality* dan *variability analysis* yang merupakan salah satu teknik pada analisa domain. Tujuannya adalah untuk mengidentifikasi kemungkinan penggunaan kembali pada rangkaian produk berikutnya.

Analisa domain pertama kali diperkenalkan oleh Neighbors (Neighbors, 1980) untuk menjelaskan studi yang berkaitan dengan domain permasalahan dari aplikasi sejenis. Analisa ini berkaitan dengan *reuse* dan identifikasi *commonality*, *similarity*, dan variability atas karakter yang menjadikan sekelompok aplikasi tergolong pada sebuah domain. Akan tetapi, metode ini juga dapat digunakan dalam pembangunan *single application project*. Fitur-fitur aplikasi dalam domain yang sama dapat digunakan untuk membantu proses elisitasi produk baru. Sebagai contoh, seorang sistem analyst yang ingin

melakukan proses elisitasi untuk sebuah aplikasi anti-virus baru dapat mengevaluasi fitur-fitur yang ada pada aplikasi anti-virus yang telah ada. Proses tersebut dapat mengurangi biaya pengembangan dan waktu serta meningkatkan kualitas dan *competitiveness* dari produk yang dibangun.

Teknik yang ada untuk melakukan analisa domain seperti *Feature-Oriented Domain Analysis (FODA)* (Cohen & Kang, 1990) atau *Feature Oriented Reuse Method (FORM)* (Kang & Kim, 1998) membutuhkan analisa manual. Analisa biasa dilakukan dengan review manual pada spesifikasi kebutuhan dari versi-versi sebelumnya maupun dari brosur atau *website* produk milik kompetitor. Oleh karena itu, pendekatan ini membutuhkan usaha yang besar dan rentan terhadap kesalahan. Kesuksesan dari proses ini juga dipengaruhi oleh ketersediaan dokumen yang relevan serta *analyst* yang ahli dalam domain tersebut. Pendekatan lain adalah metode *Domain Analysis and Reuse Environment (DARE)* (Frakes & Pierto, 1998) yang memanfaatkan metode penggalian data serta informasi sehingga mampu melakukan ekstraksi dan identifikasi fitur secara otomatis. Pendekatan ini membutuhkan metode untuk melakukan analisa data secara otomatis dan ketersediaan data yang cukup. Penelitian yang telah ada sebelumnya memanfaatkan dokumen spesifikasi kebutuhan perangkat lunak (Nan, 2014) dan brosur (Alessio & Spagnolo, 2013) dari produk perangkat lunak. Data-data tersebut sulit untuk beberapa aplikasi dalam domain tertentu dan dalam terjumlah banyak. Penelitian lain memanfaatkan deskripsi sistem dari *softpedia* untuk menyusun rekomendasi fitur (Davril, Delfosse, & Hariri, 2913). Akan tetapi, penelitian tersebut hanya memanfaatkan deskripsi dari produk untuk menyusun rekomendasi fitur.

Data lain yang dapat bermanfaat dalam penyusunan rekomendasi fitur adalah review produk dari pengguna. Pengguna dapat memberikan opininya mengenai sebuah aplikasi melalui fitur review. Pengguna dapat menuliskan fitur yang diinginkan, komentar akan aplikasi, maupun kepuasan atau ketidakpuasan terhadap fitur dalam aplikasi. Timbal balik yang diberikan oleh pengguna ini merepresentasikan keinginan pengguna dan dapat digunakan untuk membantu mengarahkan alokasi usaha pengembangan perangkat lunak serta meningkatkan kualitas produk. Penelitian sebelumnya menunjukkan bahwa review yang terdapat pada *app store* dapat digunakan oleh *analyst* dan *designer* sistem untuk menggali informasi yang berguna seperti *user requirement, bug report, feature request*, dan dokumentasi dari pengalaman user menggunakan fitur tertentu pada aplikasi (Maleej

& Nabil, 2015). Namun, banyaknya review yang terdapat pada *app store* mengakibatkan sulitnya melakukan analisa terhadap review. Penelitian menunjukkan bahwa pengguna iOS memasukkan rata-rata 22 review setiap hari untuk setiap aplikasi. Aplikasi yang memiliki banyak pengguna seperti facebook menerima 4000 review setiap harinya. Masalah lainnya adalah jenis review yang beragam. Review yang diberikan pengguna dapat berisi saran yang mendukung, ide inovatif, bahkan komen yang buruk. Struktur kalimat yang terdapat dalam review juga cenderung tidak formal. Oleh karena itu, dibutuhkan sebuah metode yang dapat menganalisa review dari pengguna dalam jumlah besar secara otomatis dan untuk mengklasifikasikan review berdasarkan tipenya. Banyak penelitian telah melakukan analisa dokumen yang berisi opini terhadap suatu produk untuk mengekstraksi fitur (Guzman & Maleej, 2014) (Hu & Gong, 2010) maupun sentimen dari sebuah review (Pang & Lee, 2008) (Bing & Zhang, 2012). Salah satu penelitian terbaru menggunakan algoritma *collocation* untuk mendapatkan kandidat fitur dari review produk (Guzman & Maleej, 2014). Dari kandidat-kandidat fitur tersebut, diambil kandidat yang memiliki frekuensi kemunculan paling tinggi. Penelitian ini mampu mendapatkan nilai presisi 60% dan *recall* sebesar 50%. Kekurangan penelitian tersebut adalah tidak dapat mendeteksi fitur-fitur yang tidak banyak disebutkan dalam review. Sebagai pengembangan dari analisa sentimen, dokumen review juga dapat diklasifikasikan dalam beberapa tipe diantaranya *bug report* dan *future feature*. Penelitian menunjukkan bahwa metode Naive bayes mampu mengklasifikasikan review lebih baik dibandingkan metode decision tree dan maximum entropy (Maleej & Nabil, 2015). Hasil klasifikasi ini dapat lebih membantu pengembang untuk menganalisa kebutuhan perawatan dan pengembangan aplikasi. Salah satu sumber data yang menyediakan data deskripsi serta review dari berbagai macam aplikasi adalah *application distribution platform* atau *app store.* Beberapa *app store* yang ada seperti Google Play, Apple Playstore, dan Windows Phone menyimpan lebih dari 3 juta aplikasi. Pengguna dapat memanfaatkan *app store* untuk mencari, memperoleh informasi, mengunduh, dan melakukan instalasi dengan mudah terutama untuk aplikasi yang berjalan pada perangkat *mobile*.

Oleh karena itu, pada penelitian ini diusulkan sebuah kerangka kerja untuk membantu *analyst* dan *designer* sistem dalam menggali informasi untuk membantu proses elisitasi, *maintenance*, maupun pengembangan versi baru perangkat lunak dengan

memanfaatkan informasi yang dapat diperoleh dari *app store*. Penelitian ini berusaha untuk membuat rekomendasi yang lengkap berisi fitur dan keterangan mengenai fitur tersebut. Keterangan berisi informasi apakah fitur tersebut pernah dilaporkan mengandung *bug* atau merupakan permintaan fitur baru dari pengguna. Aplikasi yang dianalisa adalah beberapa aplikasi yang tergolong dalam satu domain. Proses yang akan dilakukan dalam kerangka kerja ini antara lain ekstraksi fitur dari informasi pada app store, penggolongan jenis review, dan pembentukan rekomendasi fitur. Proses ekstraksi fitur akan memanfaatkan metode *collocation*. Penggolongan jenis fitur memanfaatkan metode naive bayes yang telah terbukti mampu menunjukkan hasil yang maksimal. Dengan kerangka kerja tersebut diharapkan dapat menghasilkan rekomendasi fitur yang lebih akurat dan informatif.

1.2 Perumusan Masalah

Rumusan masalah yang diangkat dalam penelitian ini adalah:

1. Bagaimana mengekstraksi fitur-fitur aplikasi secara presisi dan akurat dari informasi yang bisa didapatkan pada *app store*?
2. Bagaimana menganalisa review yang diberikan pengguna untuk meningkatkan rekomendasi pada *analyst* dan *designer* aplikasi?

1.3 Tujuan

Tujuan yang akan dicapai dalam pembuatan tesis ini adalah membangun sebuah rekomendasi yang dapat digunakan pengembang perangkat lunak untuk membantu proses elisitasi, *maintenance*, maupun pengembangan versi baru dari perangkat lunak. Kerangka kerja yang dibangun memanfaatkan analisa informasi yang didapatkan dari *app store*.

1.4 Manfaat

Manfaat dari penelitian ini adalah untuk membantu pengembang perangkat lunak dalam melakukan proses elisitasi, *maintenance*, maupun pengembangan versi baru sebuah perangkat lunak dengan memanfaatkan informasi dari aplikasi-aplikasi dalam domain yang sama.

1.5 Kontribusi Penelitian

Kontribusi yang diharapkan dari penelitian ini adalah penyusunan rekomendasi untuk membantu pengembang perangkat lunak dalam melakukan proses elisitasi,

maintenance, maupun pengembangan versi baru sebuah perangkat lunak dengan memanfaatkan informasi dari aplikasi-aplikasi dalam domain yang sama.

Dalam melakukan analisa digunakan informasi aplikasi yang terdapat pada *app store*. Proses ekstraksi fitur memanfaatkan algoritma collocation. Fitur-fitur yang diekstraksi akan dilengkapi dari informasi yang dapat diperoleh dari review terutama review mengenai *bug report* dan *future feature* atau saran fitur.

Seperti yang telah diterangkan pada Sub bab 1.1, algoritma collocation memiliki kekurangan berhubungan dengan fitur yang jarang disebutkan dalam review. Padahal pada kenyataannya, fitur yang tergolong pada saran fitur cenderung jarang disebutkan pada review. Oleh karena itu, penelitian ini mengusulkan untuk menambahkan proses identifikasi fitur yang dikhususkan pada review yang tergolong dalam kelompok saran fitur. Sehingga, diharapkan hasil dari analisa dapat memberikan informasi yang lebih lengkap kepada pengembang perangkat lunak.

1.6 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

1. Deskripsi dan review dari aplikasi-aplikasi yang dianalisa disusun menggunakan bahasa inggris.
2. Deskripsi dan review dari aplikasi-aplikasi didapatkn dari *app store.*

[Halaman ini sengaja dikosongkan]

BAB 2
KAJIAN PUSTAKA

2.1 Product Feature Extraction

Product feature extraction merupakan proses yang penting dalam *opinion mining* atau *sentiment analysis.* Proses ini secara signifikan mempengaruhi performa dari identifikasi opini serta efektifitas dari perangkuman opini (Wei & Chen, 2010). Dari proses ini diperoleh fitur produk yang dibahas oleh pengguna. Ekstraksi fitur produk dari opini pengguna telah banyak diteliti sebelumnya. Metode-metode yang ada dapat dibagi menjadi dua kelompok, *supervised* dan *unsupervised.* Metode *supervised* menggunakan data opini yang telah dianotasi sebelumnya sebagai data training. Meskipun metode ini dapat menghasilkan keluaran yang baik, butuh waktu yang banyak untuk menyiapkan data pelatihannya. Selain itu, keefektifan metode *supervised* sangat tergantung pada kualitas data latihnya. Sebaliknya pada metode *unsupervised* tidak dibutuhkan data latih untuk melakukan ekstraksi fitur. Metode ini banyak memanfaatkan pola atau susunan bahasa pada dokumen (Jacob, Claudia, & Harrison, 2013) (Wei & Chen, 2010).

2.2 Natural Language Processing

Data opini yang digunakan dalam penelitian ini ditulis dengan menggunakan *natural language* atau bahasa sehari-hari. Oleh karena itu digunakan *Natural language Processing* (NLP) dalam mengolah data. Proses dalam NLP yang digunakan dalam penelitian ini salah satunya adalah POS Tagging. Sering juga disebut *grammatical tagging* atau *word category disambiguation* merupakan proses penandaan kata pada suatu teks atau *corpus* dalam kaitannya dengan suatu kelas kata tertentu berdasarkan definisi dan maknanya. POS Tagging dapat memberikan berbagai informasi tentang kelas kata dari suatu kata dan kata lainnya yang di sekitar kata tersebut. Tujuan dari POS Tagging ini sendiri untuk menentukan imbuhan apa saja yang mungkin ditambahkan atau juga menentukkan kata apa saja yang biasa muncul di sekitar suatu kata.

2.3 Collocation Finding

Collocation merupakan ungkapan yang terdiri dari 2 atau lebih kata yang digunakan untuk mengungkapkan sesuatu (Manning & Schutze, 1999). *Collocation* bisa

terdiri dari pasangan kata benda seperti *strong tea*, frase kata kerja seperti *to make up*, atau frase kata lain seperti *the rich and powerful*. Hal ini sangat banyak digunakan dalam berbagai macam aplikasi yaitu *natural language generation, parsing, computational lexicography*, dan lain-lain. Proses untuk mendapatkan pasangan kata ini sering disebut sebagai *collocation finding*. Pasangan kata yang terdiri dari dua kata disebut sebagai - *bigram*. *Collocation* tidak selalu berasal dari kata yang berdampingan dalam kalimat, pasangan kata juga dapat dipisahkan oleh beberapa kata lain (Guzman & Maleej, 2014). Metode yang paling mudah untuk dilakukan adalah menggunakan perhitungan kemunculan pasangan kata. Metode ini sangat cocok digunakan untuk menemukan frase kata yang sudah pasti berdampingan. Akan tetapi, metode ini kurang cocok untuk menemukan pasangan kata yang tidak berdampingan secara langsung atau dipisahkan oleh kata lain. Metode lain yang dapat digunakan adalah perhitungan mean dan variance serta *hypothesis testing*.

Penelitian ini memanfaatkan *hypothesis testing* karena nilai frekuensi yang tinggi dan *variance* yang rendah dapat merupakan sebuah ketidaksengajaan. Frekuensi yang tinggi mungkin saja terjadi karena kedua kata tersebut banyak muncul secara bersamaan. Digunakan peritungan statistik untuk mengetahui apakah pasangan kata yang sering muncul adalah *collocation*. Perhitungan yang digunakan adalah *likelihood ratio* (Manning & Schutze, 1999).

$$p = \frac{C_2}{N} \tag{2.1}$$

$$p_1 = \frac{C_{12}}{C_1} \tag{2.2}$$

$$p_2 = \frac{C_2 - C_{12}}{N - C_1} \tag{2.3}$$

$$LH_1 = b(C_{12}; C_1, p)b(C_{2\_}C_{12}; N - C_1, p) \tag{2.4}$$

$$LH_2 = b(C_{12}; C_1, p_1)b(C_{2\_}C_{12}; N - C_1, p_2) \tag{2.5}$$

$$log\ \lambda = \log\frac{LH_1}{LH_2} \tag{2.6}$$

Persamaan (2.1) sampai (2.6) merupakan langkah-langkah perhitungan nilai likelihood ratio λ. C_1, C_2, C_{12} adalah frekuensi kemunculan kata pertama, kata kedua, dan kemunculan keduanya bersamaan dalam data teks. N adalah jumlah kata. Likehood ratio

dihitung dari dua hipotesis LH_1 dan LH_2. Hipotesis pertama menunjukkan dimana kata kedua tidak bergantung pada kata pertama dan sebaliknya untuk hipotesis kedua. Untuk menghitung nilai hipotesis digunakan perhitungan distribusi binomial.

2.4 Apple App Store

Apple App Store adalah platform distribusi aplikasi untuk iOS yang dikembangkan dan dikelola Apple Inc. Layanan ini memungkinkan pengguna menjelajah dan mengunduh aplikasi yang dikembangkan dengan Apple iOS SDK. Aplikasi dapat diunduh langsung ke sebuah perangkat iOS atau komputer pribadi (Macintosh atau PC) melalui iTunes. Apllikasi di App Store umumnya ditargetkan untuk perangkat iOS seperti iPhone dan iPad dan dapat memanfaatkan fitur-fitur khusus pada perangkat seperti sensor gerak untuk kontrol permainan dan kamera untuk panggilan video daring. Aplikasi dapat diunduh gratis atau dengan harga yang ditetapkan. Selain itu bisa juga menggunakan cara monetisasi dalam aplikasi seperti iklan atau pembelian item. Apple mengambil 30 persen keuntungan yang didapat melalui aplikasi dan 70 persen sisanya dikembalikan ke produsen aplikasi (2015).

Saat ini sudah ada lebih dari 1,4 juta aplikasi yang tersedia di App Store dan lebih dari 100 milyar jumlah unduhan dari pengguna (Perez, 2015) dengan rata-rata unduhan per aplikasi mencapai 62.500. Selain dapat mengunduh aplikasi, pengguna juga dapat memberikan review, kritik, maupun saran mengenai suatu aplikasi pada App Store. Sehingga App Store dapat menjadi media komunikasi antara pihak pengembang aplikasi dengan pengguna aplikasi. Dalam upaya untuk menjaga kualitas aplikasi yang ada pada App Store, pihak Apple Inc. memberlakukan proses approval yang cukup ketat. Proses approval ini terdiri dari serangkaian proses testing pada aplikasi yang akan dimasukan Store. Sejak tahun 2013, Apple memberlakukan static analysis dalam melakukan proses review aplikasi yang menyebabkan kebanyakan aplikasi yang menggunakan teknik dynamic code reassembly gagal lolos pada proses ini (Nathan, 2015).

2.5 Presisi dan Recall

Untuk menghitung tingkat performansi suatu sistem yang memiliki kemampuan untuk me-*retrieve* informasi-informasi tertentu dapat digunakan perhitungan presisi dan

recall. Presisi adalah pobabilitas informasi yang relevan dari semua informasi yang di-*retrie*

ve sistem. Secara matematis, rumus untuk menghitung presisi dan recall dapat dilihat pada Persamaan (2.7) dan Persamaan (2.8).

$$precision = \frac{relevant\ items\ retrieved}{retrieved\ items} \tag{2.7}$$

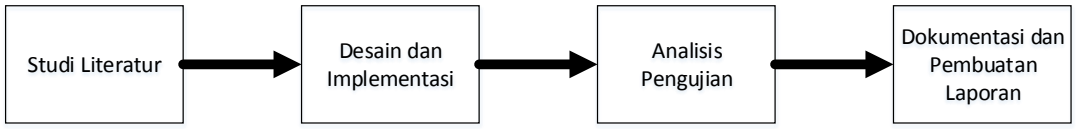
Recall adalah probabilitas informasi relevan yang di-*retrieve* dibandingkan jmlah informasi-informasi yang relevan. Secara matematis *recall* dituliskan sebagai:

$$recall = \frac{relevant\ items\ retrieved}{relevant\ items} \tag{2.8}$$

Recall merupakan ukuran sensitivitas suatu *information retriever* dalam menemukan informasi-informasi yang sebenarnya relevan dengan permintaan (Raghavan, Manning, & Schutze, 2008)

BAB 3
METODOLOGI PENELITIAN

Bab ini akan memaparkan tentang metodologi penelitian yang digunakan pada penelitian ini, yang terdiri dari (1) studi literatur, (2) desain dan implementasi, (3) analisis pengujian, dan (4) dokumentasi sistem dan jadwal kegiatan. Ilustrasi alur metodologi penelitian dapat dilihat pada Gambar 3.1.



Gambar 3.1. Alur metodologi penelitian

Penjelasan dari tahapan metode penelitian pada Gambar 3.1 akan diterangkan secara terperinci pada sub bab 3.1.

3.1 Studi Literatur

Penelitian diawali dengan proses pengkajian yang berkaitan dengan topik penelitian yang diambil. Pada penelitian ini, referensi yang digunakan adalah jurnal-jurnal yang berkaitan dengan penggalian fitur produk, pengolahan opini pengguna, algoritma *collocation*, klasifikasi jenis opini. Dari studi literatur yang telah dilakukan

maka diperoleh informasi yang berkaitan dengan penelitian yang dilakukan ini, seperti berikut:

1 Saat ini *application distrubution platform* merupakan sumber yang sangat penting bagi pengembang untuk mengumpulkan informasi mengenai aplikasi.
2 Review atau opini dari pengguna merupakan salah satu informasi penting dimana pengembang dapat memperoleh banyak masukan untuk mengembangkan aplikasi.
3 Proses ekstraksi fitur produk atau *feature mining* dari data opini pengguna merupakan proses yang sulit dilakukan dikarenakan struktur kalimat yang tidak baku dan jumlah opini yang sangat banyak.
4 Dari data opini juga dapat diperoleh informasi mengenai adanya *bug* atau permintaan fitur baru oleh pengguna.
5 Collocation finding dengan memanfaatkan perhitungan likelihood ratio merupakan salah satu algoritma yang bisa digunakan untuk mengekstraksi fitur, akan tetapi algoritma ini belum dapat menangani fitur-fitur yang jarang disebutkan dalam opini seperti pada data opini yang tergolong permintaan fitur baru.
6 Metode lain yang dapat digunakan untuk menganalisa data opini adalah dengan memanfaatkan aturan bahasa.

Dari studi literatur, dapat disimpulkan juga mengenai kondisi saat ini bahwa:

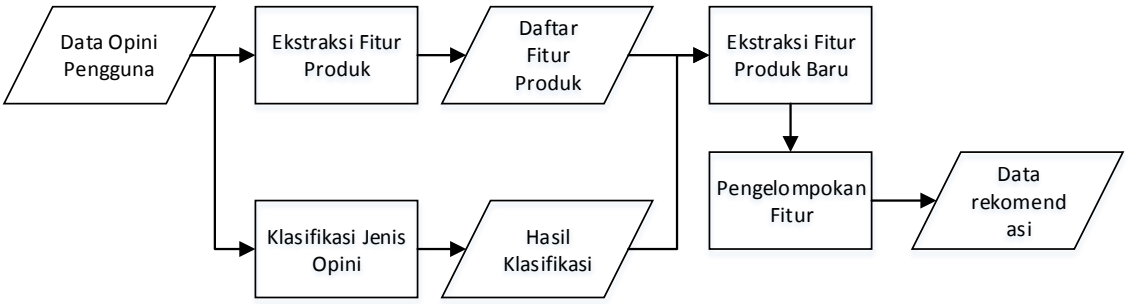
- Sebagai rekomendasi bagi pengembang perangkat lunak, data opini pengguna dapat digunakan untuk menggali informasi mengenai fitur apa saja yang dilaporkan mengandung *bug* dan fitur apa saja yang diinginkan oleh pengguna.
- Dalam melakukan ekstraksi fitur menggunakan *collocation finding*, perlu ditambahkan proses untuk menggali fitur yang frekuensi kemunculannya rendah dalam data opini.

Kedua hasil observasi diatas yang akan dijadikan kontribusi utama dalam penelitian kali ini.

3.2 Desain dan Implementasi

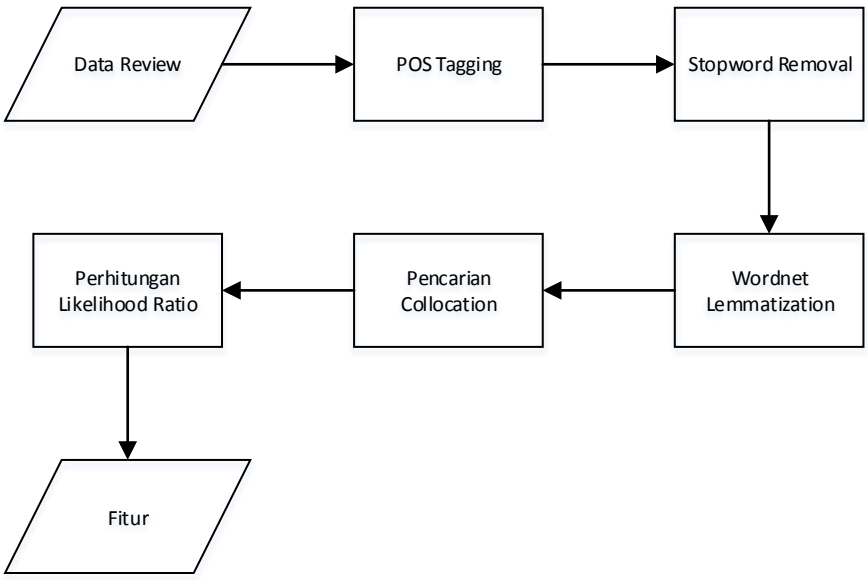
Penelitian ini memanfaatkan beberapa proses dalam menunjang penyusunan rekomendasi fitur dari data opini. Data yang diolah dalam penelitian ini adalah opini masyarakat mengenai sebuah perangkat lunak. Dari data tersebut diekstraksi fitur-fitur yang disebutkan dalam opini. Selain itu, data opini juga akan melalui proses klasifikasi jenis opini untuk mengetahui jenisnya. Hasil dari proses ini akan menunjukkan opini-

opini yang tergolong pelaporan kerusakan dan saran fitur. Daftar fitur dan klasifikasinya kemudian akan dipetakan untuk menunjukkan secara lebih jelas fitur mana yang dilaporkan mengandung kerusakan dan fitur mana yang disarankan oleh pengguna. Dalam proses ini, dimungkinkan muncul opini yang tergolong saran akan tetapi fiturnya belum terdapat dalam daftar. Untuk itu ditambahkan sebuah proses yang secara khusus mengekstraksi fitur dari opini-opini tersebut dan kemudian ditambahkan kedalam daftar hasil pemetaan. Proses selanjutnya adalah pengelompokan fitur-fitur yang sama. Hasil akhir dari keseluruhan proses ini adalah rekomendasi berisi fitur dan keterangan jenisnya. Alur proses secara keseluruhan dapat kita lihat pada Gambar 3.2.



Gambar 3.2. Alur Sistem Pembentukan Rekomendasi

3.2.1 Ekstraksi Fitur



Gambar 3.3 Alur proses ekstraksi fitur

Pada bagian ini akan dilakukan serangkaian proses untuk memperoleh fitur yang terdapat pada setiap opini pengguna. Fitur-fitur hasil dari proses inilah yang nantinya akan digunakan sebagai dasar penyusunan rekomendasi bagi pengembang. Data opini pengguna didapatkan melalui *application distribution platform* atau *app store.* Data yang digunakan merupakan data dari beberapa aplikasi dalam satu domain. Gambar 3.3 menunjukkan alur proses untuk mengekstraksi fitur dari opini. Sebelum dapat menjalankan algoritma *collocation finding* pada opini, perlu dilakukan beberapa proses awal.

Tahapan preprocessing yang perlu dilakukan antara lain:

1. POS Tagging

   Proses pertama adalah penandaan kata dalam kalimat sesuai dengan perannya dalam kalimat. Akan tetapi, dari proses ini hanya diambil kata-kata yang termasuk kata benda, kata kerja, dan kata sifat. Ketiga tipe inilah yang paling sering menyusun fitur produk [].
2. Stopword Removal

   Proses ini ditujukan untuk menghilangkan kata-kata yang sering muncul dan sangat umum digunakan seperti “the”, “of”, “this”, dan kata-kata lain. Selain kata-kata umum, pada proses ini juga akan dihilangkan kata-kata yang menunjukkan sentiment. Penghilangan kata-kata yang berhubungan dengan sentiment dalam opini dapat membantu proses ekstraksi fitur [].
3. WordNet Lemmatization

   Pada proses ini dilakukan pengelompokan untuk semua kata-kata yang sama secara semantik dengan memanfaatkan WordNet.

Algoritma collocation finding digunakan untuk mendapatkan fitur dari sebuah opini. Fitur yang diambil terdiri dari 2 kata (*bigram*) dan memanfaatkan perhitungan likelihood ratio []. Penelitian ini menggunakan 3 *word window* dalam menentukan pasangan kata, yaitu pasangan kata hanya boleh dipisahkan oleh kurang dari 3 kata. Setelah didapatkan semua pasangan collocation, dilakukan perhitungan likelihood ratio untuk mendapatkan fitur. Sebagai contoh terdapat 7 opini pelanggan yang diambil dari salah satu applikasi *messenger* pada Tabel 3.1.

Tabel 3.1 Contoh opini

I have latest update and I still have problem with messages and call notifications! Come on ! FIX THAT!
Notification Error Every message that my friends send I miss until I open the app. The notification should be improved too
Notification problem!!! Almost every Line user talking about problem with notification and you dont give a F..k!!! Very unprofesional!!
No notification. There are no notification if any messages comes, not event call notification pop up if i dont open the application was good, now it's the worse (notifications) Notification problem getting worse every day now.... Line almost doesn't exist on phone until I check it out.. don't receive a single notification without activating the app manually while there is no killing apps or anything shut it down from the first place.. I love it before no I don't like it at all
Was a great app, stopped getting notifications of messages. Uninstalling, won't be coming back nor recommending to friends.
It doesn't give me notification that someone messaged me, if I'm not on the app. I went to send I have notifications turned on but it still doesn't give me a notification. My rating won't go up until you fix it
When my nexus5 update lollipop. Line don't message notification on notification bar. please! fix it. Thank you.

Preprocessing dilakukan pada semua opini diatas sehingga didapatkan hasil pada Tabel 3.2. Setelah itu, proses pencarian *collocation* dijalankan dan dihitung jumlah kemunculannya. Urutan kata tidak dipermasalahkan dan dihitung sama. Hasilnya dapat dilihat pada Tabel 3.3.

Tabel 3.2 Hasil *preprocessing*

late update problem message call notification . Come . FIX . Notification Error. message friend send miss open .notification improve. Notification problem . Line user talk problem notification give F..k. unprofesional . Notification problem get day . Line exist phone check . receive single notification activate kill anything shut first place . stop get notification message. Uninstall, be come recommend friend .give notification someone message . went send notification turn give notification . rating go fix. nexus update lollipop. Line message notification notification bar. Please. Fix. Thank.

Tabel 3.3 Hasil proses pencarian *collocation*

No	Pasangan Kata		Jumlah kemunculan		Total	
1	give notification	notification give	2	2	4	-
2	line notification		2		2	1,71
3	message notification	notification message	3	2	5	8,61
4	notification problem	problem notification	2	2	4	6,73

Tabel 3.3 berisi pasangan kata yang memiliki frekuensi kemunculan lebih dari 1 kali. Langkah selanjutnya adalah melakukan perhitungan likelihood ratio seperti yang telah dijelaskan pada bab sebelumnya. Hasilnya dapat dilihat pada tabel tersebut bahwa pasangan kata nomor 3 memiliki hasil paling tinggi. Oleh karena itu, pasangan kata "message notification" dipilih sebagai fitur.

3.2.2 Klasifikasi Jenis Opini

Opini diklasifikasikan dalam 2 jenis yaitu *bug report* dan *future* feature. Algoritma naive bayes digunakan dalam proses klasifikasi karena algortima ini terbukti mampu memberikan hasil yang lebih baik dibandingkan algoritma MaxEnt dan Dicision Tree [].

3.2.3 Ekstraksi Fitur Tambahan

Ekstraksi fitur tambahan dilakukan dengan memanfaatkan aturan linguistik. Proses ini hanya dilakukan pada opini yang tergolong saran fitur namun tidak memiliki pasangan fitur yang sesuai. Aturan linguistik yang diimplementasikan memanfaatkan 24 kata kunci []. Opini yang diolah minimal harus memiliki satu kata dari daftar kata kunci. Kata-kata kunci tersebut menentukan letak fitur yang disebutkan. Daftar kata kunci dapat dilihat pada Tabel 3.4.

Tabel 3.4 Kata kunci untuk menunjukkan saran fitur

Add, allow, complaint, cloud, hope, if only, improvement, instead of, lacks, to maybe, mising, must, needs, please, prefer, request, should, suggest, waiting for, want, will, wish, would.

Penelitian sebelumnya juga telah meneliti beberapa aturan linguistik untuk mendapatkan fitur dari opini. Sebagai contoh terdapat sebuah opini yang meminta

penambahan fitur dan aturan linguistik yang sesuai dengan opini tersebut. Contohnya dapat dilihat pada Tabel 3.5.

Tabel 3.5 Contoh ekstraksi fitur tambahan

Opini	*"Please add LED color notification like a other chat app"*
Aturan	*Please add <request>*
Request	*LED color notification*

3.2.4 Pengelompokan Fitur

Proses ini ditujukan untuk mengelompokkan fitur-fitur sejenis, termasuk fitur-fitur yang sama secara semantik. Algoritma yang digunakan dalam proses ini adalah aglomerative hierarchical clustering. Setiap fitur akan dianggap sebagai kelompok terpisah dan dihitung kesamaannya dengan fitur-fitur lain menggunakan perhitungan *Jcn* []. Fitur yang memiliki nilai kesamaan tinggi dijadikan satu kelompok fitur. Perhitungan *Jcn* digunakan karena dalam penelitian serupa, perhitungan ini dinilai lebih cocok dari perhitungan lain. Perhitungannya dikembangkan dari perhitungan kesamaan Resnik. Nilai kesamaannya dihitung berdasarkan derajat *Least Common Subsumer* (LCS) seperti telah diterangkan pada bab sebelumnya.

Sebagai contoh terdapat perhitungan kesamaan antara dua fitur yang sama secara semantik pada Tabel 3.6 dan Tabel 3.7. Dari kedua tabel tersebut dapat dilihat bahwa fitur "message notification" dan "chat notification" dinilai lebih dekat. Oleh karena itu, kedua fitur ini dapat dikelompokkan.

Tabel 3.6 Hasil perhitungan kesamaan antara fitur "message notification" dan "chat notification"

	Message	Notification
Chat	0.1168	0.0496
Notification	0.07368	12876699.5

Tabel 3.7 Hasil perhitungan kesamaan antara fitur "message notification" dan "group chat"

	Message	Notification
Group	0.1896	0.0807

Chat	0.1168	0.0496

3.3.2 Skenario Pengujian

a. Dataset

Dalam melakukan penelitian ini digunakan dataset berupa opini pengguna pada beberapa aplikasi dalam satu domain. Dataset diambil dari Apple store, yaitu *app store* khusus untuk aplikasi perangkat iOS. Pengambilan data dilakukan menggunakan *library* yang memanfaatkan teknologi *screen scrapping*. Data yang diambil disimpan dalam database untuk diproses. Selain itu, data pelatihan untuk digunakan pada proses klasifikasi diperoleh melalui penelitian sejenis [].

b. Sistem Penguji

Dalam melakukan pengujian, dibangun sebuah *truth set* dengan bantuan ahli. Ahli-ahli akan melakukan penilaian terhadap data opini sesuai dengan panduan yang telah panduan yang diberikan. Ahli akan diminta untuk menganalisa beberapa data opini dan menyimpulkan beberapa data yaitu fitur yang disebutkan dalam opini dan jenis opini tersebut. Sebelum melakukan penilaian, ahli akan diberikan panduan mengenai definisi dari fitur produk dan jenis-jenis opini. Penilaian oleh ahli akan memanfaatkan kakas bantu yang telah dikembangkan sebelumnya []. *Truth set* yang telah dibangun kemudian dibandingkan dengan hasil rekomendasi baik dari fitur maupun jenis opininya. Penilaian akan dilakukan dengan memanfaatkan perhitungan presisi dan *recall* seperti yang telah diterangkan pada bab sebelumnya.

3.3 Dokumentasi Jadwal dan Pembuatan Sistem

Pada tahap dokumentasi sistem ini akan dilakukan penulisan laporan hasil penelitian dari setiap tahapan yang dilakukan. Tujuan dari tahapan ini adalah menghasilkan dokumentasi tertulis dari penelitian yang dilakukan. Jadwal penelitian yang dilakukan dapat dilihat pada Tabel 3.8.

Tabel 3.8. Jadwal Kegiatan Penelitian Tesis

No.	Kegiatan	Bulan															
		Oktober 2015				Nopember 2015				Desember 2015				Januari 2016			
1.	Studi Literatur																
2.	Analisa dan Perancangan																
3.	Implementasi																
4.	Pengujian dan Evaluasi																

5.	Penyusunan Buku Tugas Thesis																

DAFTAR PUSTAKA

Alessio, F., & Spagnolo, G. O. (2013). Mining commonalities and variabilities from natural language documents.". *Proceedings of the 17th International Software Product Line Conference. ACM.*

Apple's App Store: An economy for 1 percent of developers. (2015, July 4).

Bing, L., & Zhang, L. (2012). A survey of opinion mining and sentiment analysis. *Mining text data. Springer US.*

Cohen, S. G., & Kang, C. K. (1990). eature-oriented domain analysis (FODA) feasibility study. . *Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 1990.*

Davril, j.-m., Delfosse, E., & Hariri, n. (2913). Feature model extraction from large collections of informal product descriptions. *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering. ACM.*

Frakes, W., & Pierto, R. (1998). DARE: Domain analysis and reuse environment. *Annals of Software Engineering*.

Guzman, E., & Maleej, W. (2014). How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews. *Requirements Engineering Conference (RE), 2014 IEEE 22nd International.*

Hu, W., & Gong, Z. (2010). Mining Product Features from Online Reviews. *IEEE International Conference on E-Business Engineering.*

Jacob, Claudia, & Harrison, R. (2013). Retrieving and analyzing mobile apps feature requests from online reviews. *Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on. IEEE.*

Kang, K. C., & Kim, S. (1998). FORM: A feature oriented reuse method with domain specific reference architectures. *Annals of Software Engineering 5.1*.

Maleej, W., & Nabil, H. (2015). Bug Report, Feature Request, or Simply Praise? On Automatically Classifying App Reviews. *Requirements Engineering (RE'15).*

Manning, C., & Schutze, H. (1999). *Foundation of statistical natural language processing.* MIT Press.

Nan, N. (2014). A Systems Approach to Product Line Requirements Reuse. *Systems Journal, IEEE 8.3.*

Nathan, I. (2015). *"Apple's App Store has passed 100 billion app downloads.*

Neighbors, J. (1980). *Software Construction using Components.* Technical Report 160, Department of Information and Computer Sciences University of California.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and trends in information retrieval 2*.

Perez, S. (2015). *iTunes App Store Now Has 1.2 Million Apps, Has Seen 75 Billion Downloads To Date*.

Raghavan, P., Manning, C. D., & Schutze, H. (2008). Evaluation in Information Retrieval. *in Introduction to Information Retrieval*.

Wei, C.-P., & Chen, Y.-M. (2010). Understanding what concerns consumers: a semantic approach to product feature extraction from consumer reviews . *Information Systems and E-Business Management*.