ANALISIS DAN IMPLEMENTASI *CONVERSATIONAL RECOMMENDER SYSTEM* DENGAN MEKANISME PEMILIHAN PERTANYAAN MENGGUNAKAN *REINFORCEMENT LEARNING* BERBASIS *ONTOLOGY*

Ilham Mujaddid Al Masyriq[1], Z.K Abdurahman Baizal [2], Erliansyah Nasution [3]

[1,2,3]Prodi Ilmu Komputasi Telkom University, Bandung
[1]**zielham23@gmail.com**, [2]**bayzal@gmail.com**, [3]**erlinst@yahoo.com**

Abstrak

***Conversational recommender system* merupakan salah satu variasi dari *recommender system* yang ada saat ini. Namun, proses pemilihan pertanyaan yang akan diajukan lebih dulu menjadi masalah jika jumlah pertanyaan yang ada dirasa cukup banyak dan tidak semua pertanyaan bisa diajukan, sehingga hanya dimunculkan beberapa saja.**

Oleh karena itu, diperlukan suatu mekanisme pembelajaran untuk memilih pertanyaan yang layak diajukan terlebih dahulu. Dalam tugas akhir ini digunakan *reinforcement learning* sebagai metode pembelajaran dalam proses pemilihan pertanyaan tersebut.

Dengan *ontology* sebagai basis pengetahuanya, *conversational system* ini melakukan pembelajaran dengan menggali preferensi pengguna dengan mengajukan pertanyaan-pertanyaan berupa kebutuhan fungsional. *Conversational recommender system* ini diharapkan mampu memberikan pertanyaan yang sesuai dengan kebutuhan pengguna saat itu dan memberikan hasil rekomendasi yang akurat sesuai dengan preferensi pengguna sehingga interaksi pengguna pada *system* pun dapat dilakukan seefisien mungkin.

Kata kunci: *conversational recommender system*, *reinforcement learning*, *ontology*, *history*.

Abstract

Conversational recommender system is one of recommender system variation that exist today. However, the process of selecting questions to be asked first will be a problem if the number of questions are quite a lot and not all questions can be asked, only a few are shown.

Therefor, we need a learning mechanism for selecting proper question that feasible to be asked in advance. In this thesis, reinforcement learning is used as a learning method in the process of selecting the questions.

With the ontology as a knowledge base, this conversational recommender system is eliciting user preference by asking the questions about functional reqirement. Conversational recommender system is expected to provide questions that correspond to the current user needs and provide accurate recommendation results according to user preferences so that the user interaction in the system can be done as efficiently as possible.

Keywords: conversational recommender system, reinforcement learning, ontology, history.

1. Pendahuluan

Perkembangan teknologi saat ini telah banyak memberikan kemudahan kepada kita semua dalam segala aspek kehidupan. Dalam hal jual-beli misalnya, tak sedikit penjual yang memanfaatkan teknologi untuk memasarkan produk yang dijualnya. Fenomena ini kemudian terus berkembang dengan munculnya situs-situs yang khusus menjual berbagai macam produk yang melayani jasa jual-beli secara *online* atau lebih dikenal juga dengan sebutan *e-Commerce*.

Alih-alih semakin banyak yang mulai melirik kegiatan *e-Commerce* ini, tingkat persaingan pun semakin bertambah diantara penjual. Banyak kemudian situs jual-beli *online* yang memberikan fitur-fitur yang mampu memberikan layanan dan kemudahan kepada para penggunanya. Salah satu yang paling banyak dikembangkan saat ini adalah *recommender system* yang bisa memberikan rekomendasi produk kepada calon pembeli dari banyaknya katalog produk yang ditawarkan di situs jual-beli. Sehingga pembeli tidak kesulitan untuk mencari dan bisa dengan cepat mendapatkan produk yang mereka.

Conversational recommender system merupakan salah satu variasi dari *recommender system* yang ada pada saat ini. *System* ini mengggali preferensi pengguna dengan cara melakukan tanya jawab dengan pengguna seputar produk yang sedang dicari. Namun, jika saja pertanyaan-pertanyaan yang akan diajukan oleh *system* ini hanya sedikit mungkin tidak akan terjadi masalah. Tetapi, jika pertanyaan yang akan diajukan cukup banyak, akan jadi masalah dalam hal memilih pertanyaan yang harus didahulukan. Hal yang paling mudah dalam mengatasi hal ini adalah dengan memilih pertanyaan secara *random*, namun kurang baik karena kemungkinan ada pertanyaan-pertanyaan yang

seharusnya tidak perlu kemudian diajukan kepada pengguna. Oleh karena itu diperlukan suatu pembelajaran untuk memilih pertanyaan-pertanyaan yang kemungkinan besar akan mendapat respon yang positif dari pengguna saat berintekasi dengan *system*.

Berangkat dari hal ini, kemudian penulis ingin mengajukan sebuah *conversational recommender system* yang mampu melakukan pembelajaran untuk bisa memilih pertanyaan-pertanyan mana saja yang lebih baik didahulukan untuk ditanyakan. Dengan menerapkan *reinforcement lerarning*, *system* ini diharapkan mampu mempelajari interaksi pengguna sebagai bahan pembelajaran *system*. Sehingga pada akhirnya bisa memberikan susunan pertanyaan yang sesuai dengan kebutuhan pengguna baru dengan melihat kecenderungan *history* interaksi *system* dengan pengguna-pengguna sebelumnya. Susunan pertanyaan yang diajukan diharapkan mendapat respon positif dari pengguna karena ketepatan pertanyaan yang diajukan sehingga dapat mempercepat waktu interaksi pengguna dengan *system*.

Hal ini penting karena harapan pengguna dalam menggunakan sebuah fasilitas tentunya untuk mempermudah keperluanhya. Bayangkan saja jika saat pengguna menggunakan sebuah *conversational recommender system*, ternyata pengguna dihadapkan dengan pertanyaan-pertanyaan yang sebenarnya tidak diharapkan atau tidak sesuai dengan kebutuhannya. Bukannya jadi mempermudah, akhirnya jadi mempersulit dan memperlama waktu interaksi dengan *system*. Hal ini bisa membuat pengguna bosan dan memberikan respon yang negatif atau bahkan pergi begitu saja dari *system* karena ketidakpuasannya. Oleh karena itulah, pertanyaan yang efisien sangat dibutuhkan selama proses interaksi pengguna dengan *system*.

2. *Landasan Teori*

2.1 *Conversational Recommender System*

Recommender system bisa dikatakan sebagai sebuah *system* yang mampu memberikan petunjuk atau rekomendasi kepada pengguna dengan menggunakan cara-cara tertentu untuk memberikan pilihan yang kemungkinan besar berguna atau menarik bagi pengguna dari sejumlah besar data [5]. *Recommender system* diperkenalkan untuk membantu pengguna dalam memilih produk ketika sejumlah besar katalog tersedia dan pengguna memiliki pengetahuan yang tidak cukup untuk mengambil pilihan yang terbaik secara mandiri [6].

Berbeda dengan *recommender system* pada umumnya, *conversational recommender system* dibuat lebih interaktif. Pengguna dapat berinteraksi dengan *system* dengan melakukan tanya-jawab seputar kebutuhannya. Layaknya seorang sales di dunia nyata, *conversational recommender system* berinteraski dengan memberikan susunan pertanyaan yang harus dijawab oleh penggunanya.

Dalam setiap sesinya, *conversational recommender system* kemudian memberikan rekomendasi berdasarkan respon atau jawaban dari penggunannya. Ini dilakukan dengan cara mengubah respon atau jawaban dari pengguna ke dalam bentuk *query* yang kemudian digunakan untuk mencari produk yang cocok dengan kriteria pengguna dengan produk yang ada di dalam katalog produk.

Dalam rangka meniru perilaku yang dilakukan oleh seorang sales yang sedang berinteraksi dengan customernya, tentunya pemilihan pertanyaan yang akan diajukan kepada *customer* merupakan hal yang sangat penting untuk dipertimbangkan. Mengingat tujuan utama sebuah *recommender system* adalah untuk mempermudah seseorang dalam menentukan pilihan. Akan jadi memberatkan jika pertanyaan-pertanyaan yang diajukan ternyata tidak sesuai dengan yang dibutuhkan penggunanya. Bukan terbantu, tapi akhirnya menjadi disusahkan oleh *recommender system* itu sendiri. Oleh karenanya, menentukan pertanyaan yang kemungkinan besar mendapakan respon positif dari pengguna merupakan hal yang sangat penting yang bisa dilakukan oleh sebuah *conversational recommender system*.

2.2 Rinforcement Learning

Reinforcement learning adalah belajar tentang apa yang sebaiknya dilakukan, yaitu dengan cara memetakan situasi yang dihadapi kedalam sebuah aksi untuk memaksimalkan nilai *reward* yang diberikan *system* jika aksi yang diambil itu benar [1] [8]. Ibarat manusia yang selalu mencoba-coba bermacam tindakan disaat dihadapkan pada suatu masalah, kemudian mencari jalan keluar yang dianggap terbaik diantara pilihan-pilihan tindakan yang tersedia dengan mengamati hasil yang didapat dari setiap tindakan yang telah diambil. Jika tindakan yang diambil itu ternyata baik, maka hasil yang didapat juga baik, begitu sebaliknya.

Sebagai contoh lain, seorang murid yang sedang belajar di kelas. Jika dia selalu bersikap baik dalam setiap tindakannya, tentu sang guru akan mengapresiasi murid tersebut dengan memberi nilai yang baik. Namu sbaliknya, jika ternyata suatu saat murid tersebut melakukan tindakan yang kurang baik, tentu guru tersebut juga akan memberi hukuman atas tindakannya tersebut.

2.2.1 Action Value

Merupakan nilai estimasi atau nilai rata-rata dari *reward* yang didapat oleh sebuah aksi yang dalam tugas akhir ini berupa pertanyaan setiap kali pertanyaan tersebut dipilih untuk diajukan. Dengan kata lain, jika pada saat interaksi ke *t*, pertanyaan *a* telah diajukan sebanyak k_a kali, dengan hasil *reward* $r_1, r_2, \ldots, r_{ka}$, maka nilai estimasinya,

$$Q_t(a) = \frac{(r_1 + r_2 + \ldots + r_{k_a})}{k_a} \quad (1)$$

Untuk menghindari alokasi memori dan proses komputasi yang besar karena nilai *reward* yang terus bertambah setiap waktu, dengan melakukan *incremental update*, rata-rata *reward* pada saat k+1 dapat dihitung dengan

$$\begin{aligned} Q_{k+1} &= \frac{1}{k+1}.\sum_{i=1}^{k+1} r(i) \quad ket: i = 1 \ldots k+1 \\ &= \frac{1}{k+1}[r_{k+1} + \sum_{i=1}^{k+1} r_i \,] \quad ket: i = 1 \ldots k+1 \\ &= \frac{1}{k+1}[r_{k+1} + kQ_k + Q_k - Q_k] \\ &= \frac{1}{k+1}[r_{k+1} + (k+1).Q_k - Q_k] \\ &= Q_k + \frac{1}{k+1}[r_{k+1} - Q_k] \end{aligned} \quad (2)$$

atau secara umum dapat ditulis sebagai berikut [8]

$$NewEstimate \leftarrow OldEstimate + Stepsize[T\arg et - OldEstimate]$$

2.2.2 Epsilon Greedy (Є-Greedy)

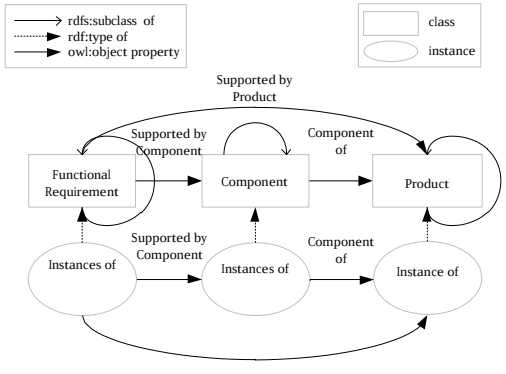
Untuk menyeimbangkan proses ekplorasi dan eksploitasi ini dapat dilakukan dengan cara memilih *greedy action* dengan probabilitas *(1- $Є_t$)* dan memilih satu dari aksi yang lain dengan probabilitas $Є_t$. Sebagai contoh, nilai $Є_t$ dapat didefinisikan sebagai berikut [3].

$$\varepsilon_t = \frac{G_1}{G_2 + t} \quad (3)$$

positif constants. Semakin besar nilai G1 dan G2, Dimana $G_2 > G_1$ dan G_1 dan G_2 adalah *large*

maka proses pembelajaran akan semakin lama.
Kita dapat mengenerate nilai *random* $rand_t$ diantara 0 sampai 1. Jika $rand_t$ lebih besar dari $Є_t$ maka pilih *greedy action*, sebaliknya pilih aksi yang lain.

2.3 Struktur Ontology



- *Product* = merupakan *class* yang merepresentasikan produk yang ada dalam domain.

- *Component* = merupakan *class* yang merepresentasikan spesifikasi produk yang memetakan antara kebutuhan fungsional dan produk.

Pada [11] diajukan suatu model *ontology* yang terdiri dari tiga *class* yang merepresentasikan kebutuhan fungsional, produk dan spesifikasi produk. Semua *class* tersebut mempunyai *subclass* dan individu sebagai representasi dari sebuah hirarki. Hihrarki *class* dihubungkan dengan *class* lainnya lewat object property seperti yang ditunjukkan pada gambar 1.

2.4 Menentukan Hasil Rekomendasi

Selama proses interaksi dengan pengguna, *system* secara tidak langsung membangun preferensi (profil) pengguna yang nantinya digunakan untuk mencari kecocokan kebutuhan pengguna dengan katalog item yang ada dalam *ontology*. Profil pengguna ini menyimpan *hardconstraint*, dan *softconstraint* beserta derajat ketertarikan (*degree of interest*) dari setiap *softcontraint*-nya.

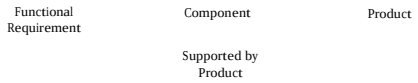
Setelah profil pengguna terbentuk, produk kemudian dibangkitkan berdasarkan kebutuhan fungsional yang dipilih dan dilakukan perankingan untuk didapatkan produk yang paling cocok dengan kebutuhan pengguna. Pada [11] disebutkan beberapa definisi dalam pembangkitan produk.

Definisi 1. Sebuah produk p dikatakan memenuhi kebutuhan fungsional f, dinotasikan $p \mapsto f$ jika dan hanya jika

$$UpperS\{c \,|\, \forall c.SuppBy(f,c)\} = UpperS\left|\begin{array}{l}\{c \,|\, \forall c.SuppBy(f,c)\} \\ \cap \\ \{c \,|\, \forall c.HasComp(p,c)\}\end{array}\right|$$

Dimana:
- *UpperS* = Fungsi untuk medapatkan himpunan parents dari sebuah himpunan *node* dalam ontology.
- *SuppBy* = Relasi yang meghubungkan elemen-elemen himpunan individu pada hirarki *Functional Requirement* dengan elemen-elemen himpunan individu pada hirarki *Component*. Relasi ini menunjukkan bahwa suatu kebutuhan fungsional didukung oleh komponen tertentu.
- *HasComp* = Relasi yang menghubungkan elemen-elemen himpunan pada hirarki *Product* dengan elemen-elemen himpunan pada hirarki *Component*. Relasi ini menunjukkan bahwa suatu produk mempunyai *Component* tertentu.

Definisi 2. Fungsi untuk mencari himpunan



Gambar 1 Struktur Ontology

Keterangan:

- *Functional Requirement* = merupakan *class* yang merepresentasikan kebutuhan fungsional.

produk yang sesuai dengan kebutuhan fungsional didefinisikan dengan fungsi $SearchP$.

$$SearchP : N \rightarrow N$$

Untuk suatu kebutuhan fungsional $F = R_{soft} \cup R_{hard}$,

Maka himpunan produk yang sesuai dengan kebutuhan fungsional adalah,

$$SearchP\ F = \left(p \forall p. \left(\vee \quad p \mapsto s \right) \quad p \mapsto h \right)$$

$$\{ \mid \quad \mid \quad {}_i \mid \wedge \quad {}_j \}$$

$$s_i \in R_{soft} \qquad h_j \in R_{hard}$$

2.4.1 Derajat Ketertarikan *(Degree of Interest)*

Derajat ketertarikan *(degree of interest* disingkat *DOI)* adalah sebuah nilai yang merepresentasikan tingkat ketertarikan pengguna terhadap sesuatu, dalam hal ini adalah *softconstraint* atau kebutuhan fungsional yang ada. *DOI* bisa didapatkan dengan menormalisasi dari nilai *reward* yang didapat oleh setiap kebutuhan fungsional selama interaksi dengan pengguna sebelumnya.

$$DOI_i = \frac{i}{\sum_{i=1}^{n} i} \tag{4}$$

Jika belum ada data interaksi dari pengguna sebelumnya, nilai *DOI* dibagi rata ke setiap kebutuhan fungsional yang dipilih. Nilai *DOI* ini nantinya digunakan dalam perhitungan bersama dengan *utility function* sebagai pertimbangan *system* dalam perankingan produk yang ada.

2.4.2 Utility Function

Utility function digunakan untuk meranking produk yang didapat dari hasil penelusuran *semantik* pada *ontology*. Sebuah model preferensi berdasarkan metode MAUT (*Multi-Attribute Utility Theory*) seperti yang disebutkan pada [2] dinyatakan sebagai pasangan $(\{V_1, V_2, ..., V_n\}, \{w_1, w_2, ..., w_n\})$ dengan V_i adalah nilai dari atribut A_i dan w_i adalah *relatif importance* dari A_i. Dengan demikian *utility* dari produk $(\langle a_1, a_2, ..., a_n \rangle)$ adalah sebagai berikut:

$$U(\langle a_1, a_2, ..., a_n \rangle) = \sum_{i=1}^{n} w_i V_i(a_i) \tag{5}$$

Dengan $\sum_{i=1}^{n} w_i = 1$

Metode MAUT tersebut lebih sesuai untuk sebuah preferensi yang mengacu pada atribut atau fitur teknis dari sebuah produk secara langsung. Sedangkan dalam penelitian ini pengguana menyatakan preferensinya berupa kebutuhan fungsional dari produk (bukan fitur), juga terdapat beberapa pertimbangan lain seperti *DOI* dan nilai support level dari setiap produk yang juga harus dilibatkan dalam perhitungan. Maka persamaan diatas dapat disesuaikan menjadi:

$$U(FR) = DOI \times \sum_{i=1}^{n_j} S(c_i) \times w_i(FR) \tag{6}$$

- n_j = banyaknya tingkat dukungan komponen untuk kebutuhan fungsional FR.

2.5 Penjelasan (Explanation)

Sebuah penjelasan (*explanation*) dalam sebuah *recommender system* dapat memegang peranan penting dalam meningkatkan *user experience* dalam penggunaan *recommender system* tersebut. Penjelasan dapat memiliki banyak keuntungan, mulai dari menginspirasi kepercayaan pengguna untuk membantu pengguna mebuat suatu keputusan yang baik.

Berdasarkan [9] [10] ada tujuh kemungkinan tujuan *explanation* dalam sebuah recmmender *system*, yaitu:

1. *Transparency*, menjelaskan bagaimana suatu *system* dapat bekerja.
2. *Scrutability*, mengijinkan pengguna untuk mengatakan kepada *system* bahwa sesuatu itu salah.
3. *Trustworthiness*, meningkatkan kepercayaan pengguna terhadap *system*.
4. *Effectiveness* , membantu pengguna untuk membuat suatu keputusan yang baik.
5. *Persuasiveness*, meyakinkan pengguna untuk mencoba atau membeli suatu produk.
6. *Efficiency*, membantu pengguna untuk membuat keputusan lebih cepat.
7. *Satisfaction*, meningkatkan kenyamanan pengguna pada saat berinteraksi dengan *system*.

Proses pembangkitan penjelasan dilakukan dengan cara penelusuran *backtrack* pada masing-masing lintasan *node* dan relasinya dalam *user* profil model, berawal dari suatu *node* produk yang akan direkomendasikan. Untuk menghasilkan penjelasan yang lebih natural, pembangkitan penjelasan akan menggunakan template diapdukan dengan relasi serta info dari *node* hasil penelusuran *backtrack* pada *user* profil model [11]. Template yang digunakan adalah sebagai berikut,

<produk> cocok dengan kebutuhan Anda, karena <produk> <relasi> <nodeinfo> <relasi> <nodeinfo>…

2.6 User Case Framework

Merupakan skenario yang dipersiapkan untuk menyelesaikan beberapa kasus yang mungkin terjadi pada saat *system* berinteraksi dengan pengguna. Setidaknya ada lima skenario yang dibahas pada [11].

Dimana:

- DOI = DOI untuk masing-masing kebutuhan fungsional FR
- $S(c_i)$ = nilai support level dari tingkat dukungan *Component* c_i (normalized).
- $w_i(FR)$= *relatif importance* dari tingkat dukungan *Component* c_i untuk kebutuhan fungsional FR .

2.6.1 ***Case U1 – Empty User Profile***

Ini adalah kondisi pengguna saat pertama kali berinteraksi dengan *system* dimana profil pengguna masih kosong. Strategi yang dilakukan adalah dengan menentukan pertanyaan awal untuk memulai interaksi.

GenerateU1(UserModel(user))

/ memilih sejumlah maksimal α nodes dari himpunan nodes pada level 1.*
Input : UserModel(User), current user model
*Output : Q, Set of Question */*
Begin
/ F1 : set of Functional Requirement nodes level 1 */*
$Q \leftarrow SelectQ(\alpha, F_1, UserModel(user))$
End

Pada [11] dijelaskan bahwa *SelecQ* merupakan fungsi yang digunakan pada saat pembangkitan pertanyaan pada saat interaksi dengan cara memilih sejumlah himpunan *nodes* yang memang potensial untuk ditanyakan berdasarkan keadaan *user* profile saat ini.

$$SelecQ\ (\max Q, A, UserModel(user)):$$
$$\{B \mid B \in P(A), |B| = \min\{\max Q, |A|\}\}$$

Dimana:

- A = himpunan *nodes* yang potensial untuk ditanyakan.
- $P(A)$ = *powerset* dari himpunan A.
- $|B|$ = cardinalitas dari himpunan B.

2.6.2 *Case U2* – Terdapat Lebih Dari Satu Produk Hasil Rekomendasi Yang Dipilih Pengguna

Hal ini menandakan pengguna masih ragu-ragu antara beberapa produk (*k* produk) yang dia pilih. Oleh karena itu harus dibangkitkan pertanyaan yang mengacu pada elemen pembeda antar kelompok produk untuk membantu pengguna memilih dari sekian n produk yang dia pilih.

GenerateU2(UserModel(user))
/ Input : UserModel(User), current user model*
*Output : Q, Set of Question */*
Begin
/ Menentukan supporting elemen dari k produk yang dipilih pengguna */*
For i← 1 to k do
$Temp \leftarrow DL[\exists CompOf\{p_i\}]$

$$SuppElement(p_i) = \left(Temp \cup \left(\bigcup_{z_j \in Temp} DL[\exists SuppBy\{z_j\}]\right)\right)$$

/ Melakukan clustering terhadap himpunan produk P */*
$Clustering(P)$
/ Himpunan pertanyaan Q diinisialisasi dengan himpunan kosong */*
$Q \leftarrow \{\}$
/ Menentukan Distinguishing element sekaligus menentukan pertanyaan */*
/ c adalah jumlah cluster*/*
For j ← 1 to c do
sejumlah maksimal $\lfloor a/c \rfloor$ *nodes dari masing-masing himpunan* $DG(g_j)$ **/*
$Q_j \leftarrow SelectQ\left(\lfloor a/c \rfloor DE(g_j), UserModel(user)\right)$
$Q \leftarrow Q \cup Q_j$
End

2.6.3 *Case 3* – Tidak Ada Satupun Produk Yang Dipilih Pengguna

Dengan demikian *system* harus mempertimbangkan semua *user* profil model sebelumya yang tidak relevan. Strategi yang dilakukan adalah dengan cara menelusuri *nodes* pada *user preference model*, untuk menanyakan alternatif kebutuhan fungsional lain dalam satu level maupun *backtrack* ke *nodes* kebutuhan fungsional pada level sebelumnya yang belum ditanyakan.

GenerateU3(UserModel(user),i)
/ Input : i, (i>0) adalah level hirarki kebutuhan fungsional pada ontology yang terakhir ditanyakan.*

UserModel(user), current user model

Output : Q, Set of Question
*F_{ab} adalah set nodes kebutuhan fungsional pada level a dengan status b */*
Begin
if i=1 then

$candidat \leftarrow F_i - \{F_{ivh} \cup F_{ivs} \cup F_{ix}\}$

/ candidat adalah set nodes dari kebutuhan fungsional yang potensial untuk ditanyakan*/*
if $candidat \neq \{\}$ *then*
$Q \leftarrow SelectQ(\alpha, candidat, UserModel(user))$
else
$GenerateU\ 1(UserModel(user))$
else
$candidat \leftarrow (children(F_{(i-1)vs} \cup F_{(i-1)vh}) - \{F_{ivh} \cup F_{ivs} \cup F_{ix}\})$
if $candidat \neq \{\}$ *then*
$Q \leftarrow SelectQ(\alpha, candidat, UserModel(user))$
else
$GenerateU\ 3(UserModel(user), i-1)$
End

2.6.4 *Case 4* – Definisi Kebutuhan Belum Cukup Untuk Membangkitkan Rekomendasi

Kebutuhan yang dimasukkan masih terlalu umum, untuk itu perlu dibangkitkan kebutuhan fungsional yang lebih spesifik (level berikutnya), dan tanyakan sejumlah α kebutuhan fungsional pada level tersebut agar kebutuhan lebih spesifik. Pertanyaan yang diajukan *system* dalam interaksi selanjutnya adalah kebutuhan fungsional yang terkait dengan jawaban sebelumnya dari pengguna.

GenerateU3(UserModel(user),i)

$$DE(g_j) \leftarrow SuppElements(g_j) \cap \left| \bigoplus_{i=1}^{c} SuppElements(g_j) \right|$$

*/*masing-masing Qj didapatkan dengan memilih*

/ Input : i, (i>0) adalah level hirarki kebutuhan fungsional pada ontology yang terakhir ditanyakan.*

UserModel(user), current user model

*Output : Q, Set of Question */*

Begin

/* *Kandidat adalah set nodes dari kebutuhan fungsional yang potensial untuk ditanyakan* */
candidat $\leftarrow children(F_{ivh} \cup F_{ivs})$
if candidat $\neq \{\}$ *then*

$Q \leftarrow SelectQ(\alpha, candidat, UserModel(user))$
else

$GenerateU3(UserModel(user), i)$
End

2.6.5 *Case 5* – Tidak Ada Hasil Rekomendasi Yang Cocok Dengan Profil Pengguna

Tidak ada produk yang sesuai dengan kebutuhan pengguna disebabkan oleh adanya kontradiksi dari kombinasi *hardconstraint* pada saat produk *retrieval*, karena kombinasi query dari *softconstraint* pasti menghasilkan produk. Dengan demikian diberikan penjelasan bahwa ada satu atau beberapa *hardconstraint* yang telah dimasukkan pengguna menyebabkan produk tidak ditemukan. Strategi yang dialkukan adalah dengan merubah *hardconstraint*.

2.7 Evaluasi Hasil Rekomendasi

Evalusi hasil rekomendasi diperlukan untuk melihat seberapa baik *conversational recommender system* yang sudah dibuat dalam merekomendasikan suaatu produk. Untuk menguji hal itu, dapat dilakukan dengan menghitung rangking hasil rekomendasi yang diberikan *system* kemudian mebandingkannya dengan ranking sebenarnya yang diberikan oleh *expert user*, dalam hal ini orang yang dianggap kompeten dan pengetahuan yang baik mengenai domain produk yang digunakan dalam study kasus tugas akhir ini, yakni *smartphone*.

Untuk pengujiannya sendiri, kita gunakan *Mean Absolute Error (MAE)* seperti yand dijelaskan pada [7]. *MAE* dapat digunakna untuk menghitung kesalahan absolut dantara ranking yang diberikan *system* dengan ranking yang sebenarnya diberikan oleh *expert user*.

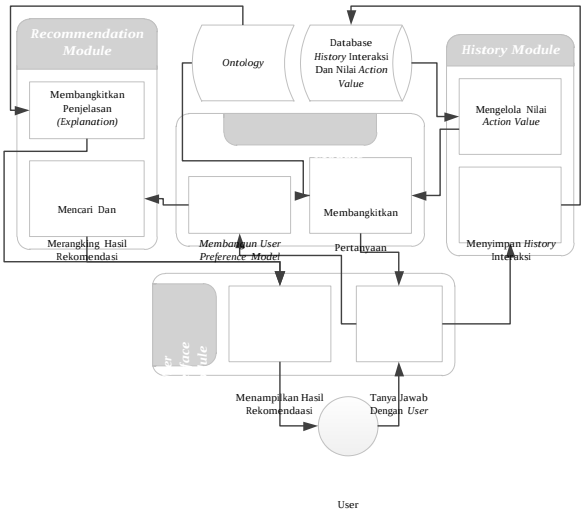
$$\left|\overline{E}\right| = \frac{1}{n}\sum_{i=1}^{n}\left|p_i - r_i\right| \quad (7)$$

Dimana:
- n = jumlah produk yang direkomendasikan.
- p_i = ranking yang diberikan oleh *system*.
- r_i = ranking sesungguhnya yang diberikan oleh *expert user*.

3. Perancangan Sistem

3.1 Gambaran Umum Sistem

Secara garis besar, *system* yang dibangun dalam tugas akhir ini dibagi menjadi empat buah modul yang masing-masing memiliki tugas tersendri, yaitu *history module*, *user model module*, *recommendation module*, dan *user interface module*.



Gambar 2 Gambaran Umum Sistem

1. History Module

Modul ini bertugas untuk mengelola data *history* interaksi pengguna dan juga nilai *action value* yang sudah didapat oleh setiap kebutuhan fungsional dalam setiap interaksi. *Reward* yang didapat setiap kebutuhan fungsional dalam setiap interaksi, semuanya diakumulasi dan disimpan untuk kemudian digunakan kembali sebagai bahan pertimbangan pembangkitan pertanyaan di interaksi berikutnya dilakukan oleh modul ini juga.

2. User Model Module

Modul ini berfokus pada interaksi *system* dengan *ontology*. Bertugas untuk menyiapkan setiap pertanyaan yang akan diajukan kepada pengguna. Bersama dengan *history module*, modul ini mengelola *node-node* kebutuhan fungsional dari *ontology* dan mengurutkannya berdasarkan nilai *action value* masing-masing kebutuhan fungsional untuk kemudian diajukan kepada pengguna.

3. User Interface Module

Modul ini bertugas sebagai *user interface system* terhadap pengguna. Semua interaksi *system* dengan pengguna dilakukan pada modul ini. Pertanyaan yang sudah disiapkan oleh *user model module* dan *history module* ditampilkan kepada pengguna oleh modul ini. Sehingga memudahkan pengguna untuk berinteraksi secara langsung dengan *system*. Respon dari pengguna di setiap sesi, kemudian dikirimkan ke *history module* untuk disimpan dan diolah untuk keperluan selanjutnya.

Pada saat sesi rekomendasi juga, modul ini yang bertugas untuk menampilkan semua hasil rekomendasi yang sudah diurutkan dan diberi penjelasan oleh *recommendation module*.

4. *Recommendation Module*

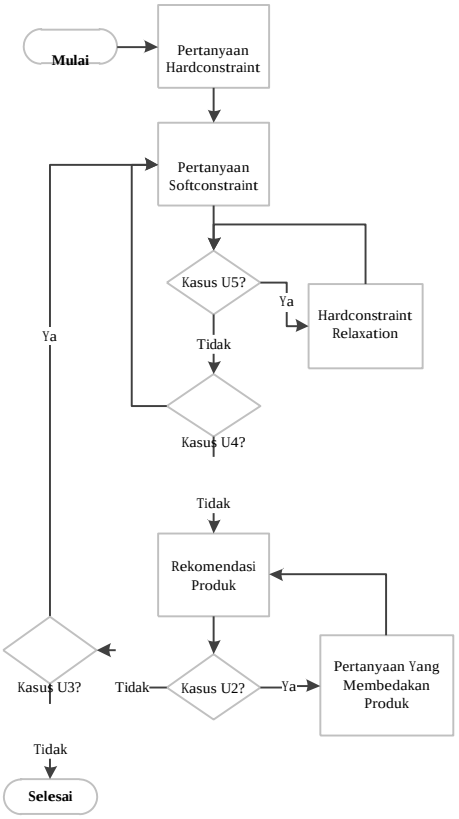
Modul ini bertugas untuk mencari produk yang sesuai dengan profil pengguna kemudian mengurutkannya berdasarkan nilai *utility function*-nya untuk kemudian dikirimkan ke *user interface module* untuk ditampilkan kepada pengguna

sehingga mempermudah pengguna untuk memilih produk yang sesuai dengan keinginannya. Pada sesi

rekomendasi, modul ini akan membangkitkan *explanation* (penjelasan) dari setiap produk yang akan direkomendasikan kepada pengguna, sehingga pengguna lebih mudah dalam memahami kelebihan dan kekurangan dari setiap produk yang direkomendasikan oleh *system*.

3.2 Alur Interaksi

Pada gambar 3 diperlihatkan alur interaksi *system* dari mulai awal interaksi hingga selesai. Pada saat pertama kali pengguna berinteraksi dengan *conversational recommender system*, pengguna akan diberikan pertanyaan *hardconstraint*, dimana produk yang nantinya direkomendasikan harus memenuhi semua hardconstrain yang dimasukkan. *Hardconstraint* yang terdapat pada *conversational recommender system* ini berupa harga dan *brand* (merk) dari produk.



Gambar 3 Alur Interaksi Sistem

Setelah memberikan pertanyaan *hardconstraint*, *system* kemudian akan memberikan pertanyaan *softconstraint* berupa kebutuhan fungsional dari produk yang ingin dicari oleh pengguna. Produk yang nantinya dibangkitkan, tak harus memenuhi semua softcontrain yang dipilih pengguna, jika produk hanya mensupport satu fungsional saja, maka produk tersebut akan tetap direkomendasikan.

Jika produk yang memenuhi *hardconstraint* dan *softconstraint* masih terlalu banyak, yakni dia atas *quantity threshold* yang telah ditentukan sebelumnya, maka *system* mencari produk yang memenuhi *utility threshold*. Jika ternyata yang memenuhi *utility threshold* juga masih melebih *quantity threshold*, maka *system* akan memberikan pertanyaan berupa kebutuhan fungsional yang lebih spesifik (*Case U4*).

Jika tidak ada produk yang memenuhi *hardconstraint* dan *softconstraint* (*Case U5*), maka pengguna akan diminta untuk merubah *hardconstraint* ataupun *softconstraint* yang sudah dimasukkan untuk kemudian dilakukan pencarian ulang berdasarkan *hardconstraint* atau *softconstraint* yang baru.

4. Pengujian dan Analisis

4.1 Skenario Pengujian

Ada tiga scenario pengujian yang disiapkan untuk mengevaluasi performansi dari *system* yang telash dibuat, yakni pengjuian akurasi oleh *expert*, pengujian efisiensi interaksi, dan pengujian kepuasan pengguna.

4.1.1 Pengujian Akurasi Oleh *Expert*

Pengujian akurasi oleh *expert* ini dilakukan untuk mengevaluasi ranking yang telah diberikan oleh *system*. Pengujian ini dilakukan dengan cara membandingkan hasil perangkingan yang diberikan oleh *system* dengan ranking yang diberikan oleh *expert*. *Expert* yang dipilih untuk melakukan pengujian ini adalah *mobile application developer* yang mengerti seluk beluk mengenai domain

Pada proses pengujian ini, dilakukan masing-masing tiga kali percobaan ke setiap *expert*.

jumlah masing-masing 5, 8, dan 10 produk yang akan diuji rankingnya. Hasil perankingan oleh *expert* ini kemudian dibandingkan dengan hasil perangkingan *system* untuk dilihat errornya dengan menggunakan persamaan 9.

4.1.2 Pengujian Efisiensi Interaksi

Pengujian ini dilakukan untuk mengevaluasi hasil learning pertanyaan yang diajukan kepada pengguna. Pada pengujian ini, tidak semua state interaksi akan diuji, hanya sebagian saja. Pengujian ini dilakukan dengan mengamati banyaknya kemunculan case U3 dalam sekali interaksi. Case U3

ini merupakan kondisi yang mengasumsikan pertanyaan yang lebih dulu ditanyakan tidak sesuai dengan kebutuhan pengguna atau hasil rekomendasi yang diajukan tidak sesuai dengan harapan pengguna karena kemungkinan pertanyaan yang diajukan sebelumnya kurang memuaskan pengguna sehingga perlu dibangkitkan pertanyaan potensial lain yang belum ditanyakan kepada pengguna.

Dengan begitu kita dapat menilai pertanyaan yang diajukan pertama kali itu sudah sesuai dengan kebutuhan pengguna atau tidak. Jika tidak sesuai maka perlu dibangkitkan pertanyaan lain yaitu case U3 yang berakibat lebih lamanya interaksi pengguna dengan *system*.

4.1.3 Pengujian Kepuasan Pengguna

Pengujian kepuasan pengguna ini dilakukan untuk mengevaluasi *system* yang telah dibuat dari segi kenyamanan pengguna. Beberapa faktor yang dievaluasi diantaranya:

- Metode tanya jawab yang digunakan memudahkan pengguna dalam menentukan kebutuhannya.
- *System* dapat membantu dengan baik selama proses rekomendasi.
- *System* membantu pengguna dalam mempercepat pengambilan keputusan.
- Penjelasan (Explanation) untuk setiap produk yang direkomendasikan membantu proses pengambilan keputusan pengguna.
- Kesesuaian pertanyaan dengan kebutuhan pengguna.

Pengujian ini dilakukan kepada pengguna dalam rentang usia 19 s.d 23 tahun dengan jumlah total 50 orang pengguna.

4.2 Hasil Pengujian

4.2.1 Analisis Pengujian Akurasi Oleh *Expert*

Setelah dilakukan pengujian dengan dua orang *expert*, didapatkan nilai *MAE* dari masing-maing jumlah produk dengan menggunakan persamaan 9.

1. Hasil pengujian dengan *expert* pertama, yaitu Toufan D Tambunan.

Tabel 1 Hasil pengujian *expert* pertama

Jumlah Produk	Ranking										*MAE*
	1	2	3	4	5	6	7	8	9	10	
5	2	1	4	3	5						0.8
8	7	8	1	2	6	4	3	5			3.25
10	5	2	8	3	4	1	10	7	9	6	2.4

2. Hasil pengujian dengan *expert* kedua, yaitu Pramuko Aji.

Tabel 2 Hasil pengujian dengan *expert* kedua

Jumlah Produk	Ranking										*MAE*
	1	2	3	4	5	6	7	8	9	10	
5	2	4	1	3	5						1.2
8	1	3	5	7	2	4	6	8			1.5
10	4	2	3	6	1	7	5	9	8	10	1.4

Dari kedua hasil pengujian pada tabel 1-2 didapat rata-rata dari *MAE* untuk setiap jumlah produk adalah seperti yang ditunjukkan pada tabel 3.

Tabel 3 Rata-rata *MAE*

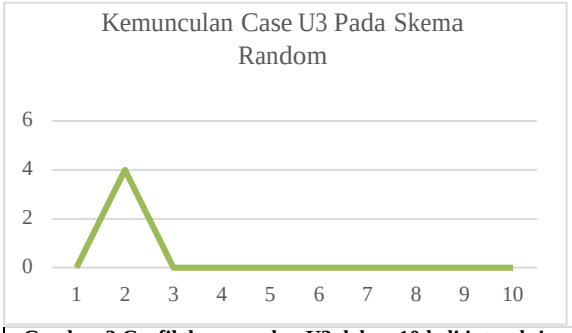
Jumlah Produk	*MAE*
5	1
8	2.375
10	1.9

Pada tabel 3 terlihat bahwa jumlah produk 5 memiliki nilai *MAE* paling kecil, ini dikarenakan jumlahnya yang sedikit sehingga kemungkinan kesalahan saat perankingngan juga kecil karena proses sesleksi yang lebih ketat. Sedangkan untuk jumlah prduk 8 maupun 10, memmpunyai nilai *MAE* cenderung lebih besar, ini menunjukkan bahwa semakin banyak produk, maka kemungkinan kesalahan perangkingan juga semakin besar.

Namun, pada jumlah produk 10 nilai *MAE* bisa lebih kecil dari pada jumlah produk 8 dikarenakan pada jumlah produk 10 kemungkinan produk yang rankingnya terbawah tepat berada pada ranking seharusnya sehingga nilai *MAE* nya lebih kecil dibandingkan dengan jumlah produk 8.

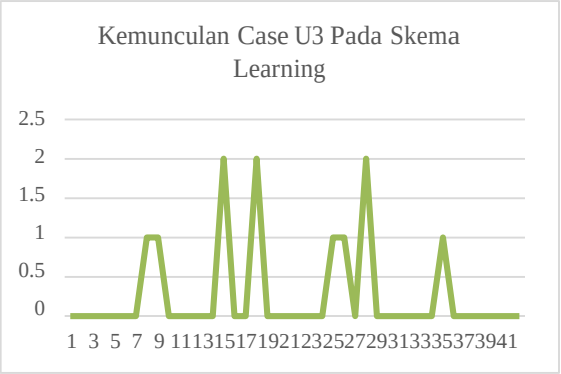
4.2.2 Analisis Pengujian Efisiensi Interaksi

Pada gambar 10 dapat dilihat data kemunculan case U3 dalam 50 kali interaksi selama proses pengujian dilakukan.



Gambar 2 Grafik kemunculan U3 dalam 10 kali interaksi

Pada gambar 11 dapat dilihat bahwa masih menunjukkan bahwa learning yang dilakukan masih belum baik sehingga interaksi yang terjadi juga jadi kurang efisien. Kemungkinan ini terjadi karena probabilitas untuk proses eksplorasi seperti yang dijelasakan pada persamaan 3 masih terlalu besar untuk memunculkan pertanyaan-pertanyaan yang belum optimal masih dilakukan. Namun kemunculan dibanding dengan random. Case U3 pada hali learning paling banyak hanya muncul 2 kali dalam satu kali interaksi, berbeda dengan pada saat random yang kemunculan U3 mencapai 4 kali dalam satu kali interaksi.



Gambar 3 Grafik kemunculan U3 dalam 42 kali interaksi pada skema pembangkitan pertanyaan hasil learning

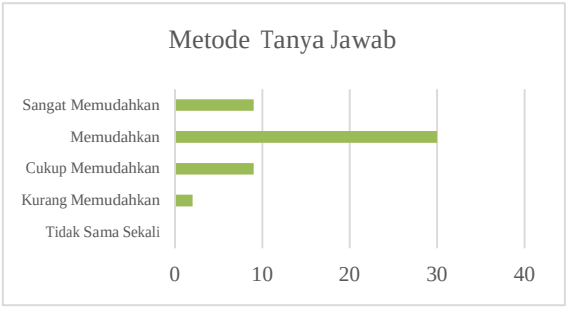
Dalam proses pembelajaran pertanyaan yang optimal dengan menggunakan metode ini memang memerlukan pengujian yang cukup lama atau dengan data uji yang banyak sampai nilai probabilitas eksplorasinya konvergen ke nol. Sehingga yang tersisa nantinya hanya proses eksploitasi saja terhadap pengetahuan yang sudah didapat, yakni pengetahuan tentang pertanyaan-pertanyaan yang benar-benar sudah optimal berdasarkan hasil pembelajaran.

4.2.3 Analisis Pengujian Kepuasan Pengguna

Berdasarkan hasil pengujian kepuasan terhadap pengguna, diperoleh hasil sebagai berikut:

1. Metode tanya jawab yang digunakan

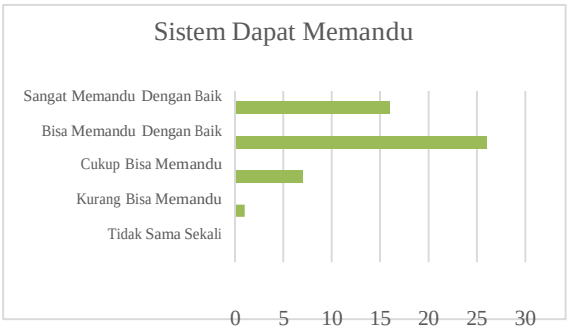
Sebanyak 60% pengguna menyatakan bahwa metode tanya jawab yang digunakan memudahkan, 18% menyatakan sangat memudahkan, 18% cukup dan 4% sisanya menyatakan kurang memudahkan. Ini menunjukkan bahwa secara keseluruhan metode tanya jawab yang digunakan sudah baik. Detailnya dapat dilihat pada gambar 12.



Gambar 4 Grafik kepuasan pengguna terhadap metode tanya jawab yang digunakan

2. Kemampuan *system* dalam memandu pengguna

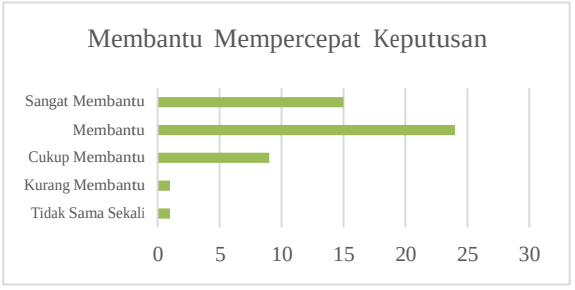
Sebanyak 52% pengguna menyatakan *system* yang dibuat bisa memandu pengguna dengan baik selama proses rekomendasi. Sebanyak 32% lainnya juga menyatakan bahwa *system* sangat bisa memandu mereka dalam mencari produk yang mereka inginkan. Hanya 14% saja yang menyatakan dengan baik oleh *system*. Lebih jelasnya dapat dilihat pada gambir 13.



Gambar 5 Grafik kepuasan pengguna terhadap kemampuan *system* dalam memandu pengguna

3. Kemampuan *system* dalam membantu mempercepat pengambilan keputusan

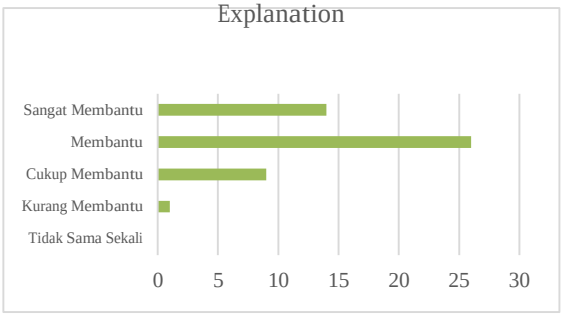
Dari segi waktu, 30% pengguna menyatakan sangat terbantu dalam mempercepat pengambilan keputusan, sehingga waktu mereka dapat digunakan secara efisien. Sebanyak 48% merasa terbantu, 18% merasa cukup terbantu, dan sisanya ada yang merasa kurang terbantu dan ada yang merasa tidak terbantu sama sekali. Selengkapnya dapat dilihat pada gambar 14.



Gambar 6 Grafik kepuasan pengguna terhadap kemampuan *system* dalam membantu mempercepat keputusan

4. *Explanation*

Sebagian besar pengguna merasa terbantu dengan adanya penjelasan (*explanation*) mengenai produk yang direkomendasikan, yakni sebanyak 52%. Sebanyak 28% merasa sangat terbantu, 18% merasa cukup dan hanya 2% saja yang tidak merasa terbantu.



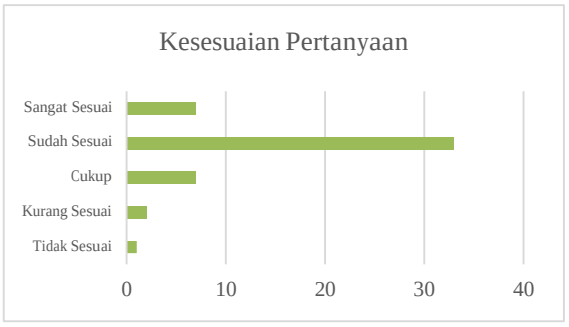
cukup dan 2% sisanya merasa kurang bisa dipandu

Gambar 7 Grafik kepuasan pengguna terhadap *explanation* yang disediakan oleh sistem

Ini menandakan bahwa penjelasan (*explanation*) cukup dibutuhkan oleh pengguna, terutama bagi mereka yang memang belum tau lebih jauh terhadap produk yang direkomendasikan, sehingga memerlukan adanya penjelasan terhadap produk tersebut.

5. Kesesuaian Pertanyaan

Sebagian besar pengguna menyatakan bahwa pertanyaan mengenai kebutuhan fungsional yang diajukan *system* sudah sesuai dengan kebutuhan pengguna pada saat itu. Namun, meski begitu ada juga yang menyatakan bahwa pertanyaan yang diajukan kurang sesuai dengan kebutuhannya
bahkan ada yang menyatakan tidak sesuai sama sekali. Hal ini dikarenakan proses learning yang memang belum baik sehingga masih ada pertanyaan yang seharusnya tidak diajukan kepada pengguna tetap diajukan, sehingga terjadi ketidakcocokan. Lebih detailnya dapat dilihat pada gambar 16, yakni sebanyak 14% merasa sangat sesuai, 66% sudah sesuai, 14% merasa cukup, 4% merasa kurang sesuai, dan 2% sisanya merasa tidak sesuai sama sekali.



Gambar 8 Grafik kepuasan pengguna terhadap kesesuaian pertanyaan tentang kebutuhan fungsional

5. Penutup

5.1 Kesimpulan

Berdasarkan hasil pengujian terhadap lebih kurang 50 orang pengguna, masih belum dapat terlihat efisiensi interaksi yang terjadi. Ini dikarenakan nilai probabilitas *epsilon greedy* untuk proses eksplorasi masih cukup besar atau masih belum mendekati nol. Sehingga *conversational recommender system* yang dibuat masih memberikan pertanyaan-pertanyaan yang belum optimal yang berakibat pada kurang efisiennya interaksi yang terjadi.

Berdasarkan hasil pengujian juga dapat dilihat, akurasi perankingan pada saat rekomendasi produk yang paling baik adalah dengan jumlah produk 5 dibandingkan dengan jumlah produk 8 dan 10. Ini dikarenakan, semakin sedikitnya produk yang ditampilkan, semakin kecil kemungkinan kesalahan dalam melakukan perankingan.

5.2 Saran

Untuk penelitian kedepannya, proses learning pertanyaan kebutuhan fungsional bisa dicoba dikombinasikan dengan relasi *semantik* yang terdapat pada produk, sehingga pertanyaan kebutuhan fungsional yang diajukan hanya yang berelasi dengan produk yang yang diinginkan saja.

Daftar Pustaka

[1] Barakbah, A. R. (n.d.). Reinforcement Learning Paradigma baru dalam Machine Learning. Soft Computation Research Group, EEPIS-ITS.

[2] Chen, L., & Pu, P. (2012). *Preference-based Organization Interfaces: Aiding User Critiques in Recommender Systems* . Lausanne, Switzerland: EPFL.

[3] Gosavi, A. (2013). *A Tutorial for Reinforcement Learning.* Rolla: Department of Engineering Management and Systems Engineering Missouri University of Science and Technology.

[4] HERLOCKER, J. L., KONSTAN, J. A., TERVEEN, L. G., & RIEDL, J. T. (2004). Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems* (pp. 5-53). New York, USA: ACM, Inc.

[5] Jannach, D., Zanker, M., Relfenig, A., & Friedrich, G. (2012). *Recommender System - An Introduction.* New York: Cambridge University Press.

[6] Mirzadeh, N., Ricci, F., & Bansal, M. (n.d.). Feature Selection Methods for Conversational Recommender System.

[7] Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (2011). *Recommender System Handbook.* New York, USA: Springer Science+Business Media, LLC.

[8] Sutton, R. S., & Barto, A. G. (2005). *Reinforcement Learning: An Introduction.* Cambridge, Massachusetts, London, England: The MIT Press.

[9] Tintatev, N. (2007). Explanations of Recommendations . Minneapolis, Minnesota, USA: ACM.

[10] Tintatev, N., & Masthoff , J. (2007). Effective Explanations of Recommendations: User-Centered Design. Minneapolis, Minnesota, USA: ACM.

[11] Widyantoro, D. H., & Baizal, Z. (2014). A Framework of Conversational Recommender System based on User Functional Requirement. *International Conference on Information and Communication Technology* (pp. 160-165). IEEE Conference Publications.